Document Title	Specification of ECU Configuration Parameters
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Version	1.2.0
Document Status	Draft
Part of Release	2.1
Revision	0020

Document Change History			
Date	**Version**	**Changed by**	**Description**
31.05.2010	1.2.0	AUTOSAR Administration	• [Mcu] Allow multiplicity of sub-container Mcu Clock Setting • [Com] metamodel missing COM_IPDU_SIGNAL_ PROCESSING parameter • [Com] How can COM_RX_DATA_INVALID work with COM_SIGNAL_GROUP? • [Can] Passing random values to the application • [Dcm] correction for DcmDslROETransType
31.01.2006	1.1.0	AUTOSAR Administration	• Maintenance of ECUC Parameters • Added support for multiple configuration sets • General rework of the Rte configuration • General rework of the NM configuration • “Advice for users” revised • “Release Notes” added • “Revision Information” added
09.08.2006	1.0.0	AUTOSAR Administration	Initial release

Release Notes

Compatibility considerations with respect to current release

The configuration parameters captured in this document may not be in sync with the BSW SWS documents. In case of inconsistencies the BSW SWS has the priority. The release notes of the BSW SWS document's also apply to this specification.

Errata and known deficiencies

There has been no full review cycle on the configuration parameters provided in this document. The document is released in DRAFT status.

Known and potential problems resulting from known deficiencies

Inconsistency in the definition of the configuration parameters.

Changes planned for next release

The following changes are planned for the next Release 3.0/4.0:

- Update configuration parameters with BSW SWS input.
- Replace BSW SWS chapter 10 with the information from this document.
- Maintain configuration parameters in this document (single source).

Disclaimer

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

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

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

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

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

Advice for users:

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

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

Table of Contents

1 Introduction 12

1.1 Formalization aspects . . . 12
1.1.1 ECU Configuration Parameter Definition table . . . 13
1.1.2 Common Parameter Definitions to all Modules . . . 14
1.2 Overview . . . 17
1.3 Requirements Traceability . . . 20

2 Application Software Components 24

3 RTE 25

3.1 RTE Generation Parameters . . . 27
3.2 Handling of Software Component instances . . . 29
3.2.1 Selection of SW-Component Implementation . . . 30
3.2.2 Runnable Entity to task mapping . . . 31
3.2.3 Exclusive Area implementation . . . 34
3.2.4 NVRam Allocation . . . 35
3.3 CalprmComponent instance configuration . . . 37
3.4 Communication infrastructure . . . 38

4 Communications 39

4.1 COM-Stack Configuration Patterns . . . 40
4.1.1 Handle IDs . . . 40
4.1.1.1 Handle ID concept . . . 41
4.1.1.2 Definition of Handle IDs . . . 42
4.1.1.3 Definition of PDUs . . . 43
4.1.1.4 Agreement on Handle IDs . . . 44
4.1.1.5 Handle IDs with symbolic names . . . 46
4.2 Com . . . 47
4.2.1 ComGeneral . . . 49
4.2.2 ComSignal . . . 50
4.2.2.1 ComFilter . . . 56
4.2.2.2 ComNotification . . . 59
4.2.2.3 ComRxDataTimeoutSub . . . 61
4.2.2.4 ComTxDataInvalid and ComRxDataInvalid . . . 63
4.2.2.5 ComSignalRouteDest and ComSignalInternalDest . . . 65
4.2.3 ComSignalGroup . . . 67
4.2.4 ComNetworkSignal . . . 70
4.2.5 ComIPdu . . . 75
4.2.5.1 ComTransmissionMode . . . 80
4.2.5.2 Interaction Com and PduR . . . 83
4.2.6 ComIPduGroup . . . 85
4.3 PDU Router . . . 87
4.3.1 PduRGeneral . . . 88
4.3.2 PduRRoutingTable . . . 96
4.3.3 PduRTxBufferTable . . . 100

4.3.4 PduRTpBufferTable . . . 102
4.3.5 Configuration examples for the Pdu Router . . . 103
4.3.5.1 Tx from Com to CanIf . . . 103
4.3.5.2 Rx from CanIf to Com . . . 105
4.3.5.3 Gateway from CanIf to FrIf . . . 106
4.4 IPdu Multiplexer . . . 107
4.4.1 IPduMGeneral . . . 108
4.4.2 IPduMTxPathway . . . 110
4.4.2.1 IPduMTxRequest . . . 111
4.4.2.2 IPduMTxConfirmation . . . 116
4.4.3 IPduMRxPathway . . . 119
4.4.4 IPduMBitField . . . 122
4.4.5 IPduMCopyBitField . . . 123
4.5 Can . . . 124
4.5.1 CAN Interface . . . 124
4.5.1.1 CanIfPrivateConfig . . . 127
4.5.1.2 CanIfPublicConfig . . . 129
4.5.1.3 CanIfControllerConfig . . . 133
4.5.1.4 CanIfDispatchConfig . . . 135
4.5.1.5 CanIfRxPduConfig . . . 136
4.5.1.6 CanIfTxPduConfig . . . 140
4.5.1.7 CanIfNetworkConfig . . . 144
4.5.1.8 CanIfDriverConfig . . . 146
4.5.2 CAN Driver . . . 158
4.5.2.1 CanGeneral . . . 159
4.5.2.2 CanController . . . 161
4.5.3 CAN Transport Layer . . . 168
4.5.3.1 CanTpGeneral . . . 169
4.5.3.2 CanTpRxNSdu . . . 170
4.5.3.3 CanTpTxNSdu . . . 176
4.5.3.4 CanTpNTa and CanTpNSa . . . 182
4.5.3.5 CanTpReferences . . . 183
4.5.4 CAN Transceiver Driver . . . 184
4.5.4.1 CanTrcvGeneral . . . 185
4.5.4.2 CanTrcvChannel . . . 187
4.5.4.3 CanTrcvAccess . . . 189
4.6 FlexRay . . . 192
4.6.1 FlexRay Interface . . . 192
4.6.1.1 FlexRay Cluster . . . 195
4.6.1.2 Communication Controllers . . . 205
4.6.1.3 Job list and Jobs . . . 212
4.6.1.4 Frame structure and PDUs . . . 215
4.6.2 FlexRay Driver . . . 220
4.6.2.1 FlexRay Controller Configuration . . . 221
4.6.2.2 FlexRay Driver Timer Configuration . . . 231
4.6.3 FlexRay Transport Layer . . . 233
4.6.3.1 FlexRay TP General Configuration . . . 234

4.6.3.2 FlexRay TP Channel Configuration . . . 236
4.6.3.3 FlexRay TP PDU Configuration . . . 243
4.6.3.4 FlexRay TP Connection Configuration . . . 245
4.6.4 FlexRay Transceiver Driver . . . 249
4.6.4.1 FrTrcvGeneral . . . 250
4.6.4.2 FrTrcvNode . . . 252
4.7 LIN . . . 255
4.7.1 LIN Interface . . . 256
4.7.1.1 LIN Interface Channel Configuration . . . 258
4.7.1.2 LIN Interface Frame Configuration . . . 261
4.7.1.3 LIN Interface Schedule Table Configuration . . . 265
4.7.2 LIN Driver . . . 267
4.7.3 LIN Transport Layer . . . 270
4.8 Network Management . . . 274
4.8.1 Network Management Interface . . . 275
4.8.2 Can Generic Network Management . . . 279
4.8.2.1 Can Generic Network Management global configuration 280
4.8.2.2 Can Generic Network Management channel configuration 285
4.8.3 Can Network Management . . . 290
4.8.3.1 Can Network Management feature configuration . . . 291
4.8.3.2 Can Network Management global configuration . . . 294
4.8.3.3 Can Network Management channel configuration . . . 297
4.8.3.4 Can Network Management references . . . 302
4.8.4 FlexRay Network Management . . . 303
4.8.4.1 Flexray Network Management global configuration . . . 304
4.8.4.2 Flexray Network Management channel configuration . . 309
4.8.4.3 Flexray Network Management references . . . 319

5 System Services 320

5.1 Operating System . . . 321
5.1.1 OS Configuration Object . . . 323
5.1.1.1 Hooks of the OsOS . . . 325
5.1.2 OS Applications . . . 327
5.1.2.1 Hooks of the OsApplication . . . 332
5.1.2.2 AppMode of the OsApplication . . . 333
5.1.3 Schedule Tables . . . 334
5.1.3.1 Schedule Table Actions . . . 336
5.1.3.2 Schedule Table Synchronization . . . 337
5.1.4 Tasks . . . 339
5.1.5 Interrupt Service Routines . . . 345
5.1.6 Alarms . . . 349
5.1.7 Messages . . . 353
5.1.7.1 OsMessage properties . . . 354
5.1.7.2 OsMessage notifications . . . 356
5.1.8 Counters . . . 359
5.1.9 Events . . . 361
5.1.10 Resources . . . 361

5.1.11 Communications . . . 363
5.2 Watchdog . . . 365
5.2.1 Watchdog Manager . . . 365
5.2.1.1 General Watchdog Manager configuration parameters . . . 366
5.2.1.2 Supervision Entities . . . 369
5.2.1.3 Triggers . . . 373
5.2.2 Watchdog Interface . . . 375
5.2.3 Watchdog Driver . . . 377
5.2.3.1 Hardware-specific mode configuration . . . 379
5.2.3.2 Module-specific Published Information . . . 380
5.3 Diagnostics . . . 382
5.3.1 FIM - Function Inhibition Manager . . . 383
5.3.1.1 FimEventSummaryConfiguration . . . 388
5.3.1.2 FimInhibitionConfiguration . . . 390
5.3.2 DEM - Diagnostics Event Manager . . . 393
5.3.2.1 DemConfiguration . . . 394
5.3.3 DCM - Diagnostic Communication Manager . . . 415
5.3.3.1 DcmConfiguration . . . 415
5.3.3.2 DcmGeneralConfiguration . . . 418
5.3.3.3 DcmDslConfiguration . . . 420
5.3.3.4 DcmDsdConfiguration . . . 440
5.3.3.5 DcmDspConfiguration . . . 444
5.4 ECU State Manager . . . 454
5.5 Communication Manager . . . 466
5.5.1 ComMGeneral . . . 467
5.5.2 ComMChannel . . . 471
5.5.3 ComMNetworkManagement . . . 476
5.5.4 ComMUser . . . 477
5.6 CRC Routines . . . 479
5.7 Crypto Routines . . . 482
5.8 Interpolation Routines . . . 483
5.9 Development Error Tracer . . . 484
5.10 External Time Service . . . 485
6 I/O Services 486
6.1 I/O Hardware Abstraction . . . 487
6.1.1 Generic configuration . . . 489
6.1.2 Signal configuration . . . 490
6.1.2.1 Signal groups . . . 491
6.1.2.2 Discrete signals . . . 493
6.1.2.3 Analog signals . . . 497
6.1.2.4 Pulse Width signals . . . 501
6.2 Port driver . . . 508
6.3 DIO Driver . . . 513
6.4 ADC Driver . . . 519
6.5 ICU Driver . . . 530
6.6 PWM Driver . . . 542

6.7 SPI 548
6.7.1 Interfaces to other driver 554
6.7.2 Description of the parameter 554

7 Memory 562

7.1 AUTOSAR Architecture 562
7.2 Memory Services 564
7.2.1 NVRAM Manager 564
7.3 Memory Hardware Abstraction 576
7.3.1 Memory Abstraction Interface 576
7.3.2 FLASH EEPROM Emulation Module 578
7.3.3 EEPROM Abstraction Module 584
7.4 Memory Drivers 590
7.4.1 Flash Driver 590
7.4.2 EEPROM Driver 602

8 Miscellaneous 611

8.1 BSW Scheduler Module 611
8.2 MCU Driver 616
8.3 GPT Driver 623
8.4 RAM Test 630

A Parameter Change History 640

A.1 Release 1.1.0 640

Bibliography

[1] General Requirements on Basic Software Modules
AUTOSAR_SRS_General.pdf

[2] Glossary
AUTOSAR_Glossary.pdf

[3] Requirements on Basic Software: Layered Software Architecture
AUTOSAR_LayeredSoftwareArchitecture.pdf

[4] Requirements on Basic Software Module Description Template
AUTOSAR_RS_BSW_ModuleDescription.pdf

[5] Requirements on ECU Configuration
AUTOSAR_RS_ECU_Configuration.pdf

[6] Software Component Template
AUTOSAR_SoftwareComponentTemplate.pdf

[7] Specification of ECU Configuration
AUTOSAR_ECU_Configuration.pdf

[8] System Template
AUTOSAR_SystemTemplate.pdf

[9] Specification of Module DEM
AUTOSAR_SWS_DEM.pdf

[10] Template UML Profile and Modeling Guide
AUTOSAR_TemplateModelingGuide.pdf

[11] API Specification of Module Development Error Tracer
AUTOSAR_SWS_DET.pdf

[12] Specification of ADC Driver
AUTOSAR_SWS_AdcDriver.pdf

[13] Specification of CAN Driver
AUTOSAR_SWS_CAN_Driver.pdf

[14] Specification of CAN Interface
AUTOSAR_SWS_CAN_Interface.pdf

[15] Specification of Communication
AUTOSAR_SWS_COM.pdf

[16] Specification of DIO Driver
AUTOSAR_SWS_DioDriver.pdf

[17] Specification of ECU State Manager
AUTOSAR_SWS_ECU_StateManager.pdf

[18] Specification of FlexRay Driver
AUTOSAR_SWS_FlexRayDriver.pdf

[19] Specification of GPT Driver
AUTOSAR_SRS_GPT_Driver.pdf

[20] Specification of I-PDU Multiplexer
AUTOSAR_SWS_IPDUM.pdf

[21] Specification of ICU Driver
AUTOSAR_SWS_IcuDriver.pdf

[22] Specification of IO Hardware Abstraction
AUTOSAR_SRS_IOHW_Abstraction.pdf

[23] Specification of LIN Driver
AUTOSAR_SWS_LIN_Driver.pdf

[24] Specification of LIN Interface
AUTOSAR_SWS_LIN_Interface.pdf

[25] Specification of MCU Driver
AUTOSAR_SWS_Mcu_Driver.pdf

[26] Specification of Module DCM
AUTOSAR_SWS_DCM.pdf

[27] Specification of Module FIM
AUTOSAR_SWS_FIM.pdf

[28] Specification of Module FlexRay Interface
AUTOSAR_SWS_FlexRayInterface.doc

[29] Specification of Module FlexRay Transport Layer
AUTOSAR_SWS_FlexRay_TP.pdf

[30] Specification of Module Operating System
AUTOSAR_SWS_OS.pdf

[31] Specification of Module Watchdog Driver
AUTOSAR_SWS_WatchdogDriver.pdf

[32] Specification of Module Watchdog Interface
AUTOSAR_SWS_WatchdogInterface.pdf

[33] Specification of Module Watchdog Manager
AUTOSAR_SWS_WatchdogManager.pdf

[34] Specification of PDU Router
AUTOSAR_SWS_PDU_Router.pdf

[35] Specification of PORT Driver
AUTOSAR_SWS_PortDriver.pdf

[36] Specification of PWM Driver
AUTOSAR_SWS_PwmDriver.pdf

[37] Specification of RAM Test
AUTOSAR_SWS_RAMTest.pdf

[38] Specification of Serial Peripheral Interface (SPI) Handler/Driver
AUTOSAR_SWS_SPI_HandlerDriver.pdf

[39] System Generation OIL: OSEK Implementation Language
oil25.pdf

1 Introduction

This document describes the standardized configuration parameters of the AUTOSAR software modules. The general methodology of the ECU Configuration approach can be found in the ECU Configuration Specification document [7].

1.1 Formalization aspects

The goal of this document is to provide an exchange format for the configuration of an AUTOSAR ECU. Therefore there is not necessarily a simple translation of the configuration parameter's values in the ECU Configuration Description (XML file) into the module's configuration (header file). It is the duty of the module's generation tool to transform the configuration information from the XML file into a header file.

The configuration parameters described in this document are based on the BSW specification documents and have been formalized into an UML model. This UML model is used to partly generate this document and to generate the ECU Configuration Parameter Definition XML file, which is delivered together with this document.

Several deviations from the BSW SWS documents have been applied for the ECU Configuration Parameter Definition:

- *Modified parameter names*: Due to the limitations imposed by the AUTOSAR XML format (32 character limit starting with a letter, etc.) the names of parameters and containers have been redefined. Also a different naming schema has been applied to explicitly make a difference between the configuration parameters expected in the software's implementation (BSW SWS Chapter 10) and the configuration parameters used in the ECU Configuration Description XML file. The original names from the SWS are provided in this document as well.
- *Added references*: To allow a better interaction of the configuration descriptions of several modules references between the configuration have been introduced (this is just a starting point and there are a lot of possible references not yet introduced).
- *Changed parameter types*:
  - Boolean: Some parameters have been defined as `enumeration` or `#define` where the actual information stored is of type boolean. In those cases they have been modeled as boolean in this document.
  - Float: Some parameters store a time value as `integer` where it is stated that this is a time in e.g. micro-seconds. If the time specified is an absolute time it has been formalized as a `float` in seconds. If the time is a factor of some given time-base the `integer` is preserved.
- *Added parameter multiplicities*: In the original tables from the BSW SWS there is no possibility to specify the optionality and multiplicity of parameters. These have been added in this document.

The long term goal is to replace the configuration chapter of the BSW SWS documents with the sections from this document. For the collection of all configuration parameters

and their relationships the BSW SWS documents are needed in their current form.

1.1.1 ECU Configuration Parameter Definition table

The configuration parameters are structured into containers which can hold parameters, references and other containers. Beside the graphical visualization in UML diagrams tables are used to specify the structure of the parameters.

In the following table one container is specified which holds two parameters and also two additional containers.

SWS Item	[SWS requirement IDs]		
Container Name	ContainerName {original name from SWS}		
Description	Container description.		
Configuration Parameters			

Name last modified in:	ParameterName {original name from SWS} [SWS requiremenets IDs]		
Description	Parameter description.		
Multiplicity	Parameter multiplicity		
Type	Parameter type		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Name last modified in:	EnumerationTest {ENUMERATION_TEST} [SWS0815]		
Description	description.		
Multiplicity	0..1 (optional)		
Type	EnumerationParamDef		
Range	ReceiveUnqueuedExternal		
	ReceiveUnqueuedInternal		
	SendStaticExternal		
	SendStaticInternal		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Included Containers

Container Name	Multiplicity	Scope / Depedency
Container_1	0..1	Optional sub-container.
Container_2	0..*	Optional sub-container which can be present several times.

For a detailed description of the elements in the tables please refer to the ECU Configuration Specification document [7].

1.1.2 Common Parameter Definitions to all Modules

Besides the module-specific configuration parameters there exists a configuration container that defines the common published information that every module needs to provide: Note that these parameters do not have any configuration class setting, since they are published information.

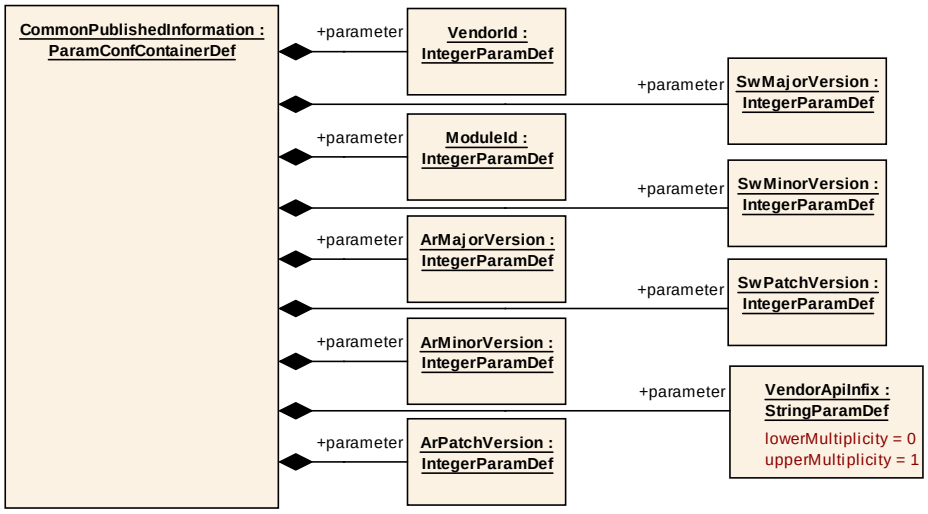


Figure 1.1: CommonPublishedInformation

CommonPublishedInformation

SWS Item			
Container Name	CommonPublishedInformation		
Description	Common container, aggregated by all modules. It contains published information about vendor and versions.		
Configuration Parameters			

Name last modified in: V1.0.0	ArMajorVersion		
Description	Major version number of AUTOSAR specification on which the appropriate implementation is based on.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	ArMinorVersion		
Description	Minor version number of AUTOSAR specification on which the appropriate implementation is based on.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	ArPatchVersion		
Description	Patch level version number of AUTOSAR specification on which the appropriate implementation is based on.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	ModuleId		
Description	Module ID of this module from Module List		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	SwMajorVersion		
Description	Major version number of the vendor specific implementation of the module. The numbering is vendor specific.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	SwMinorVersion		
Description	Minor version number of the vendor specific implementation of the module. The numbering is vendor specific.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	SwPatchVersion		
Description	Patch level version number of the vendor specific implementation of the module. The numbering is vendor specific.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	VendorApiInfix		
Description	In driver modules which can be instantiated several times on a single ECU, BSW00347 requires that the name of APIs is extended by the VendorId and a vendor specific name. This parameter is used to specify the vendor specific name. In total, the implementation specific name is generated as follows: <ModuleName>_>VendorId>_<VendorApiInfix><Api name from SWS>. E.g. assuming that the VendorId of the implementor is 123 and the implementer chose a VendorApiInfix of ”v11r456” a api name Can_Write defined in the SWS will translate to Can_123_v11r456Write. This parameter is mandatory for all modules with upper multiplicity > 1. It shall not be used for modules with upper multiplicity =1.		
Multiplicity	0..1		
Type	StringParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	VendorId		
Description	Vendor ID of the dedicated implementation of this module according to the AUTOSAR vendor list		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			
No Included Containers			

Elements `VendorId` and `VendorApiInfix` are used for driver modules to generate implementation specific API names, see detailed explanation for parameter `VendorApiInfix`.

This container will not be depicted in any further module later in that document as it always contains the same parameter definitions.

1.2 Overview

The software architecture of an AUTOSAR ECU has been divided into several parts to allow independent modules with clean definitions of the interfaces between the different modules. This architecture is depicted in figure 1.2.

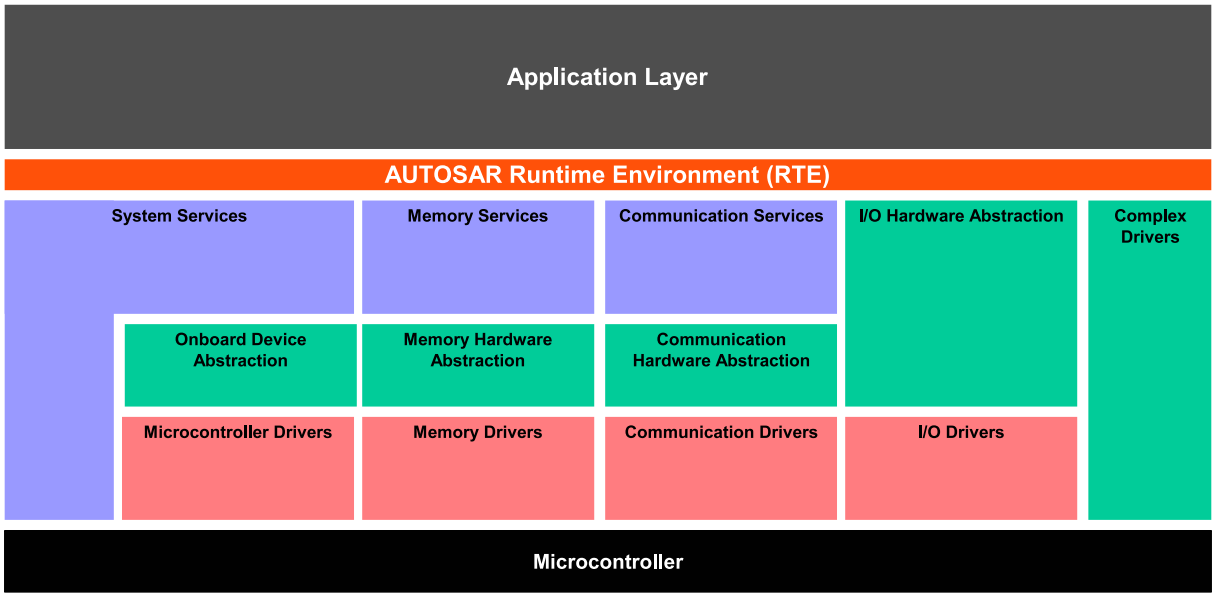


Figure 1.2: ECU Architecture Overview [3]

The application SW-Components are located at the top and can gain access to the rest of the ECU and also to other ECUs only through the RTE.

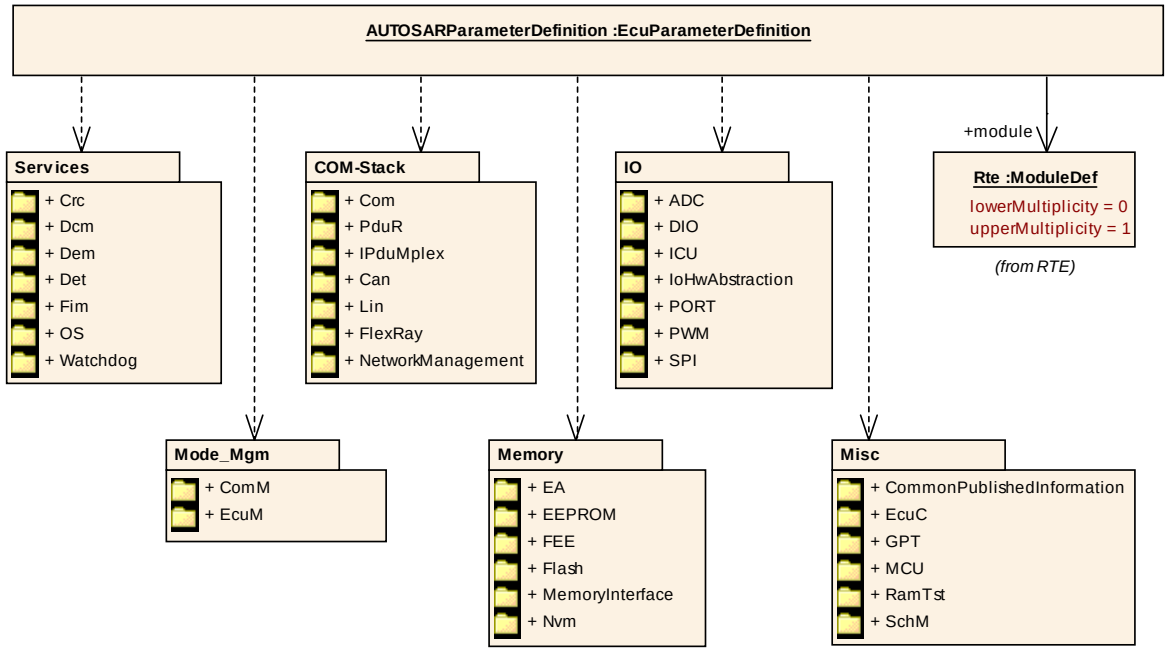


Figure 1.3: AUTOSAR Parameter Definition Overview

The RTE provides the encapsulation of communication and basic services to the Application SW-Components, so it is possible to map the Application SW-Components between different ECUs.

The Basic Software Modules are located below the RTE. The Basic Software itself is divided into the subgroups: System Services, Memory, Communication and IO HW-Abstraction. The Complex Drivers are also located below the RTE.

Among other, the Operating System (OS), the Watchdog manager and the Diagnostic services are located in the System Services subgroup.

The Memory subgroup contains modules to provide access to the non-volatile memories, namely Flash and EEPROM.

In the Communication subgroup the whole AUTOSAR communication stack (COM-Stack) is specified including the COM, Network Management and the communication drivers.

The top-level structure of the `AUTOSARParameterDefinition` is shown in figure 1.3.

The container `AUTOSARParameterDefinition` is the top-level element of the AUTOSAR ECU Configuration Parameter Definition structure. Inside this container references to the diverse configuration container definitions for the different SW modules are defined.

ECU Conf. Name	AUTOSARParameterDefinition	
ECU Conf. Description	Top level container for the definition of AUTOSAR configuration parameters. All of the parameter definitions for the different modules are contained in this container.	
Included Modules		
Module Name	**Multiplicity**	**Scope / Dependency**
Adc	0..1	Configuration of the Adc (Analog Digital Conversion) module.
CNm	0..1	The Can Generic Nm Module.
Can	1	This container holds the configuration of a single CAN Driver.
CanIf	0..1	This container includes all necessary configuration sub-containers according the CAN Interface configuration structure.
CanNm	0..1	The Can Nm module
CanTp	0..1	Configuration of the CanTp (CAN Transport Protocol) module.
CanTrcv	0..*	Configuration of the CanTrcv (CAN Transceiver driver) module.
Com	0..1	Configuration of the Com module.
ComM	0..1	Configuration of the ComM (Communications Manager) module.
Crc	0..1	Configuration of the Crc (Crc routines) module.
Dcm	0..1	Configuration of the Dcm (Diagnostic Communications Manager) module.
Dem	0..1	Configuration of the Dem (Diagnostic Event Manager) module.

Module Name	Multiplicity	Scope / Dependency
Det	0..1	Configuration of the Det (Development Error Tracer) module. There are NO standardized configuration parameters defined. All the configuration of the Det is done via vendor-specific configuration parameters which need to be defined inside this ModuleDef.
Dio	0..1	Configuration of the Dio (Digital IO) module.
Ea	0..1	Configuration of the Ea (EEPROM Abstraction) module. The module shall abstract from the device specific addressing scheme and segmentation and provide the upper layers with a virtual addressing scheme and segmentation as well as a ”virtually” unlimited number of erase cycles.
EcuC	0..1	Virtual module to collect ECU Configuration specific / global configuration information.
EcuM	0..1	Configuration of the EcuM (ECU State Manager) module.
Eep	0..*	Configuration of the Eep (internal or external EEPROM driver) module. Its multiplicity describes the number of EEPROM drivers present, so there will be one container for each EEPROM driver in the ECUC template. When no EEPROM driver is present then the multiplicity is 0.
Fee	0..1	Configuration of the Fee (Flash EEPROM Emulation) module.
Fim	0..1	Configuration of the Fim (Function Inhibition Manager) module.
Fls	0..*	Configuration of the Fls (internal or external flash driver) module. Its multiplicity describes the number of flash drivers present, so there will be one container for each flash driver in the ECUC template. When no flash driver is present then the multiplicity is 0.
Fr	0..*	Configuration of the Fr (FlexRay driver) module.
FrIf	0..1	Configuration of the FrIf (FlexRay Interface) module.
FrNm	0..1	The Flexray Nm module
FrTp	0..1	Configuration of the FrTp (FlexRay Transport Protocol) module.
FrTrcv	0..*	Configuration of the FrTrcv (FlexRay Transceiver driver) module.
Gpt	0..1	Configuration of the Gpt (General Purpose Timer) module.
IPduMplex	0..1	Configuration of the IPduMplex (IPdu Multiplexer) module.
Icu	0..*	Configuration of the Icu (Input Capture Unit) module.
IoHwAbstraction	0..1	Configuration of IO HW Abstraction.
Lin	0..*	Configuration of the Lin (LIN driver) module.
LinIf	0..1	Configuration of the LinIf (LIN Interface) module.
LinTp	0..1	Singleton descriptor for the LIN Transport Protocol.
Mcu	0..1	Configuration of the Mcu (Microcontroler Unit) module.
MemIf	0..1	Configuration of the MemIf (Memory Abstraction Interface) module.
Nm	0..1	The Nm Interface module

Module Name	Multiplicity	Scope / Dependency
NvM	0..1	Configuration of the NvM (NvRam Manager) module.
Os	0..1	Configuration of the Os (Operating System) module.
PduR	0..1	Configuration of the PduR (PDU Router) module.
Port	0..1	Configuration of the Port module.
Pwm	0..*	Configuration of Pwm (Pulse Width Modulation) module.
RamTst	0..1	Configuration of the RamTst module.
Rte	0..1	Configuration of the Rte (Runtime Environment) module.
SchM	0..1	Configuration of the SchM (BSW Scheduler) module.
Spi	0..1	Configuration of the Spi (Serial Peripheral Interface) module.
Wdg	0..*	Configuration of the Wdg (Watchdog driver) module. This should be the extern watchdog driver. For internal watchdogs, no port pins / DIO is used.
WdgIf	0..1	Configuration of the WdgIf (Watchdog Interface) module.
WdgM	0..1	Configuration of the WdgM (Watchdog Manager) module.

In the next sections the parameter definitions of each of these modules will be described according to the rules defined in the ECU Configuration Specification document [7].

1.3 Requirements Traceability

Following table references the requirements specified in AUTOSAR ECU Configuration Requirements [5] and in General Requirements on Basic Software Modules [1] and links to the ”ecuc_sws” fulfillments of these.

Requirement	Description	Satisfied by
[BSW00344]	Reference to link-time configuration	N/A; BSW implementation issue
[BSW00345]	Pre-compile-time configuration	N/A; BSW implementation issue
[BSW00347]	Naming separation of different instances of BSW drivers	Ch. 1.1.2
[BSW00380]	Separate C-File for configuration parameters	N/A; BSW implementation issue
[BSW00381]	Separate configuration header file for pre-compile time parameters	N/A; BSW implementation issue
[BSW00382]	Not-used configuration elements need to be listed	N/A; requirement on the BSW Module Description [4]
[BSW00383]	List dependencies of configuration files	N/A; requirement on the BSW Module Description [4]
[BSW00384]	List dependencies to other modules	N/A; requirement on the BSW Module Description [4]
[BSW00385]	List possible error notificatons	N/A; requirement on the BSW Module Description [4]
[BSW00386]	Configuration for detecting an error	requirement on BSW Module Description [4]
[BSW00387]	Specify the configuration class of callback function	N/A; requirement on ECU Configuration Specification [7]
[BSW00388]	Introduce containers	N/A; requirement on ECU Configuration Specification [7]

Requirement	Description	Satisfied by
[BSW00389]	Containers shall have names	N/A; requirement on ECU Configuration Specification [7]
[BSW00390]	Parameter content shall be unique within the module	N/A; requirement on the BSW SWS
[BSW00391]	Parameter shall have unique names	N/A; requirement on ECU Configuration Specification [7]
[BSW00392]	Parameters shall have a type	N/A; requirement on ECU Configuration Specification [7]
[BSW00393]	Parameters shall have a range	N/A; requirement on ECU Configuration Specification [7]
[BSW00394]	Specify the scope of the parameters	N/A; requirement on the BSW SWS
[BSW00395]	List the required parameters (per parameter)	N/A; requirement on ECU Configuration Specification [7]
[BSW00396]	Configuration classes	N/A; requirement on ECU Configuration Specification [7]
[BSW00397]	Pre-compile-time parameters	N/A; requirement on ECU Configuration Specification [7]
[BSW00398]	Link-time parameters	N/A; requirement on ECU Configuration Specification [7]
[BSW00399]	Loadable Post-build time parameters	N/A; requirement on ECU Configuration Specification [7]
[BSW00400]	Selectable Post-build time parameters	N/A; requirement on ECU Configuration Specification [7]
[BSW00401]	Documentation of multiple instances of configuration parameters	N/A; requirement on the BSW SWS
[BSW00402]	Published information	N/A; requirement on the BSW Module Description [4]
[BSW00404]	Reference to post build time configuration	N/A; BSW implementation issue
[BSW00405]	Reference to multiple configuration sets	N/A; BSW implementation issue
[BSW00408]	Configuration parameter naming convention	
[BSW00410]	Compiler switches shall have defined values	N/A; BSW implementation issue
[BSW00411]	Get version info keyword	N/A; BSW implementation issue
[BSW00412]	Separate H-File for configuration parameters	N/A; BSW implementation issue
[BSW00413]	Accessing instances of BSW modules	
[BSW00414]	Parameter of init function	N/A; BSW implementation issue
[BSW00415]	User dependent include files	N/A; BSW implementation issue
[BSW00416]	Sequence of Initialization	requirement on the BSW SWS
[BSW00417]	Reporting of Error Events by Non-Basic Software	N/A; non-BSW implementation issue
[BSW00419]	Separate C-Files for pre-compile time configuration parameters	N/A; BSW implementation issue
[BSW00420]	Production relevant error event rate detection	N/A; requirement on the BSW SWS
[BSW00421]	Reporting of production relevant error events	N/A; requirement on the BSW SWS and BSW Module Description [4]
[BSW00422]	Debouncing of production relevant error status	N/A; requirement on the BSW SWS
[BSW00423]	Usage of SW-C template to describe BSW modules with AUTOSAR Interfaces	N/A; requirement on BSW Module Description [4]
[BSW00424]	BSW main processing function task allocation	N/A; requirement on BSW scheduler module

Requirement	Description	Satisfied by
[BSW00425]	Trigger conditions for schedulable objects	N/A; requirement on BSW Module Description [4]
[BSW00426]	Exclusive areas in BSW modules	N/A; requirement on BSW Module Description [4] and BSW implementation issue
[BSW00427]	ISR description for BSW modules	N/A; requirement on BSW Module Description [4]
[BSW00428]	Execution order dependencies of main processing functions	N/A; requirement on BSW Module Description [4]
[BSW00429]	Restricted BSW OS functionality access	N/A; BSW implementation issue
[BSW00431]	The BSW Scheduler module implements task bodies	N/A; requirement on BSW scheduler module
[BSW00432]	Modules should have separate main processing functions for read/receive and write/transmit data path	N/A; BSW implementation issue
[BSW00433]	Calling of main processing functions	N/A; BSW implementation issue
[BSW00434]	The Schedule Module shall provide an API for exclusive areas	N/A; BSW implementation issue
[BSW159]	Tool-based configuration	N/A; requirement on ECU Configuration Specification [7]
[BSW167]	Static configuration checking	N/A; requirement on ECU Configuration Specification [7]
[BSW170]	Data for reconfiguration of AUTOSAR SW-Components	N/A; requirement on the Software-Component Template [6]
[BSW171]	Configurability of optional functionality	N/A; requirement on ECU Configuration Specification [7]
[ECUC0002]	Support of vendor-specific ECU Configuration Parameters	N/A; requirement on ECU Configuration Specification [7]
[ECUC0008]	Definition of post-build time changeable configuration of BSW	N/A; requirement on ECU Configuration Specification [7]
[ECUC0012]	One description mechanism for different configuration classes	N/A; requirement on ECU Configuration Specification [7]
[ECUC0015]	Configuration of multiple instances of BSW modules	N/A; requirement on ECU Configuration Specification [7]
[ECUC0016]	Execution order of runnable entities	see chapter 3
[ECUC0018]	Extension handling	N/A; requirement on ECU Configuration Specification [7]
[ECUC0021]	Select Application SW Component and BSW module implementation	N/A; requirement on ECU Configuration Specification [7]
[ECUC0025]	Compatible with iterative design	N/A; requirement on ECU Configuration Specification [7]
[ECUC0029]	Identify mechanisms not criteria	N/A; requirement on ECU Configuration Specification [7]
[ECUC0030]	Clarify configuration terminology	terms defined in AUTOSAR Glossary [2]
[ECUC0032]	ECU Configuration Description shall be the root for the whole configuration information of an ECU	N/A; requirement on ECU Configuration Specification [7]
[ECUC0039]	Support configuration of BSW	covered by this document
[ECUC0040]	Support configuration of RTE	see chapter 3
[ECUC0041]	Support AUTOSAR SW Component Integration	see chapter 2
[ECUC0043]	Duplication free description	N/A; requirement on ECU Configuration Specification [7]
[ECUC0046]	Support definition of configuration class	N/A; requirement on ECU Configuration Specification [7]

Requirement	Description	Satisfied by
[ECUC0047]	Pre-compile time configuration of BSW	N/A; requirement on ECU Configuration Specification [7]
[ECUC0048]	Link time configuration of BSW	N/A; requirement on ECU Configuration Specification [7]
[ECUC0049]	ECU Configuration description shall be tool processable	N/A; requirement on ECU Configuration Specification [7]
[ECUC0050]	Specify ECU Configuration Parameter Definition	N/A; requirement on ECU Configuration Specification [7]
[ECUC0053]	Support for multiple configuration sets	N/A; feature 128 canceled
[ECUC0055]	Support mandatory and optional configuration parameters	N/A; requirement on ECU Configuration Specification [7]
[ECUC0065]	Development according to the AUTOSAR Metamodeling Guide	N/A; requirement on ECU Configuration Specification [7]
[ECUC0066]	Transformation of ECUC modeling according to the AUTOSAR Model Persistence Rules for XML	N/A; requirement on ECU Configuration Specification [7]
[ECUC0070]	Support mandatory and optional containers	N/A; requirement on ECU Configuration Specification [7]
[ECUC0071]	Support for Generic Configuration Editor	N/A; requirement on ECU Configuration Specification [7]
[ECUC0072]	Support for referencing from dependent containers	N/A; requirement on ECU Configuration Specification [7]
[ECUC0073]	Support Service Configuration of AUTOSAR SW Components	postponed; no concept available yet
[ECUC0074]	Support Sequential ECU Configuration	N/A; requirement on ECU Configuration Specification [7]

2 Application Software Components

TODO *This BSW module is not part of the current release review cycle.*

3 RTE

The RTE provides the glue layer between the application SW-Components and the Basic Software thus enabling several application SW-Components to be integrated on one ECU. The RTE layer is shown in figure 3.1.

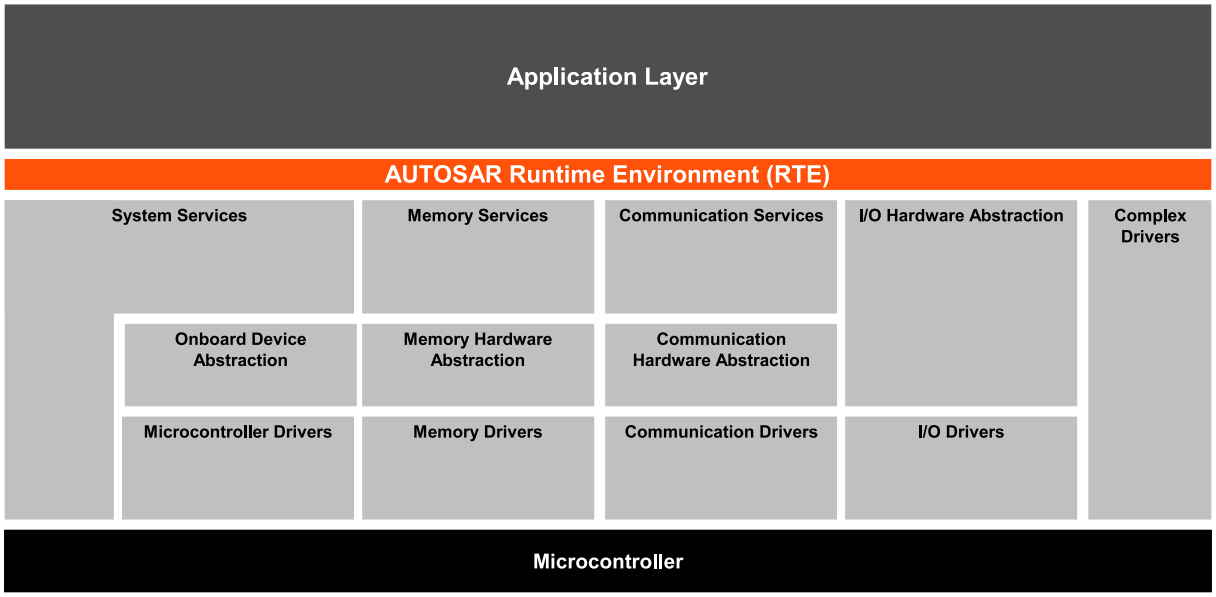


Figure 3.1: ECU Architecture RTE

The RTE is included twice in the development methodology of SW-Components. In the `RTE Contract phase` only the SW-Component description is used as an input. The configuration parameters defined in this section are used for the `RTE Generation phase`.

The overall structure of the RTE configuration parameters is shown in figure 3.2. It has to be distinguished between the configuration parameters for the RTE generator and the configuration parameters for the generated RTE itself.

Most of the information needed to generate an RTE is already available in the ECU extract of the System Description. From this extract also the links to the SW-Component descriptions and ECU Resource description are available. So only additional information not covered by the three aforementioned formats needs to be provided by the ECU Configuration description.

To additionally allow the most flexibility and freedom in the implementations of the RTE, only configuration parameters which are common to all implementations are standardized in the ECU Configuration Parameter definition. Any additional configuration parameters which might be needed to configure a full functional RTE have to be specified using the vendor specific parameter definition mechanism described in the ECU Configuration specification document [7].

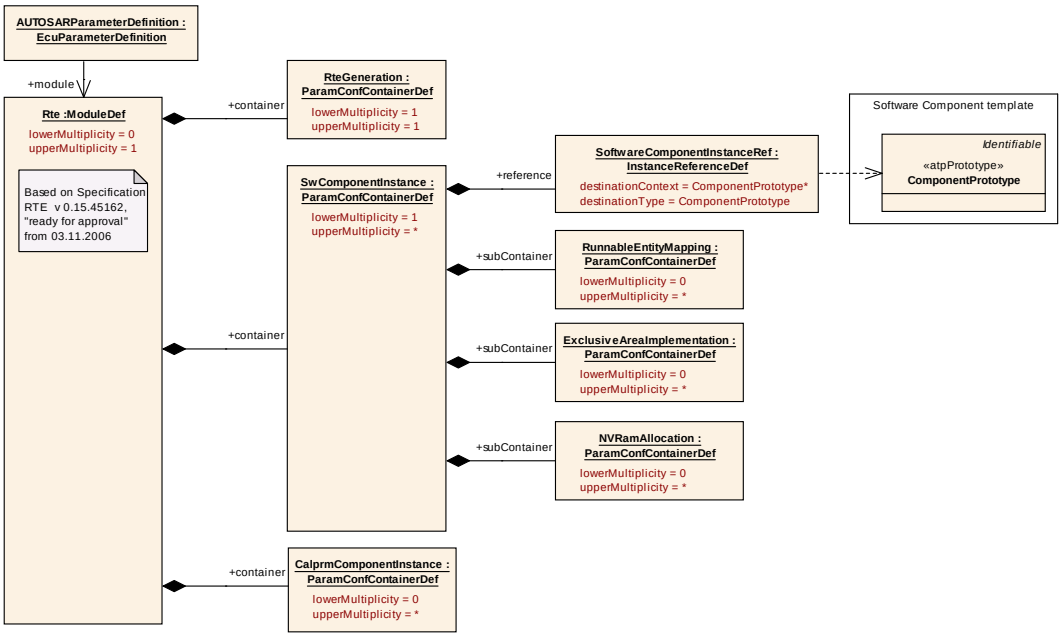


Figure 3.2: RTE configuration overview

The configuration of the RTE is structured in the following groups:

- `RteGeneration` in section 3.1
- `ImplementationSelection` in section 3.2.1
- `RunnableEntityMapping` in section 3.2.2
- `ExclusiveAreaImplementation` in section 3.2.3
- `NVRamAllocation` in section 3.2.4
- `CalprmComponentInstance` in section 3.3

Module Name	Rte	
Module Description	Configuration of the Rte (Runtime Environment) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CalprmComponentInstance	0..*	Specifies for each CalprmComponentType whether calibration is enabled.
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
RteGeneration	1	This container holds the parameters for the configuration of the RTE Generator tool itself.
SwComponentInstance	1..*	Representation of one SW-Component instance located on the to be configured ECU. All Rte configuration aspects are in relation to this SW-Component instance.

3.1 RTE Generation Parameters

The parameters in the container `RteGeneration` are used to configure the RTE generator. They all need to be defined during pre-compile time.

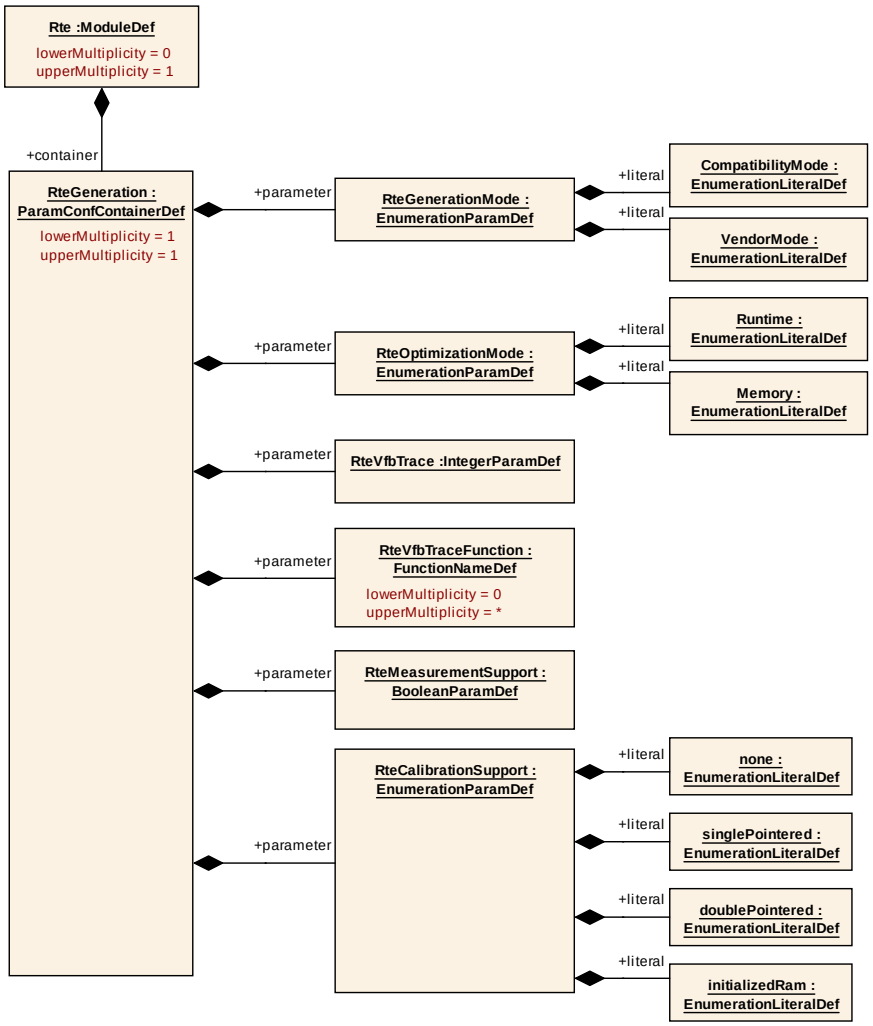


Figure 3.3: RTE generation parameters

RteGeneration

SWS Item	
Container Name	RteGeneration
Description	This container holds the parameters for the configuration of the RTE Generator tool itself.
Configuration Parameters	

Name last modified in: V1.1.0	RteCalibrationSupport		
Description	The RTE generator shall have the option to switch off support for calibration for generated RTE code. This option shall influence complete RTE code at once.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	doublePointered		
	initializedRam		
	none		
	singlePointered		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	RteGenerationMode		
Description	Switch between the two available generation modes of the RTE generator.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	CompatibilityMode		
	VendorMode		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	RteMeasurementSupport		
Description	The RTE generator shall have the option to switch off support for measurement for generated RTE code. This option shall influence complete RTE code at once.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	RteOptimizationMode		
Description	Switch between the two available optimization modes of the RTE generator.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	Memory		
	Runtime		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	RteVfbTrace {RTE_VFB_TRACE}		
Description	The RTE generator shall globally enable VFB tracing when RTE_VFB_TRACE is defined in the RTE configuration header file as a non-zero integer. The RTE generator shall globally disable VFB tracing when RTE_VFB_TRACE is defined in the RTE configuration header file as 0.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	RteVfbTraceFunction		
Description	The RTE generator shall enable VFB tracing for a given hook function when there is a #define in the RTE configuration header file for the hook function name and tracing is globally enabled. Example: #define Rte_WriteHook_i1_p1_a_Start		
Multiplicity	0..*		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

3.2 Handling of Software Component instances

When entities of Software Components are to be configured there is the need to actually address the instances of the `AtomicSoftwareComponentType`. AUTOSAR defines the *instance reference* as the mechanism to allow this addressing.

The special semantics of the `InstanceReferenceDef`[1] does allow to address each ”instance” of that `ComponentPrototype` in the Software Component template. Since the whole vehicle is described as one `top-level composition` using the Software Component template [6], each actual instance of any `AtomicSoftwareComponentType` can be addressed using this mechanism.

In figure 3.2 the container `SwComponentInstance` is shown which contains the instance reference `SoftwareComponentInstanceRef`. This instance reference is used to unambiguously identify each instance of a `ComponentType` within the ECU.

[1]The `InstanceReferenceDef` is based on the `<<instanceRef>>` mechanism introduced in the ”Template UML Profile and Modeling Guide” document [10].

SwComponentInstance

SWS Item			
Container Name	SwComponentInstance		
Description	Representation of one SW-Component instance located on the to be configured ECU. All Rte configuration aspects are in relation to this SW-Component instance.		
Configuration Parameters			
Name last modified in: V1.0.0	ImplementationRef		
Description	The Implementation which is assiged to the ComponentPrototype.		
Multiplicity	1		
Type	Foreign reference to Implementation		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	SoftwareComponentInstanceRef		
Description	Reference to a SW-Component ComponentPrototype. Semantic Constraint: Only ComponentPrototypes which have an AtomicSoftwareComponentType as <<isOfType>> shall be referenced here.		
Multiplicity	1		
Type	Instance reference to ComponentPrototype context: Component Prototype*		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
ExclusiveArea Implementation	0..*	
NVRamAllocation	0..*	
RunnableEntityMapping	0..*	

The container `SwComponentInstance` collects all the configuration information related to one specific instance of a `ComponentType`. The individual aspects will be described in the next sections.

3.2.1 Selection of SW-Component Implementation

During the system development there is no need to select the actual implementation which will be later integrated on one ECU. Therefore the ECU Extract of System Description may not contain the selection information yet.

In the SW-Component template an `Implementation` is always associated – via an `InternalBehavior` – with an `AtomicSoftwareComponentType`.

There is the possibility to have several instances of the same `AtomicSoftwareComponentType` within one ECU, therefore it shall be possible to assign a different `Implementation` to each of these `AtomicSoftwareComponentType`s. To achieve

this the association in the ECU Configuration is between the ComponentPrototype and the Implementation as shown in figure 3.4. As a restriction only ComponentPrototype which have an AtomicSoftwareComponentType as their type defined may be used in this assignment.

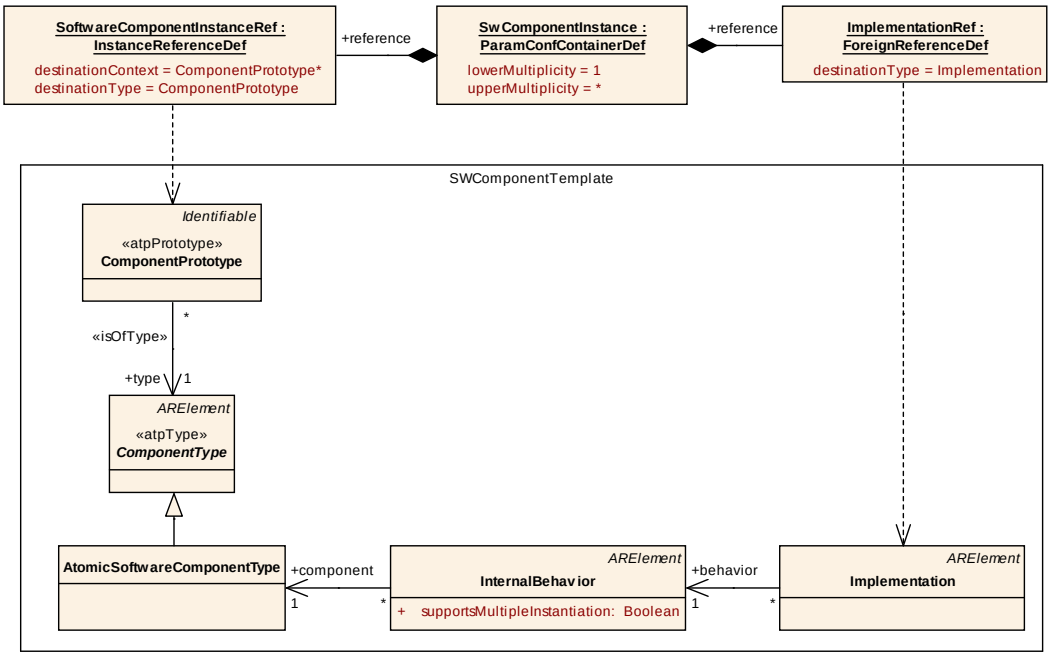


Figure 3.4: Selection of the Implementation for an AtomicSoftwareComponentType

With the reference ImplementationRef it can be specified for each ComponentPrototype which actual Implementation should be used.

3.2.2 Runnable Entity to task mapping

One of the major fragments of the RTE configuration is the mapping of SW-Component's RunnableEntitys on OS Tasks. The parameters defined to achieve this are shown in figure 3.5.

The mapping is based on the RTEEvent because it is the source of the activation. For each RunnableEntity which belongs to a SW-Component instance mapped on the ECU there needs to be a mapping container specifying how this RunnableEntity should be handled.

One major constraint is posed by the canBeInvocedConcurrently attribute of each RunnableEntity because data consistency issues have to be considered.

The MappedToTaskRef OsTask is part of the ECU Configuration description, so a plain ReferenceDef can be used to establish the link.

Another important parameter is the PositionInTask which provides an order of RunnableEntitys within the associated OsTask. When the task is executed periodically the PositionInTask parameter defines the order of execution

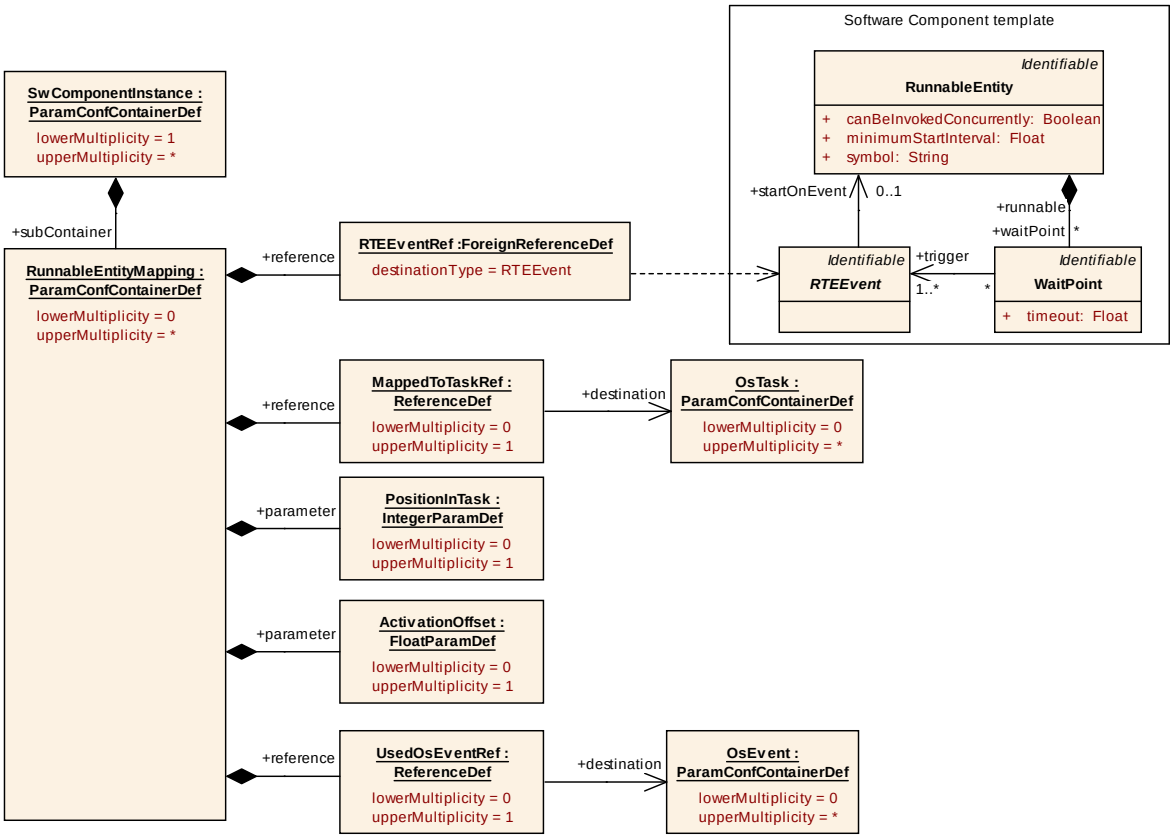


Figure 3.5: RTE runnable entity to task mapping

within the test. When the task is used to define a context for event activated `RunnableEntity`s the `PositionInTask` parameter defines the order of evaluation which actual `RunnableEntity` shall be executed. Thus providing means to define a deterministic delay between the beginning of execution of the task and the actual execution of the `RunnableEntity`'s code.

When an `OsEvent` is used to activate the `OsTask` the reference `UsedOsEventRef` specifies which `OsEvent` is used.

RunnableEntityMapping

SWS Item	
Container Name	RunnableEntityMapping
Description	Maps a RunnableEntity onto one OS Task based on the activating RTEEvent. Even if a RunnableEntity is executed in the caller's context this RunnableEntityMapping shall be specified, but no MappedToTask and PositionInTask elements given.
Configuration Parameters	

Name last modified in: V1.1.0	ActivationOffset		
Description	Activation offset in seconds.		
Multiplicity	0..1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	MappedToTaskRef		
Description	Reference to the OsTask the RunnableEntity is mapped to. If no reference to the OsTask is specified the RunnableEntity is executed in the context of the caller.		
Multiplicity	0..1		
Type	Reference to OsTask		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	PositionInTask		
Description	Each RunnableEntities mapped to an OsTask has a specific position within the task execution. For periodic activation this is the order of execution. For event driver activation this is the order of evaluation which actual RunnableEntity has to be executed.		
Multiplicity	0..1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	RTEEventRef		
Description	Reference to the description of the RTEEvent which is pointing to the RunnableEntity being mapped. This allows a fine grained mapping of RunnableEntites based on the activating RTEEvent.		
Multiplicity	1		
Type	Foreign reference to RTEEvent		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	UsedOsEventRef		
Description	If an OsEvent is used to activate the RunnableEntity it shall be referenced here.		
Multiplicity	0..1		
Type	Reference to OsEvent		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

3.2.3 Exclusive Area implementation

The RTE Generator can be configured to implement a different data consistency mechanism for each `ExclusiveArea` defined for a SW-Component.

In figure 3.6 the configuration of the actually selected data consistency mechanism is shown.

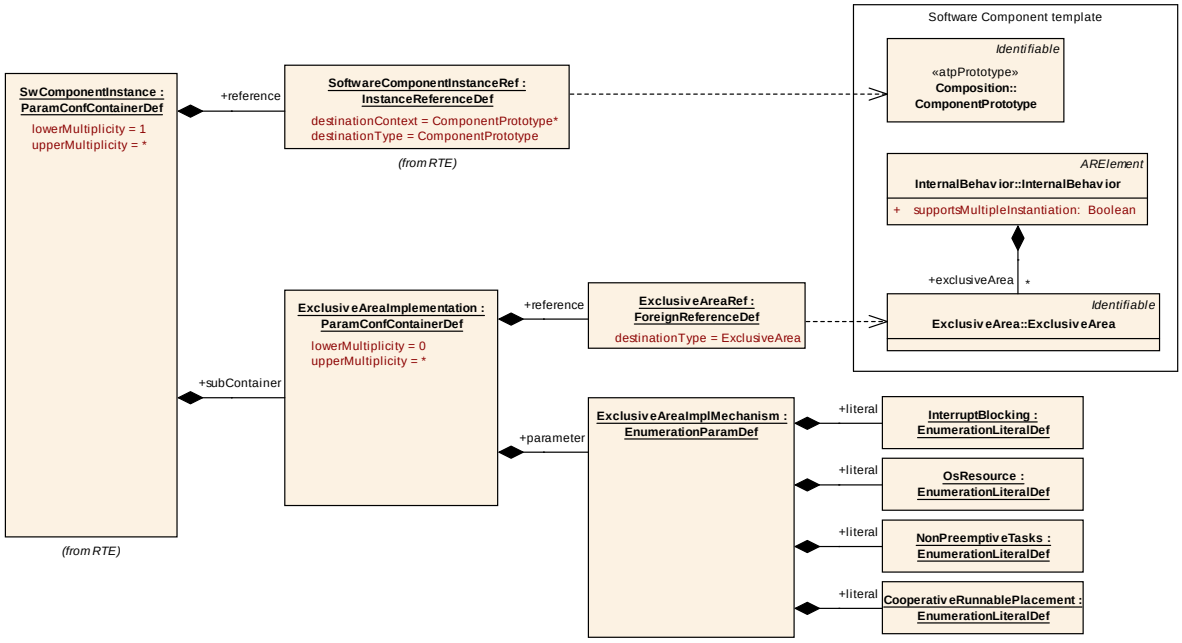


Figure 3.6: Configuration of the ExclusiveArea implementation

ExclusiveAreaImplementation

SWS Item	
Container Name	ExclusiveAreaImplementation
Description	Specifies the implementation to be used for the data consistency of this ExclusiveArea.
Configuration Parameters	

Name last modified in: V1.1.0	ExclusiveAreaImplMechanism		
Description	To be used implementation mechanism for the specified ExclusiveArea.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	CooperativeRunnable Placement		
	InterruptBlocking		
	NonPreemptiveTasks		
	OsResource		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	ExclusiveAreaRef		
Description	Reference to the ExclusiveArea.		
Multiplicity	1		
Type	Foreign reference to ExclusiveArea		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency **No Included Containers**			

3.2.4 NVRam Allocation

The configuration of the NVRam access does involve several templates, because it closes the gap between the SW-Components, the Services and the BSW Modules.

In figure 3.7 the related information from the SW-Component template is shown.

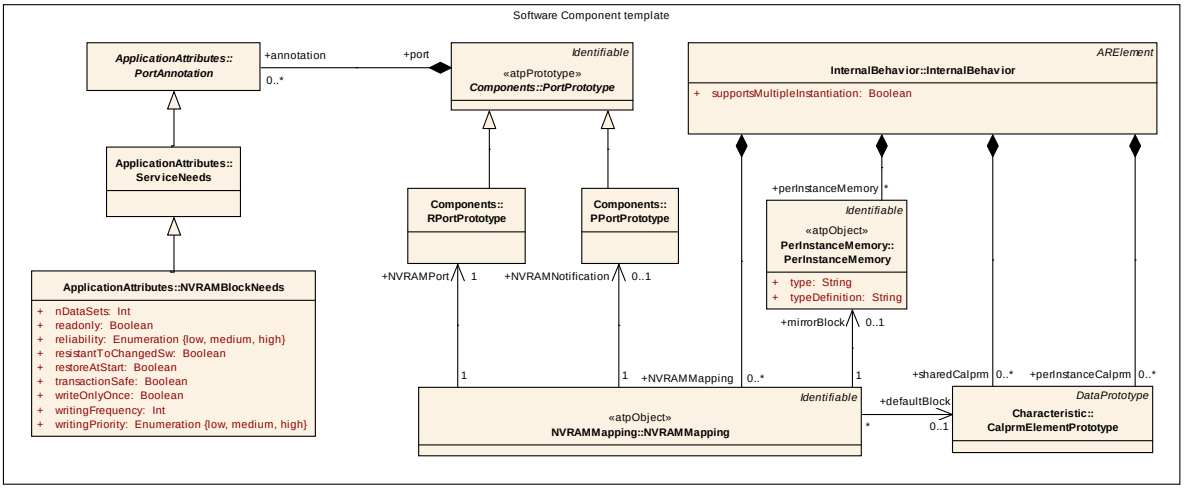


Figure 3.7: SW-Component information of NVRam Service needs

In figure 3.8 the ECU Configuration part of the NVRam allocation is shown. It relates the SW-Components `NVRAMMapping` information with the NVRam Managers `NvmBlockDescriptor` and the linker symbols of the RAM and ROM sections to be used.

NVRamAllocation

SWS Item	
Container Name	NVRamAllocation
Description	Specifies the relationship between the SW-Components NVRAMMapping / NVRAM needs and the NvM module configuration.
Configuration Parameters	

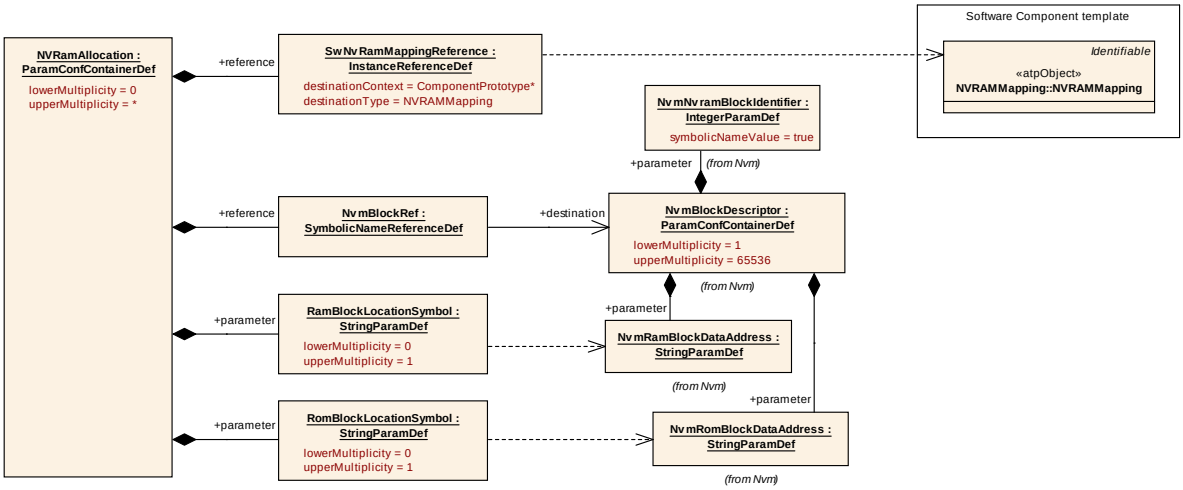


Figure 3.8: ECU Configuration of the NVRam Service

Name last modified in: V1.1.0	NvmBlockRef		
Description	Reference to the used NvM block for storage of the NVRAMMapping information.		
Multiplicity	1		
Type	Symbolic name reference to NvmBlockDescriptor		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	RamBlockLocationSymbol		
Description	This is the name of the linker object name where the NVRam Block will be mirrored by the Nvm. This symbol will be resolved into the parameter ”NvmRamBlockDataAddress” from the ”NvmBlockDescriptor”.		
Multiplicity	0..1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	RomBlockLocationSymbol		
Description	This is the name of the linker object name where the NVRom Block will be accessed by the Nvm. This symbol will be resolved into the parameter ”NvmRomBlockDataAddress” from the ”NvmBlockDescriptor”.		
Multiplicity	0..1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	SwNvRamMappingReference		
Description	Reference to the NVRAMMapping instance to be configured.		
Multiplicity	1		
Type	Instance reference to NVRAMMapping context: ComponentPrototype*		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

3.3 CalprmComponent instance configuration

In the SW-Component template the `CalprmComponentType` is the representation of the actual calibration data to be used in the ECU. Whether the calibration is enabled for a specific instance of the calibration data is configurable, as shown in figure 3.9.

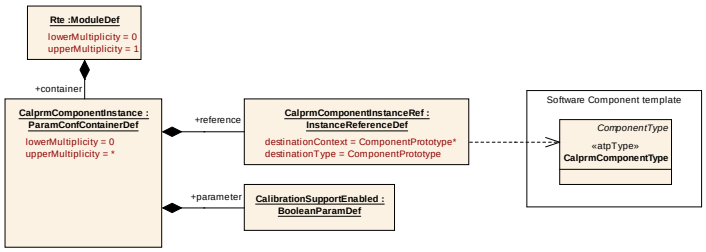


Figure 3.9: Configuration of the calibration for the CalprmComponentType

The instance reference `CalprmComponentInstanceRef` identifies one instance of the `CalprmComponentType` within the ECU and `CalibrationSupportEnabled`defines whether the calibration shall be supported for th specific instance.

CalprmComponentInstance

SWS Item			
Container Name	CalprmComponentInstance		
Description	Specifies for each CalprmComponentType whether calibration is enabled.		
Configuration Parameters			

Name last modified in: V1.1.0	CalibrationSupportEnabled		
Description	Enables calibration support for the specified CalprmComponentType instance.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	CalprmComponentInstanceRef		
Description	Reference to the CalprmComponentType instance.		
Multiplicity	1		
Type	Instance reference to ComponentPrototype context: Component Prototype*		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

3.4 Communication infrastructure

The configuration of the communication infrastructure (interaction of the RTE with the Com-Stack) is entirely predetermined by the ECU Extract provided as an input. The required input can be found in the AUTOSAR System Template [8] sections ”Data Mapping” and ”Communication”.

In case the RTE does utilize the Com module for intra-ECU communication it is up to the vendor-specific configuration of the RTE to ensure configuration consistency.

4 Communications

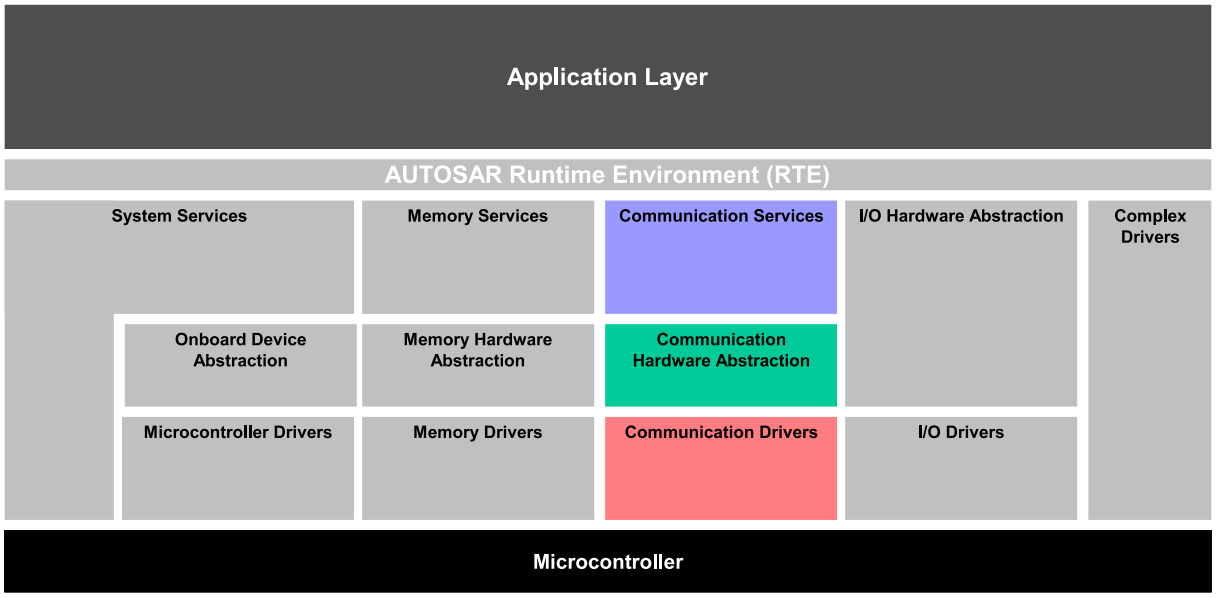


Figure 4.1: ECU Architecture COM

This chapter defines the configuration parameters of the AUTOSAR Basic Software Communication modules. The top level configuration model is shown in figure 4.2 with all the BSW modules defined in the COM-Stack.

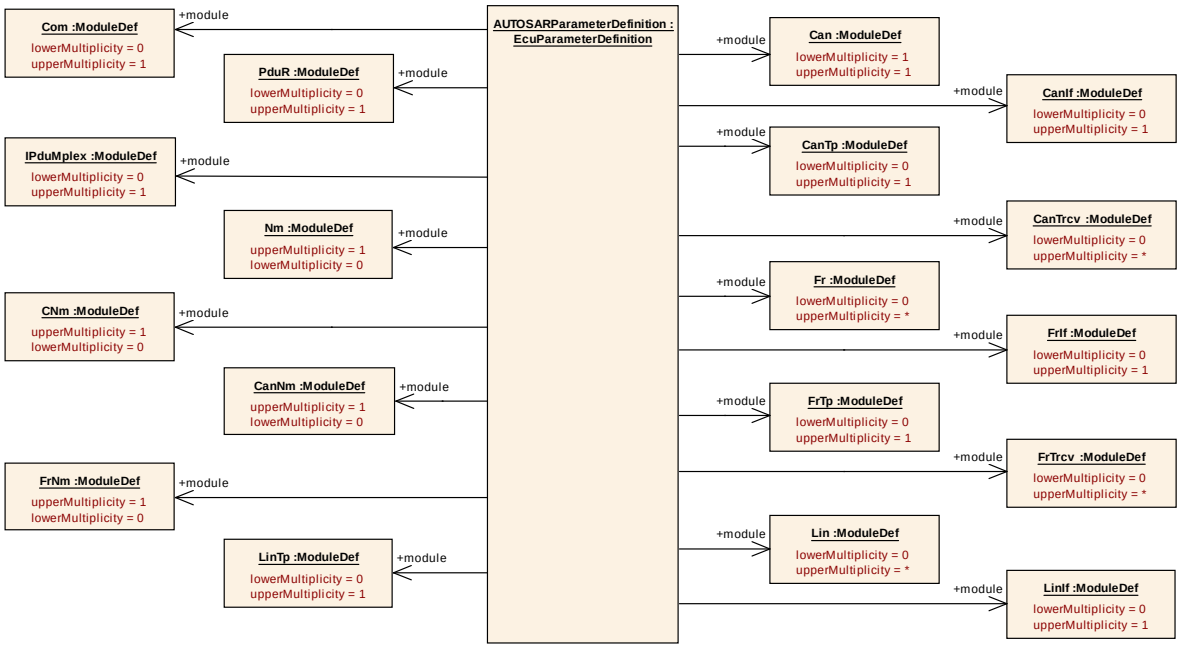


Figure 4.2: Overview: Parts of Communication configuration

4.1 COM-Stack Configuration Patterns

To cope with the complexity of the COM-Stack configuration reoccurring patterns have been applied which will be described in this section. Only the patterns, together with some examples, are shown. To get detailed specification of the configuration for each individual module please refer to the actual chapters of these modules in this document.

4.1.1 Handle IDs

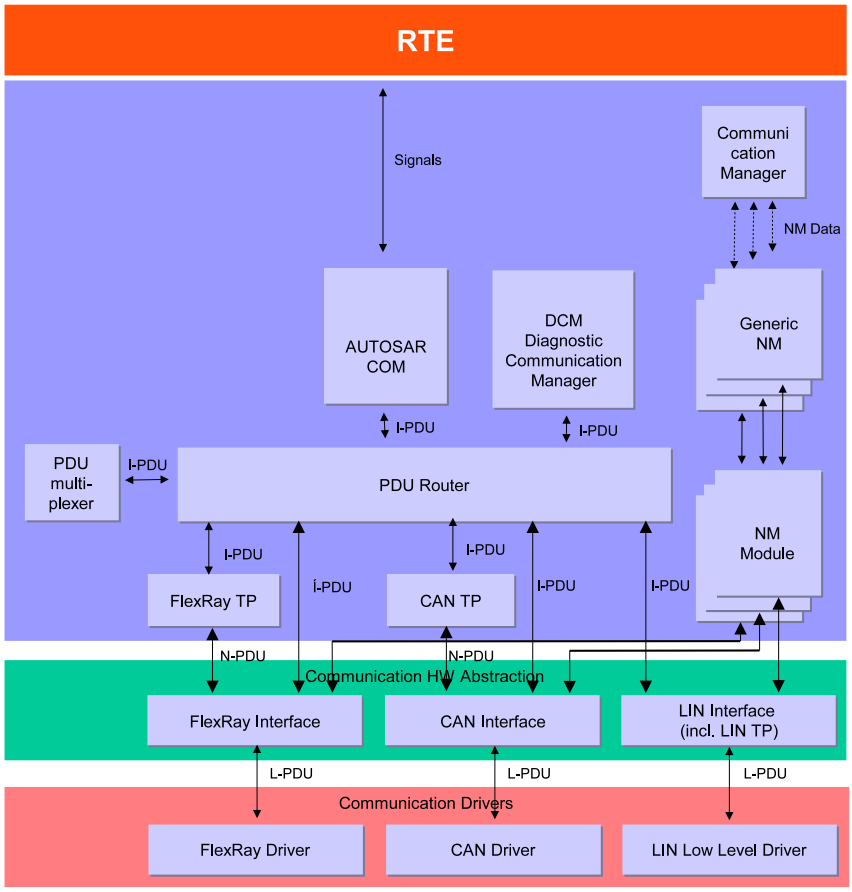


Figure 4.3: Interfaces in the COM-Stack [3]

In figure 4.3 a detailed view of the COM-Stack modules and their interaction is shown. There are several kinds of interactions between adjacent[1] modules.

[1]Modules are called adjacent if they share an interface, so PduR and Com are adjacent, while PduR and Can driver are not.

4.1.1.1 Handle ID concept

The API definitions in the COM-Stack utilize two concepts to achieve the interaction between adjacent modules:

- Pointers to PDU data buffer (the PDU data buffer contains the actual communicated information, depending on the actual layer the interaction happens)
- Handle IDs to identify to what PDU the pointer is referring to.

A typical API call is for instance:

```
PduR_ComTransmit(PduIdType ComTxPduId, PduInfoType *PduInfoPtr)
```

Handle IDs are defined by the module providing the API and used by the module calling the API.

The choice of the value for a Handle ID is open to the implementation of the providing module. There might be different strategies to optimize the Handle ID values and therefore the internal structures of the implementation may have an influence on the choice of the values.

Also the Handle IDs can be chosen freely per module, so a PDU might be sent from Com to the PduR with the `ID=5` and then the PduR transmits it further to the CanIf with `ID=19`. In the configuration information of the PduR it has to be possible to conclude that if a PDU arrives from Com with `ID=5` it has to be forwarded to the CanIf with `ID=19`.

It has to be guaranteed that each Pdu does have a unique handle ID within the scope of the corresponding API. For example: The PduR gets transmission requests from both, the Com and the Dcm modules. But there are also two distinct APIs defined for those requests:

- `PduR_ComTransmit(...)`
- `PduR_DcmTransmit(...)`

Therefore the PduR can distinguish two PDUs, even when they have the same handle ID but are requested via different APIs.

Another use-case in the COM-Stack only provides one API for all the callers: the interface layer (CanIf, FrIf, LinIf).

- `CanIf_Transmit(...)`

Here it has to be guaranteed that each transmit request for a distinct PDU does have a unique handle ID.

The actual values of the handle IDs can only be allocated properly when the configuration of one module is completed, since only then the internal data structures can be defined.

In the next sections the patterns used to define and utilize Handle IDs are described.

4.1.1.2 Definition of Handle IDs

The ECU Configuration Description (which holds the actual values of configuration parameters) is structured according to the individual BSW Module instances. Therefore the ECU Configuration Parameter Definition is also structured in this way.

In figure 4.4 an exemplary definition of a partial Can Interface transmit configuration is shown.

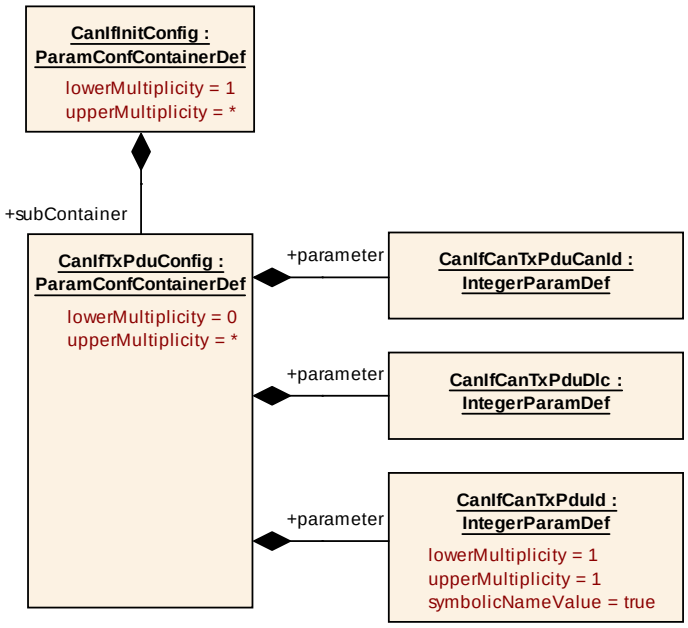


Figure 4.4: Example of Can Interface Tx configuration

The configuration of the module `CanIf` may contain several `CanIfTxPduConfig` objects.

Each `CanIfTxPduConfig` object contains information on one Pdu which is coming from an upper layer (e.g. PduR or Nm) and is going to some Can driver. In this example the `CanIfCanTxPduCanId` and `CanIfCanTxPduDlc` are specified for each to be transmitted Pdu. There is a similar structure needed for the receive use-case as well.

Additionally the parameter `CanIfCanTxPduId` is specified. This integer parameter will later hold the actual value for the handle ID. So the handle ID value is stored inside the structure of the defining module.

Since the handle ID `CanIfCanTxPduId` is part of the container `CanIfTxPduConfig` the semantics of the symbolic names can be applied (see section 4.1.1.5).

4.1.1.3 Definition of PDUs

With the possibility to define Handle IDs a module is now able to publish the Handle ID values to other modules. In this section it shall be described how other modules can relate to those Handle IDs and how to define the flow of a PDU[2] through the COM-Stack.

To be able to define a PDU ”flowing” through the COM-Stack two modules need to be able to refer to the same PDU object. Therefore a generic `Pdu` container has been defined which does not belong to any module but to the whole COM-Stack.

Since the PDU flowing through the COM-Stack does not belong to an individual module, the ”virtual” module `EcuC` has been introduced in the ECU Configuration. This module is used to collect configuration information not associated with any specific standardized module.

The `PduCollection` may contain several ”global” `Pdu` objects as shown in figure 4.5. Each `Pdu` may be representing an actual `FrameInstance` from the AUTOSAR System Description (the ECU Extract), therefore there is an optional reference to an element in the System Template. The reference is optional because the `PDU`s which are transmitted / received within an ECU only (i.e. `PDU`s used for diagnostic purposes) are not part of the ECU Extract.

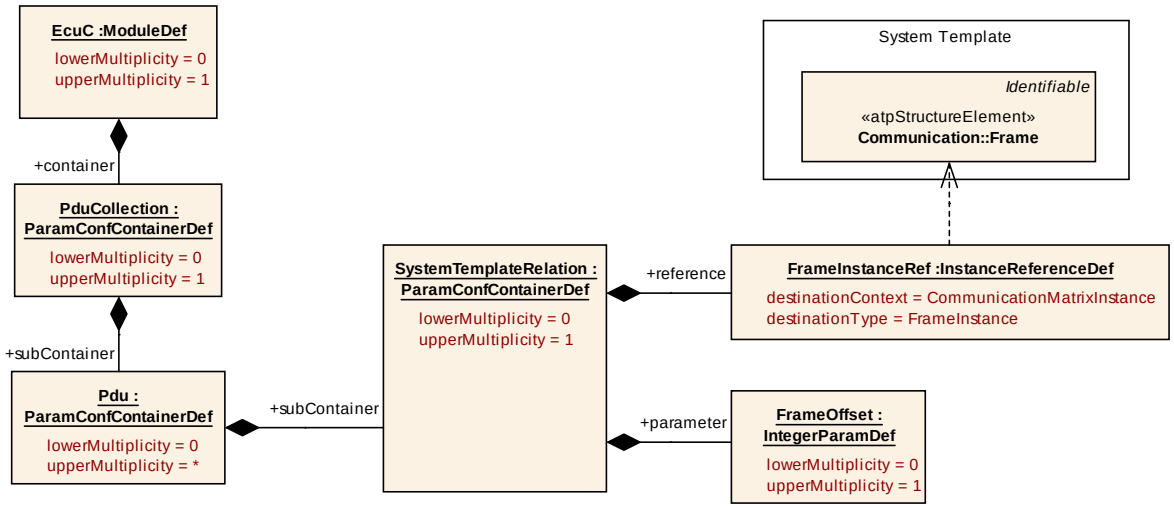


Figure 4.5: Generic Pdu Container

Pdu

SWS Item	
Container Name	Pdu
Description	One Pdu flowing through the COM-Stack. This Pdu is used by all Com-Stack modules to aggree on referencing the same Pdu.
Configuration Parameters	

[2]For the configuration it does not matter what kind of PDU it is, i.e. I-PDU, L-PDU or N-PDU.

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
SystemTemplateRelation	0..1	

SystemTemplateRelation

SWS Item			
Container Name	SystemTemplateRelation		
Description	Container to hold the relationship information between the Pdu and the FrameInstance in the System Template. If this container is not present the Pdu is only ”local” to the ECU (e.g. to define the communication between the TPs, the PduR and Dcm).		
Configuration Parameters			

Name last modified in: V1.0.0	FrameInstanceRef		
Description	Reference to the System Template’s FrameInstance this Pdu is contained in.		
Multiplicity	1		
Type	Instance reference to FrameInstance context: CommunicationMatrix Instance		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FrameOffset		
Description	If the Pdu is only a part of a bigger Frame on the bus (e.g. FlexRay), this parameter holds the offset where this Pdu starts within the Frame. In bits.		
Multiplicity	0..1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

But there are also PDUs possible which are not directly related to the ECU Extract. Especially PDUs handled by the Transport Protocol modules have no representation in the ECU Extract (There is a PDU coming over the bus which is represented by a `Pdu` object, but when the TP does the conversion and a new `Pdu` is created which then is forwarded to the upper layer. This created `Pdu` does not have a reference to a `FrameInstance`).

4.1.1.4 Agreement on Handle IDs

During the configuration of a module, information for each `Pdu` flowing through this module is created (see again figure 4.4: `CanIfTxPduConfig`) which hold module-specific configuration information. Now each of these ”local” `Pdu` configurations needs to be related to a ”global” `Pdu` element representing information flowing through the COM-Stack. This is done by introducing a `ReferenceDef` from the ”local” `Pdu` to the ”global” `Pdu`.

In figure 4.6 this relationship is shown for the `PduRDestPdu` and the `CanIfTxPduConfig`.

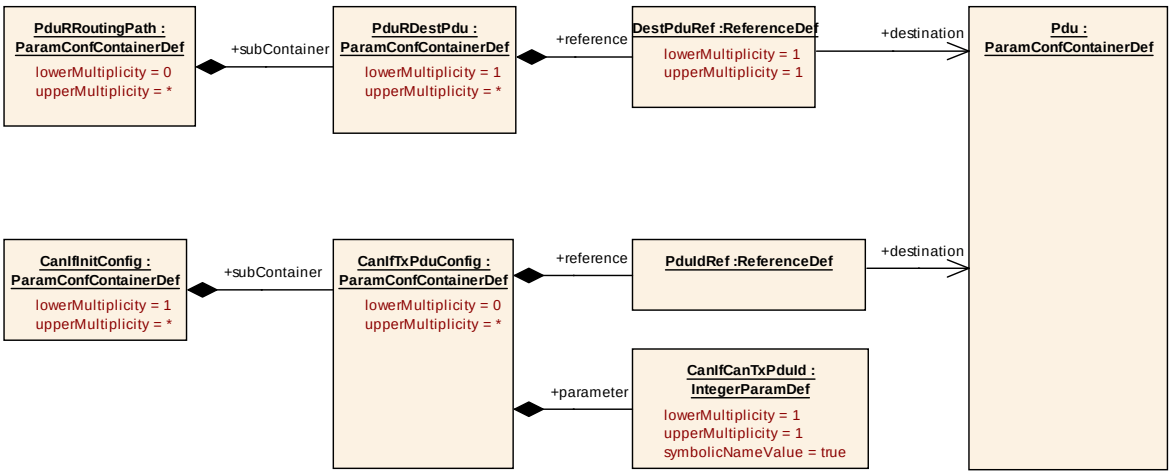


Figure 4.6: Transmission from PduR to CanIf

There are two reasons why the ”global” PDU has been introduced and why all ”local” PDUs have to point to the ”global” PDU only.

- When doing the configuration of module PduR only the ”global” PDU needs to be present, there is no need for the ”local” PDU in the CanIf to be present yet.
- The References are stored in the ”local” PDU structure, so changes applied do only influence the structure of the changed module.

Taking the structure shown in figure 4.6 it is now possible to generate both modules.

The CanIf (automatic) configuration editor collects all ”local” `CanIfTxPduConfig`s and generates/stores the values for their handle ID in `CanIfCanTxPduId`. If the CanIf needs to know where the Pdu transmit request is coming from it can follow the `PduIdRef` to the ”global” Pdu and then ”query” all references pointing to that Pdu. By following those references in reversed direction the transmitting module can be found.

The PduR generator has to know which handle ID to use for each Pdu that has to be sent to the CanIf. To get the actual handle ID value the mechanism is the same in the CanIf use-case: follow the ”global” Pdu reference and ”query” the modules pointing to that ”global” Pdu. Then find the module(s) type this Pdu is going to be transmitted to. In case of a multicast there might be several modules to send the same PDU to.

With this approach a high degree of decoupling has been achieved between the configuration information of the involved modules. Even when modules are adjacent and need to share information like handle ID, the references between the modules are always indirect using the ”global” Pdu elements.

4.1.1.5 Handle IDs with symbolic names

In the current approach for configuring the handle IDs it is assumed that all involved modules have been configured completely. Then the actual handle ID values are assigned using a special (automatic) mode of the configuration editors of these modules.

To allow an even higher degree of decoupling the configuration and generation of individual modules an addition, using symbolic names for the handle IDs instead of the actual values, can be applied.

By adding an annotation to the container holding the handle ID parameter (in figure 4.6 this would be the container CanIfTxPduConfig) the code generator doing the CanIf could generate a #define in the CanIf_cfg.h file. The name of the define would be the name of the container (each container does have a short name) and the value would be the actual number assigned to that handle ID.

For example in CanIf_cfg.h:

```
#define CanIfTx_Pdu_2345634_985_symbol   17
```

The benefit is that the generator of the PduR would not need to wait for the CanIf to be configured completely and handle IDs generated. If the CanIf publishes the symbolic names and values for the handle IDs, the PduR can expect those symbolic names and generate the PduR code using those symbolic names.

For example in PduR.c:

```
CanIf_Transmit( CanIfTx_Pdu_2345634_985_symbol, PduPtr )
```

Therefore the PduR could be generated as soon as its own configuration is finished and there is no need to wait for the CanIf to be finished as well. However, at least the ”local” Pdu in the CanIf has to be already created to allow this, because the name of the symbol has to be fetched from this configuration.

Of course the PduR can only be compiled after the CanIf has been generated as well, but with the utilization of the symbolic names together with handle IDs an even higher degree of decoupling in the configuration process can be achieved.

4.2 Com

Remark/30.1.2007: This section conforms R2.0

, due to the fact, that there is no R2.1 SWS available.

The top-level structure of the Com [15] module configuration parameters is shown in figure 4.7. The Com module configuration is split into several containers holding different aspects of the configuration. These will be described in the following sections.

- ComGeneral, see section 4.2.1
- ComSignal, see section 4.2.2
- ComSignalGroup, see section 4.2.3
- ComNetworkSignal, see section 4.2.4
- ComIPdu, see section 4.2.5
- ComIPduGroup, see section 4.2.6

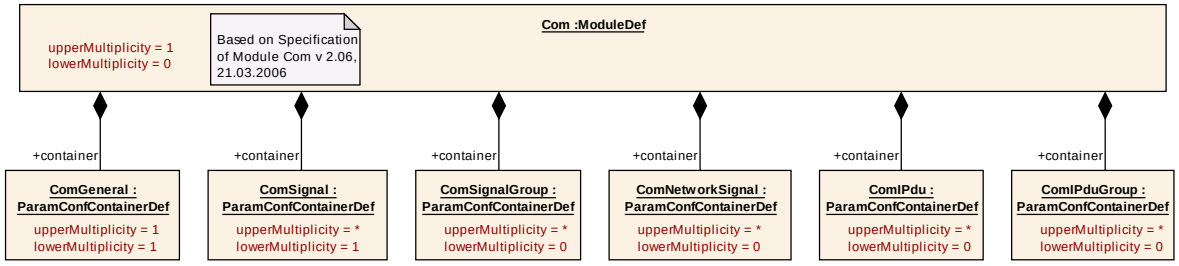


Figure 4.7: Com module configuration overview

Com

Module Name	Com	
Module Description	Configuration of the Com module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
ComGeneral	1	Contains the general configuration parameters of the Com module.
ComIPdu	0..*	Contains the configuration parameters of Com I-Pdus. The shortName is used as the symbolic name (COM_IPDU_NAME) of this I-Pdu when communicating with the PduR. Is optional because the Com module might be used for internal communication only. This parameter is only stored in the XML file, and must not be used within the implementation.

Container Name	Multiplicity	Scope / Dependency
ComIPduGroup	0..*	Contains the configuration parameters of Com I-Pdu groups. The shortName is used as the symbolic name of the I-Pdu group (COM_IPDU_GROUP_NAME). This parameter is only stored in the XML file, and must not be used within the implementation. The minimum is 2 because we expect that the ComM addresses one RX-group and one TX-group to implement normal and listen only mode. Since a group can only have send or receive I-PDUs you could theoretically have just one group if you only send or receive I-PDUs. But we expected that this would cause problems with the ComM. Therefore the minimum was set to 2. (Bug#11347).
ComNetworkSignal	0..*	Contains the configuration parameters of Com network signals. The shortName of this container holds the ”COM_NETWORK_SIGNAL_NAME” [COM156]. The symbolic name of the network signal. This parameter is only stored in the XML file, and must not be used within the implementation.
ComSignal	1..*	Contains the configuration parameters of Com signals. The shortName is used as the symbolic name of the signal (COM_SIGNAL_NAME). This name is also used as the handle name for the signal. This parameter is only stored in the XML file, and must not be used within the implementation.
ComSignalGroup	0..*	Contains the configuration parameters of Com signal groups. The shortName is used as the symbolic name of the signal group (COM_SIGNAL_GROUP_NAME). This name is also used as the handle name for the signal group. This parameter is only stored in the XML file, and must not be used within the implementation.
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.

4.2.1 ComGeneral

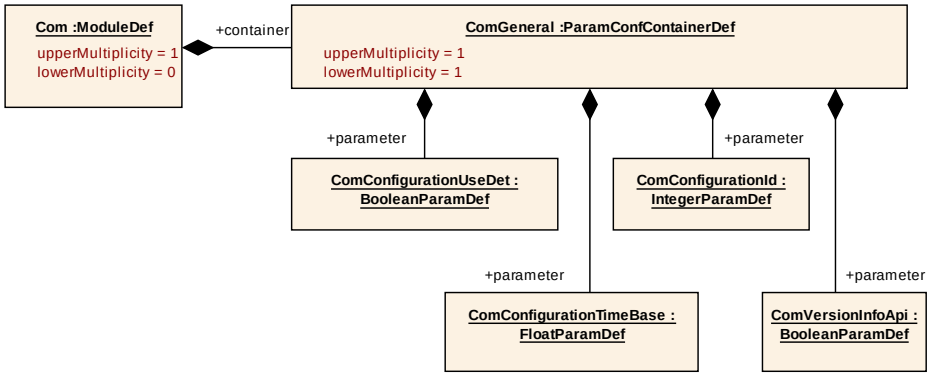


Figure 4.8: Com module general configuration parameters

ComGeneral

SWS Item	[COM337]
Container Name	ComGeneral {COM_CONFIGURATION}
Description	Contains the general configuration parameters of the Com module.
Configuration Parameters	

Name last modified in: V1.0.0	ComConfigurationId {COM_CONFIGURATION_ID} [COM394]		
Description	This ID is returned by a call to Com_GetConfigurationId().		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	ComConfigurationTimeBase {COM_CONFIGURATION_TIME_BASE} [COM186]		
Description	The period between successive calls to the Main Functions (Rx, Tx, Routing) of AUTOSAR COM in seconds.		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComConfigurationUseDet {COM_CONFIGURATION_USE_DET} [COM141]		
Description	The error hook shall contain code to call the Det. If this parameter is configured COM_DEV_ERROR_DETECT shall be set to ON as output of the configuration tool. (as input for the source code).		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComVersionInfoApi {COM_VERSION_INFO_API} [COM438]		
Description	Activate/Deactivate the version information API (see Com specification chapter 8.3.1.9).		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency **No Included Containers**			

4.2.2 ComSignal

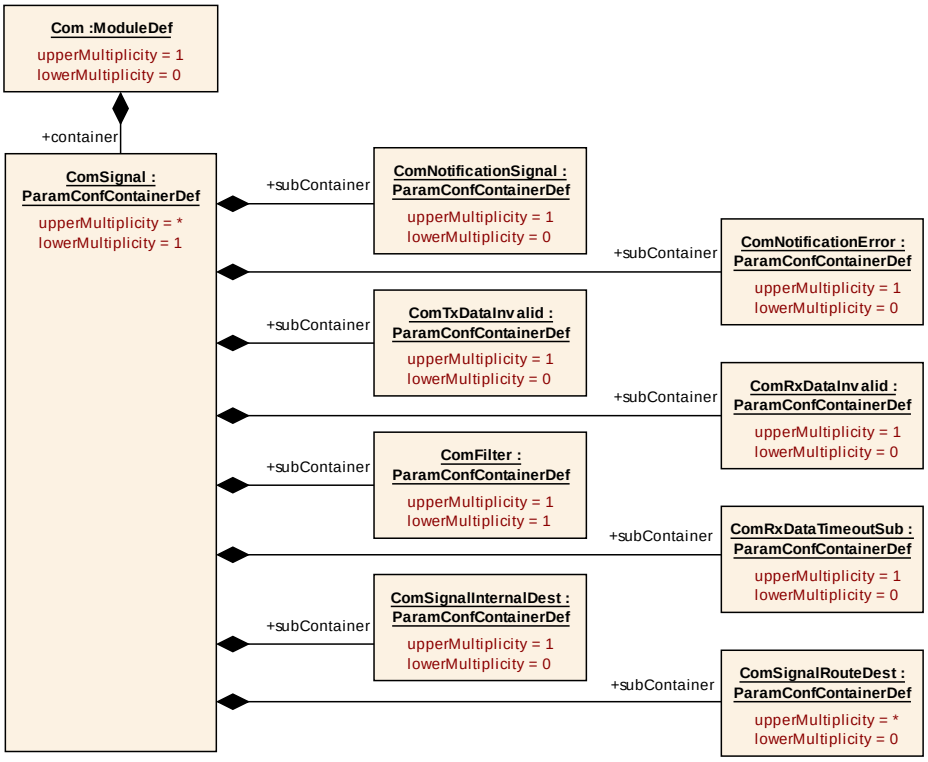


Figure 4.9: ComSignal configuration containers

Additional containers, which the ComSignal configuration parameter may hold, are shown in figure 4.9. The description of these containers can be found in the following sections:

- ComFilter in section 4.2.2.1
- ComNotificationError in section 4.2.2.2
- ComNotificationSignal in section 4.2.2.2
- ComRxDataInvalid in section 4.2.2.4
- ComRxDataTimeoutSub in section 4.2.2.3
- ComSignalInternalDest in section 4.2.2.5
- ComSignalRouteDest in section 4.2.2.5
- ComTxDataInvalid in section 4.2.2.4

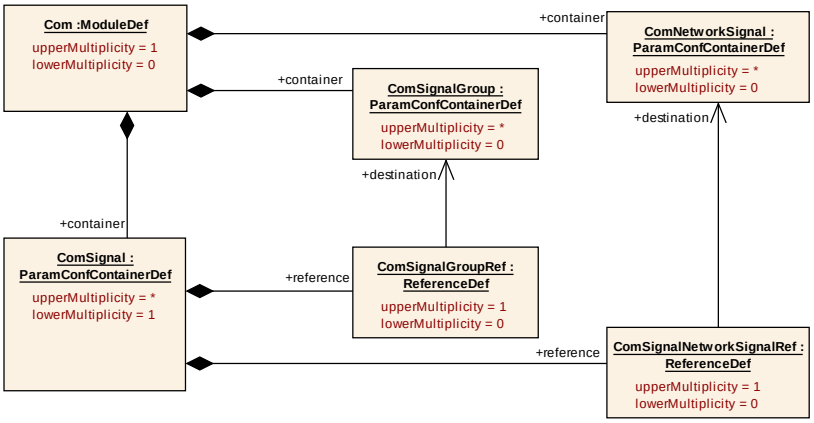


Figure 4.10: ComSignal references

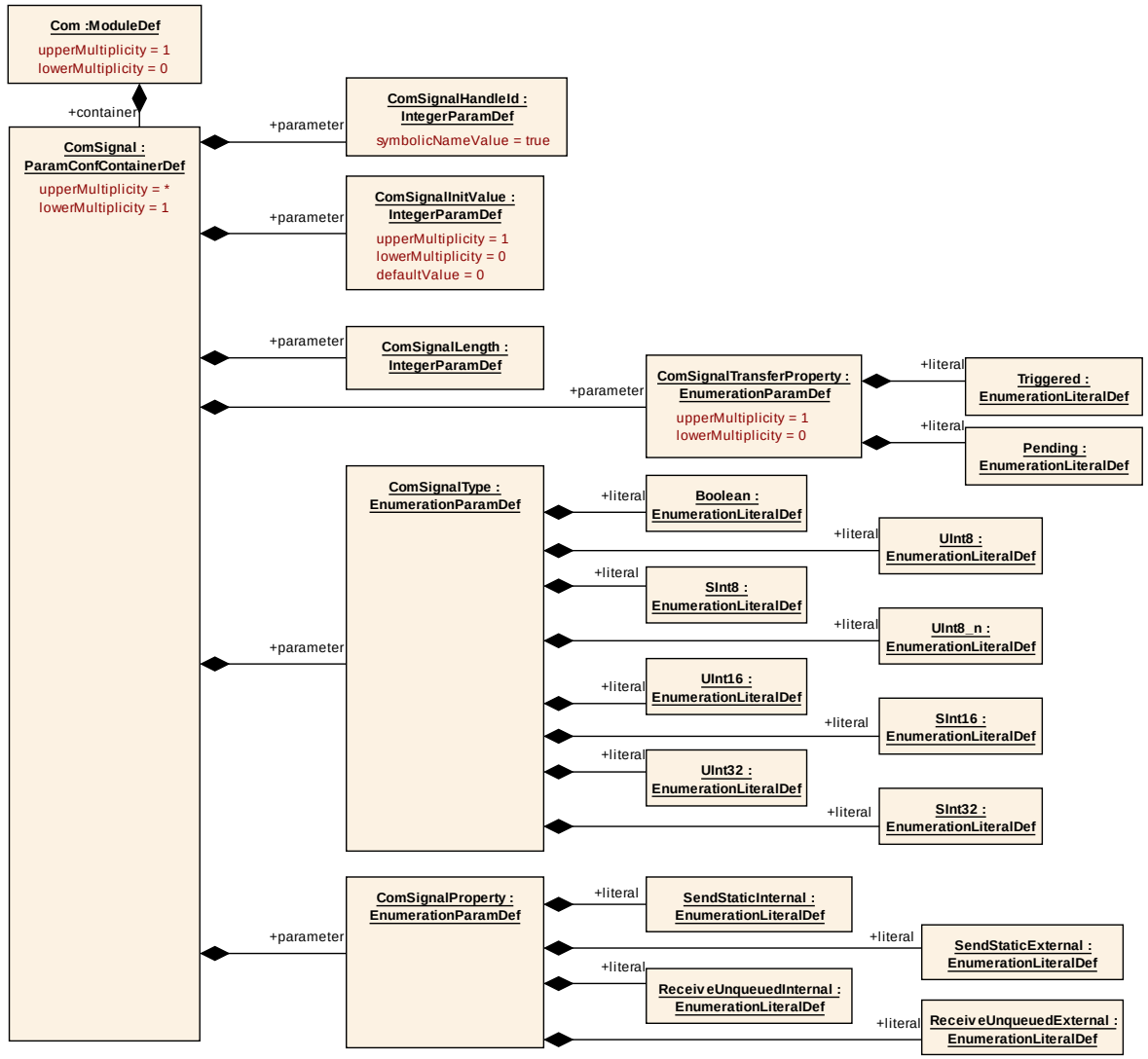


Figure 4.11: ComSignal configuration parameters

ComSignal

SWS Item	[COM344, COM407]
Container Name	ComSignal {COM_SIGNAL}
Description	Contains the configuration parameters of Com signals. The shortName is used as the symbolic name of the signal (COM_SIGNAL_NAME). This name is also used as the handle name for the signal. This parameter is only stored in the XML file, and must not be used within the implementation.
Configuration Parameters	

Name last modified in: V1.0.0	ComSignalGroupRef {COM_SIGNAL_GROUP_REF} [COM045]		
Description	If the signal belongs to a signal group, this is the reference to the containing signal group. If the signal does not belong to any signal group, the parameter shall not appear.		
Multiplicity	0..1		
Type	Reference to ComSignalGroup		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComSignalHandleId {COM_SIGNAL_HANDLE_ID} [COM165]		
Description	The numerical value used as the ID of this signal . The ComSignalHandleId is required by the API calls to send and receive signals.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComSignalInitValue {COM_SIGNAL_INIT_VALUE} [COM170]		
Description	Initial value for this signal. Default value is 0.		
Multiplicity	0..1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration)
	Post-build time	X	Variant3 (Post-build Configuration)
Scope / Dependency			

Name last modified in: V1.0.0	ComSignalLength {COM_SIGNAL_LENGTH} [COM437]		
Description	The COM_SIGNAL_LENGTH specifies the length of the signal in Bytes. This is necessary specially for the type UINT8[n], where it specifies ”n”, but also for the other types it has to be filled in.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComSignalNetworkSignalRef {COM_SIGNAL_NETWORK_SIGNAL_REF} [COM167]		
Description	If this signal is sent over a network this attribute is the reference to the matching network signal.		
Multiplicity	0..1		
Type	Reference to ComNetworkSignal		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComSignalProperty {COM_SIGNAL_PROPERTY} [COM166]		
Description	Signal property definition according to section MESSAGEPROPERTY name 3.2.10.1 in the [STD_OSEK_OIL] specification.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	ReceiveUnqueuedExternal		
	ReceiveUnqueuedInternal		
	SendStaticExternal		
	SendStaticInternal		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComSignalTransferProperty {COM_SIGNAL_TRANSFER_PROPERTY} [COM232]		
Description	Derived from [STD_OSEK_OIL].		
Multiplicity	0..1		
Type	EnumerationParamDef		
Range	Pending		
	Triggered		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration)
	Post-build time	X	Variant3 (Post-build Configuration)
Scope / Dependency			

Name last modified in: V1.0.0	ComSignalType {COM_SIGNAL_TYPE} [COM127]		
Description	The AUTOSAR type of the signal. Whether or not the signal is signed or unsigned can be found by examining the value of this attribute. This type could also be used to reserved appropriate storage in AUTOSAR COM.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	Boolean		
	SInt16		
	SInt32		
	SInt8		
	UInt16		
	UInt32		
	UInt8		
	UInt8_n		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Included Containers

Container Name	Multiplicity	Scope / Depedency
ComFilter	1	
ComNotificationError	0..1	
ComNotificationSignal	0..1	
ComRxDataInvalid	0..1	
ComRxDataTimeoutSub	0..1	
ComSignalInternalDest	0..1	
ComSignalRouteDest	0..*	
ComTxDataInvalid	0..1	

4.2.2.1 ComFilter

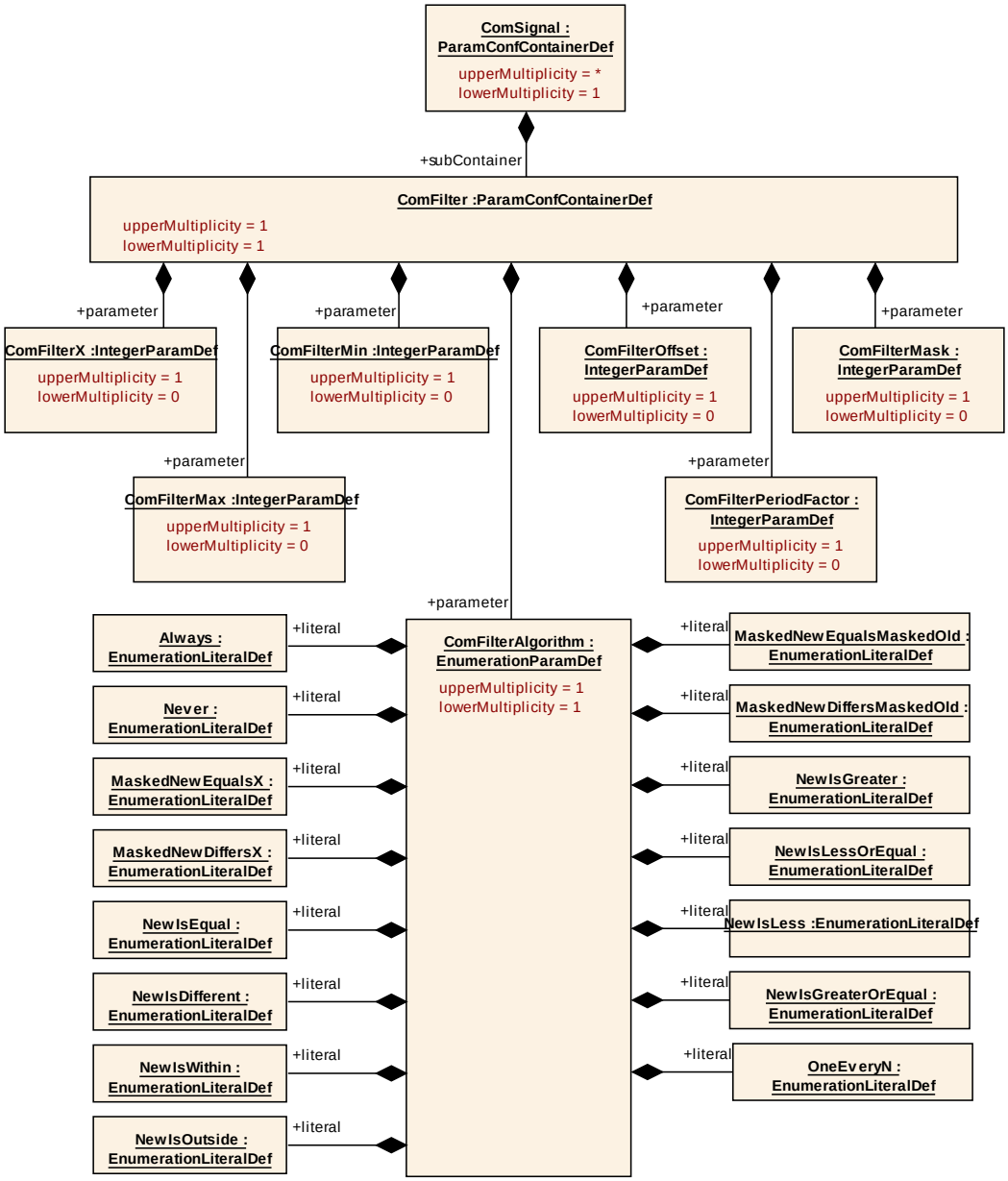


Figure 4.12: ComFilter configuration parameters

ComFilter

SWS Item	[COM339]
Container Name	ComFilter {COM_FILTER}
Description	This container contains the configuration parameters of COM Filters.
Configuration Parameters	

Name last modified in: V1.0.0	ComFilterAlgorithm {COM_FILTER_ALGORITHM} [COM146]		
Description	The range of values is specified in the [STD_OSEK_COM] specification, chapter 2.2.2, Reception Filtering.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	Always		
	MaskedNewDiffersMaskedOld		
	MaskedNewDiffersX		
	MaskedNewEqualsMaskedOld		
	MaskedNewEqualsX		
	Never		
	NewIsDifferent		
	NewIsEqual		
	NewIsGreater		
	NewIsGreaterOrEqual		
	NewIsLess		
	NewIsLessOrEqual		
	NewIsOutside		
	NewIsWithin		
	OneEveryN		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComFilterMask {COM_FILTER_MASK} [COM235]		
Description	The name of this attribute corresponds to the parameter name in the [STD_OSEK_COM] specification of Reception Filtering. Only the least significant 32 bits are significant.		
Multiplicity	0..1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComFilterMax {COM_FILTER_MAX} [COM317]		
Description	The name of this attribute corresponds to the parameter name in the [STD_OSEK_COM] specification of Reception Filtering. Only the least significant 32 bits are significant.		
Multiplicity	0..1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComFilterMin {COM_FILTER_MIN} [COM318]		
Description	The name of this attribute corresponds to the parameter name in the [STD_OSEK_COM] specification of Reception Filtering. Only the least significant 32 bits are significant.		
Multiplicity	0..1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComFilterOffset {COM_FILTER_OFFSET} [COM313]		
Description	The name of this attribute corresponds to the parameter name in the [STD_OSEK_COM] specification of Reception Filtering. Only the least significant 32 bits are significant.		
Multiplicity	0..1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComFilterPeriodFactor {COM_FILTER_PERIOD} [COM312]		
Description	The name of this attribute corresponds to the parameter name in the [STD_OSEK_COM] specification of Reception Filtering. Only the least significant 32 are significant.		
Multiplicity	0..1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComFilterX {COM_FILTER_X} [COM147]		
Description	The name of this attribute corresponds to the parameter name in the [STD_OSEK_COM] specification of Reception Filtering. Only the least significant 32 bits are significant.		
Multiplicity	0..1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency **No Included Containers**			

4.2.2.2 ComNotification

Both, the ComSignal and the ComSignalGroup, are using the two means of notification: Signal and Error notification.

Therefore the two containers `ComNotificationSignal` and `ComNotification-Error` have been introduced. Both contain a `ComNotification` container which holds the actual information on the notification configuration.

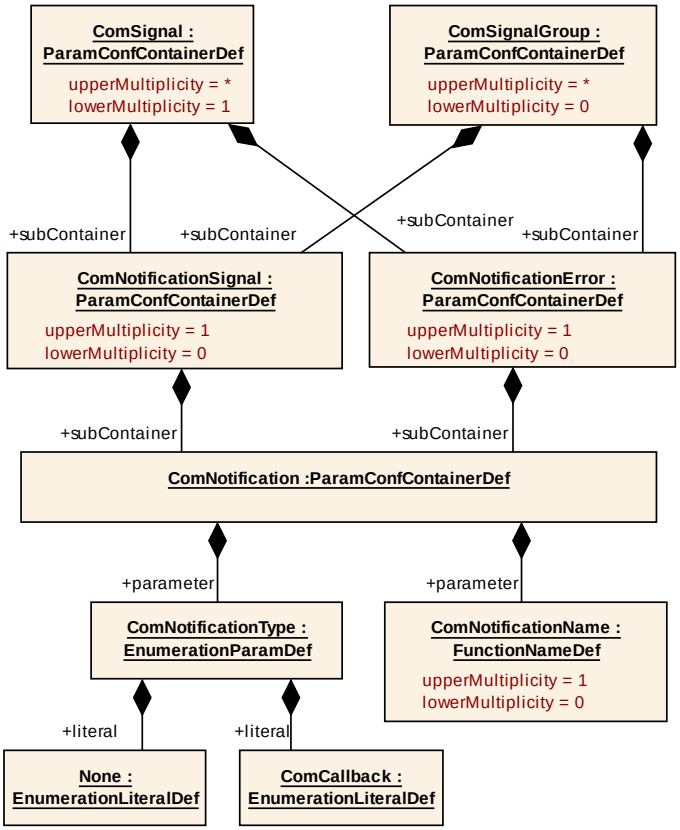


Figure 4.13: ComNotificationSignal and -Error configuration parameters

ComNotificationSignal

SWS Item	[COM448, COM450, COM452]	
Container Name	ComNotificationSignal {COM_NOTIFICATION_SIGNAL}	
Description	Contains the configuration parameters of Com notifications in the case a notification for a signal has to be done.	
Configuration Parameters Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
ComNotification	1	

ComNotificationError

SWS Item	[COM449, COM451, COM453]	
Container Name	ComNotificationError {COM_NOTIFICATION_ERROR}	
Description	Contains the configuration parameters of Com notifications in the case a notification for an error has to be done.	
Configuration Parameters Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
ComNotification	1	

ComNotification

SWS Item	[COM343]		
Container Name	ComNotification {COM_NOTIFICATION}		
Description	Contains the configuration parameters of Com notifications.		
Configuration Parameters			
Name last modified in: V1.0.0	ComNotificationName {COM_NOTIFICATION_NAME} [COM247]		
Description	If ComNotificationType is ComCallback then this is the name of a callback function.		
Multiplicity	0..1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComNotificationType {COM_NOTIFICATION_TYPE} [COM246]		
Description	Type of notification provided to RTE. Note that FLAG, ACTIVATETASK and SETEVENT, although present in OIL, have been omitted as they are not used in AUTOSAR RTE.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	ComCallback		
	None		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency **No Included Containers**			

4.2.2.3 ComRxDataTimeoutSub

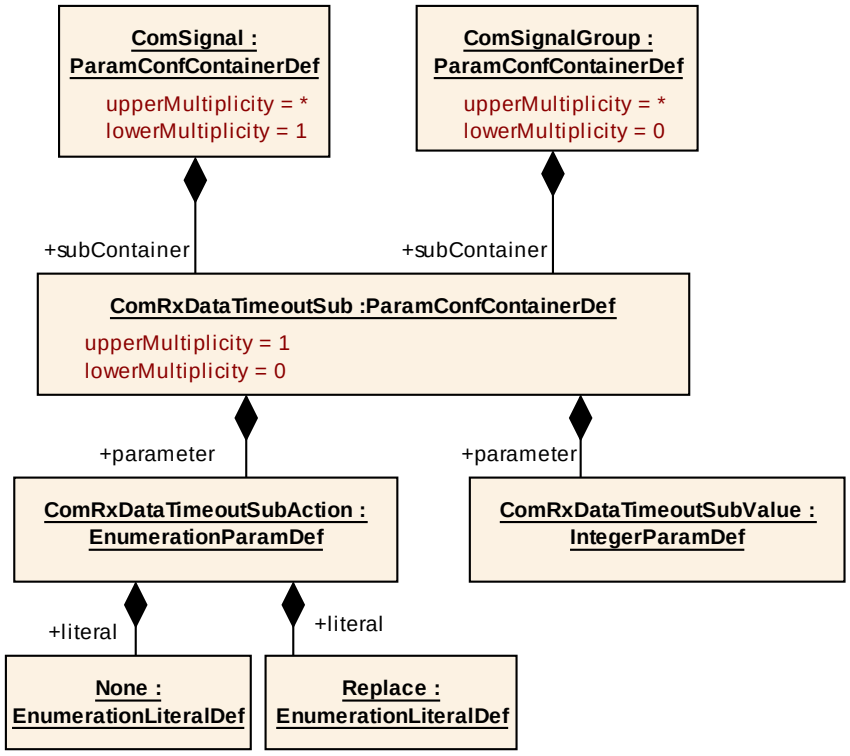


Figure 4.14: ComRxDataTimeoutSub configuration parameters

ComRxDataTimeoutSub

SWS Item	[COM394, COM416]		
Container Name	ComRxDataTimeoutSub {COM_RX_DATA_TIMEOUT_SUBSTITUTION}		
Description	Contains the configuration parameters of Com data substitution items if an Rx timeout violation occurs. This container is used only for reception of signals.		
Configuration Parameters			

Name last modified in: V1.0.0	ComRxDataTimeoutSubAction {COM_RX_DATA_TIMEOUT_SUBSTITUTION_ACTION} [COM412]		
Description	This parameter defines the action performed upon a reception timeout violation.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	None		
	Replace		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComRxDataTimeoutSubValue {COM_RX_DATA_TIMEOUT_SUBSTITUTION_VALUE} [COM413]		
Description	When the reception deadline monitoring timeout is violated the value of the related signal is according to the ComRxDataTimeoutSubstitutionAction substituted by this value.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			
No Included Containers			

4.2.2.4 ComTxDataInvalid and ComRxDataInvalid

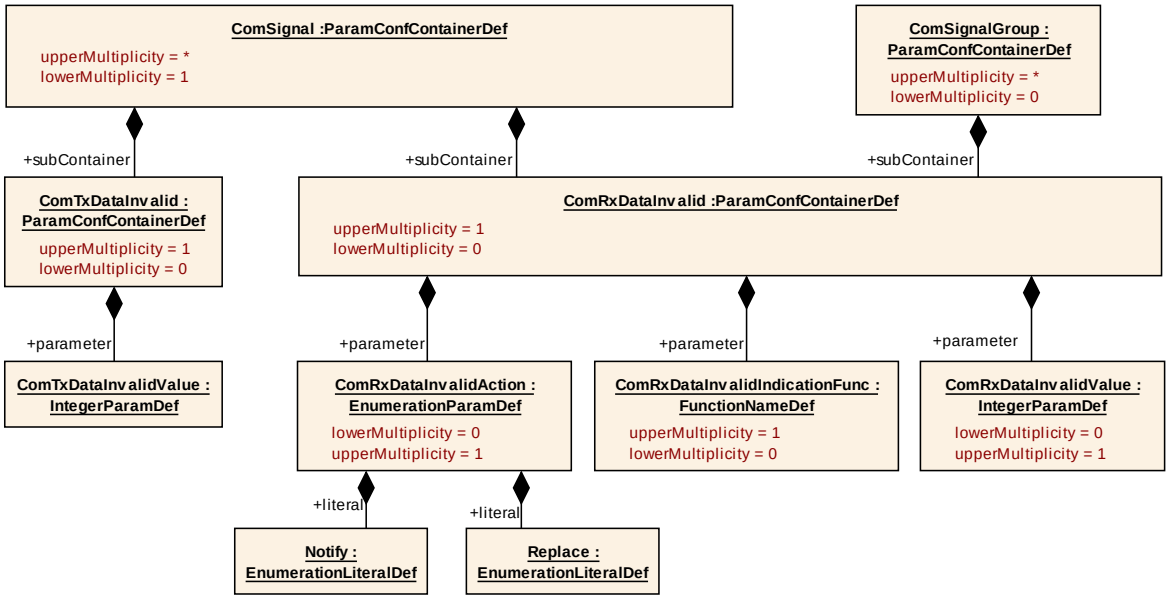


Figure 4.15: ComTxDataInvalid and ComRxDataInvalid configuration parameters

ComTxDataInvalid

SWS Item	[COM446, COM447]		
Container Name	ComTxDataInvalid {COM_TX_DATA_INVALID}		
Description	Contains the configuration parameters of Com data invalid items. This container is used for transmission of signals.		
Configuration Parameters			
Name last modified in: V1.0.0	ComTxDataInvalidValue {COM_TX_DATA_INVALID_VALUE} [COM316]		
Description	The value will be set on sender side by a Com_InvalidateSignal() and Com_InvalidateShadowSignal() call.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			
No Included Containers			

ComRxDataInvalid

SWS Item	[COM410]		
Container Name	ComRxDataInvalid {COM_RX_DATA_INVALID}		
Description	Contains the configuration parameters of Com data invalid items. This container is used for reception of signals.		
Configuration Parameters			

Name last modified in: V1.0.0	ComRxDataInvalidAction {COM_RX_DATA_INVALID_ACTION} [COM314]		
Description	This parameter defines the action performed upon reception of an invalid signal. If Replace is used the initValue will be used for the replacement.		
Multiplicity	0..1		
Type	EnumerationParamDef		
Range	Notify		
	Replace		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComRxDataInvalidIndicationFunc {COM_RX_DATA_INVALID_INDICATION_FUNCTION} [COM315]		
Description	Name of the function which notifies the RTE about the reception of an invalidated signal. Only applicable if COM_DATA_INVALID_ACTION = Notify		
Multiplicity	0..1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComRxDataInvalidValue {COM_RX_DATA_INVALID_VALUE} [COM391]		
Description	When this value is received it is recognized as the invalid value and the appropriate invalid action (as specified by COM_DATA_INVALID_ACTION) is performed.		
Multiplicity	0..1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			
No Included Containers			

4.2.2.5 ComSignalRouteDest and ComSignalInternalDest

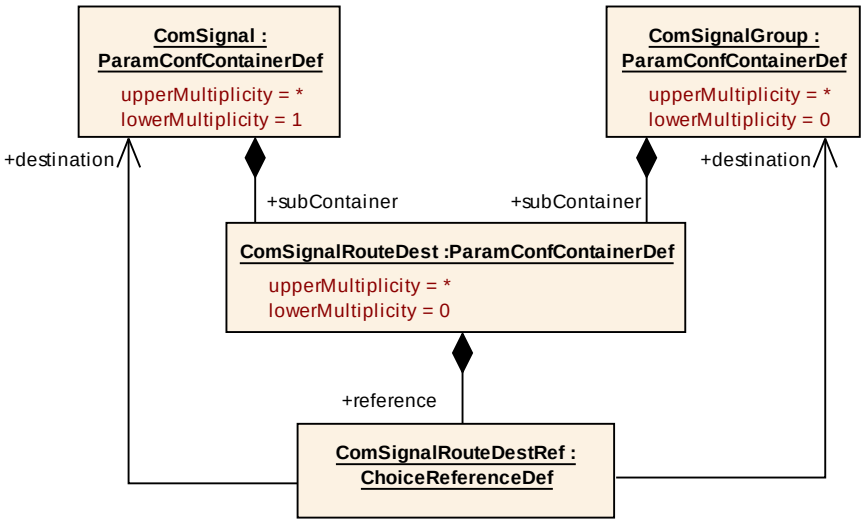


Figure 4.16: ComSignalRouteDest configuration parameters

ComSignalRouteDest

SWS Item	[COM382, COM415]		
Container Name	ComSignalRouteDest {COM_SIGNAL_ROUTE_DEST}		
Description	Reference to a signal as destination of a signal gateway relation.		
Configuration Parameters			
Name last modified in: V1.0.0	ComSignalRouteDestRef {COM_SIGNAL_ROUTE_DEST_REF} [COM355]		
Description	Name of a signal or signal group, which is destination of signal gateway relation.		
Multiplicity	1		
Type	Choice Reference to [ComSignal, ComSignalGroup]		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration)
	Post-build time	X	Variant3 (Post-build Configuration)
Scope / Dependency			
No Included Containers			

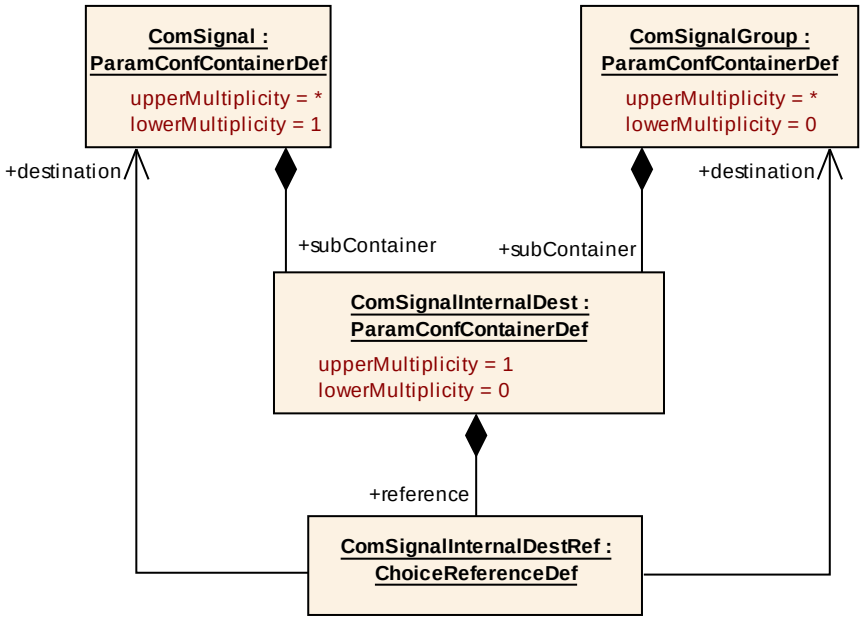


Figure 4.17: ComSignalInternalDest configuration parameters

ComSignalInternalDest

SWS Item	[COM440, COM441, COM456]		
Container Name	ComSignalInternalDest {COM_SIGNAL_INTERNAL_DEST}		
Description	Reference to a signal/signal group as internal destination.		
Configuration Parameters			
Name last modified in: V1.0.0	ComSignalInternalDestRef {COM_SIGNAL_INTERNAL_DEST_REF} [COM457]		
Description	Name of a signal or signal group, which is internal destination of the signal or signal group.		
Multiplicity	1		
Type	Choice Reference to [ComSignalGroup, ComSignal]		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration)
	Post-build time	X	Variant3 (Post-build Configuration)
Scope / Dependency			
No Included Containers			

4.2.3 ComSignalGroup

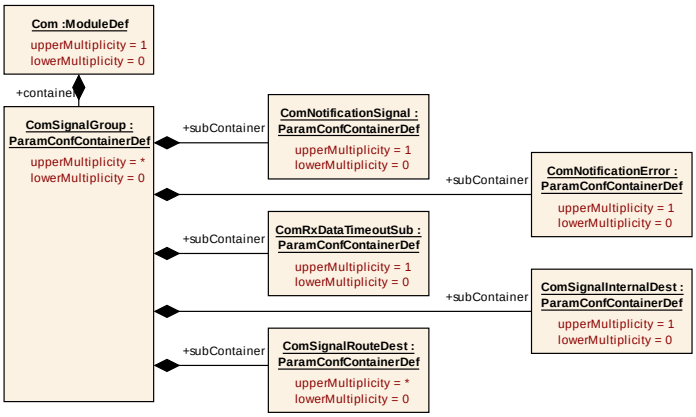


Figure 4.18: ComSignalGroup configuration containers

Additional containers, which the ComSignalGroup configuration parameter may hold, are shown in figure 4.18. The description of these containers can be found in the following sections:

- ComNotificationError in section 4.2.2.2
- ComNotificationSignal in section 4.2.2.2
- ComRxDataTimeoutSub in section 4.2.2.3
- ComSignalInternalDest in section 4.2.2.5
- ComSignalRouteDest in section 4.2.2.5

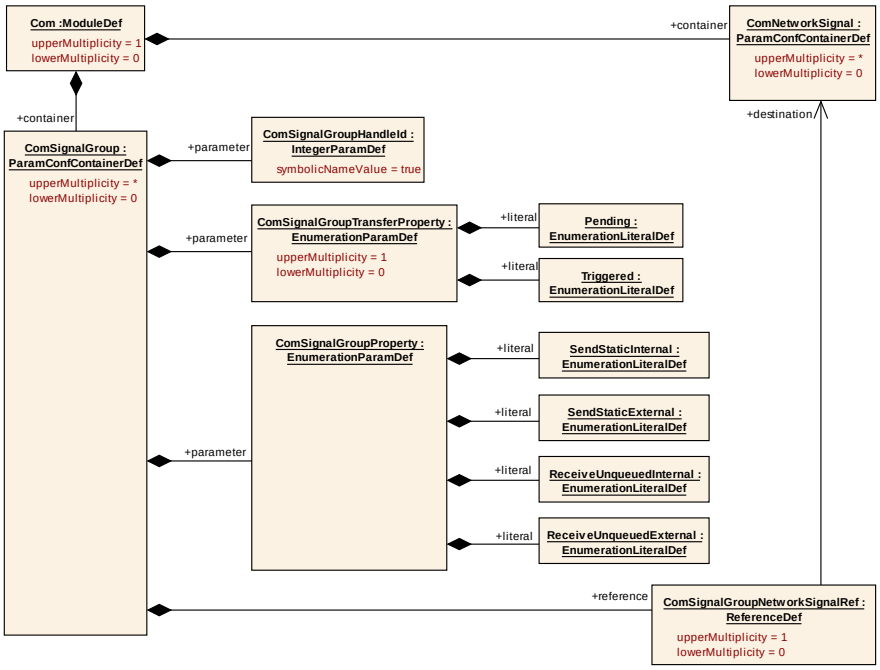


Figure 4.19: ComSignalGroup configuration parameters

ComSignalGroup

SWS Item	[COM345, COM408]		
Container Name	ComSignalGroup {COM_SIGNAL_GROUP}		
Description	Contains the configuration parameters of Com signal groups. The shortName is used as the symbolic name of the signal group (COM_SIGNAL_GROUP_NAME). This name is also used as the handle name for the signal group. This parameter is only stored in the XML file, and must not be used within the implementation.		
Configuration Parameters			

Name last modified in: V1.0.0	ComSignalGroupHandleId {COM_SIGNAL_GROUP_HANDLE_ID} [COM149]		
Description	The numerical value used as the ID of this signal group. The Com_SignalGroupHandleId is required by the Com_SendSignalGroup() and Com_ReceiveSignalGroup() calls.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComSignalGroupNetworkSignalRef {COM_SIGNAL_GROUP_NETWORK_SIGNAL_REF} [COM152]		
Description	Network signal that contains this signal group.		
Multiplicity	0..1		
Type	Reference to ComNetworkSignal		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComSignalGroupProperty {COM_SIGNAL_GROUP_PROPERTY} [COM349]		
Description	Signal property definition according to section MESSAGEPROPERTY name 3.2.10.1 in the [STD_OSEK_OIL] specification.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	ReceiveUnqueuedExternal		
	ReceiveUnqueuedInternal		
	SendStaticExternal		
	SendStaticInternal		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComSignalGroupTransferProperty {COM_SIGNAL_GROUP_TRANSFER_PROPERTY} [COM350]		
Description	Derived from [STD_OSEK_OIL].		
Multiplicity	0..1		
Type	EnumerationParamDef		
Range	Pending		
	Triggered		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration)
	Post-build time	X	Variant3 (Post-build Configuration)
Scope / Dependency			

Included Containers

Container Name	Multiplicity	Scope / Depedency
ComNotificationError	0..1	
ComNotificationSignal	0..1	
ComRxDataInvalid	0..1	
ComRxDataTimeoutSub	0..1	
ComSignalInternalDest	0..1	
ComSignalRouteDest	0..*	

4.2.4 ComNetworkSignal

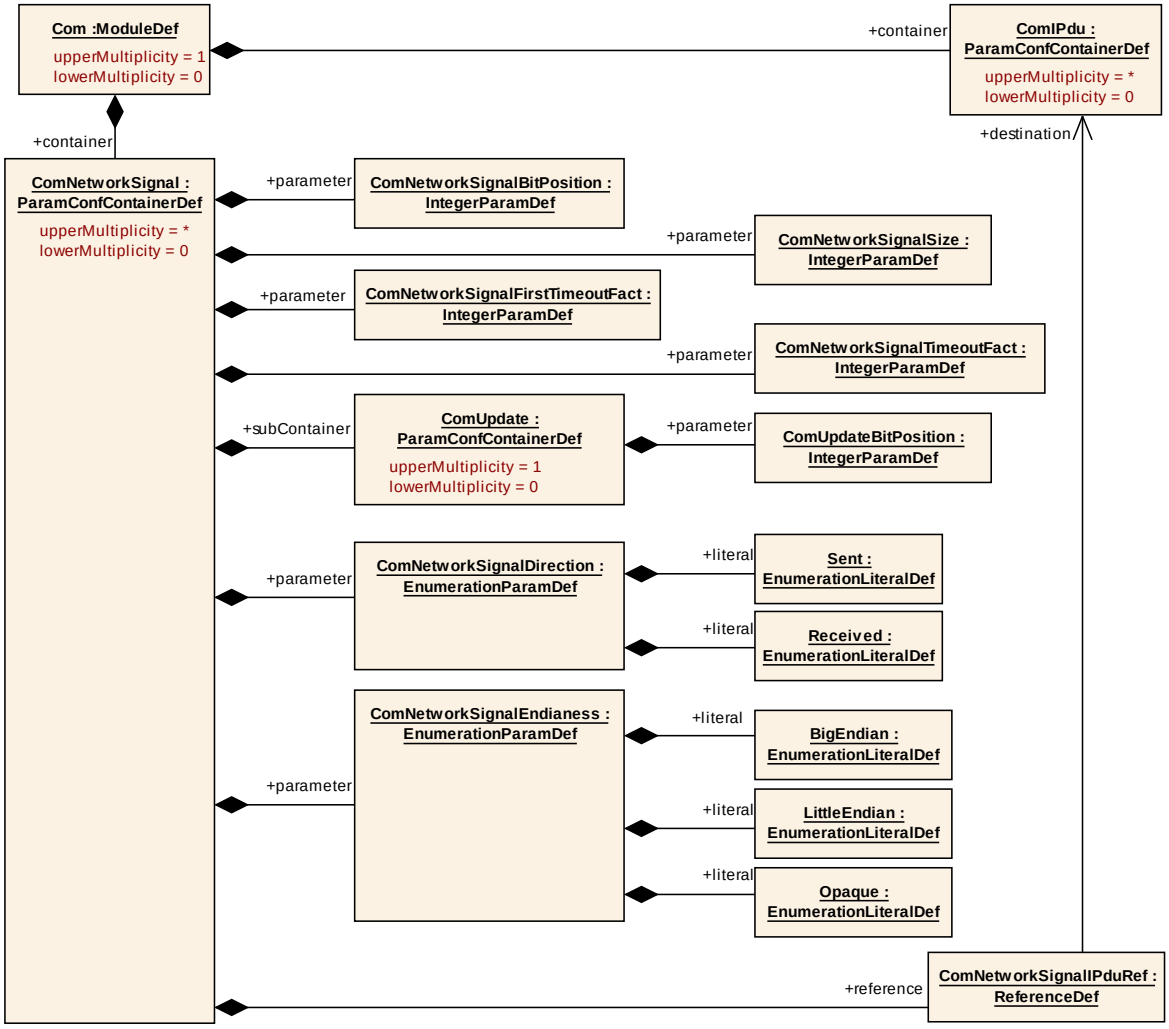


Figure 4.20: ComNetworkSignal configuration parameters

ComNetworkSignal

SWS Item	[COM342, COM406, COM156]		
Container Name	ComNetworkSignal		
Description	Contains the configuration parameters of Com network signals. The shortName of this container holds the ”COM_NETWORK_SIGNAL_NAME” [COM156]. The symbolic name of the network signal. This parameter is only stored in the XML file, and must not be used within the implementation.		
Configuration Parameters			

Name last modified in: V1.0.0	ComNetworkSignalBitPosition {COM_NETWORK_SIGNAL_BIT_POSITION} [COM259]		
Description	Starting position in the I-PDU of the network signal. If the network signal refers to a signal inside a signal group, this parameter still refers to the position in the I-PDU and not in the shadow buffer.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration)
	Post-build time	X	Variant3 (Post-build Configuration)
Scope / Dependency			

Name last modified in: V1.0.0	ComNetworkSignalDirection {COM_NETWORK_SIGNAL_DIRECTION} [COM160]		
Description	Describes whether this is a received or a sent network signal.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	Received		
	Sent		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComNetworkSignalEndianess {COM_NETWORK_SIGNAL_ENDIANESS} [COM157]		
Description	Defines the endianness of the network signal.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	BigEndian		
	LittleEndian		
	Opaque		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComNetworkSignalFirstTimeoutFact {COM_NETWORK_SIGNAL_FIRST_TIMEOUT} [COM183]		
Description	Defines the first timeout period for the deadline monitoring. Details can be found in [STD_OSEK_COM]. Note:See also COM263 for the configuration of the remaining timeout periods. Although this value is of size uint32 certain implementations may only use 16 bit counters and, therefore, some values may not always be possible.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComNetworkSignalIPduRef {COM_NETWORK_SIGNAL_IPDU_REF} [COM161]		
Description	Reference to the I-PDU that contains this signal or signal group.		
Multiplicity	1		
Type	Reference to ComIPdu		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration)
	Post-build time	X	Variant3 (Post-build Configuration)
Scope / Dependency			

Name last modified in: V1.0.0	ComNetworkSignalSize {COM_NETWORK_SIGNAL_SIZE} [COM158]		
Description	Size of the network signal in bits.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComNetworkSignalTimeoutFact {COM_NETWORK_SIGNAL_TIMEOUT} [COM263]		
Description	Defines the timeout period for the deadline monitoring. Details can be found in [STD_OSEK_COM]. Note: The period for the first timeout could differ from the TIMEOUTAlthough this value is specified as a uint32 certain implementations may only use 16 bit counters and, therefore, some values may not always be possible. COM264: If deadline monitoring is used on a signal with an update bit this defines the timeout for deadline monitoring. COM333: If the timeout is configured to 0 than no timeout monitoring shall take place		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
ComUpdate	0..1	

ComUpdate

SWS Item	[COM267, COM429]
Container Name	ComUpdate {COM_UPDATE}
Description	This container contains the configuration parameters of COM Update items.
Configuration Parameters	

Name last modified in: V1.0.0	ComUpdateBitPosition {COM_UPDATE_BIT_POSITION} [COM257]		
Description	Bit position of update bit inside I-PDU. COM258: If this signal is part of a signal group the signal must have no update bit. This setting must be consistently on sender and on receiver side. If this attribute is omitted then there is no update bit.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration)
	Post-build time	X	Variant3 (Post-build Configuration)
Scope / Dependency			
No Included Containers			

4.2.5 ComIPdu

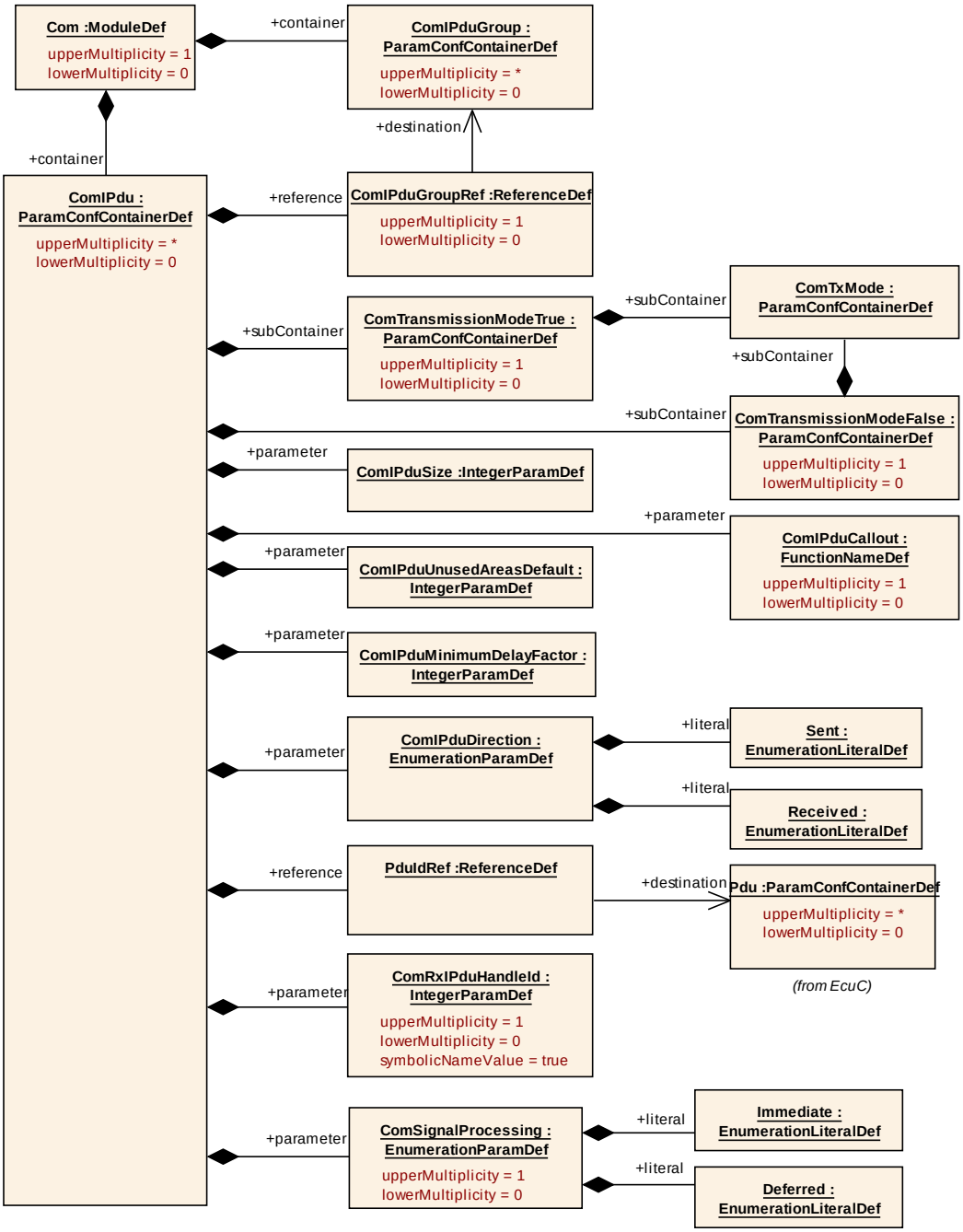


Figure 4.21: ComIPdu configuration parameters

Each `ComIPdu` has a reference `PduIdRef` to one `Pdu` element which is used to establish the relationship between the adjacent modules in the COM-Stack. For details on this concept please see section 4.1.1.

Here the `ComIPdu` can only be related to some `Pdu` which is flowing through the `PduR` module, because the `Com` module only has two interfaces, one with the `Rte` (which does not handle `Pdu`s and the other with the `PduR`.

The two interaction use-cases between Com and the PduR are described in section 4.2.5.2.

ComIPdu

SWS Item	[COM340, COM404]		
Container Name	ComIPdu {COM_IPDU}		
Description	Contains the configuration parameters of Com I-Pdus. The shortName is used as the symbolic name (COM_IPDU_NAME) of this I-Pdu when communicating with the PduR. Is optional because the Com module might be used for internal communication only. This parameter is only stored in the XML file, and must not be used within the implementation.		
Configuration Parameters			

Name last modified in: V1.0.0	ComIPduCallout {COM_IPDU_CALLOUT} [COM387]		
Description	If there is a callout defined for this I-PDU this parameter contains the name of the callout function..		
Multiplicity	0..1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComIPduDirection {COM_IPDU_DIRECTION} [COM177]		
Description	Indicates if this I-PDU is sent by AUTOSAR COM or received by AUTOSAR COM.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	Received		
	Sent		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComIPduGroupRef {COM_IPDU_GROUP_REF} [COM206]		
Description	If the I-PDU belongs to an I-PDU group, this is the reference to the IPDUGROUP. If the I-PDU does not belong to any I-PDU group, the reference does not exists.		
Multiplicity	0..1		
Type	Reference to ComIPduGroup		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration)
	Post-build time	X	Variant3 (Post-build Configuration)
Scope / Dependency			

Name last modified in: V1.0.0	ComIPduMinimumDelayFactor {COM_IPDU_MINIMUM_DELAY_TIME} [COM181]		
Description	Minimum delay between successive transmissions of this I-PDU, independent of the TRANSMISSION MODE. Although this value is of size uint32 certain implementations may only use 16 bit counters and, therefore, some values may not always be possible. There is only one minimum delay time parameter for the I-PDU. This minimum delay time does not change with mode changes. Neither is the timer reset. This means that mode changes are not allowed to violate the minimum delay time. It is not possible to monitor the MINIMUMDELAYTIME for I-PDUs that are requested using the Com_TriggerTransmit() API.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration)
	Post-build time	X	Variant3 (Post-build Configuration)
Scope / Dependency			

Name last modified in: V1.0.0	ComIPduSize {COM_IPDU_SIZE} [COM176]		
Description	The size of the I-PDU in bits. In AUTOSAR currently this is limited to 64 bits.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration)
	Post-build time	X	Variant3 (Post-build Configuration)
Scope / Dependency			

Name last modified in: V1.0.0	ComIPduUnusedAreasDefault {COM_IPDU_UNUSED_AREAS_DEFAULT} [COM017]		
Description	AUTOSAR COM fills not used areas of an I-PDU with this bit-pattern. This attribute is mandatory to avoid undefined behaviour.This byte-pattern will be repeated throughout the I-PDU.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComRxIPduHandleId {COM_IPDU_HANDLE_ID} [COM175]		
Description	The numerical value used as the ID of this I-PDU. The Com_IPduHandleId is required by the API calls to receive I-PDUs from the PduR. It is only present for I-PDU is received from the PduR, because Com is the starting module for Tx I-PDUs and there is no need to define IDs for Tx I-PDUs in the Com module.		
Multiplicity	0..1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComSignalProcessing {COM_IPDU_SIGNAL_PROCESSING} [COM119]		
Description	For the definition of the two modes Immidiate and deffered, see COM298.		
Multiplicity	0..1		
Type	EnumerationParamDef		
Range	Deferred		
	Immediate		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration)
	Post-build time	X	Variant3 (Post-build Configuration)
Scope / Dependency			

Name last modified in: V1.0.0	PduIdRef		
Description	Reference to the ”global” Pdu structure to allow harmonization of handle IDs in the COM-Stack.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			

Included Containers

Container Name	Multiplicity	Scope / Depedency
ComTransmissionMode False	0..1	
ComTransmissionMode True	0..1	

4.2.5.1 ComTransmissionMode

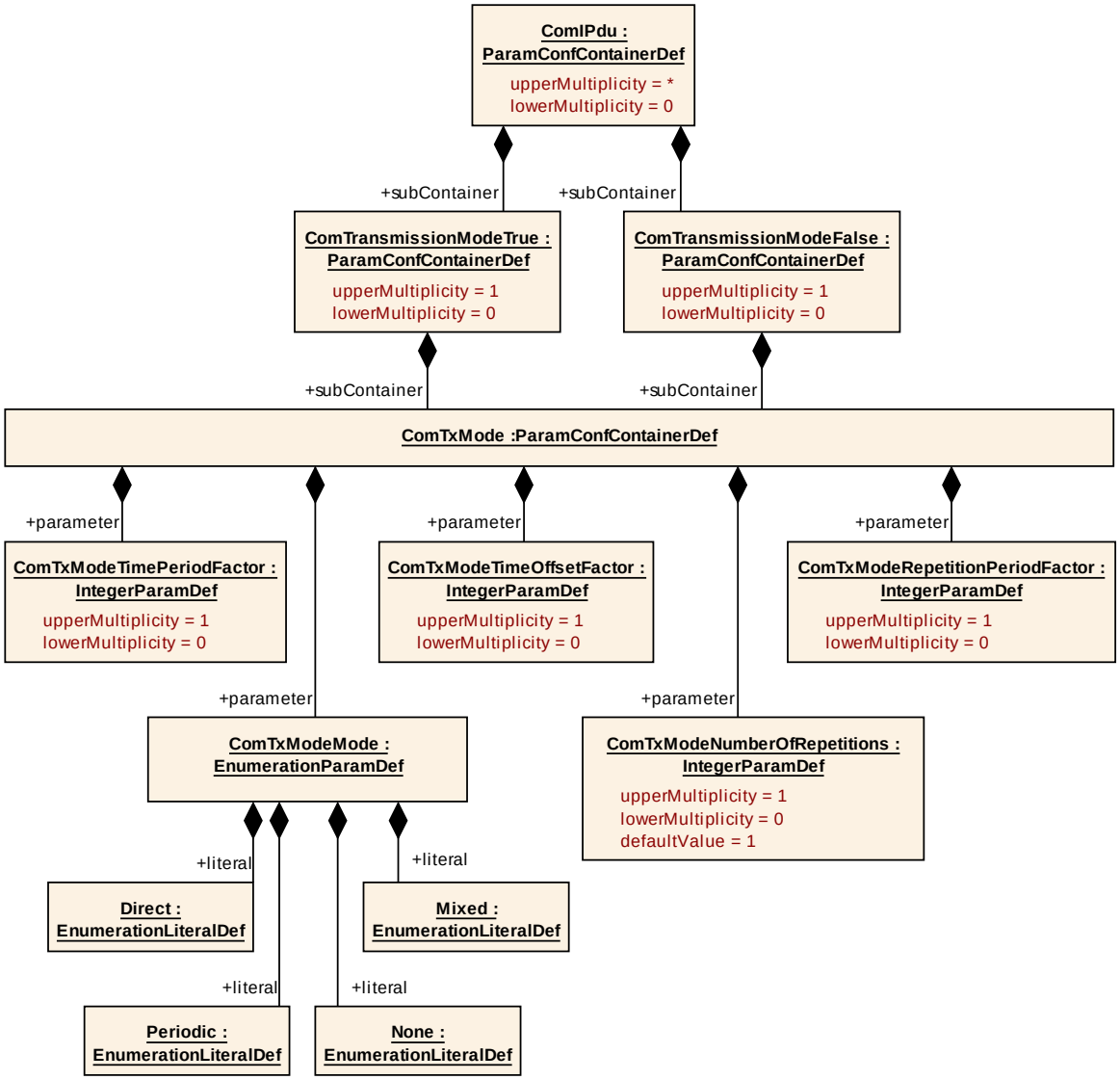


Figure 4.22: ComTransmissionModeTrue and -False configuration parameters

ComTransmissionModeTrue

SWS Item	[COM233, COM455]	
Container Name	ComTransmissionModeTrue {COM_TRANSMISSION_MODE_TRUE}	
Description	This container contains the configuration parameters of COM Transmission Modes in the case the COM_FILTER evaluates to true.	
Configuration Parameters Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
ComTxMode	1	

ComTransmissionModeFalse

SWS Item	[COM234, COM454]	
Container Name	ComTransmissionModeFalse {COM_TRANSMISSION_MODE_FALSE}	
Description	This container contains the configuration parameters of COM Transmission Modes in the case the COM_FILTER evaluates to false.	
Configuration Parameters Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
ComTxMode	1	

ComTxMode

SWS Item	[COM351]		
Container Name	ComTxMode {COM_TRANSMISSION_MODE}		
Description	This container contains the configuration parameters of COM Transmission Modes.		
Configuration Parameters			
Name last modified in: V1.0.0	ComTxModeMode {COM_TRANSMISSION_MODE_MODE} [COM137]		
Description	The available TRANSMISSION MODEs described in [STD_OSEK_OIL] shall be extended by the additional mode ”none”. The TRANSMISSION MODE ”none” shall not have any further sub-attributes in the TRANSMISSIONMODE object.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	Direct		
	Mixed		
	None		
	Periodic		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration)
	Post-build time	X	Variant3 (Post-build Configuration)
Scope / Dependency			

Name last modified in: V1.0.0	ComTxModeNumberOfRepetitions {COM_TRANSMISSION_MODE_NUMBER_OF_REPETITIONS} [COM281]		
Description	Defines the number of repetitions for the TRANSMISSION MODE "direct/n-times" and the event driven part of "mixed". Default for this parameter shall be 1.		
Multiplicity	0..1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration)
	Post-build time	X	Variant3 (Post-build Configuration)
Scope / Dependency			

Name last modified in: V1.0.0	ComTxModeRepetitionPeriodFactor {COM_TRANSMISSION_MODE_REPETITION_PERIOD} [COM282]		
Description	Period of the repetition of the n transmission for the TRANSMISSION MODE "direct/n-times" and the event driven part of "mixed". Although this value is specified as a uint32 certain implementations may only use 16 bit counters. Therefore the range is ECU and implementation dependant.		
Multiplicity	0..1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration)
	Post-build time	X	Variant3 (Post-build Configuration)
Scope / Dependency			

Name last modified in: V1.0.0	ComTxModeTimeOffsetFactor {COM_TRANSMISSION_MODE_TIME_OFFSET} [COM180]		
Description	Time until first transmission of this I-PDU. Although this value is specified as a uint32 certain implementations may only use 16 bit counters and, therefore, some values may not always be possible. When REPETITION is 0, TIMEOFFSET is the time between Com_IpduGroupStart() and the first transmission request for the IPDU. When REPETITION is >0 then TIMEOFFSET is ignored.		
Multiplicity	0..1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration)
	Post-build time	X	Variant3 (Post-build Configuration)
Scope / Dependency			

Name last modified in: V1.0.0	ComTxModeTimePeriodFactor {COM_TRANSMISSION_MODE_TIME_PERIOD} [COM178]		
Description	Period of the repetition of cyclic transmissions. Although this value is specified as a uint32 certain implementations may only use 16 bit counters. Therefore the range is ECU and implementation dependant.		
Multiplicity	0..1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration)
	Post-build time	X	Variant3 (Post-build Configuration)
Scope / Dependency **No Included Containers**			

4.2.5.2 Interaction Com and PduR

In the interaction between the `Com` module and the `PduR` module the transmission and reception use-cases have to be considered. The supporting parameter structure is shown in figure 4.23.

In case of transmission the `Com` module uses the `PduR_ComTransmit( ComTxPduId, *PduInfoPtr )` API to request the transmission of the `Pdu` the pointer is pointing to. The `ComTxPduId` is a handle ID (see also section 4.1.1) which is managed by the `PduR` module's configuration.

The actual value of the `ComTxPduId` can be found in the `HandleId` parameter which is part of the `PduRSrcPdu` container of the `PduR` module configuration information in the container `PduRRoutingPath`, see also section 4.3.2.

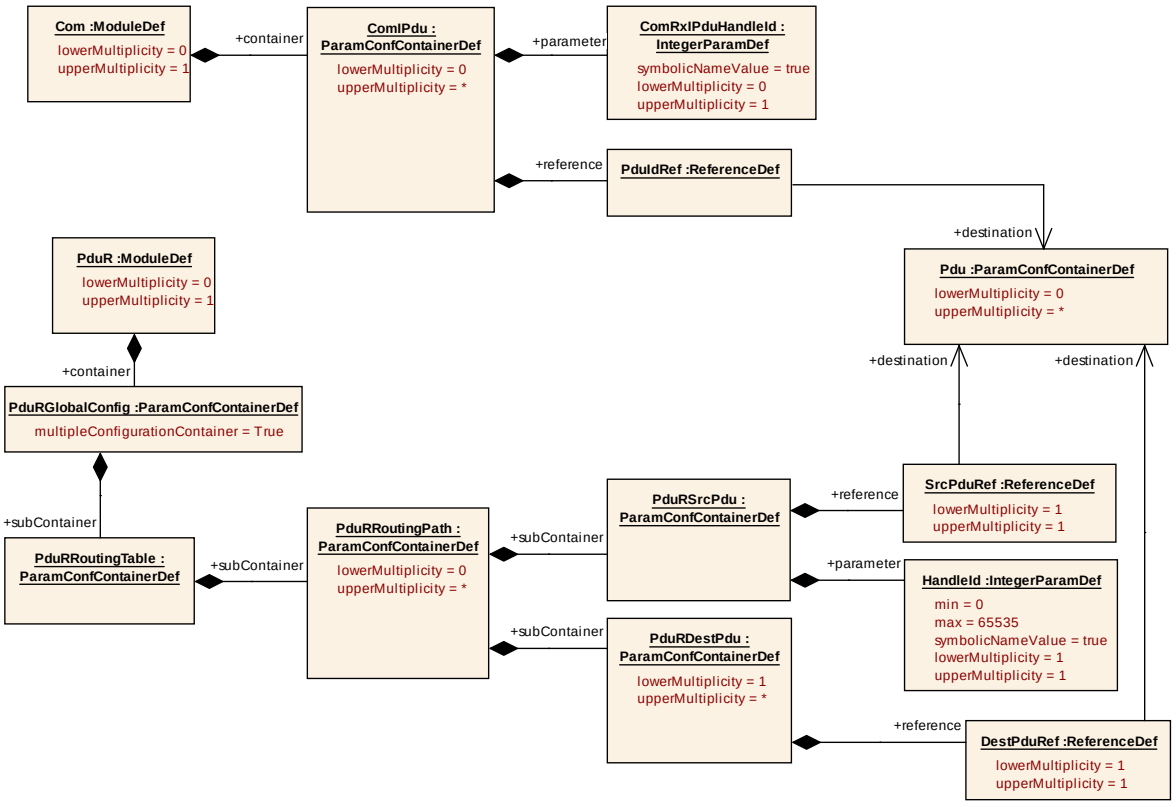


Figure 4.23: Configuration interaction between Com and PduR

In case of reception the PduR module indicates to the Com module via the Com_RxIndication(ComRxPduId, *SduPtr) API that a Pdu has been received. In this case the value for the ComRxPduId is managed by the Com module and therefore is stored in the configuration information of the Com module, ComRxIPduHandleId in the container ComIPdu, see also section 4.2.5.

4.2.6 ComIPduGroup

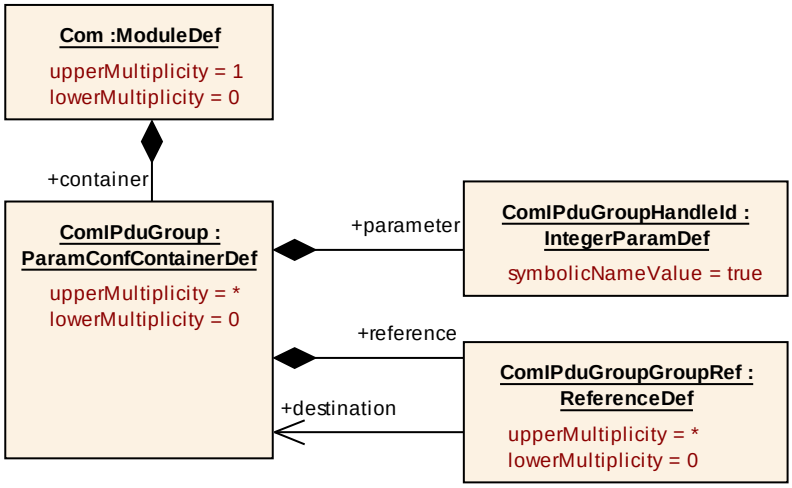


Figure 4.24: ComIPduGroup configuration parameters

ComIPduGroup

SWS Item	[COM341, COM405]
Container Name	ComIPduGroup {COM_IPDU_GROUP}
Description	Contains the configuration parameters of Com I-Pdu groups. The shortName is used as the symbolic name of the I-Pdu group (COM_IPDU_GROUP_NAME). This parameter is only stored in the XML file, and must not be used within the implementation. The minimum is 2 because we expect that the ComM addresses one RX-group and one TX-group to implement normal and listen only mode. Since a group can only have send or receive I-PDUs you could theoretically have just one group if you only send or receive I-PDUs. But we expected that this would cause problems with the ComM. Therefore the minimum was set to 2. (Bug#11347).
Configuration Parameters	

Name last modified in: V1.0.0	ComIPduGroupGroupRef {COM_IPDU_GROUP_REF} [COM185]		
Description	If the I-PDU Group belongs to an I-PDU group, this is the NAME of the IPDUGROUP it belongs to. If the I-PDU Group does not belong to another I-PDU group, the string is empty.		
Multiplicity	0..*		
Type	Reference to ComIPduGroup		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration)
	Post-build time	X	Variant3 (Post-build Configuration)
Scope / Dependency			

Name last modified in: V1.0.0	ComIPduGroupHandleId {COM_IPDU_GROUP_HANDLE_ID} [COM184]		
Description	The numerical value used as the ID of this I-PDU Group . The ComIPduGroupHandleId is required by the API calls to start and stop I-PDU Groups.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	Variant1 (Pre-compile Configuration)
	Link time	X	Variant2 (Link-time Configuration), Variant3 (Post-build Configuration)
	Post-build time	–	
Scope / Dependency			
No Included Containers			

4.3 PDU Router

The PduR (Pdu Router [34]) Parameters are structured according to figure 4.25 and will be described in detail in the following sections:

- PduRGeneral in section 4.3.1
- PduRRoutingTable in section 4.3.2
- PduRTpBufferTable in section 4.3.4
- PduRTxBufferTable in section 4.3.3

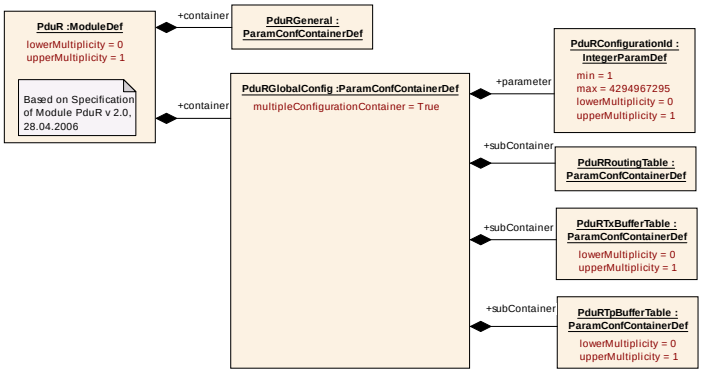


Figure 4.25: PduR container overview

PduR

Module Name	PduR	
Module Description	Configuration of the PduR (PDU Router) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
PduRGeneral	1	This container is a subcontainer of PduR and specifies the general configuration parameters of the PDU Router.
PduRGlobalConfig	1	This container contains the global configuration parameter of the PduR. It is a MultipleConfigurationContainer, i.e. this container and its sub-containers exit once per configuration set.

4.3.1 PduRGeneral

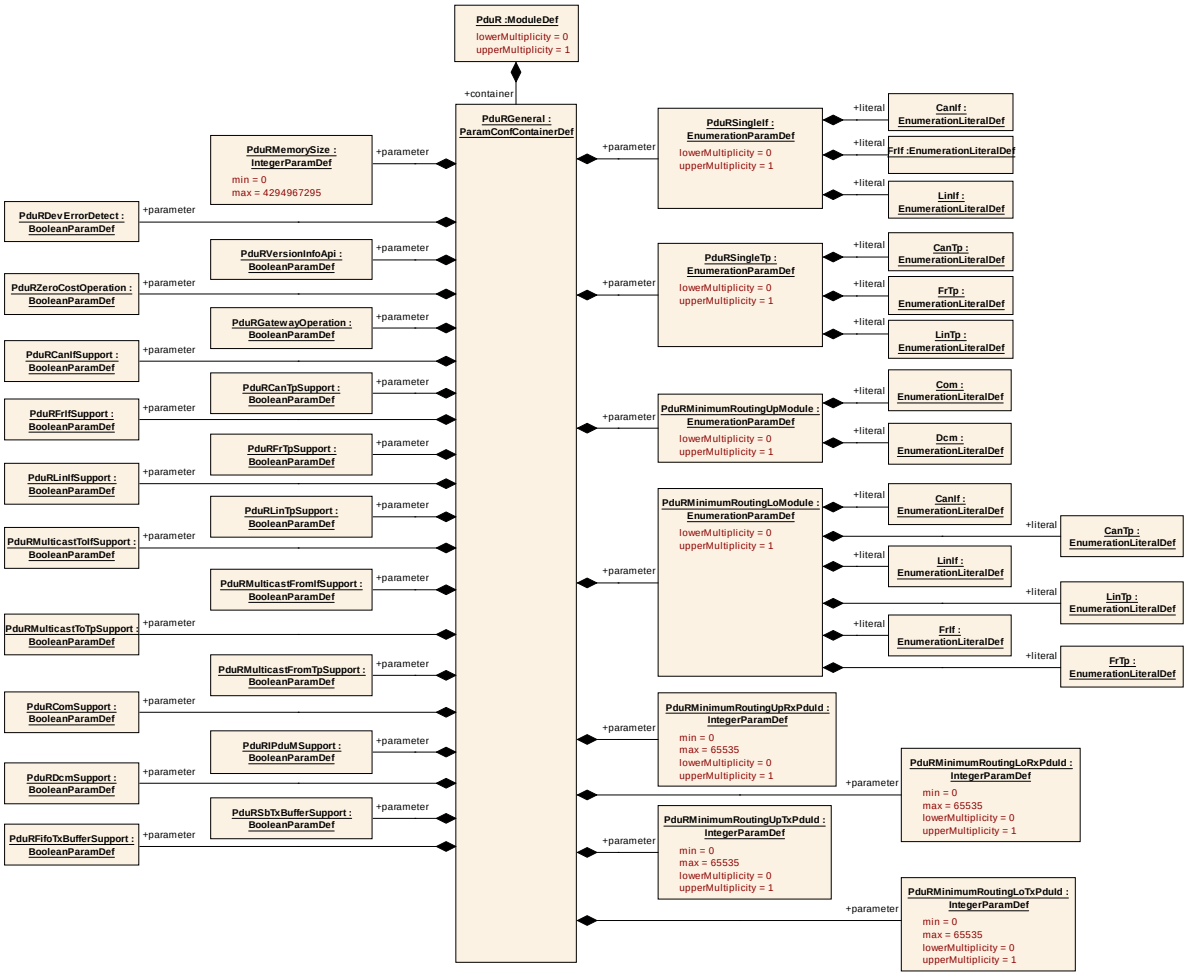


Figure 4.26: PduRGeneral configuration parameters

PduRGeneral

SWS Item			
Container Name	PduRGeneral		
Description	This container is a subcontainer of PduR and specifies the general configuration parameters of the PDU Router.		
Configuration Parameters			
Name last modified in: V1.0.0	PduRCanIfSupport {PDUR_CANIF_SUPPORT}		
Description	Configuration parameter to enable or disable PDU Router support for CAN interface.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
Name last modified in: V1.0.0	PduRCanTpSupport {PDUR_CANTP_SUPPORT}		
Description	Configuration parameter to enable or disable PDU Router support for CAN TP.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
Name last modified in: V1.0.0	PduRComSupport {PDUR_COM_SUPPORT}		
Description	Configuration parameter to enable or disable PDU Router support for COM.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
Name last modified in: V1.0.0	PduRDcmSupport {PDUR_DCM_SUPPORT}		
Description	Configuration parameter to enable or disable PDU Router support for DCM.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
Name last modified in: V1.0.0	PduRDevErrorDetect {PDUR_DEV_ERROR_DETECT}		
Description	Switches the Development Error Detection and Notification ON or OFF.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	PduRFifoTxBufferSupport {PDUR_FIFO_TX_BUFFER_SUPPORT}		
Description	Configuration parameter to enable or disable PDU Router support for FIFOs as PDU transmit buffers; if PDUR_GATEWAY_OPERATION is disabled, this parameter has to be disabled.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	PduRFrIfSupport {PDUR_FRIF_SUPPORT}		
Description	Configuration parameter to enable or disable PDU Router support for FlexRay interface.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	PduRFrTpSupport {PDUR_FRTP_SUPPORT}		
Description	Configuration parameter to enable or disable PDU Router support for FlexRay TP.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	PduRGatewayOperation {PDUR_GATEWAY_OPERATION}		
Description	Configuration parameter to enable or disable PDU Router gateway operation; if PDUR_ZERO_COST_OPERATION is enabled, this parameter has to be disabled.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	PduRIPduMSupport {PDUR_IPDUM_SUPPORT}		
Description	Configuration parameter to enable or disable PDU Router support for IPDUM; if PDUR_ZERO_COST_OPERATION is enabled, this parameter has to be disabled.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	PduRLinIfSupport {PDUR_LINIF_SUPPORT}		
Description	Configuration parameter to enable or disable PDU Router support for LIN interface.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	PduRLinTpSupport {PDUR_LINTP_SUPPORT}		
Description	Configuration parameter to enable or disable PDU Router support for LIN TP.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	PduRMemorySize {PDUR_MEMORY_SIZE}		
Description	Memory size reserved for PDU Router buffers. Only required for gateway operation.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 4294967295		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	PduRMinimumRoutingLoModule {PDUR_MINIMUM_ROUTING_LO_MODULE}		
Description	Lower layer module to be used for minimum routing; this parameter shall be used if PDUR_ZERO_COST_OPERATION is disabled; otherwise it shall not be used.		
Multiplicity	0..1		
Type	EnumerationParamDef		
Range	CanIf	Can Interface	
	CanTp	Can TP	
	FrIf	FlexRay Interface	
	FrTp	FlexRay TP	
	LinIf	Lin Interface	
	LinTp	Lin TP	
Configuration Class	**Pre-compile time**	X	Variant 2: pre-compile and post-build
	Link time	X	Variant 3: pre-compile, link and post-build
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	PduRMinimumRoutingLoRxPduId {PDUR_MINIMUM_ROUTING_LO_RXPDUID}		
Description	Receive PDU identifier of the lower layer module which shall be used at the PDU Router interface to the lower layer module specified by PDUR_MINIMUM_ROUTING_LO_MODULE for minimum routing; this parameter shall be used if PDUR_ZERO_COST_OPERATION is disabled; otherwise it shall not be used.		
Multiplicity	0..1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	Variant 2: pre-compile and post-build
	Link time	X	Variant 3: pre-compile, link and post-build
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	PduRMinimumRoutingLoTxPduId {PDUR_MINIMUM_ROUTING_LO_TXPDUID}		
Description	Transmit PDU identifier of the lower layer module which shall be used at the PDU Router interface to the lower layer module specified by PDUR_MINIMUM_ROUTING_LO_MODULE for minimum routing; this parameter shall be used if PDUR_ZERO_COST_OPERATION is disabled; otherwise it shall not be used.		
Multiplicity	0..1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	Variant 2: pre-compile and post-build
	Link time	X	Variant 3: pre-compile, link and post-build
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	PduRMinimumRoutingUpModule {PDUR_MINIMUM_ROUTING_UP_MODULE}		
Description	Upper layer module to be used for minimum routing; this parameter shall be used if PDUR_ZERO_COST_OPERATION is disabled; otherwise it shall not be used.		
Multiplicity	0..1		
Type	EnumerationParamDef		
Range	Com	COM	
	Dcm	DCM	
Configuration Class	**Pre-compile time**	X	Variant 2: pre-compile and post-build
	Link time	X	Variant 3: pre-compile, link and post-build
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	PduRMinimumRoutingUpRxPduId {PDUR_MINIMUM_ROUTING_UP_RXPDUID}		
Description	Receive PDU identifier of the upper layer module which shall be used at the PDU Router interface to the upper layer module specified by PDUR_MINIMUM_ROUTING_UP_MODULE for minimum routing; this parameter shall be used if PDUR_ZERO_COST_OPERATION is disabled; otherwise it shall not be used.		
Multiplicity	0..1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	Variant 2: pre-compile and post-build
	Link time	X	Variant 3: pre-compile, link and post-build
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	PduRMinimumRoutingUpTxPduId {PDUR_MINIMUM_ROUTING_UP_TXPDUID}		
Description	Transmit PDU identifier of the upper layer module which shall be used at the PDU Router interface to the upper layer module specified by PDUR_MINIMUM_ROUTING_UP_MODULE for minimum routing; this parameter shall be used if PDUR_ZERO_COST_OPERATION is disabled; otherwise it shall not be used.		
Multiplicity	0..1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	Variant 2: pre-compile and post-build
	Link time	X	Variant 3: pre-compile, link and post-build
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	PduRMulticastFromIfSupport {PDUR_MULTICAST_FROMIF_SUPPORT}		
Description	Configuration parameter to enable or disable PDU Router support for multicasts from an interface module to upper layer modules or lower layer interface modules; if PDUR_ZERO_COST_OPERATION is enabled, this parameter has to be disabled.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	PduRMulticastFromTpSupport {PDUR_MULTICAST_FROMTP_SUPPORT}		
Description	Configuration parameter to enable or disable PDU Router support for multicasts from a TP module to upper layer modules or lower layer TP modules; if PDUR_ZERO_COST_OPERATION is enabled, this parameter has to be disabled.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	PduRMulticastToIfSupport {PDUR_MULTICAST_TOIF_SUPPORT}		
Description	Configuration parameter to enable or disable PDU Router support for multicasts from an upper layer module to interface modules; if PDUR_ZERO_COST_OPERATION is enabled, this parameter has to be disabled.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	PduRMulticastToTpSupport {PDUR_MULTICAST_TOTP_SUPPORT}		
Description	Configuration parameter to enable or disable PDU Router support for multicasts from an upper layer module to TP modules; if PDUR_ZERO_COST_OPERATION is enabled, this parameter has to be disabled.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	PduRSbTxBufferSupport {PDUR_SB_TX_BUFFER_SUPPORT}		
Description	Configuration parameter to enable or disable PDU Router support for single buffers as PDU transmit buffers; if PDUR_GATEWAY_OPERATION is disabled, this parameter has to be disabled.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	PduRSingleIf {PDUR_SINGLE_IF}		
Description	Single interface module in case zero cost operation is enabled (PDUR_ZERO_COST_OPERATION).		
Multiplicity	0..1		
Type	EnumerationParamDef		
Range	CanIf	Can interface	
	FrIf	FlexRay interface	
	LinIf	Lin interface	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	PduRSingleTp {PDUR_SINGLE_TP}		
Description	Single transport protocol module in case zero cost operation is enabled (PDUR_ZERO_COST_OPERATION).		
Multiplicity	0..1		
Type	EnumerationParamDef		
Range	CanTp	Can TP	
	FrTp	FlexRay TP	
	LinTp	Lin TP	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	PduRVersionInfoApi {PDUR_VERSION_INFO_API}		
Description	Activates/Deactivates the Version Info API.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	PduRZeroCostOperation {PDUR_ZERO_COST_OPERATION}		
Description	If all conditions stated in PDUR165 are fulfilled, all routing paths are implicitly defined and the communication modules directly above or below the PDU Router shall directly call each other without using PDU Router functions (zero cost operation). The configuration parameters PDUR_SINGLE_IF and PDUR_SINGLE_TP are used to specify the related lower layer module.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

4.3.2 PduRRoutingTable

The routing table is the major section of the PduR module's configuration. It contains detailed information on the path a Pdu takes in the PduR.

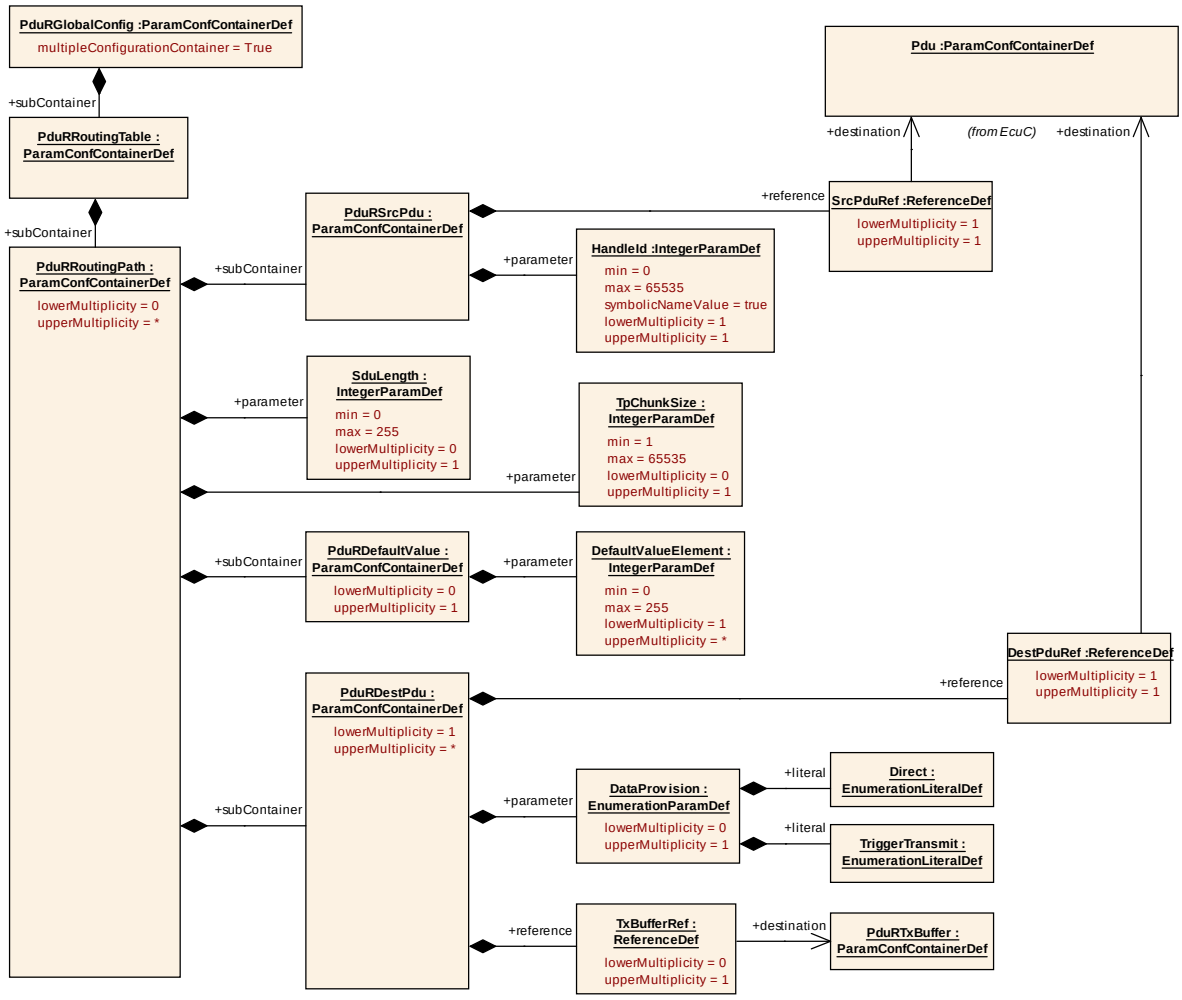


Figure 4.27: PduRRoutingTable configuration parameters

PduRRoutingTable

SWS Item	[PDUR247]	
Container Name	PduRRoutingTable	
Description	PDU Router routing table is a subcontainer ofPduR. This container shall only be considered by the PDU Router Configuration Generator if PduRGeneral/PDUR_ZERO_COST_OPERATION is disabled.	
Configuration Parameters Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
PduRRoutingPath	0..*	

PduRRoutingPath

SWS Item	[PDUR248]		
Container Name	PduRRoutingPath		
Description	This container is a subcontainer of PduRRoutingTable and specifies the routing path of a PDU.		
Configuration Parameters			

Name last modified in: V1.0.0	SduLength		
Description	Length of PDU data (SDU). Only required if a TX buffer is configured.		
Multiplicity	0..1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	L	Variant 2: pre-compile and post-build, Variant 3: pre-compile, link and post-build
Scope / Dependency			

Name last modified in: V1.0.0	TpChunkSize		
Description	Chunk size for routing on the fly. Defines the number of bytes which shall be received before transmission on the destination bus may start. Only required for TP gateway PDUs. The TpChunkSize shall not be larger than the length of the related TP Buffer.		
Multiplicity	0..1		
Type	IntegerParamDef		
Range	1 .. 65535		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	L	Variant 2: pre-compile and post-build, Variant 3: pre-compile, link and post-build
Scope / Dependency			
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
PduRDefaultValue	0..1		
PduRDestPdu	1..*		
PduRSrcPdu	1		

PduRDefaultValue

SWS Item	
Container Name	PduRDefaultValue
Description	This container is a subcontainer of PduRRoutingPath and specifies the default value of the I-PDU. Only required for gateway operation and if at least one PDU specified by PduRDestPdu uses TriggerTransmit Data provision. Represented as an array of IntegerParamDef. Each element is one Byte of the default value.
Configuration Parameters	

Name last modified in: V1.0.0	DefaultValueElement		
Description	The default value consists of a number of elements. Each element is one byte long and the number of elements is specified by SduLength. The position of this parameter in the container specifies the byte position of the element within the default value.		
Multiplicity	1..*		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	L	Variant 2: pre-compile and post-build, Variant 3: pre-compile, link and post-build
Scope / Dependency			
No Included Containers			

PduRSrcPdu

SWS Item	[PDUR288]
Container Name	PduRSrcPdu
Description	This container is a subcontainer of PduRRoutingPath and specifies the source of the PDU to be routed.
Configuration Parameters	

Name last modified in: V1.0.0	HandleId		
Description	PDU identifier assigned by PDU Router.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	Variant 1: pre-compile (zero-cost)
	Link time	–	
	Post-build time	L	Variant 2: pre-compile and post-build, Variant 3: pre-compile, link and post-build
Scope / Dependency			

Name last modified in: V1.0.0	SrcPduRef		
Description	Source PDU reference; reference to unique PDU identifier which shall be used for the requested PDU Router operation.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	X	Variant 1: pre-compile (zero-cost)
	Link time	–	
	Post-build time	L	Variant 2: pre-compile and post-build, Variant 3: pre-compile, link and post-build
Scope / Dependency			
No Included Containers			

PduRDestPdu

SWS Item	[PDUR249]		
Container Name	PduRDestPdu		
Description	This container is a subcontainer of PduRRoutingPath and specifies one destination for the PDU to be routed.		
Configuration Parameters			

Name last modified in: V1.0.0	DataProvision		
Description	Specifies how data are provided: direct (as part of the Transmit call) or via the TriggerTransmit callback function. Only required for non-TP gateway PDUs.		
Multiplicity	0..1		
Type	EnumerationParamDef		
Range	Direct	direct data provision	
	TriggerTransmit	trigger transmit data provision	
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	L	Variant 2: pre-compile and post-build, Variant 3: pre-compile, link and post-build
Scope / Dependency			

Name last modified in: V1.0.0	DestPduRef		
Description	Destination PDU reference; reference to unique PDU identifier which shall be used by the PDU Router instead of the source PDU ID when calling the related function of the destination module.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	X	Variant 1: pre-compile (zero-cost)
	Link time	–	
	Post-build time	L	Variant 2: pre-compile and post-build, Variant 3: pre-compile, link and post-build
Scope / Dependency			

Name last modified in: V1.0.0	TxBufferRef		
Description	Specifies the assigned transmit buffer. Only required for specific non-TP gateway PDUs.		
Multiplicity	0..1		
Type	Reference to PduRTxBuffer		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	L	Variant 2: pre-compile and post-build, Variant 3: pre-compile, link and post-build
Scope / Dependency			
No Included Containers			

To be able to identify each PDU's path through the COM-Stack unambiguously, each `PduRSrcPdu` shall reference one distinct `Pdu` object within the `PduCollection`. The same `Pdu` cannot be references by any other `PduRSrcPdu`.

Within a `PduRRoutingPath`, each `PduRDestPdu` shall reference one distinct `Pdu` object. The same `Pdu` cannot be referenced by any other `PduRDestPdu`.

4.3.3 PduRTxBufferTable

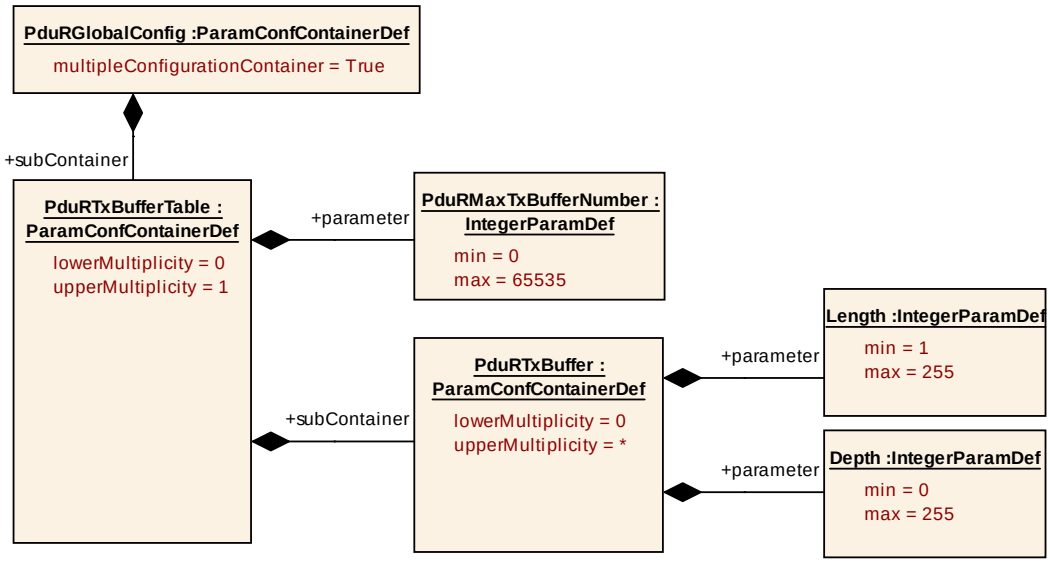


Figure 4.28: PduRTxBufferTable configuration parameters

PduRTxBufferTable

SWS Item	[PDUR243]		
Container Name	PduRTxBufferTable		
Description	This container is a subcontainer of PduR and contains the definition of all transmit buffers (used by specific non-TP PDUs; only required for PDU Router gateway operation). This container shall only be considered by the PDU Router Configuration Generator if PduRGeneral/PDUR_GATEWAY_OPERATION is enabled.		
Configuration Parameters			
Name last modified in: V1.0.0	PduRMaxTxBufferNumber {PDUR_MAX_TX_BUFFER_NUMBER}		
Description	maximum number of transmit buffers		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	Variant 2: pre-compile and post-build, Variant 3: pre-compile, link and post-build
	Link time	–	
	Post-build time	–	
Scope / Dependency			
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
PduRTxBuffer	0..*		

PduRTxBuffer

SWS Item	[PDUR244]		
Container Name	PduRTxBuffer		
Description	This container is a subcontainer of PduRTxBufferTable and specifies a transmit buffer for a non-TP PDU.		
Configuration Parameters			
Name last modified in: V1.0.0	Depth		
Description	Specifies the depth of the buffer		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	L	Variant 2: pre-compile and post-build, Variant 3: pre-compile, link and post-build
Scope / Dependency			

Name last modified in: V1.0.0	Length		
Description	Length of the buffer.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 255		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	L	Variant 2: pre-compile and post-build, Variant 3: pre-compile, link and post-build
Scope / Dependency			
No Included Containers			

4.3.4 PduRTpBufferTable

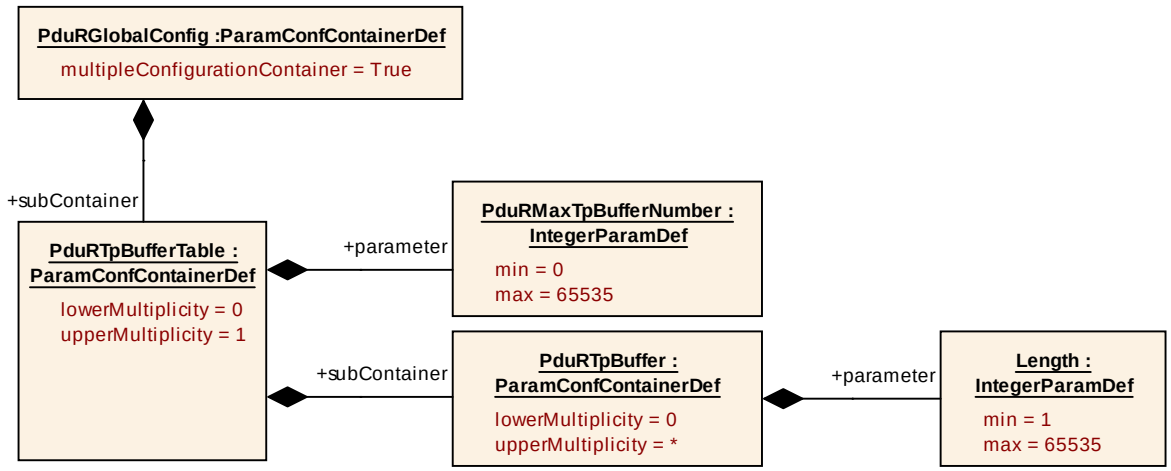


Figure 4.29: PduRTpBufferTable configuration parameters

PduRTpBufferTable

SWS Item	[PDUR245]
Container Name	PduRTpBufferTable {TpBufferTable}
Description	This container is a subcontainer of PduR and contains the definition of all TP buffers (only required for PDU Router gateway operation). This container shall only be considered by the PDU Router Configuration Generator if PduRGeneral/PDUR_GATEWAY_OPERATION is enabled.
Configuration Parameters	

Name last modified in: V1.0.0	PduRMaxTpBufferNumber {PDUR_MAX_TP_BUFFER_NUMBER}		
Description	maximum number of TP buffers.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	Variant 2: pre-compile and post-build, Variant 3: pre-compile, link and post-build
	Link time	–	
	Post-build time	–	
Scope / Dependency **Included Containers**			
Container Name	**Multiplicity**	**Scope / Depedency**	
PduRTpBuffer	0..*		

PduRTpBuffer

SWS Item	[PDUR246]		
Container Name	PduRTpBuffer		
Description	This container is a subcontainer of PduRTpBufferTable and specifies a TP buffer.		
Configuration Parameters			
Name last modified in: V1.0.0	Length		
Description	Length of the buffer.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 65535		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	L	Variant 2: pre-compile and post-build, Variant 3: pre-compile, link and post-build
Scope / Dependency **No Included Containers**			

4.3.5 Configuration examples for the Pdu Router

In this section several use-cases of the PduR are described from the configuration point of view. The focus is on the interaction of the PduR configuration with the configuration of the other COM-Stack modules. Therefore only some configuration parameters are actually shown in these examples.

4.3.5.1 Tx from Com to CanIf

In the example in figure 4.30 a Pdu is sent from the Com module – via the Pdu Router – to the Can Interface. Since this one Pdu is handed over through these layers there is only need for one global Pdu object `System_Pdu`.

The Com module's configuration points to the `System_Pdu` to indicate which Pdu shall

be sent. The actual Handle Id which has to be used in the API call will however be defined by the PduR in the parameter `PduRSrcPdu::HandleId`. In this example the Com module has to use the Hanlde Id `23` to transmit this Pdu to the PduR.

Then, since the CanIf is pointing to the same `System_Pdu` the PduR can be configured to send this Pdu to the CanIf. The Handle Id is defined in the CanIf configuration in the parameter value of `CanIfCanTxPduId`.

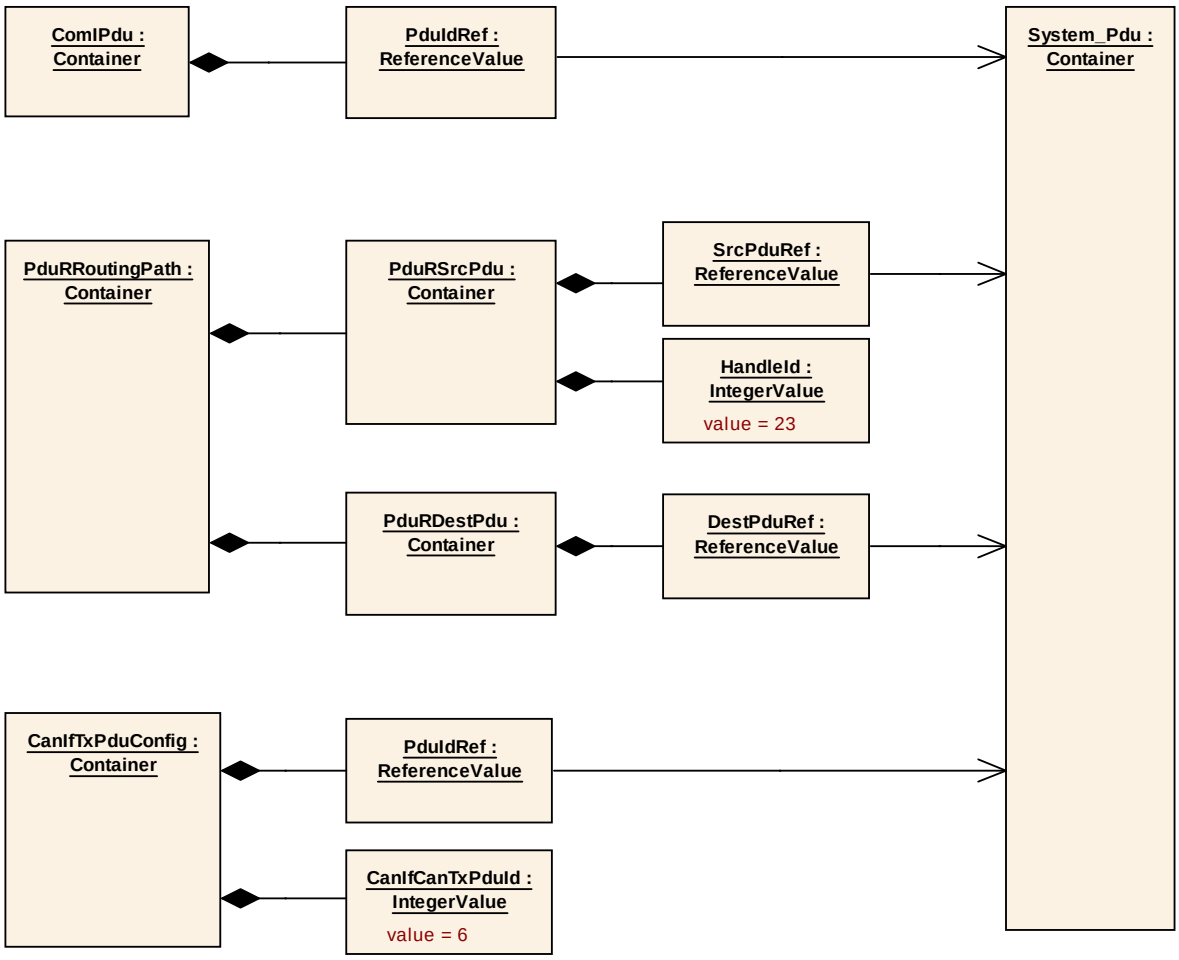


Figure 4.30: Tx from Com to CanIf example

4.3.5.2 Rx from CanIf to Com

In the example in figure 4.31 the reception use-case from the CanIf to the Com module is configured. Here the Handle Ids are defined in the PduR and the Com module's configuration.

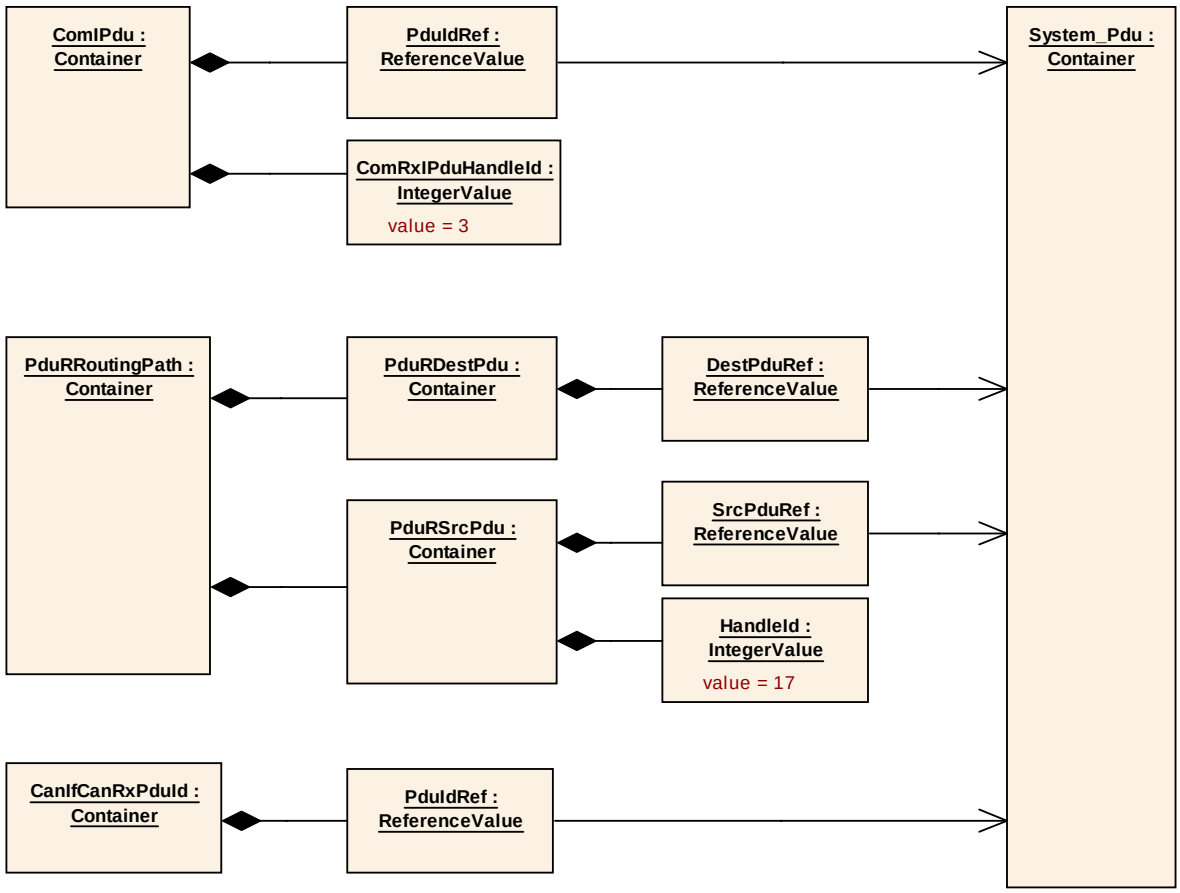


Figure 4.31: Rx from CanIf to Com example

4.3.5.3 Gateway from CanIf to FrIf

In the example in figure 4.32 the gateway use-case is shown. Since there are two Pdus involved there are two `System_Pdu` objects defined: one which is representing the Can Pdu and one which represents the Fr Pdu. Via the references to these two `System_Pdu` objects the gateway is configured.

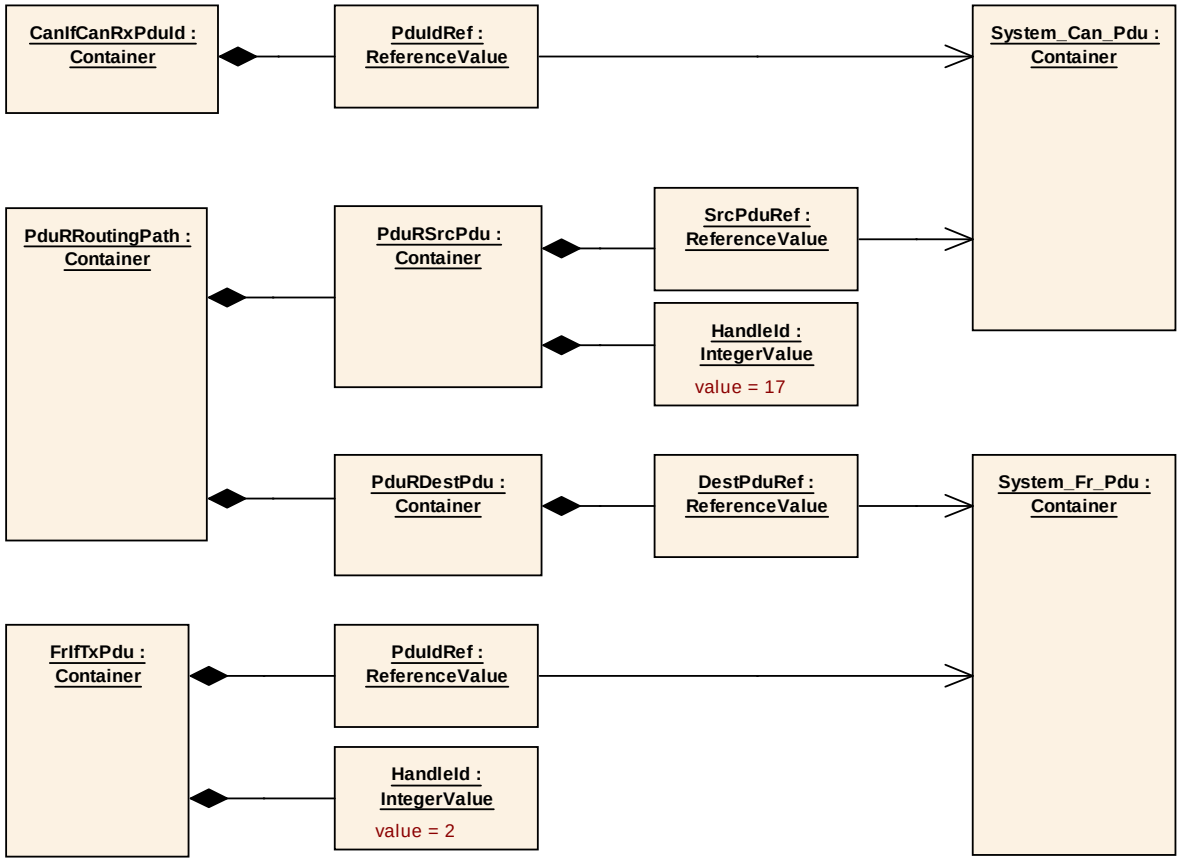


Figure 4.32: Gateway from CanIf to FrIf example

4.4 IPdu Multiplexer

Remark/30.1.2007: This section conforms R2.0

, due to the fact, that there is no R2.1 SWS available.

The IPduM (IPdu Multiplexer [20]) Parameters are structured according to figure 4.33 and will be described in detail in the following sections:

- IPduMGeneral in section 4.4.1
- IPduMTxPathway in section 4.4.2
- IPduMRxPathway in section 4.4.3

For a description of the interaction between COM-Stack modules please refer to section 4.1.

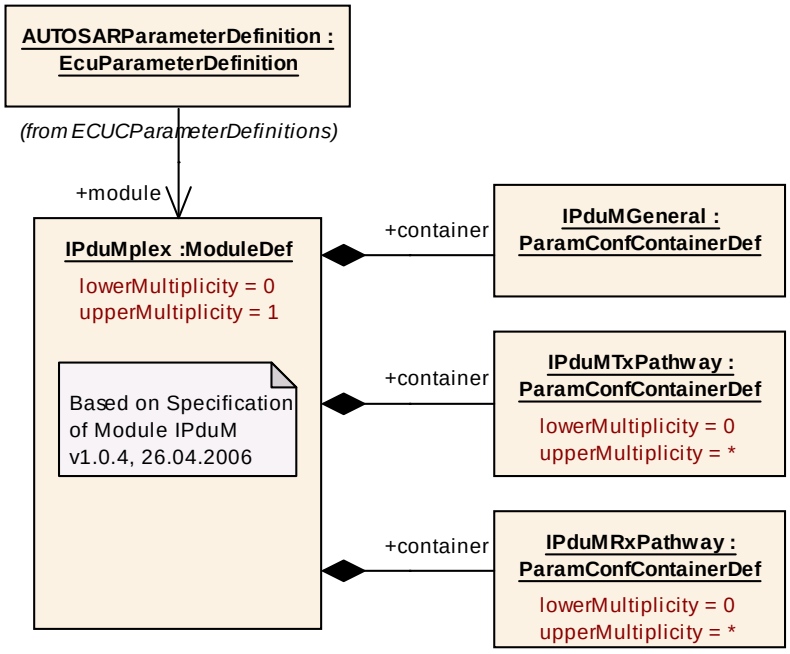


Figure 4.33: IPduMplex container overview

IPduMplex

Module Name	IPduMplex	
Module Description	Configuration of the IPduMplex (IPdu Multiplexer) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
IPduMGeneral	1	Contains the general configuration parameters of IPduMplex.

Container Name	Multiplicity	Scope / Dependency
IPduMRxPathway	0..*	Contains the configuration parameters received I-PDUs by the IPduM module.
IPduMTxPathway	0..*	Contains the configuration parameters transmitted I-PDUs by the IPduM module.

4.4.1 IPduMGeneral

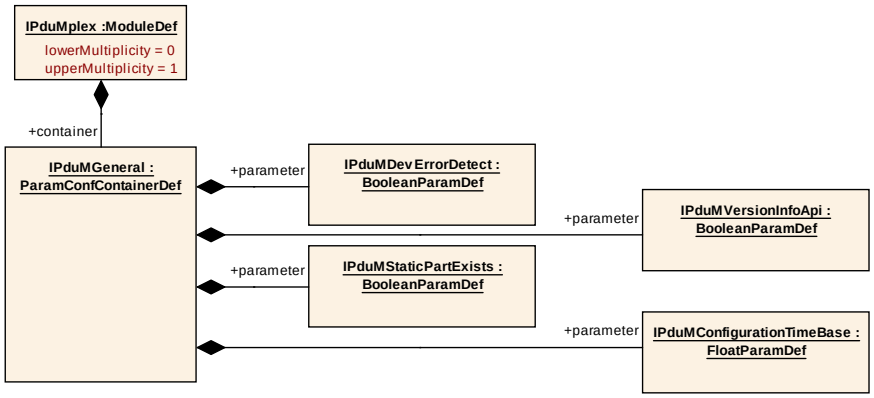


Figure 4.34: IPduMGeneral configuration parameters

IPduMGeneral

SWS Item	[IPDUM059]		
Container Name	IPduMGeneral {IPDUM_CONFIGURATION}		
Description	Contains the general configuration parameters of IPduMplex.		
Configuration Parameters			
Name last modified in: V1.0.0	IPduMConfigurationTimeBase {IPDUM_CONFIGURATION_TIME_BASE}		
Description	The period between successive ticks of AUTOSAR COM in seconds.		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	IPduMDevErrorDetect {IPDUM_DEV_ERROR_DETECT}		
Description	Active/Deactivate the detection of development errors, for production code this parameter has to be OFF.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	IPduMStaticPartExists {IPDUM_STATIC_PART_EXISTS}		
Description	This is to allow optimizations in the case the IPduM will never be used with a static part. Note that this is a pre-compile option. If this is set to NO then it will not be possible to add static parts after compilation.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	IPduMVersionInfoApi {IPDUM_VERSION_INFO_API}		
Description	Active/Deactivate the version information API.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

4.4.2 IPduMTxPathway

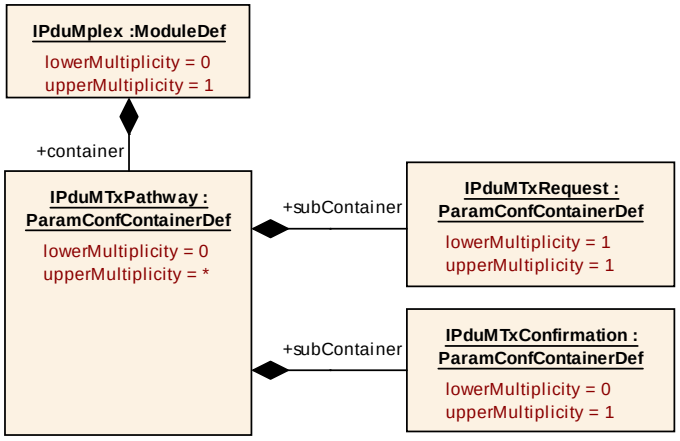


Figure 4.35: IPduMTxPathway configuration overview

IPduMTxPathway

SWS Item	[IPDUM070]	
Container Name	IPduMTxPathway {IPDUM_TX_PATHWAY}	
Description	Contains the configuration parameters transmitted I-PDUs by the IPduM module.	
Configuration Parameters Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
IPduMTxConfirmation	0..1	
IPduMTxRequest	1	

4.4.2.1 IPduMTxRequest

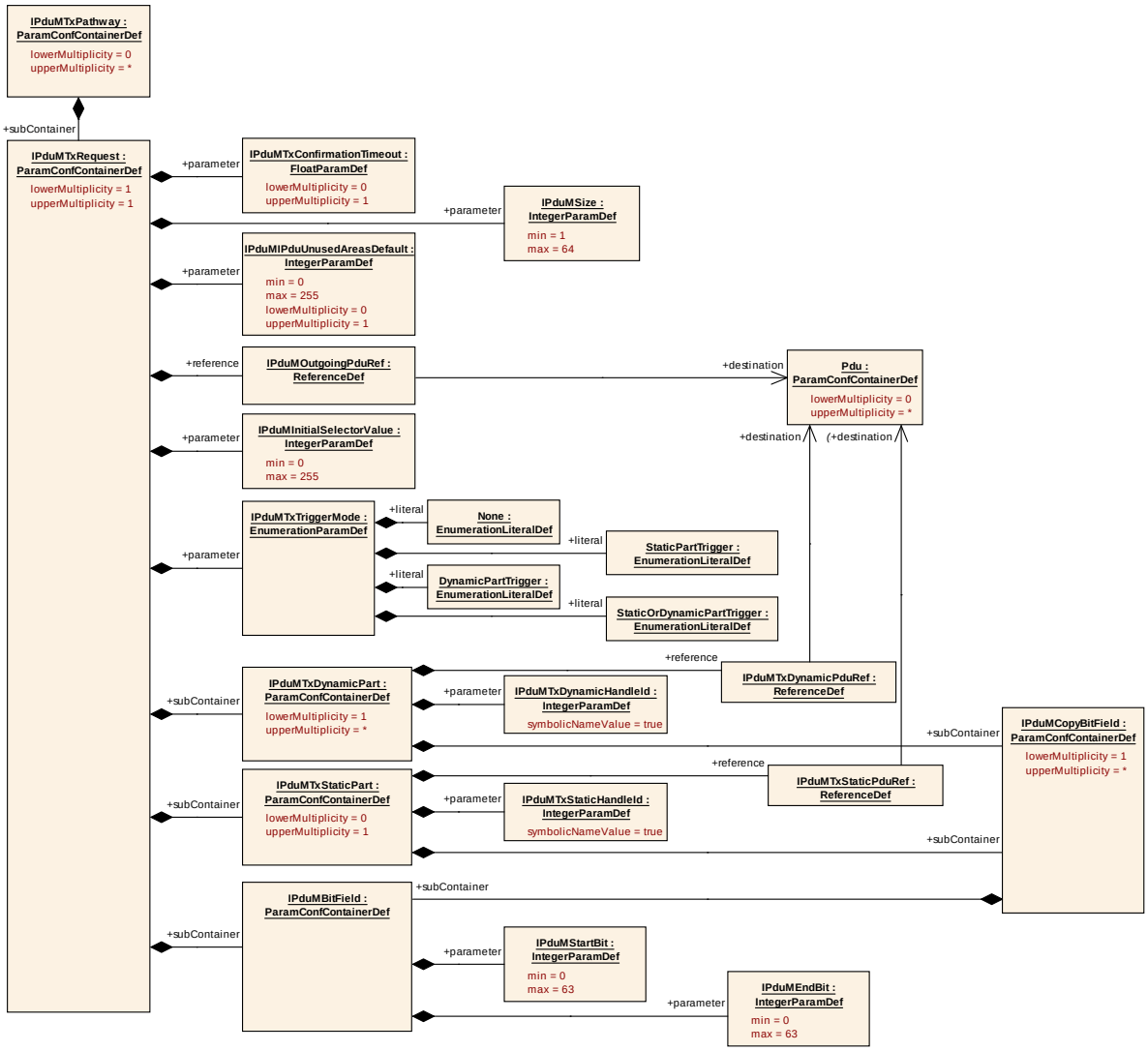


Figure 4.36: IPduMTxRequest configuration parameters

IPduMTxRequest

SWS Item	[IPDUM052]		
Container Name	IPduMTxRequest {IPDUM_TX_REQUEST_CONFIGURATION}		
Description	This is used to specify the configuration for Transmit requests. There will one instance of this container for each I-PDU that can be requested for transmission (the outgoing I-PDUs) by the IPduM.		
Configuration Parameters			

Name last modified in: V1.0.0	IPduMIPduUnusedAreasDefault {IPDUM_IPDU_UNUSED_AREAS_DEFAULT}		
Description	IPduM module fills not used areas of an I-PDU with this bit-pattern If this attribute is omitted the IPduM module does not fill the I-PDU.		
Multiplicity	0..1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	–	
	Link time	X	Fixed
	Post-build time	X	Reconfigurable
Scope / Dependency			

Name last modified in: V1.0.0	IPduMInitialSelectorValue {IPDUM_INITIAL_SELECTOR_VALUE}		
Description	This value is used by the initialization function to set the initial value of the selector field.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	–	
	Link time	X	Fixed
	Post-build time	X	Reconfigurable
Scope / Dependency			

Name last modified in: V1.0.0	IPduMOutgoingPduRef {IPDUM_OUTGOING_IPDU_ID}		
Description	Reference to the Pdu defining the outgoing I-Pdu. When the outgoing I-PDU is sent this is the I-Pdu ID to give it. It is the IPduM I-PDU ID of the assembled I-PDU.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	–	
	Link time	X	Fixed
	Post-build time	X	Reconfigurable
Scope / Dependency			

Name last modified in: V1.0.0	IPduMSize {IPDUM_SIZE}		
Description	Size of the outgoing I-PDU in bits.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 64		
Configuration Class	**Pre-compile time**	–	
	Link time	X	Fixed
	Post-build time	X	Reconfigurable
Scope / Dependency			

Name last modified in: V1.0.0	IPduMTxConfirmationTimeout {IPDUM_TX_CONFIRMATION_TIMEOUT}		
Description	This timeout defines the timeout period for monitoring the reception of the TxConfirmation. It is not used when an I-PDU is requested using the trigger transmit API.		
Multiplicity	0..1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	X	Fixed
	Post-build time	X	Reconfigurable
Scope / Dependency			

Name last modified in: V1.0.0	IPduMTxTriggerMode {IPDUM_TX_TRIGGER_MODE}	
Description	Selects whether to send the multiplexed I-PDU immediately or at some later date.	
Multiplicity	1	
Type	EnumerationParamDef	
Range	DynamicPartTrigger	Writing the I-PDU representing the dynamic part does trigger a sending of the I-PDU.
	None	Only the buffer in the IPduM are written but not send is triggered, used for IPduM I-PDUs which are requested by TriggerTransmit.
	StaticOrDynamicPart Trigger	Writing the I-PDU representing the static or the dynamic part does trigger a sending of the I-PDU.
	StaticPartTrigger	Writing the I-PDU representing the static part does trigger a sending of the I-PDU.

Configuration Class	**Pre-compile time**	–	
	Link time	X	Fixed
	Post-build time	X	Reconfigurable
Scope / Dependency			

Included Containers

Container Name	Multiplicity	Scope / Depedency
IPduMBitField	1	
IPduMTxDynamicPart	1..*	
IPduMTxStaticPart	0..1	

For `IPduMBitField` see section 4.4.4.

The configuration parameter `IPDUM_OUTGOING_IPDU_ID` specified in the IPduM SWS document [20] table `IPDUM_TX_REQUEST_CONFIGURATION` has been replaced by the reference `IPduMOutgoingPduRef` to allow the formalization of the COM-Stack as described in section 4.1.1.

IPduMTxDynamicPart

SWS Item	[IPDUM056]		
Container Name	IPduMTxDynamicPart {IPDUM_TX_DYNAMIC_PART}		
Description	Configuration parameters for an instance of a Tx_Request call into the IPduM. When a Tx Request with the IPDUM_HANDLE_ID is received by the IPduM, the bit fields in the incoming I-PDU are packed into the outgoing I-PDU buffer and then the send mode honored. This container is used the dynamic part of a TxRequest configuration. Therefore, for each outgoing I-PDU there will be one instance of this container for the dynamic part.		
Configuration Parameters			

Name last modified in: V1.0.0	IPduMTxDynamicHandleId {IPDUM_HANDLE_ID}		
Description	This is an incoming handle id. When the handle of an incoming Tx Request matches this, the bits fields (see Ipdum_CopyBitField) are copied and the Ipdum_TxTriggerMode is honored.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	–	
	Link time	X	Fixed
	Post-build time	X	Reconfigurable
Scope / Dependency			

Name last modified in: V1.0.0	IPduMTxDynamicPduRef		
Description	Reference to the Pdu representation in the ECU Configuration Description exchange file to be transmitted.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	–	
	Link time	X	Fixed
	Post-build time	X	Reconfigurable
Scope / Dependency			
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
IPduMCopyBitField	1..*		

For `IPduMCopyBitField` see section 4.4.5.

The reference `IPduMTxDynamicPduRef` has been introduced to allow the formalization of the COM-Stack introduced in section 4.1.1.

IPduMTxStaticPart

SWS Item	[IPDUM082]		
Container Name	IPduMTxStaticPart {IPDUM_TX_STATIC_PART}		
Description	Configuration parameters for an instance of a Tx_Request call into the IPduM. When a Tx Request with the IPDUM_HANDLE_ID is received by the IPduM, the bit fields in the incoming I-PDU are packed into the outgoing I-PDU buffer and then the send mode honored. This container is used for the static part of a TxRequest configuration. Therefore, for each outgoing I-PDU there will be one instance of this container for the static part if it exists.		
Configuration Parameters			

Name last modified in: V1.0.0	IPduMTxStaticHandleId {IPDUM_HANDLE_ID}		
Description	This is an incoming handle id. When the handle of an incoming Tx Request matches this, the bits fields (see Ipdum_CopyBitField) are copied and the Ipdum_TxTriggerMode is honored.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	–	
	Link time	X	Fixed
	Post-build time	X	Reconfigurable
Scope / Dependency			

Name last modified in: V1.0.0	IPduMTxStaticPduRef		
Description	Reference to the Pdu representation in the ECU Configuration Description exchange file to be transmitted.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	–	
	Link time	X	Fixed
	Post-build time	X	Reconfigurable
Scope / Dependency			
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
IPduMCopyBitField	1..*		

For `IPduMCopyBitField` see section 4.4.5.

The reference `IPduMTxStaticPduRef` has been introduced to allow the formalization of the COM-Stack introduced in section 4.1.1.

4.4.2.2 IPduMTxConfirmation

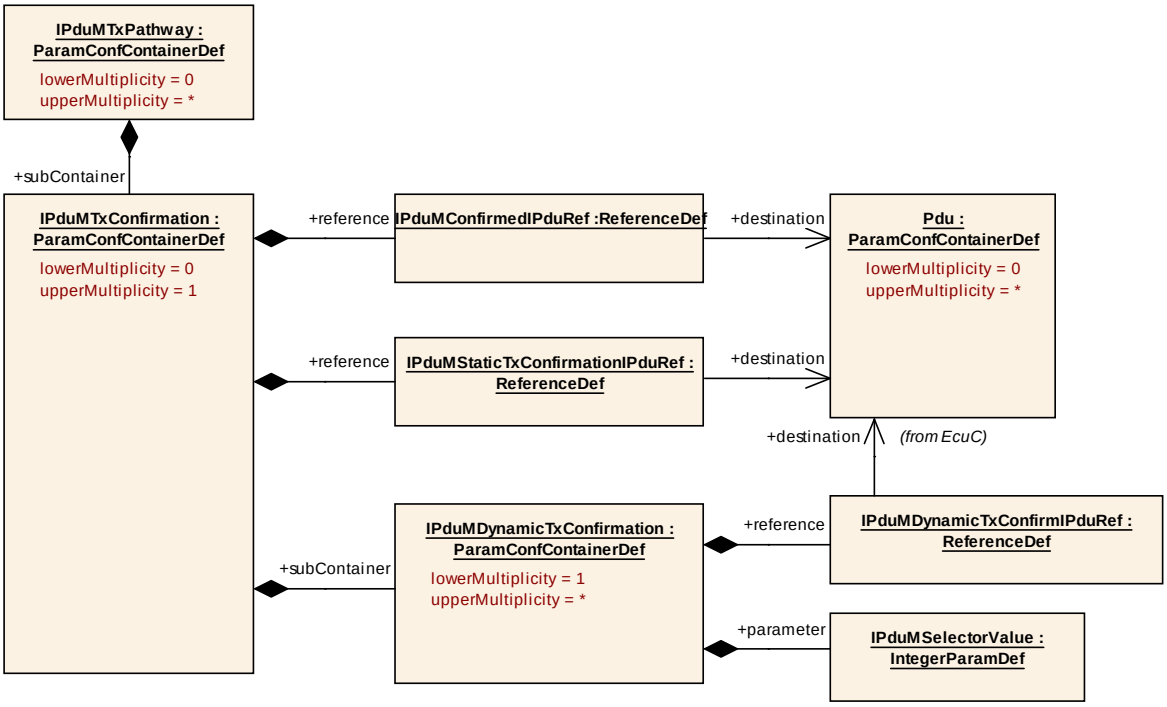


Figure 4.37: IPduMTxConfirmation configuration parameters

IPduMTxConfirmation

SWS Item	[IPDUM050]		
Container Name	IPduMTxConfirmation {IPDUM_TX_CONFIRMATION_CONFIGURATION}		
Description	A transmit request can be confirmed by the lower layer. This container is used to generate the matching confirmations for the static and dynamic parts of a multiplexed I-PDU. When an I-PDU is transmitted by the IPduM, the selector field value in that PDU needs to be stored in the IPduM so that the confirmation for the correct dynamic part can be generated. This is state internal to the IPduM at run-time. For the purposes of this container and IPDUM_DYNAMIC_CONFIRMATION this stored state is called Stored_Selector.		
Configuration Parameters			
Name last modified in: V1.0.0	IPduMConfirmedIPduRef {IPDUM_CONFIRMED_IPDU_ID}		
Description	This is the I-PDU ID of the previously transmitted I-PDU that is to be confirmed. Therefore, the value of this entity must match one of the set of IPDUM_OUTGOING_IPDU_ID in IPDUM_TX_REQUEST_CONFIGURATION. This is the ID of the IPduM I-PDU.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	–	
	Link time	X	Fixed
	Post-build time	X	Reconfigurable
Scope / Dependency			
Name last modified in: V1.0.0	IPduMStaticTxConfirmationIPduRef {IPDUM_STATIC_TX_CONFIRMATION_ID}		
Description	This is the I-PDU ID to use in the TxConfirmation for the static part. It is a COM I-PDU ID.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	–	
	Link time	X	Fixed
	Post-build time	X	Reconfigurable
Scope / Dependency			

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
IPduMDynamicTx Confirmation	1..*	

The configuration parameter `IPDUM_CONFIRMED_IPDU_ID` specified in the IPduM SWS document [20] table `IPDUM_TX_CONFIRMATION_CONFIGURATION` has been replaced by the reference `IPduMConfirmedIPduRef` to allow the formalization of the COM-Stack as described in section 4.1.1.

The configuration parameter `IPDUM_STATIC_TX_CONFIRMATION_ID` specified in the IPduM SWS document [20] table `IPDUM_TX_CONFIRMATION_CONFIGURATION` has been replaced by the reference `IPduMStaticTxConfirmationIPduRef` to allow the formalization of the COM-Stack as described in section 4.1.1.

IPduMDynamicTxConfirmation

SWS Item	[IPDUM051]		
Container Name	IPduMDynamicTxConfirmation {IPDUM_DYNAMIC_TX_CONFIRMATION}		
Description	The dynamic part of an I-PDU can have more than one I-PDU IDs for confirmations. The correct I-PDU ID for the confirmation is found from the selector field value of a previously transmitted I-PDU. It is assumed that this selector field is stored in some internal value called Stored_Selector. When a transmit confirmation is received the Stored_Selector is used to select an instance of IPDUM_DYNAMIC_CONFIRMATION by matching the Stored_Selector with the IPDUM_SELECTOR_VALUE.		
Configuration Parameters			
Name last modified in: V1.0.0	IPduMDynamicTxConfirmIPduRef {IPDUM_DYNAMIC_TX_CONFIRMATION_ID}		
Description	This is the I-PDU ID to use in the outgoing confirmation (confirmation for the COM I-PDU) when an incoming confirmation (for an IPduM I-PDU) is received and matches the stored Stored_Selector.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	–	
	Link time	X	Fixed
	Post-build time	X	Reconfigurable
Scope / Dependency			
Name last modified in: V1.0.0	IPduMSelectorValue {IPDUM_SELECTOR_VALUE}		
Description	When the selector field of the confirmed I-PDU matches the value in here then generate a TxConfirmation with the I-PDU ID indicated using the IPDUM_CONFIRMATION_ID value.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	X	Fixed
	Post-build time	X	Reconfigurable
Scope / Dependency			
No Included Containers			

The configuration parameter `IPDUM_DYNAMIC_TX_CONFIRMATION_ID` specified in the IPduM SWS document [20] table `IPDUM_DYNAMIC_TX_CONFIRMATION` has been replaced by the reference `IPduMDynamicTxConfirmIPduRef` to allow the formalization of the COM-Stack as described in section 4.1.1.

4.4.3 IPduMRxPathway

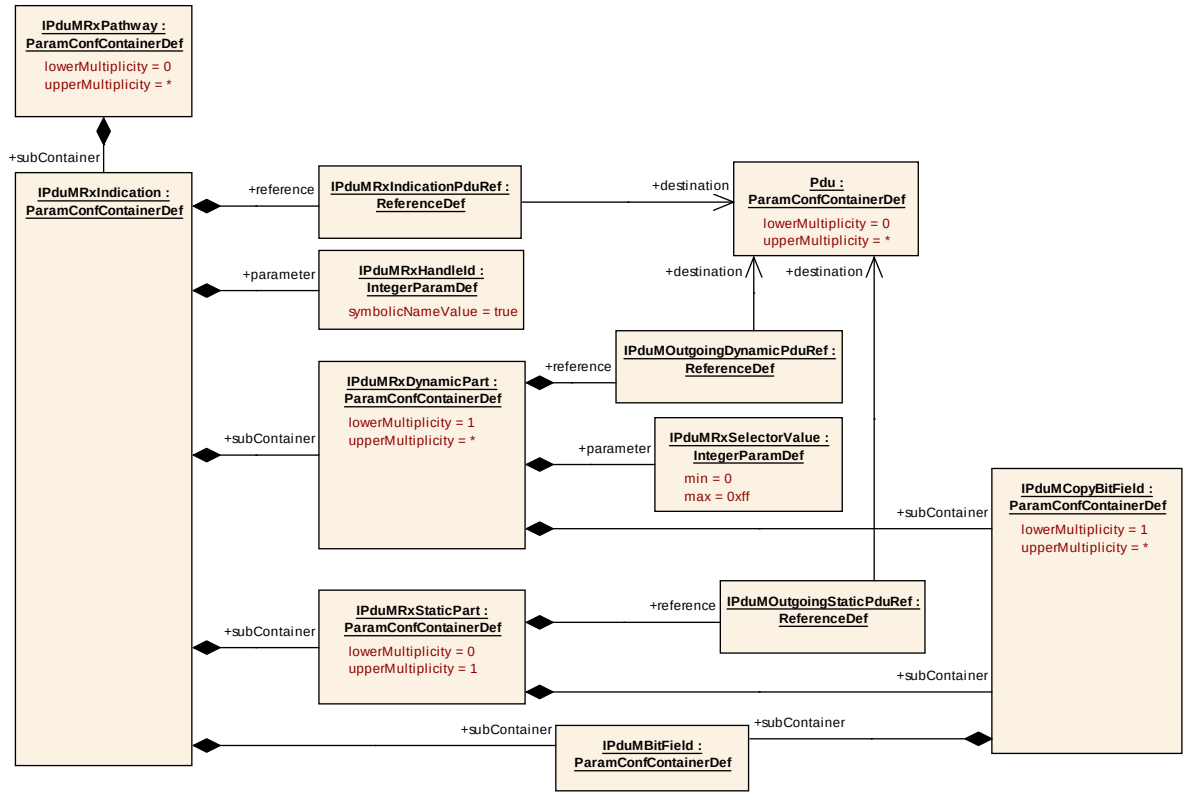


Figure 4.38: IPduMRxPathway configuration parameters

IPduMRxPathway

SWS Item	[IPDUM071]	
Container Name	IPduMRxPathway {IPDUM_RX_PATHWAY}	
Description	Contains the configuration parameters received I-PDUs by the IPduM module.	
Configuration Parameters Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
IPduMRxIndication	1	

IPduMRxIndication

SWS Item	[IPDUM047]		
Container Name	IPduMRxIndication {IPDUM_RX_INDICATION_CONFIGURATION}		
Description	Contains the configuration for incoming RxIndication calls.		
Configuration Parameters			
Name last modified in: V1.0.0	IPduMRxHandleId {IPDUM_HANDLE_ID}		
Description	This is the I-PDU ID of the incoming I-PDU. If an incoming RxIndication's I-PDU ID matches this value then it is unpacked according to the specification in this container.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	–	
	Link time	X	Fixed
	Post-build time	X	Reconfigurable
Scope / Dependency			

Name last modified in: V1.0.0	IPduMRxIndicationPduRef		
Description	Reference to the received Pdu representation in the ECU Configuration Description exchange file.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	–	
	Link time	X	Fixed
	Post-build time	X	Reconfigurable
Scope / Dependency			

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
IPduMBitField	1	
IPduMRxDynamicPart	1..*	
IPduMRxStaticPart	0..1	

The reference `IPduMRxIndicationPduRef` has been introduced to allow the formalization of the COM-Stack introduced in section 4.1.1.

IPduMRxDynamicPart

SWS Item	[IPDUM048]		
Container Name	IPduMRxDynamicPart {IPDUM_RX_DYNAMIC_PART}		
Description	This container contains the configuration for the dynamic part of incoming RxIndication calls. When an incoming received I-PDU's selector field matches the IPDUM_SELECTOR_VALUE the I-PDU is unpacked according to the values in the IPDUM_COPY_BITFIELD and then the new I-PDU constructed and sent out with the I-PDU ID in IPDUM_OUTGOING_HANDLE_ID.		
Configuration Parameters			

Name last modified in: V1.0.0	IPduMOutgoingDynamicPduRef {IPDUM_OUTGOING_HANDLE_ID}		
Description	When the new I-PDU is sent out it is sent with this I-PDU ID. Reference to the sent Pdu representation in the ECU Configuration Description exchange file.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	–	
	Link time	X	Fixed
	Post-build time	X	Reconfigurable
Scope / Dependency			

Name last modified in: V1.0.0	IPduMRxSelectorValue {IPDUM_SELECTOR_VALUE}		
Description	This is the selector value that this container refers to.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	–	
	Link time	X	Fixed
	Post-build time	X	Reconfigurable
Scope / Dependency			
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
IPduMCopyBitField	1..*		

For `IPduMCopyBitField` see section 4.4.5.

The configuration parameter `IPDUM_OUTGOING_HANDLE_ID` specified in the IPduM SWS document [20] table `IPDUM_RX_DYNAMIC_PART` has been replaced by the reference `IPduMOutgoingDynamicPduRef` to allow the formalization of the COM-Stack as described in section 4.1.1.

IPduMRxStaticPart

SWS Item	[IPDUM049]		
Container Name	IPduMRxStaticPart {IPDUM_RX_STATIC_PART}		
Description	This container contains the information on how to unpack the static part of an incoming I-PDU.		
Configuration Parameters			
Name last modified in: V1.0.0	IPduMOutgoingStaticPduRef {IPDUM_OUTGOING_HANDLE_ID}		
Description	When the new I-PDU is sent out it is sent with this I-PDU ID. Reference to the sent Pdu representation in the ECU Configuration Description exchange file.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	–	
	Link time	X	Fixed
	Post-build time	X	Reconfigurable
Scope / Dependency			
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
IPduMCopyBitField	1..*		

For `IPduMCopyBitField` see section 4.4.5.

The configuration parameter `IPDUM_OUTGOING_HANDLE_ID` specified in the IPduM SWS document [20] table `IPDUM_RX_STATIC_PART` has been replaced by the reference `IPduMOutgoingDynamicPduRef` to allow the formalization of the COM-Stack as described in section 4.1.1.

4.4.4 IPduMBitField

IPduMBitField

SWS Item	[IPDUM054]		
Container Name	IPduMBitField {IPDUM_BIT_FIELD}		
Description	This is used to specify a contiguous range of bits within an I-PDU. The range is inclusive.		
Configuration Parameters			
Name last modified in: V1.0.0	IPduMEndBit {IPDUM_END_BIT}		
Description	Bit position in an I-PDU of the end of the bit field.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 63		
Configuration Class	**Pre-compile time**	–	
	Link time	X	Fixed
	Post-build time	X	Reconfigurable
Scope / Dependency			

Name last modified in: V1.0.0	IPduMStartBit {IPDUM_START_BIT}		
Description	Bit position in an I-PDU of the start of the bit field.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 63		
Configuration Class	**Pre-compile time**	–	
	Link time	X	Fixed
	Post-build time	X	Reconfigurable
Scope / Dependency			
No Included Containers			

4.4.5 IPduMCopyBitField

IPduMCopyBitField

SWS Item	[IPDUM053]	
Container Name	IPduMCopyBitField {IPDUM_COPY_BIT_FIELD}	
Description	Two bit fields are specified, a source and a destination, so that the bits in the source can be copied to the bits in the destination. Within one I-PDU multiple instances of this container are used to specify the bit fields in that I-PDU. Adjacent bit fields could be merged in order to reduce the number of instances of this container.	
Configuration Parameters **Included Containers**		
Container Name	**Multiplicity**	**Scope / Depedency**
IPduMBitField	1	

For `IPduMBitField` see section 4.4.4.

4.5 Can

TODO ***ATTENTION: The configuration parameters of the CAN stack will change significantly after the validator 2 implementation. Currently the configuration isn't aligned to the approach that was taken in the other communication stacks.***

Overview of the Can Stack, figure 4.39.

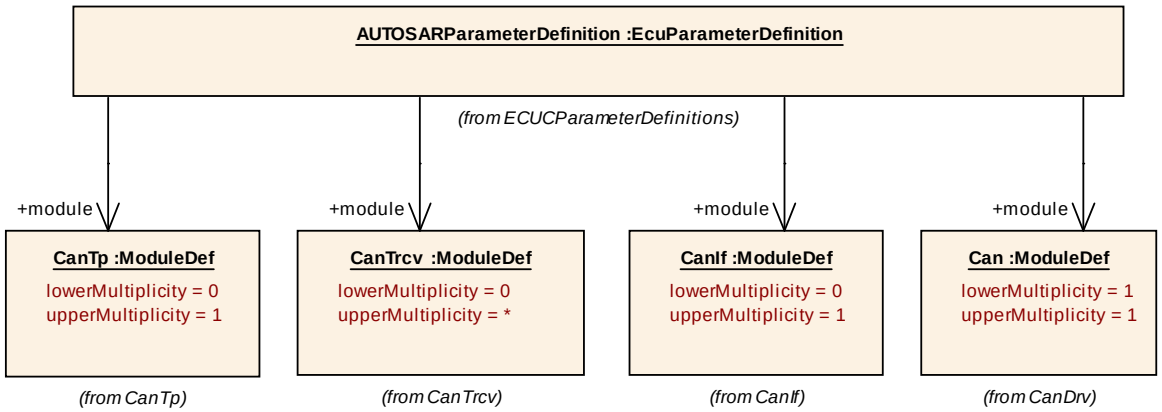


Figure 4.39: Can Stack

4.5.1 CAN Interface

The Can Interface [14] configuration parameters are structured according to figure 4.40.

- CanIfPrivateConfig in section 4.5.1.1
- CanIfPublicConfig in section 4.5.1.2
- CanIfControllerConfig in section 4.5.1.3
- CanIfDispatchConfig in section 4.5.1.4
- CanIfRxPduConfig in section 4.5.1.5
- CanIfTxPduConfig in section 4.5.1.6
- CanIfNetworkConfig in section 4.5.1.7
- CanIfDriverConfig in section 4.5.1.8

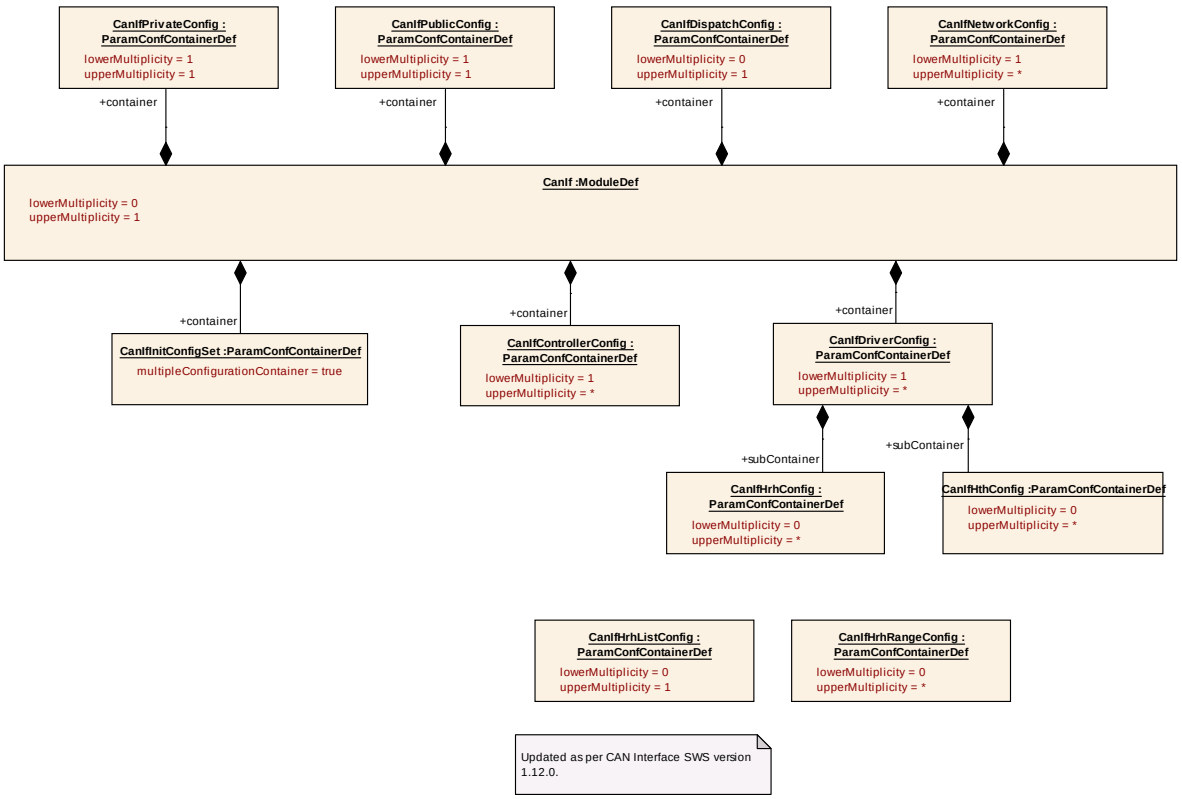


Figure 4.40: CanIf parameters overview

The following tables contain the definition of all parameters concerning the CAN interface.

CanIf

Module Name	CanIf	
Module Description	This container includes all necessary configuration sub-containers according the CAN Interface configuration structure.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CanIfControllerConfig	1..*	This container contains the configuration (parameters) of all addressed CAN controllers by each underlying CAN driver.
CanIfDispatchConfig	0..1	Holds: Call-out functions for CANIF with respect to Network. This call-out functions defined in this container are common to all the configured networks.
CanIfDriverConfig	1..*	Configuration parameters for all the undelying CAN driver are aggregated under this container.
CanIfInitConfigSet	1	
CanIfNetworkConfig	1..*	This container contains parameter required for configuring Network.
CanIfPrivateConfig	1	This container contains the private configuration (parameters) of the CAN Interface.
CanIfPublicConfig	1	This container contains the public configuration (parameters) of the CAN Interface.
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.

4.5.1.1 CanIfPrivateConfig

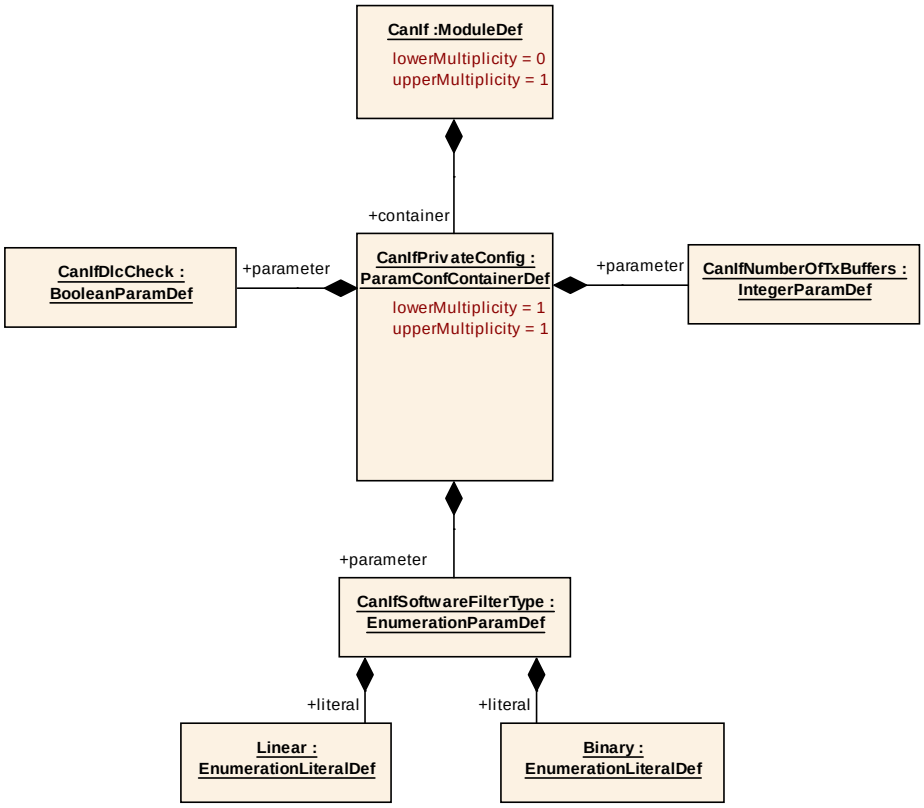


Figure 4.41: CanIf Private Configuration

CanIfPrivateConfig

SWS Item			
Container Name	CanIfPrivateConfig {CanInterfacePrivateConfiguration}		
Description	This container contains the private configuration (parameters) of the CAN Interface.		
Configuration Parameters			
Name last modified in: V1.0.0	CanIfDlcCheck {CANIF_DLC_CHECK}		
Description	Enables or Disables the DLC check. Whenever this parameter is set to TRUE, the DLC of received L-PDU is validated against the pre-defined DLC. The Rx L-PDU is passed to upper layer only when it passes the DLC check. Where as when the value of this parameter is FALSE, all the received L-PDUs are passed to <user> without any DLC check.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfNumberOfTxBuffers {CANIF_NUMBER_OF_TXBUFFERS}		
Description	Defined the number of L-PDU elements for the transmit buffering. The Tx L-PDU buffers shall be used to store an L-PDU once for each different L-PDU handle.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfSoftwareFilterType {CANIF_SOFTWARE_FILTER_TYPE}		
Description	This parameter provides different types of software filter methods that can be implemted in the CANIF. The value of this parameter is vendor specific.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	Binary	Selects Binary Filter method.	
	Linear	Selects Linear Filter method.	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

4.5.1.2 CanIfPublicConfig

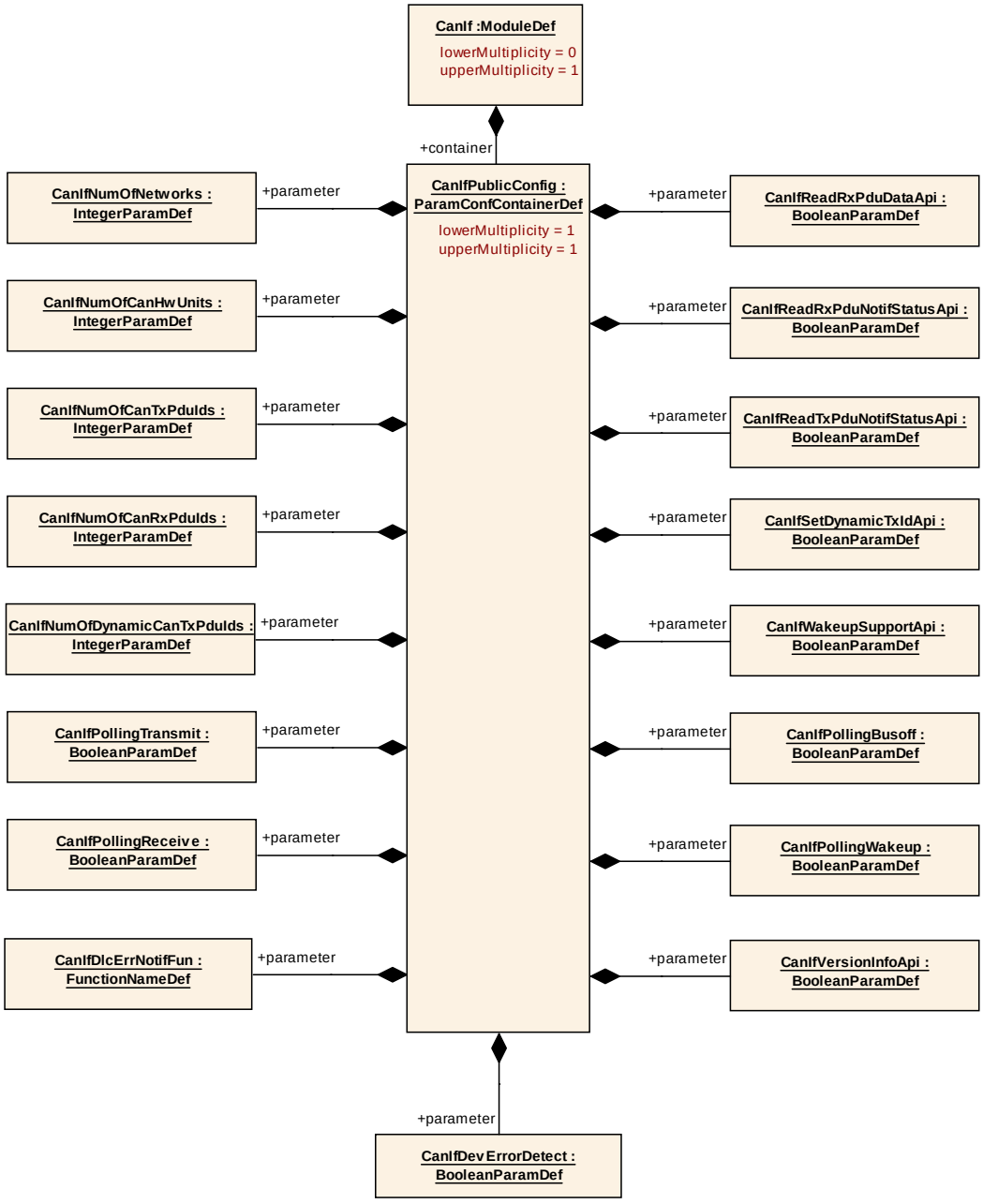


Figure 4.42: CanIf Public Configuration

CanIfPublicConfig

SWS Item			
Container Name	CanIfPublicConfig {CanInterfacePublicConfiguration}		
Description	This container contains the public configuration (parameters) of the CAN Interface.		
Configuration Parameters			

Name last modified in: V1.0.0	CanIfDevErrorDetect {CANIF_DEV_ERROR_DETECT}		
Description	Enables and disables the development error detection and notification mechanism.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfDlcErrNotifFun {CANIF_DLC_ERROR_NOTIFICATION}		
Description	Holds: The call-out function name which will be invoked by CANIF whenever DLC Check fails. The function confgured is common to all the receive L-Pdus. If parameter is not configured, no call-out function is provided.		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfNumOfCanHwUnits {CANIF_NUMBER_OF_CAN_HW_UNITS}		
Description	Holds: Number of served CAN hardware units.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfNumOfCanRxPduIds {CANIF_NUMBER_OF_CANRXPDUIDS}		
Description	Total number of CanRxPduIds to be handled.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfNumOfCanTxPduIds {CANIF_NUMBER_OF_CANTXPDUIDS}		
Description	Total number of CanTxPduIds to be handled.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfNumOfDynamicCanTxPduIds {CANIF_NUMBER_OF_DYNAMIC_CANTXPDUIDS}		
Description	Total number of CanTxPduIds to be handled.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfNumOfNetworks {CANIF_NUMBER_OF_NETWORKS}		
Description	Number of served CAN networks.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfPollingBusoff {CANIF_POLLING_BUSOFF}		
Description	Selects polling mode for BusOff events for each underlying CAN driver.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfPollingReceive {CANIF_POLLING_RECEIVE}		
Description	Selects polling mode for receive events for each underlying CAN driver.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfPollingTransmit {CANIF_POLLING_TRANSMIT}		
Description	Selects polling mode for transmit events for each underlying CAN driver.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfPollingWakeup {CANIF_POLLING_WAKEUP}		
Description	Selects polling mode for wakeup events for each underlying CAN driver.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfReadRxPduDataApi {CANIF_READRXPDU_DATA_API}		
Description	Enables / Disables the API CanIf_ReadRxPduData() for reading received L-PDU data.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfReadRxPduNotifStatusApi {CANIF_READRXPDU_NOTIF_STATUS_API}		
Description	Enables / Disables the API CanIf_ReadRxNotifStatus() for reading the notification status of receive L-PDUs.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfReadTxPduNotifStatusApi {CANIF_READTXPDU_NOTIF_STATUS_API}		
Description	Enables / Disables the API CanIf_ReadTxNotifStatus() for reading the notification status of transmit L-PDUs.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfSetDynamicTxIdApi {CANIF_SETDYNAMICTXID _API}		
Description	Enables / Disables the API CanIf_SetDynamicTxId(), for reconfiguring CAN ID of Dynamic Transmit L-PDUs.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfVersionInfoApi {CANIF_VERSION_INFO _API}		
Description	Enables / Disables the API CanIf_GetVersionInfo(), for reading the version information of CAN Interface module.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfWakeupSupportApi {CANIF_WAKEUP_SUPPORT _API}		
Description	Enables / Disables the API, for 'wakeup notification'. When value of this parameter is set to FALSE, no call-out functions will be provided to CAN Driver Module. If it is set to TRUE, CANIF provides call-out functions for each underlying CAN Driver,		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency **No Included Containers**			

4.5.1.3 CanIfControllerConfig

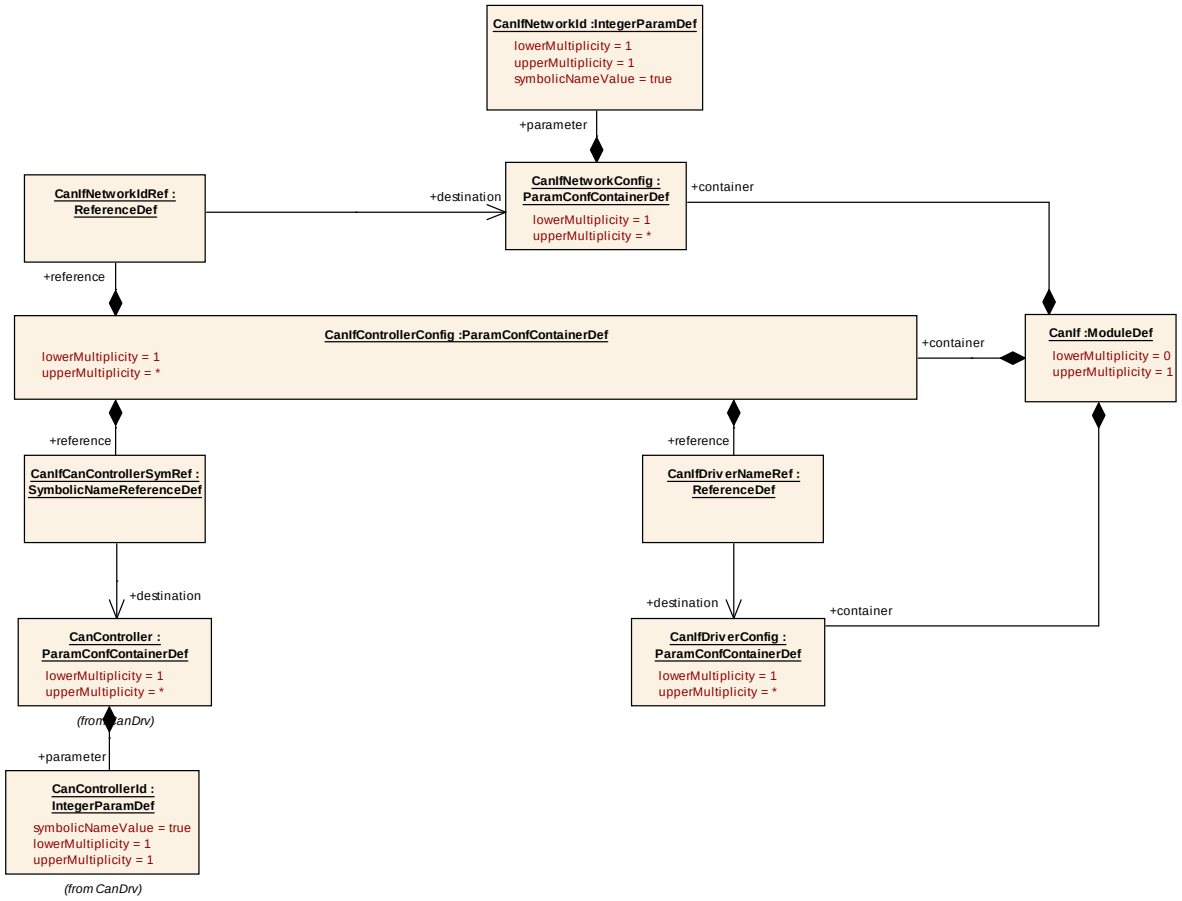


Figure 4.43: CanIf Controller Configuration

CanIfControllerConfig

SWS Item			
Container Name	CanIfControllerConfig {CanInterfaceControllerConfiguration}		
Description	This container contains the configuration (parameters) of all addressed CAN controllers by each underlying CAN driver.		
Configuration Parameters			

Name last modified in: V1.0.0	CanIfCanControllerSymRef {CANIF_CONTROLLER_REF_ID}		
Description	Reference to the CanIfControllerConfig. Refers to Handle ID of controller. Unique inside a given CAN Driver. The value of this parameter should start with 0 and continue without any gaps. The Handle Ids for controller are defined in CAN Driver Configuration. Those handle Ids are referred through this parameter		
Multiplicity	1		
Type	Symbolic name reference to CanController		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfDriverNameRef {CANIF_DRIVER_REF_NAME}		
Description	Refers to the CAN Driver Name to which the controller belongs to. This parameter refers to CanIfDriverConfig container.		
Multiplicity	1		
Type	Reference to CanIfDriverConfig		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfNetworkIdRef {CANIF_NETWORK_ REF_ID}		
Description	Reference to CanIfNetworkConfig container. This parameter refers to the Network Id to which the current controller belongs to. Note that more than one controllers can be connected to same CAN Network.		
Multiplicity	1		
Type	Reference to CanIfNetworkConfig		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			
No Included Containers			

4.5.1.4 CanIfDispatchConfig

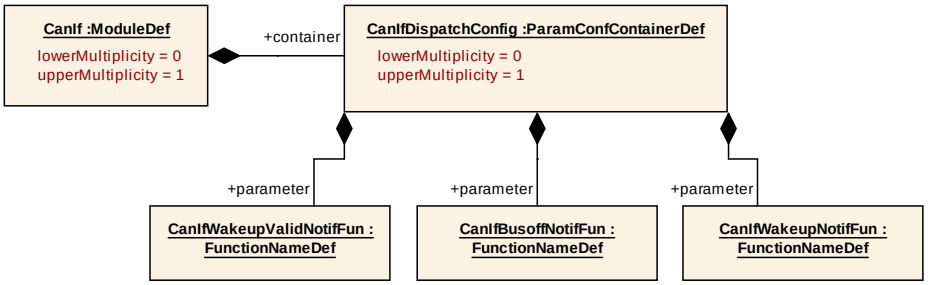


Figure 4.44: CanIf Dispatch Configuration

CanIfDispatchConfig

SWS Item			
Container Name	CanIfDispatchConfig {CanInterfaceDispatcherConfiguration}		
Description	Holds: Call-out functions for CANIF with respect to Network. This call-out functions defined in this container are common to all the configured networks.		
Configuration Parameters			

Name last modified in: V1.0.0	CanIfBusoffNotifFun {CANIF_USER_BUSOFF_NOTIFICATION}		
Description	The call-out function invoked by CANIF after receiving busoff notification from CAN Driver. If this parameter is empty no call-out function is provided.		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfWakeupNotifFun {CANIF_USER_WAKEUP_NOTIFICATION}		
Description	The call-out function invoked by CANIF after receiving wakeup notification from CAN Driver. If this parameter is empty no call-out function is provided.		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfWakeupValidNotifFun {CANIF_USER_WAKEUP_VALIDATION_NOTIFICATION}		
Description	The call-out function invoked by CANIF at first reception of L-PDU after wakeup. If this parameter is empty no call-out function is provided.		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency **No Included Containers**			

4.5.1.5 CanIfRxPduConfig

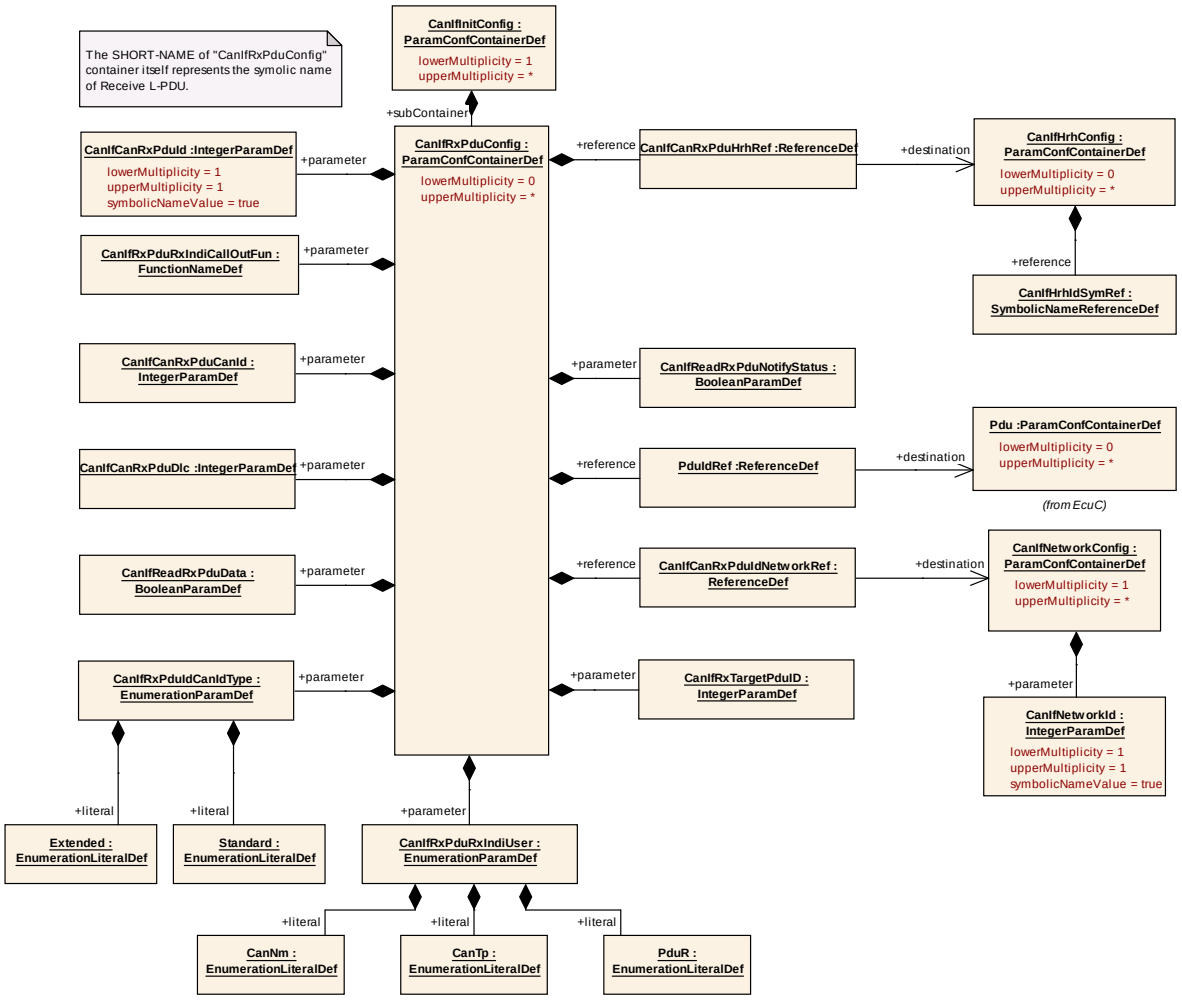


Figure 4.45: CanIf Rx Pdu Configuration

CanIfRxPduConfig

SWS Item			
Container Name	CanIfRxPduConfig {CanInterfaceRxPduConfiguration}		
Description	This container contains the configuration (parameters) of each receive CAN L-PDU.		
Configuration Parameters			
Name last modified in: V1.0.0	CanIfCanRxPduCanId {CANIF_CANRXPDUID_CANID}		
Description	CAN Identifier of Receive CAN L-PDUs used by the CAN Interface. Exa: Software Filtering.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			
Name last modified in: V1.0.0	CanIfCanRxPduDlc {CANIF_CANRXPDUID_DLC}		
Description	Data Length code of received CAN L-PDUs used by the CAN Interface. Exa: DLC check.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			
Name last modified in: V1.0.0	CanIfCanRxPduHrhRef {CANIF_HRH_REF_ID}		
Description	The HRH to which Rx L-PDU belongs to, is referred through this parameter.		
Multiplicity	1		
Type	Reference to CanIfHrhConfig		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			
Name last modified in: V1.0.0	CanIfCanRxPduId {CANIF_CANRXPDUID}		
Description	ECU wide unique, symbolic handle for receive CAN L-PDU. The CanIfRxPduId is configurable.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			
Name last modified in: V1.0.0	CanIfCanRxPduIdNetworkRef {CANIF_NETWORK_REF_ID}		
Description	Reference to the CAN network ID to which the receive CAN L-PDU belongs to.		
Multiplicity	1		
Type	Reference to CanIfNetworkConfig		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	CanIfReadRxPduData {CANIF_READRXPDU_DATA}		
Description	Enables and disables the Rx buffering for reading of received L-PDU data.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	CanIfReadRxPduNotifyStatus {CANIF_READRXPDU_NOTIFY_STATUS}		
Description	Enables / Disables receive indication for each receive CAN L-PDU for reading its' notification status.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	CanIfRxPduIdCanIdType {CANIF_CANRXPDUID_CANIDTYPE}		
Description	The parameter defines the CANID type. The value of this parameter shall be used for validating CANID of Rx L-PDUs.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	Extended	The CANID is of type Extended (29 bits)	
	Standard	The CANID is of type Standard (11 bits)	
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	CanIfRxPduRxIndiCallOutFun {CANIF_USER RX_INDICATION}		
Description	Name of target indication services to target upper layers (PduR, CanNm and CanTp). If parameter is not configured no call-out function is configured in the upper layer for this Rx L-PDU.		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfRxPduRxIndiUser {CANIF_RX_USER_TYPE}		
Description	This parameter defines the type of the receive indication call-outs called to the corresponding upper layer the used TargetRxPduId belongs to.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	CanNm	CAN Network Management Module	
	CanTp	CAN Transport Protocol Module	
	PduR	PDU Router Module	
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	CanIfRxTargetPduID {CANIF_TARGETRXPDUID}		
Description	This parameter holds the PDU handle Id which will be passed to upper layers during receive indication. The PDU handle Id defined for CanIfCanRxPduId parameter and target PDU handle ID need not be same. The target pdu hadle IDs of L-PDUs belonging to PduR and CanTp module should be unique. The target pdu hadle ID of L-PDUs belonging to CanNm module should be same as its Network Id.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	PduIdRef		
Description	Reference to the ”global” Pdu structure to allow harmonization of handle IDs in the COM-Stack.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

4.5.1.6 CanIfTxPduConfig

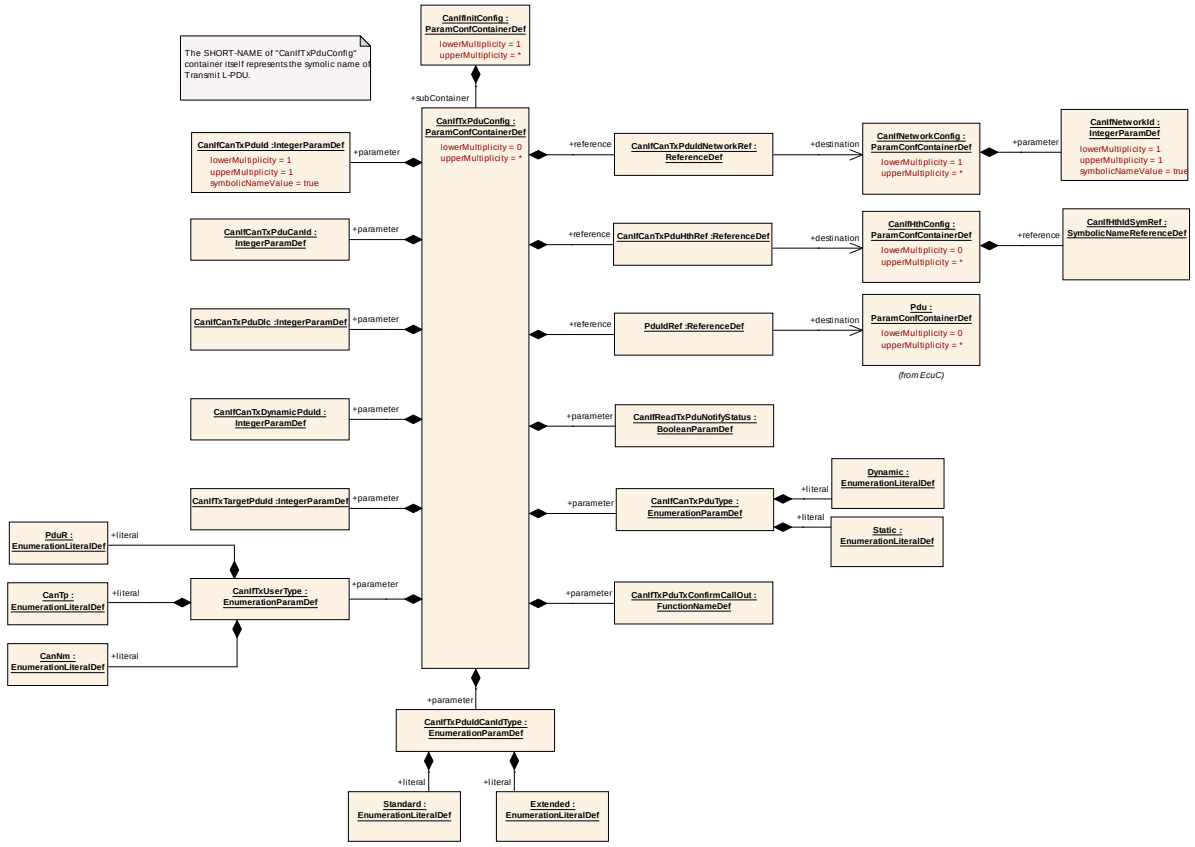


Figure 4.46: CanIf Tx Pdu Configuration

CanIfTxPduConfig

SWS Item			
Container Name	CanIfTxPduConfig {CanInterfaceTxPduConfiguration}		
Description	This container contains the configuration (parameters) of each transmit CAN L-PDU.		
Configuration Parameters			

Name last modified in: V1.0.0	CanIfCanTxDynamicPduId {CANIF_CANDYNAMICTXPDUID}		
Description	ECU wide unique, symbolic handle number for dynamic transmit CAN L-PDU. The CanIfCanTxDynamicPduId is configurable. The handle Ids for DYNAMIC PDUs will start after handle Ids of STATIC PDUs.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	CanIfCanTxPduCanId {CANIF_CANTXPDUID_CANID}		
Description	CAN Identifier of transmit CAN L-PDUs. Used by the CAN Interface and CAN Driver Modules for CAN L-PDU transmission.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	CanIfCanTxPduDlc {CANIF_CANTXPDUID_DLC}		
Description	Data Length Code of transmit CAN L-PDUs. Used CAN Interface and CAN Driver Modules for CAN L-PDU transmission.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	CanIfCanTxPduHthRef {CANIF_HTH_REF_ID}		
Description	Handle, that defines the hardware object or the pool of hardware objects configured for transmission. The parameter refers HTH Id, to which the L-PDU belongs to.		
Multiplicity	1		
Type	Reference to CanIfHthConfig		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfCanTxPduId {CANIF_CANTXPDUID}		
Description	ECU wide unique, symbolic handle for transmit CAN L-PDU. The CanIfCanTxPduId is configurable at pre-compile and post-built time.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	CanIfCanTxPduIdNetworkRef {CANIF_CANTXPDUID_NETWORK}		
Description	This parameter holds the NETWORK ID to which this PDU belongs to.		
Multiplicity	1		
Type	Reference to CanIfNetworkConfig		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	CanIfCanTxPduType {CANIF_CANTXPDUID_TYPE}		
Description	Defines the type of each transmit CAN L-PDU.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	Dynamic	For L-PDUs of this type, CAN ID can be re-configured at runtime.	
	Static	For L-PDUs of this type, CAN ID can not be re-configured.	
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	CanIfReadTxPduNotifyStatus {CANIF_READTXPDU_NOTIFY_STATUS}		
Description	Enables and disables the API for reading the notification status of transmit L-PDUs.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	CanIfTxPduIdCanIdType {CANIF_CANIFTXPDUID_CANIDTYPE}		
Description	The parameter decides the CANID type. The value of this parameter shall be used for validating the input parameter in CanIf_SetDynamicTxId() API.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	Extended	The CANID is of type Extended (29 bits)	
	Standard	The CANID is of type Standard (11 bits)	
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	CanIfTxPduTxConfirmCallOut {CANIF_USER TX_CONFIRMATION}		
Description	Name of target confirmation services to target upper layers (PduR, CanNm and CanTp. If parameter is not configured then no call-out function is provided by the upper layer for this Tx L-PDU.		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfTxTargetPduId {CANIF_TARGETTXPDUID}		
Description	This parameter holds the PDU handle Id which will be passed to upper layers during transmission confirmation. The PDU handle Id defined for CanIfCanTxPduId parameter and target PDU handle ID need not be same. The target pdu hadle IDs of L-PDUs belonging to PduR and CanTp module should be unique. The target pdu hadle ID of L-PDUs belonging to CanNm module should be same as its Network Id.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	CanIfTxUserType {CANIF_TX_USER_TYPE}		
Description	This parameter defines the type of the transmit confirmation call-out called to the corresponding upper layer the used TargetTxPduId belongs to.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	CanNm	CAN Network Management Module	
	CanTp	CAN Transport Protocol Module	
	PduR	PDU Router Module	
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	PduIdRef		
Description	Reference to the ”global” Pdu structure to allow harmonization of handle IDs in the COM-Stack.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	–	
	Post-build time	–	
Scope / Dependency **No Included Containers**			

4.5.1.7 CanIfNetworkConfig

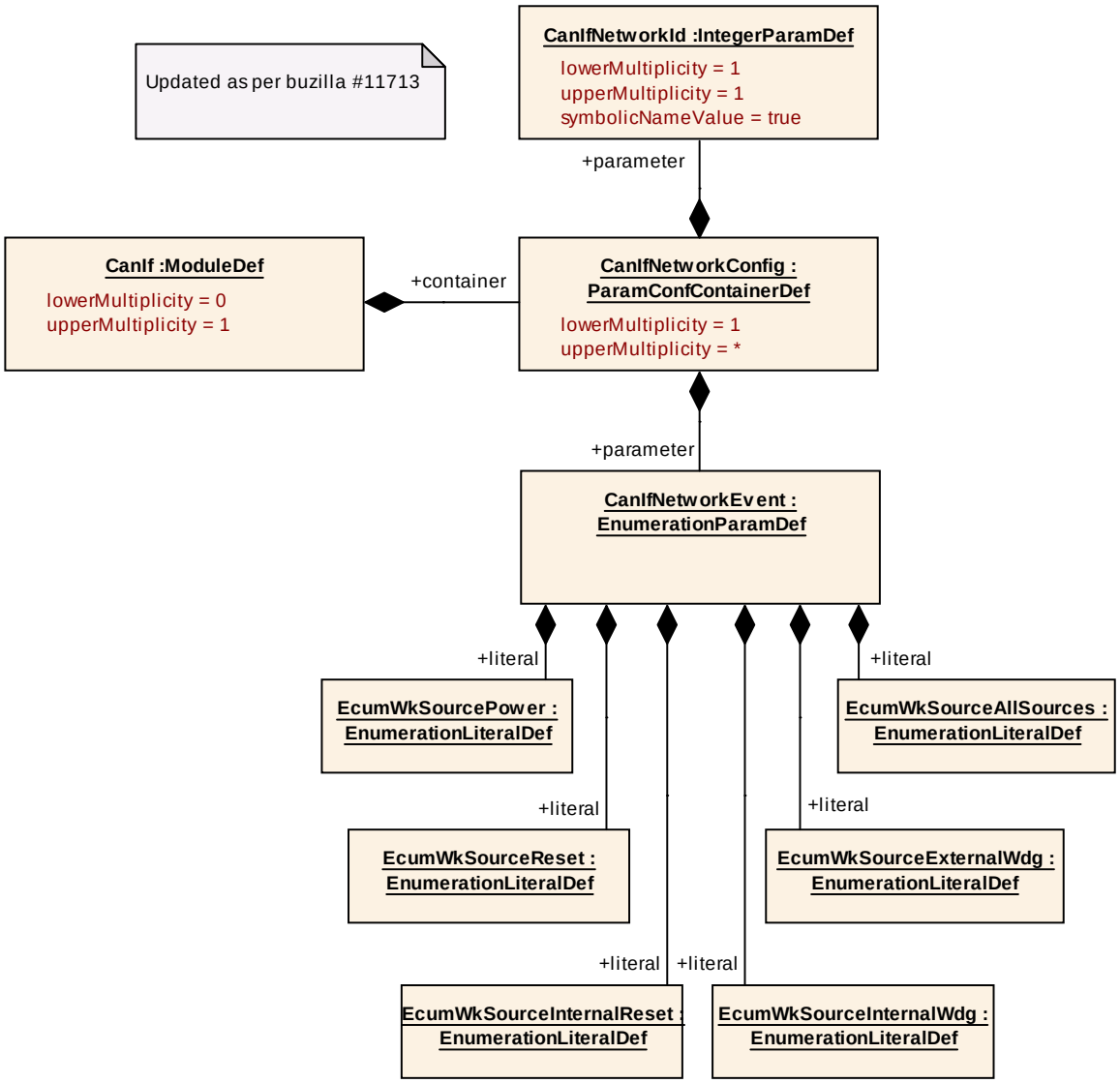


Figure 4.47: CanIf Network Configuration

CanIfNetworkConfig

SWS Item			
Container Name	CanIfNetworkConfig {CanInterfaceNetworkConfiguration}		
Description	This container contains parameter required for configuring Network.		
Configuration Parameters			

Name last modified in: V1.0.0	CanIfNetworkEvent {CANIF_NETWORK_EVENT}		
Description	Defines different types of sources for controller wakeup.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	EcumWkSourceAllSources		
	EcumWkSourceExternal Wdg		
	EcumWkSourceInternal Reset		
	EcumWkSourceInternal Wdg		
	EcumWkSourcePower		
	EcumWkSourceReset		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	CanIfNetworkId {CANIF_NETWORK_ID}		
Description	This parameter holds the handle Id value. The value for Network ID should start with 0 and continue without any gaps. The value of Network Id is unique across the system.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			
No Included Containers			

4.5.1.8 CanIfDriverConfig

CanIfDriverConfig

SWS Item			
Container Name	CanIfDriverConfig {CanInterfaceDriverConfiguration}		
Description	Configuration parameters for all the undelying CAN driver are aggregated under this container.		
Configuration Parameters			

Name last modified in: V1.0.0	CanIfBusoffNotifFun {CANIF_CONTROLLER_BUSOFF_NOTIFICATION}		
Description	Whenever value of this parameter is set to TRUE, a call-out function for 'Controller Busoff Notification' , is provided to CAN Driver. The syntax of call -out function will be: CanIf_ControllerBusOff_<CAN Driver Name>. Whenever value of this parameter is set to FALSE, no call-out function is provided.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfDriverName {CANIF_DRIVER_NAME}		
Description	Name of CAN driver. The value of this parameter can be found in the "CommonPublishedInformation" of the corresponding CAN driver's configuration description.		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfDriverRef {CANIF_DRIVER_VENDOR_ID}		
Description	CAN Interface Driver Reference. This reference can be used to get any information (Ex. Driver Name, Vendor ID) from the CAN driver. The CAN Driver name can be derived from the ShortName of the CAN driver module. Ref: Bug# 13687		
Multiplicity	1		
Type	Reference to CanGeneral		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfReceiveIndFun {CANIF_RECEIVE _INDICATION}		
Description	Whenever value of this parameter is set to TRUE, a call-out function for 'Receive Indication', is provided to CAN Driver. The syntax of call -out function will be: CanIf_RxIndication_<CAN Driver Name>. Whenever value of this parameter is set to FALSE, no call-out function is provided.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfTransmitCancFun {CANIF_TRANSMIT_CANCELLATION}		
Description	Whenever value of this parameter is set to TRUE, a call-out function for 'Transmit Cancellation' , is provided to CAN Driver. The syntax of call -out function will be: CanIf_CancelTxConfirmation_<CAN Driver Name>. Whenever value of this parameter is set to FALSE, no call-out function is provided.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfTxConfirmation {CANIF_TRANSMIT_CONFIRMATION}		
Description	Whenever value of this parameter is set to TRUE, a call-out function for 'Transmit Confirmation', is provided to CAN Driver. The syntax of call -out function will be: CanIf_TxConfirmation_<CAN Driver Name>. Whenever value of this parameter is set to FALSE, no call-out function is provided.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfWakeupNotifFun {CANIF_CONTROLLER_WAKEUP_NOTIFICATION}		
Description	Whenever value of this parameter is set to TRUE, a call-out function for 'Controller Wakeup Notification' , is provided to CAN Driver. The syntax of call -out function will be: CanIf_ControllerWakeup_<CAN Driver Name>. Whenever value of this parameter is set to FALSE, no call-out function is provided.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
CanIfHrhConfig	0..*		
CanIfHthConfig	0..*		

4.5.1.8.1 CanIfHthConfig

CanIfHthConfig

SWS Item	
Container Name	CanIfHthConfig {CanInterfaceHthConfiguration}
Description	This container contains parameters related to each HTH.
Configuration Parameters	

Name last modified in: V1.0.0	CanIfCanControllerHthIdRef {CANIF_CONTROLLER_REF_ID}		
Description	Reference to controller Id to which the HTH belongs to. A controller can contain one or more HTHs. The parameter refers to "CanIfCanControllerSymRef" parameter in "CanIfControllerConfig" container.		
Multiplicity	1		
Type	Reference to CanIfControllerConfig		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfHthIdSymRef {CANIF_HTH_REF_ID}		
Description	The parameter refers to a particular HTH object in the CAN Driver Module configuration. The HTH id is unique in a given CAN Driver. The HTH Ids are defined in the CAN Driver Module and hence it is derived from CAN Driver Configuration.		
Multiplicity	1		
Type	Symbolic name reference to CanHardwareObject		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfHthType {CANIF_HTH_TYPE}		
Description	This parameter defines the type of HTH i.e, ”BasicCan” or ”FullCan”.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	BasicCan	For a BasicCAN HTH buffers have to be reserved for buffering Tx L-PDUs. The size of Tx Queue is specified in parameter CanIfNumberOfTxBuffers.	
	FullCan	The HTH is of type Full CAN. At the maximum only one Tx L-PDU should be assigned to this type of HTH.	
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			
No Included Containers			

4.5.1.8.2 CanIfHrhConfig

CanIfHrhConfig

SWS Item			
Container Name	CanIfHrhConfig {CanInterfaceHrhConfiguration}		
Description	This container contains parameters specific to HRH.		
Configuration Parameters			
Name last modified in: V1.0.0	CanIfCanControllerHrhIdRef {CANIF_CONTROLLER_REF_ID}		
Description	Reference to controller Id to which the HRH belongs to. A controller can contain one or more HRHs. The parameter refers to ”CanIfCanControllerSymRef” parameter in ”CanIfControllerConfig” container.		
Multiplicity	1		
Type	Reference to CanIfControllerConfig		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			
Name last modified in: V1.0.0	CanIfHrhIdSymRef {CANIF_HRH_REF_ID}		
Description	The parameter refers to a particular HRH object in the CAN Driver Module configuration. The HRH id is unique in a given CAN Driver. The HRH Ids are defined in the CAN Driver Module and hence it is derived from CAN Driver Configuration.		
Multiplicity	1		
Type	Symbolic name reference to CanHardwareObject		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			
Name last modified in: V1.0.0	CanIfHrhType {CANIF_ HRH_TYPE}		
Description	Defines the HRH type i.e, whether its a BasicCan or FullCan.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	BasicCan	For a BasicCAN HTH buffers have to be reserved for buffering Tx L-PDUs. The size of Tx Queue is specified in parameter CanIfNumberOfTxBuffers.	
	FullCan	The HTH is of type Full CAN. At the maximum only one Tx L-PDU should be assigned to this type of HTH.	
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanIfSoftwareFilterHrh {CANIF_SOFTWARE_FILTER_HRH}		
Description	Enables/Disables the software filtering for this particular HRH.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
CanIfHrhListConfigSet	1		
CanIfHrhRangeConfigSet	1		

CanIfHrhListConfig

SWS Item	
Container Name	CanIfHrhListConfig {CanInterfaceHrhListConfiguration}
Description	Defines the parameters required for configuraing list of CANIDs for a given same HRH.
Configuration Parameters	

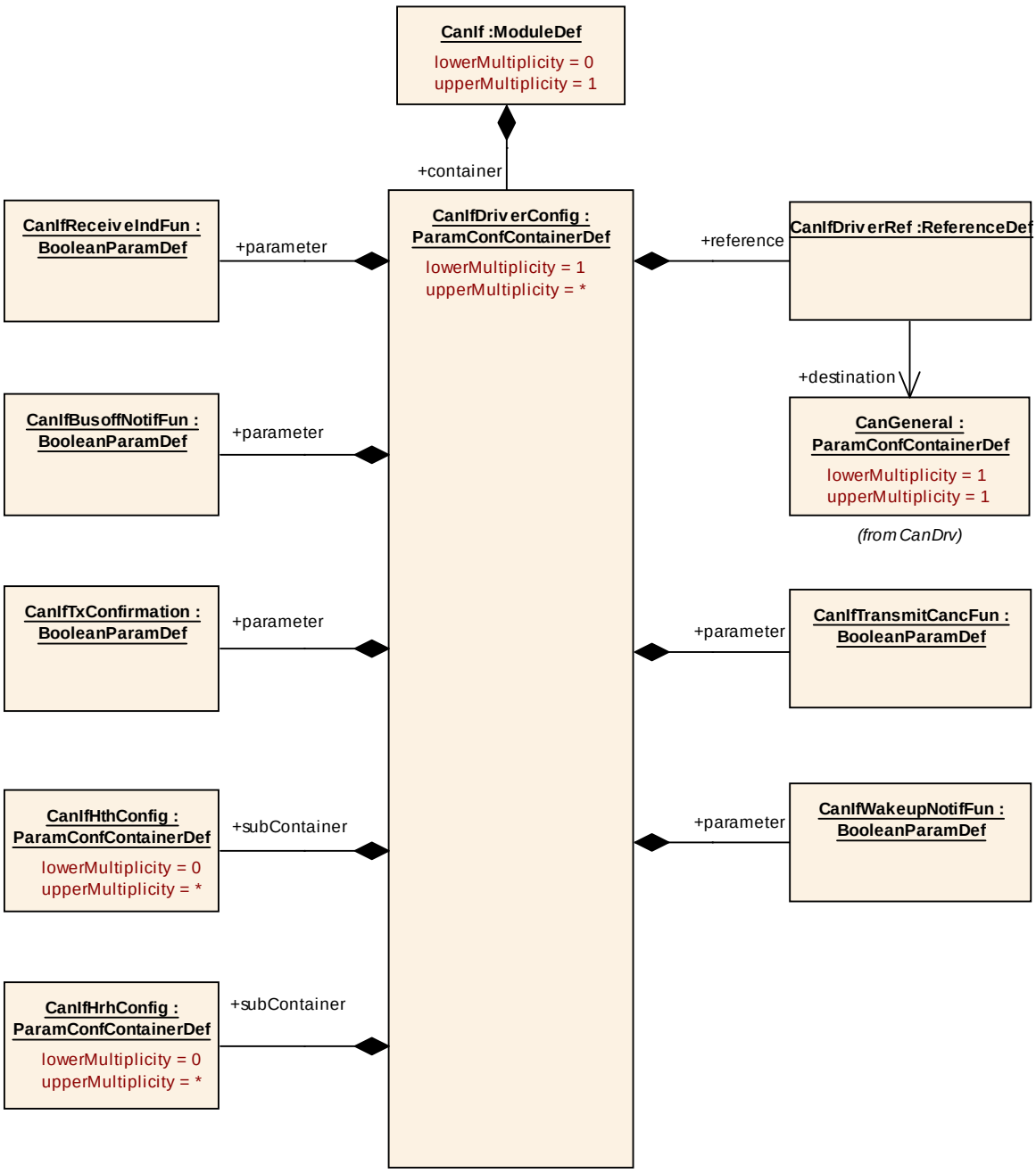
Name last modified in: V1.0.0	CanIfHrhListCanId {CANIF_HRHLIST_CANID}		
Description	Every instance of this parameter holds single CAN ID. Since there can be more than one CANID assigned to same HRH, the lowermultiplicity is 1 and uppermultiplicity is *. Note that for every CANID mentioned with this parameter should have corresponding entry in the ”CanIfRxPduConfig” container.		
Multiplicity	1..*		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			
No Included Containers			

CanIfHrhRangeConfig

SWS Item			
Container Name	CanIfHrhRangeConfig {CanInterfaceHrhRangeConfiguration}		
Description	Defines the parameters required for configuraing multiple CANID ranges for a given same HRH.		
Configuration Parameters			

Name last modified in: V1.0.0	CanIfRxPduLowerCanId {CANIF_HRHRANGE_LOWER_CANID}		
Description	While configuring a CANID RANGEs, the lower CANID is assigned to this parameter. The upper CANID of the corresponding RANGE is assigned to parameter CanIfRxPduUpperCanId. Exa 1: If the CANID range is 0x500 to 0x510, then CanIfRxPduLowerCanId = 0x500 and CanIfRxPduUpperCanId = 0x510 Note that for every CANID mentioned in RANGE, should have corresponding entry in the ”CanIfRxPduConfig” container. Exa 2: Considering ”Exa 1”, there should be entry for every CANID which lies between 0x500 to 0x510 in ”CanIfRxPduConfig” container.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	CanIfRxPduUpperCanId {CANIF_HRHRANGE_UPPER_CANID}		
Description	While configuring a RANGE, the upperl CANID is assigned to this parameter. The lower CANID of the corresponding RANGE is assigned to parameter CanIfRxPduLowerCanId. Note that for every CANID mentioned in RANGE, should have corresponding entry in the ”CanIfRxPduConfig” container.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			
No Included Containers			



The values for "CanIfDriverName" and "CanIfDriverVendorId" can be found in the "CommonPublishedInformation" of the corresponding CAN driver's configuration description.

Figure 4.48: CanIf Driver Configuration

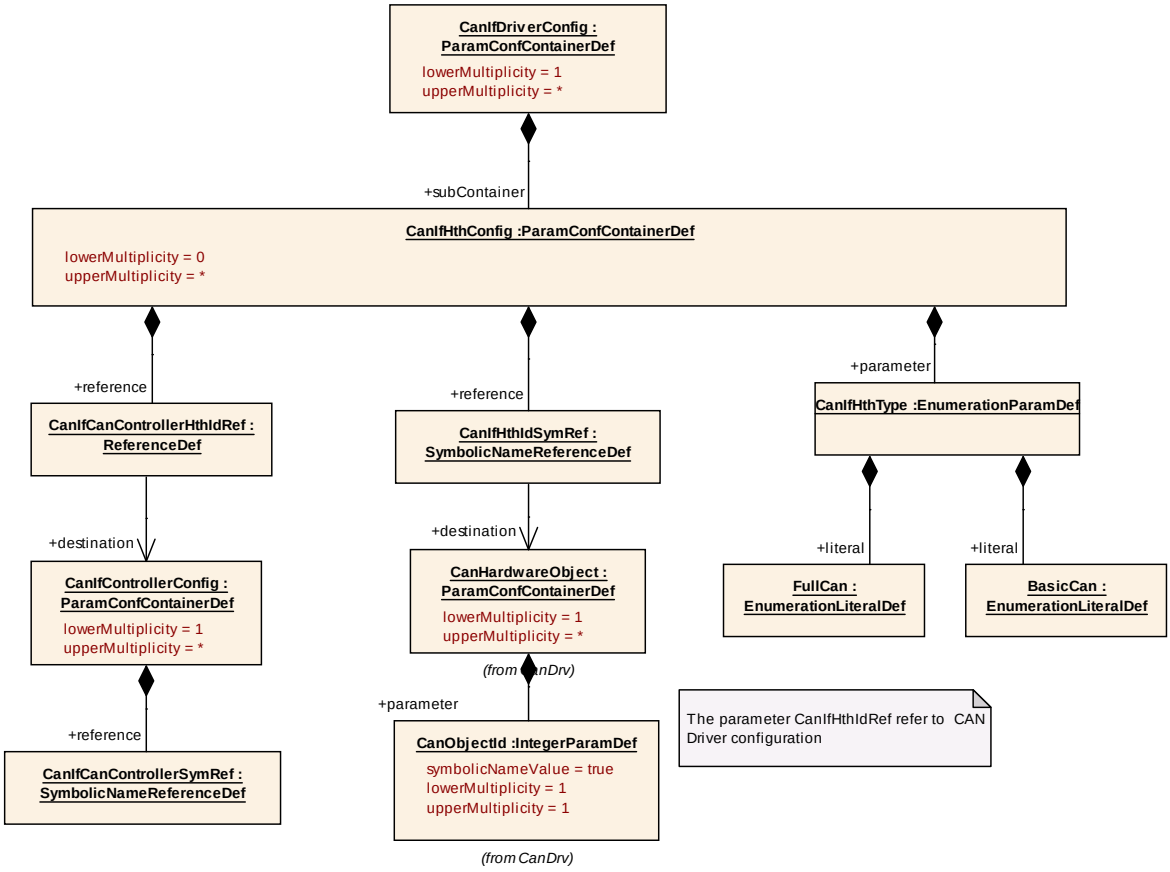


Figure 4.49: CanIf Hth Configuration

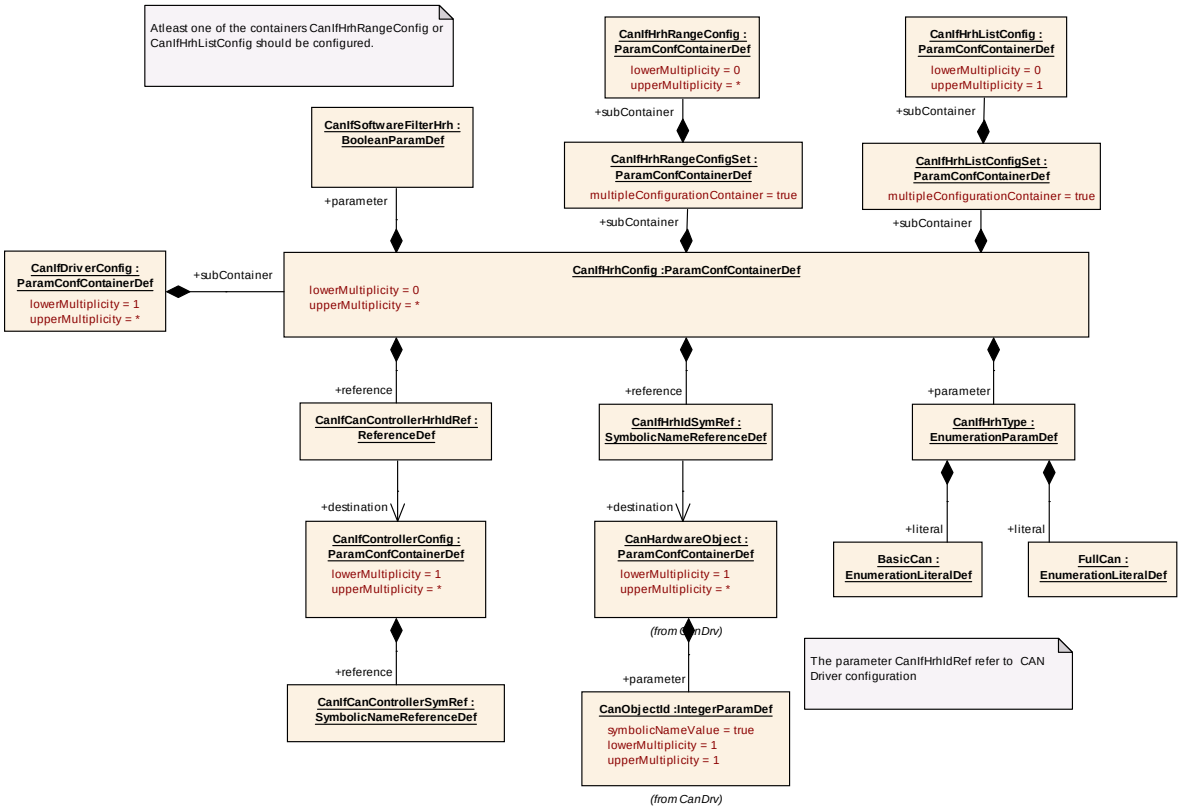


Figure 4.50: CanIf Hrh Configuration

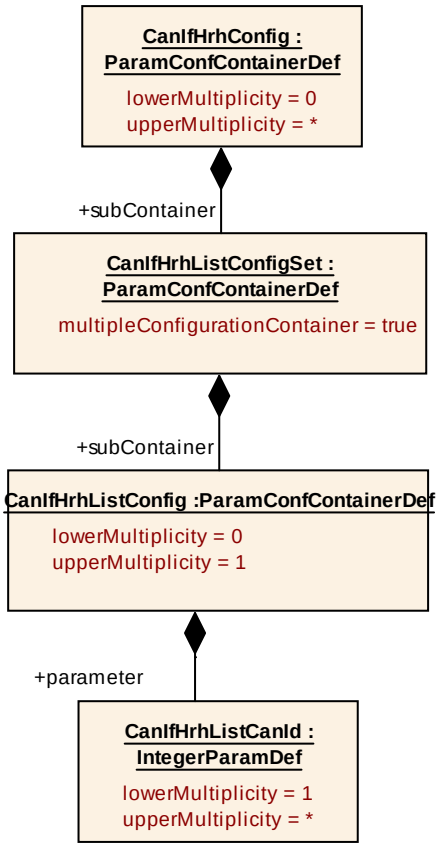


Figure 4.51: CanIf Hrh List Configuration

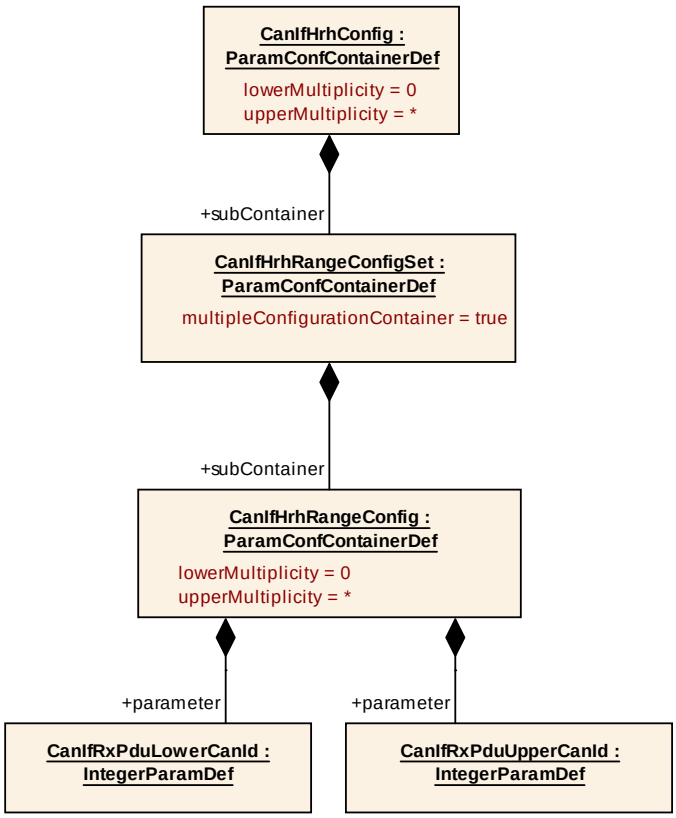


Figure 4.52: CanIf Hrh Range Configuration

4.5.2 CAN Driver

The Can Driver [13] configuration parameters are structured according to figure 4.53.

- CanGeneral in section 4.5.2.1
- CanController in section 4.5.2.2

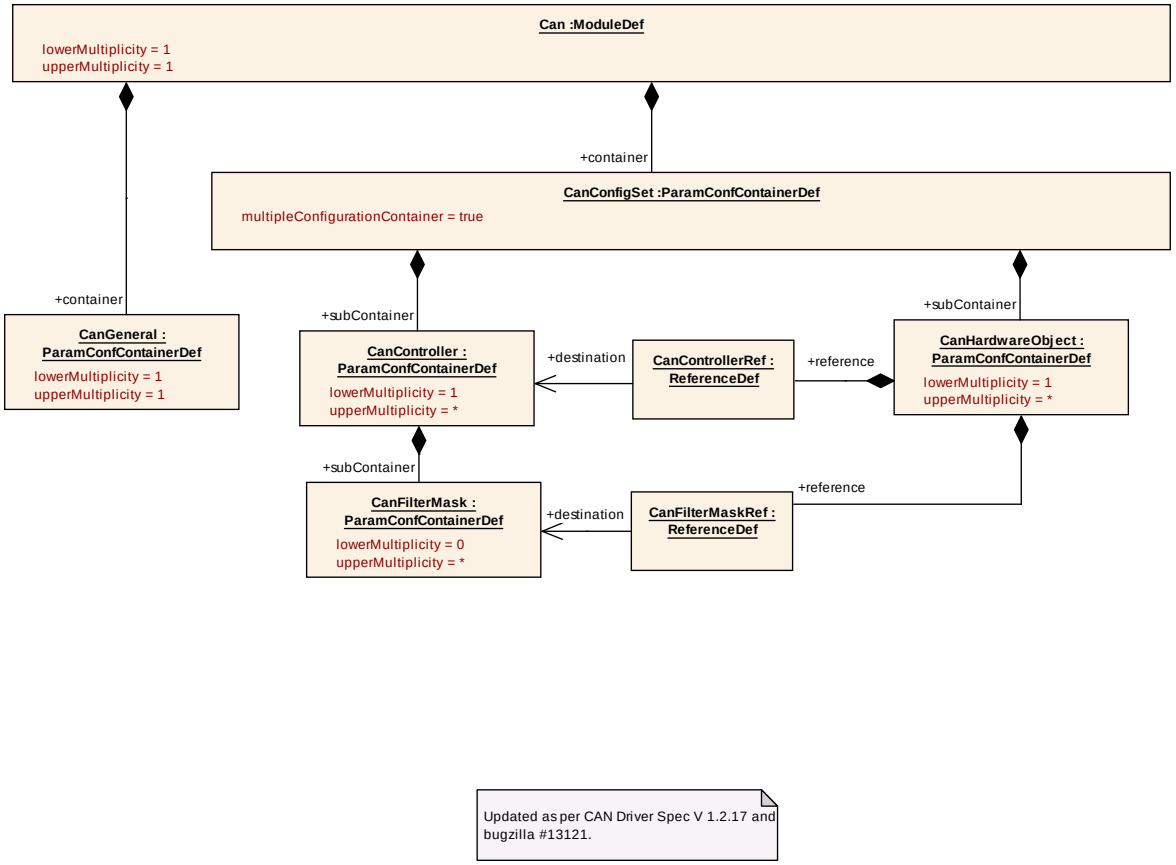


Figure 4.53: Can parameters overview

The following tables contain the definition of all parameters concerning the Can driver.

Can

Module Name	Can	
Module Description	This container holds the configuration of a single CAN Driver.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CanConfigSet	1	
CanGeneral	1	This container contains the parameters related each CAN Driver Unit.
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.

4.5.2.1 CanGeneral

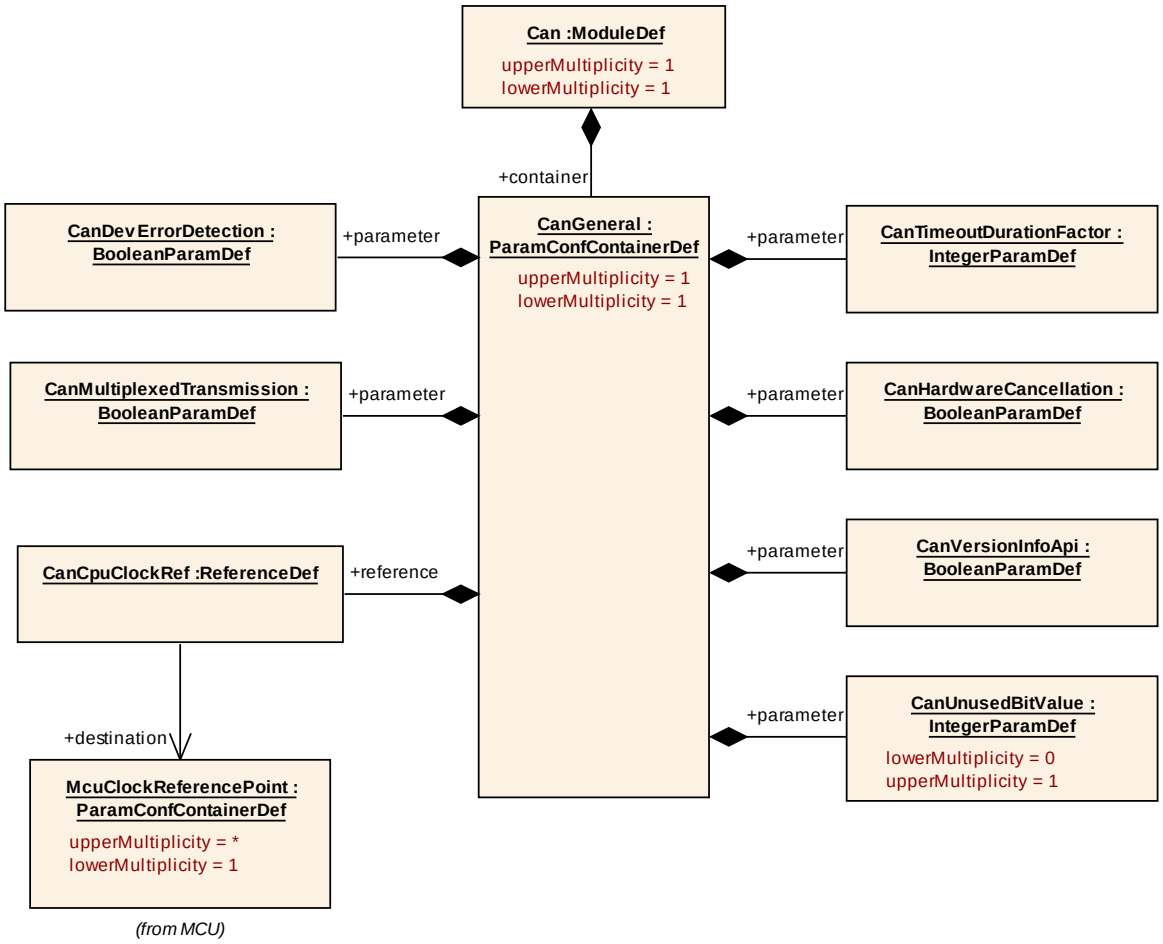


Figure 4.54: CanDrv General Configuration

CanGeneral

SWS Item	
Container Name	CanGeneral {CanDriverGeneralConfiguration}
Description	This container contains the parameters related each CAN Driver Unit.
Configuration Parameters	

Name last modified in: V1.1.0	CanCpuClockRef {CAN_CPU_CLOCK_REFERENCE}		
Description	Reference to the CPU clock configuration, which is set in the MCU driver configuration		
Multiplicity	1		
Type	Reference to McuClockReferencePoint		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanDevErrorDetection {CAN_DEV_ERROR_DETECT} [CAN064]		
Description	Switches the Development Error Detection and Notification ON or OFF.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanHardwareCancellation {CAN_HW_TRANSMIT_CANCELLATION} [CAN069]		
Description	Specifies if hardware cancellation shall be supported.ON or OFF		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanMultiplexedTransmission {CAN_MULTIPLEXED_TRANSMISSION} [CAN095]		
Description	Specifies if multiplexed transmission shall be supported.ON or OFF		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanTimeoutDurationFactor {CAN_TIMEOUT_DURATION} [CAN113]		
Description	Specifies the maximum number of loops for blocking function until a timeout is raised in short term wait loops.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanUnusedBitValue {CAN_UNUSED_BIT_VALUE} [CAN116]		
Description	Set unused bits to a defined value.		
Multiplicity	0..1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	CanVersionInfoApi {CAN_VERSION_INFO_API} [CAN106]		
Description	Switches the Can_GetVersionInfo() API ON or OFF.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

4.5.2.2 CanController

CanController

SWS Item			
Container Name	CanController {CanController}		
Description	This container contains the configuration parameters of the CAN controller(s).		
Configuration Parameters			

Name last modified in: V1.0.0	CanBusoffProcessing {CAN_BUSOFF_PROCESSING}		
Description	Enables / disables API Can_MainFunction_BusOff() for handling busoff events in polling mode.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	Interrupt	Interrupt Mode of operation.	
	Polling	Polling Mode of operation.	
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanControllerActivation {CAN_CONTROLLER_ACTIVATION}		
Description	Defines if a CAN controller is used in the configuration.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanControllerBaudRate {CAN_CONTROLLER_BAUD_RATE} [CAN005]		
Description	Specify the baud rate of the controller.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	–	
	Post-build time	X	Variant 1
Scope / Dependency			

Name last modified in: V1.0.0	CanControllerId {CAN_DRIVER_CONTROLLER_ID}		
Description	This parameter provides the controller ID which is unique in a given CAN Driver. The value for this parameter starts with 0 and continue without any gaps.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	CanControllerPropSeg {CAN_CONTROLLER_PROP_SEG} [CAN073]		
Description			
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	–	
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	CanControllerSeg1 {CAN_CONTROLLER_PHASE_SEG1} [CAN074]		
Description	Specifies TSEG1		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	–	
	Post-build time	X	Variant 1
Scope / Dependency			

Name last modified in: V1.0.0	CanControllerSeg2 {CAN_CONTROLLER_PHASE_SEG2} [CAN075]		
Description	Specifies TSEG2		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	–	
	Post-build time	X	Variant 1
Scope / Dependency			

Name last modified in: V1.0.0	CanControllerTimeQuanta {CAN_CONTROLLER_TIME_QUANTA} [CAN063]		
Description	Specifies the time quanta for the controller. The calculation of the resulting prescaler value depending on module clocking and time quanta shall be done offline Hardware specific.		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	–	
	Post-build time	X	Variant 1
Scope / Dependency			

Name last modified in: V1.0.0	CanRxProcessing {CAN_RX_PROCESSING}		
Description	Enables / disables API Can_MainFunction_Read() for handling PDU reception events in polling mode.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	Interrupt	Interrupt Mode of operation.	
	Polling	Polling Mode of operation.	
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanTxProcessing {CAN_TX_PROCESSING}		
Description	Enables / disables API Can_MainFunction_Write() for handling PDU transmission events in polling mode.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	Interrupt	Interrupt Mode of operation.	
	Polling	Polling Mode of operation.	
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanWakeupProcessing {CAN_WAKEUP_PROCESSING}		
Description	Enables / disables API Can_MainFunction_Wakeup() for handling wakeup events in polling mode.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	Interrupt	Interrupt Mode of operation.	
	Polling	Polling Mode of operation.	
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Included Containers

Container Name	Multiplicity	Scope / Depedency
CanFilterMask	0..*	

4.5.2.2.1 CanHardwareObject

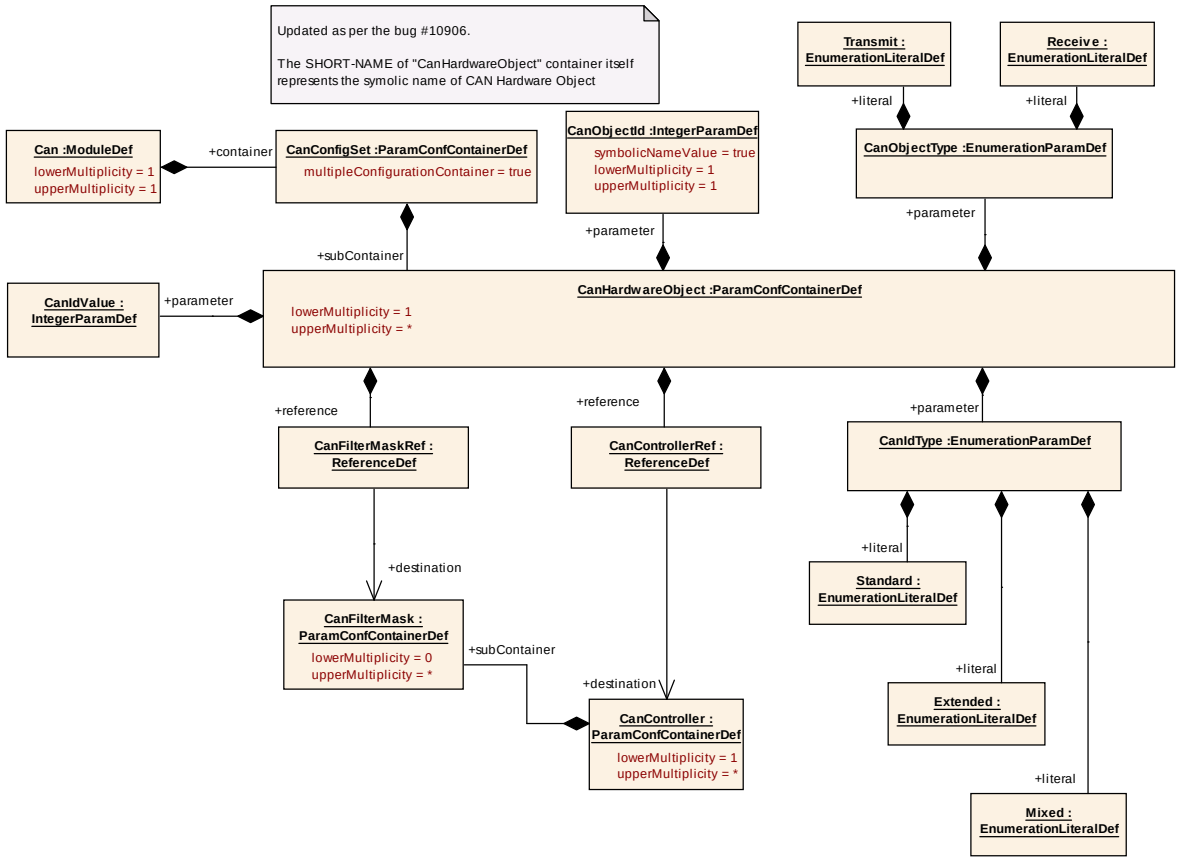


Figure 4.55: Can Hardware Object

CanHardwareObject

SWS Item			
Container Name	CanHardwareObject {CanHardwareObject}		
Description	This container contains the configuration (parameters) of CAN Hardware Objects.		
Configuration Parameters			

Name last modified in: V1.0.0	CanControllerRef {CAN_CONTROLLER_REFERENCE}		
Description	Reference to CAN Controller to which the HOH is associated to.		
Multiplicity	1		
Type	Reference to CanController		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	–	
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	CanFilterMaskRef {CAN_MASK_REFERENCE}		
Description	Reference to the filter mask that is used for hardware filtering togerther with the CAN_ID_VALUE		
Multiplicity	1		
Type	Reference to CanFilterMask		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	–	
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	CanIdType {CAN_ID_TYPE} [CAN065]		
Description	Specifies whether the IdValue is of type - standard identifier - extended identifier - mixed mode		
Multiplicity	1		
Type	EnumerationParamDef		
Range	Extended	All the CANIDs are of type extended only (29 bit).	
	Mixed	The type of CANIDs can be both Standard or Extended.	
	Standard	All the CANIDs are of type standard only (11bit).	
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	–	
	Post-build time	X	Variant 1
Scope / Dependency			

Name last modified in: V1.0.0	CanIdValue {CAN_ID_VALUE} [CAN067]		
Description	Specifies (together with the filter mask)- the identifiers range that passes the hardware filter for of RX objects.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	–	
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	CanObjectId {CAN_OBJECT_HANDLE_ID}		
Description	Holds the handle ID of HRH or HTH. The value of this parameter is unique in a given CAN Driver, and it should start with 0 and continue without any gaps. The HRH and HTH Ids are defined under two different name-spaces. Therefore the IDs of HRH/HTH can overlap.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanObjectType {CAN_OBJECT_TYPE}		
Description	Specifies if the HardwareObject is used as Transmit or as Receive object		
Multiplicity	1		
Type	EnumerationParamDef		
Range	Receive	Receive HOH	
	Transmit	Transmit HOH	
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	–	
	Post-build time	X	Variant 3
Scope / Dependency **No Included Containers**			

4.5.2.2.2 CanFilterMask

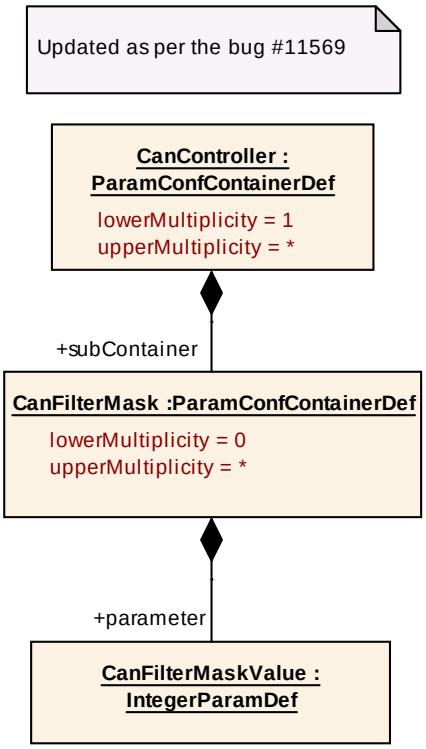


Figure 4.56: Can Filter Mask

CanFilterMask

SWS Item			
Container Name	CanFilterMask {CanFilterMask}		
Description	This container contains the configuration (parameters) of the CAN Filter Mask(s).		
Configuration Parameters			

Name last modified in: V1.0.0	CanFilterMaskValue {CAN_FILTER_MASK_VALUE} [CAN066]		
Description	Describes a mask for hardware-based filtering of CAN identifiers It shall be distinguished between - Standard identifier mask - Extended identifier mask.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	–	
	Post-build time	X	Variant 3
Scope / Dependency			
No Included Containers			

4.5.3 CAN Transport Layer

The CAN Transport Layer parameters are structured according to figure 4.57.

- CanTpGeneral in section 4.5.3.1
- CanTpRxNSdu in section 4.5.3.2
- CanTpTxNSdu in section 4.5.3.3
- CanTpNTa and CanTpNSa in section 4.5.3.4

The references to other BSW modules are described in section 4.5.3.5

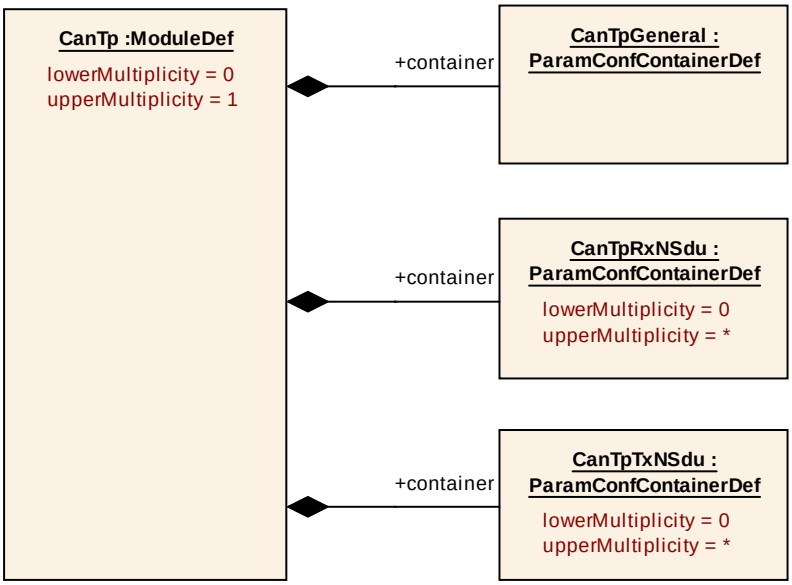


Figure 4.57: CAN Transport Protocol Parameter Overview

There are two variants defined for CAN Transport layer:

- A pre-compile time configuration.
- A mix of pre-compile and post-build loadable configuration.

Module Name	CanTp	
Module Description	Configuration of the CanTp (CAN Transport Protocol) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CanTpGeneral	1	This container contains the general configuration parameters of the CanTp module.
CanTpRxNSdu	0..*	The following parameters needs to be configured for each CAN N-SDU that the CanTp module shall receive.
CanTpTxNSdu	0..*	The following parameters needs to be configured for each CAN N-SDU that the CanTp module shall transmitt.
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.

4.5.3.1 CanTpGeneral

General parameters for CAN Transport Layer are described in figure 4.58.

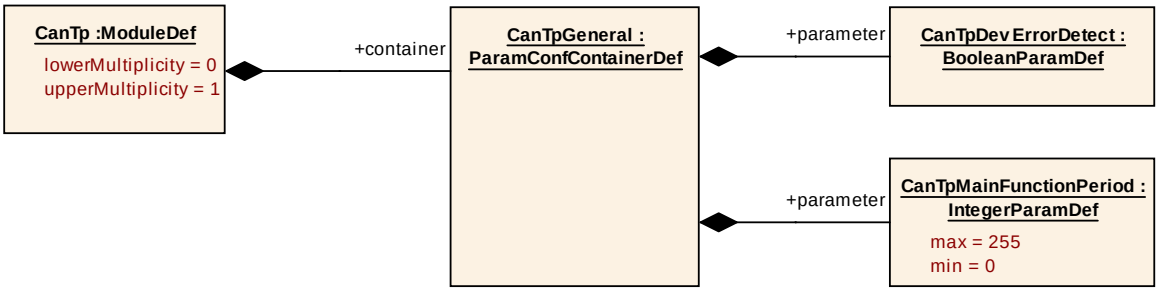


Figure 4.58: CAN Transport Layer general parameters

CanTpGeneral

SWS Item			
Container Name	CanTpGeneral {CanTpConfiguration}		
Description	This container contains the general configuration parameters of the CanTp module.		
Configuration Parameters			

Name last modified in: V1.0.0	CanTpDevErrorDetect {CANTP_DEV_ERROR_DETECT}		
Description	Switches the Development Error Detection and Notification ON or OFF		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	CanTpMainFunctionPeriod {CANTP_MAIN_FUNCTION_PERIOD}		
Description	Allow to configure the time for the MainFunction (in ms). Please note: This configuration value shall be equal to the value in the ScheduleManger module.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	Pre-Compile Configuration
	Link time	–	
	Post-build time	L	Pre-Compile and Post-build Configuration
Scope / Dependency	scope: ECU		
No Included Containers			

4.5.3.2 CanTpRxNSdu

Parameters for the receive part of the CAN Transport Layer are described in figure 4.59.

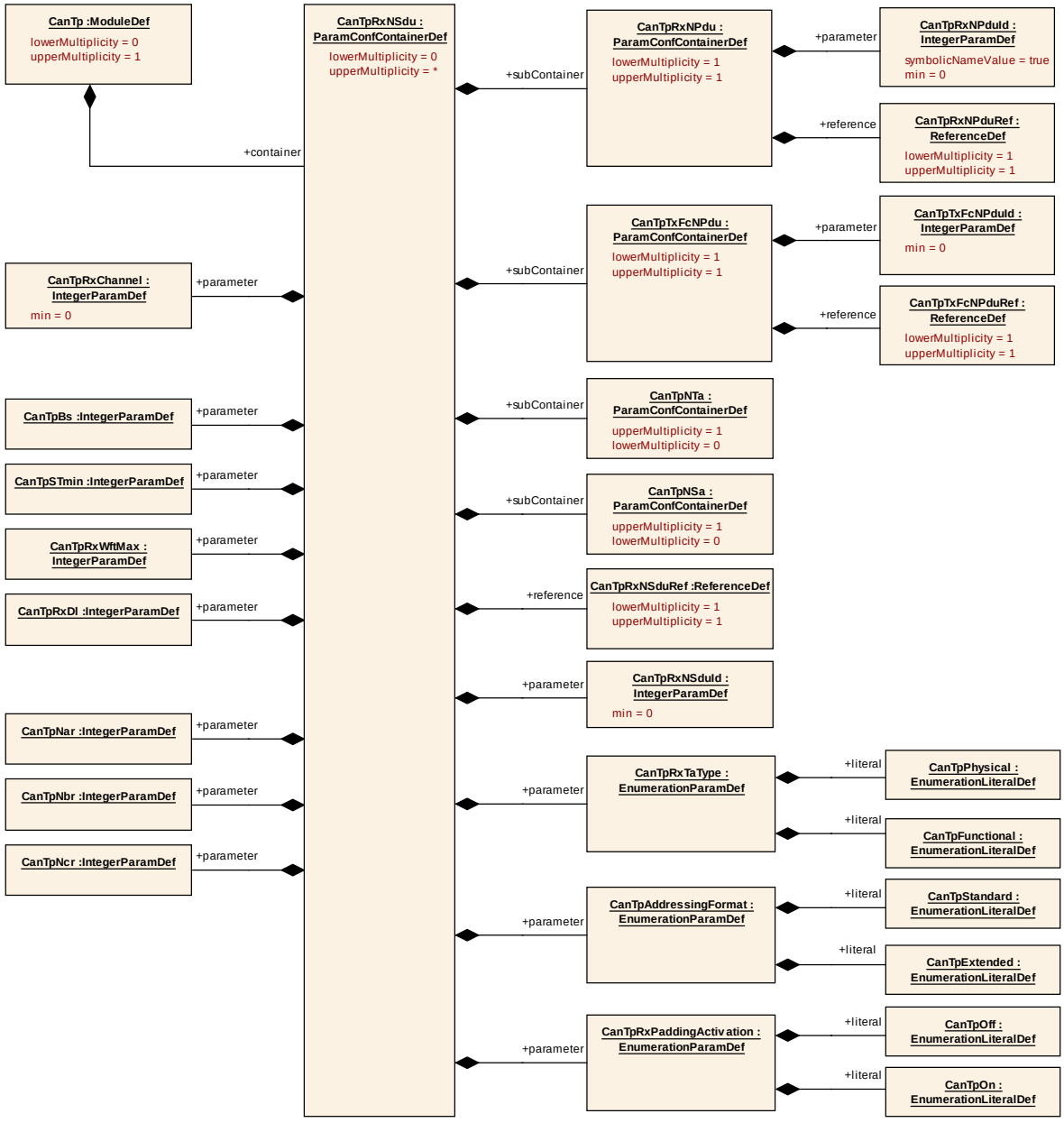


Figure 4.59: CAN Transport Layer RX parameters

CanTpRxNSdu

SWS Item	[CanTp137]		
Container Name	CanTpRxNSdu {RxNsdu}		
Description	The following parameters needs to be configured for each CAN N-SDU that the CanTp module shall receive.		
Configuration Parameters			
Name last modified in: V1.1.0	CanTpAddressingFormat {CANTP_ADDRESSING_FORMAT}		
Description	Declares which communication addressing mode is supported for this Rx N-SDU. Enum values: CanTpStandard. To use normal addressing format. CanTpExtended. To use extended addressing format.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	CanTpExtended		
	CanTpStandard		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		
Name last modified in: V1.0.0	CanTpBs {CANTP_BS}		
Description	Sets the maximum number of N-PDUs the CanTp receiver allows the sender to send, before waiting for an authorization to continue transmission of the following N-PDUs.For further details on this parameter value see ISO 15765-2 specification. Note: For reasons of buffer length, the CAN Transport Layer can adapt the BS value within the limit of this maximum BS		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Pre-Compile Configuration
	Link time	–	
	Post-build time	L	Pre-Compile and Post-build Configuration
Scope / Dependency	scope: module		
Name last modified in: V1.1.0	CanTpNar {CANTP_NAR}		
Description	Value in ms of the N_Ar timeout. N_Ar is the time for transmission of a CAN frame (any N_PDU) on the receiver side.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Pre-Compile Configuration
	Link time	–	
	Post-build time	L	Pre-Compile and Post-build Configuration
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	CanTpNbr {CANTP_NBR}		
Description	Value in ms of the performance requirement for (N_Br + N_Ar). N_Br is the elapsed time between the receiving indication of a FF or CF or the transmit confirmation of a FC, until the transmit request of the next FC.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Pre-Compile Configuration
	Link time	–	
	Post-build time	L	Pre-Compile and Post-build Configuration
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	CanTpNcr {CANTP_NCR}		
Description	Value in ms of the N_Cr timeout. N_Cr is the time until reception of the next Consecutive Frame N_PDU.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Pre-Compile Configuration
	Link time	–	
	Post-build time	L	Pre-Compile and Post-build Configuration
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	CanTpRxChannel {CANTP_RX_CHANNEL}		
Description	Link to the Rx connection channel, which has to be used for receiving this N-PDU.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	CanTpRxDl {CANTP_DL}		
Description	Data Length Code of this RxNsdu. In case of variable message length, this value indicates the minimum data length. Depending on SF or FF N-SDU the value will be limited to 7 (6 for an extended addressing format) and 4095 respectively.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	CanTpRxNSduId {CANTP_RXNSDU_ID}		
Description	Unique identifier to a structure that contains all useful information to process reception of a Rx N-SDU.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	CanTpRxNSduRef		
Description			
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	X	Pre-Compile Configuration
	Link time	X	Pre-Compile and Post-build Configuration
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	CanTpRxPaddingActivation {CANTP_PADDING_ACTIVATION}		
Description	Defines if the receive frame uses padding or not. Definition of enumeration values: CanTpOn: The N-PDU received uses padding for SF, FC and the last CF. (N-PDU length is always 8 bytes) CanTpOff: The N-PDU received does not use padding for SF, CF and the last CF. (N-PDU length is dynamic)		
Multiplicity	1		
Type	EnumerationParamDef		
Range	CanTpOff		
	CanTpOn		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	CanTpRxTaType {CANTP_TA_TYPE}		
Description	Declares the communication type of this Rx N-SDU.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	CanTpFunctional		
	CanTpPhysical		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	CanTpRxWftMax {CANTP_WFTMAX}		
Description	This parameter indicates how many Flow Control wait N-PDUs can be consecutively transmitted by the receiver. It is local to the node and is not transmitted inside the FC protocol data unit. CanTpRxWftMax is used to avoid sender nodes being potentially hooked-up in case of a temporarily reception inability on the part of the receiver nodes, whereby the sender could be waiting continuously.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Pre-Compile Configuration
	Link time	–	
	Post-build time	L	Pre-Compile and Post-build Configuration
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	CanTpSTmin {CANTP_STMIN}		
Description	Sets the duration of the minimum time the CanTp sender shall wait between the transmissions of two CF N-PDUs. For further details on this parameter value see ISO 15765-2 specification.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Pre-Compile Configuration
	Link time	–	
	Post-build time	L	Pre-Compile and Post-build Configuration
Scope / Dependency	scope: module		
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
CanTpNSa	0..1		
CanTpNTa	0..1		
CanTpRxNPdu	1		
CanTpTxFcNPdu	1		

CanTpRxNPdu

SWS Item	
Container Name	CanTpRxNPdu
Description	Used for grouping of the ID of a PDU and the Reference to a PDU.
Configuration Parameters	

Name last modified in: V1.1.0	CanTpRxNPduId {CANTP_RXNPDU_ID}		
Description	The N-PDU identifier attached to the RxNsdu is identified by CanTpRxNSduId. Each RxNsdu identifier is linked to only one SF/FF/CF N-PDU identifier. Nevertheless, in the case of extended addressing format, the same N-PDU identifier can be used for several N-SDU identifiers. The distinction is made by the N_TA value (first data byte of SF or FF frames).		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	CanTpRxNPduRef		
Description	Reference to a Pdu in the COM-Stack.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	X	Pre-Compile Configuration
	Link time	–	
	Post-build time	L	Pre-Compile and Post-build Configuration
Scope / Dependency			
No Included Containers			

CanTpTxFcNPdu

SWS Item			
Container Name	CanTpTxFcNPdu		
Description	Used for grouping of the ID of a PDU and the Reference to a PDU.		
Configuration Parameters			

Name last modified in: V1.1.0	CanTpTxFcNPduId {CANTP_TXFC_NPDU_ID}		
Description	The N-PDU identifier attached to the FC N-PDU of this RxNSdu is identified by CanTpRxNSduId. Each RxNSdu identifier is linked to only one Tx FC N-PDU identifier. But, in case of extended addressing mode, the same FC N-PDU identifier can be used for several N-SDU identifiers. The distinction is done on the N_TA value (first data byte of SF or FF frames).		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	CanTpTxFcNPduRef		
Description	Reference to a Pdu in the COM-Stack.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	X	Pre-Compile Configuration
	Link time	–	
	Post-build time	L	Pre-Compile and Post-build Configuration
Scope / Dependency			
No Included Containers			

4.5.3.3 CanTpTxNSdu

Parameters for the transmit part of the CAN Transport Layer are described in figure 4.60.

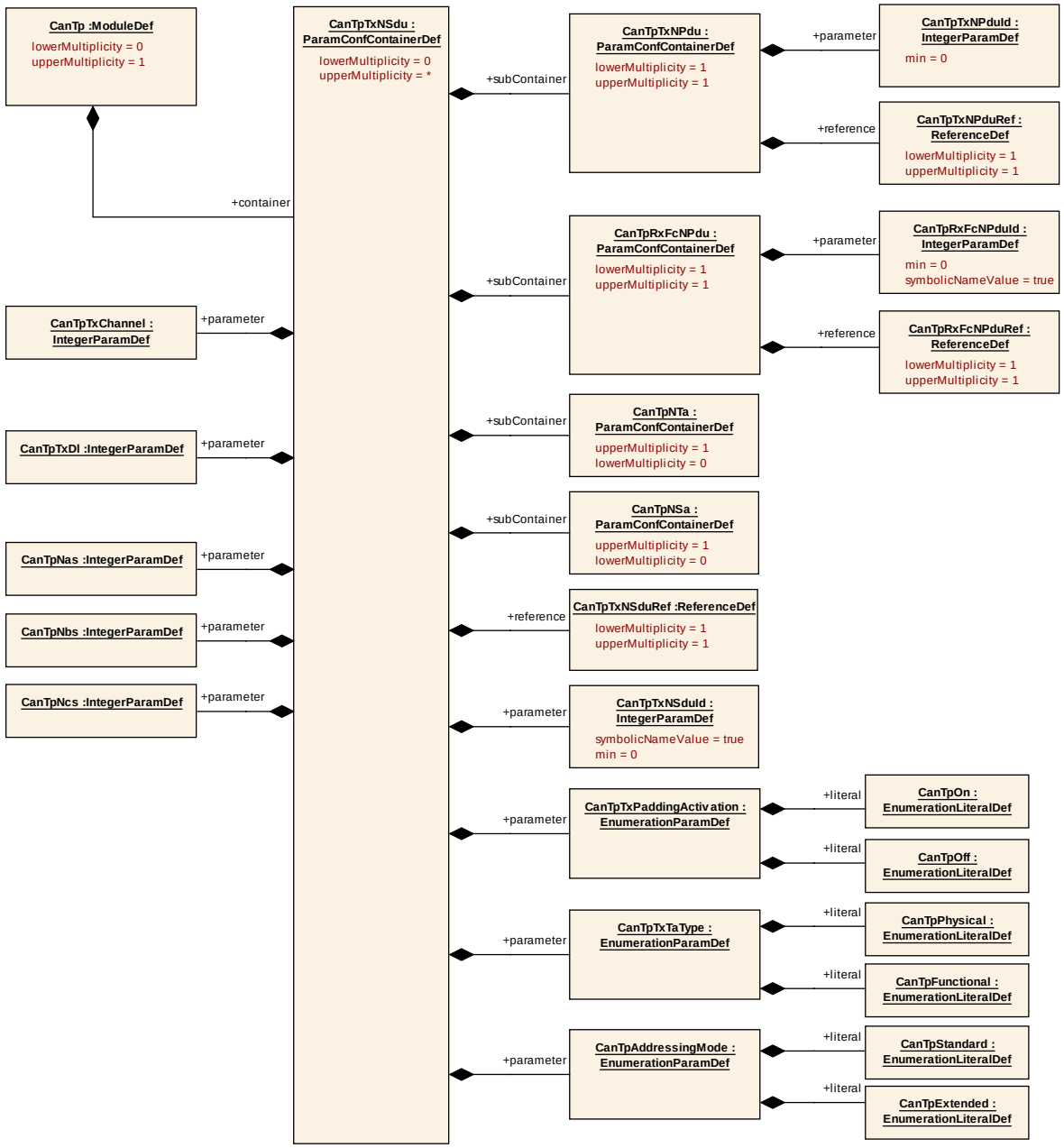


Figure 4.60: CAN Transport Layer TX parameters

CanTpTxNSdu

SWS Item	[CanTp138]		
Container Name	CanTpTxNSdu {TxNsdu}		
Description	The following parameters needs to be configured for each CAN N-SDU that the CanTp module shall transmitt.		
Configuration Parameters			
Name last modified in: V1.1.0	CanTpAddressingMode {CANTP_ADDRESSING_MODE}		
Description	Declares which communication addressing format is supported for this TxNsdu. Definition of Enumeration values: CanTpStandard to use normal addressing format. CanTpExtended to use extended addressing format (the N_TA container of this TxNsdu will be used).		
Multiplicity	1		
Type	EnumerationParamDef		
Range	CanTpExtended		
	CanTpStandard		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		
Name last modified in: V1.0.0	CanTpNas {CANTP_NAS}		
Description	Value in ms of the N_As timeout. N_As is the time for transmission of a CAN frame (any N_PDU) on the part of the sender.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Pre-Compile Configuration
	Link time	–	
	Post-build time	L	Pre-Compile and Post-build Configuration
Scope / Dependency	scope: module		
Name last modified in: V1.0.0	CanTpNbs {CANTP_NBS}		
Description	Value in ms of the N_Bs timeout. N_Bs is the time of transmission until reception of the next Flow Control N_PDU.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Pre-Compile Configuration
	Link time	–	
	Post-build time	L	Pre-Compile and Post-build Configuration
Scope / Dependency	scope: module		
Name last modified in: V1.0.0	CanTpNcs {CANTP_NCS}		
Description	Value in ms of the performance requirement of (N_Cs + N_As). N_Cs is the time which elapses between the transmit request of a CF N-PDU until the transmit request of the next CF N-PDU.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Pre-Compile Configuration
	Link time	–	
	Post-build time	L	Pre-Compile and Post-build Configuration
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	CanTpTxChannel {CANTP_TX_CHANNEL}		
Description	Link to the connection channel which has to be used for transmission of this N-PDU.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	CanTpTxDl {CANTP_DL}		
Description	Data Length Code of this TxNsdu. In case of variable length message, this value indicates the minimum data length.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	CanTpTxNSduId {CANTP_TXNSDU_ID}		
Description	Unique identifier to a structure that contains all useful information to process the transmission of a TxNsdu.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	CanTpTxNSduRef		
Description	Reference to a Pdu in the COM-Stack.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	X	Pre-Compile Configuration
	Link time	X	Pre-Compile and Post-build Configuration
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	CanTpTxPaddingActivation {CANTP_PADDING_ACTIVATION}		
Description	Defines if the transmit frame use padding or not. Definition of Enumeration values: CanTpOn The transmit N-PDU uses padding for SF, FC and the last CF. (N-PDU length is always 8 bytes) CanTpOff The transmit N-PDU does not use padding for SF, CF and the last CF. (N-PDU length is dynamic)		
Multiplicity	1		
Type	EnumerationParamDef		
Range	CanTpOff		
	CanTpOn		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	CanTpTxTaType {CANTP_TA_TYPE}		
Description	Declares the communication type of this TxNsdu. Enumeration values: CanTpPhysical. Used for 1:1 communication. CanTpFunctional. Used for 1:n communication.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	CanTpFunctional		
	CanTpPhysical		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Included Containers

Container Name	Multiplicity	Scope / Depedency
CanTpNSa	0..1	
CanTpNTa	0..1	
CanTpRxFcNPdu	1	
CanTpTxNPdu	1	

CanTpTxNPdu

SWS Item			
Container Name	CanTpTxNPdu		
Description	Used for grouping of the ID of a PDU and the Reference to a PDU.		
Configuration Parameters			

Name last modified in: V1.1.0	CanTpTxNPduId {CANTP_TXNPDU_ID}		
Description	N-PDU identifier attached to the TxNdsu identified by CanTpTxNSduId. Each TxNsdu identifier is linked to one SF/FF/CF N-PDU identifier only. However, in the case of extended addressing format, the same N-PDU identifier can be used for several N-SDU identifiers. The distinction is made by means of the N_TA value (first data byte of SF or FF frames).		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	CanTpTxNPduRef		
Description	Reference to a Pdu in the COM-Stack.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	X	Pre-Compile Configuration
	Link time	–	
	Post-build time	X	Pre-Compile and Post-build Configuration
Scope / Dependency			
No Included Containers			

CanTpRxFcNPdu

SWS Item			
Container Name	CanTpRxFcNPdu		
Description	Used for grouping of the ID of a PDU and the Reference to a PDU.		
Configuration Parameters			

Name last modified in: V1.1.0	CanTpRxFcNPduId {CANTP_RXFC_NPDU_ID}		
Description	N-PDU identifier attached to the FC N-PDU of this TxNsdu identified by CanTpTxNSduId. Each TxNsdu identifier is linked to one Rx FC N-PDU identifier only. However, in the case of extended addressing format, the same FC N-PDU identifier can be used for several N-SDU identifiers. The distinction is made by means of the N_TA value (first data byte of FC frames).		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	CanTpRxFcNPduRef		
Description	Reference to a Pdu in the COM-Stack.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	X	Pre-Compile Configuration
	Link time	–	
	Post-build time	L	Pre-Compile and Post-build Configuration
Scope / Dependency			
No Included Containers			

4.5.3.4 CanTpNTa and CanTpNSa

If a RxNsdu or a TxNsdu is configured for extended addressing format, CanTpNTa and CanTpNSa defines the target address's. See figure 4.61.

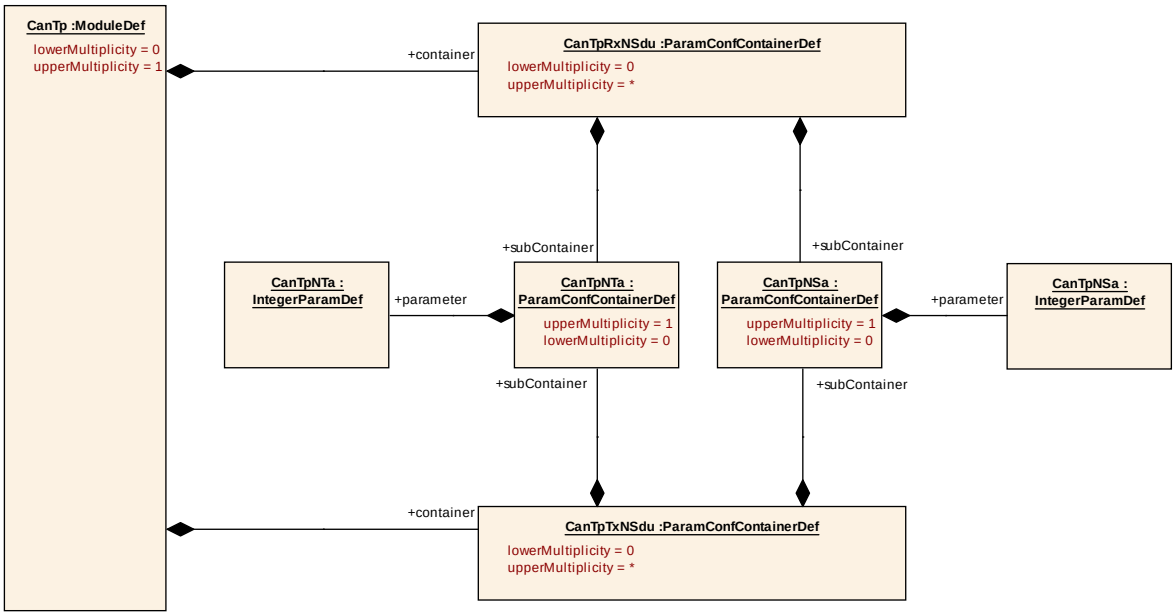


Figure 4.61: Extended Addressing format in CAN Transport Layer.

CanTpNTa

SWS Item	[CanTp139]
Container Name	CanTpNTa {N_Ta}
Description	The following parameters need to be configured for each RxNsdu or TxNsdu with the CanTpAddressingFormat set to CanTpExtended.
Configuration Parameters	

Name last modified in: V1.0.0	CanTpNTa {CANTP_NTA}		
Description	If an RxNsdu or a TxNsdu is configured for extended addressing format, this parameter contains the transport protocol target address's value.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		
No Included Containers			

CanTpNSa

SWS Item			
Container Name	CanTpNSa {N_Sa}		
Description	Contains the parameters needed to configure each RxNSdu or TxNSdu with CanTpAddressingFormat set to CanTpExtended.		
Configuration Parameters			
Name last modified in: V1.1.0	CanTpNSa {CANTP_NSA}		
Description	If an RxNSdu ora TxNSdu is configured for extended addressing format, this parameter contains the transport protocol source address's value.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

4.5.3.5 CanTpReferences

The references to other modules are shown in figure 4.62.

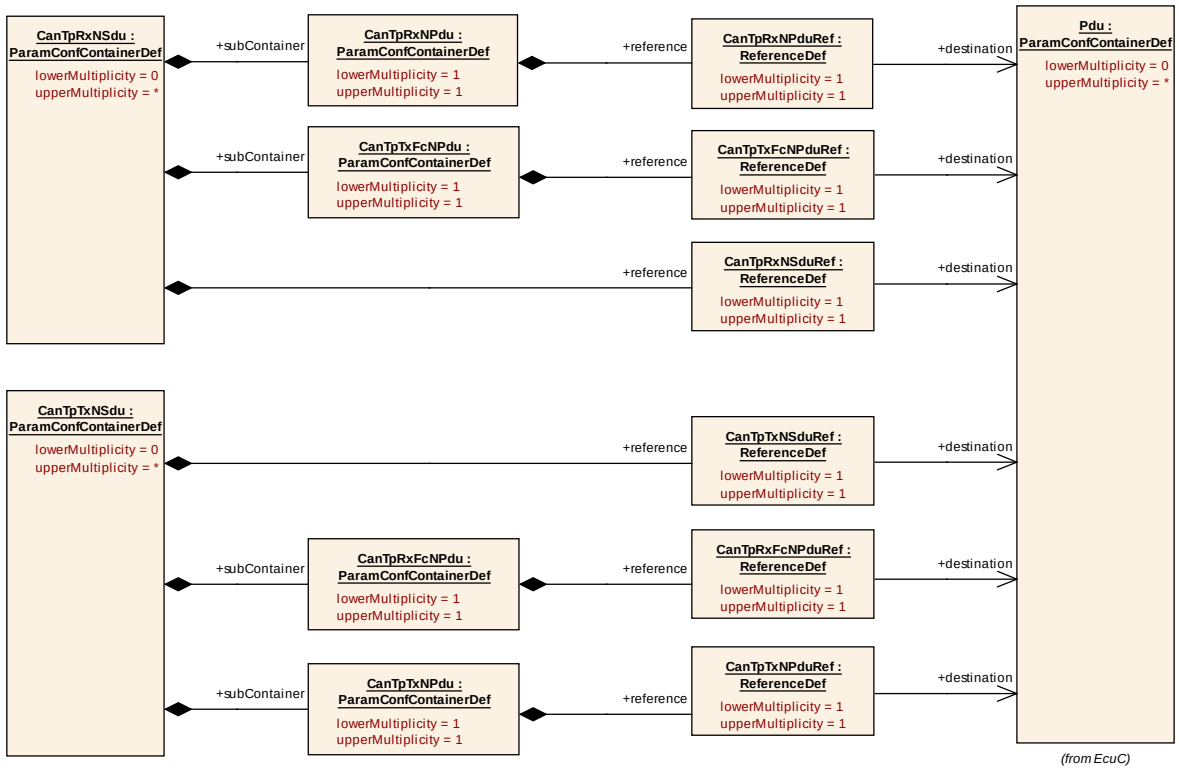


Figure 4.62: CAN Transport Layer Parameter References

4.5.4 CAN Transceiver Driver

The CAN Transceiver Driver configuration parameters are structured according to figure 4.63.

- CanTrcvGeneral in section 4.5.4.1
- CanTrcvChannel in section 4.5.4.2
- CanTrcvAccess in section 4.5.4.3

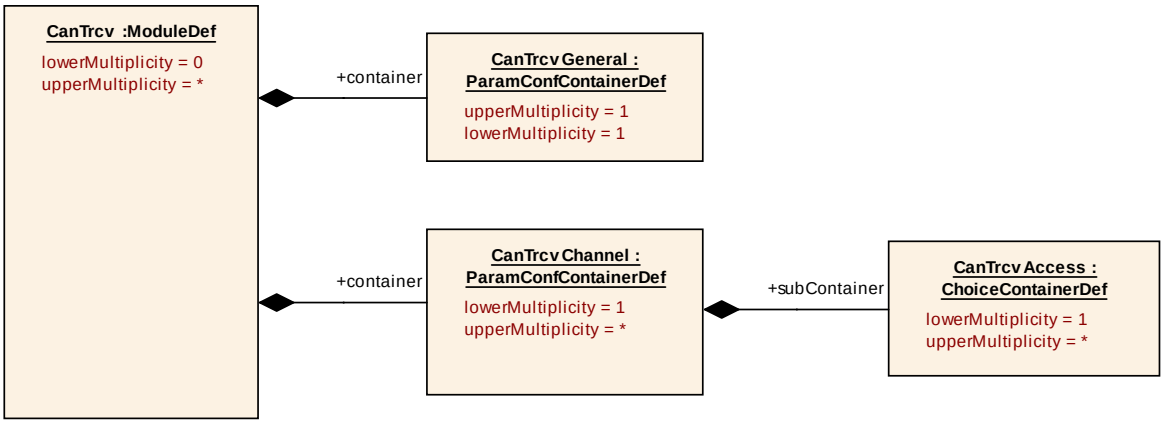


Figure 4.63: CAN Transceiver Driver Parameters

Module Name	CanTrcv	
Module Description	Configuration of the CanTrcv (CAN Transceiver driver) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CanTrcvChannel	1..*	Container gives CAN transceiver driver information about a single CAN transceiver channel. Any CAN transceiver driver has such CAN transceiver channels.
CanTrcvGeneral	1	Container gives CAN transceiver driver basic information.
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.

4.5.4.1 CanTrcvGeneral

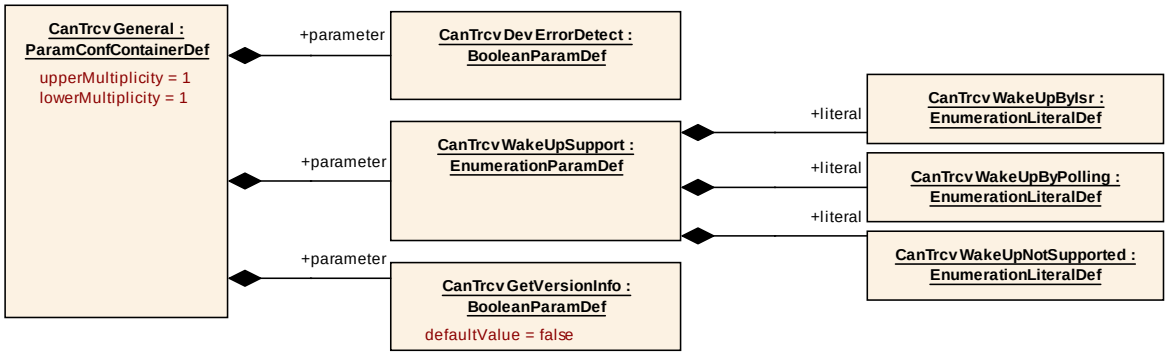


Figure 4.64: CAN Transceiver Driver General Parameters

CanTrcvGeneral

SWS Item	[CanTrcv090]		
Container Name	CanTrcvGeneral {CanTransceiverDriverBasic}		
Description	Container gives CAN transceiver driver basic information.		
Configuration Parameters			
Name last modified in: V1.0.0	CanTrcvDevErrorDetect {CANTRCV_DEV_ERROR_DETECT}		
Description	Switches development error detection and notification on and off. If switched on, #define CANTRCV_DEV _ERROR_DETECT ON shall be generated. If switched off, #define CANTRCV_DEV_ERROR _DETECT OFF shall be generated. Define shall be part of file CanTrcv_Cfg.h.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	CanTrcvGetVersionInfo {CANTRCV_GET_VERSION_INFO}		
Description	Switches version information API on and off. If switched off, function need not be present in compiled code.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	CanTrcvWakeUpSupport {CANTRCV_GENERAL_WAKE_UP_SUPPORT}		
Description	Informs whether wake up is supported by ISR, by polling or whether it is not supported. In case no wake up is supported by CAN transceiver hardware setting has to be always NO. Only in case wake up is supported by polling main function CanTrcv_main has to be present in source code and called by CanIf. Only in case wake up is supported by ISR callback function CanTrcv_CB_WakeupByBus has to be present in source code. In case of support for wake up either by ISR or by polling wake up ability may be switched on or off for each channel of one CAN transceiver channel independently by CanTrcvWakeupByBusUsed.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	CanTrcvWakeUpByIsr		
	CanTrcvWakeUpByPolling		
	CanTrcvWakeUpNot Supported		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module dependency: CanTrcvWakeupByBusUsed		
No Included Containers			

4.5.4.2 CanTrcvChannel

Each CAN transceiver driver supports one or several channels.

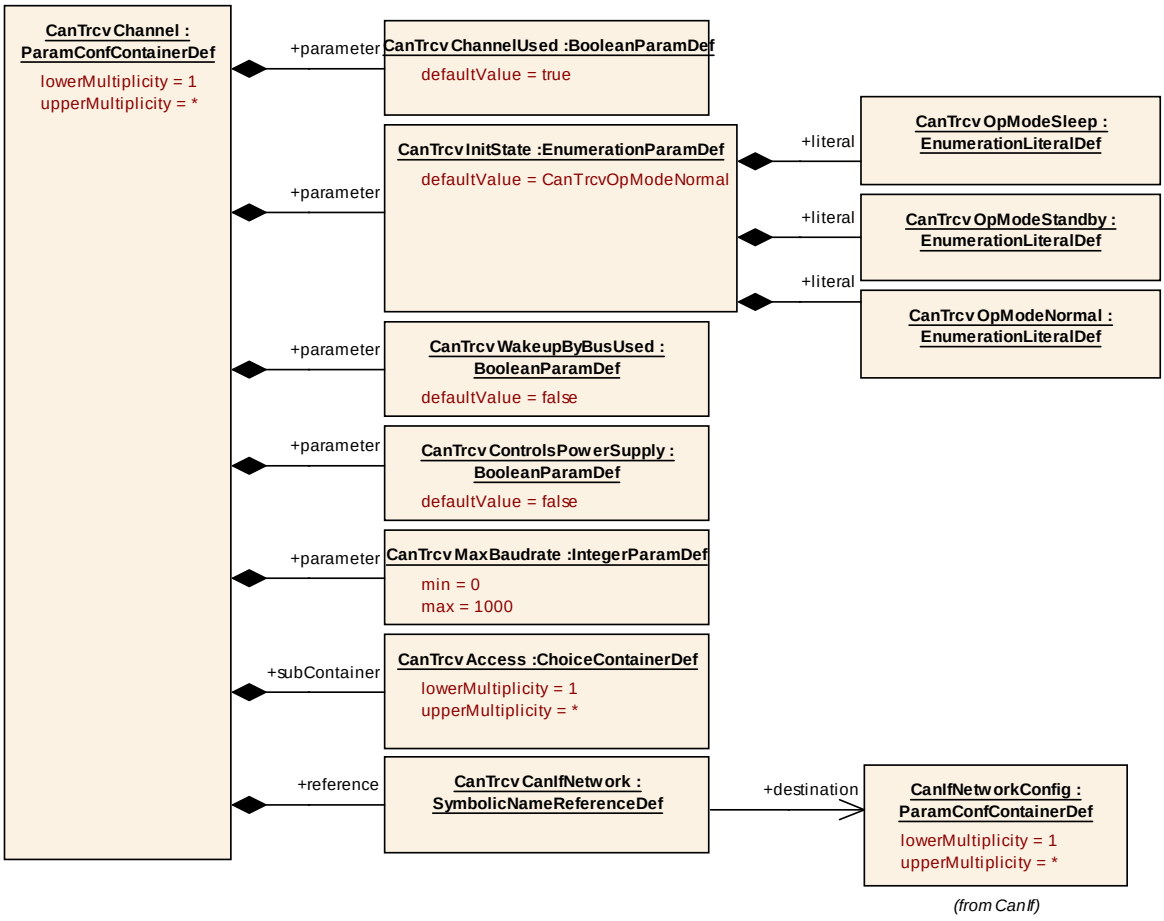


Figure 4.65: CAN Transceiver Driver Channel Parameters

CanTrcvChannel

SWS Item	[CanTrcv091]
Container Name	CanTrcvChannel {CanTranceiverChannels}
Description	Container gives CAN transceiver driver information about a single CAN transceiver channel. Any CAN transceiver driver has such CAN transceiver channels.
Configuration Parameters	

Name last modified in: V1.1.0	CanTrcvCanIfNetwork {CANTRCV_CANIF_NETWORK}		
Description	Unique network number. It is used by ComM by Icu and internally. Reference to a CanIf configuration item CANIF_NETWORK.		
Multiplicity	1		
Type	Symbolic name reference to CanIfNetworkConfig		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Instance		

Name last modified in: V1.1.0	CanTrcvChannelUsed {CANTRCV_CHANNEL_USED}		
Description	Shall the related CAN transceiver channel be used?		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Instance		

Name last modified in: V1.1.0	CanTrcvControlsPowerSupply {CANTRCV_CONTROLS_POWER_SUPPLY}		
Description	Is ECU power supply controlled by this transceiver? TRUE = Controlled by transceiver. FALSE = Not controlled by transceiver.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Instance		

Name last modified in: V1.1.0	CanTrcvInitState {CANTRCV_INIT_STATE}		
Description	State of CAN transceiver after call to CanTrcv_Init.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	CanTrcvOpModeNormal		
	CanTrcvOpModeSleep		
	CanTrcvOpModeStandby		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Instance		

Name last modified in: V1.1.0	CanTrcvMaxBaudrate {CANTRCV_MAX_BAUDRATE}		
Description	Max baudrate for transceiver hardware type. Only used for validation purposes. Value shall be configured by configuration tool based on transceiver hardware type.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 1000		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Instance		

Name last modified in: V1.1.0	CanTrcvWakeupByBusUsed {CANTRCV_WAKEUP_BY_BUS_USED}		
Description	Is wake up by bus supported? If CAN transceiver hardware does not support wake up by bus value is always FALSE. If CAN transceiver hardware supports wake up by bus value is TRUE or FALSE depending whether it is used or not. TRUE = Is used. FALSE = Is not used.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Instance dependency: CanTrcvWakeUpSupport		

Included Containers

Container Name	Multiplicity	Scope / Depedency
CanTrcvAccess [CanTrcvDioAccess, CanTrcvSpiSequence]	1..*	

4.5.4.3 CanTrcvAccess

The CAN transceiver can be connected to a DIO port or a SPI interface. The references to parameters in other modules are found in figure 4.67.

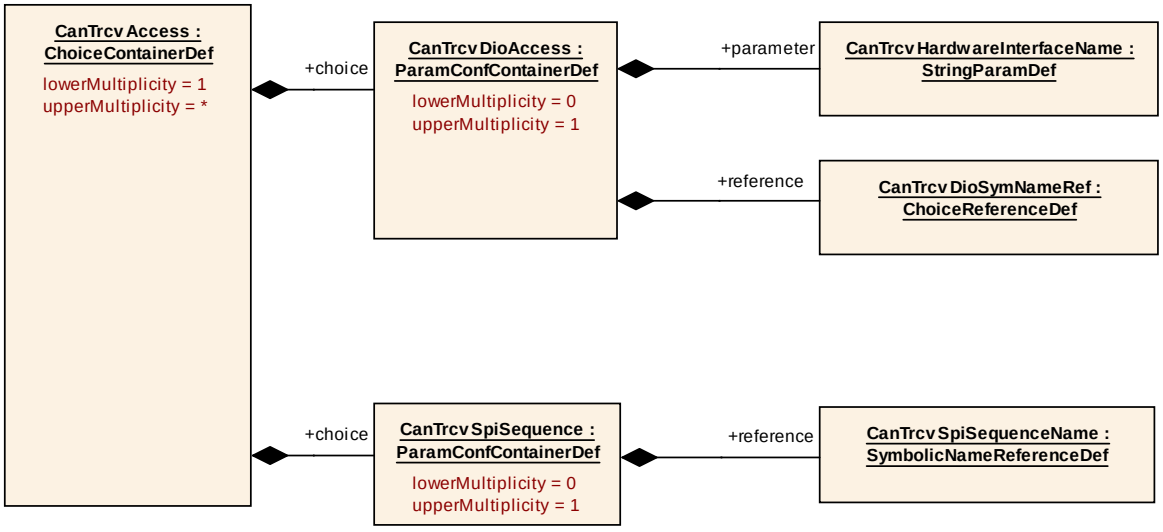


Figure 4.66: CAN Transceiver Driver Access Parameters

CanTrcvDioAccess

SWS Item	[CanTrcv094]		
Container Name	CanTrcvDioAccess {CanTransceiverDioAccess}		
Description	Container gives CAN transceiver driver information about accessing ports and port pins. In addition relation between CAN transceiver hardware pin names and Dio port access information is given. If a CAN transceiver hardware has no Dio interface, there is no instance of this container.		
Configuration Parameters			

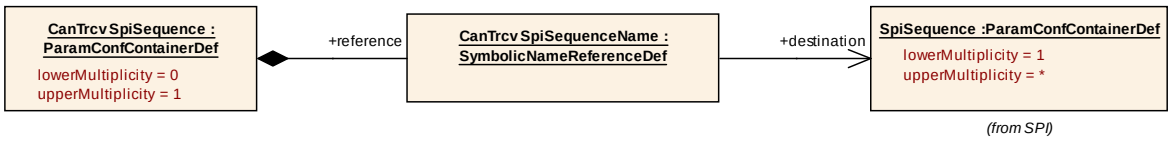
Name last modified in: V1.1.0	CanTrcvDioSymNameRef		
Description	Choice Reference to a DIO Port, DIO Channel or DIO Channel Group. This reference replaces the CANTRCV_DIO_PORT_SYM_NAME, CANTRCV_DIO_CHANNEL_SYM_NAME and CANTRCV_DIO_GROUP_SYM_NAME references in the Can Trcv SWS.		
Multiplicity	1		
Type	Choice Reference to [DioChannel, DioPort, DioChannelGroup]		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CanTrcvHardwareInterfaceName {CANTRCV_HARDWARE_INTERFACE_NAME}		
Description	CAN transceiver hardware interface name. It is typically the name of a pin. From a Dio point of view it is either a port, a single channel or a channel group. Depending on this fact either CANTRCV_DIO_PORT_SYMBOLIC_NAME or CANTRCV_DIO_CHANNEL_SYMBOLIC_NAME or CANTRCV_DIO_CHANNEL_GROUP_SYMBOLIC_NAME shall reference a Dio configuration. The CAN transceiver driver implementation description shall list up this name for the appropriate CAN transceiver hardware.		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Instance		
No Included Containers			

CanTrcvSpiSequence

SWS Item	[CanTrcv092]		
Container Name	CanTrcvSpiSequence {CanTransceiverSPISequences}		
Description	Container gives CAN transceiver driver information about one SPI sequence. One SPI sequence used by CAN transceiver driver is in exclusive use for it. No other driver is allowed to access this sequence. CAN transceiver driver may use one sequence to access n CAN transceiver hardwares chips of the same type or n sequences are used to access one single CAN transceiver hardware chip. If a CAN transceiver hardware has no SPI interface, there is no instance of this container.		
Configuration Parameters			

Name last modified in: V1.0.0	CanTrcvSpiSequenceName {CANTRCV_SPI_SEQUENCE_NAME}		
Description	Reference to a Spi sequence configuration container.		
Multiplicity	1		
Type	Symbolic name reference to SpiSequence		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Instance dependency: SpiSequence		
No Included Containers			



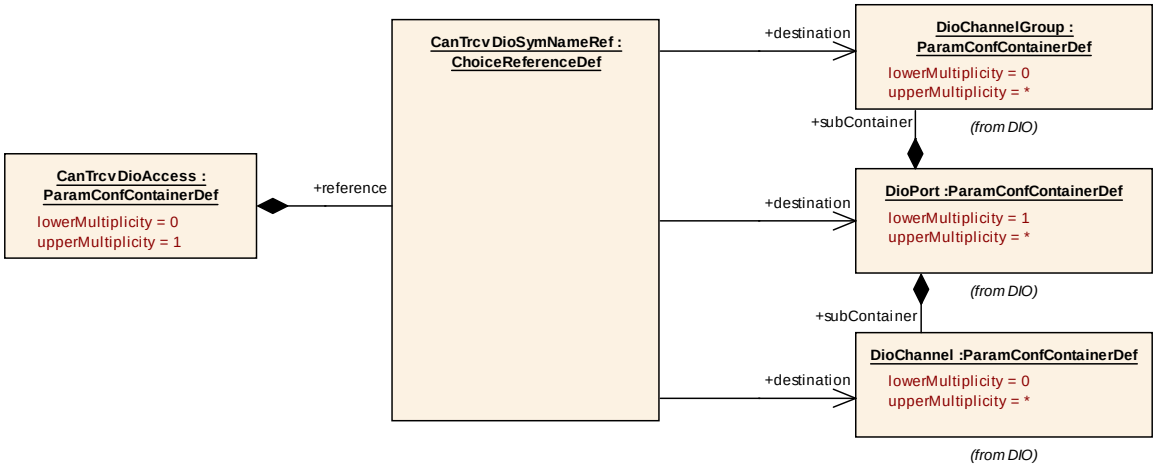


Figure 4.67: CAN Transceiver Driver Parameter References

4.6 FlexRay

The Flexray stack

4.6.1 FlexRay Interface

The FlexRay interface (FrIf) configuration is specified in [28].

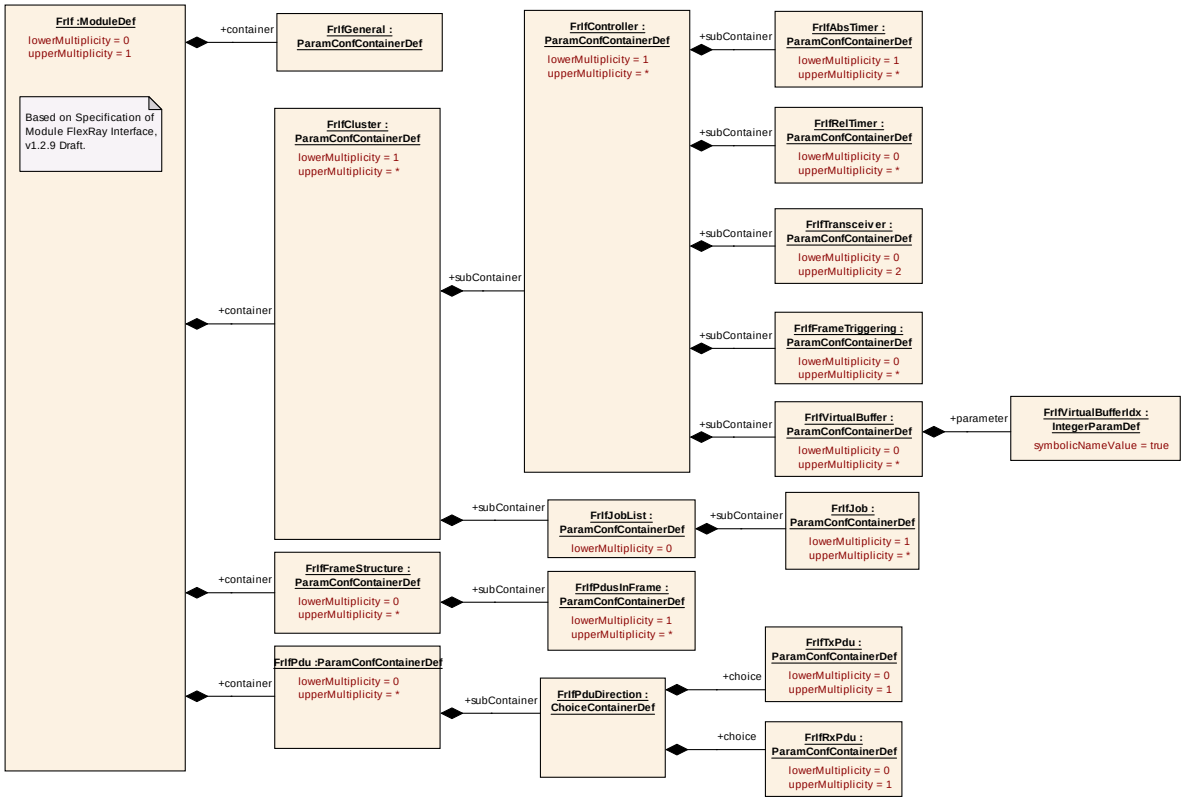


Figure 4.68: FlexRay Interface top level structure

Module Name	FrIf	
Module Description	Configuration of the FrIf (FlexRay Interface) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
FrIfCluster	1..*	This container specifies a FrIf Cluster and all related data which is required to enable communication of the Cluster. A Cluster may consist of more than one Controller.
FrIfFrameStructure	0..*	The Frame structure specifies a Construction Plan how a Frame is assembled with PDUs and their respective Update-Bits.
FrIfGeneral	1	This container contains the general configuration of the FlexRay Interface.
FrIfPdu	0..*	Contains PDU information. A PDU may be either a transmission PDU or a reception PDU. So, this container holds exactly one sub container.

Figure 4.69 depicts the general configuration parameters of FrIf.

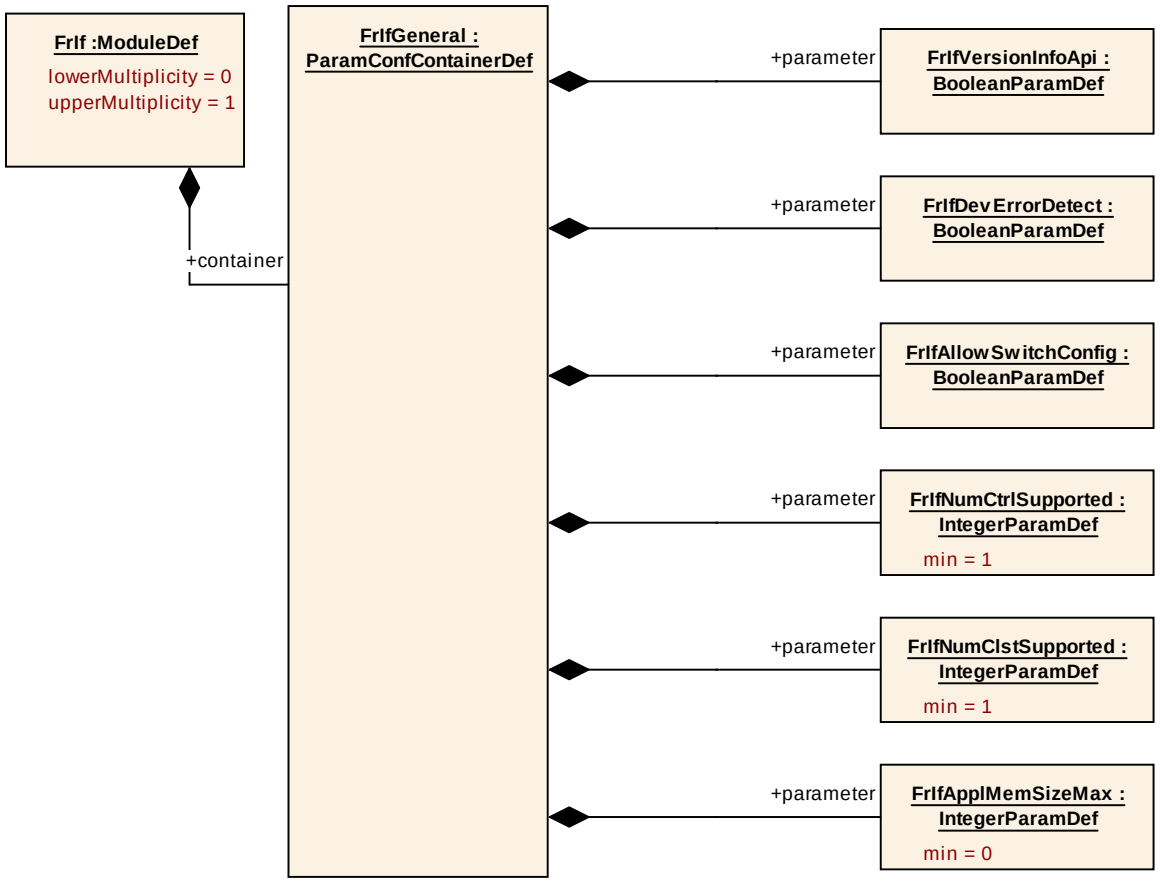


Figure 4.69: FlexRay Interface general configuration parameters

FrIfGeneral

SWS Item	[FrIf05360]		
Container Name	FrIfGeneral		
Description	This container contains the general configuration of the FlexRay Interface.		
Configuration Parameters			

Name last modified in: V1.0.0	FrIfAllowSwitchConfig		
Description	Enables/disables the existence of the FrIf_SwitchConfig() API service. In AUTOSAR R2.0 this parameter has to be set to OFF.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	FrIfApplMemSizeMax		
Description	The maximum RAM memory size to be used by the FrIf for variable data, i.e. storage of flags, states, and temporary data. If the FrIf is configured at post build time, it is not allowed to use more RAM memory than specified by this parameter.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	X	All Variants
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	FrIfDevErrorDetect		
Description	Switches the Development Error Detection and Notification on or off		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	FrIfNumClstSupported		
Description	Maximum number of FlexRay Clusters that the FlexRay Interface supports.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	X	All Variants
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	FrIfNumCtrlSupported		
Description	Maximum number of FlexRay CCs that the FlexRay Interface supports		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	X	All Variants
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	FrIfVersionInfoApi		
Description	Enables/disables the existence of the FrIf_GetVersionInfo() API service		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
No Included Containers			

4.6.1.1 FlexRay Cluster

The FlexRay Cluster configuration defines a lot of low level parameters as shown in figure 4.70. The low level parameters are configured as part of the `FrIf`, since they must be the same for all communication controllers used on that cluster. Note, however, that all parameters with scope *Module/Fr* will be used by the FlexRay drivers. The code generators for the drivers need to retrieve this information from the `FrIf` configuration. Likewise, the `FrIf` code generator will ignore this information.

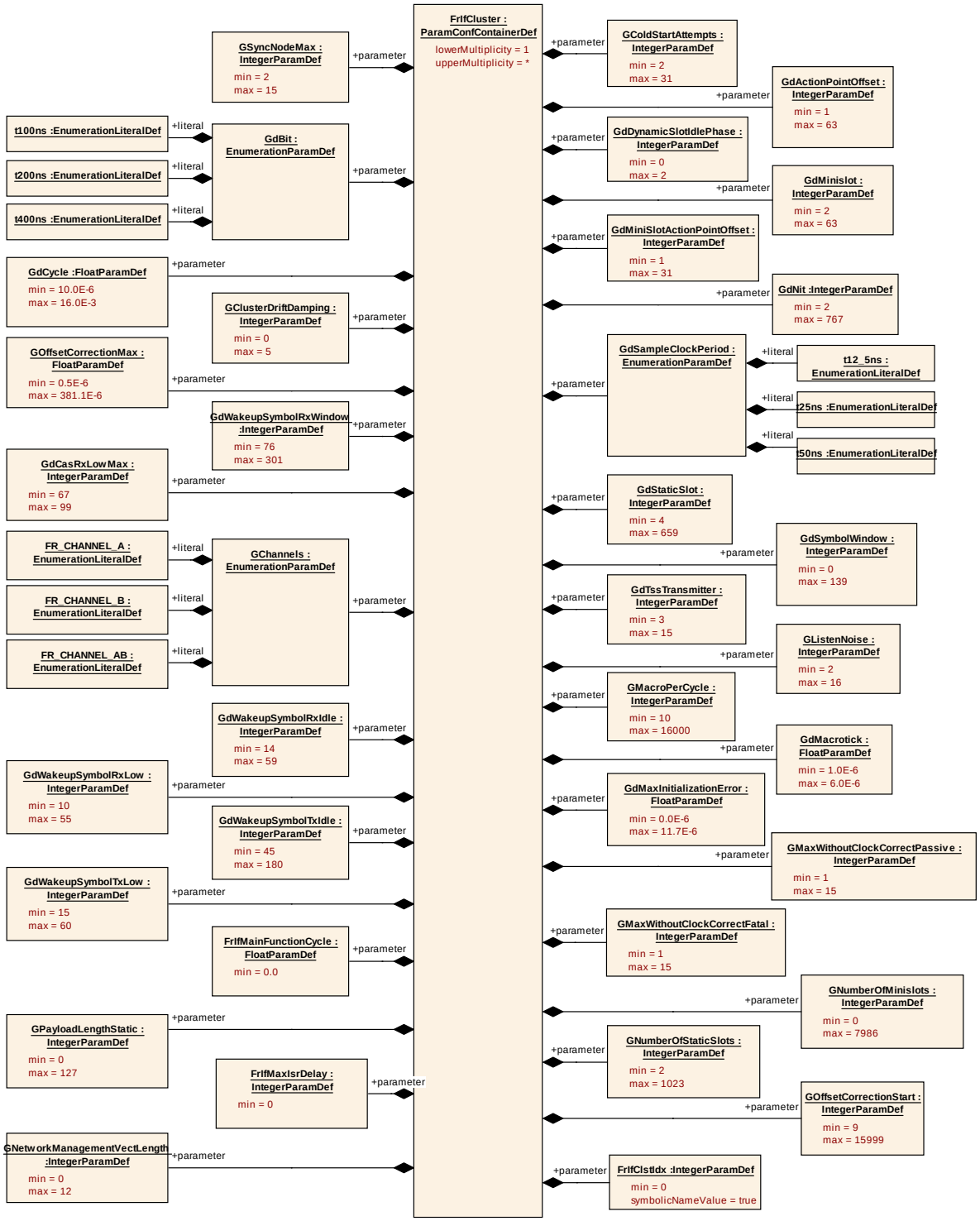


Figure 4.70: FlexRay Cluster configuration: low level parameters

The cluster configuration also contains the configuration of the individul controllers that pare part of the cluster (see section 4.6.1.2), and the configuration of the job list, see section 4.6.1.3.

FrIfCluster

SWS Item	[FrIf05366]		
Container Name	FrIfCluster		
Description	This container specifies a FrIf Cluster and all related data which is required to enable communication of the Cluster. A Cluster may consist of more than one Controller.		
Configuration Parameters			

Name last modified in: V1.0.0	FrIfClstIdx		
Description	This parameter provides a zero-based consecutive index of the FlexRay Clusters. Upper layer BSW modules and the FrIf itself use this index to identify a FlexRay Cluster.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	FrIfMainFunctionCycle		
Description	The execution cycle of the FrIf_MainFunction_<cluster>() in seconds. The FrIf does not require this information but the BSW scheduler, which invokes the cluster main functions, needs it in order to plan its tasks.		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	FrIfMaxIsrDelay		
Description	The maximum delay in macroticks the FrIf_JoblistExec_<cluster>() function is processed after the absolute timer interrupt was triggered.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	GChannels		
Description	The channels that are used by the cluster		
Multiplicity	1		
Type	EnumerationParamDef		
Range	FR_CHANNEL_A	Cluster uses channel A	
	FR_CHANNEL_AB	Cluster uses channel A and B	
	FR_CHANNEL_B	Cluster uses channel B	
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GClusterDriftDamping		
Description	The cluster drift damping factor, based on the longest microtick gdMaxMicrotick used in the cluster. Used to compute the local cluster drift damping factor pClusterDriftDamping [Micrroticks].		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 5		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GColdStartAttempts		
Description	Maximum number of times a node in the cluster is permitted to attempt to start the cluster by initiating schedule synchronization		
Multiplicity	1		
Type	IntegerParamDef		
Range	2 .. 31		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GListenNoise		
Description	Upper limit for the start up listen timeout and wake up listen timeout in the presence of noise. It is used as a multiplier of the cluster parameter pdListenTimeout.		
Multiplicity	1		
Type	IntegerParamDef		
Range	2 .. 16		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GMacroPerCycle		
Description	Number of macroticks in a communication cycle.		
Multiplicity	1		
Type	IntegerParamDef		
Range	10 .. 16000		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GMaxWithoutClockCorrectFatal		
Description	Threshold used for testing the vClockCorrectionFailed counter. Defines the number of consecutive even/odd Cycle pairs with missing clock correction terms that will cause the protocol to transition from the POC:normal active or POC:normal passive state into the POC:halt state. [Even/odd cycle pairs].		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 15		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GMaxWithoutClockCorrectPassive		
Description	Threshold used for testing the vClockCorrectionFailed counter. Defines the number of consecutive even/odd Cycle pairs with missing clock correction terms that will cause the protocol to transition from the POC:normal active state to the POC:normal passive state. [Even/Odd cycle pairs]		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 15		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GNetworkManagementVectLength		
Description	Length of the Network Management vector in a cluster [bytes]		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 12		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	GNumberOfMinislots		
Description	Number of minislots in the dynamic segment		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 7986		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GNumberOfStaticSlots {G_NUMBER_OF_STATIC_SLOTS}		
Description	Number of static slots in the static segment		
Multiplicity	1		
Type	IntegerParamDef		
Range	2 .. 1023		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GOffsetCorrectionMax		
Description	describes the maximum value which the offset correction should assume in seconds.		
Multiplicity	1		
Type	FloatParamDef		
Range	0.5E-6 .. 381.1E-6		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GOffsetCorrectionStart {G_OFFSET_CORRECTION_START}		
Description	Start of the offset correction phase within the NIT, expressed as the number of macroticks from the start of cycle.		
Multiplicity	1		
Type	IntegerParamDef		
Range	9 .. 15999		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GPayloadLengthStatic		
Description	Payload length of a static frame [16 bit words]		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 127		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	GSyncNodeMax		
Description	Maximum number of nodes that may send frames with the sync frame indicator bit set to one.		
Multiplicity	1		
Type	IntegerParamDef		
Range	2 .. 15		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GdActionPointOffset		
Description	Number of Macroticks the action point is offset from the beginning of a Static Slots or symbol window.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 63		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GdBit		
Description	Nominal bit time		
Multiplicity	1		
Type	EnumerationParamDef		
Range	t100ns		
	t200ns		
	t400ns		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GdCasRxLowMax		
Description	Upper limit of the CAS acceptance window [gdBit]		
Multiplicity	1		
Type	IntegerParamDef		
Range	67 .. 99		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GdCycle		
Description	Length of the cycle, expressed in s		
Multiplicity	1		
Type	FloatParamDef		
Range	10.0E-6 .. 16.0E-3		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GdDynamicSlotIdlePhase		
Description	Duration of the idle phase within a dynamic slot [Minislots].		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 2		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GdMacrotick		
Description	Duration of the cluster wide nominal macrotick, expressed in s		
Multiplicity	1		
Type	FloatParamDef		
Range	1.0E-6 .. 6.0E-6		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GdMaxInitializationError		
Description	Maximum error that a node may have following integration in seconds.		
Multiplicity	1		
Type	FloatParamDef		
Range	0.0E-6 .. 11.7E-6		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GdMiniSlotActionPointOffset		
Description	Number of Macroticks the Minislot action point is offset from the beginning of a Minislot [Macroticks].		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 31		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GdMinislot		
Description	Duration of a minislot [Macroticks]		
Multiplicity	1		
Type	IntegerParamDef		
Range	2 .. 63		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GdNit		
Description	Duration of the Network Idle Time [Macroticks]		
Multiplicity	1		
Type	IntegerParamDef		
Range	2 .. 767		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GdSampleClockPeriod		
Description	Sample clock period		
Multiplicity	1		
Type	EnumerationParamDef		
Range	t12_5ns		
	t25ns		
	t50ns		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GdStaticSlot		
Description	Duration of a Static Slot [Macroticks].		
Multiplicity	1		
Type	IntegerParamDef		
Range	4 .. 659		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GdSymbolWindow		
Description	Duration of the symbol window [Macroticks].		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 139		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GdTssTransmitter		
Description	Number of bits in the Transmission Start Sequence [gdBits].		
Multiplicity	1		
Type	IntegerParamDef		
Range	3 .. 15		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GdWakeupSymbolRxIdle		
Description	Number of bits used by the node to test the duration of the 'idle' portion of a received wakeup symbol. Duration is equal to (gdWakeupSymbolTxIdle - gdWakeupSymbolTxLow)/2 minus a safe part. (Collisions, clock differences, and other effects can deform the Tx-wakeup pattern.) [gdBit].		
Multiplicity	1		
Type	IntegerParamDef		
Range	14 .. 59		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GdWakeupSymbolRxLow		
Description	Number of bits used by the node to test the LOW portion of a received wakeup symbol. This lower limit of zero bits has to be received to detect the LOW portion by the receiver. The duration is equal to gdWakeupSymbolTxLow minus a safe part. (Active stars, clock differences, and other effects can deform the Tx-wakeup pattern.) [gdBits].		
Multiplicity	1		
Type	IntegerParamDef		
Range	10 .. 55		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GdWakeupSymbolRxWindow		
Description	The size of the window used to detect wakeups. Detection of a wakeup requires a low and idle period (from one WUS) and a low period (from another WUS) to be detected entirely within a window of this size. The duration is equal to gdWakeupSymbolTxIdle + 2 * gdWakeupSymbolTxLow plus a safe part. (Clock differences and other effects can deform the Tx-wakeup pattern.) [gdBit].		
Multiplicity	1		
Type	IntegerParamDef		
Range	76 .. 301		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Name last modified in: V1.0.0	GdWakeupSymbolTxIdle		
Description	Number of bits used by the node to transmit the 'idle' part of a wakeup symbol. The duration is equal to cdWakeupSymbolTxIdle [gdBit].		
Multiplicity	1		
Type	IntegerParamDef		
Range	45 .. 180		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	GdWakeupSymbolTxLow		
Description	Number of bits used by the node to transmit the LOW part of a wakeup symbol. The duration is equal to cdWakeupSymbolTxLow [gdBit].		
Multiplicity	1		
Type	IntegerParamDef		
Range	15 .. 60		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module/Fr		

Included Containers

Container Name	Multiplicity	Scope / Depedency
FrIfController	1..*	
FrIfJobList	0..1	

4.6.1.2 Communication Controllers

The FrIf presents all FlexRay communication controllers (FrController) under control by any FlexRay driver. Configuration defines the IDs used to the upper layer. It also defines the references to the controller configuration in the driver(s) to correlate FrIfControllers with FrControllers (see figure 4.71).

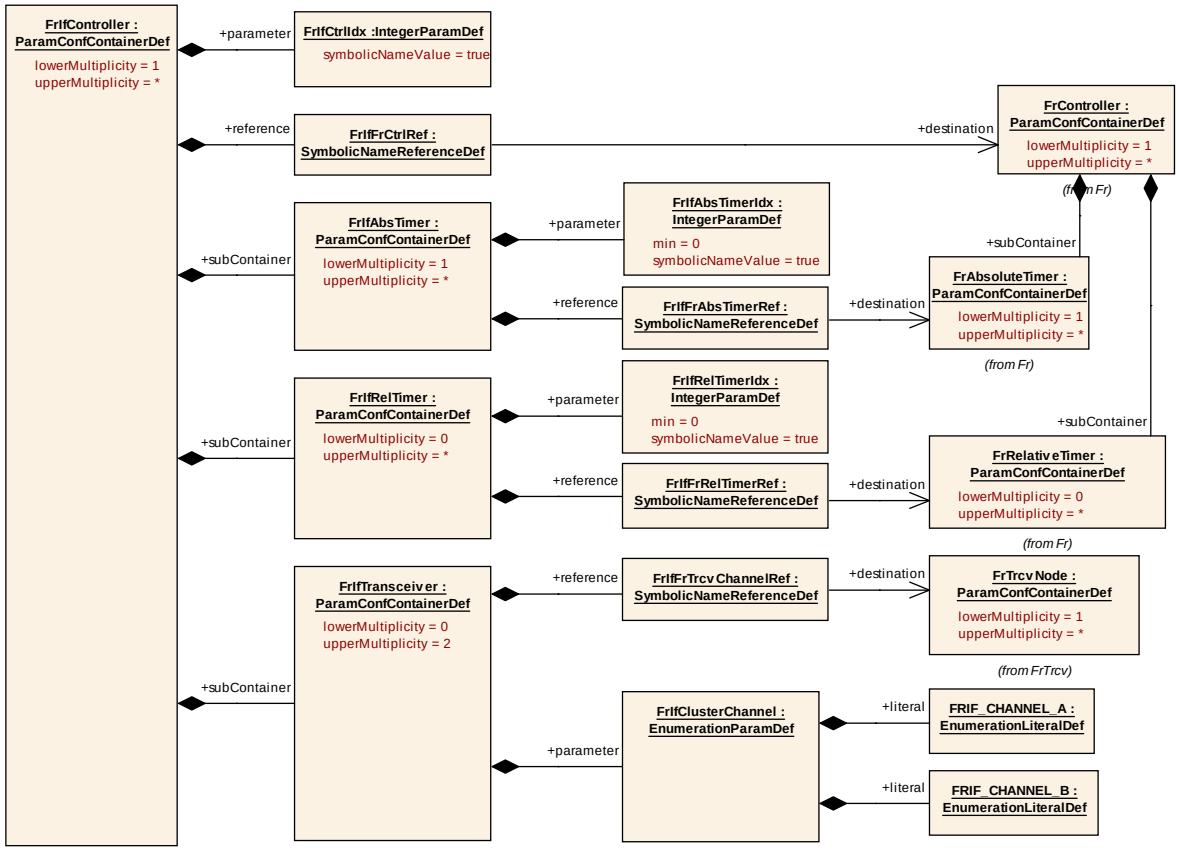


Figure 4.71: FlexRay Timer and Communication controller configuration

FrIfController

SWS Item	[FrIf05363]		
Container Name	FrIfController		
Description	Defines the communication controllers provided by FrIf.		
Configuration Parameters			

Name last modified in: V1.0.0	FrIfCtrlIdx		
Description	This parameter provides a zero-based consecutive index of the FlexRay Communication Controllers. Upper layer BSW modules and the FrIf itself use this index to identify a FlexRay CC.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	FrIfFrCtrlRef		
Description	Reference to a Controller, which is handled by a specific Driver. This reference is unique for the ECU.		
Multiplicity	1		
Type	Symbolic name reference to FrController		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Included Containers

Container Name	Multiplicity	Scope / Depedency
FrIfAbsTimer	1..*	
FrIfFrameTriggering	0..*	
FrIfRelTimer	0..*	
FrIfTransceiver	0..2	
FrIfVirtualBuffer	0..*	

A FlexRay controller needs to support Absolute Timers and may support Relative Timers. `FrIf` abstracts these timers to upper layers with own IDs. The configuration relates these `FrIfxxxTimer`s to the timers defined in the FlexRay drivers by referencing the corresponding configuration element in the driver.

FrIfAbsTimer

SWS Item	[frIf05361]		
Container Name	FrIfAbsTimer		
Description	This container contains the configuration of an absolute timer of a FlexRay CC.		
Configuration Parameters			

Name last modified in: V1.0.0	FrIfAbsTimerIdx		
Description	This parameter provides a zero-based consecutive index of the absolute timers. Upper layer BSW modules use this index to identify an absolute timer.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: ECU		

Name last modified in: V1.0.0	FrIfFrAbsTimerRef		
Description	Reference to an absolute timer, which is handled by a specific FlexRay Driver. This reference is unique for the ECU, therefore, an explicit reference to the FlexRay Driver is not necessary.		
Multiplicity	1		
Type	Symbolic name reference to FrAbsoluteTimer		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: ECU		
No Included Containers			

FrIfRelTimer

SWS Item	[FrIf05362]		
Container Name	FrIfRelTimer		
Description	This container contains the configuration of a relative timer of a FlexRay CC.		
Configuration Parameters			

Name last modified in: V1.0.0	FrIfFrRelTimerRef		
Description	Reference to a relative timer, which is handled by a specific FlexRay Driver. This reference is unique for the ECU, therefore, an explicit reference to the FlexRay Driver is not necessary.		
Multiplicity	1		
Type	Symbolic name reference to FrRelativeTimer		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: ECU		

Name last modified in: V1.0.0	FrIfRelTimerIdx		
Description	This parameter provides a zero-based consecutive index of the relative timers. Upper layer BSW modules use this index to identify a relative timer.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: ECU		
No Included Containers			

Additionally the `FrIfController` keeps track of the FlexRay transceivers that services that communication controller.

FrIfTransceiver

SWS Item	[FrIf05391]		
Container Name	FrIfTransceiver {FRIF_TRANSCEIVER}		
Description	Up to two FlexRay Transceivers may connect a Controller to a Cluster. This container realizes a Controller-Transceiver assignment.		
Configuration Parameters			

Name last modified in: V1.0.0	FrIfClusterChannel		
Description	This parameter identifies to which one of the two Channels ”A” or ”B” of the Cluster the Transceiver is connected.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	FRIF_CHANNEL_A	Channel A	
	FRIF_CHANNEL_B	Channel B	
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: ECU		

Name last modified in: V1.0.0	FrIfFrTrcvChannelRef		
Description	Reference to a Transceiver Driver Channel. This reference is unique for the ECU.		
Multiplicity	1		
Type	Symbolic name reference to FrTrcvNode		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: ECU		
No Included Containers			

Figure 4.72 depicts the configuration of frame triggerings and virtual buffers that the `FrIf` defines per communication controller.

`FrIf` communicates with the FlexRay drivers using so-called virtual buffers. Each virtual buffer references a frame triggering.

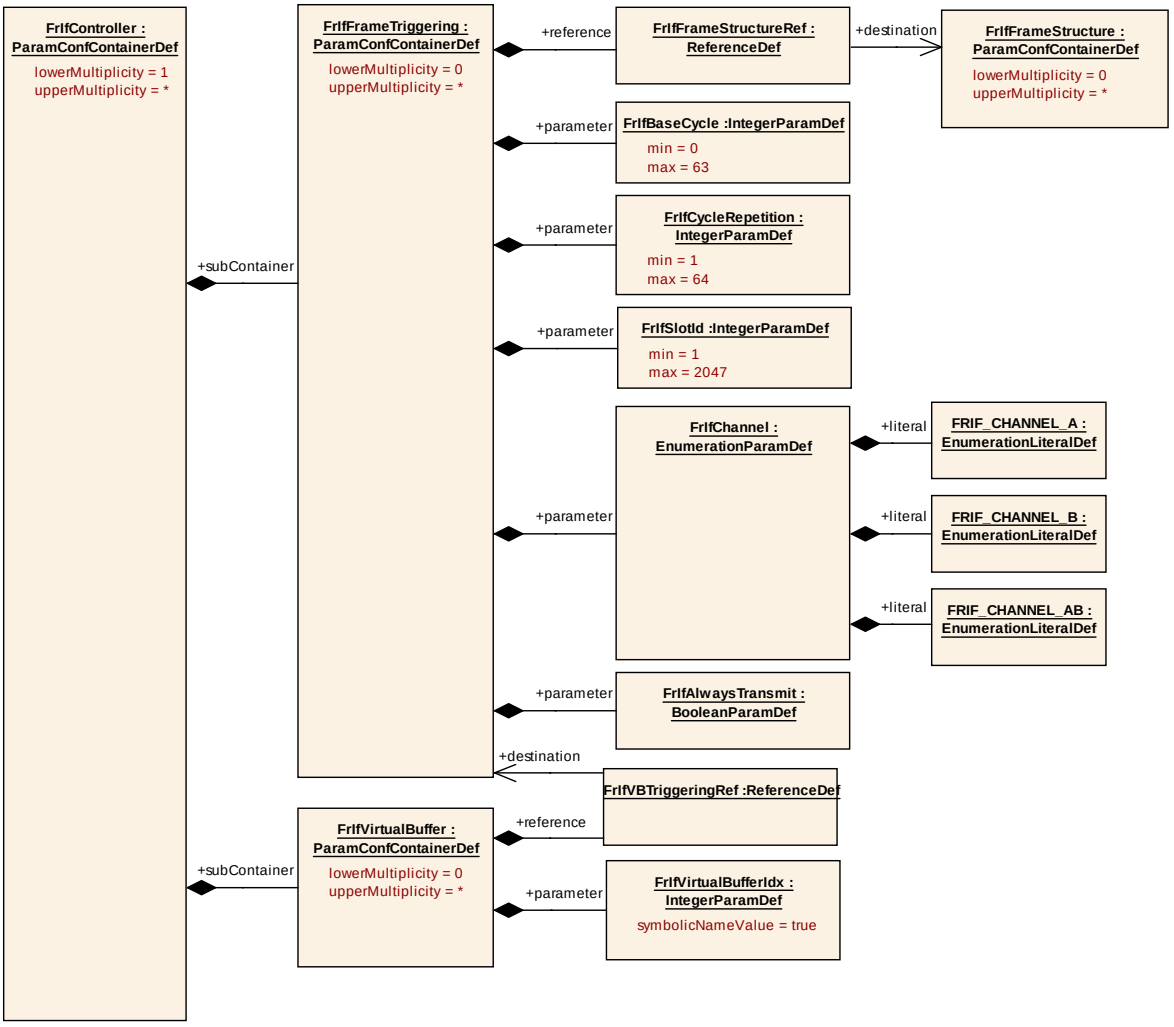


Figure 4.72: FlexRay Controller configuration (continued)

FrIfVirtualBuffer

SWS Item	[FrIf05364]
Container Name	FrIfVirtualBuffer
Description	FlexRay interface and FlexRay driver(s) communicate with each other using ”virtual buffer index” as a handle. Each (cycle, slot) pair that is used for transmission or reception on a cluster gets an own virtual buffer index assigned. That way, FlexRay Interface does not need to know about any hardware restrictions a certain controller might have. It is up to the FlexRay driver to map virtual buffers to real hardware buffers.
Configuration Parameters	

Name last modified in: V1.0.0	FrIfVBTriggeringRef		
Description	Reference to the assigned Frame triggering.		
Multiplicity	1		
Type	Reference to FrIfFrameTriggering		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	FrIfVirtualBufferIdx		
Description	This parameter identifies the virtual buffer.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		
No Included Containers			

A frame structure (defined in chapter 4.6.1.4) may be used in different frame triggerings. A frame triggering defines in which slot and cycle(s) a frame has to be sent or received.

FrIfFrameTriggering

SWS Item			
Container Name	FrIfFrameTriggering		
Description	A frame triggering defies that a frame shall be sent in a certain (slot, base cycle, cycle repetition, channel) pattern.		
Configuration Parameters			

Name last modified in: V1.0.0	FrIfAlwaysTransmit		
Description	The FlexRay Driver API service Fr_TransmitTxLSdu() will be called for this FlexRay Frame even if FrIf_Transmit() has not been called for any of the PDUs in the Frame.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	FrIfBaseCycle		
Description	This parameter contains the FlexRay Base Cycle used to transmit this FlexRay Frame.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 63		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	FrIfChannel		
Description	This parameter contains the FlexRay Channel used to transmit this FlexRay Frame.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	FRIF_CHANNEL_A	Channel A	
	FRIF_CHANNEL_AB	Channel A and B	
	FRIF_CHANNEL_B	Channel B	
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	FrIfCycleRepetition		
Description	This parameter contains the FlexRay Cycle Repetition used to transmit this FlexRay Frame..		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 64		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	FrIfFrameStructureRef		
Description	Reference to the Construction Plan of the FlexRay Frame.		
Multiplicity	1		
Type	Reference to FrIfFrameStructure		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	FrIfSlotId		
Description	This parameter contains the FlexRay Slot ID used to transmit this FlexRay Frame.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 2047		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		
No Included Containers			

4.6.1.3 Job list and Jobs

As shown in figure 4.68, FlexRay cluster configuration consists of job list configuration (details shown in figure 4.73).

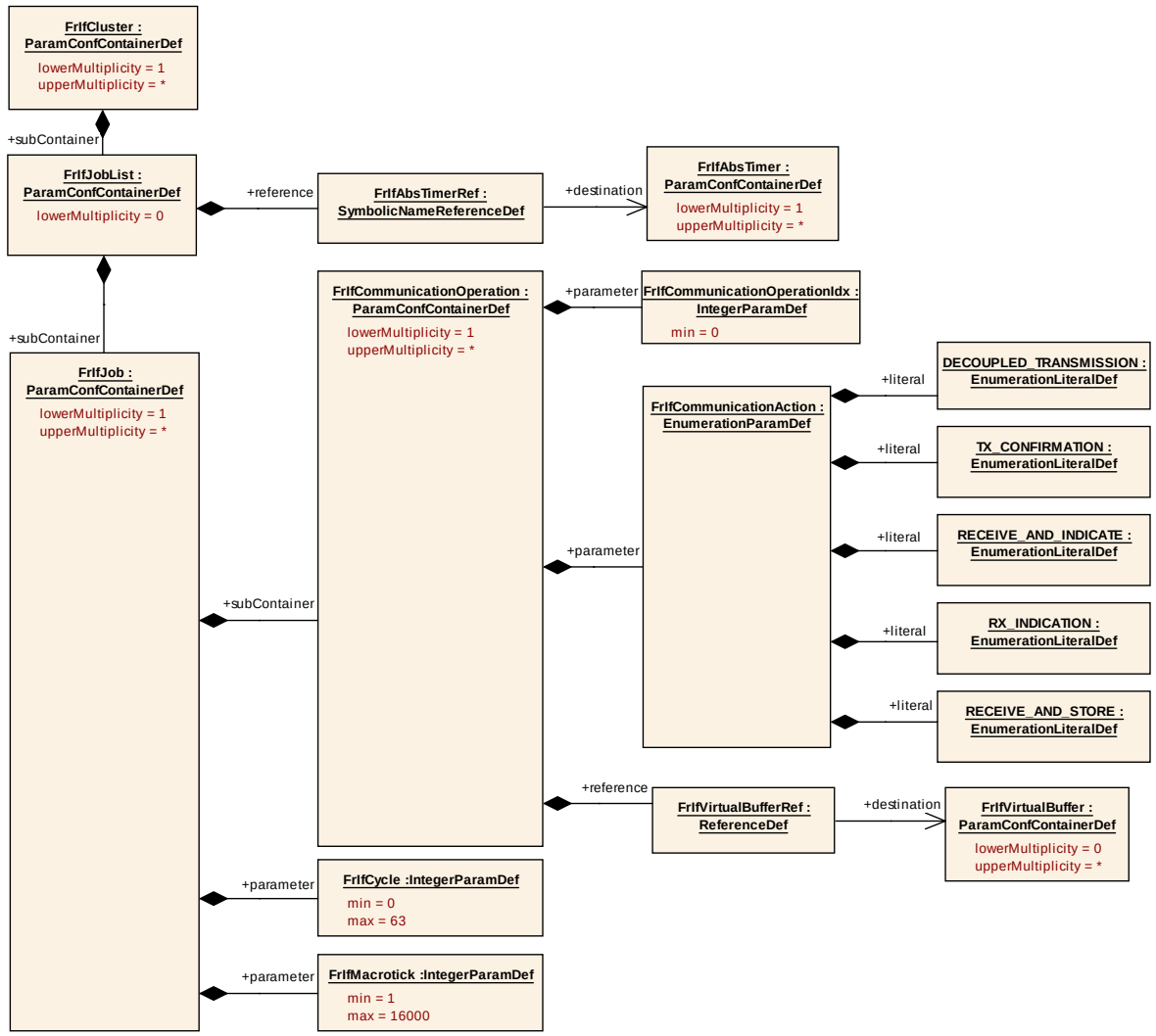


Figure 4.73: FlexRay Cluster configuration: Job lists

Transmit and receive operations in FlexRay need to be synchronized with the FlexRay cluster timing and schedule. This is done using jobs in a job list for each cluster. E.g. a transmit operation (i.e. sending the data to the FlexRay driver for storage in a hardware buffer) needs to be called in `FrIf` slightly before the frame needs to be transmitted by HW on the cluster. Thus, the corresponding job timing needs to be slightly before the data is really send out over the cluster.

FrIfJobList

SWS Item	[FrIf05367]		
Container Name	FrIfJobList		
Description	This container specifies a list of all FlexRay Jobs of the Cluster to be performed by FrIf_JobListExec_<ClstIdx>().		
Configuration Parameters			
Name last modified in: V1.0.0	FrIfAbsTimerRef		
Description	Reference to the absolute timer to be used to trigger the interrupt whose ISR contains the FrIf_JobListExec_<ClstIdx>() function.		
Multiplicity	1		
Type	Symbolic name reference to FrIfAbsTimer		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
FrIfJob	1..*		

A job list contains individual jobs, each executed at a different time in the schedule.

FrIfJob

SWS Item	[FrIf05368]		
Container Name	FrIfJob		
Description	A job may contain more than one operation that are executed at a specific point in time.		
Configuration Parameters			
Name last modified in: V1.0.0	FrIfCycle		
Description	The FlexRay Cycle in which to execute this job		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 63		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	FrIfMacrotick		
Description	Macrotick offset in the Cycle [Macrotick]		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 16000		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
FrIfCommunication Operation	1..*		

A single job may contain several communication operations which are executed in one batch. Each operation references a virtual buffer it relates to.

FrIfCommunicationOperation

SWS Item	[FrIf05369]		
Container Name	FrIfCommunicationOperation		
Description	A job may contain several communication operations which are executed sequentially		
Configuration Parameters			

Name last modified in: V1.0.0	FrIfCommunicationAction		
Description	The action to be performed in the FlexRay Operation		
Multiplicity	1		
Type	EnumerationParamDef		
Range	DECOUPLED_TRANSMISSION	Decoupled transmission	
	RECEIVE_AND_INDICATE	Immediate reception	
	RECEIVE_AND_STORE	Decoupled reception	
	RX_INDICATION	Reception indication	
	TX_CONFIRMATION	Transmission confirmation	
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	FrIfCommunicationOperationIdx		
Description	For each FlexRay Communication Job, this index spans a range of zero-based consecutive values and thus defines the order of the FlexRay Communication Operation in the respective FlexRay Communication Job.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	FrIfVirtualBufferRef		
Description	Reference to a virtual buffer.		
Multiplicity	1		
Type	Reference to FrIfVirtualBuffer		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		
No Included Containers			

4.6.1.4 Frame structure and PDUs

The `FrIfFrameStructure` defines how individual PDUs are packed into frames (see figure 4.74):

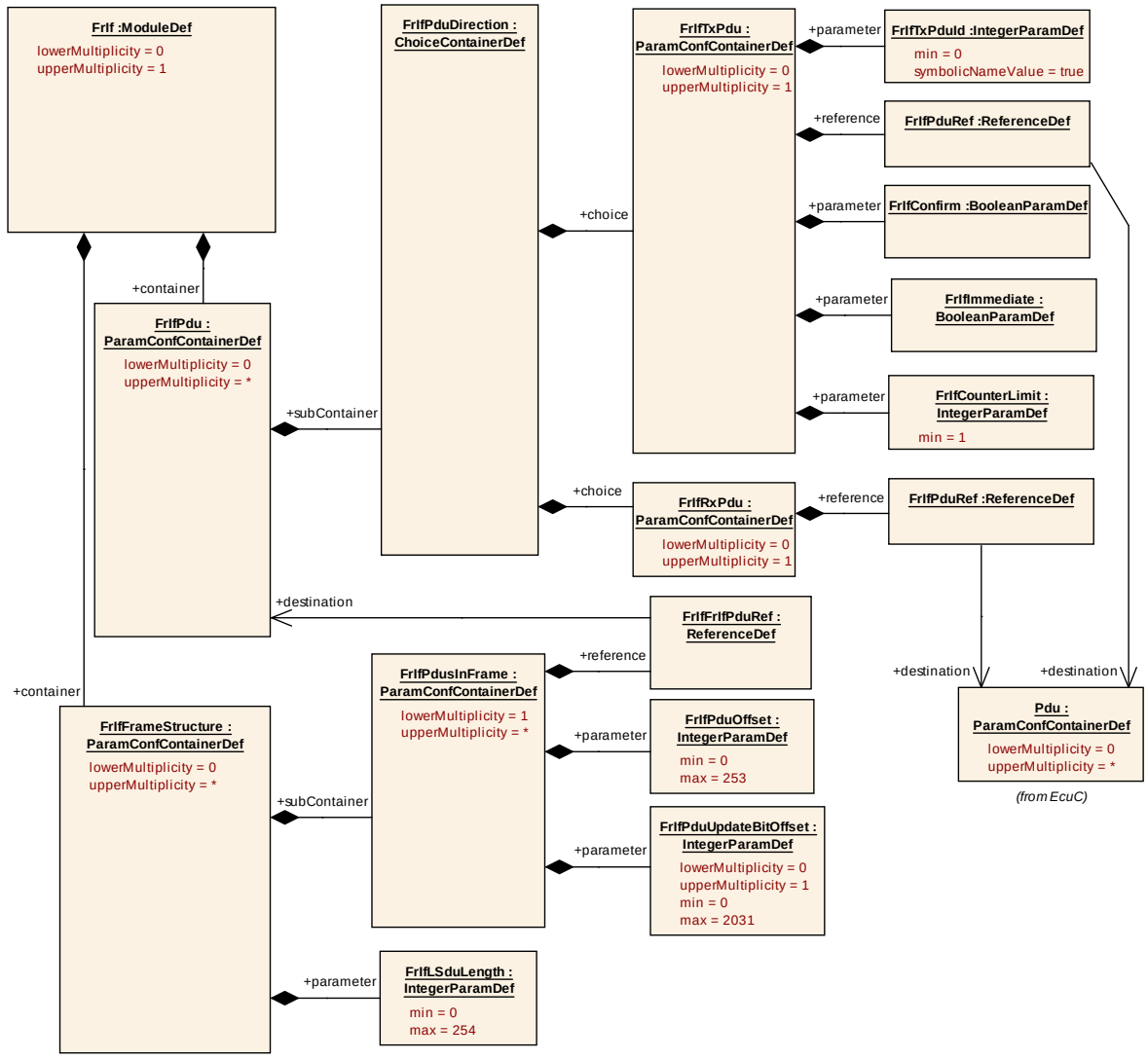


Figure 4.74: FlexRay Interface frame structure and PDU configuration

FrIfFrameStructure

SWS Item	[FrIf05370]		
Container Name	FrIfFrameStructure		
Description	The Frame structure specifies a Construction Plan how a Frame is assembled with PDUs and their respective Update-Bits.		
Configuration Parameters			
Name last modified in: V1.0.0	FrIfLSduLength		
Description	The payload length of the Frame is given here. This parameter is required for validation if configured PDUs and update information fits into the Frame at configuration time [bytes].		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 254		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module dependency: The parameter depends on the low level parameters of the FlexRay CC		
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
FrIfPdusInFrame	1..*		

The individual PDUs within the frame structure are defined by offset.

FrIfPdusInFrame

SWS Item	[FrIf05371]		
Container Name	FrIfPdusInFrame		
Description	This container holds all the information about a PDU in a FlexRay Frame.		
Configuration Parameters			
Name last modified in: V1.0.0	FrIfFrIfPduRef		
Description	This is the reference to the local definition of a PDU.		
Multiplicity	1		
Type	Reference to FrIfPdu		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	FrIfPduOffset		
Description	The value specifies the offset of the PDU within the Frame.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 253		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module dependency: This parameter depends on the number of PDUs contained in the Frame, PDU length, and Update-Bits of other PDUs in the Frame. In addition, if the Frame will is sent in static segment, this parameter depends on GPayloadLengthStatic.		

Name last modified in: V1.0.0	FrIfPduUpdateBitOffset		
Description	This value specifies where the PDU's Update-Bit is stored in the Frame. If the value is marked as invalid, no Update-Bit is used for the PDU.		
Multiplicity	0..1		
Type	IntegerParamDef		
Range	0 .. 2031		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module dependency: This parameter depends on the number of PDUs contained in the Frame, PDU length, and Update-Bits of other PDUs in the Frame. In addition, if the Frame will is sent in static segment, this parameter depends on GPayloadLengthStatic.		
No Included Containers			

The `FrIfPdu` container collects all `FrIf` specific information about a PDU and references to the global PDU element that is shared between COM stack modules (see figure 4.74). A PDU may either be transmitted or received.

FrIfPdu

SWS Item	[FrIf05372]	
Container Name	FrIfPdu	
Description	Contains PDU information. A PDU may be either a transmission PDU or a reception PDU. So, this container holds exactly one sub container.	
Configuration Parameters **Included Containers**		
Container Name	**Multiplicity**	**Scope / Depedency**
FrIfPduDirection [FrIfRxPdu, FrIfTxPdu]	1	

For transmit PDUs, the handle ID needs to be defined. This handle ID will be used by upper layers when sending the PDU to `FrIf`.

FrIfTxPdu

SWS Item	[FrIf05374]		
Container Name	FrIfTxPdu		
Description	This container specifies transmission PDUs.		
Configuration Parameters			

Name last modified in: V1.0.0	FrIfConfirm		
Description	Defines whether the transmission of a PDU should be checked and confirmed to the PDU owning BSW module.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	FrIfCounterLimit		
Description	This value states the maximum number of indication of ready PDU data to the FrIf (i.e. maximum number of invocations of FrIf_Transmit) without an intermediate transmission of the PDU.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	FrIfImmediate		
Description	Defines whether the the PDU is transmitted immediate or decoupled..		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	FrIfPduRef		
Description	Reference to the external PDU definition.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: ECU		

Name last modified in: V1.0.0	FrIfTxPduId		
Description	The global PDU identifier, which has to be used by the upper layer BSW module. The identifier has to be zero based and consecutive.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: ECU		
No Included Containers			

For receive PDUs, the handle ID used toward the upper layer is defined by that upper layer module. It has to be retrieved from the upper layer's configuration using the mechanisms described in chapter 4.1.1.

FrIfRxPdu

SWS Item	[FrIf05373]		
Container Name	FrIfRxPdu		
Description	Receive PDU		
Configuration Parameters			

Name last modified in: V1.0.0	FrIfPduRef		
Description	Reference to the external PDU definition.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: ECU		
No Included Containers			

4.6.2 FlexRay Driver

The FlexRay driver (`Fr`) configuration is specified in [18].

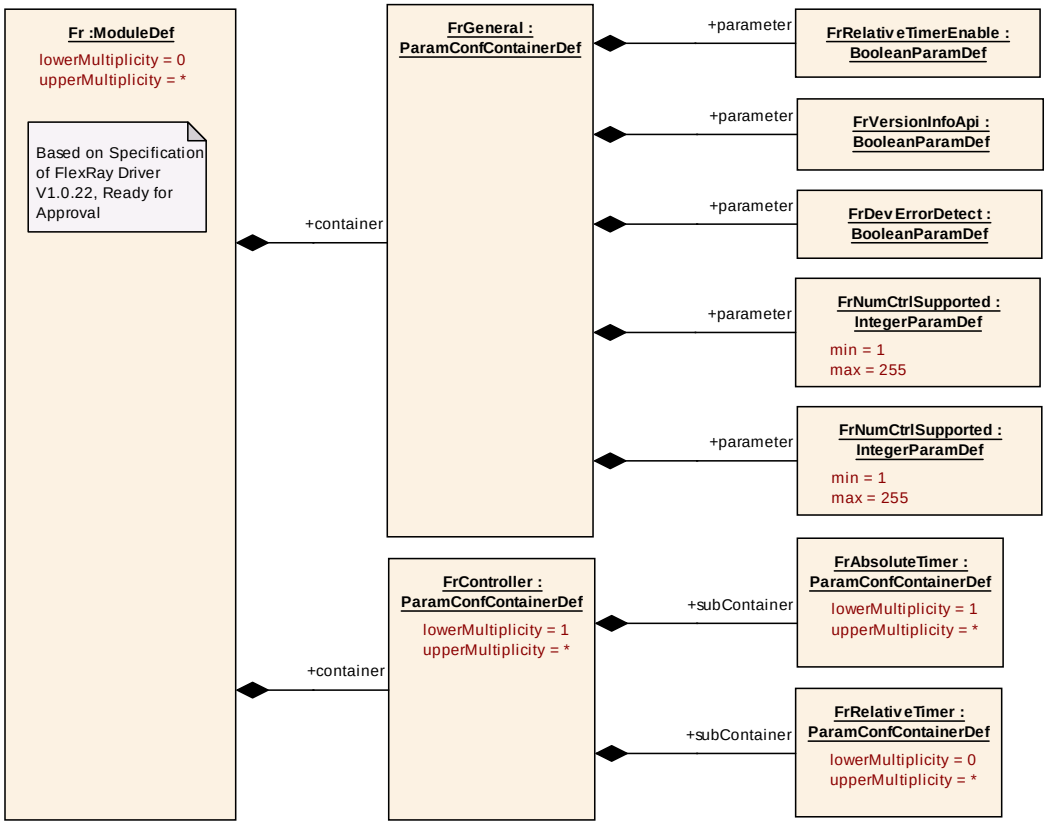


Figure 4.75: General FlexRay Driver configuration

The module defines two containers: `FrGeneral` and `FrController`.

Module Name	Fr	
Module Description	Configuration of the Fr (FlexRay driver) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
FrController	1..*	Configuration of the individual controllers.
FrGeneral	1	General configuration (parameters) of the FlexRay Driver module.

The parameters relevant for the complete FlexRay driver are stored in container `FrGeneral` as shown in figure 4.75.

FrGeneral

SWS Item	[FR082]		
Container Name	FrGeneral		
Description	General configuration (parameters) of the FlexRay Driver module.		
Configuration Parameters			
Name last modified in: V1.0.0	FrDevErrorDetect		
Description	Switches the Development Error Detection and Notification ON or OFF.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
Name last modified in: V1.0.0	FrNumCtrlSupported		
Description	Determines the maximum number of communication controllers that the driver supports.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 255		
Configuration Class	**Pre-compile time**	–	
	Link time	X	All Variants
	Post-build time	–	
Scope / Dependency	scope: Module		
Name last modified in: V1.0.0	FrRelativeTimerEnable		
Description	Enables or disables the usage of relative timers		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
Name last modified in: V1.0.0	FrVersionInfoApi [ADC105]		
Description	Enables/disables the existence of the Fr_GetVersionInfo API		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
No Included Containers			

4.6.2.1 FlexRay Controller Configuration

The second top level container `FrController` holds all controller specific configuration.

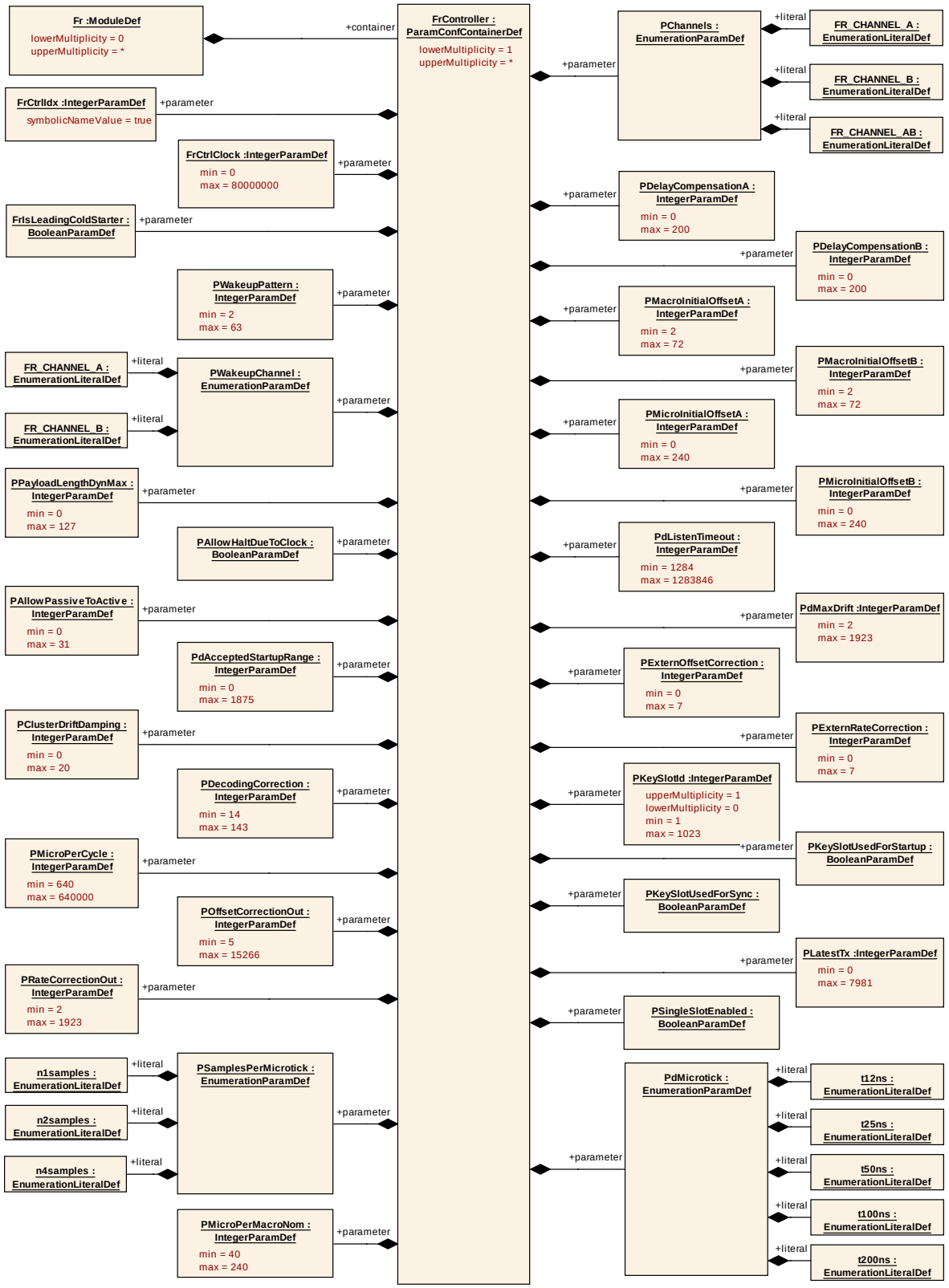


Figure 4.76: FlexRay Driver: Controller parameters

During development of the configuration of the FlexRay driver, all cluster specific configuration is defined centrally in the FlexRay Interface configuration. Please note that a code generator for FlexRay driver needs to get those parameters and needs to integrate them in the generated code.

All controller specific parameters are shown in figure 4.76.

FrController

SWS Item	[FR083]		
Container Name	FrController		
Description	Configuration of the individual controllers.		
Configuration Parameters			

Name last modified in: V1.0.0	FrCtrlClock		
Description	Determines clock connected to the CC [Hz].		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 80000000		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	FrCtrlIdx		
Description	Determines index of CC within Fr.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	FrIsLeadingColdStarter		
Description	Enables/Disables the leading transmission of SUP frames.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PAllowHaltDueToClock		
Description	Boolean flag that controls the transition to the POC:halt state due to a clock synchronization errors. If set to true, the CC is allowed to transition to POC:halt. If set to false, the CC will not transition to the POC:halt state but will enter or remain in the POC:normal passive state (self healing would still be possible)		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PAllowPassiveToActive		
Description	Number of consecutive even/odd cycle pairs that must have valid clock correction terms before the CC will be allowed to transition from the POC:normal passive state to POC:normal active state. If set to zero, the CC is not allowed to transition from POC:normal passive to POC:normal active		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 31		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PChannels		
Description	Channels to which the node is connected		
Multiplicity	1		
Type	EnumerationParamDef		
Range	FR_CHANNEL_A	Cluster uses channel A	
	FR_CHANNEL_AB	Cluster uses channel A and B	
	FR_CHANNEL_B	Cluster uses channel B	
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PClusterDriftDamping		
Description	Local cluster drift damping factor used for rate correction [Microticks]		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 20		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PDecodingCorrection		
Description	Value used by the receiver to calculate the difference between primary time reference point and secondary time reference point [Microticks]		
Multiplicity	1		
Type	IntegerParamDef		
Range	14 .. 143		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PDelayCompensationA		
Description	Value used to compensate for reception delays on the indicated channel. This covers assumed propagation delay up to cPropagationDelayMax for microticks in the range of 0.0125 μs to 0.05 μs. In practice, the minimum of the propagation delays of all sync nodes should be applied [Microticks].		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 200		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PDelayCompensationB		
Description	Value used to compensate for reception delays on the indicated channel. This covers assumed propagation delay up to cPropagationDelayMax for microticks in the range of 0.0125 μs to 0.05 μs. In practice, the minimum of the propagation delays of all sync nodes should be applied [Microticks].		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 200		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PExternOffsetCorrection		
Description	Number of microticks added or subtracted to the NIT to carry out a host-requested external offset correction [Microticks].		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 7		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PExternRateCorrection		
Description	Number of microticks added or subtracted to the cycle to carry out a host-requested external rate correction [Microticks].		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 7		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PKeySlotId		
Description	ID of the slot used to transmit the startup frame, sync frame, or designated single slot frame		
Multiplicity	0..1		
Type	IntegerParamDef		
Range	1 .. 1023		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PKeySlotUsedForStartup		
Description	Flag indicating whether the Key Slot is used to transmit a startup frame		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PKeySlotUsedForSync		
Description	Flag indicating whether the Key Slot is used to transmit a sync frame		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PLatestTx		
Description	Number of the last minislot in which a frame transmission can start in the dynamic segment [Minislots].		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 7981		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PMacroInitialOffsetA		
Description	Integer number of macroticks between the static slot boundary and the following macrotick boundary of the secondary time reference point based on the nominal macrotick duration [Macroticks].		
Multiplicity	1		
Type	IntegerParamDef		
Range	2 .. 72		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PMacroInitialOffsetB		
Description	Integer number of macroticks between the static slot boundary and the following macrotick boundary of the secondary time reference point based on the nominal macrotick duration [Macroticks].		
Multiplicity	1		
Type	IntegerParamDef		
Range	2 .. 72		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PMicroInitialOffsetA		
Description	Number of microticks between the closest macrotick boundary described by pMacroInitialOffset[Ch] and the secondary time reference point. The parameter depends on pDelayCompensation[Ch] and therefore it has to be set independently for each channel [Microticks].		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 240		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PMicroInitialOffsetB		
Description	Number of microticks between the closest macrotick boundary described by pMacroInitialOffset[Ch] and the secondary time reference point. The parameter depends on pDelayCompensation[Ch] and therefore it has to be set independently for each channel [Microticks].		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 240		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PMicroPerCycle		
Description	Nominal number of microticks in the communication cycle of the local node. If nodes have different microtick durations this number will differ from node to node [Microticks].		
Multiplicity	1		
Type	IntegerParamDef		
Range	640 .. 640000		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PMicroPerMacroNom		
Description	Number of microticks per nominal macrotick that all implementations must support [Microticks].		
Multiplicity	1		
Type	IntegerParamDef		
Range	40 .. 240		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	POffsetCorrectionOut		
Description	Magnitude of the maximum permissible offset correction value [Microticks].		
Multiplicity	1		
Type	IntegerParamDef		
Range	5 .. 15266		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PPayloadLengthDynMax		
Description	Maximum payload length for dynamic frames [16 bit words].		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 127		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PRateCorrectionOut		
Description	Magnitude of the maximum permissible rate correction value [Microticks].		
Multiplicity	1		
Type	IntegerParamDef		
Range	2 .. 1923		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PSamplesPerMicrotick		
Description	Number of samples per microtick		
Multiplicity	1		
Type	EnumerationParamDef		
Range	n1samples	1 sample	
	n2samples	2 samples	
	n4samples	4 samples	
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PSingleSlotEnabled		
Description	Flag indicating whether or not the node shall enter single slot mode following startup		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PWakeupChannel		
Description	Channel used by the node to send a wakeup pattern		
Multiplicity	1		
Type	EnumerationParamDef		
Range	FR_CHANNEL_A		
	FR_CHANNEL_B		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PWakeupPattern		
Description	Number of repetitions of the wakeup symbol that are combined to form a wakeup pattern when the node enters the POC:wakeup send state		
Multiplicity	1		
Type	IntegerParamDef		
Range	2 .. 63		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PdAcceptedStartupRange		
Description	Expanded range of measured clock deviation allowed for startup frames during integration [Microticks].		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 1875		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PdListenTimeout		
Description	Upper limit for the start up listen timeout and wake up listen timeout [Microticks].		
Multiplicity	1		
Type	IntegerParamDef		
Range	1284 .. 1283846		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PdMaxDrift		
Description	Maximum drift offset between two nodes that operate with unsynchronized clocks over one communication cycle [Microticks].		
Multiplicity	1		
Type	IntegerParamDef		
Range	2 .. 1923		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PdMicrotick		
Description	Duration of a microtick.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	t100ns	100 ns	
	t12ns	12 ns	
	t200ns	200 ns	
	t25ns	25 ns	
	t50ns	50 ns	
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: Module		

Included Containers

Container Name	Multiplicity	Scope / Depedency
FrAbsoluteTimer	1..*	
FrRelativeTimer	0..*	

4.6.2.2 FlexRay Driver Timer Configuration

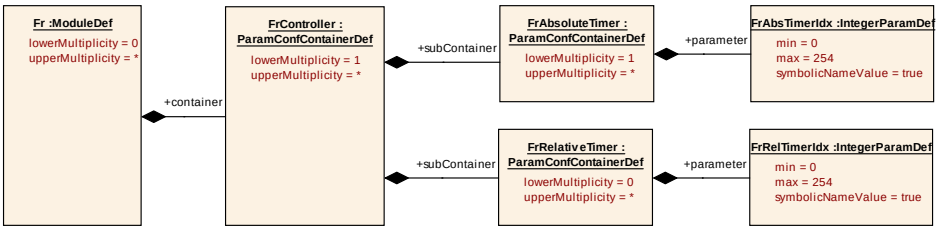


Figure 4.77: FlexRay Controller: Timer configuration

`FrController` defines two subcontainers for configuration of the timers, as shown in figure 4.77.

FrAbsoluteTimer

SWS Item	[FR082]		
Container Name	FrAbsoluteTimer		
Description	Specifies the absolute timer configuration parameters of the Fr.		
Configuration Parameters			
Name last modified in: V1.0.0	FrAbsTimerIdx		
Description	Contains the index of an absolute timer contained in Fr on a certain FlexRay CC.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 254		
Configuration Class	**Pre-compile time**	–	
	Link time	X	All Variants
	Post-build time	–	
Scope / Dependency	scope: Module		
No Included Containers			

FrRelativeTimer

SWS Item	[FR082]
Container Name	FrRelativeTimer
Description	Specifies the relative timer configuration parameters of the Fr.
Configuration Parameters	

Name last modified in: V1.0.0	FrRelTimerIdx		
Description	Contains the index of a relative timer contained in Fr on a certain FlexRay CC.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 254		
Configuration Class	**Pre-compile time**	–	
	Link time	X	All Variants
	Post-build time	–	
Scope / Dependency	scope: Module		
No Included Containers			

4.6.3 FlexRay Transport Layer

This section contains all the configuration parameters of the FlexRay Transport Layer as specified in [29]. Figure 4.78 depicts the top-level structure of that module.

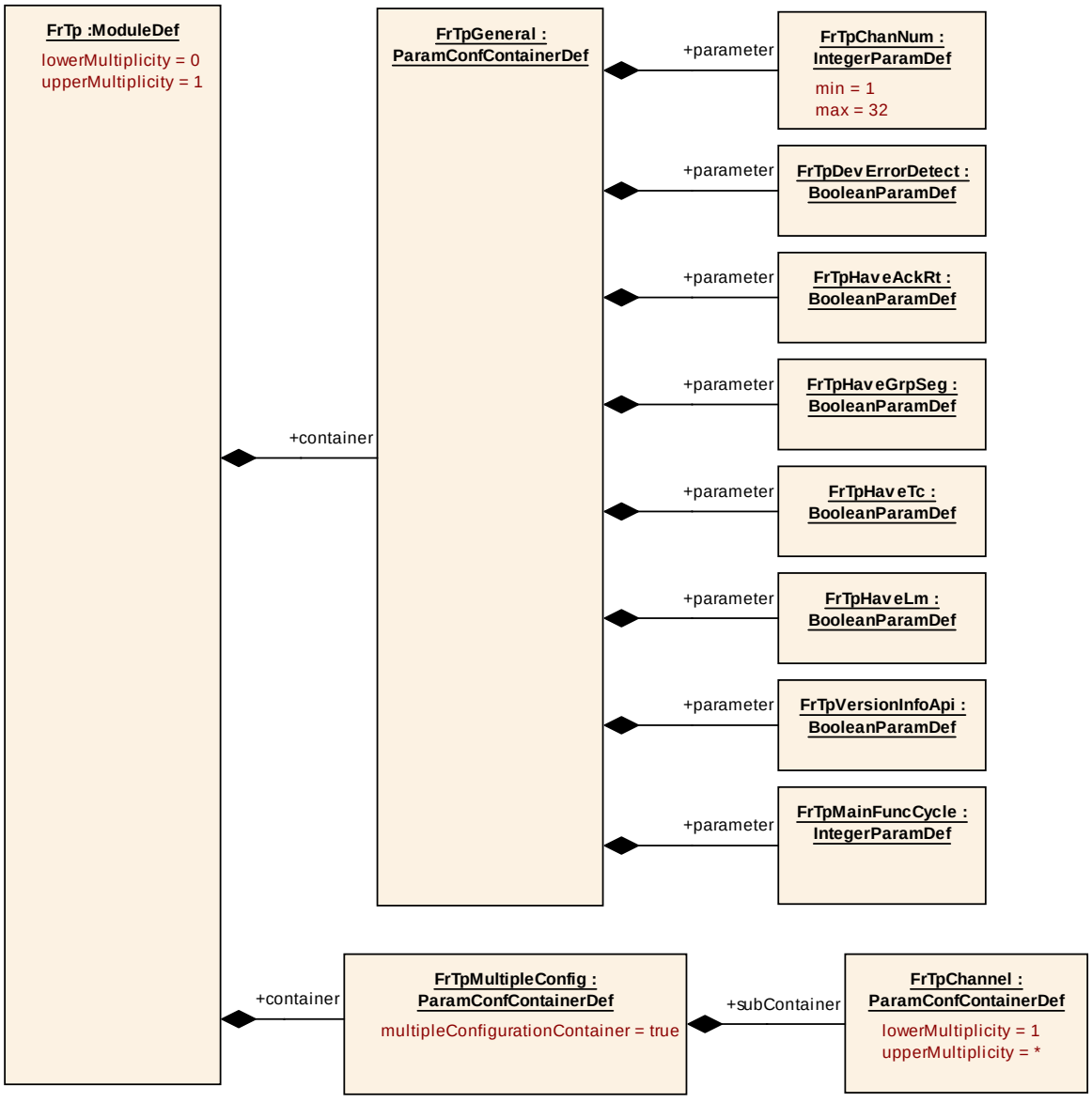


Figure 4.78: FlexRay Transport Layer Overview

Module Name	FrTp	
Module Description	Configuration of the FrTp (FlexRay Transport Protocol) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
FrTpGeneral	1	This container contains the general configuration (parameters) of the FlexRay TP.
FrTpMultipleConfig	1	This container holds one or several multiple configuration sets.

4.6.3.1 FlexRay TP General Configuration

FrTpGeneral

SWS Item			
Container Name	FrTpGeneral		
Description	This container contains the general configuration (parameters) of the FlexRay TP.		
Configuration Parameters			

Name last modified in: V1.1.0	FrTpChanNum {FRTP_CHAN_NUM}		
Description	Preprocessor switch for defining the number of concurrent channels the module supports. Up to 32 channels shall be definable here.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 32		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpDevErrorDetect {FRTP_DEV_ERROR_DETECT}		
Description	Preprocessor switch for enabling development error detection.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpHaveAckRt {FRTP_HAVE_ACKRT}		
Description	Preprocessor switch for enabling the Acknowledgement and retry mechanisms.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpHaveGrpSeg {FRTP_HAVE_GRPSEG}		
Description	Preprocessor switch for enabling segmentation of 1:n messages.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpHaveLm {FRTP_HAVE_LM}		
Description	Preprocessor switch for enabling the mechanism for message longer than allowed by.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpHaveTc {FRTP_HAVE_TC}		
Description	Preprocessor switch for enabling Transmit Cancellation.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpMainFuncCycle {FRTP_MAINFUNC_CYCLE}		
Description	This parameter contains the calling period of the TPs Main Function.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	L	Pre Compile and Post Build
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpVersionInfoApi {FRTP_VERSION_INFO_API}		
Description	Preprocessor switch for enabling the Version info API.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		
No Included Containers			

4.6.3.2 FlexRay TP Channel Configuration

The FrTp module manages several channels. Figure 4.79 depicts all configuration parameters that constitute a specific channel configuration.

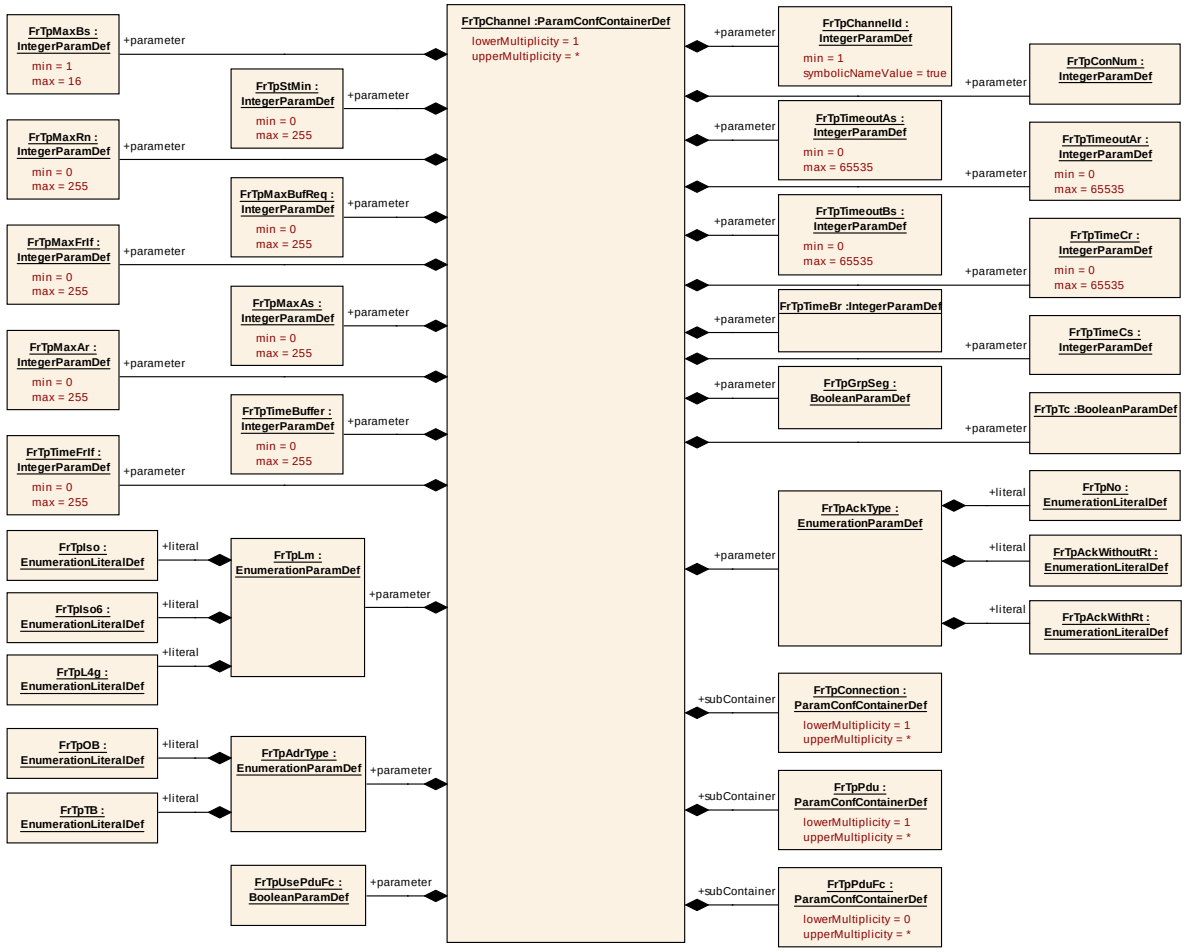


Figure 4.79: FlexRay Transport Layer Channel Configuration

FrTpChannel

SWS Item			
Container Name	FrTpChannel {FrTpChannelConfiguration}		
Description	This container contains the configuration (parameters) of one FlexRay TP channel.		
Configuration Parameters			
Name last modified in: V1.1.0	FrTpAckType {FRTP_ACKTYPE}		
Description	This parameter defines the type of acknowledgement which is used for the specific channel.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	FrTpAckWithRt	no acknowledgement	
	FrTpAckWithoutRt	no acknowledgement	
	FrTpNo	no acknowledgement	
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency	scope: module		
Name last modified in: V1.1.0	FrTpAdrType {FRTP_LM}		
Description	This parameter states the addressing type this connection has. The meanings of the values are one byte and two byte.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	FrTpOB	One Byte	
	FrTpTB	Two Bytes	
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency	scope: module		
Name last modified in: V1.1.0	FrTpChannelId {FRTP_CHANNEL_ID}		
Description	The Id of the channel.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: module		
Name last modified in: V1.1.0	FrTpConNum {FRTP_CON_NUM}		
Description	This parameter states the number of connections used in this channel. At least 256 shall be configurable here.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpGrpSeg [FRTP_GRPSEG]		
Description	Here can be specified, whether segmentation within a 1:n connection is allowed or not.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpLm {FRTP_LM}		
Description	This specifies the maximum message length for the particular channel.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	FrTpIso	Up to (2**12)-1 Byte message length (No FF-Ex or SF-E or AF shall be used and recognized)	
	FrTpIso6	As ISO, but the maximum payload length is limited to 6 byte (SF-I, FF-I, CF). This is necessary to route TP on CAN when using Extended Addressing or Mixed Addressing on CAN.	
	FrTpL16m	SF-E allowed (SF of arbitrary length depending on FrTpPduLength), up to (2**24)-1 byte message length (no FF-E4 shall be used and recognized)	
	FrTpL4g	SF-E allowed (SF of arbitrary length depending on FrTpPduLength), up to (2**32)-1 byte message length (all FF-x allowed).	
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpMaxAr {FRTP_MAX_AR}		
Description	This parameter defines the maximum number of trying to send a frame when a TIMEOUT AR occurs (depending on whether retry is configured).		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpMaxAs {FRTP_MAX_AS}		
Description	This parameter defines the maximum number of trying t osend a frame when a TIMEOUT AS occurs (depending on whether retry is configured)		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpMaxBs {FRTP_MAXBS}		
Description	This parameter is only relevant when having retry activated. It limits the maximal block size the FrTp can choose in order to limit the amount of Tx buffer that will be requested at the sender side in a segmented transfer.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 16		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpMaxBufReq {FRTP_MAX_BUFREQ}		
Description	This parameter defines the maximum number of trying to get a buffer (Transmit / Receive), depending of the return value of PduR_FrTpProvideTxBuffer / PduR_FrTpProvideRxBuffer and on whether retry is configured.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpMaxFrIf {FRTP_MAX_FRIF}		
Description	This parameter defines the maximum number of trying to send a frame when the FrIf returns an error.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpMaxRn {FRTP_MAX_RN}		
Description	This parameter defines the maximum number of retries (if retry is configured for the particular channel).		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpStMin {FRTP_STMIN}		
Description	This parameter defines the minimum amount of time between two succeeding CFs.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpTc [FRTP_TC]		
Description	With this switch Transmit Cancellation can be turned on or off for this channel.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpTimeBr {FRTP_TIME_BR}		
Description	This parameter defines the time in ms between receiving the last CF of a block or an FF-x (or SF-x) and sending out an FC or AF. It is obvious that FRTP_TIME_BR + FRTP_TIMEOUT_AR $<$ FRTP_TIMEOUT_BS must hold (because the transmission duration on the bus has also to be considered). This parameter is defined in ISO 15765-2. It is contained in the configuration as a performance requirement.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	FrTpTimeBuffer {FRTP_TIME_BUFFER}		
Description	This parameter defines the time in s of waiting for the next try (if retry is activated) to get a Tx or Rx buffer.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpTimeCr {FRTP_TIME_CR}		
Description	This parameter defines the timeout value in s for waiting for a CF or FF-x (in case of retry) after receiving the last CF or after sending an FC or AF on the receiver side.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpTimeCs {FRTP_TIME_CS}		
Description	This parameter defines the time between the sending of two consecutive CFs or between a CF and a FC (for Transmit Cancellation) or between reception of an FC or AF and sending of the next CF or a FC (for Transmit Cancellation). It is obvious that FRTP_TIME_CS + FRTP_TIMEOUT_AS < FRTP_TIMEOUT_CR must hold (because the transmission duration on the bus has also to be considered). This parameter is defined in ISO 15765-2. It is contained in the configuration as a performance requirement.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	FrTpTimeFrIf {FRTP_TIME_FRIF}		
Description	This parameter defines the time in s of waiting for the next try (if retry is activated) to send via FrIf_Transmit.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpTimeoutAr {FRTP_TIMEOUT_AR}		
Description	This parameter states the timeout in s between the PDU transmit request of the Transport Layer to the FlexRay Interface and the corresponding confirmation of the FlexRay Interface on the receiver side (for FC or AF).		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpTimeoutAs {FRTP_TIMEOUT_AS}		
Description	This parameter states the timeout in s between the PDU transmit request for the first PDU of the group used in the current connection of the Transport Layer to the FlexRay Interface and the corresponding confirmation of the FlexRay Interface (when having sent the last PDU of the group used in this connection) on the sender side (SF-x, FF-x, CF or FC (in case of Transmit Cancellation)).		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpTimeoutBs {FRTP_TIMEOUT_BS}		
Description	This parameter defines the timeout for waiting for an FC or AF on the sender side in a 1:1 connection. in [s]		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpUsePduFc {FRTP_USE_PDU_FC}		
Description	This switch defines, whether within this channel the dedicated FC/ACK PDU (FrTpPduFc) shall be used or not. If this is not used FC / ACK frames are sent using the normal IDs, otherwise only FrTpPduFc shall be used for sending / receiving FC / ACK frames.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency	scope: module		

Included Containers

Container Name	Multiplicity	Scope / Depedency
FrTpConnection	1..*	
FrTpPdu	1..*	
FrTpPduFc	0..*	

4.6.3.3 FlexRay TP PDU Configuration

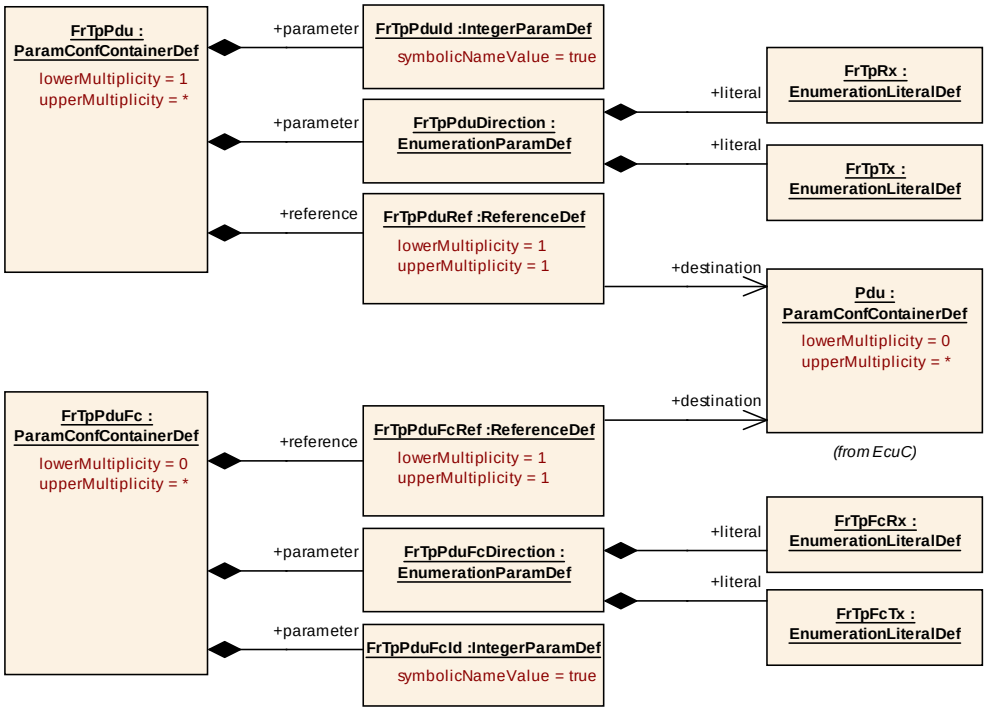


Figure 4.80: FlexRay Transport Layer PDU Configuration

FrTpPdu

SWS Item			
Container Name	FrTpPdu		
Description			
Configuration Parameters			

Name last modified in: V1.1.0	FrTpPduDirection		
Description	This parameter defines the direction of the PDU.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	FrTpRx	no acknowledgement	
	FrTpTx	no acknowledgement	
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency			

Name last modified in: V1.1.0	FrTpPduId {FRTP_PDU}		
Description	This is the identifier of the FlexRay Interface PDUs (Fr N-PDU, Fr L-SDU) in which the Transport Layer Flow Control and Acknowledgement Frames of this channel should be transmitted.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpPduRef		
Description			
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency			
No Included Containers			

Instead of having an array of `FrTp_PduInfoType` as specified in [29] the `FrTpPdu` defines a reference to the global PDU definition. Hence the handle ID of the according `FrIf` PDU can be retrieved there.

FrTpPduFc

SWS Item			
Container Name	FrTpPduFc		
Description	This is the identifier of the FlexRay Interface PDUs (Fr N-PDU, Fr L-SDU) in which the Transport Layer Flow Control and Acknowledgement Frames of this channel should be transmitted.		
Configuration Parameters			

Name last modified in: V1.1.0	FrTpPduFcDirection		
Description	This parameter defines the direction of the PDU.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	FrTpFcRx	no acknowledgement	
	FrTpFcTx	no acknowledgement	
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency			

Name last modified in: V1.1.0	FrTpPduFcId {FRTP_PDU_FC}		
Description	This is the identifier of the FlexRay Interface PDUs (Fr N-PDU, Fr L-SDU) in which the Transport Layer Flow Control and Acknowledgement Frames of this channel should be transmitted.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpPduFcRef		
Description			
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency **No Included Containers**			

4.6.3.4 FlexRay TP Connection Configuration

Each `FrTpChannel` has several `FrTpConnection`s. Each `FrTpConnection` as an ID (called `FrTpSduId`) that represents this connection to the upper layer. The reference `FrTpConPduRef` binds this message to the used PDUs that are defined in `FrTpChannel`. The structure of the `FrTpConnection` is shown in figure 4.81.

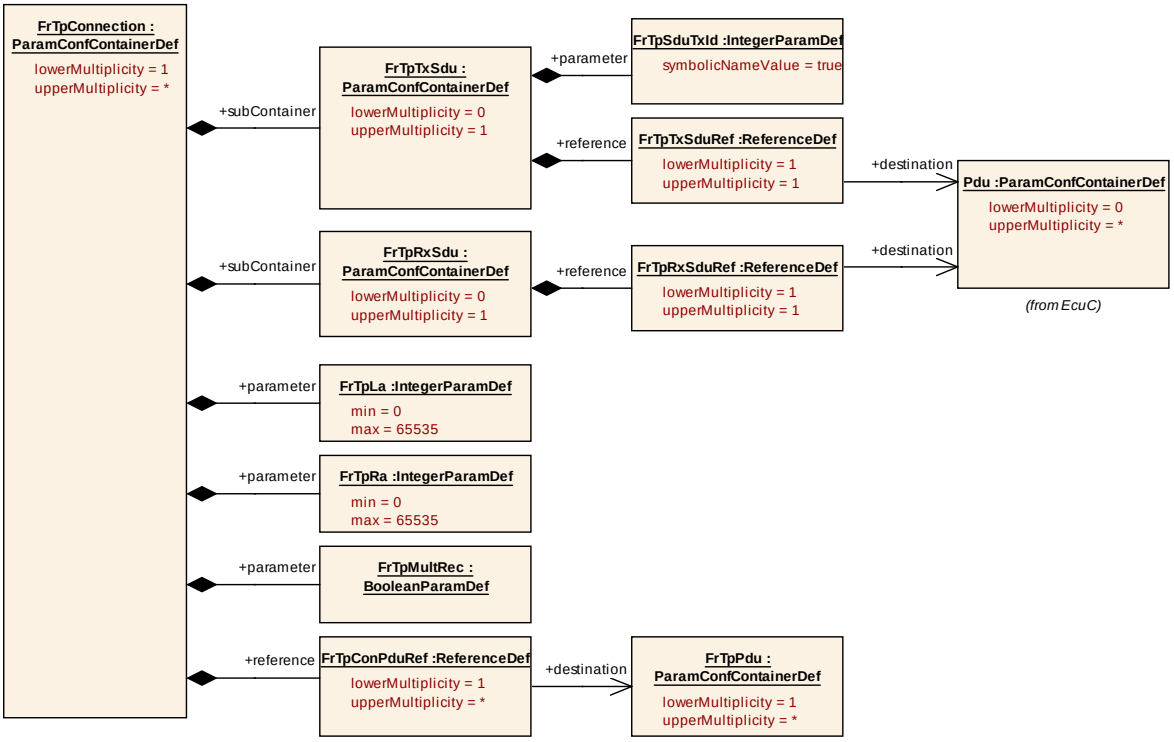


Figure 4.81: FlexRay Transport Layer Connection Configuration

FrTpConnection

SWS Item			
Container Name	FrTpConnection {FrTpConnectionConfiguration}		
Description	This container contains the configuration (parameters) of one FlexRay TP connection. A connection can only belong to one channel.		
Configuration Parameters			

Name last modified in: V1.1.0	FrTpConPduRef {FRTP_CON_PDU}		
Description	Each value defines a PDU to be used for this connection. Thus each value is a PDU-ID given in FRTP_PDUID and this array cannot be longer than the array FRTP_PDUID. Please note: Only PDUs of the same size shall be used within a connection. Of course the PDU having the TxConfirmation configured has to be used by every connection.		
Multiplicity	1..*		
Type	Reference to FrTpPdu		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpLa {FRTP_LA}		
Description	This parameter defines the Local Address for the respective connection. When the local instance is the sender, this is the Source Address within the TP frame. When the local instance is the receiver, this is the Target Address within the TP frame. Note that in case of 1 byte addressing only the values from 0x0000 - 0x00FF are valid.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpMultRec {FRTP_MULT_REC}		
Description	This parameter defines, whether this connection is an 1:1 (’false’) or an 1:n (’true’) connection. Of course, if the channel to which the connection is configured has retry or acknowledgement enabled, no retry or acknowledgement will occur in case the connection is an 1:n connection.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpRa {FRTP_RA}		
Description	This parameter defines the Remote Address for the respective connection. When the local instance is the sender, this is the Target Address within the TP frame. When the local instance is the receiver, this is the Source Address within the TP frame. Note that in case of 1 byte addressing only the values from 0x0000 - 0x00FF are valid.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency	scope: module		
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
FrTpRxSdu	0..1		
FrTpTxSdu	0..1		

FrTpTxSdu

SWS Item	
Container Name	FrTpTxSdu
Description	
Configuration Parameters	

Name last modified in: V1.1.0	FrTpSduTxId {FRTP_SDUID}		
Description	This is a unique identifier for a received or a to be transmitted message. With this (and by means of e.g. a lookup table) the PDU Router can route the message appropriately without dealing with the particularities of the Transport Layer. This parameter can also be seen as the identifier of a connection.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTpTxSduRef		
Description			
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency			
No Included Containers			

FrTpRxSdu

SWS Item			
Container Name	FrTpRxSdu		
Description			
Configuration Parameters			

Name last modified in: V1.1.0	FrTpRxSduRef		
Description			
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	X	Pre Compile
	Link time	–	
	Post-build time	X	Pre Compile and Post Build
Scope / Dependency			
No Included Containers			

4.6.4 FlexRay Transceiver Driver

The FlexRay Transceiver Driver configuration parameters are structured according to figure 4.82.

- FrTrcvGeneral in section 4.6.4.1
- FrTrcvNode in section 4.6.4.2

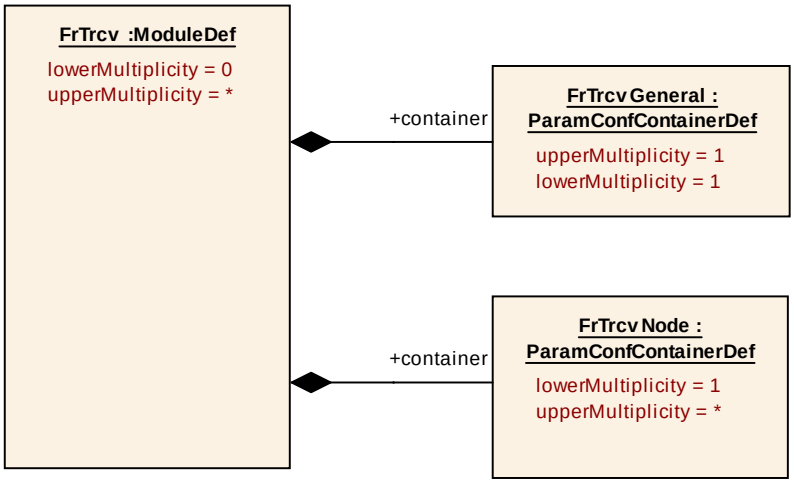


Figure 4.82: FlexRay Transceiver Driver Parameters

Module Name	FrTrcv	
Module Description	Configuration of the FrTrcv (FlexRay Transceiver driver) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
FrTrcvGeneral	1	Container gives CAN transceiver driver basic information.
FrTrcvNode	1..*	Container gives FlexRay transceiver driver information about a single FlexRay transceiver node. Any FlexRay transceiver driver has such FlexRay transceiver nodes.

4.6.4.1 FrTrcvGeneral

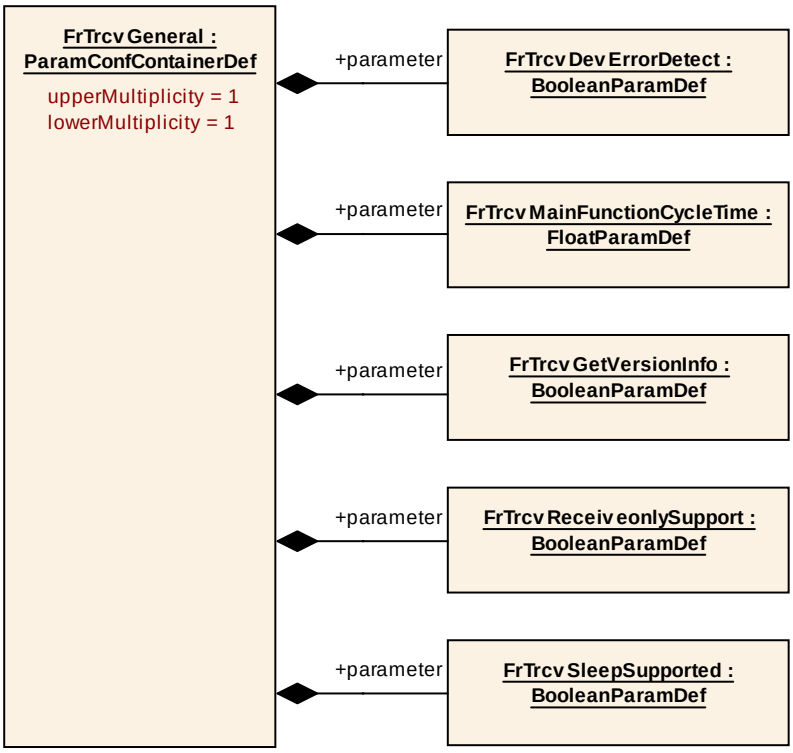


Figure 4.83: FlexRay Transceiver Driver General Parameters

FrTrcvGeneral

SWS Item	[FrTrcv055,FrTrcv106,FrTrcv123]
Container Name	FrTrcvGeneral {FlexRayTransceiverDriverBasic}
Description	Container gives CAN transceiver driver basic information.
Configuration Parameters	

Name last modified in: V1.0.0	FrTrcvDevErrorDetect {FRTRCV_DEV_ERROR_DETECT}		
Description	Switches development error detection and notification on and off. If switched on, #define FRTRCV_DEV _ERROR_DETECT ON shall be generated. If switched off, #define FRTRCV_DEV_ERROR _DETECT OFF shall be generated. Define shall be part of file FrTrcv_Cfg.h.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	FrTrcvGetVersionInfo {FRTRCV_GET_VERSION_INFO}		
Description	Switches version information API on and off. If switched off, function need not be present in compiled code.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	FrTrcvMainFunctionCycleTime {FRTRCV_WAKEUP_POLLING}		
Description	Cyclic call time for function FrTrcvMainFunction in seconds. A call time of 0ms indicates no calls for this function. In this case function need not be present in compiled code.		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module dependency: FRTRCV_MAIN_FUNCTION_CYCLE_TIME		

Name last modified in: V1.1.0	FrTrcvReceiveonlySupport {FRTRCV_RECEIVEONLY_SUPPORT}		
Description	Information if the optional transceiver state RECEIVEONLY is supported by the driver and hardware.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	FrTrcvSleepSupported {FRTRCV_SLEEP_SUPPORTED}		
Description	Information if the optional transceiver state SLEEP is supported by the driver and hardware.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		
No Included Containers			

4.6.4.2 FrTrcvNode

Each FlexRay transceiver driver supports one or several FlexRay nodes.

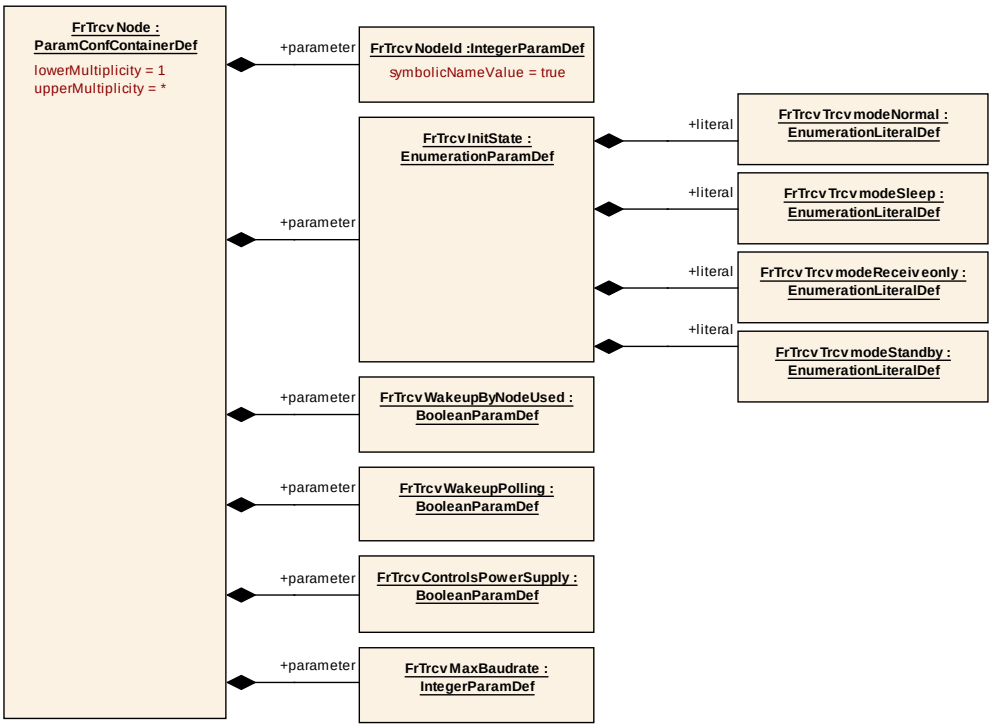


Figure 4.84: FlexRay Transceiver Driver Node Parameters

FrTrcvNode

SWS Item	[FrTrcv091]		
Container Name	FrTrcvNode {FlexRayTransceiverNode}		
Description	Container gives FlexRay transceiver driver information about a single FlexRay transceiver node. Any FlexRay transceiver driver has such FlexRay transceiver nodes.		
Configuration Parameters			
Name last modified in: V1.0.0	FrTrcvControlsPowerSupply {FRTRCV_CONTROLS_POWER_SUPPLY}		
Description	Is ECU power supply controlled by this transceiver?		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Instance		

Name last modified in: V1.0.0	FrTrcvInitState {FRTRCV_INIT_STATE}		
Description	State of FlexRay transceiver after power on.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	FrTrcvTrcvmodeNormal		
	FrTrcvTrcvmode Receiveonly		
	FrTrcvTrcvmodeSleep		
	FrTrcvTrcvmodeStandby		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Instance		

Name last modified in: V1.1.0	FrTrcvMaxBaudrate {FRTRCV_MAX_BAUDRATE}		
Description	Max baudrate for transceiver hardware type. Only used for validation purposes. Value shall be configured by configuration tool based on FRTRCV_HARDWARE_NAME and internal information about ability of this hardware typel.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Instance		

Name last modified in: V1.1.0	FrTrcvNodeId {FRTRCV_NODE_ID}		
Description	Unique node id. It is used by ComM by Icu and internally.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Instance		

Name last modified in: V1.1.0	FrTrcvWakeupByNodeUsed {FRTRCV_WAKEUP_BY_NODE_USED}		
Description	Is wake up by node supported? If FlexRay transceiver hardware does not support wake up by node value is always FALSE. If FlexRay transceiver hardware supports wake up by node value is TRUE or FALSE depending whether it is used or not.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Instance dependency: FRTRCV_WAKEUP_POLLING		

Name last modified in: V1.1.0	FrTrcvWakeupPolling {FRTRCV_WAKEUP_POLLING}		
Description	Information of this entitiy is only relevant if parameter FrTrcvWakeup-ByTransceiverUsed is TRUE. Information whether wake up events will be performed in main function (TRUE) or by a callback function called by an interrupt service (FALSE).		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Instance dependency: FRTRCV_WAKEUP_BY_NODE_USED, FRTRCV_MAIN_FUNCTION_CYCLE_TIME		
No Included Containers			

4.7 LIN

This section depicts the configuration parameters of the LIN communication stack, namely LIN Interface, LIN Transport Protocol and LIN Driver (see figure 4.85).

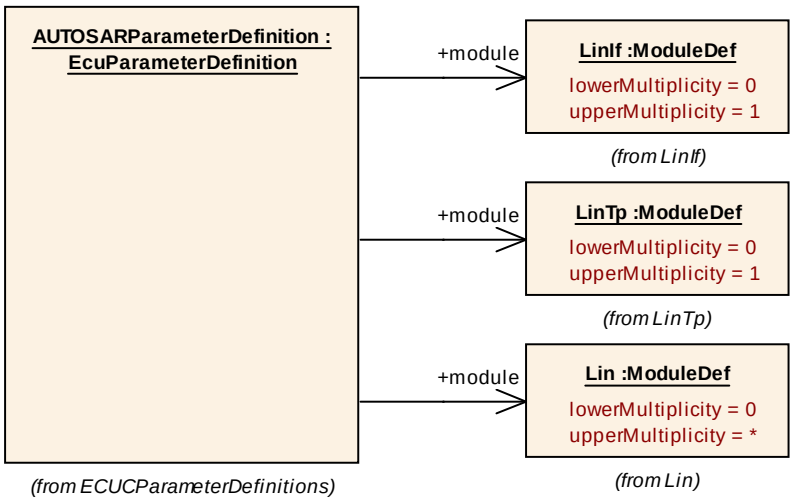


Figure 4.85: Overview of the LIN communication stack

4.7.1 LIN Interface

This section lists all configuration parameters of the LIN Interface as described in [24]. Figure 4.86 pictures the top-level structure of this module.

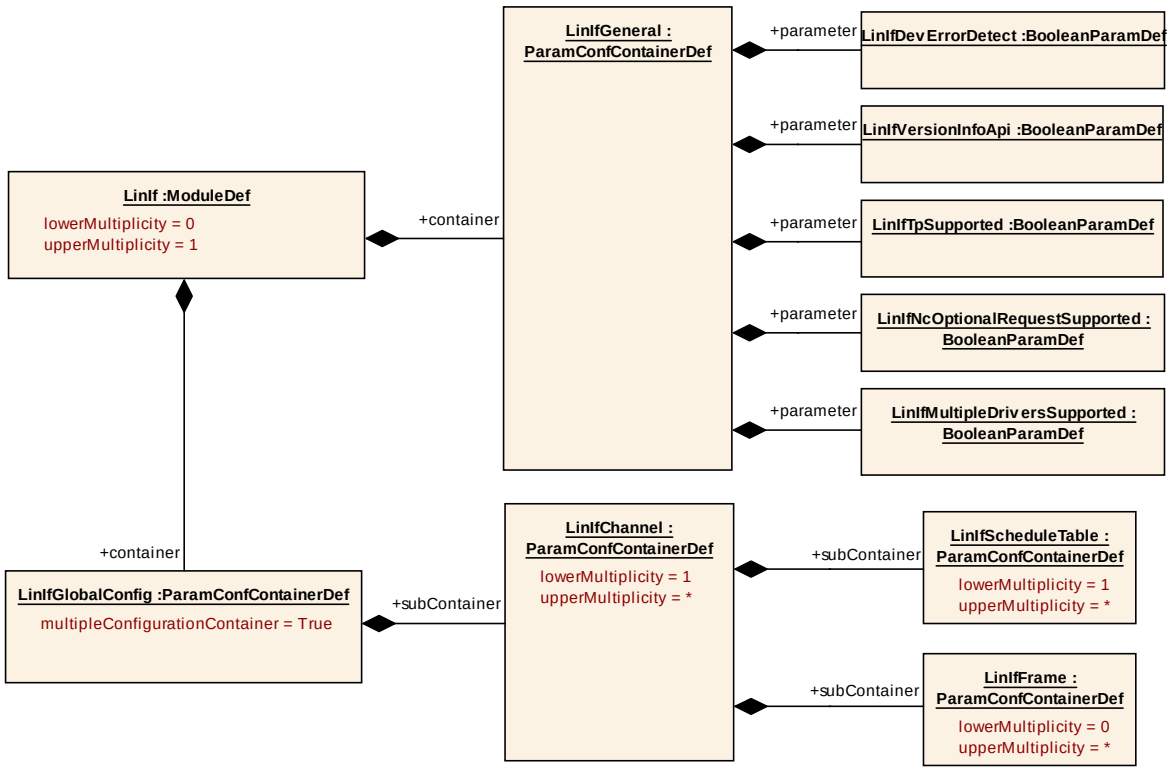


Figure 4.86: LIN Interface configuration

Module Name	LinIf	
Module Description	Configuration of the LinIf (LIN Interface) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
LinIfGeneral	1	
LinIfGlobalConfig	1	This container contains the global configuration parameter of the LinIf. It is a MultipleConfigurationContainer, i.e. this container and its sub-containers exit once per configuration set.

The LIN interface global configuration parameters are gathered by the `LinIfGeneral` container as described below.

LinIfGeneral

SWS Item			
Container Name	LinIfGeneral		
Description			
Configuration Parameters			

Name last modified in: V1.0.0	LinIfDevErrorDetect {LINIF_DEV_ERROR_DETECT}		
Description	Switches the Development Error Detection and Notification ON or OFF.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	LinIfMultipleDriversSupported {LINIF_MULTIPLE_DRIVER_SUPPORT}		
Description	States if multiple drivers are included in the LIN Interface or not. The reason for this parameter is to reduce the size of LIN Interface if multiple drivers are not used.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	LinIfNcOptionalRequestSupported {LINIF_NC_OPTIONAL_REQUEST_SUPPORTED}		
Description	States if the node configuration commands Assign NAD and Conditional Change NAD are supported.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	LinIfTpSupported {LINIF_TP_SUPPORTED}		
Description	States if the TP is included in the LIN Interface or not. The reason for this parameter is to reduce the size of LIN Interface if the TP is not used.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	LinIfVersionInfoApi {LINIF_VERSION_INFO_API}		
Description	Switches the LinIf_GetVersionInfo function ON or OFF.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

No Included Containers

4.7.1.1 LIN Interface Channel Configuration

The LIN interface correlates a phyiscal LIN channel in the LIN driver with according `LinIfFrame`s and `LinIfScheduleTable`s. Hence it defines its own `LinIfChannel` that is depicted in figure 4.87.

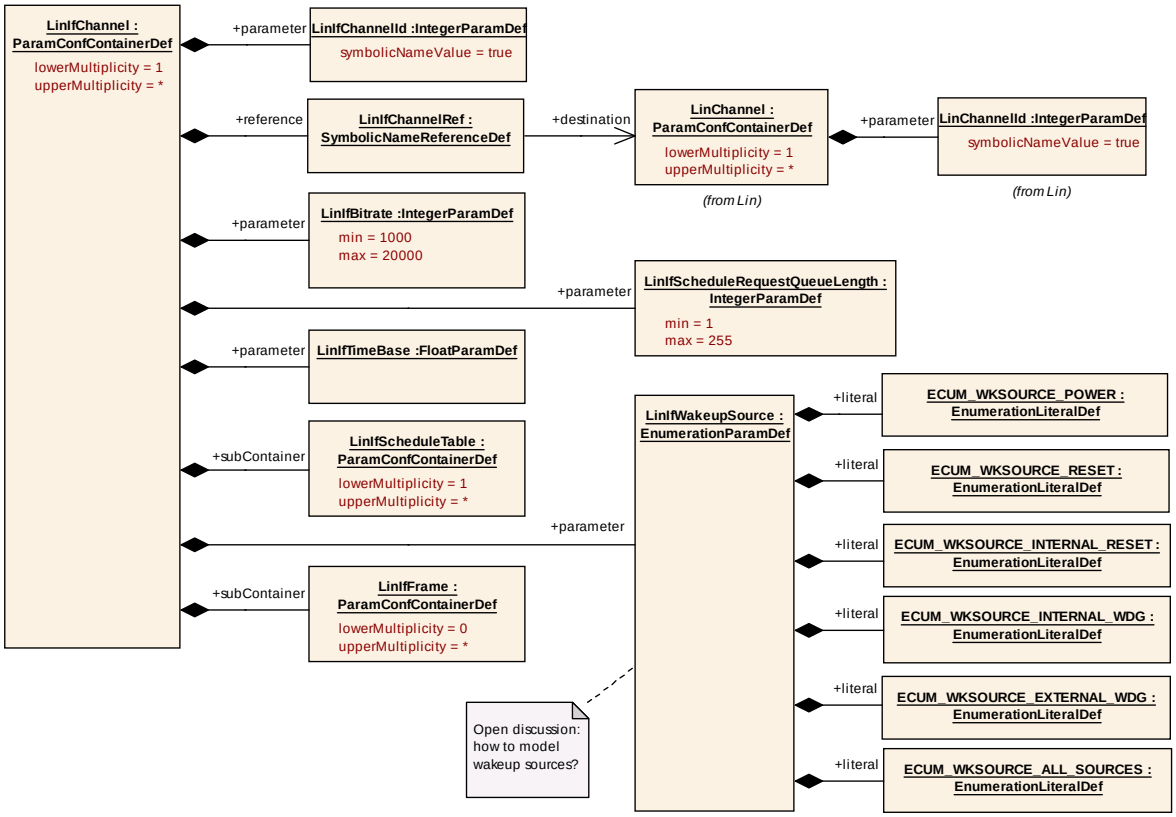


Figure 4.87: LIN Interface Channel configuration

LinIfChannel

SWS Item	[LINIF364]
Container Name	LinIfChannel {LinIf_Channel}
Description	
Configuration Parameters	

Name last modified in: V1.0.0	LinIfBitrate {LINIF_BITRATE}		
Description	Bitrate of the channel. This parameter is needed to calculate the delays in the schedule tables.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1000 .. 20000		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	LinIfChannelId		
Description	Internal ID for the channel on LIN Interface level		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	LinIfChannelRef {LINIF_CHANNEL_INDEX}		
Description	Reference to the used channel in Lin. Replaces LINIF_CHANNEL_INDEX		
Multiplicity	1		
Type	Symbolic name reference to LinChannel		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	LinIfScheduleRequestQueueLength {LINIF_SCHEDULE_REQUEST_QUEUE_LENGTH}		
Description	Number of schedule requests the schedule table manager can handle for this channel.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 255		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	LinIfTimeBase {LINIF_TIME_BASE}		
Description	The time-base for this channel in [s]. (Usually 0.002, 0.005 or 0.010 s)		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	LinIfWakeupSource {LINIF_WAKEUP_SOURCE}		
Description	Bit-field with one unique bit set for each channel. This is used when indicating a wakeup to the ECU State Manager. Open discussion: how to model wakeup sources?		
Multiplicity	1		
Type	EnumerationParamDef		
Range	ECUM_WKSOURCE_ALL_SOURCES		
	ECUM_WKSOURCE_EXTERNAL_WDG	Reset by external watchdog (bit 4), if detection supported by hardware.	
	ECUM_WKSOURCE_INTERNAL_RESET	Internal reset of μC (bit 2) The internal reset typically only resets the μC core but not peripherals or memory controllers. The exact behavior is hardware specific. This source may also indicate an unhandled exception.	
	ECUM_WKSOURCE_INTERNAL_WDG	Reset by internal watchdog (bit 3)	
	ECUM_WKSOURCE_POWER	Power cycle (bit 0)	
	ECUM_WKSOURCE_RESET	Hardware reset (bit 1). If hardware cannot distinguish between a power cycle and a reset reason, then this shall be the default wakeup source.	
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			
Included Containers			

Container Name	**Multiplicity**	**Scope / Depedency**
LinIfFrame	0..*	
LinIfScheduleTable	1..*	

In case of multiple LIN drivers handling, the LinIf can use the `LinIfChannelRef` parameter to retrieve the `VendorId` and `VendorApiInFix` common published parameters of the LinChannel.

4.7.1.2 LIN Interface Frame Configuration

Figure 4.88 shows the structure of the `LinIfFrame`. In there you'll find the association of a `LinIfFrame` to the global PDU (see section 4.1) provided by `LinIfRxPdu` and `LinIfTxPdu`.

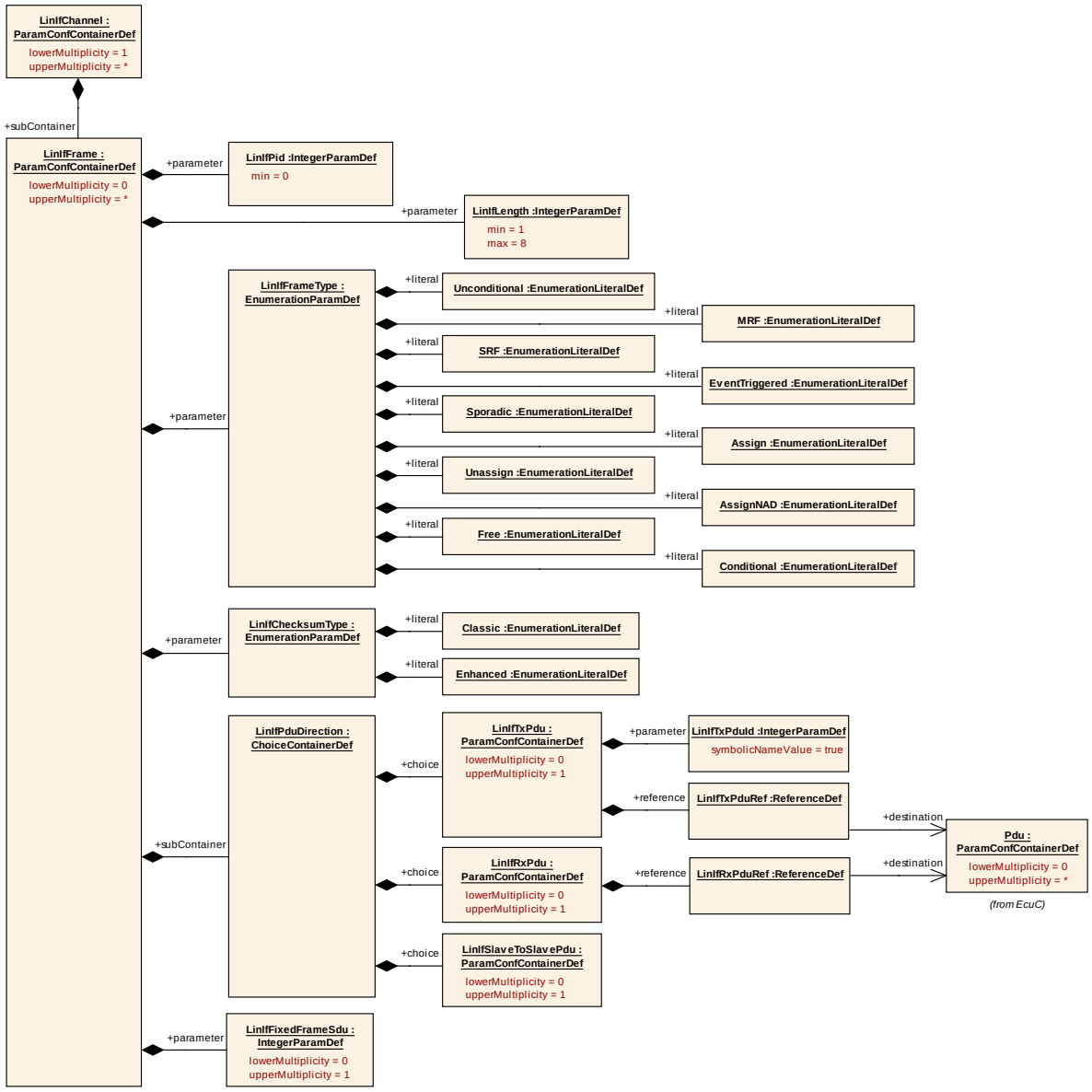


Figure 4.88: LIN Interface Frame configuration

LinIfFrame

SWS Item	[LINIF367]		
Container Name	LinIfFrame {LinIf_Frame}		
Description	Generic container for all types of LIN frames.		
Configuration Parameters			

Name last modified in: V1.0.0	LinIfChecksumType {LINIF_CHECKSUM_TYPE}		
Description	Type of checksum that the frame is using.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	Classic	Classic	
	Enhanced	Enhanced	
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	LinIfFixedFrameSdu {LINIF_FIXED_FRAME_SDU}		
Description	In case this is a fixed frame this is the SDU (response). This value should represent an eight byte array.		
Multiplicity	0..1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	LinIfFrameType {LINIF_FRAME_TYPE}		
Description	Type of frame that is described (e.g. sporadic frame). Note that types 7-11 are the fixed MRF types. The sporadic slot is not found among the frame types. A sporadic slot is a set of sporadic frames.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	Assign	AssignFrameId	
	AssignNAD	AssignNAD	
	Conditional	Conditional Change NAD	
	EventTriggered	Event triggered frame	
	Free	FreeFormat	
	MRF	MRF	
	SRF	SRF	
	Sporadic	Sporadic frame	
	Unassign	UnassignFrameId	
	Unconditional	Unconditional Frame	
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	LinIfLength {LINIF_LENGTH}		
Description	Length of the LIN SDU in bytes.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 8		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	LinIfPid {LINIF_PID}		
Description	Protected ID of the LIN frame. There is no reason to calculate the Parity in run-time.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Included Containers

Container Name	Multiplicity	Scope / Depedency
LinIfPduDirection [LinIfRxPdu, LinIfSlaveTo SlavePdu, LinIfTxPdu]	1	

In contrast to the definition of configuration paramater `LINIF_DIRECTION` in [24] the ECU Configuration parameter `LinIfPduDirection` wasn't just specified as of type boolean. Due to the formalization strategy in the ECU Configuration each local PDU needs to be associated with a globally defined PDU, so that an upper layer can detect, which foreign ID to use when calling a lower layer. To introduce this reference mechanism that direction parameter was adopted accordingly.

LinIfRxPdu

SWS Item			
Container Name	LinIfRxPdu		
Description	represents a received PDU/frame		
Configuration Parameters			

Name last modified in: V1.0.0	LinIfRxPduRef		
Description	Reference to the PDU that is received in this frame.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			

No Included Containers

LinIfTxPdu

SWS Item			
Container Name	LinIfTxPdu		
Description	represents a transmitted PDU/frame		
Configuration Parameters			

Name last modified in: V1.0.0	LinIfTxPduId {LINIF_PDU_ID}		
Description	Identifier of the frame for the upper layer. This id is only relevant for sporadic frames.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	LinIfTxPduRef		
Description	Reference to the PDU that is transmitted in this frame.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

LinIfSlaveToSlavePdu

SWS Item	
Container Name	LinIfSlaveToSlavePdu
Description	represents a slave-to-slave PDU/frame. Master does only send the header but doesn't receive the response. Added for completeness
Configuration Parameters	
No Included Containers	

4.7.1.3 LIN Interface Schedule Table Configuration

Figure 4.89 depicts the structure of the configuration parameters of a LIN interface's schedule table.

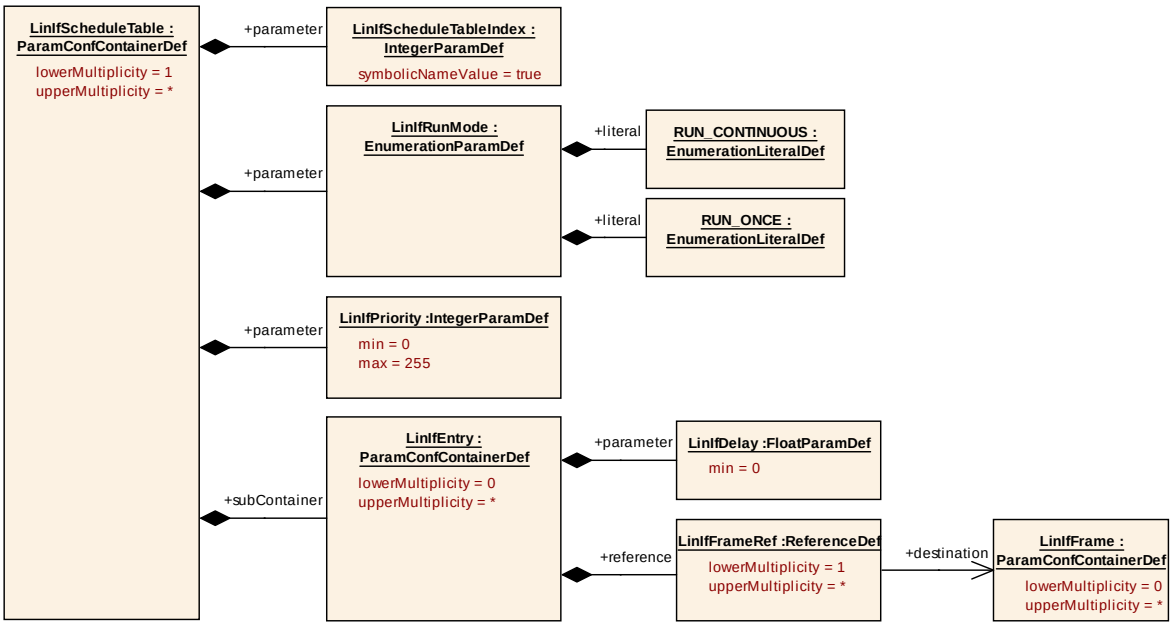


Figure 4.89: LIN Interface Schedule Table configuration

LinIfScheduleTable

SWS Item			
Container Name	LinIfScheduleTable {LinIf_ScheduleTable}		
Description	Describes a schedule table. Each LinIfChannel may have several schedule tables.		
Configuration Parameters			

Name last modified in: V1.0.0	LinIfPriority {LINIF_PRIORITY}		
Description	Priority of the schedule table. The priority is used in the schedule table manager. The RUN_ONCE run mode schedules shall not have equal priority. Priority 0 is reserved for the NULL_SCHEDULE. Priority 255 is reserved for the RUN_CONTINUOUS run mode.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	LinIfRunMode {LINIF_RUN_MODE}		
Description	The schedule table can be executed in two different modes.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	RUN_CONTINUOUS		
	RUN_ONCE		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	LinIfScheduleTableIndex {LINIF_SCHEDULE_INDEX}		
Description	This is the unique index used by upper layers to identify a schedule. Note that the NULL_SCHEDULE for each channel has index 0.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
LinIfEntry	0..*		

LinIfEntry

SWS Item	[LINIF366]		
Container Name	LinIfEntry {LinIf_Entry}		
Description	Describes an entry in the schedule table (also known as Frame Slot).		
Configuration Parameters			

Name last modified in: V1.0.0	LinIfDelay {LINIF_DELAY}		
Description	Delay to next frame in schedule table in [s]		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	LinIfFrameRef		
Description	Reference to the frames that belong to this schedule table entry.		
Multiplicity	1..*		
Type	Reference to LinIfFrame		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			
No Included Containers			

4.7.2 LIN Driver

This section depicts the configuration parameters of LIN Driver as specified in [23]. Figure 4.90 pictures the general configurations parameters of the LIN driver and the managed channels.

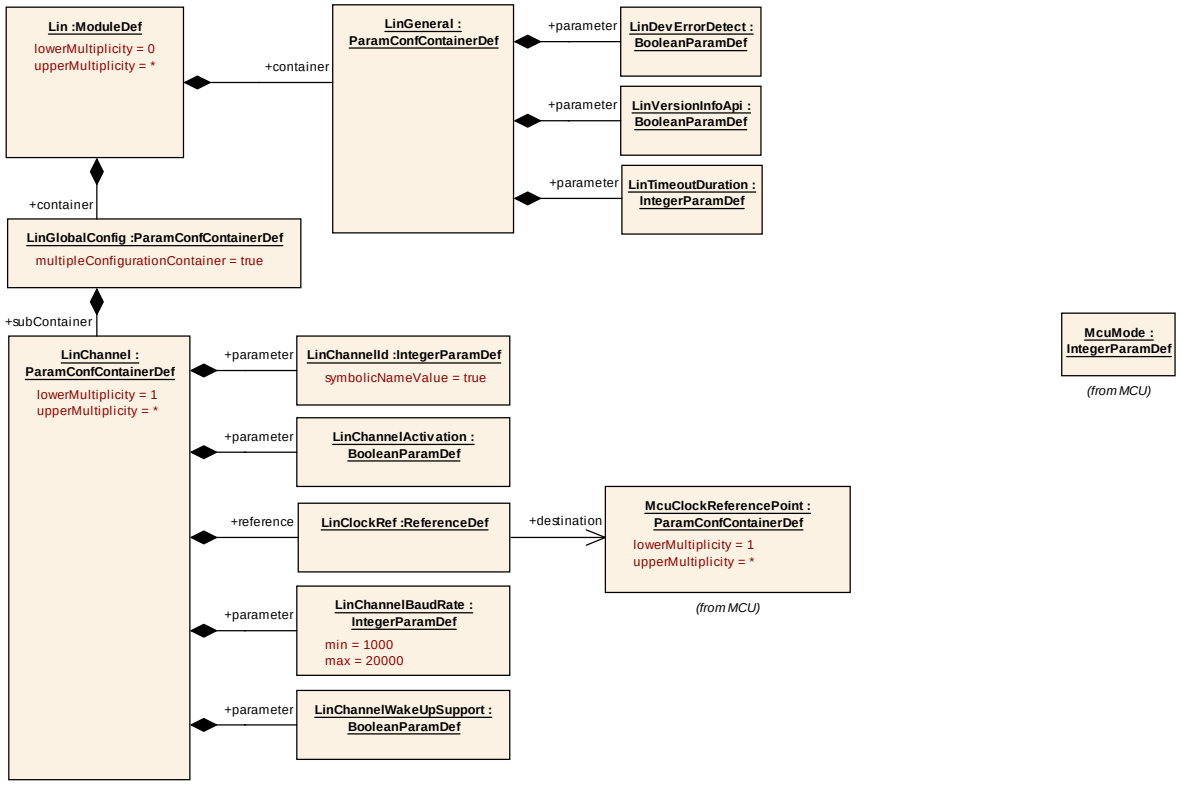


Figure 4.90: LIN Driver configuration

Module Name	Lin	
Module Description	Configuration of the Lin (LIN driver) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
LinGeneral	1	
LinGlobalConfig	1	This container contains the global configuration parameter of the Lin driver. This container is a MultipleConfigurationContainer, i.e. this container and its sub-containers exit once per configuration set.

LinGeneral

SWS Item			
Container Name	LinGeneral		
Description			
Configuration Parameters			

Name last modified in: V1.0.0	LinDevErrorDetect {LIN_DEV_ERROR_DETECT} [LIN066]		
Description	Switches the Development Error Detection and Notification ON or OFF.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	LinTimeoutDuration {LIN_TIMEOUT_DURATION} [LIN093]		
Description	Specifies the maximum number of loops for blocking function until a timeout is raised in short term wait loops		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	LinVersionInfoApi {LIN_VERSION_INFO_API} [LIN067]		
Description	Switches the Lin_GetVersionInfo function ON or OFF.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

LinChannel

SWS Item	[LIN069]
Container Name	LinChannel
Description	This container contains teh configuration (parameters) of the LIN Controller(s).
Configuration Parameters	

Name last modified in: V1.0.0	LinChannelActivation {LIN_CHANNEL_ACTIVATION}		
Description	Defines if a LIN channel is used in the configuration.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	Variant1
	Link time	–	
	Post-build time	M	Variant2
Scope / Dependency			

Name last modified in: V1.0.0	LinChannelBaudRate {LIN_CHANNEL_BAUD_RATE}		
Description	Specifies the baud rate of the LIN channel		
Multiplicity	1		
Type	IntegerParamDef		
Range	1000 .. 20000		
Configuration Class	**Pre-compile time**	X	Variant1
	Link time	–	
	Post-build time	M	Variant2
Scope / Dependency			

Name last modified in: V1.0.0	LinChannelId		
Description	Identifies the LIN channel. Replaces LIN_CHANNEL_INDEX_NAME from the LIN SWS.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	LinChannelWakeUpSupport {LIN_CHANNEL_WAKE_UP_SUPPORT}		
Description	Specifies if the LIN hardware channel supports wake up functionality		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	LinClockRef {LIN_CLOCK_SRC_REFERENCE} [LIN094]		
Description	Reference to the LIN clock source configuration, which is set in the MCU driver configuration.		
Multiplicity	1		
Type	Reference to McuClockReferencePoint		
Configuration Class	**Pre-compile time**	X	Variant1
	Link time	–	
	Post-build time	X	Variant2
Scope / Dependency			
No Included Containers			

4.7.3 LIN Transport Layer

This section depicts the configuration parameters of LIN TP as specified in [24].

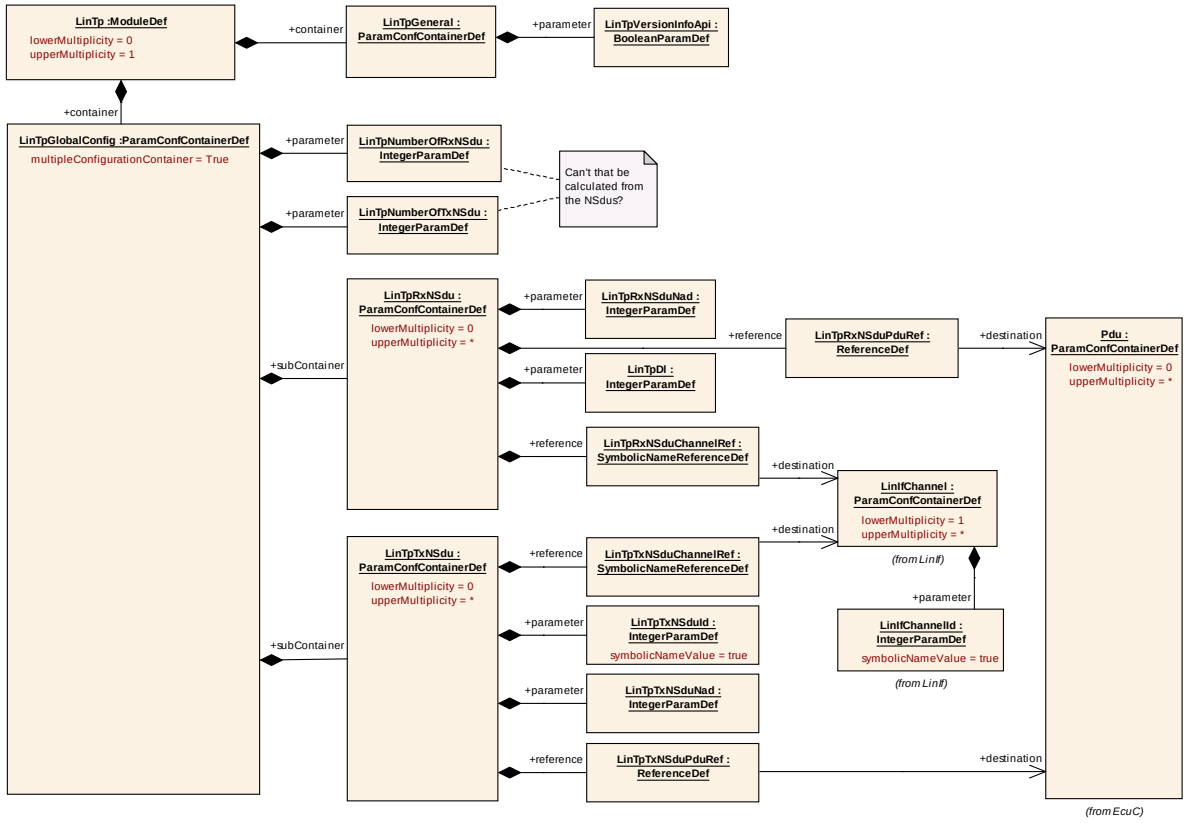


Figure 4.91: LIN Transport Layer configuration

Module Name	LinTp	
Module Description	Singleton descriptor for the LIN Transport Protocol.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
LinTpGeneral	1	Container that holds all LIN transport protocol general parameters.
LinTpGlobalConfig	1	This container contains the global configuration parameter of the LinTp. It is a MultipleConfigurationContainer, i.e. this container and its sub-containers exit once per configuration set.

LinTpGeneral

SWS Item			
Container Name	LinTpGeneral		
Description	Container that holds all LIN transport protocol general parameters.		
Configuration Parameters			

Name last modified in: V1.0.0	LinTpVersionInfoApi {LINTP_VERSION_INFO_API}		
Description	Switches the LinTp_GetVersionInfo function ON or OFF.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

LinTpRxNSdu

SWS Item	[LINIF428]		
Container Name	LinTpRxNSdu {LinTp_rx_NSDU}		
Description	For each received N-SDU on any channel the node is connected to.		
Configuration Parameters			
Name last modified in: V1.0.0	LinTpDl {LINTP_DL}		
Description	Data Length Code of this RxNsdu. In case of variable length message, this value indicates the minimum data length. Range of minimum length is 1 to 4095. Note that this is not relevant for Tx. The reason for this is to have identical structures for Tx and Rx.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			
Name last modified in: V1.0.0	LinTpRxNSduChannelRef {LINTP_CHANNEL_INDEX}		
Description	Index of the channel this N-SDU belongs to.		
Multiplicity	1		
Type	Symbolic name reference to LinIfChannel		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			
Name last modified in: V1.0.0	LinTpRxNSduNad {LINTP_NAD}		
Description	A N-SDU transported on LIN is identified using the NAD for the specific slave.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			
Name last modified in: V1.0.0	LinTpRxNSduPduRef		
Description	Reference to the global PDU		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

LinTpTxNSdu

SWS Item	[LINIF428]		
Container Name	LinTpTxNSdu {LinTp_tx_NSDU}		
Description	For each transmitted N-SDU on any channel the node is connected to.		
Configuration Parameters			

Name last modified in: V1.0.0	LinTpTxNSduChannelRef {LINTP_CHANNEL_INDEX}		
Description	Index of the channel this N-SDU belongs to.		
Multiplicity	1		
Type	Symbolic name reference to LinIfChannel		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	LinTpTxNSduId {LINTP_NSDU_ID}		
Description	The identifier of the Transport Protocol message. This ID will be the one that is communicated with upper layers.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	LinTpTxNSduNad {LINTP_NAD}		
Description	A N-SDU transported on LIN is identified using the NAD for the specific slave.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.0.0	LinTpTxNSduPduRef		
Description	Reference to the global PDU		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

4.8 Network Management

The Network Management Stack consists of the modules:

- Network Management Interface(Nm). See chapter 4.8.1
- Can Generic Network Management(CNm). See chapter 4.8.2
- Can Network Management(CanNm). See chapter 4.8.3
- Flexray Network Management(FrNm). See chapter 4.8.4

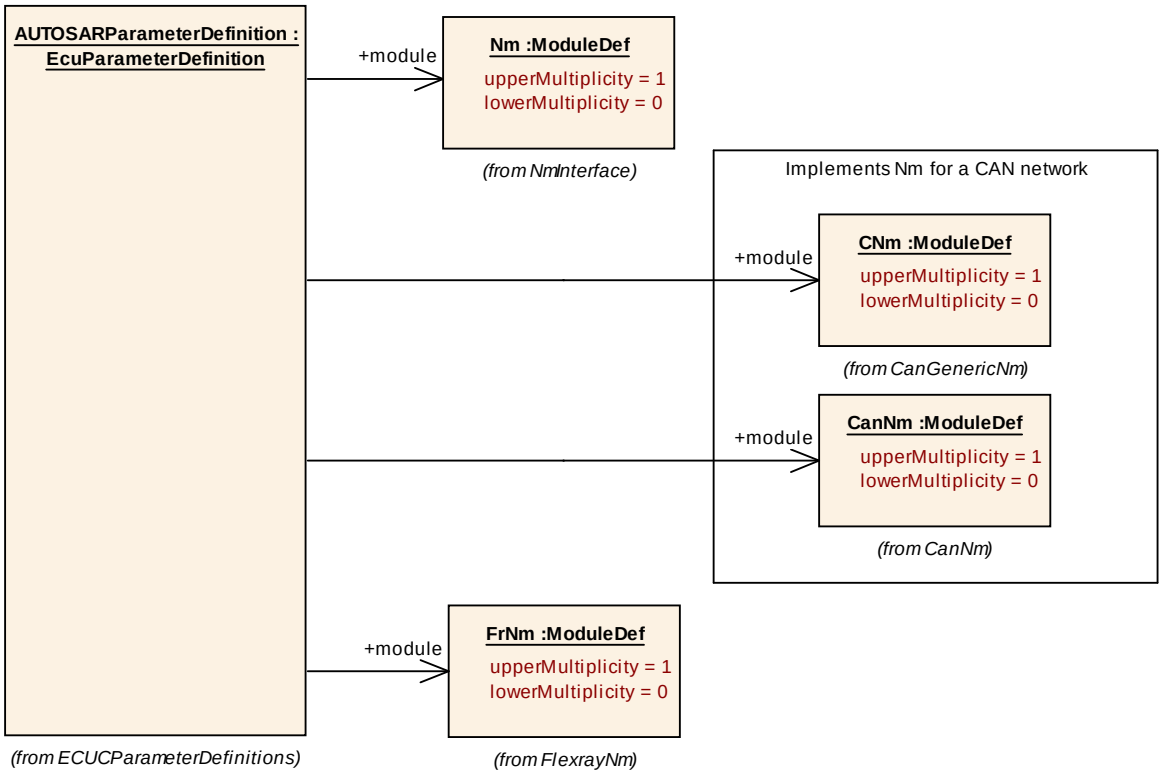


Figure 4.92: Overview of Network Management Stack

Throughout the Network Management stack there are several dependencies between parameters in the different modules. E.g. Enabling a function in CanNm or CNm can only be done if the corresponding function is enabled in Nm. There are also dependencies within the different modules. For some parameters the dependencies are desribed by using derived parameters. A derived paramter can be computed by using a calculation language and a calculation formula. An Informal calculation language is used.

For derived parameters the calculation formula is defined in the description of each derived parameter. The notation used is: calculationFormula = "'The formula"'. An example: The derived parameter CNmChannelId has an calculationFormula = Equal(NmChannelId). This means that the value of the parameter is the value of the NmChannelId parameter.

4.8.1 Network Management Interface

TODO *This chapter is based on a draft version of the SWS. (V0.9 Dated 09.23.2996).*
The date should probably be 09.23.2006.
Needs to be updated when the SWS is released.

There is only one variant defined for the Network Management Interface. All parameters are pre-compile time and there is only one container (NmGlobalConfig) for all parameters.

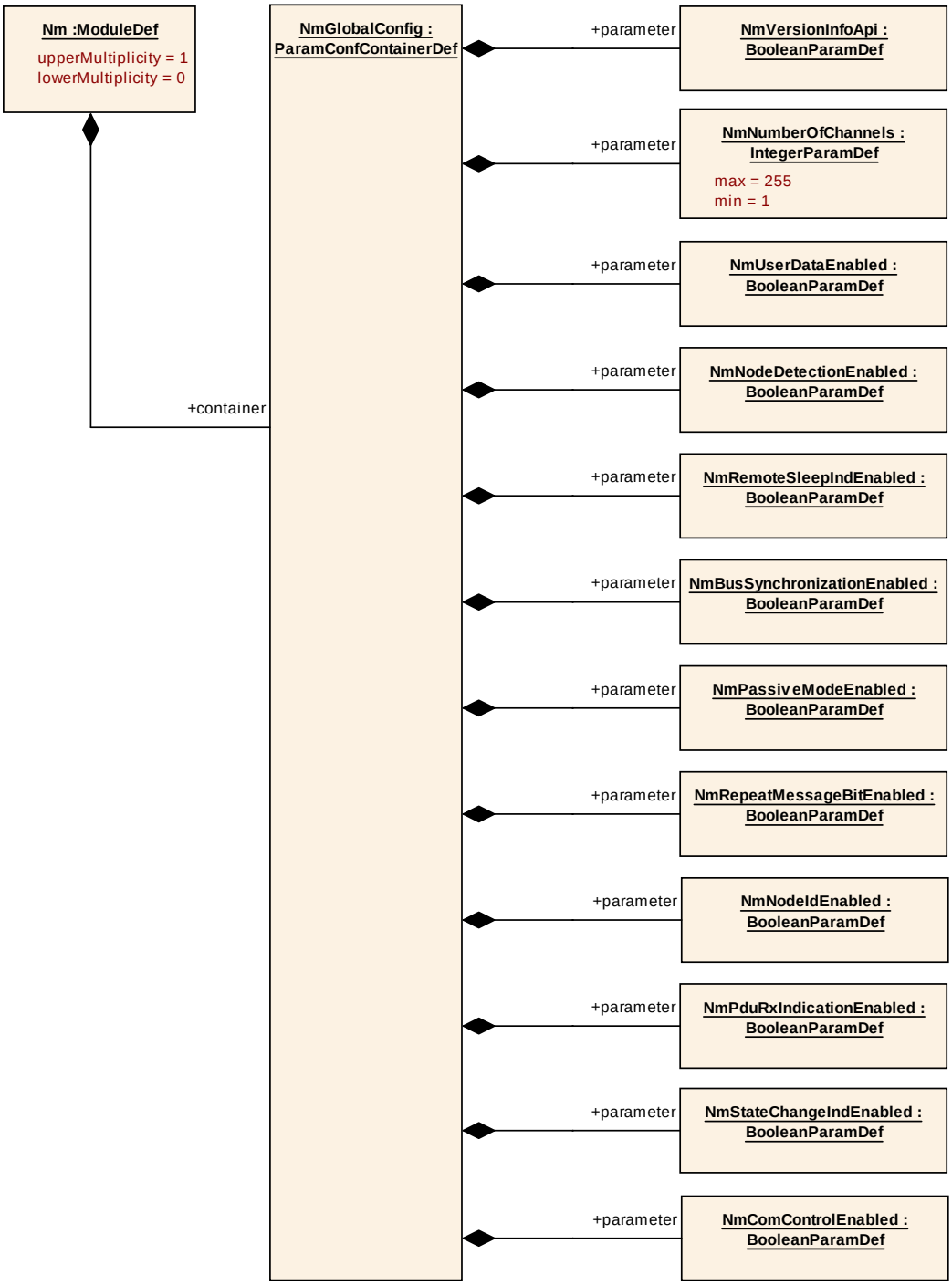


Figure 4.93: Nm Global Parameters.

NmGlobalConfig

SWS Item	
Container Name	NmGlobalConfig {Nm_GlobalConfig}
Description	This container contains all global configuration parameters of the Nm Interface.
Configuration Parameters	

Name last modified in: V1.1.0	NmBusSynchronizationEnabled {NM_BUS_SYNCHRONIZATION_ENABLED}		
Description	This feature is required for gateway nodes only.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	dependency: It must not be enabled if NmPasssiveModeEnabled is enabled.		

Name last modified in: V1.1.0	NmComControlEnabled {NM_COM_CONTROL_ENABLED}		
Description	Pre-processor switch for enabling the Communication Control support.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	NmNodeDetectionEnabled {NM_NODE_DETECTION_ENABLED}		
Description	Pre-processor switch for enabling the Request Repeat Message Request support.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	NmNodeIdEnabled {NM_NODE_ID_ENABLED}		
Description	Pre-processor switch for enabling the source node identifier.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	NmNumberOfChannels {NM_NUMBER_OF_CHANNELS}		
Description	Number of NM channels allowed within one ECU.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 255		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	NmPassiveModeEnabled {NM_PASSIVE_MODE_ENABLED}		
Description	Pre-processor switch for enabling support of the Passive Mode.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	NmPduRxIndicationEnabled {NM_PDU_RX_INDICATION_ENABLED}		
Description	Pre-processor switch for enabling the PDU Rx Indication.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	NmRemoteSleepIndEnabled {NM_REMOTE_SLEEP_IND_ENABLED}		
Description	This feature is required for gateway nodes only.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	dependency: It must not be enabled if NmPassiveModeEnabled is enabled.		

Name last modified in: V1.1.0	NmRepeatMessageBitEnabled {NM_REPEAT_MESSAGE_BIT_ENABLED}		
Description	Pre-processor switch for enabling support for repeat message		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	NmStateChangeIndEnabled {NM_STATE_CHANGE_IND_ENABLED}		
Description	Pre-processor switch for enabling the CAN Generic NM state change notification.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	NmUserDataEnabled {NM_USER_DATA_ENABLED}		
Description	Pre-processor switch for enabling user data support.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	NmVersionInfoApi {NM_VERSION_INFO_API}		
Description	Pre-processor switch for enabling version info API support.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		
No Included Containers			

4.8.2 Can Generic Network Management

TODO *This chapter is based on a draft version of the SWS. (V1.9.3)*
Needs to be updated when the SWS is released.

The structure of the Can Generic Network Management (CNm) configuration parameters are structured as described in figure 4.94 and the containers are:

- CNmGlobalConfig in section 4.8.2.1
- CNmChannelConfig in section 4.8.2.2

There are three variants defined for the module:

- Variant 1: Pre-compile time configuration.
- Variant 2: A mix of pre-compile tima and link time configuration.
- Variant 3: A mix of pre-compile time and post-build time configuration

The module supports post-build loadable and post-build selectable. In case of post-build selectable, the container CNmGlobalConfig is the multiple configuration container.



Figure 4.94: Generic Network Management Parameters.

4.8.2.1 Can Generic Network Management global configuration

Global configuration parameters for the Can Generic Network Management are described in figure 4.95. Some of the parameters are derived parameters.

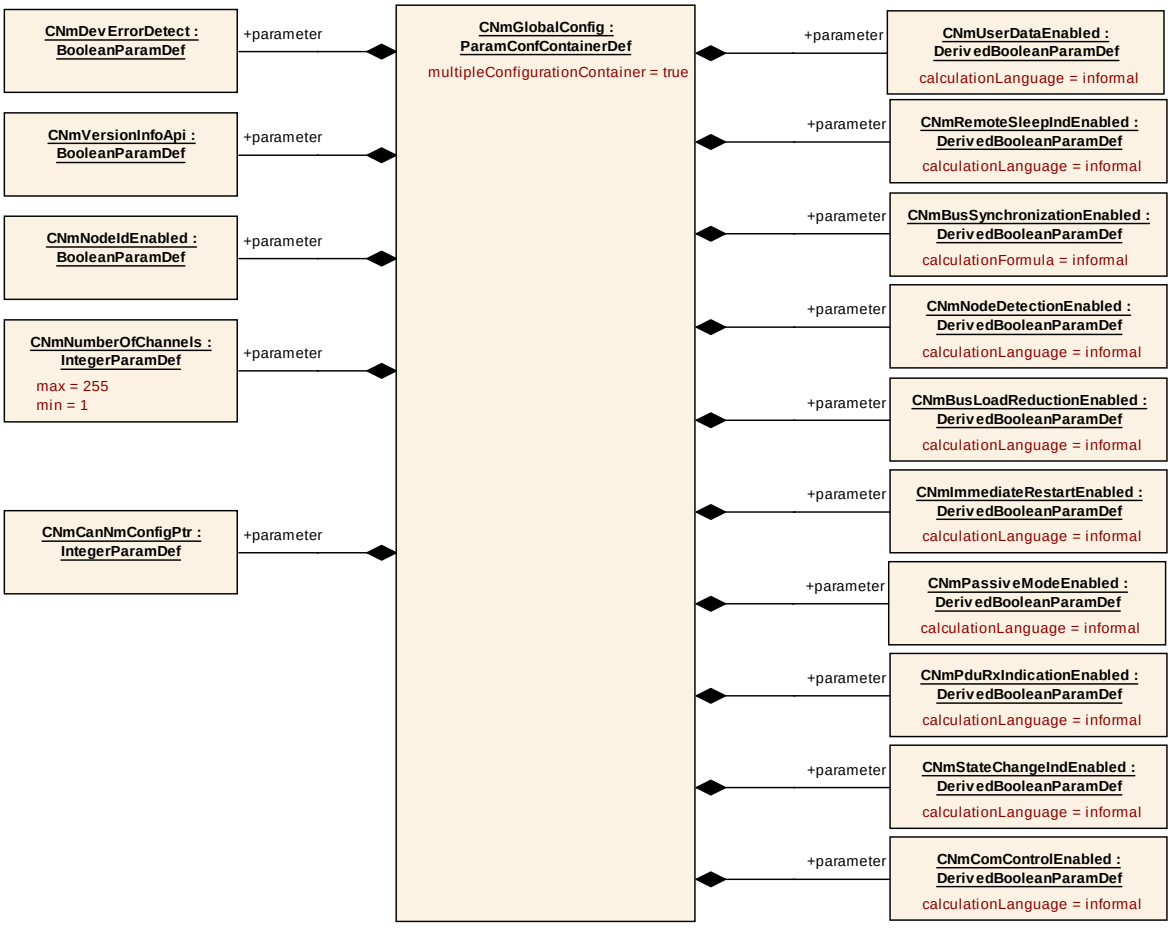


Figure 4.95: Can Generic Nm global parameters

CNmGlobalConfig

SWS Item			
Container Name	CNmGlobalConfig {CNm_GlobalConfig}		
Description	This container contains all global configuration parameters of CAN Generic NM configured from the NM Module perspective.		
Configuration Parameters			

Name last modified in: V1.1.0	CNmBusLoadReductionEnabled {CNM_BUS_LOAD_REDUCTION_ENABLED}		
Description	Pre-processor switch for enabling busload reduction support. calculationFormula = If(CNmPassiveModeEnabled == False) then Equal(NmPassiveModeEnabled) else Equal(False)		
Multiplicity	1		
Type	DerivedBooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	CNmBusSynchronizationEnabled {CNM_BUS_SYNCHRONIZATION_ENABLED}		
Description	Pre-processor switch for enabling bus synchronization support. This feature is required for gateway nodes only. calculationFormula = If (CNmPassiveModeEnabled == True) then Equal(False) else Equal(False or True)		
Multiplicity	1		
Type	DerivedBooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	CNmCanNmConfigPtr {CNM_CANNM_CONFIG_PTR}		
Description	Pointer to configuration of CanNm		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency			

Name last modified in: V1.1.0	CNmComControlEnabled {CNM_COM_CONTROL_ENABLED}		
Description	Pre-processor switch for enabling the Communication Control support. calculationformula = Equal(NmComControlEnabled)		
Multiplicity	1		
Type	DerivedBooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	CNmDevErrorDetect {CNM_DEV_ERROR_DETECT}		
Description	Pre-processor switch for enabling development error detection support.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	CNmImmediateRestartEnabled {CNM_IMMEDIATE_RESTART_ENABLED}		
Description	Pre-processor switch for enabling the asynchronous transmission of a NM-message upon bus-communication request in Prepare-Bus-Sleep mode. calculationFormula = If{CNmPassiveModeEnabled == True} then Equal(False) else Equal(True or False)		
Multiplicity	1		
Type	DerivedBooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	CNmNodeDetectionEnabled {CNM_NODE_DETECTION_ENABLED}		
Description	Pre-processor switch for enabling the node detection support. calculationFormula = If (CNmNodeIdEnabled == True) then Equal(NmNodeDetectionEnabled) else Equal(False)		
Multiplicity	1		
Type	DerivedBooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	CNmNodeIdEnabled {CNM_NODE_ID_ENABLED}		
Description	Pre-processor switch for enabling the source node identifier.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	CNmNumberOfChannels {CNM_NUMBER_OF_CHANNELS}		
Description	Number of NM channels allowed within one ECU.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 255		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	CNmPassiveModeEnabled {CNM_PASSIVE_MODE_ENABLED}		
Description	Pre-processor switch for enabling support of the Passive Mode. calculationFormula = Equal(NmPassiveModeEnabled)		
Multiplicity	1		
Type	DerivedBooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	CNmPduRxIndicationEnabled {CNM_PDU_RX_INDICATION_ENABLED}		
Description	Pre-processor switch for enabling the PDU Rx Indication. calculationFormula = Equal(NmPduRxIndicationEnabled)		
Multiplicity	1		
Type	DerivedBooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	CNmRemoteSleepIndEnabled {CNM_REMOTE_SLEEP_IND_ENABLED}		
Description	Pre-processor switch for enabling remote sleep indication support. This feature is required for gateway nodes only. calculationFormula = If (CNmPassiveModeEnabled == False) then Equal(NmRemoteSleepIndEnabled) else Equal(False)		
Multiplicity	1		
Type	DerivedBooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	CNmStateChangeIndEnabled {CNM_STATE_CHANGE_IND_ENABLED}		
Description	Pre-processor switch for enabling the CAN Generic NM state change notification. calculationFormula = Equal(NmStateChangeIdEnabled)		
Multiplicity	1		
Type	DerivedBooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	CNmUserDataEnabled {CNM_USER_DATA_ENABLED}		
Description	Pre-processor switch for enabling user data support. calculationFormula = Equal(NmUserDataEnabled)		
Multiplicity	1		
Type	DerivedBooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.1.0	CNmVersionInfoApi {CNM_VERSION_INFO_API}		
Description	Pre-processor switch for enabling version info API support.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
CNmChannelConfig	1..*	

4.8.2.2 Can Generic Network Management channel configuration

Channel configuration parameters for the Can Generic Network Management are described in figure 4.96. Some of the parameters are derived parameters.

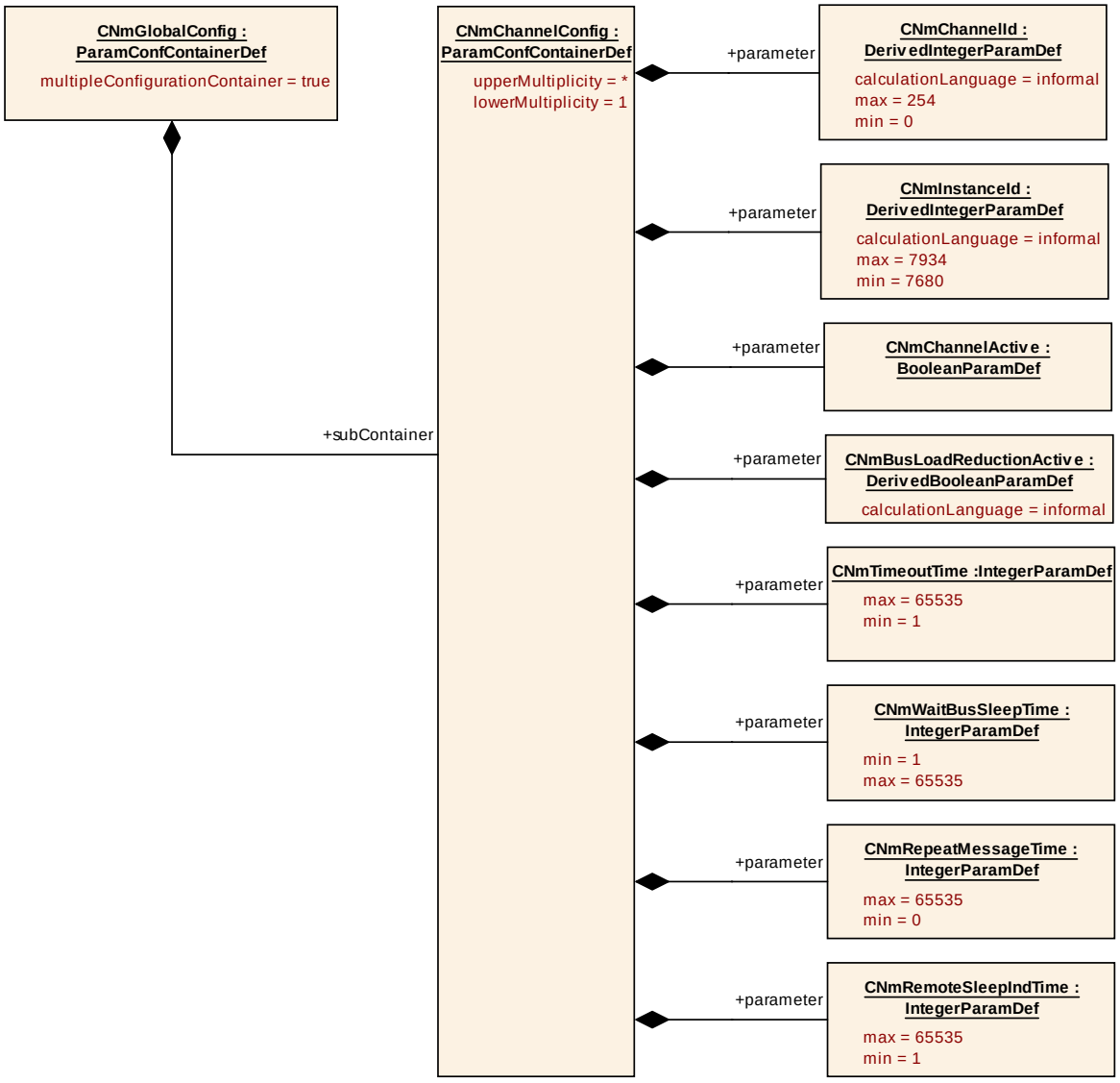


Figure 4.96: Can Generic Nm channel parameters

CNmChannelConfig

SWS Item			
Container Name	CNmChannelConfig {CNm_ChannelConfig}		
Description	This container contains the channel specific configuration parameter of the CNm.		
Configuration Parameters			

Name last modified in: V1.1.0	CNmBusLoadReductionActive {CNM_BUS_LOAD_REDUCTION_ACTIVE}		
Description	It determines if bus load reduction for the respective NM channel is active or not. calculationFormula = If(CNmBusLoadReductionEnabled == False) then DontCare else Equal(True or False)		
Multiplicity	1		
Type	DerivedBooleanParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency	scope: instance		

Name last modified in: V1.1.0	CNmChannelActive {CNM_CHANNEL_ACTIVE}		
Description	It determines if the respective NM channel is active or not. Indicates whether a particular NM-channel shall be initialized (TRUE) or not (FALSE)		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency	scope: instance		

Name last modified in: V1.1.0	CNmChannelId {CNM_CHANNEL_ID}		
Description	Channel identifier configured for the respective AUTOSAR NM cluster. It is used by referring to the respective NM channel handle. It must be unique for each AUTOSAR NM cluster within one ECU. calculationFormula = Equal(NmChannelId)		
Multiplicity	1		
Type	DerivedIntegerParamDef		
Range	0 .. 254		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency	scope: instance dependency: This parameter shall be derived from NmChannelId		

Name last modified in: V1.1.0	CNmInstanceId {CNM_INSTANCE_ID}		
Description	Instance identifier configured for the respective AUTOSAR NM cluster. It is defined as follows: most significant byte = NM_MODULE_ID, least significant byte = NM_CHANNEL_ID It is used by reporting of development errors to DET. It must be unique for each AUTOSAR NM cluster within one ECU. calculationFormula = Equal(NmInstanceId)		
Multiplicity	1		
Type	DerivedIntegerParamDef		
Range	7680 .. 7934		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency	scope: instance dependency: This parameter shall be derived from NmInstanceId		

Name last modified in: V1.1.0	CNmRemoteSleepIndTime {CNM_REMOTE_SLEEP_IND_TIME}		
Description	Timeout for Remote Sleep Indication. It defines the time [ms] how long it shall take to recognize that all other nodes are ready to sleep. Typically it should be equal to: n * CanNmMsgCycleTime, where n denotes the number of NM-Messages that are normally sent before Remote Sleep Indication is detected. The value of n decremented by one determines the amount of lost NM-Messages that can be tolerated by the Remote Sleep Indication procedure. The value 0 denotes that no Remote Sleep Indication functionality is configured.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 65535		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency	scope: instance		

Name last modified in: V1.1.0	CNmRepeatMessageTime {CNM_REPEAT_MESSAGE_TIME}		
Description	Timeout for Repeat Message State. It defines the time [ms] how long the NM shall stay in the Repeat Message State. Typically it should be equal to: n * CanNmMsgCycleTime, where n denotes the number of NM-Messages that are normally sent in the Repeat Message State. The value of n decremented by one determines the amount of lost NM-Messages that can be tolerated by the node detection procedure. The value 0 denotes that no Repeat Message State is configured. It means that Repeat Message State is transient what implicates that it is left immediately after entrance and in result no start-up stability is guaranteed and no node detection procedure is possible.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency	scope: instance		

Name last modified in: V1.1.0	CNmTimeoutTime {CNM_TIMEOUT_TIME}		
Description	Network Timeout for NM-Messages. It denotes the time [ms] how long the NM shall stay in the Network Mode before transition into Prepare Bus-Sleep Mode shall take place. It shall be equal for all nodes in the cluster. It shall be greater than CanNmMsgCycleTime. Typically it should be equal to: n * CanNmMsgCycleTime, where n denotes the number of NM-Message cycle times in the Ready Sleep State before transition into the Bus-Sleep Mode is initiated. The value of n decremented by one determines the amount of lost NM-Messages that can be tolerated by the coordination algorithm.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 65535		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency	scope: instance dependency: CNmTimeoutTime = n * CanNmMsgCycleTime		

Name last modified in: V1.1.0	CNmWaitBusSleepTime {CNM_WAIT_BUS_SLEEP_TIME}		
Description	Timeout for bus calm down phase. It denotes the time [ms] how long the NM shall stay in the Prepare Bus-Sleep Mode before transition into Bus-Sleep Mode shall take place. It shall be equal for all nodes in the cluster. It shall be long enough to make all Tx-buffer empty. The value 0 denotes that no Prepare Bus-Sleep State is configured. It means that Prepare Bus-Sleep State is transient what implicates that it is left immediately after entrance and in result if NM-Timeout Timer expires the Bus-Sleep Mode is entered directly.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 65535		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency	scope: instance		
No Included Containers			

4.8.3 Can Network Management

TODO *This chapter is based on a draft version of the SWS. (V1.9.0) Needs to be updated when the SWS is released.*

There are three variants defined for the Can Network Management module:

- Variant 1: All parameters are pre-compile time.
- Variant 2: A mix of pre-compile time and link time parameters.
- Variant 3: A mix of pre-compile time and post-build time parameters.

The module also supports multiple configuration sets on the level of CanNmGlobalConfig.

The structure of the Can Network Management (CanNm) configuration parameters are structured as described in figure 4.97. References to other modules are described in chapter 4.8.3.4.

- CanNmFeatureConfig in section 4.8.3.1
- CanNmGlobalConfig in section 4.8.3.2
- CanNmChannelConfig in section 4.8.3.3

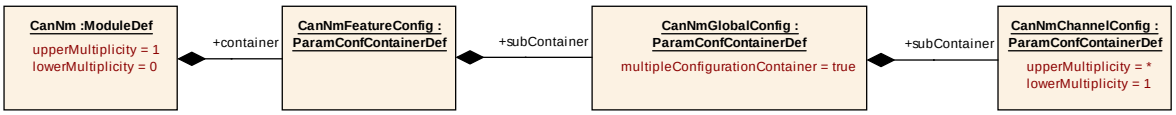


Figure 4.97: Can Network Management Parameters.

4.8.3.1 Can Network Management feature configuration

Feature configuration parameters for the Can Network Management are described in figure 4.98. Some of the parameters are derived parameters.

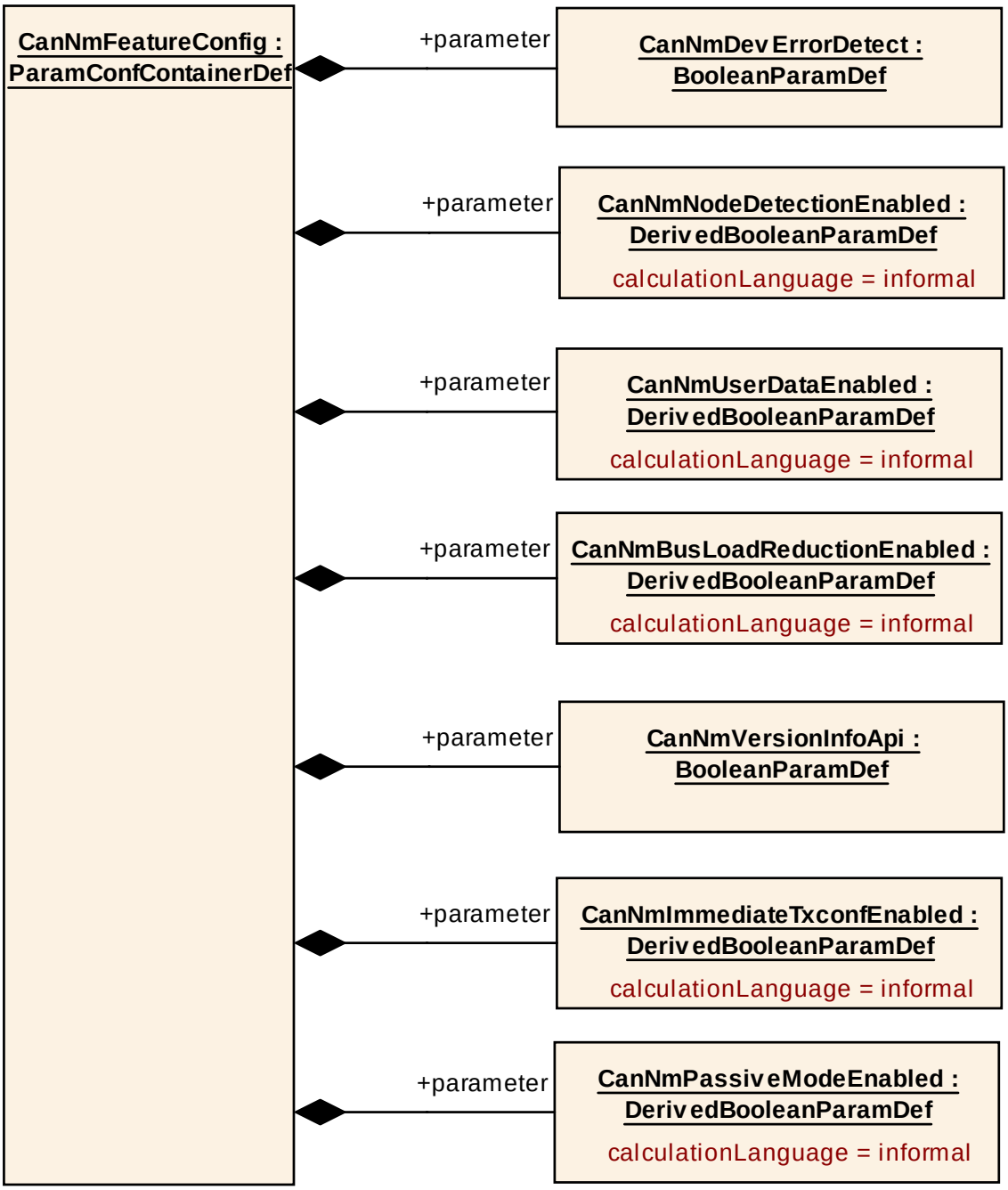


Figure 4.98: Can Nm feature parameters

CanNmFeatureConfig

SWS Item			
Container Name	CanNmFeatureConfig {CanNm_FeatureConfig}		
Description	This container contains parameter which enables or disables optional features of the CanNm.		
Configuration Parameters			
Name last modified in: V1.1.0	CanNmBusLoadReductionEnabled {CANNM_BUS_LOAD_REDUCTION_ENABLED}		
Description	Enable/disable the bus load reduction mechanism. calculationFormula = If(CanNmPassiveModeEnabled == False) then Equal(NmBusLoadReductionEnabled) else Equal(False)		
Multiplicity	1		
Type	DerivedBooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		
Name last modified in: V1.1.0	CanNmDevErrorDetect {CANNM_DEV_ERROR_DETECT}		
Description	Enable/disable the development error detection.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		
Name last modified in: V1.1.0	CanNmImmediateTxconfEnabled {CANNM_IMMEDIATE_TXCONF_ENABLED}		
Description	Enable/disable the immediate tx confirmation, i.e. transmission confirmation to CNm is raised at the transmission request. calculationFormula = If(CanNmPassiveModeEnabled == False) then Equal(True or False) else Equal(False)		
Multiplicity	1		
Type	DerivedBooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		
Name last modified in: V1.1.0	CanNmNodeDetectionEnabled {CANNM_NODE_DETECTION_ENABLED}		
Description	Enable/disable the node detection functionality. calculationFormula = If(CanNmPassiveModeEnabled == False) then Equal(NmNodeDetectionEnabled) else Equal(False)		
Multiplicity	1		
Type	DerivedBooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name last modified in: V1.1.0	CanNmPassiveModeEnabled {CANNM_PASSIVE_MODE_ENABLED}		
Description	Enable/disable the passive mode, i.e. nodes which have this switch enabled must not send NM messages, but they shall receive and process NM messages. calculationFormula = Equal(NmPassiveModeEnabled)		
Multiplicity	1		
Type	DerivedBooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name last modified in: V1.1.0	CanNmUserDataEnabled {CANNM_USER_DATA_ENABLED}		
Description	Enable/disable the user data support. calculationFormula = Equal(NmUserDataEnabled)		
Multiplicity	1		
Type	DerivedBooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name last modified in: V1.1.0	CanNmVersionInfoApi {CANNM_VERSION_INFO_API}		
Description	Enable/disable the function CanNm_GetVersionInfo.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Included Containers

Container Name	Multiplicity	Scope / Depedency
CanNmGlobalConfig	1	

4.8.3.2 Can Network Management global configuration

Global configuration parameters for the Can Network Management are described in figure 4.99.

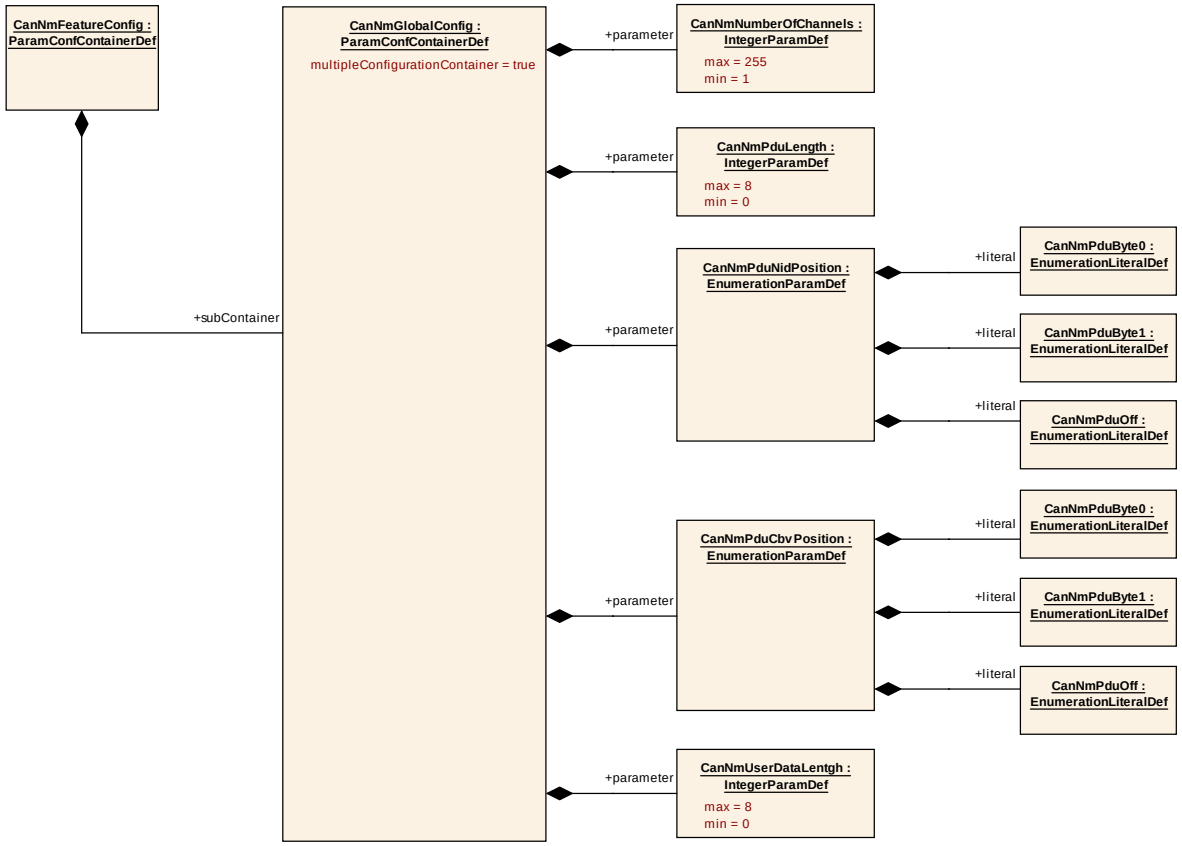


Figure 4.99: Can Nm global parameters

CanNmGlobalConfig

SWS Item			
Container Name	CanNmGlobalConfig {CanNm_GlobalConfig}		
Description	This container contains the global configuration parameter of the CanNm. The parameters and the parameters of the sub containers shall be mapped to the C data type CanNm_ConfigType (for parameters where it is possible) which is passed to the CanNm_Init function. This container is a MultipleConfigurationContainer (only for variant 3), i.e. this container and its sub-containers exit once per configuration set.		
Configuration Parameters			

Name last modified in: V1.1.0	CanNmNumberOfChannels {CANNM_NUMBER_OF_CHANNELS}		
Description	Number of CanNm channels		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 255		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name last modified in: V1.1.0	CanNmPduCbvPosition {CANNM_PDU_CBV_POSITION}		
Description	Defines the position of the control bit vector within the NM PDU. The value of the parameter represents the location of the control bit vector in the NM PDU (CanNmPduByte0 means byte 0, CanNmPduByte1 means byte 1, CanNmPduOff means source node identifier is not part of the NM PDU)		
Multiplicity	1		
Type	EnumerationParamDef		
Range	CanNmPduByte0		
	CanNmPduByte1		
	CanNmPduOff		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency	scope: ECU		

Name last modified in: V1.1.0	CanNmPduLength {CANNM_PDU_LENGTH}		
Description	Defines the length of the NM PDU.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 8		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency	scope: ECU		

Name last modified in: V1.1.0	CanNmPduNidPosition {CANNM_PDU_NID_POSITION}		
Description	Defines the position of the source node identifier within the NM PDU. The value of the parameter represents the location of the source node identifier in the NM PDU (CanNMPduByte0 means byte 0, CanNmPduByte1 means byte 1, CanNmPduOff means source node identifier is not part of the NM PDU)		
Multiplicity	1		
Type	EnumerationParamDef		
Range	CanNmPduByte0		
	CanNmPduByte1		
	CanNmPduOff		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency	scope: ECU		

Name last modified in: V1.1.0	CanNmUserDataLentgh {CANNM_USER_DATA_LENGTH}		
Description	The difference between CanNmPduLength and applied standardized bytes (source node identifier and control bit vector) within the NM PDU.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 8		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency	scope: ECU		

Included Containers

Container Name	Multiplicity	Scope / Depedency
CanNmChannelConfig	1..*	

4.8.3.3 Can Network Management channel configuration

Channel configuration parameters for the Can Network Management are described in figure 4.100.

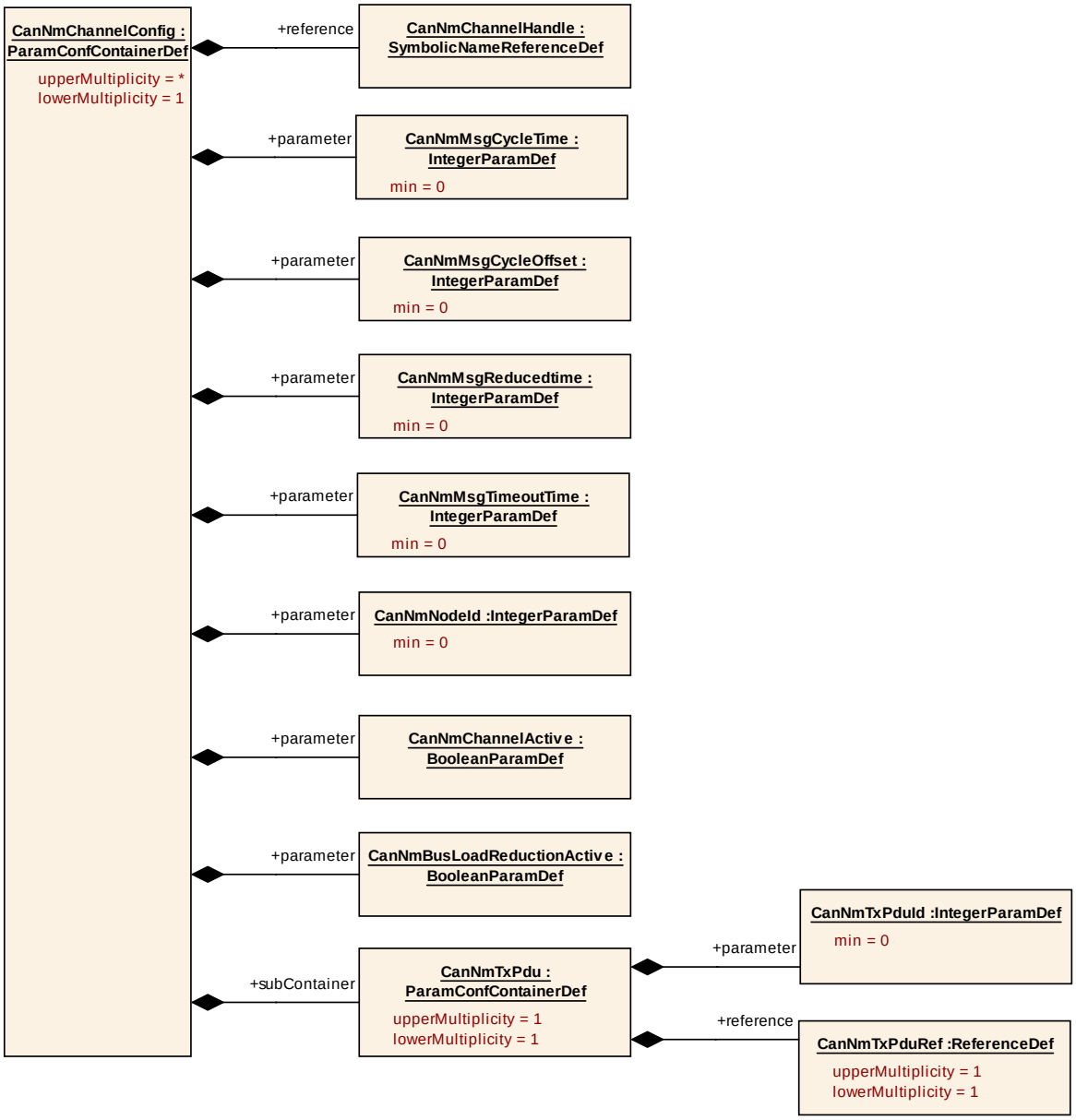


Figure 4.100: Can Nm channel parameters

CanNmChannelConfig

SWS Item			
Container Name	CanNmChannelConfig {CanNm_ChannelConfig}		
Description	This container contains the channel specific configuration parameter of the CanNm.		
Configuration Parameters			

Name last modified in: V1.1.0	CanNmBusLoadReductionActive {CANNM_BUS_LOAD_REDUCTION_ACTIVE}		
Description	Indicates whether the bus load reduction mechanism shall be enabled (True) or disabled (False) for a particular NM-channel. This parameter is only valid if CanNmBusLoadReductionEnabled = True and CanNmPassiveModeEnabled = False Otherwise it is not used.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency	scope: instance		

Name last modified in: V1.1.0	CanNmChannelActive {CANNM_CHANNEL_ACTIVE}		
Description	Indicates whether a particular NM-channel shall be initialized (TRUE) or not (FALSE).		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1, Variant 3
	Link time	X	Variant 2
	Post-build time	–	
Scope / Dependency	scope: instance		

Name last modified in: V1.1.0	CanNmChannelHandle {CANNM_CHANNEL_HANDLE}		
Description	Channel identifier configured for the respective instance of the NM. The CanNmChannelHandle shall be encoded in the canNmRxPduId parameter which is passed to CanNm_RxIndication() function called by the CanIf. The CanNmChannelHandle shall be encoded in the canNmTxPduId which is passed to CanNm_TxConfirmation() function called by the CanIf.		
Multiplicity	1		
Type	Symbolic name reference to CanIfNetworkConfig		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency	scope: instance		

Name last modified in: V1.1.0	CanNmMsgCycleOffset {CANNM_MSG_CYCLE_OFFSET}		
Description	Time offset in the periodic transmission node. It determines the start delay of the transmission. This parameter is only valid if CanNmPassiveModeEnabled is False. Otherwise this parameter is not used.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency	scope: instance dependency: Condition: Parameter value < CanMsgCycleTime		

Name last modified in: V1.1.0	CanNmMsgCycleTime {CANNM_MSG_CYCLE_TIME}		
Description	Period of a NM-message. It determines the periodic rate in the ”periodic transmission mode with bus load reduction” and is the basis for transmit scheduling in the ”periodic transmission mode without bus load reduction”. This parameter is only valid if CanNmPassiveModeEnabled is False. Otherwise this parameter is not used.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency	scope: instance dependency: Condition: CNmTimeoutTime = n * CanNmMsgCycleTime		

Name last modified in: V1.1.0	CanNmMsgReducedtime {CANNM_MSG_REDUCED_TIME}		
Description	Node specific bus cycle time in the periodic transmission mode with bus load reduction. This parameter is only valid if CanNmBusLoadReductionEnabled == True and CanNmBusLoadReductionActive == True and CanNmPassiveModeEnabled == False Otherwise this parameter is not used.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency	scope: instance dependency: Condition: 0,5 * CanNmMsgCycleTime <= CanNmMsgReducedTime < CanNmMsgCycleTime		

Name last modified in: V1.1.0	CanNmMsgTimeoutTime {CANNM_MSG_TIMEOUT_TIME}		
Description	Transmission Timeout of NM-message. If there is no transmission confirmation by the CAN Interface within this timeout, the CANNM module shall give an error notification. This parameter is only valid if CanNmPassiveModeEnabled is False. Otherwise it is not used.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency	scope: instance dependency: Condition: This parameter shall be a multiple of CanNmMsgCycleTime		

Name last modified in: V1.1.0	CanNmNodeId {CANNM_NODE_ID}		
Description	Node identifier of local node. This parameter is only valid if CanNmPassiveModeEnabled = False and CanNmNodeDetectionEnabled = True Otherwise it is not used.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency	scope: instance		
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
CanNmTxPdu	1		

CanNmTxPdu

SWS Item			
Container Name	CanNmTxPdu		
Description	This container contains the CanNmTxPduId and the CanNmTxPduRef.		
Configuration Parameters			

Name last modified in: V1.1.0	CanNmTxPduId {CANNM_TX_PDU_ID}		
Description	L-PDU handle of the NM PDU to be transmitted by CanIf_Transmit and passed to CanNm_TxConfirmation by the CanIf. This handle specifies the corresponding CAN frame ID and implicitly the CAN driver instance as well as the corresponding CAN controller device. This parameter is only valid if CanNmPassiveModeEnabled = False. Otherwise it is not used.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency	scope: instance		

Name last modified in: V1.1.0	CanNmTxPduRef		
Description	The reference to the common PDU structure.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency **No Included Containers**			

4.8.3.4 Can Network Management references

There are references to other modules. They are described in figure 4.101.

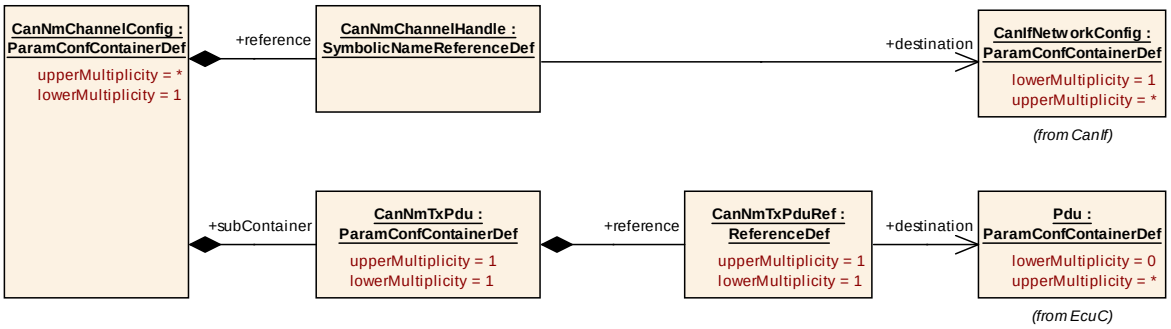


Figure 4.101: Can Nm references to other modules

4.8.4 FlexRay Network Management

TODO *This chapter is based on a draft version of the SWS. (V1.8). Needs to be updated when the SWS is released.*

There are three variants defined for the FlexRay Network Management module:

- Variant 1: All parameters are pre-compile time.
- Variant 2: A mix of pre-compile time and link time parameters.
- Variant 3: A mix of pre-compile time and post-build time parameters.

The structure of the Flexray Network Management (FrNm) configuration parameters are structured as described in figure 4.102. References to other modules are described in chapter 4.8.4.3.

- CFrNmGlobalConfig in section 4.8.4.1
- FrNmChannelConfig in section 4.8.4.2

The module supports post-build selectable configuration on the level of FrNmChannel-Config container.

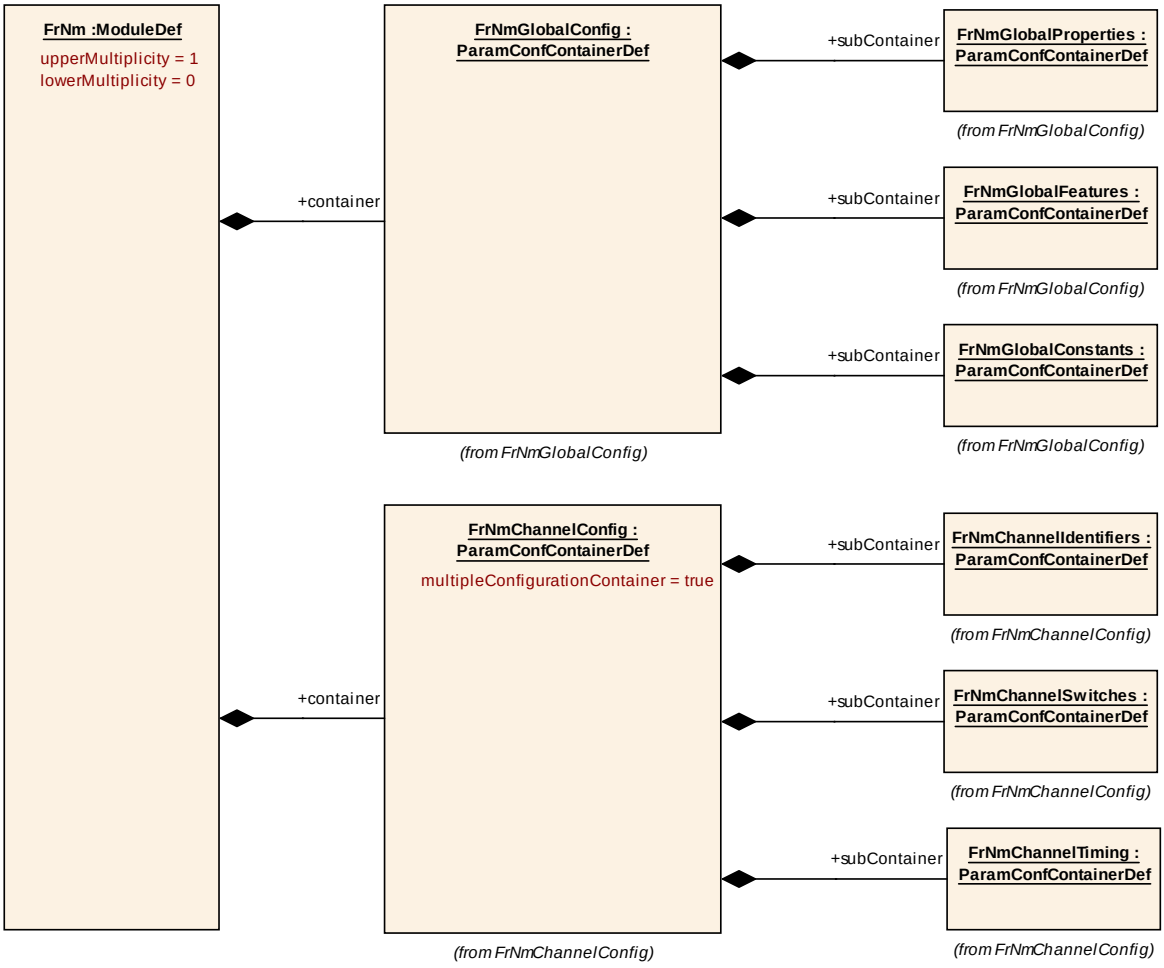


Figure 4.102: FlexRay Network Management overview.

4.8.4.1 Flexray Network Management global configuration

Global configuration parameters for the FlexRay Network Management are described in figure 4.103.

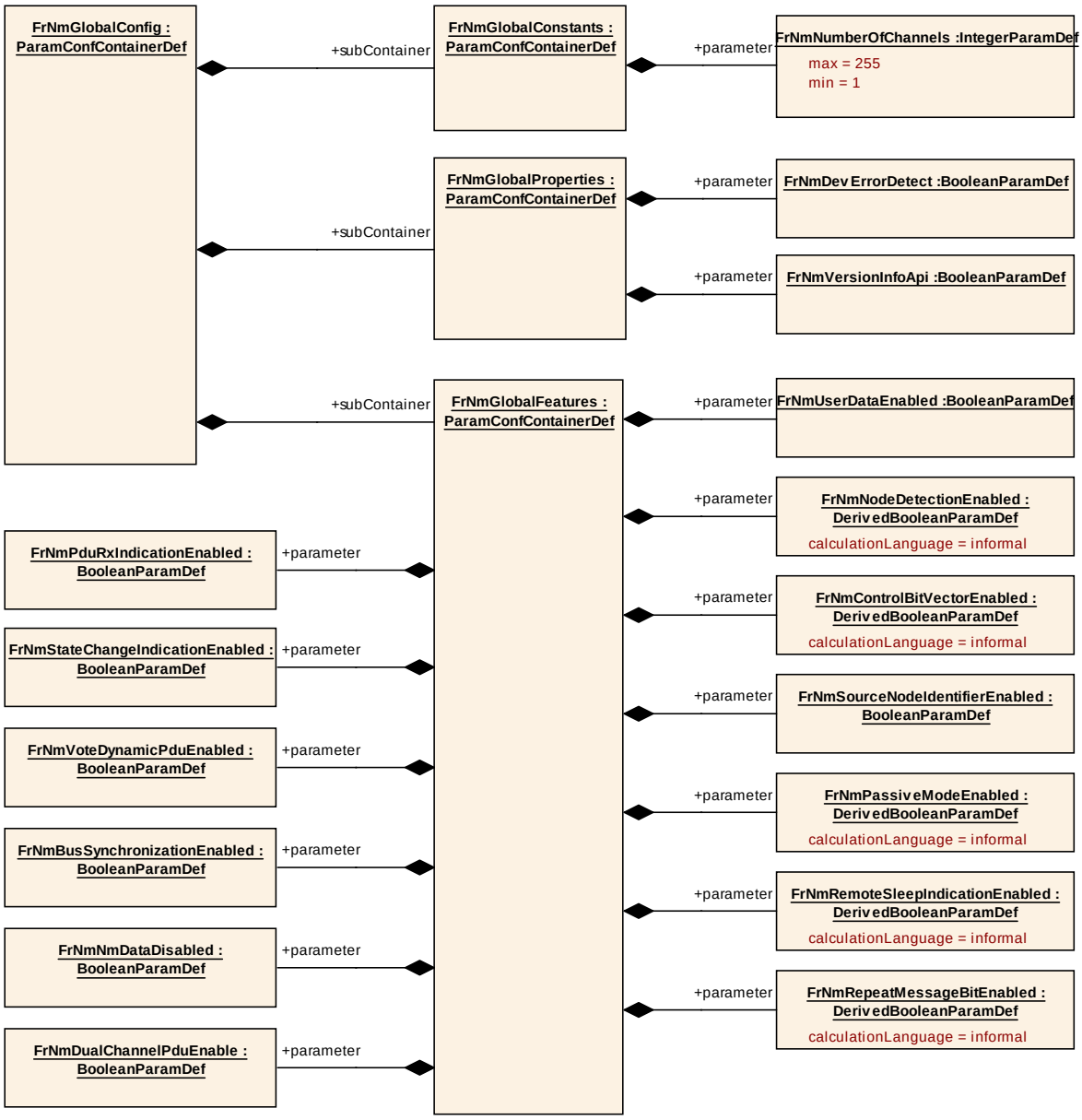


Figure 4.103: FlexRay Network Management global parameters.

FrNmGlobalConstants

SWS Item			
Container Name	FrNmGlobalConstants {FrNm_GlobalConstants}		
Description	This container contains module constants related to the Flexray NM functionality.		
Configuration Parameters			

Name last modified in: V1.1.0	FrNmNumberOfChannels {FRNM_NUMBER_OF_CHANNELS}		
Description	Number of AUTOSAR FR NM channels allowed within one ECU.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 255		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			
No Included Containers			

FrNmGlobalProperties

SWS Item			
Container Name	FrNmGlobalProperties {FrNm_GlobalProperties}		
Description	This container contains module properties related to the FlexRay NM functionality.		
Configuration Parameters			

Name last modified in: V1.1.0	FrNmDevErrorDetect {FRNM_DEV_ERROR_DETECT}		
Description	Pre-processor switch for enabling development error detection		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	FrNmVersionInfoApi {FRNM_VERSION_INFO_API}		
Description	Pre-processor switch for enabling version info API support.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

FrNmGlobalFeatures

SWS Item	
Container Name	FrNmGlobalFeatures {FrNm_GlobalFeatures}
Description	This container contains module features related to the FlexRay NM functionality.
Configuration Parameters	

Name last modified in: V1.1.0	FrNmBusSynchronizationEnabled {FRNM_BUS_SYNCHRONIZATION_ENABLED}		
Description	Pre-processor switch for enabling the bus synchronisation.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	FrNmControlBitVectorEnabled {FRNM_CONTROL_BIT_VECTOR_ENABLED}		
Description	Pre-processor switch for enabling control bit vector support. calculationFormula = If (FrNmNodeDetectionEnabled == False) then Equal(False) else Equal(False or True)		
Multiplicity	1		
Type	DerivedBooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	FrNmDualChannelPduEnable {FRNM_DUAL_CHANNEL_PDU_ENABLE}		
Description	Pre-processor switch for enabling the support of dual channel transmission and reception of NM messages.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	FrNmNmDataDisabled {FRNM_NM_DATA_DISABLED}		
Description	Pre-processor switch for enabling the transmission of NM-Data.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	FrNmNodeDetectionEnabled {FRNM_NODE_DETECTION_ENABLED}		
Description	Pre-processor switch for enabling node detection support. calculationFormula = If (FrNmPassiveModeEnabled == False) then Equal(NmNodeDetectionEnabled) else Equal(False)		
Multiplicity	1		
Type	DerivedBooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	FrNmPassiveModeEnabled {FRNM_PASSIVE_MODE_ENABLED}		
Description	Pre-processor switch for enabling Passive Mode Configuration support. calculationFormula = Equal(NmPassiveModeEnabled)		
Multiplicity	1		
Type	DerivedBooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	FrNmPduRxIndicationEnabled {FRNM_PDU_RX_INDICATION_ENABLED}		
Description	Pre-processor switch for enabling PDU reception indication.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	FrNmRemoteSleepIndicationEnabled {FRNM_REMOTE_SLEEP_INDICATION_ENABLED}		
Description	Pre-processor switch for enabling remote sleep indication. calculationFormula = If (FrNmPassiveModeEnabled == True) then Equal(False) else Equal(False or True)		
Multiplicity	1		
Type	DerivedBooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	FrNmRepeatMessageBitEnabled {FRNM_REPEAT_MESSAGE_BIT_ENABLED}		
Description	Pre-processor switch for enabling the repeat message bit support. calculationFormula = If (FrNmControlBitVectorEnabled == False) then Equal(False) else Equal(False or True)		
Multiplicity	1		
Type	DerivedBooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	FrNmSourceNodeIdentifierEnabled {FRNM_SOURCE_NODE_INDENTIFIER_ENABLED}		
Description	Pre-processor switch for enabling SourceNodeIdentifier support.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	FrNmStateChangeIndicationEnabled {FRNM_STATE_CHANGE_INDICATION_ENABLED}		
Description	Pre-processor switch for enabling state change indication.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	FrNmUserDataEnabled {FRNM_USER_DATA_ENABLED}		
Description	Pre-processor switch for enabling user data support.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	FrNmVoteDynamicPduEnabled {FRNM_VOTE_DYNAMIC_PDU_ENABLED}		
Description	Pre-processor switch for voting with dynamic PDU occurrence.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

4.8.4.2 Flexray Network Management channel configuration

Channel configuration parameters for the FlexRay Network Management are described in figure 4.104 and 4.105.

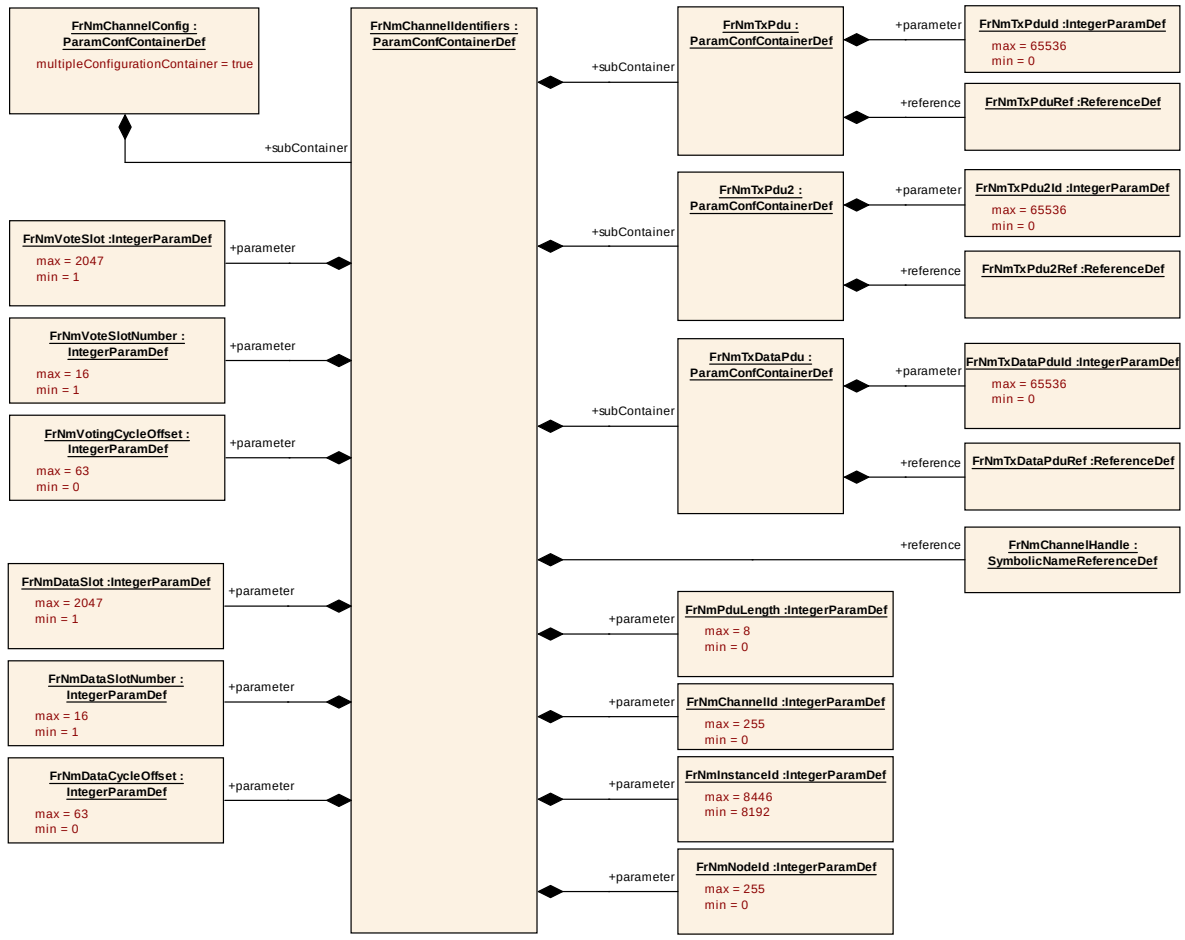


Figure 4.104: FlexRay Network Management channel parameters.

FrNmChannelIdentifiers

SWS Item			
Container Name	FrNmChannelIdentifiers {Channel Identifiers}		
Description	This container contains instance specific identifiers related to the respective FlexRay Channel.		
Configuration Parameters			

Name last modified in: V1.1.0	FrNmChannelHandle {FRNM_CHANNEL_HANDLE}		
Description	Channel identifier configured for the respective instance of the NM. The FrNmChannelHandle shall be encoded in the FrNmRxPduId parameter which is passed to FrNm_RxIndication() function called by the FrIf. The FrNmChannelHandle shall be encoded in the FrNmTxPduId which is passed to FrNm_TxConfirmation() function called by the FrIf.		
Multiplicity	1		
Type	Symbolic name reference to FrIfCluster		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.1.0	FrNmChannelId {FRNM_CHANNEL_ID}		
Description	It is used for referring to the respective NM channel handle. It must be unique for each NM instance within one ECU.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.1.0	FrNmDataCycleOffset {FRNM_DATA_CYCLE_OFFSET}		
Description	Cycle offset within the NM Data Cycle to transmit the NM-Data PDU in the dynamic segment.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 63		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency	dependency: Condition: FrNmDataCycleOffset $<$ FrNmDataCycle		

Name last modified in: V1.1.0	FrNmDataSlot {FRNM_DATA_SLOT}		
Description	Slot of FlexRay Schedule in the dynamic segment used for transmission of the NM-Data.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 2047		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	FrNmDataSlotNumber {FRNM_DATA_SLOT_NUMBER}		
Description	Number of slots of FlexRay Schedule in the dynamic segment used for transmission of the NM-Data.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 16		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	FrNmInstanceId {FRNM_INSTANCE_ID}		
Description	FlexRay NM instance identifier configured for the respective FlexRay Channel. It is used for reporting of development errors to DET. It must be unique for each NM instance within one ECU.		
Multiplicity	1		
Type	IntegerParamDef		
Range	8192 .. 8446		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.1.0	FrNmNodeId {FRNM_NODE_ID}		
Description	NM node identifier configured for the respective FlexRay Channel. It is used for identifying the respective NM node in the NM-cluster. It must be unique for each NM node within one NM cluster.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.1.0	FrNmPduLength {FRNM_PDU_LENGTH}		
Description	Length of the NM-Data PDU.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 8		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	FrNmVoteSlot {FRNM_VOTE_SLOT}		
Description	Slot of FlexRay Schedule in the dynamic segment used for transmission of the NM-Vote.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 2047		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	FrNmVoteSlotNumber {FRNM_VOTE_SLOT_NUMBER}		
Description	Number of slots of FlexRay Schedule in the dynamic segment used for transmission of the NM-Vote.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 16		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	FrNmVotingCycleOffset {FRNM_VOTING_CYCLE_OFFSET}		
Description	Cycle offset within the NM Voting Cycle to transmit the ECUs NM-Vote PDU in the dynamic segment.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 63		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency	dependency: Condition: FrNmVotingCycleOffset < FrNmVotingCycle		
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
FrNmTxDataPdu	1		
FrNmTxPdu	1		
FrNmTxPdu2	1		

FrNmTxPdu

SWS Item			
Container Name	FrNmTxPdu		
Description	Contains the ID and the reference to the TX PDU.		
Configuration Parameters			

Name last modified in: V1.1.0	FrNmTxPduId {FRNM_TX_PDU_ID}		
Description	L-PDU identifier configured for the respective FlexRay Channel. It is used for referring to the FlexRay Interface transmit function. It must be consistent with the value configured in the FlexRay Interface. This ID is used for the combined transmission of NM Vote and NM Data or for the transmission of the NM Vote if NM Data is sent in a separate PDU:		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65536		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.1.0	FrNmTxPduRef		
Description	The reference to a PDU in the global PDU structure.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			
No Included Containers			

FrNmTxPdu2

SWS Item			
Container Name	FrNmTxPdu2		
Description	Contains the ID and the reference to the optional 2nd TX PDU.		
Configuration Parameters			

Name last modified in: V1.1.0	FrNmTxPdu2Id {FRNM_TX_PDU2_ID}		
Description	2nd L-PDU identifier configured for the repsective FlexRay Channel. It is used for the optional second PDU referring to the Flexray Interface Transmit function and must be consistent wuth the value configured in the FlexRay interface. Available only if FrNmDualChannelPduEnable is set to ON.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65536		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.1.0	FrNmTxPdu2Ref		
Description	The reference to a PDU in the global PDU structure.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			
No Included Containers			

FrNmTxDataPdu

SWS Item	
Container Name	FrNmTxDataPdu
Description	Contains the ID and the reference to the Data PDU.
Configuration Parameters	

Name last modified in: V1.1.0	FrNmTxDataPduId {FRNM_TX_DATA_PDU_ID}		
Description	L-PDU identifier configured for the respective FlexRay Channel. It is used if NM data is sent in a separate PDU, for referring to the FlexRay Interface transmit function for the trnasmission of NM data messages and must be consistent with the value configured in the FlexRay interface.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65536		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.1.0	FrNmTxDataPduRef		
Description	The reference to a PDU in the global PDU structure.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency **No Included Containers**			

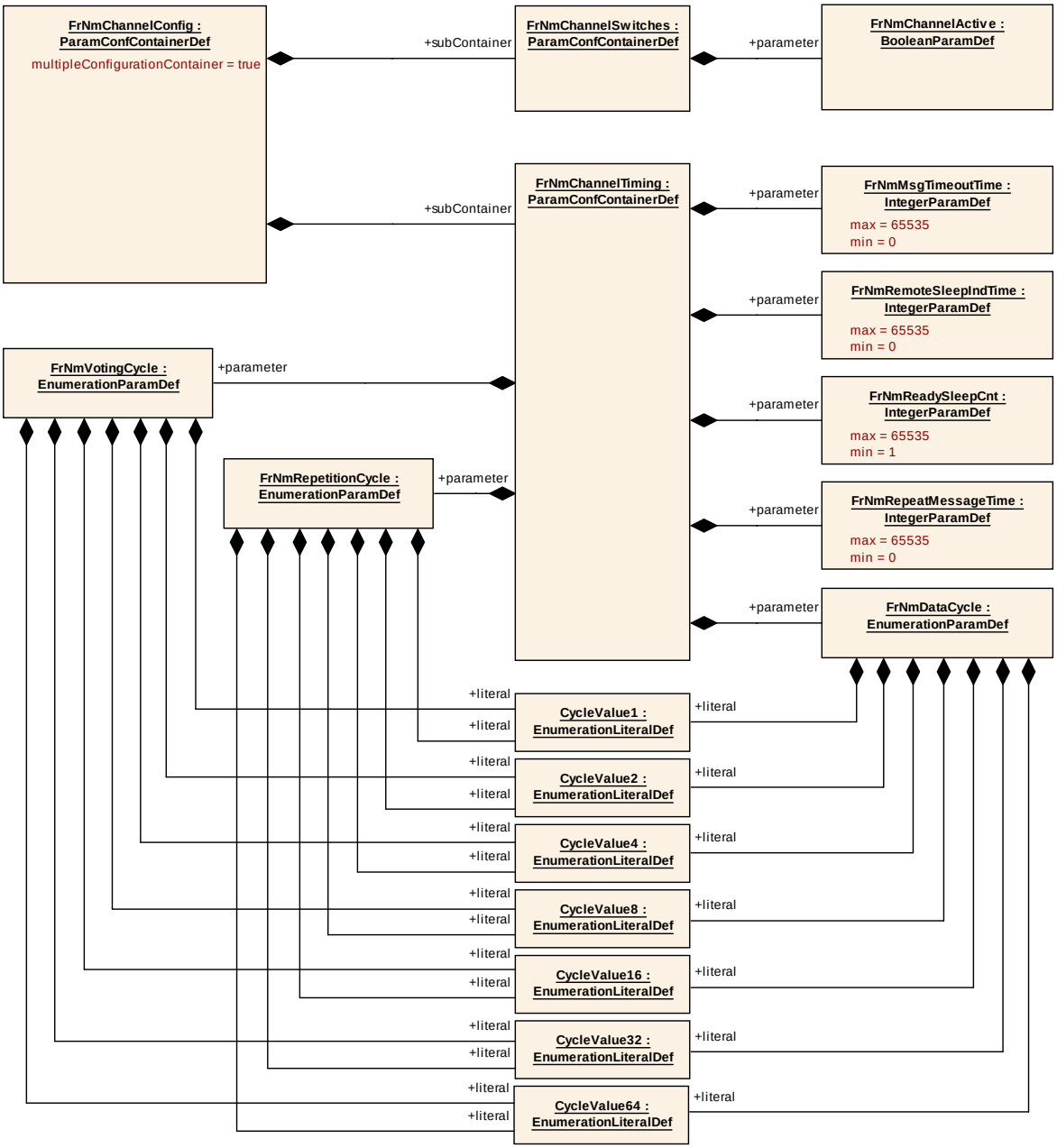


Figure 4.105: FlexRay Network Management channel parameters.

FrNmChannelSwitches

SWS Item			
Container Name	FrNmChannelSwitches {Channel Switches}		
Description	This container contains instance-specific switches related to the respective FlexRay NM PDU.		
Configuration Parameters			
Name last modified in: V1.1.0	FrNmChannelActive {FRNM_CHANNEL_ACTIVE}		
Description	It determines if the respective NM channel is active or not.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			
No Included Containers			

FrNmChannelTiming

SWS Item			
Container Name	FrNmChannelTiming {Channel Timing}		
Description	This container contains instance-specific timing related to the respective FlexRay Channel.		
Configuration Parameters			
Name last modified in: V1.1.0	FrNmDataCycle {FRNM_DATA_CYCLE}		
Description	Number of FlexRay Schedule Cycles needed to transmit the NM Data of all ECUs on the FlexRay bus		
Multiplicity	1		
Type	EnumerationParamDef		
Range	CycleValue1		
	CycleValue16		
	CycleValue2		
	CycleValue32		
	CycleValue4		
	CycleValue64		
	CycleValue8		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	FrNmMsgTimeoutTime {FRNM_MSG_TIMEOUT_TIME}		
Description	Timeout of a NM-message [number of communication cycles]. It determines how long the NM shall wait with notification of transmission failure while communication errors occur on the bus.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2
	Post-build time	X	Variant 3
Scope / Dependency			

Name last modified in: V1.1.0	FrNmReadySleepCnt {FRNM_READY_SLEEP_CNT}		
Description	Numbers of repetitions in the ready sleep state before NM switches to bus sleep mode. On a value of ”1”, the NM-State Machine will leave the Ready Sleep State after one NM Repetition Cycle with no ”keep awake” votes.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 65535		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency	dependency: Condition: FrNmReadySleepCnt >= 1		

Name last modified in: V1.1.0	FrNmRemoteSleepIndTime {FRNM_REMOTE_SLEEP_IND_TIME}		
Description	Timeout for Remote Sleep Indication. It defines the time [number of communication cycles] how long it shall take to recognize that all other nodes are ready to sleep. The value ”0” denotes that no Remote Sleep Indication functionality is configured.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency	dependency: Condition: FrNmRemoteSleepIndTime >= FrNmRepetitionCycle		

Name last modified in: V1.1.0	FrNmRepeatMessageTime {FRNM_REPEAT_MESSAGE_TIME}		
Description	Timeout for Repeat Message State. Defines the time [ms] how long the NM shall stay in the Repeat Message State. The value ”0” denotes that no Repeat Message State is configured, which means that Repeat Message State is transient and implies that it is left immediately after entry and consequently no startup stability is guaranteed and no node detection procedure is possible.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	FrNmRepetitionCycle {FRNM_REPETITION_CYCLE}		
Description	Number of Flexray Schedule Cycles used to repeat the transmission of the Nm vote of all ECUs on the Flexray Bus.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	CycleValue1		
	CycleValue16		
	CycleValue2		
	CycleValue32		
	CycleValue4		
	CycleValue64		
	CycleValue8		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency	dependency: Condition: FrNmRepetitionCycle = n * FrNmVotingCycle; n = [1,2,4,8,16,32,64]		

Name last modified in: V1.1.0	FrNmVotingCycle {FRNM_VOTING_CYCLE}		
Description	Number of FlexRay Schedule Cycles needed to transmit the Nm vote of all ECUs on the FlexRay Bus.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	CycleValue1		
	CycleValue16		
	CycleValue2		
	CycleValue32		
	CycleValue4		
	CycleValue64		
	CycleValue8		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	X	Variant 2, Variant 3
	Post-build time	–	
Scope / Dependency			
No Included Containers			

4.8.4.3 Flexray Network Management references

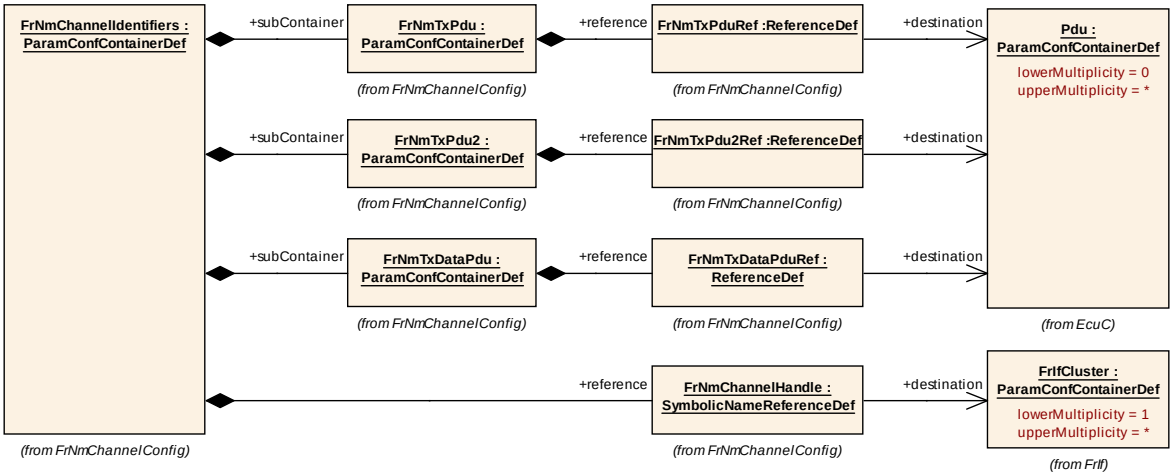


Figure 4.106: FlexRay Network Management references

5 System Services

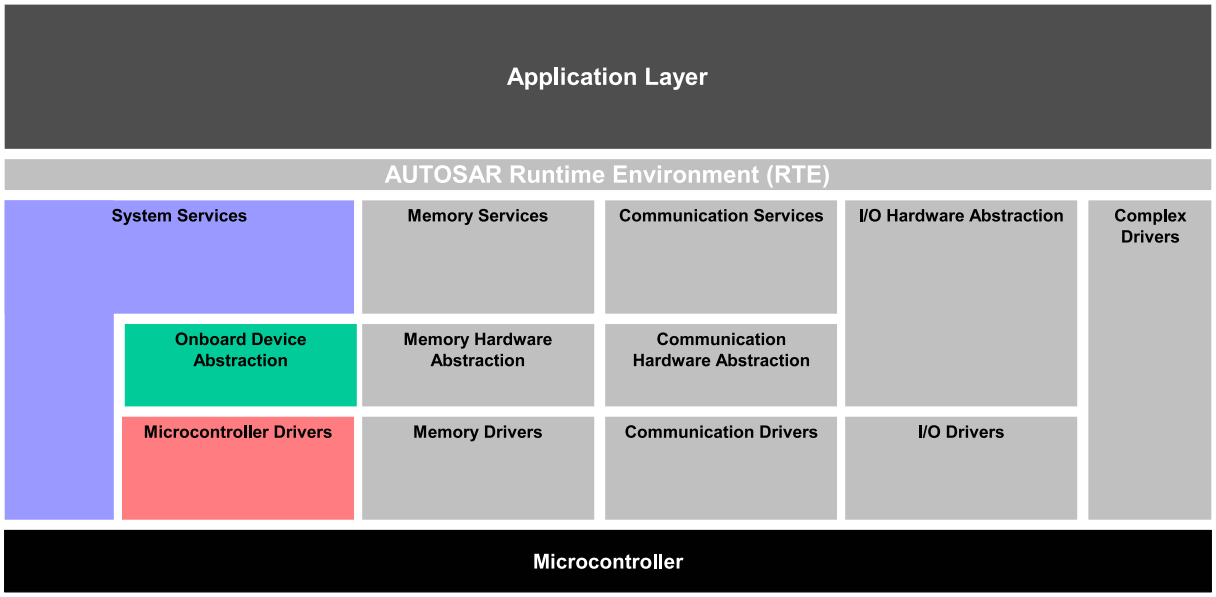


Figure 5.1: System Services of the Basic Software Architecture

This section describes all AUTOSAR Basic Software Modules that belong to the system service cluster.

5.1 Operating System

This section is dealing with the configuration of the AUTOSAR OS that is specified in [30].

The configuration of the OS is comprised of the main configuration objects, namely the `OsOS`, `OsApplication`, `OsTask`, `OsIsr`, `OsScheduleTable`, `OsAlarm`, `OsCounter`, `OsResource`, `OsEvent`, `OsMessage` and `OsAppMode`. Each configured ECU must have exactly one `OsOS`.

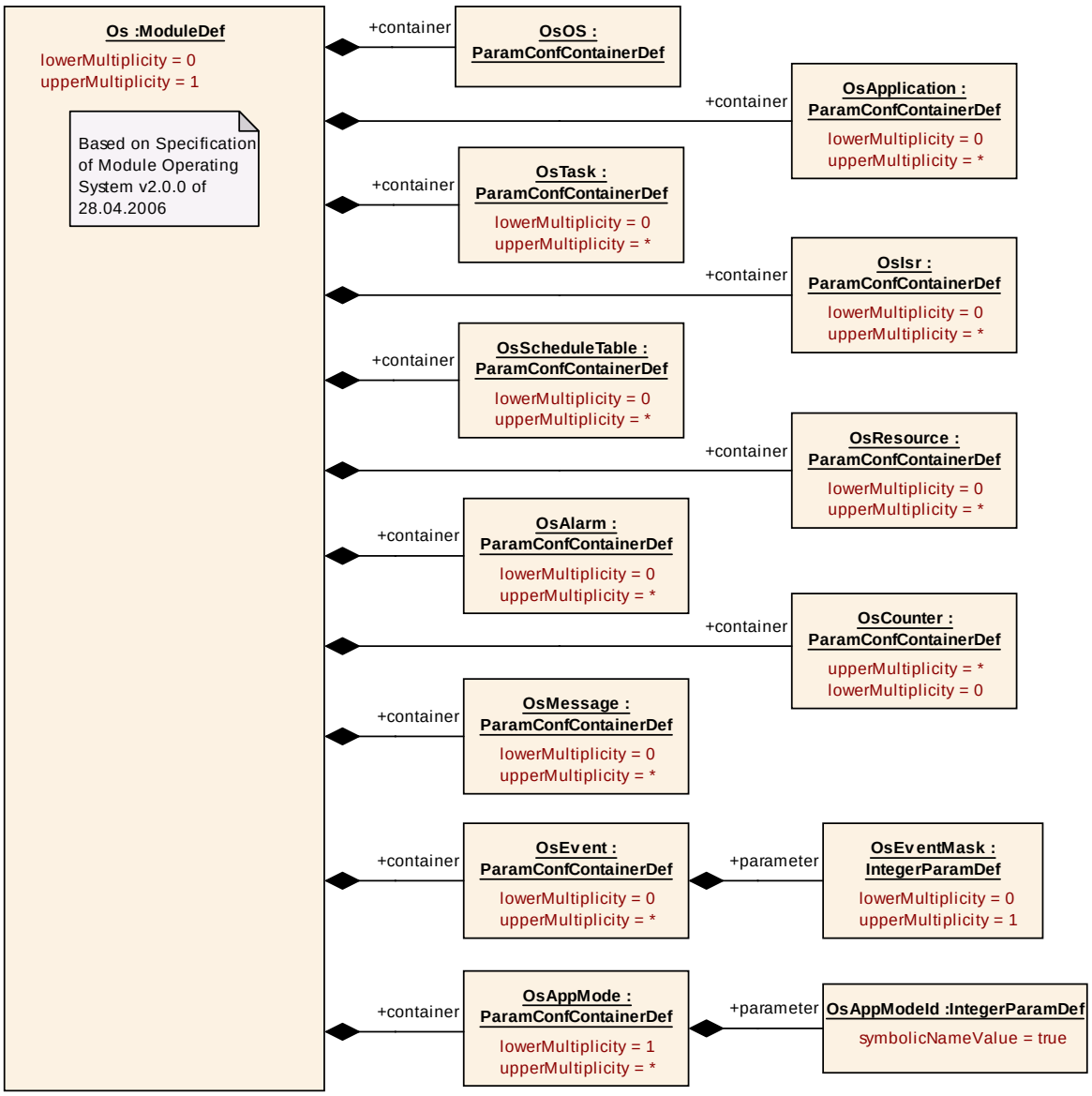


Figure 5.2: OS overview

Module Name	Os	
Module Description	Configuration of the Os (Operating System) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
OsAlarm	0..*	An OsAlarm may be used to asynchronously inform or activate a specific task. It is possible to start alarms automatically at system start-up depending on the application mode.
OsAppMode	1..*	OsAppMode is the object used to define OSEK OS properties for an OSEK OS application mode. No standard attributes are defined for AppMode. In a CPU, at least one AppMode object has to be defined. [source: OSEK OIL Spec. 2.5]
OsApplication	0..*	An AUTOSAR OS must be capable of supporting a collection of OS objects (tasks, interrupts, alarms, hooks etc.) that form a cohesive functional unit. This collection of objects is termed an OS-Application. All objects which belong to the same OS-Application have access to each other. Access means to allow to use these objects within API services. Access by other applications can be granted separately.
OsCom	1	
OsCounter	0..*	Configuration information for the counters that belong to the OsApplication.
OsEvent	0..*	Representation of OS events in the configuration context. Adopted from the OSEK OIL specification.
OsIsr	0..*	The OsISR container represents an OSEK interrupt service routine.
OsMessage	0..*	An OsMessage container represents an OSEK message. The OSEK network messages are not handled in the AUTOSAR environment.
OsNm	1	[pending]
OsOS	1	OS is the object used to define OSEK OS properties for an OSEK application. Per CPU exactly one OS object has to be defined.
OsResource	0..*	An OsResource object is used to co-ordinate the concurrent access by tasks and ISRs to a shared resource, e.g. the scheduler, any program sequence, memory or any hardware area.
OsScheduleTable	0..*	An OsScheduleTable addresses the synchronization issue by providing an encapsulation of a statically defined set of alarms that cannot be modified at runtime.
OsTask	0..*	This container represents an OSEK task.

The following configuration parameters are mainly adopted from the OSEK OIL specification [39].

5.1.1 OS Configuration Object

The parameter container `OsOS` represents the OIL object *OS*.

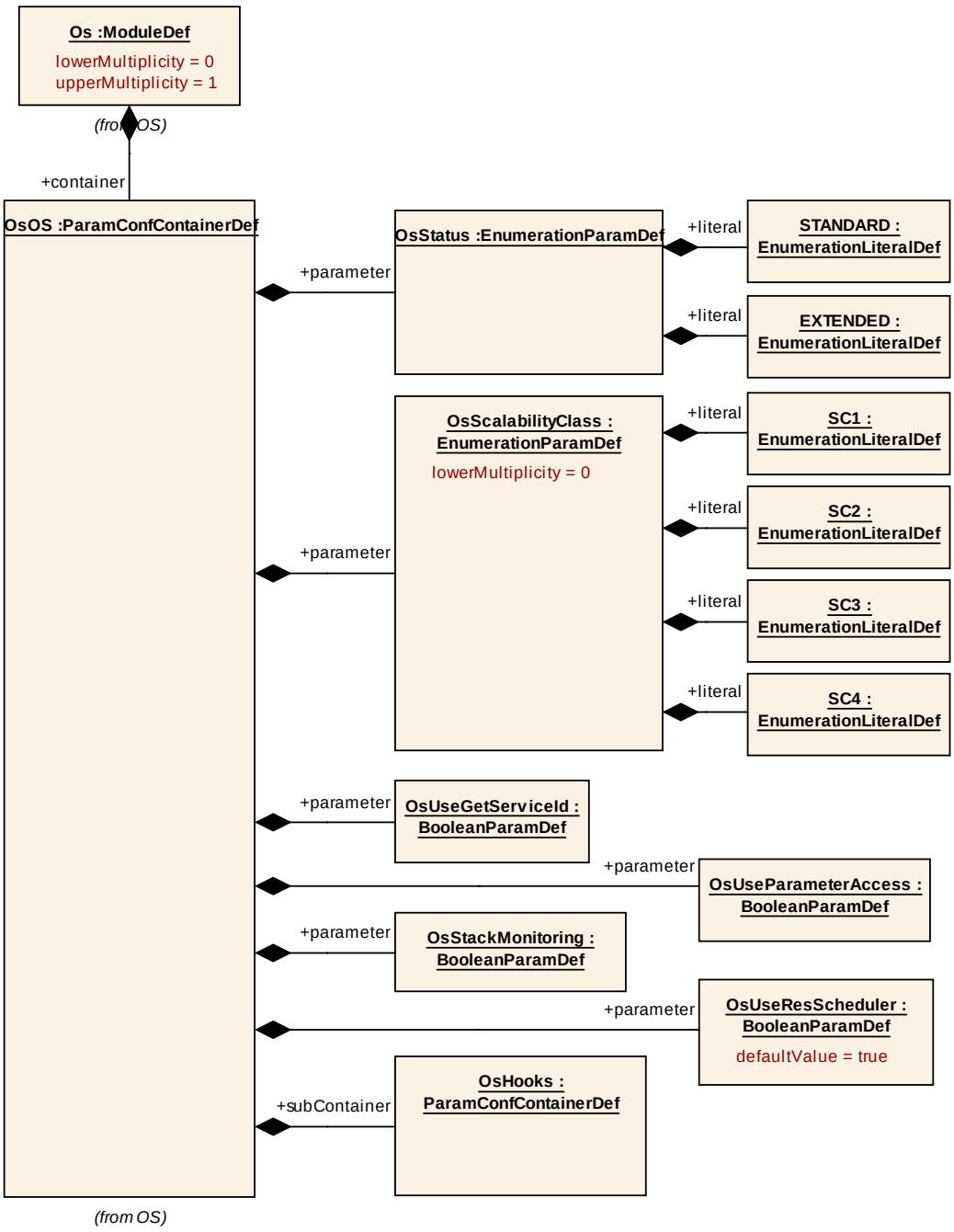


Figure 5.3: `OsOS` details

OsOS

SWS Item	[OS214, OS259, OS307]
Container Name	OsOS {OS}
Description	OS is the object used to define OSEK OS properties for an OSEK application. Per CPU exactly one OS object has to be defined.
Configuration Parameters	

Name last modified in: V1.0.0	OsScalabilityClass {SCALABILITYCLASS}		
Description	A scalability class for each System Object ”OS” has to be selected. In order to customize the operating system to the needs of the user and to take full advantage of the processor features the operating system can be scaled according to the scalability classes. If the scalability class is omitted this translates to the OIL AUTO mechanism.		
Multiplicity	0..1		
Type	EnumerationParamDef		
Range	SC1		
	SC2		
	SC3		
	SC4		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsStackMonitoring {STACKMONITORING}		
Description	Select stack monitoring of Tasks/Category 2 ISRs		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsStatus {STATUS}		
Description	The Status attribute specifies whether a system with standard or extended status has to be used. Automatic assigment is not supported for this attribute.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	EXTENDED		
	STANDARD		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsUseGetServiceId {USEGETSERVICEID}		
Description	As defined by OSEK		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsUseParameterAccess {USEPARAMETERACCESS}		
Description	As defined by OSEK		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsUseResScheduler {USERESSCHEDULER}		
Description	The OsUseResScheduler attribute defines whether the resource RES_SCHEDULER is used within the application.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
OsHooks	1	

5.1.1.1 Hooks of the OsOS

Hooks are defined to represent callback entry-points if certain events occur. Figure 5.4 depicts the structure of `OsOS` hooks.

OsHooks

SWS Item	
Container Name	OsHooks
Description	Container to structure all hooks belonging to the OS
Configuration Parameters	

Name last modified in: V1.0.0	OsErrorHook {ERRORHOOK}		
Description	Error hook as defined by OSEK		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

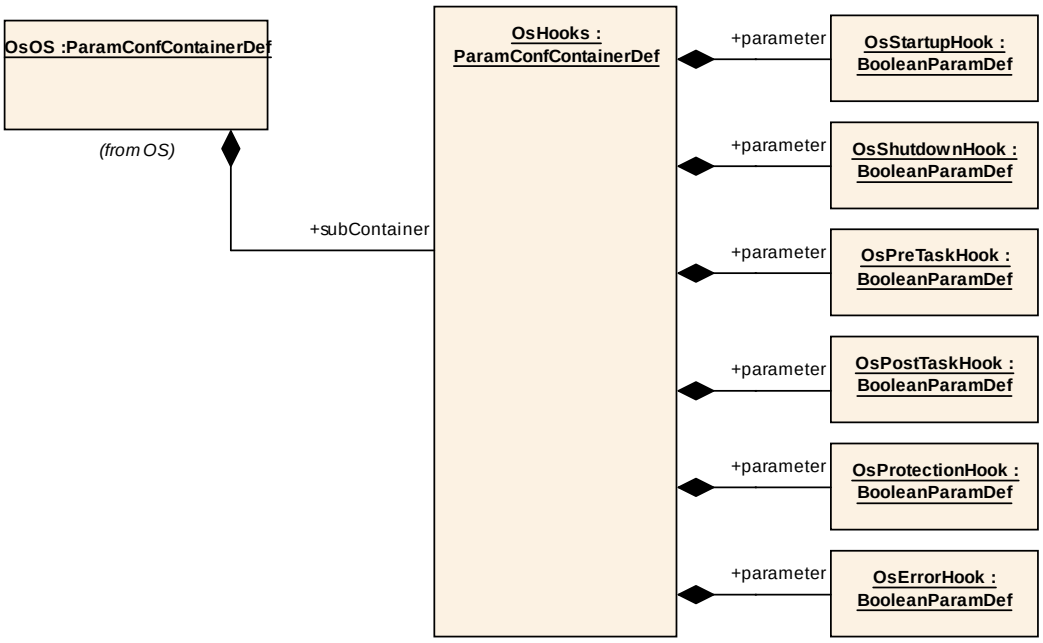


Figure 5.4: OsOSHooks

Name last modified in: V1.0.0	OsPostTaskHook {POSTTASKHOOK}		
Description	Post-task hook as defined by OSEK		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsPreTaskHook {PRETASKHOOK}		
Description	Pre-task hook as defined by OSEK		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsProtectionHook {PROTECTIONHOOK}		
Description	Switch to enable/disable the call to the (user supplied) protection hook		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsShutdownHook {SHUTDOWNHOOK}		
Description	Shutdown hook as defined by OSEK		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsStartupHook {STARTUPHOOK}		
Description	Startup hook as defined by OSEK		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

5.1.2 OS Applications

An overview of the `OsApplication` configuration object is depicted in figure 5.5. `OsApplication` references all relevant configuration attributes that belong to that OS application like tasks, schedule tables etc. There does not exist a complement for `OsApplication` in OSEK OIL.

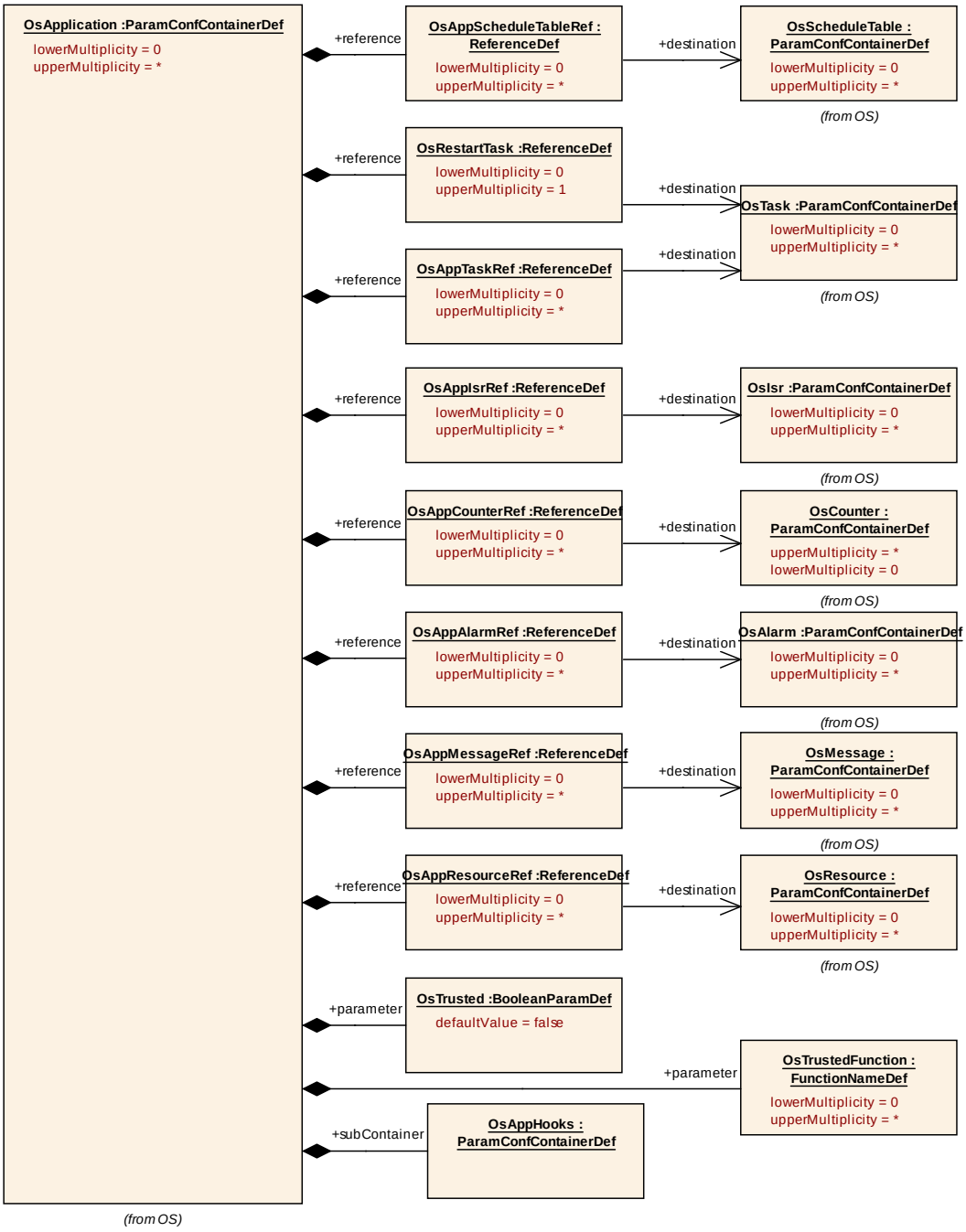


Figure 5.5: `OsApplication` details

OsApplication

SWS Item	[OS114, OS254, OS115, OS124, OS125, OS213, OS120, OS16, OS221, OS231, OS230, OS234, OS248, OS253]
Container Name	OsApplication {APPLICATION}
Description	An AUTOSAR OS must be capable of supporting a collection of OS objects (tasks, interrupts, alarms, hooks etc.) that form a cohesive functional unit. This collection of objects is termed an OS-Application. All objects which belong to the same OS-Application have access to each other. Access means to allow to use these objects within API services. Access by other applications can be granted separately.
Configuration Parameters	

Name last modified in: V1.0.0	OsAppAlarmRef {ACCESSING_APPLICATION}		
Description	Specifies the OsAlarms that belong to the OsApplication.		
Multiplicity	0..*		
Type	Reference to OsAlarm		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	dependency: Required for scalability class 3 and 4		

Name last modified in: V1.0.0	OsAppCounterRef {ACCESSING_APPLICATION}		
Description	References the OsCounters that belong to the OsApplication.		
Multiplicity	0..*		
Type	Reference to OsCounter		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsAppIsrRef		
Description	references which OsIsrs belong to the OsApplication		
Multiplicity	0..*		
Type	Reference to OsIsr		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsAppMessageRef {ACCESSING_APPLICATION}		
Description	References the OsMessages that belong to the OsApplication.		
Multiplicity	0..*		
Type	Reference to OsMessage		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsAppResourceRef {ACCESSING_APPLICATION}		
Description	References the OsResources that belong to the OsApplication.		
Multiplicity	0..*		
Type	Reference to OsResource		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsAppScheduleTableRef {ACCESSING_APPLICATION}		
Description	References the OsScheduleTables that belong to the OsApplication.		
Multiplicity	0..*		
Type	Reference to OsScheduleTable		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsAppTaskRef		
Description	references which OsTasks belong to the OsApplication		
Multiplicity	0..*		
Type	Reference to OsTask		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsRestartTask {RESTARTTASK}		
Description	Optionally one task of an OS-Application may be defined as Re-start Task.		
Multiplicity	0..1		
Type	Reference to OsTask		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsTrusted {TRUSTED}		
Description	true: OS-Application is trusted false: OS-Application is not trusted (default)		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsTrustedFunction {TRUSTED_FUNCTION}		
Description	Trusted function (as part of a trusted OS-Application) available to other OS-Applications. This also supersedes the OSEK OIL attribute TRUSTED in APPLICATION because the optionality of this parameter is describing that already.		
Multiplicity	0..*		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
OsAppHooks	1		

5.1.2.1 Hooks of the OsApplication

`OsAppHooks` represents the callback entry-points for certain events handled by the `OsApplication`.

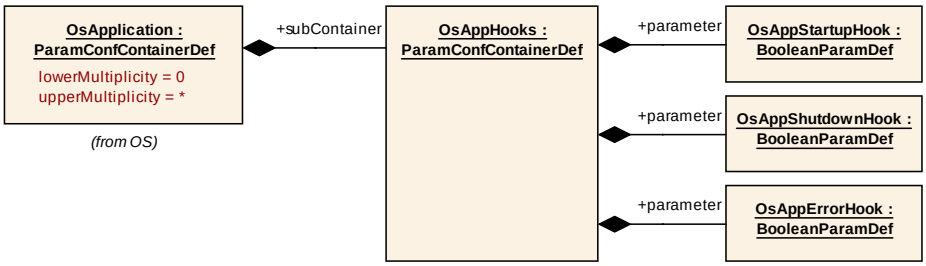


Figure 5.6: `OsAppHooks` details

OsAppHooks

SWS Item			
Container Name	OsAppHooks		
Description	Container to structure the OS-Application-specific hooks		
Configuration Parameters			

Name last modified in: V1.0.0	OsAppErrorHook {ERRORHOOK}		
Description	Select the OS-Application error hook.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsAppShutdownHook {SHUTDOWNHOOK}		
Description	Select the OS-Application specific shutdown hook for the OS-Application.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsAppStartupHook {STARTUPHOOK}		
Description	Select the OS-Application specific startup hook for the OS-Application.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

5.1.2.2 AppMode of the OsApplication

The `OsAppMode` configuration container is specified to allow vendor-specific extension to the `OsOS` configuration container.

OsAppMode

SWS Item			
Container Name	OsAppMode {APPMODE}		
Description	OsAppMode is the object used to define OSEK OS properties for an OSEK OS application mode. No standard attributes are defined for AppMode. In a CPU, at least one AppMode object has to be defined. [source: OSEK OIL Spec. 2.5]		
Configuration Parameters			

Name last modified in: V1.0.0	OsAppModeId		
Description	Internal ID of an AppMode. Necessary, so that an AppMode is addressable by other modules.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

5.1.3 Schedule Tables

The `OsApplication` can be configured to have several `OsScheduleTable`s. They are defined to specify a fine-grained schedule without having to occupy several alarms and counters. Figure 5.7 depicts the configuration structure.

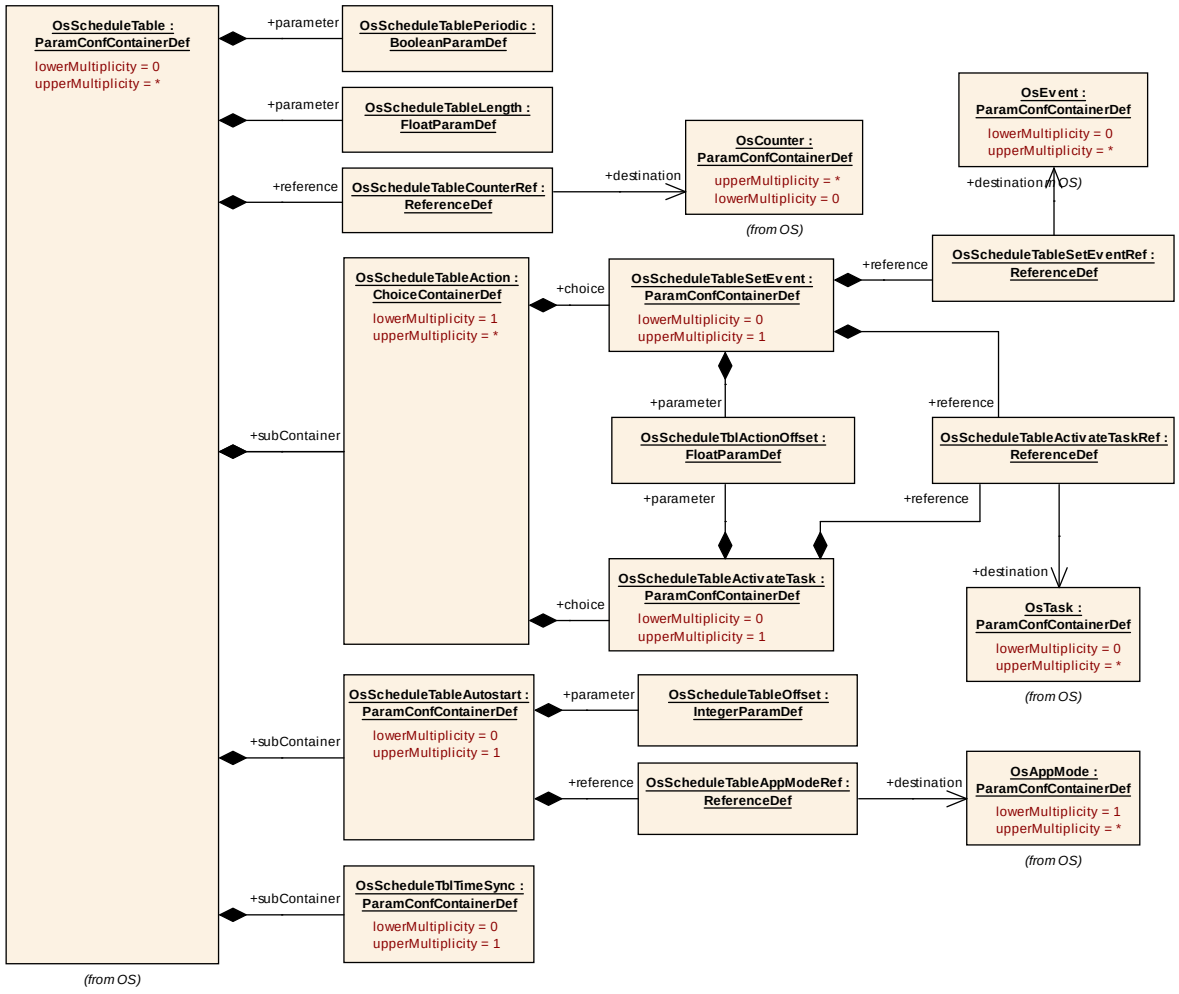


Figure 5.7: `OsScheduleTable` details

OsScheduleTable

SWS Item	[OS141, OS143, OS145, OS144, OS249, OS310, OS335]
Container Name	OsScheduleTable {SCHEDULETABLE}
Description	An OsScheduleTable addresses the synchronization issue by providing an encapsulation of a statically defined set of alarms that cannot be modified at runtime.
Configuration Parameters	

Name last modified in: V1.0.0	OsScheduleTableCounterRef {COUNTER}		
Description	OS145: The Configuration language shall allow the user to define the counter which drives the schedule table.		
Multiplicity	1		
Type	Reference to OsCounter		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsScheduleTableLength {LENGTH}		
Description	Defines the length of the schedule table. The unit is depending on the related counter type.		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsScheduleTablePeriodic {PERIODIC}		
Description	Defines if the schedule table is periodic or not.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Included Containers

Container Name	Multiplicity	Scope / Depedency
OsScheduleTableAction [OsScheduleTable ActivateTask, Os ScheduleTableSetEvent]	1..*	
OsScheduleTable Autostart	0..1	
OsScheduleTblTimeSync	0..1	

5.1.3.1 Schedule Table Actions

An `OsScheduleTable` can trigger several tasks and events namely `OsScheduleTableActivateTask` and `OsScheduleTableSetEvent`. They are described below.

OsScheduleTableActivateTask

SWS Item			
Container Name	OsScheduleTableActivateTask {ACTIVATETASK}		
Description	Task that is triggered by that schedule table.		
Configuration Parameters			

Name last modified in: V1.0.0	OsScheduleTableActivateTaskRef {TASK}		
Description	Reference to task that will be activated by action		
Multiplicity	1		
Type	Reference to OsTask		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsScheduleTblActionOffset {OFFSET}		
Description	[s] relative to the start of the period		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

OsScheduleTableSetEvent

SWS Item			
Container Name	OsScheduleTableSetEvent {SETEVENT}		
Description	Event that is triggered by that schedule table.		
Configuration Parameters			

Name last modified in: V1.0.0	OsScheduleTableActivateTaskRef {TASK}		
Description	Reference to task that will be activated by action		
Multiplicity	1		
Type	Reference to OsTask		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsScheduleTableSetEventRef {EVENT}		
Description	Reference to event that will be set by action		
Multiplicity	1		
Type	Reference to OsEvent		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsScheduleTblActionOffset {OFFSET}		
Description	[s] relative to the start of the period		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency **No Included Containers**			

5.1.3.2 Schedule Table Synchronization

Figure 5.8 depicts time synchronization information for an `OsScheduleTable`.

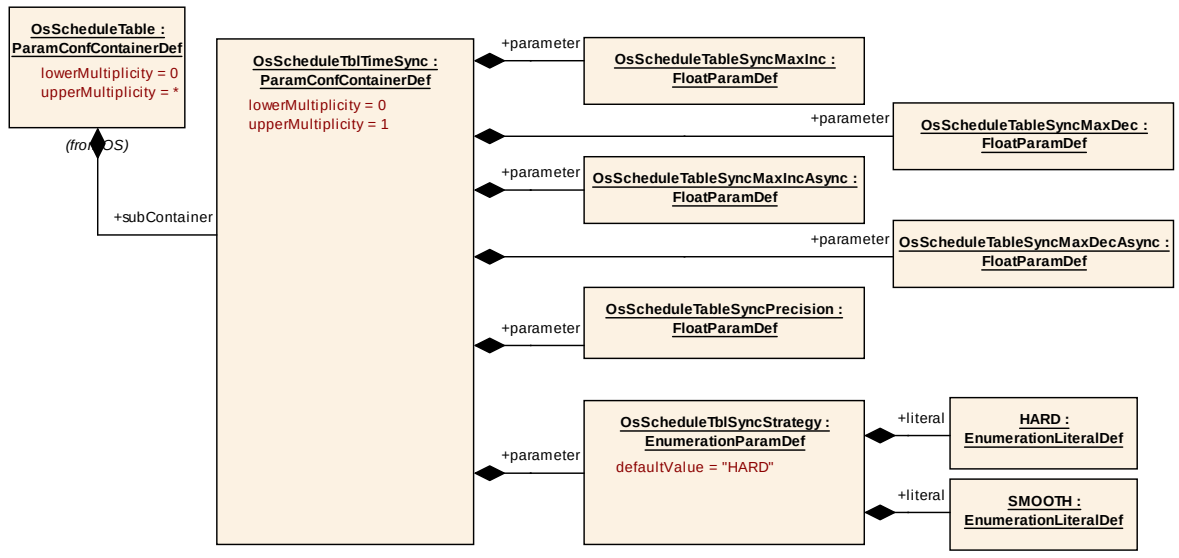


Figure 5.8: `OsScheduleTblTimeSync` details

OsScheduleTblTimeSync

SWS Item			
Container Name	OsScheduleTblTimeSync {LOCAL_TO_GLOBAL_TIME_SYNCHRONIZATION}		
Description	This parameter specifies he synchronization parameters of the schedule table. The sub parameters define the startup behavior and the maximal adjustments. The timing parameters are given in seconds.		
Configuration Parameters			

Name last modified in: V1.0.0	OsScheduleTableSyncMaxDec {MAX_DECREASE}		
Description	specified in [s]		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsScheduleTableSyncMaxDecAsync {MAX_DECREASE_ASYNC}		
Description	specified in [s]		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsScheduleTableSyncMaxInc {MAX_INCREASE}		
Description	specified in [s]		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsScheduleTableSyncMaxIncAsync {MAX_INCREASE_ASYNC}		
Description	specified in [s]		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsScheduleTableSyncPrecision {PRECISION}		
Description	specified in [s]		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	OsScheduleTblSyncStrategy {SYNC_STRATEGY}		
Description	This parameter specifies how OS should synchronize that OsScheduleTable with a global time.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	HARD	If set and a new global time is provided, OS shall adjust the time at the end of the schedule table to the new global time.	
	SMOOTH	If set and a new global time is provided, OS shall adjust each span of time between adjacent expiry points by either increasing or decreasing, whichever is faster.	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency **No Included Containers**			

5.1.4 Tasks

An `OsTask` holds all the configuration information relevant for a task. Figure 5.9 depicts the according structure.

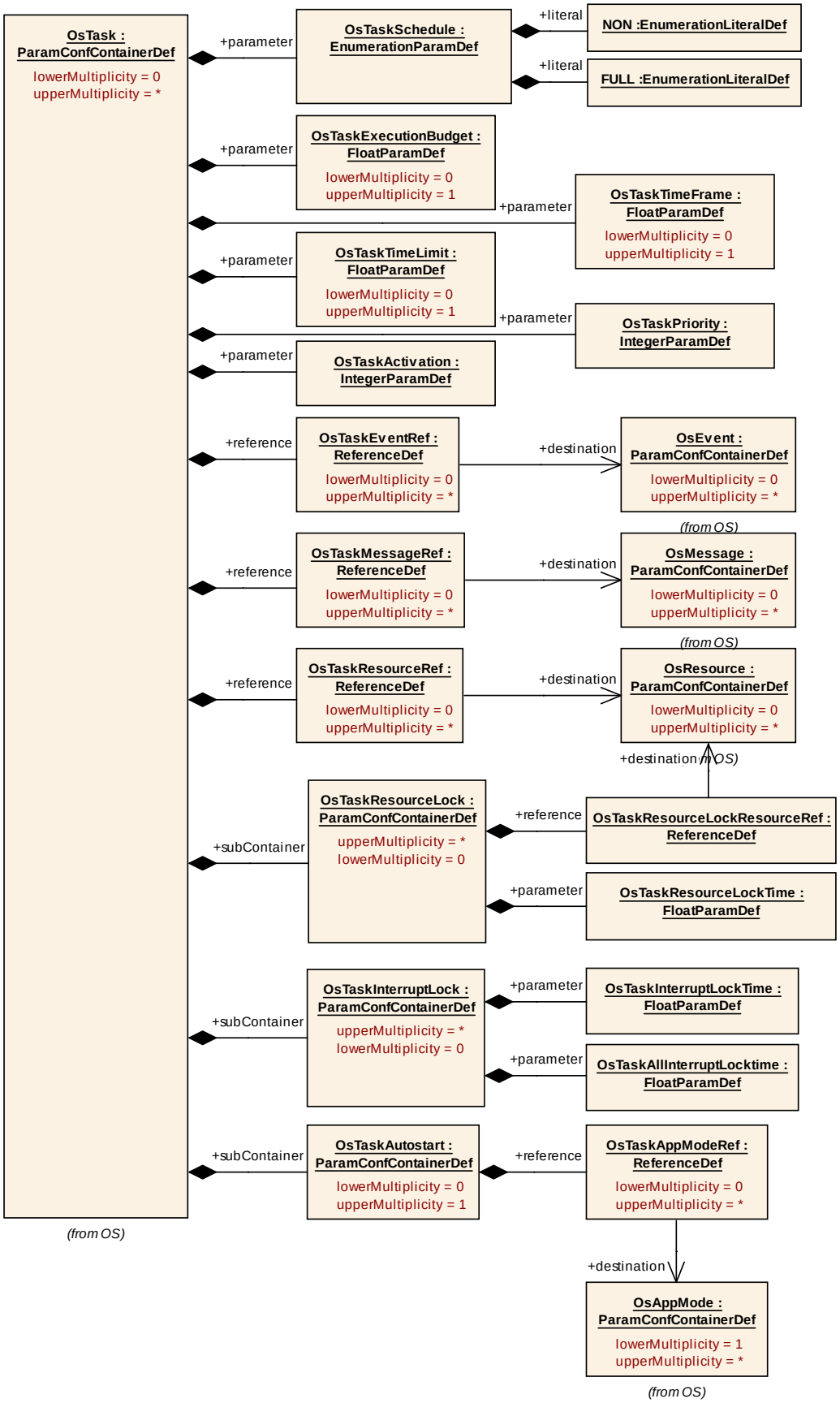


Figure 5.9: `OsTask` details

OsTask

SWS Item	[OS119, OS185, OS188, OS250, OS325]
Container Name	OsTask {TASK}
Description	This container represents an OSEK task.
Configuration Parameters	

Name last modified in: V1.0.0	OsTaskActivation {ACTIVATION}		
Description	This attribute defines the maximum number of queued activation requests for the task. A value equal to ”1” means that at any time only a single activation is permitted for this task.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsTaskEventRef {EVENT}		
Description	This reference defines the list of events the extended task may react on.		
Multiplicity	0..*		
Type	Reference to OsEvent		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsTaskExecutionBudget {EXECUTIONBUDGET}		
Description	This parameter contains the worst case execution time of the task in seconds. Only available in scalability class 2 and 4		
Multiplicity	0..1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsTaskMessageRef {MESSAGE}		
Description	This reference defines the list of messages accessed by this task.		
Multiplicity	0..*		
Type	Reference to OsMessage		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsTaskPriority {PRIORITY}		
Description	The priority of a task is defined by the value of this attribute. This value has to be understood as a relative value, i.e. the values show only the relative ordering of the tasks. OSEK OS defines the lowest priority as zero (0); larger values correspond to higher priorities.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsTaskResourceRef {RESOURCE}		
Description	This reference defines a list of resources accessed by this task.		
Multiplicity	0..*		
Type	Reference to OsResource		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsTaskSchedule {SCHEDULE}		
Description	The OsTaskSchedule attribute defines the preemptability of the task. If this attribute is set to NON, no internal resources may be assigned to this task.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	FULL	Task is preemptable.	
	NON	Task is not preemptable.	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsTaskTimeFrame {TIMEFRAME}		
Description	These two parameters contain the allowed inter-arrival time of the task. The time is specified as a time budget (TIMELIMIT) in a defined time frame (TIMEFRAME). Only available in scalability class 2 and 4		
Multiplicity	0..1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsTaskTimeLimit {TIMELIMIT}		
Description	These two parameters contain the allowed inter-arrival time of the task. The time is specified as a time budget (TIMELIMIT) in a defined time frame (TIMEFRAME). Only available in scalability class 2 and 4		
Multiplicity	0..1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
OsTaskAutostart	0..1		
OsTaskInterruptLock	0..*		
OsTaskResourceLock	0..*		

OsTaskAutostart

SWS Item	
Container Name	OsTaskAutostart {AUTOSTART}
Description	This container determines whether the task is activated during the system start-up procedure or not for some specific application modes. If the task shall be activated during the system start-up, this container is present and holds the references to the application modes in which the task is auto-started.
Configuration Parameters	

Name last modified in: V1.0.0	OsTaskAppModeRef {APPMODE}		
Description	Reference to application modes in which that task is activated on startup of the OS		
Multiplicity	0..*		
Type	Reference to OsAppMode		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

OsTaskInterruptLock

SWS Item			
Container Name	OsTaskInterruptLock {INTERRUPTLOCK}		
Description	This parameter contains the time of the task in which the task has disabled either category 1 and 2 interrupts (ALLINTERRUPTLOCKTIME) or only category 2 (OSINTERRUPTLOCKTIME) interrupts.		
Configuration Parameters			

Name last modified in: V1.0.0	OsTaskAllInterruptLocktime {ALLINTERRUPTLOCKTIME}		
Description	interrupt lock time in [s]		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsTaskInterruptLockTime {OSINTERRUPTLOCKTIME}		
Description	interrupt lock time in [s]		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

OsTaskResourceLock

SWS Item			
Container Name	OsTaskResourceLock {RESOURCELOCK}		
Description	This parameter contains the worst case time between getting and releasing a given resource (in seconds).		
Configuration Parameters			

Name last modified in: V1.0.0	OsTaskResourceLockResourceRef {RESOURCE}		
Description	Reference to the resource the locking time is depending on		
Multiplicity	1		
Type	Reference to OsResource		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsTaskResourceLockTime {RESOURCELOCKTIME}		
Description	resource lock time in [s]		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

5.1.5 Interrupt Service Routines

Figure 5.10 shows the structure of the `OsIsr` configuration container.

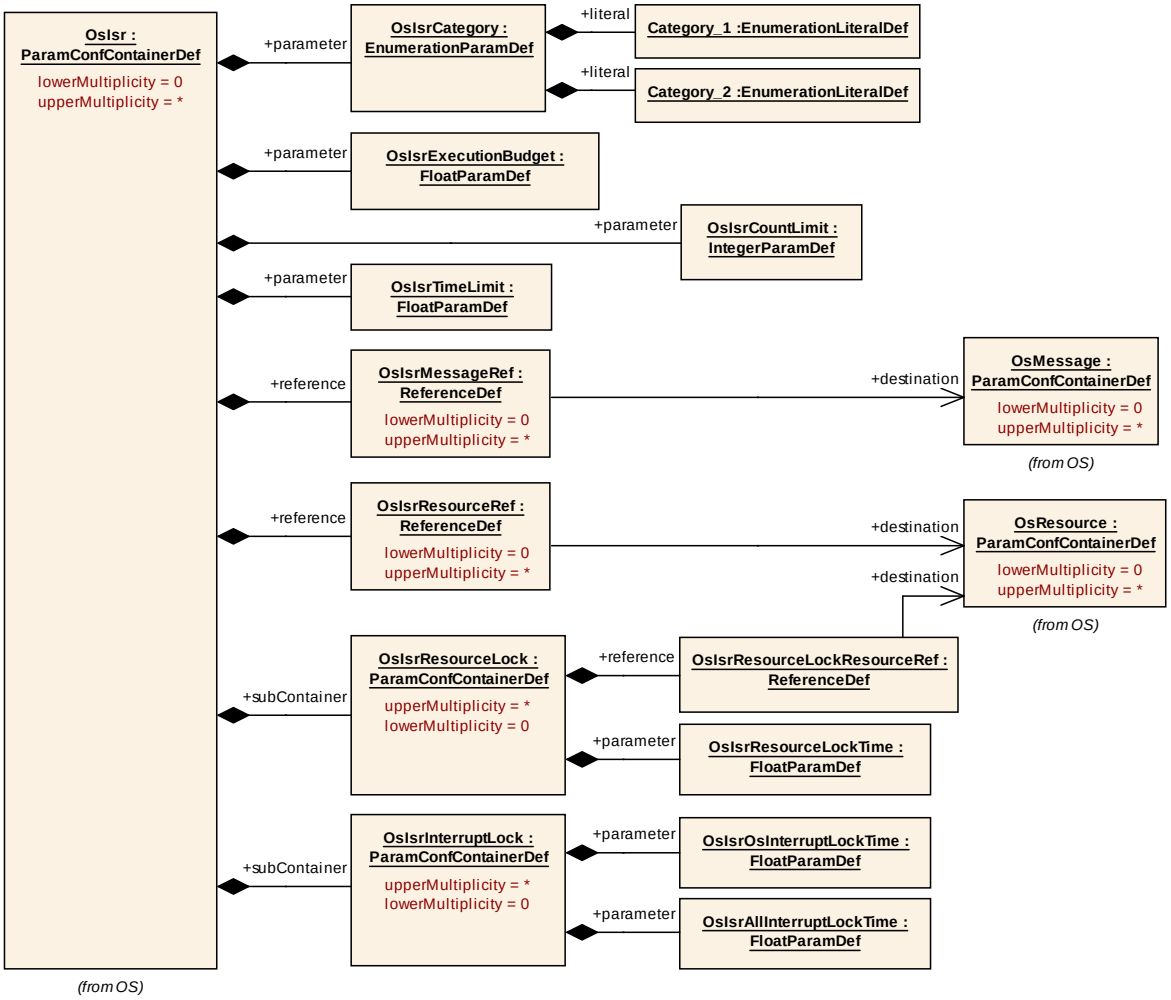


Figure 5.10: `OsIsr` details

OsIsr

SWS Item	[OS222, OS223, OS229, OS326]		
Container Name	OsIsr {ISR}		
Description	The OsISR container represents an OSEK interrupt service routine.		
Configuration Parameters			

Name last modified in: V1.0.0	OsIsrCategory {CATEGORY}		
Description	This attribute specifies the category of this ISR. Only values of 1 and 2 are allowed.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	Category_1	Interrupt is of category 1	
	Category_2	Interrupt is of category 2	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsIsrCountLimit {COUNTLIMIT}		
Description	OS223: The Configuration language shall allow th user to define the maximum arrival rate for each ISR. =COUNTLIMIT times in TIMELIMIT seconds		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsIsrExecutionBudget {EXECUTIONBUDGET}		
Description			
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsIsrMessageRef {MESSAGE}		
Description	This reference defines the list of messages accessed by this ISR.		
Multiplicity	0..*		
Type	Reference to OsMessage		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsIsrResourceRef {RESOURCE}		
Description	This reference defines the resources accessed by this ISR.		
Multiplicity	0..*		
Type	Reference to OsResource		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsIsrTimeLimit {TIMELIMIT}		
Description	OS223: The Configuration language shall allow th user to define the maximum arrival rate for each ISR. =COUNTLIMIT times in TIMELIMIT seconds		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency **Included Containers**			

Container Name	**Multiplicity**	**Scope / Depedency**
OsIsrInterruptLock	0..*	
OsIsrResourceLock	0..*	

OsIsrInterruptLock

SWS Item	
Container Name	OsIsrInterruptLock
Description	This parameter contains the time of the task in which the task has disabled either category 1 and 2 interrupts or only category 2 interrupts.
Configuration Parameters	

Name last modified in: V1.0.0	OsIsrAllInterruptLockTime {ALLINTERRUPTLOCKTIME}		
Description	specified in [s]		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsIsrOsInterruptLockTime {OSINTERRUPTLOCKTIME}		
Description	specified in [s]		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency **No Included Containers**			

OsIsrResourceLock

SWS Item			
Container Name	OsIsrResourceLock		
Description	This parameter contains the worst case time between getting and releasing a given resource.		
Configuration Parameters			
Name last modified in: V1.0.0	OsIsrResourceLockResourceRef {RESOURCE}		
Description	Reference to the resource the locking time is depending on		
Multiplicity	1		
Type	Reference to OsResource		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
Name last modified in: V1.0.0	OsIsrResourceLockTime {RESOURCELOCKTIME}		
Description	specified in [s]		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

5.1.6 Alarms

Figure 5.11 depicts the configuration structure of an `OsAlarm`.

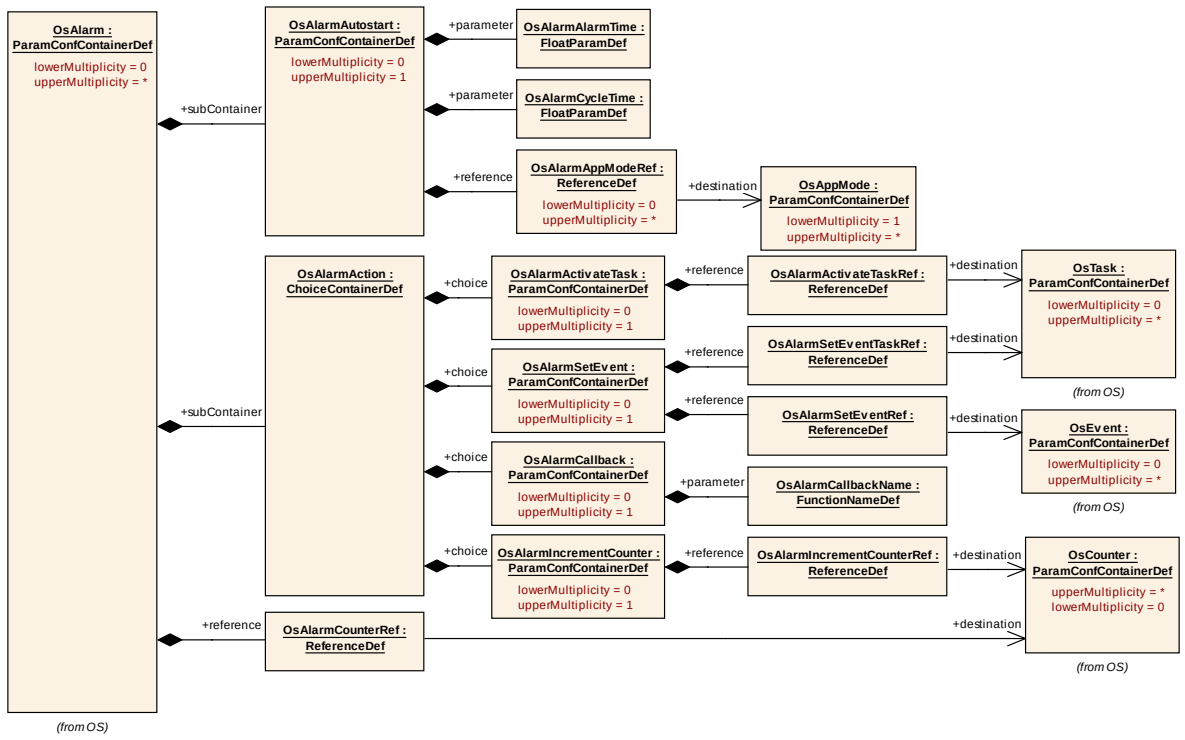


Figure 5.11: `OsAlarm` details

OsAlarm

SWS Item	[OS251, OS302]
Container Name	OsAlarm {ALARM}
Description	An OsAlarm may be used to asynchronously inform or activate a specific task. It is possible to start alarms automatically at system start-up depending on the application mode.
Configuration Parameters	

Name last modified in: V1.0.0	OsAlarmCounterRef {COUNTER}		
Description	Reference to the assigned counter for that alarm		
Multiplicity	1		
Type	Reference to OsCounter		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
OsAlarmAction [OsAlarmActivateTask, OsAlarmCallback, Os AlarmIncrementCounter, OsAlarmSetEvent]	1	
OsAlarmAutostart	0..1	

OsAlarmAutostart

SWS Item			
Container Name	OsAlarmAutostart {AUTOSTART}		
Description	If present this container defines if an alarm is started automatically at system start-up depending on the application mode.		
Configuration Parameters			
Name last modified in: V1.0.0	OsAlarmAlarmTime {ALARMTIME}		
Description	when shall the alarm expire first, specified in [s]		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsAlarmAppModeRef {APPMODE}		
Description	Reference to the application modes for which the AUTOSTART shall be performed		
Multiplicity	0..*		
Type	Reference to OsAppMode		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsAlarmCycleTime {CYCLETIME}		
Description	cycle time of a cyclic alarm, specified in [s]		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

OsAlarmActivateTask

SWS Item			
Container Name	OsAlarmActivateTask {ACTIVATETASK}		
Description			
Configuration Parameters			

Name last modified in: V1.0.0	OsAlarmActivateTaskRef {TASK}		
Description	Reference to the task that will be activated by that alarm action		
Multiplicity	1		
Type	Reference to OsTask		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

OsAlarmSetEvent

SWS Item			
Container Name	OsAlarmSetEvent {SETEVENT}		
Description			
Configuration Parameters			

Name last modified in: V1.0.0	OsAlarmSetEventRef {EVENT}		
Description	Reference to the event that will be set by that alarm action		
Multiplicity	1		
Type	Reference to OsEvent		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsAlarmSetEventTaskRef {TASK}		
Description	Reference to the task that will be activated by that event		
Multiplicity	1		
Type	Reference to OsTask		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

OsAlarmIncrementCounter

SWS Item			
Container Name	OsAlarmIncrementCounter {INCREMENTCOUNTER}		
Description			
Configuration Parameters			
Name last modified in: V1.0.0	OsAlarmIncrementCounterRef {COUNTER}		
Description	Reference to the counter that will be incremented by that alarm action		
Multiplicity	1		
Type	Reference to OsCounter		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

OsAlarmCallback

SWS Item			
Container Name	OsAlarmCallback {ALARMCALLBACK}		
Description	Callback function that is executed when this alarm is triggered		
Configuration Parameters			
Name last modified in: V1.0.0	OsAlarmCallbackName {ALARMCALLBACKNAME}		
Description	Name of the function that is called when this alarm callback is triggered.		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

5.1.7 Messages

OsMessages reflect the ECU-internal communication mechanism. In the AUTOSAR OS the NETWORKMESSAGE was removed. The ComSignal (described in section 4.2.2) is used for that purpose.

Figure 5.12 depicts the structure of the OsMessage configuration container.

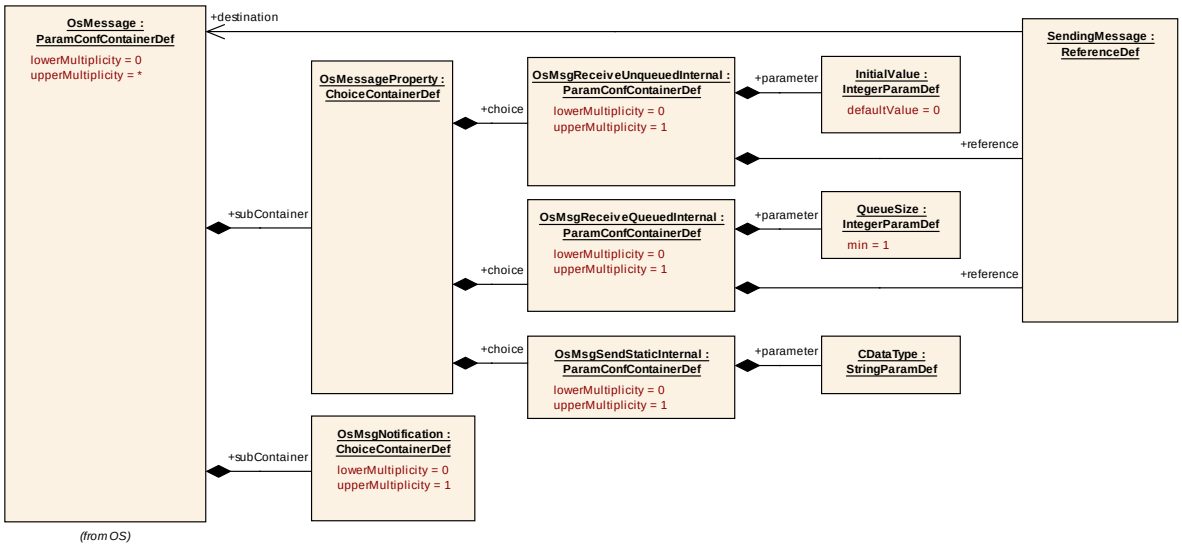


Figure 5.12: OsMessage details

OsMessage

SWS Item	[OS316]	
Container Name	OsMessage {MESSAGE}	
Description	An OsMessage container represents an OSEK message. The OSEK network messages are not handled in the AUTOSAR environment.	
Configuration Parameters Included Containers		
Container Name	Multiplicity	Scope / Depedency
OsMessageProperty [OsMsgReceiveQueued Internal, OsMsgReceive UnqueuedInternal, Os MsgSendStaticInternal]	1	
OsMsgNotification [OsMsgActivateTask, Os MsgComCallback, Os MsgFlag, OsMsgSet Event]	0..1	

5.1.7.1 OsMessage properties

Following tables describe the possible incarnations of an `OsMsgProperty` container.

OsMsgReceiveQueuedInternal

SWS Item			
Container Name	OsMsgReceiveQueuedInternal {RECEIVE_QUEUED_INTERNAL}		
Description	[pending]		
Configuration Parameters			

Name last modified in: V1.0.0	QueueSize {QUEUESIZE}		
Description	Defines the maximum number of message that the queue for a queued message can store.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	SendingMessage {SENDINGMESSAGE}		
Description	SendingMessage is used by a receiver of an internal message to identify the sender of the message. Therefore, this is a reference to a sent message.		
Multiplicity	1		
Type	Reference to OsMessage		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

OsMsgReceiveUnqueuedInternal

SWS Item			
Container Name	OsMsgReceiveUnqueuedInternal {RECEIVE_UNQUEUED_INTERNAL}		
Description	[pending]		
Configuration Parameters			

Name last modified in: V1.0.0	InitialValue {INITIALVALUE}		
Description	This parameter specifies the initial value of the message.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	SendingMessage {SENDINGMESSAGE}		
Description	SendingMessage is used by a receiver of an internal message to identify the sender of the message. Therefore, this is a reference to a sent message.		
Multiplicity	1		
Type	Reference to OsMessage		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

OsMsgSendStaticInternal

SWS Item			
Container Name	OsMsgSendStaticInternal {SEND_STATIC_INTERNAL}		
Description	[pending]		
Configuration Parameters			

Name last modified in: V1.0.0	CDataType {CDATATYPE}		
Description	This describes the data type of the message data using C language types (e.g. int or a structure name). The purpose of this parameter is the representation of the message in a form that is meaningful to the application.		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

5.1.7.2 OsMessage notifications

Figure 5.13 and the following tables describe the possible options of an `OsMsgNotification` container.

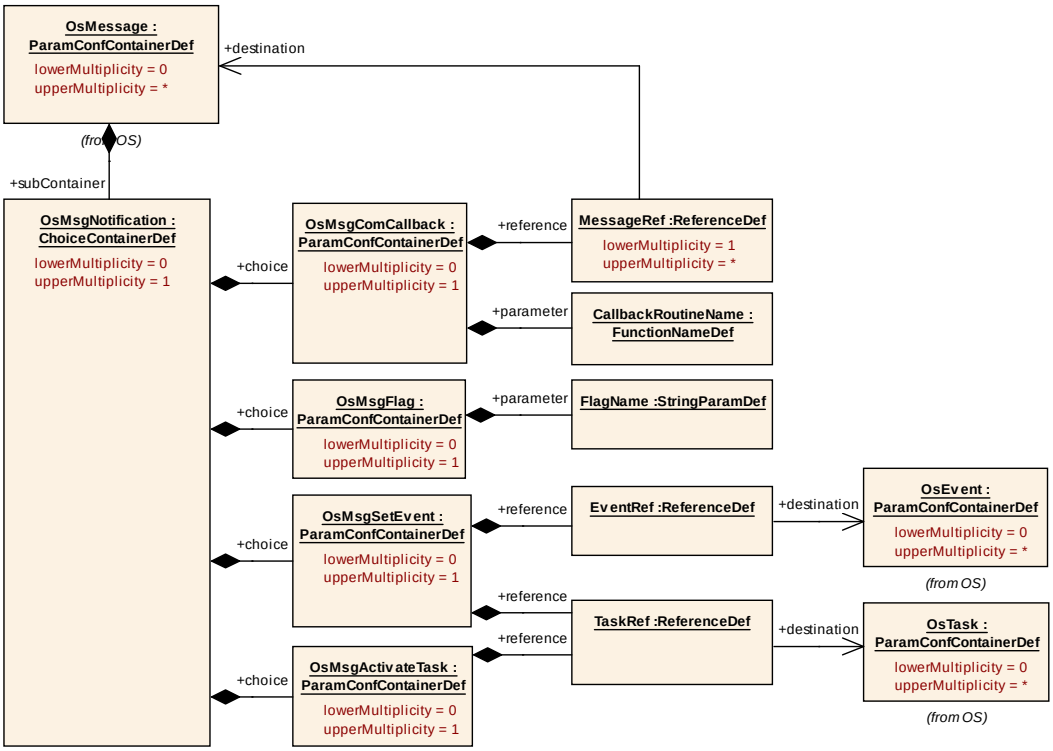


Figure 5.13: `OsMsgNotification` details

OsMsgFlag

SWS Item			
Container Name	OsMsgFlag {FLAG}		
Description	To perform the required notification a OsMsgFlag is set.		
Configuration Parameters			

Name last modified in: V1.0.0	FlagName {FLAGNAME}		
Description	Name of the OsMsgFlag		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

OsMsgSetEvent

SWS Item			
Container Name	OsMsgSetEvent {SETEVENT}		
Description	To perform a required notification an event is set for a task. The event and task are referenced by EventRef and TaskRef.		
Configuration Parameters			
Name last modified in: V1.0.0	EventRef {EVENT}		
Description	Reference to the OsEvent that is triggered by that OsMessage.		
Multiplicity	1		
Type	Reference to OsEvent		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
Name last modified in: V1.0.0	TaskRef {TASK}		
Description	Reference to the OsTask that is triggered by that OsMessage.		
Multiplicity	1		
Type	Reference to OsTask		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

OsMsgComCallback

SWS Item			
Container Name	OsMsgComCallback {COMCALLBACK}		
Description	To perform the required notification a callback routine is called. The name of the callback routine is specified in the CallbackRoutineName parameter. The reference message (MessageRef) must list all the messages that are sent and/or received by this callback routine.		
Configuration Parameters			
Name last modified in: V1.0.0	CallbackRoutineName {CALLBACKROUTINENAME}		
Description	This is the name of the callback routine in C notation.		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
Name last modified in: V1.0.0	MessageRef {MESSAGE}		
Description	Messages sent and/or received by the callback routine.		
Multiplicity	1..*		
Type	Reference to OsMessage		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

OsMsgActivateTask

SWS Item			
Container Name	OsMsgActivateTask {ACTIVATETASK}		
Description	To perform the required notification a task is activated. The task is referenced by TaskRef.		
Configuration Parameters			
Name last modified in: V1.0.0	TaskRef {TASK}		
Description	Reference to the OsTask that is triggered by that OsMessage.		
Multiplicity	1		
Type	Reference to OsTask		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

5.1.8 Counters

Figure 5.14 depicts the structure of the ECU configuration of counters.

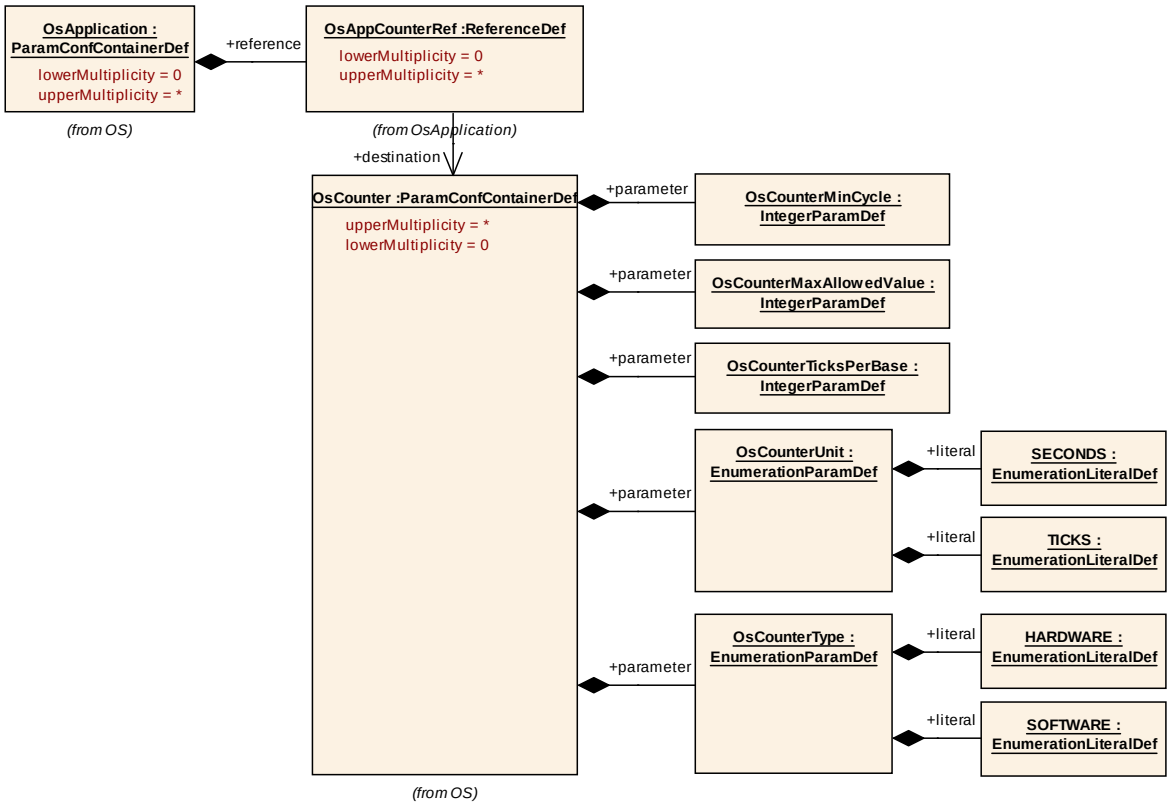


Figure 5.14: `OsCounter` details

OsCounter

SWS Item	[OS317, OS255, OS331]		
Container Name	OsCounter {COUNTER}		
Description	Configuration information for the counters that belong to the OsApplication.		
Configuration Parameters			

Name last modified in: V1.0.0	OsCounterMaxAllowedValue {MAXALLOWEDVALUE}		
Description	As defined by OSEK		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsCounterMinCycle {MINCYCLE}		
Description	As defined by OSEK		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsCounterTicksPerBase {TICKSPERBASE}		
Description			
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsCounterType {TYPE}		
Description	This parameter contains the natural type or unit of the counter.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	HARDWARE	This counter is driven by some hardware e.g. a hardware timer unit.	
	SOFTWARE	The counter is driven by some software which calls the IncrementCounter service.	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsCounterUnit {UNIT}		
Description	This parameter contains the unit type of the counter.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	SECONDS	The timing parameters of the alarms and schedule tables which belong to the counter are given in seconds.	
	TICKS	The timing parameters of the alarms and schedule tables which belong to the counter are given in ticks.	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

5.1.9 Events

Following table specifies the representation of an `OsEvent` in the ECU Configuration.

OsEvent

SWS Item			
Container Name	OsEvent {EVENT}		
Description	Representation of OS events in the configuration context. Adopted from the OSEK OIL specification.		
Configuration Parameters			
Name last modified in: V1.1.0	OsEventMask {MASK}		
Description	If event mask would be set to AUTO in OIL, this parameter should be omitted here.		
Multiplicity	0..1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

5.1.10 Resources

Figure 5.15 depicts the configuration structure of an `OsResource`.

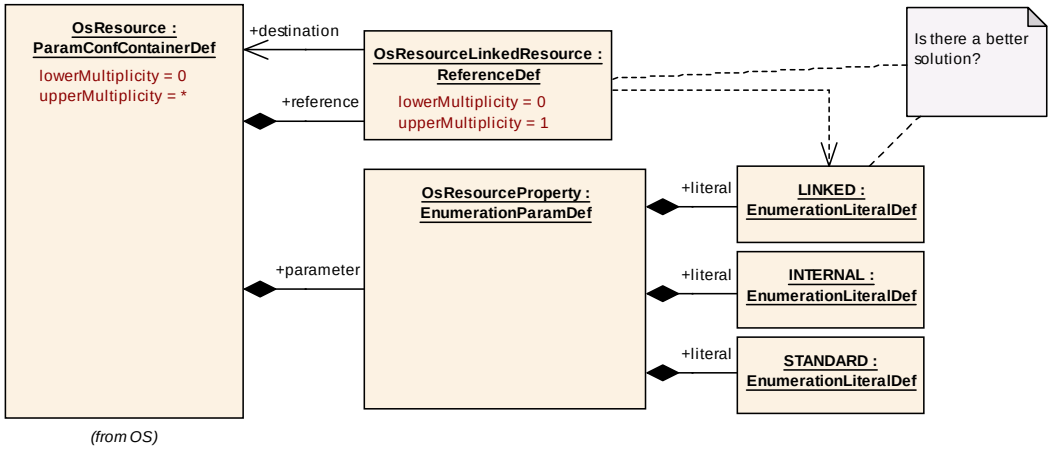


Figure 5.15: `OsResource` details

OsResource

SWS Item	[OS252]		
Container Name	OsResource {RESOURCE}		
Description	An OsResource object is used to co-ordinate the concurrent access by tasks and ISRs to a shared resource, e.g. the scheduler, any program sequence, memory or any hardware area.		
Configuration Parameters			

Name last modified in: V1.0.0	OsResourceLinkedResource {LINKEDRESOURCE}		
Description	Is there a better solution?		
Multiplicity	0..1		
Type	Reference to OsResource		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsResourceProperty {RESOURCEPROPERTY}		
Description			
Multiplicity	1		
Type	EnumerationParamDef		
Range	INTERNAL		
	LINKED	Is there a better solution?	
	STANDARD		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

5.1.11 Communications

The communication aspects of the ECU configuration were moved to the COM stack (see section 4.2 for details). The configuration artefacts defined below were kept to be compatible to an OSEK OS configuration.

Figure 5.16 depicts the structure of the `OsCom` and `OsNm` configuration containers.

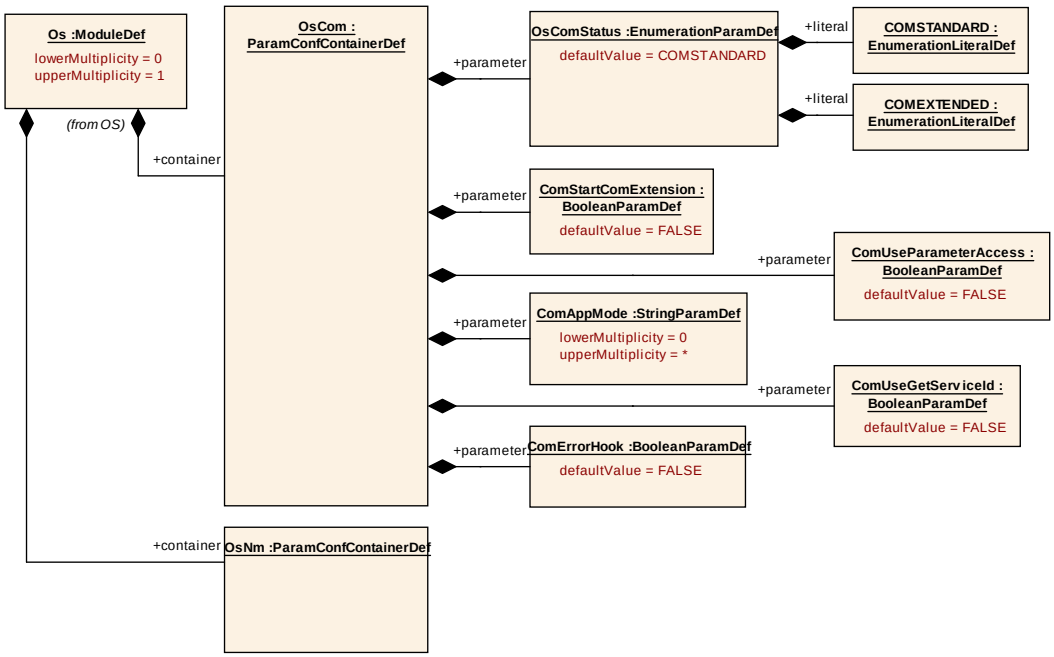


Figure 5.16: `OsCom` and `OsNm` details

OsCom

SWS Item	
Container Name	OsCom {COM}
Description	
Configuration Parameters	

Name last modified in: V1.0.0	ComAppMode {COMAPPMODE}		
Description			
Multiplicity	0..*		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComErrorHook {COMERRORHOOK}		
Description			
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComStartComExtension {COMSTARTCOMEXTENSION}		
Description			
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComUseGetServiceId {COMUSEGETSERVICEID}		
Description			
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	ComUseParameterAccess {COMUSEPARAMETERACCESS}		
Description			
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	OsComStatus {COMSTATUS}		
Description			
Multiplicity	1		
Type	EnumerationParamDef		
Range	COMEXTENDED		
	COMSTANDARD		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

OsNm

SWS Item	
Container Name	OsNm {NM}
Description	[pending]
Configuration Parameters	
No Included Containers	

5.2 Watchdog

5.2.1 Watchdog Manager

Figure 5.17 depicts the configuration structure of the BSW module `WdgM` that is specified in [33].

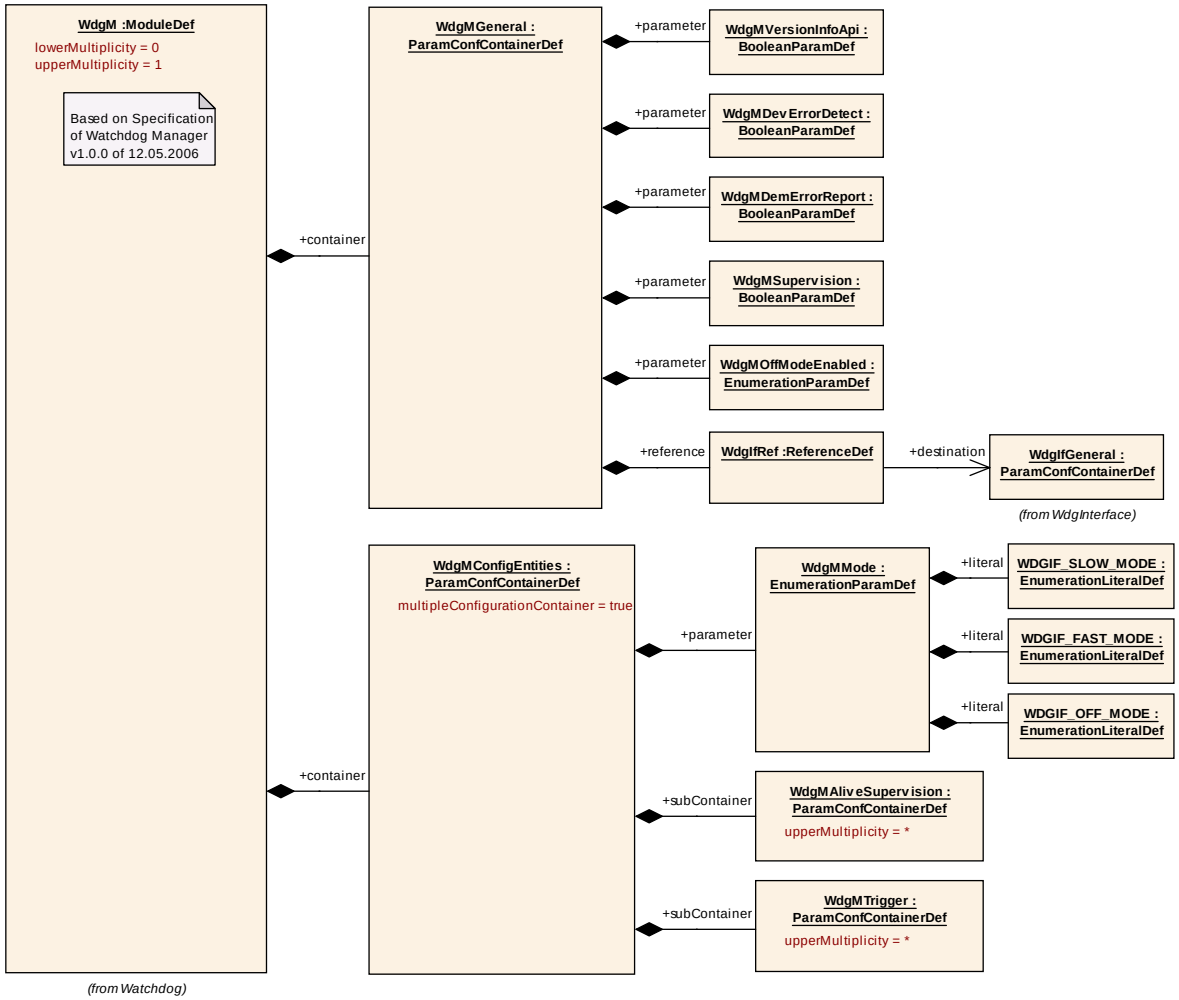


Figure 5.17: `WdgM` overview

Module Name	WdgM	
Module Description	Configuration of the WdgM (Watchdog Manager) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
WdgMConfigEntities	1	Combining container that holds all configurable supervision and trigger configuration parameters.
WdgMGeneral	1	Container structures all general configuration parameters of the Watchdog Manager.

5.2.1.1 General Watchdog Manager configuration parameters

The configuration container `WdgMGeneral` holds all configuration parameters that are globally defined for the `WdgM`.

WdgMGeneral

SWS Item			
Container Name	WdgMGeneral		
Description	Container structures all general configuration parameters of the Watchdog Manager.		
Configuration Parameters			
Name last modified in: V1.1.0	WdgIfRef		
Description	Reference to the watchdog interface that is used by the watchdog manager.		
Multiplicity	1		
Type	Reference to WdgIfGeneral		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	WdgMDemErrorReport {WDGM_DEM_ERROR_REPORT}		
Description	Preprocessor switch to enable/disable production error reporting. Shall be used to remove unneeded code segments regarding DEM features.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	WdgMDevErrorDetect {WDGM_DEV_ERROR_DETECT}		
Description	Preprocessor switch for enabling development error detection and reporting		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	WdgMOffModeEnabled {WDGM_OFF_MODE_ENABLED}		
Description	Parameter to enable/disable he access of Off-Mode selection (Off mode of the watchdog driver).		
Multiplicity	1		
Type	EnumerationParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	–	
	Post-build time	–	
Scope / Dependency	dependency: The parameter setting should be derived from the capabilities of the watchdog-HW and the related Wdg-Driver modules.		

Name last modified in: V1.0.0	WdgMSupervision {WDGM_ALIVE_SUPERVISION}		
Description	Preprocessor switch to enable/disable the alive-supervision mechanism. Shall be used to remove unneeded code segments regarding alive-supervision. Use Case: no supervised entity assigned		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	WdgMVersionInfoApi {WDGM_VERSION_INFO_API}		
Description	Preprocessor switch to enable/disable the existence of the API WdgM_GetVersionInfo. Shall be used to remove unneeded code segments.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

The configuration parameter `WDGM_NUMBER_OF_ALIVE_SUPERVISION_CONFIG` in [33] was not formalized in the ECU Configuration Specification as it can be calculated from the number of aggregated `WdgMAliveSupervision` containers.

WdgMConfigEntities

SWS Item			
Container Name	WdgMConfigEntities		
Description	Combining container that holds all configurable supervision and trigger configuration parameters.		
Configuration Parameters			

Name last modified in: V1.1.0	WdgMMode {WDGM_MODE}		
Description	The parameter shall contain the initial-mode of the Watchdog Manager.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	WDGIF_FAST_MODE	short timeout period	
	WDGIF_OFF_MODE	disabled	
	WDGIF_SLOW_MODE	long timeout period	
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	Variant 1
Scope / Dependency	dependency: The initial-mode must cooperate with the initial-mode of the watchdog-interface and watchdog-driver(s).		
Included Containers			

Container Name	Multiplicity	Scope / Depedency
WdgMAliveSupervision	1..*	
WdgMTrigger	1..*	

The configuration container `WdgMConfigEntities` was introduced to support multiple configuration sets in the `WdgM`.

5.2.1.2 Supervision Entities

A WdgM supervises several entities. The according configuration parameters are subsumed in the configuration container WdgMAliveSupervision described below. Figure 5.18 depicts the configuration structure.

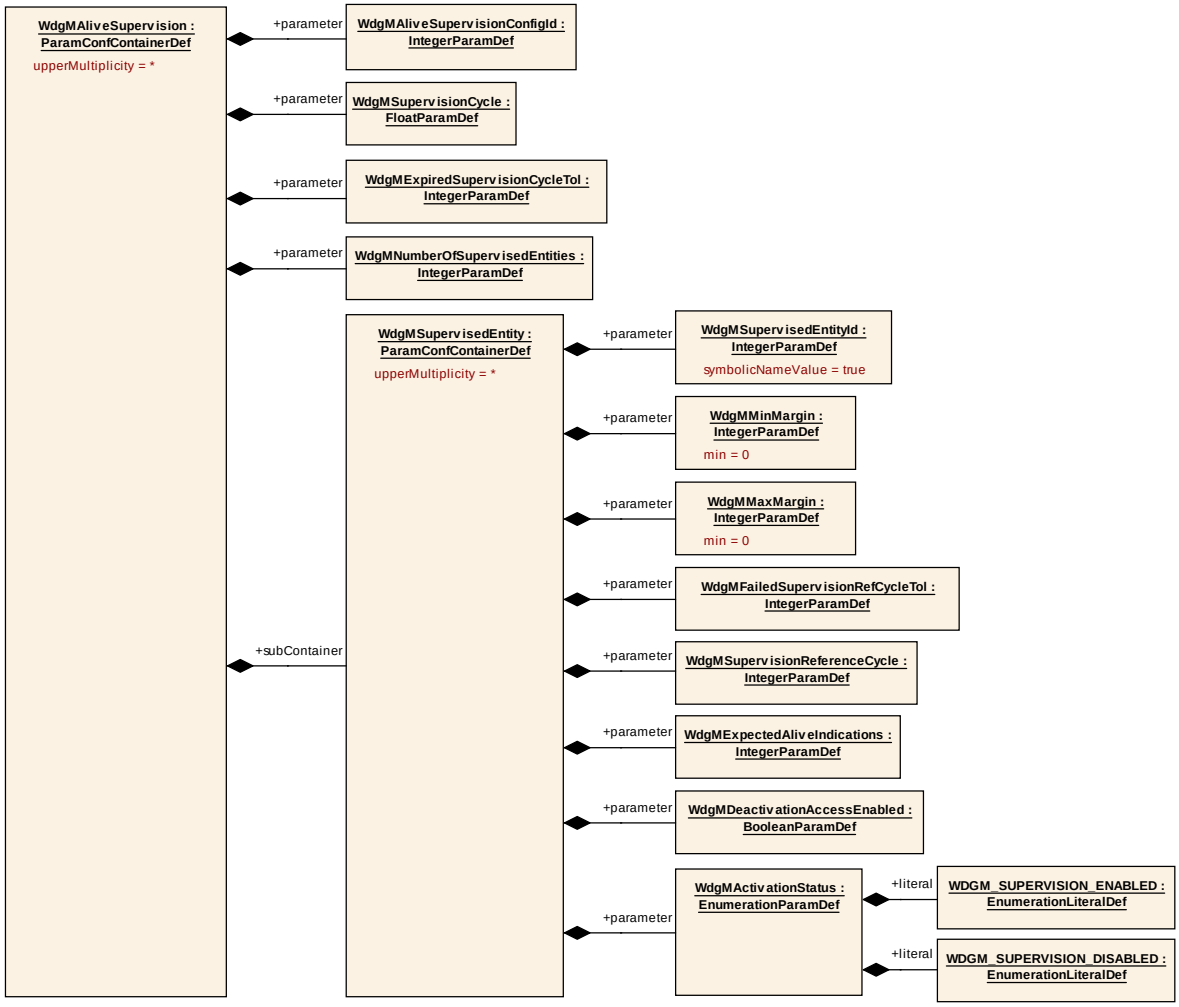


Figure 5.18: WdgMAliverSupervision

WdgMAliveSupervision

SWS Item			
Container Name	WdgMAliveSupervision		
Description	Container for a management structure to configure the alive-supervision of the Watchdog Manager. Its multiplicity describes the number of alive-supervision configuration sets.		
Configuration Parameters			

Name last modified in: V1.0.0	WdgMAliveSupervisionConfigId {WDGM_ALIVE_SUPERVISION_CONFIG_ID}		
Description	This parameter shall contain the identifier of the alive-supervision configuration set		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	Variant 1
Scope / Dependency			

Name last modified in: V1.0.0	WdgMExpiredSupervisionCycleTol {WDGM_EXPIRED_SUPERVISION_CYCLE_TOLERANCE}		
Description	This parameter shall be used to define a value that fixes the amount of expired supervision cycles for how long the blocking of watchdog triggering shall be postponed.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	Variant 1
Scope / Dependency			

Name last modified in: V1.1.0	WdgMNumberOfSupervisedEntities {WDGM_NUMBER_OF_SUPERVISED_ENTITIES}		
Description	This parameter shall contain the amount of supervised entities for this alive-supervision configuration set.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	WdgMSupervisionCycle {WDGM_SUPERVISION_CYCLE}		
Description	This parameter defines the schedule period of the main function WdgM_MainFunction_AliveSupervision. Unit: [s]		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
WdgMSupervisedEntity	1..*	

WdgMSupervisedEntity

SWS Item			
Container Name	WdgMSupervisedEntity		
Description	This container collects all configuration parameters to define the individual settings of a Supervised Entity.		
Configuration Parameters			

Name last modified in: V1.0.0	WdgMActivationStatus {WDGM_ACTIVATION_STATUS}		
Description	This parameter shall contain the activation status of the corresponding Supervised Entity.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	WDGM_SUPERVISION_DISABLED	Alive Supervision of the corresponding SE needs to be armed/activated by the service WdgM_ActivateAliveSupervision.	
	WDGM_SUPERVISION_ENABLED	Alive Supervision of the corresponding SE is armed/activated right from the start of Alive Supervision.	
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	Variant 1
Scope / Dependency			

Name last modified in: V1.0.0	WdgMDeactivationAccessEnabled {WDGM_DEACTIVATION_ACCESS_ENABLED}		
Description	This parameter shall be used to grant the access of the service WdgM_DeactivateAliveSupervision() for affecting the activation status of the corresponding Supervised Entity.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	Variant 1
Scope / Dependency			

Name last modified in: V1.0.0	WdgMExpectedAliveIndications {WDGM_EXPECTED_ALIVE_INDICATIONS}		
Description	This parameter shall contain the amount of expected alive indications within the referenced amount of defined supervision cycles according to corresponding SE.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	Variant 1
Scope / Dependency			

Name last modified in: V1.0.0	WdgMFailedSupervisionRefCycleTol {WDGM_FAILED_SUPERVISION_REFERENCE_CYCLE_TOLERANCE}		
Description	This parameter shall contain the acceptable amount of failed supervision reference cycles that shall be used by checkup of alive supervision according to corresponding SE.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	Variant 1
Scope / Dependency			

Name last modified in: V1.0.0	WdgMMaxMargin {WDGM_MAX_MARGIN}		
Description	This parameter shall contain the amount of alive indications that are acceptable to be additional on the expected alive indications within the corresponding supervision reference cycle.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	Variant 1
Scope / Dependency			

Name last modified in: V1.0.0	WdgMMinMargin {WDGM_MIN_MARGIN}		
Description	This parameter shall contain the amount of alive indications that are acceptable to be missed from the expected alive indications within the corresponding supervision reference cycle.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	Variant 1
Scope / Dependency			

Name last modified in: V1.0.0	WdgMSupervisedEntityId {WDGM_SUPERVISED_ENTITY_ID}		
Description	This parameter shall contain the identifier of he supervised entity.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	Variant 1
Scope / Dependency			

Name last modified in: V1.0.0	WdgMSupervisionReferenceCycle {WDGM_SUPERVISION_REFERENCE_CYCLE}		
Description	This parameter shall contain the amount of supervision cycles to be used as reference by the alive-supervision mechanism to perform the checkup with counted alive indications according to corresponding SE.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	Variant 1
Scope / Dependency			
No Included Containers			

5.2.1.3 Triggers

The triggering configuration information of the used watchdog devices is specified in this section. An overview of the `WdgMTrigger` configuration container is shown in figure 5.19 and the table below.

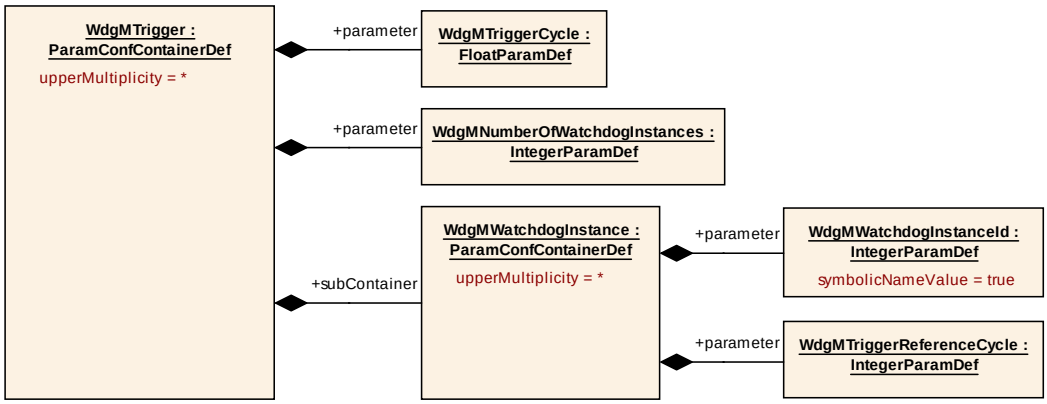


Figure 5.19: WdgMTrigger

WdgMTrigger

SWS Item			
Container Name	WdgMTrigger		
Description	Container for management structure to configure the trigger service of the Watchdog Manager. Its multiplicity describes the number of trigger configuration sets.		
Configuration Parameters			

Name last modified in: V1.0.0	WdgMNumberOfWatchdogInstances {WDGM_NUMBER_OF_WATCHDOG_INSTANCES}		
Description	This parameter shall contain the amount of watchdog instances.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	WdgMTriggerCycle {WDGM_TRIGGER_CYCLE}		
Description	This parameter defines the schedule period of the main function WdgM_MainFunction_Trigger. Unit: [s]		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	Variant 1
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
WdgMWatchdogInstance	1..*	

WdgMWatchdogInstance

SWS Item			
Container Name	WdgMWatchdogInstance		
Description	This container collects all configuration parameters to define the individual settings of a watchdog instance.		
Configuration Parameters			

Name last modified in: V1.0.0	WdgMTriggerReferenceCycle {WDGM_TRIGGER_REFERENCE_CYCLE}		
Description	This parameter shall contain the amount of trigger cycles to be used as reference by the trigger service of the Watchdog Manager to perform the triggering of the corresponding watchdog instance.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	Variant 1
Scope / Dependency			

Name last modified in: V1.0.0	WdgMWatchdogInstanceId {WDGM_WATCHDOG_INSTANCE_ID}		
Description	This parameter shall contain the identifier of the watchdog instance.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	Variant 1
Scope / Dependency			
No Included Containers			

5.2.2 Watchdog Interface

The structure of the watchdog interface specified in [32] is depicted in figure 5.20. The according configuration container can be seen in the following tables.

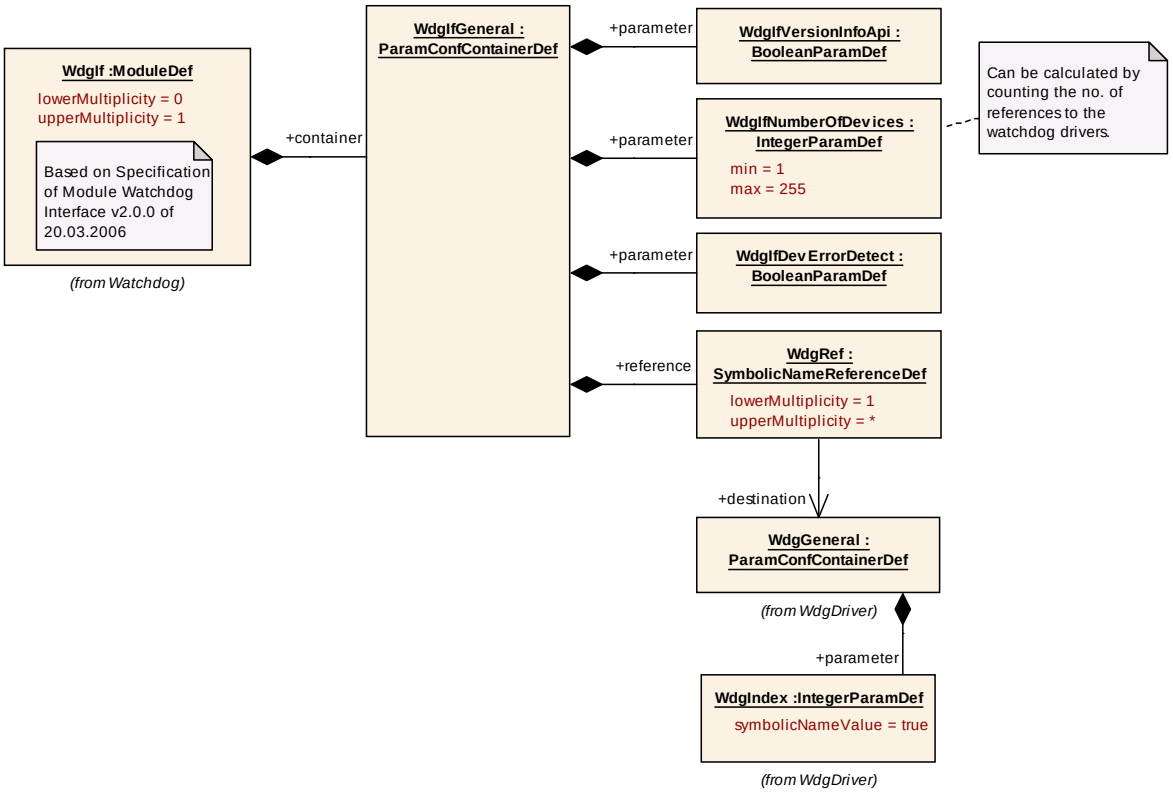


Figure 5.20: `WdgIf` overview

Module Name	WdgIf	
Module Description	Configuration of the WdgIf (Watchdog Interface) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
WdgIfGeneral	1	This container collects all generic watchdog interface parameters.

WdgIfGeneral

SWS Item			
Container Name	WdgIfGeneral		
Description	This container collects all generic watchdog interface parameters.		
Configuration Parameters			

Name last modified in: V1.0.0	WdgIfDevErrorDetect {WDGIF_DEV_ERROR_DETECT}		
Description	Pre-processor switch for enabling the development error detection and reporting		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	WdgIfNumberOfDevices {WDGIF_NUMBER_OF_DEVICES}		
Description	Constant specifying the number of controlled watchdog drivers. Can be calculated by counting the no. of references to the watchdog drivers.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 255		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	WdgIfVersionInfoApi {WDGIF_VERSION_INFO_API}		
Description	Pre-processor switch to enable / disable the service returning the version information.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	WdgRef		
Description	Reference to the watchdog drivers that are controlled by the watchdog interface.		
Multiplicity	1..*		
Type	Symbolic name reference to WdgGeneral		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

5.2.3 Watchdog Driver

The structure of the watchdog driver specified in [31] is depicted in figure 5.21. The according configuration container can be seen in the following table.

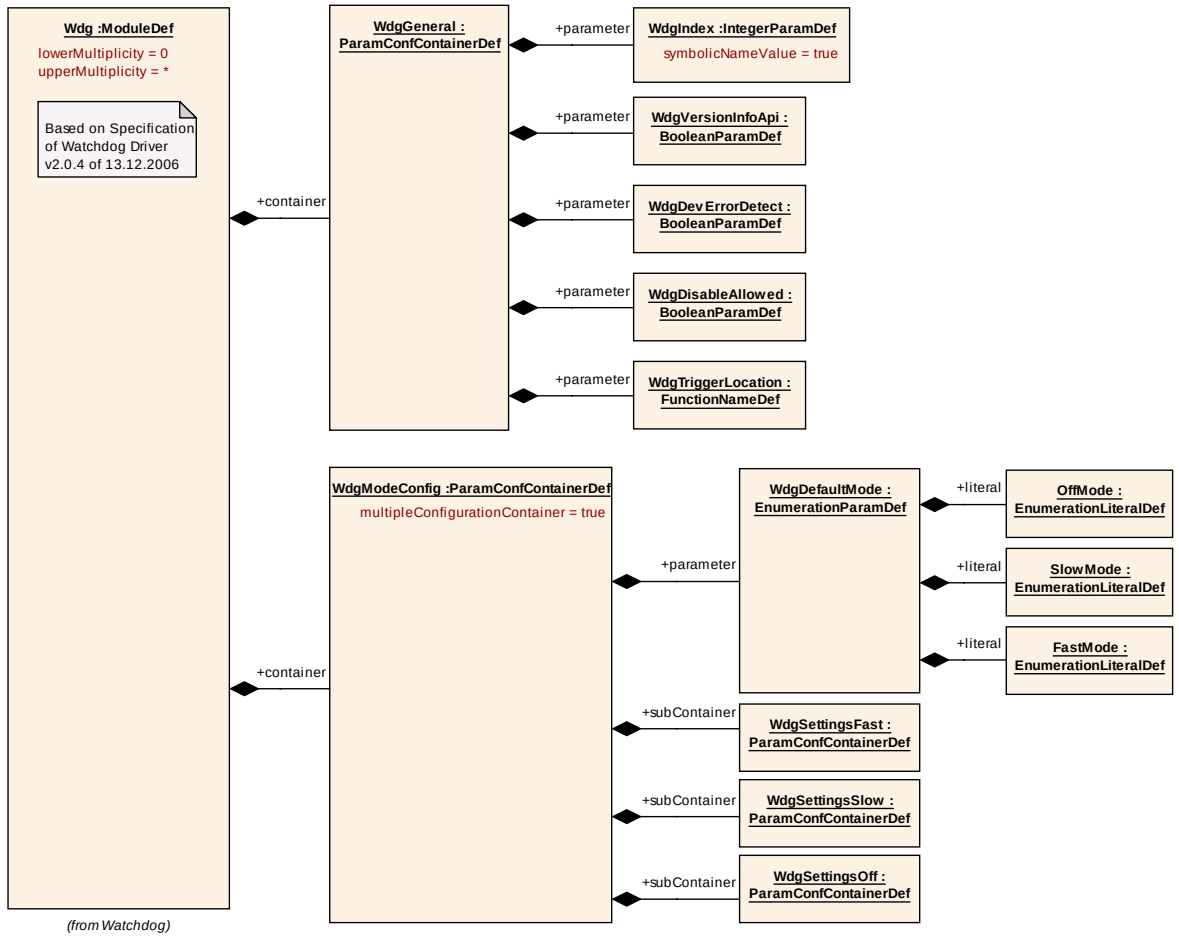


Figure 5.21: `Wdg` overview

Module Name	Wdg	
Module Description	Configuration of the Wdg (Watchdog driver) module. This should be the extern watchdog driver. For internal watchdogs, no port pins / DIO is used.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
WdgGeneral	1	All general parameters of the watchdog driver are collected here.
WdgModeConfig	1	Configuration items for the different watchdog modes
WdgPublishedInformation	1	Container holding all Wdg specific published information parameters

WdgGeneral

SWS Item			
Container Name	WdgGeneral		
Description	All general parameters of the watchdog driver are collected here.		
Configuration Parameters			

Name last modified in: V1.0.0	WdgDevErrorDetect {WDG_DEV_ERROR_DETECT}		
Description	Compile switch to enable / disable development error detection for this module.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	WdgDisableAllowed {WDG_DISABLE_ALLOWED}		
Description	Compile switch to allow / forbid disabling the watchdog driver during runtime.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	WdgIndex		
Description	Represents the watchdog driver's ID so that it can be referenced by the watchdog interface.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	WdgTriggerLocation {WDG_TRIGGER_LOCATION}		
Description	Location (memory address) of the watchdog trigger routine.		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	WdgVersionInfoApi		
Description	Compile switch to enable / disable the version information API		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

5.2.3.1 Hardware-specific mode configuration

The mode configuration of `Wdg` is hardware-dependent. Therefore it is not possible to specify a common set of configuration parameters. Nevertheless configuration containers for the standardized modes `FAST`, `SLOW` and `OFF` are defined here. These can be extended with vendor-specific configuration parameters.

WdgModeConfig

SWS Item	[WDG082]		
Container Name	WdgModeConfig {Wdg_ModeConfiguration}		
Description	Configuration items for the different watchdog modes		
Configuration Parameters			
Name last modified in: V1.1.0	WdgDefaultMode {WDG_DEFAULT_MODE}		
Description	Default mode for watchdog driver initialization		
Multiplicity	1		
Type	EnumerationParamDef		
Range	FastMode	Default watchdog mode is ”fast”	
	OffMode	Default watchdog mode is ”off”	
	SlowMode	Default watchdog mode is ”slow”	
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency			
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
WdgSettingsFast	1		
WdgSettingsOff	1		
WdgSettingsSlow	1		

WdgSettingsFast

SWS Item	
Container Name	WdgSettingsFast {WDG_SETTINGS_FAST}
Description	Hardware dependent settings for the watchdog driver’s ”fast” mode.
Configuration Parameters	
No Included Containers	

WdgSettingsSlow

SWS Item	
Container Name	WdgSettingsSlow {WDG_SETTINGS_SLOW}
Description	Hardware dependent settings for the watchdog driver’s ”slow” mode.
Configuration Parameters	
No Included Containers	

WdgSettingsOff

SWS Item	
Container Name	WdgSettingsOff {WDG_SETTINGS_OFF}
Description	Hardware dependent settings for the watchdog driver's "off" mode.
Configuration Parameters **No Included Containers**	

5.2.3.2 Module-specific Published Information

Following published configuration parameters are additionally defined by the watchdog driver, beside the parameters of the `CommonPublishedInformation` container in section 1.1.2.

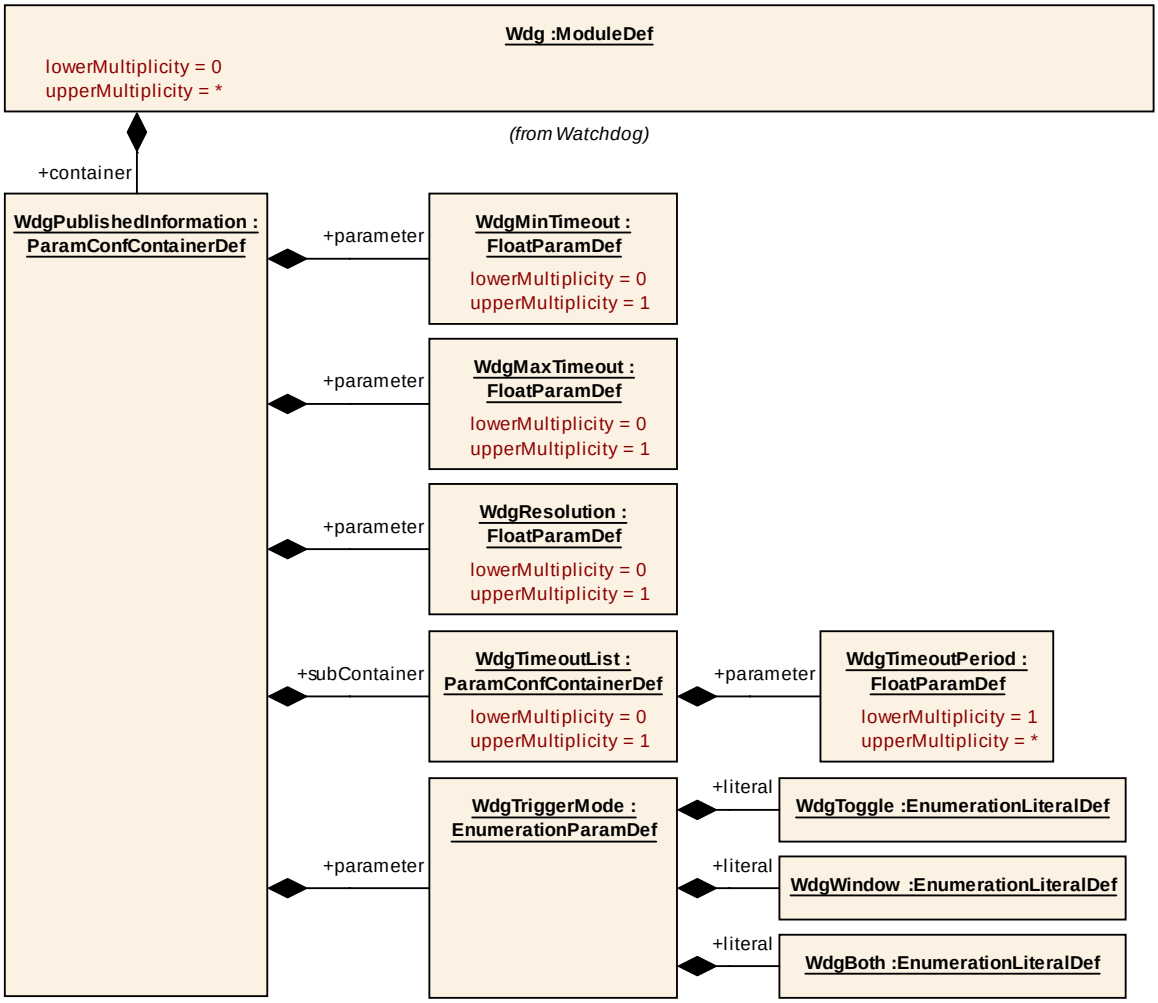


Figure 5.22: `Wdg`-specific published information

WdgPublishedInformation

SWS Item	[WDG074]		
Container Name	WdgPublishedInformation		
Description	Container holding all Wdg specific published information parameters		
Configuration Parameters			

Name last modified in: V1.1.0	WdgMaxTimeout {WDG_MAX_TIMEOUT}		
Description	Maximum timeout period in [s].		
Multiplicity	0..1		
Type	FloatParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.1.0	WdgMinTimeout {WDG_MIN_TIMEOUT}		
Description	Minimum timeout period in [s].		
Multiplicity	0..1		
Type	FloatParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.1.0	WdgResolution {WDG_RESOLUTION}		
Description	Resolution of watchdog timeout period in [s].		
Multiplicity	0..1		
Type	FloatParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.1.0	WdgTriggerMode {WDG_TRIGGER_MODE}		
Description	Watchdog trigger mode (toggle/window/both)		
Multiplicity	1		
Type	EnumerationParamDef		
Range	WdgBoth		
	WdgToggle		
	WdgWindow		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
WdgTimeoutList	0..1	

5.3 Diagnostics

The diagnostics section of Basic Software covers 3 main items:

1. FIM - Function Inhibition Manager (Section 5.3.1)
2. DEM - Diagnostic Event Manager (Section 5.3.2)
3. DCM - Diagnostic Communication Manager (Section 5.3.3)

The configuration of each Basic Software item is covered in a separate sub-section. It should be noted that run state values have been explicitly put in place during modeling to represent a requirement (either from the SWS or ECU Configuration methodology) even where that value is the same as the default value for the attribute. No explicit run state value has been entered where no requirement could be identified by reference to the SWS or ECU Configuration methodology, in which cases the run state will assume the global Parameter Definition default for those attributes but the values are not visible in the diagrams.

Each diagram or table may have notes associated but not necessarily for every feature or parameter. Only special features of specific note will be mentioned. These may be occasions when explicit values are defined but their derivation is not obvious; it may be that the SWS definition is unclear and assumptions have been made during Configuration Parameter modeling; it may be to indicate missing or incomplete configuration definition or known errors.

5.3.1 FIM - Function Inhibition Manager

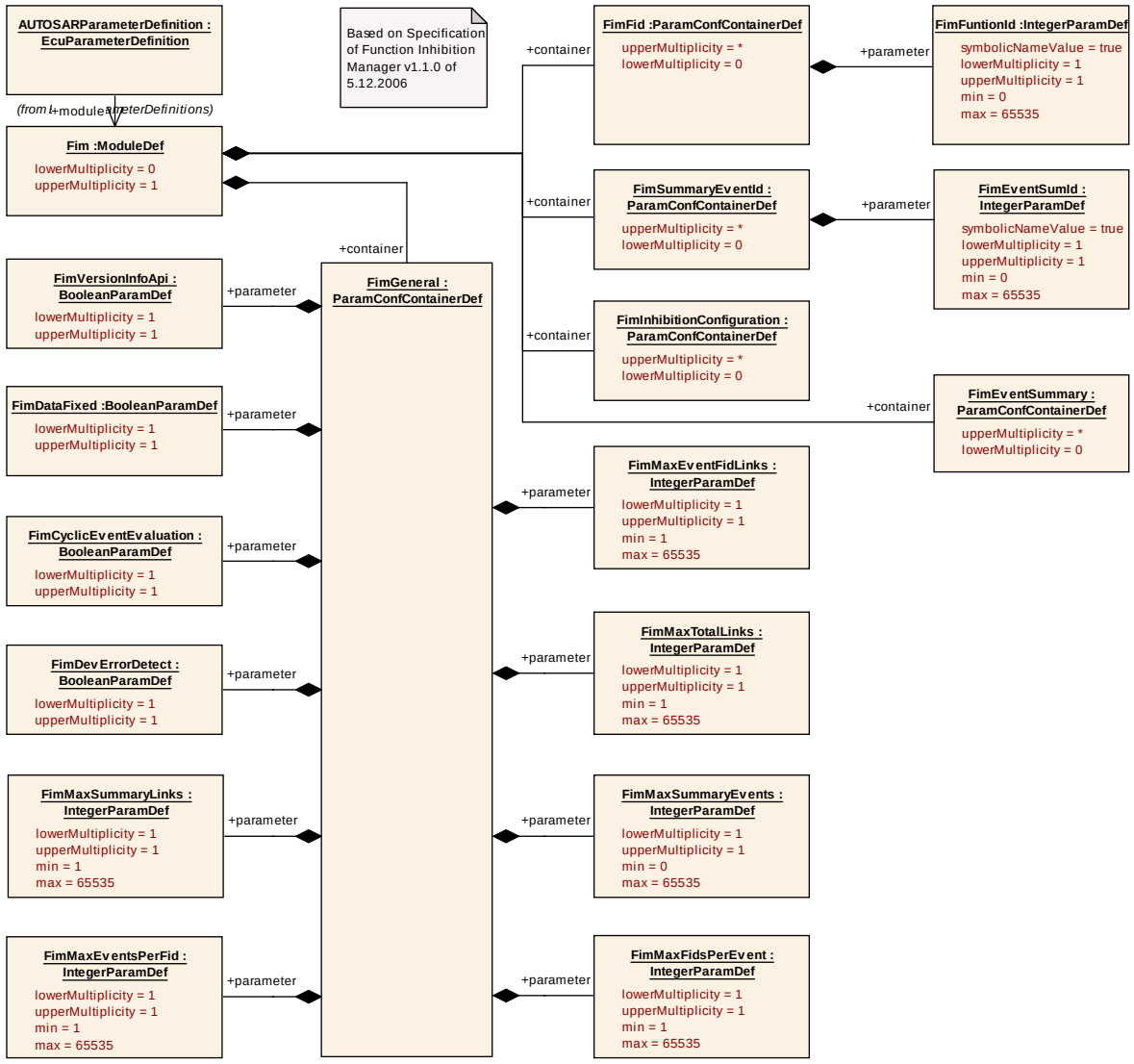


Figure 5.23: FIM - Function Inhibition Manager

The Function Inhibition Manager is responsible for running & inhibition of software components based on events. A function consists of one or several sets of lines of code located inside a software components. The software component provides relationship data that link events to affected function units. The function is controlled by FIM requests to run and to inhibit, addressed via numbers called function unit identifiers (FIDs).

The goal of the FIM is to control functionality of SW components based on:

- Faults (such as Diagnostic Trouble Code events)
- ECU and vehicle states
- Diagnostic tester commands
- EOL configuration (function enabling/disabling in EEPROM)
- Special case functions / driver demands (e.g. ESP on/off)

The FIM is linked to the Diagnostic Event Manager to which the statuses of events are reported.

The current configuration parameters for FIM are defined from the FIM Software Specification v1.00. The following sections descibe the configuration parameters of DEM in details. Further details may be obtained from [27].

Variants

- Variant 1: This variant is limited to pre-compile configuration parameters only.
- Variant 2: This variant allows a mix of pre-compile time- and post build time-configuration parameters. However, there is also another meaning of variants namely variant coding and applicability of the same software/ECU to different applications/vehicles. This is not explicitly supported due to high resource usage. The inhibit conditions provided by the software components can be considered as a superset of all inhibit conditions for all variants. Based on that, the inhibit configuration has to be derived for all events and FIDs in one project. If an EventId or FID is not supported in a certain variant, the link between them has no effect. The requirements BSW00404 and BSW00405 are therefore not applicable.

Module Name	Fim	
Module Description	Configuration of the Fim (Function Inhibition Manager) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
FimEventSummary	0..*	The summarized EventId definition record consists of a summarized event ID and a specific EventId. This record means that a particular FID that has to be disabled in case of summarized event (defined above) is to be disabled in any of the specific events. A possible solution could be assigning events as summarized events along with a list of specific events. During the configuration process the summarized event substitutes the referenced single events. However, it is not outlined how this requirement is solved - whether by configuration process or by implementation within the FIM. The FIM configuration tool could also build up a suitable data structure for summarized events and deal with it in the FIM implementation.

Container Name	Multiplicity	Scope / Dependency
FimFid	0..*	This container includes symbolic names of all FIDs.
FimGeneral	1	
FimInhibitionConfiguration	0..*	This container includes all configuration parameters concerning the relationship between event and FID.
FimSummaryEventId	0..*	The summarized EventId definition record defines the existence of a summarized event with a specific name. This summarized event can be referenced in the EventSummary (as FIM_OUTPUT_SUMMARIZED_EVENT) and Inhibition configuration (as FIM_INH_EVENT_ID).

FimGeneral

SWS Item	[FIM040]
Container Name	FimGeneral {FimConfiguation}
Description	
Configuration Parameters	

Name (last modified in: V1.0.0)	FimCyclicEventEvaluation {FIM_CYCLIC_EVENT_EVALUATION} [FIM040]		
Description	This configuration parameter specifies whether the evaluation of DEM events is performed by the FIM cyclically or the DEM informs FIM about changes of event states. ON: FIM cyclically evaluates event states in the DEM. OFF: DEM informs FIM about changes of event states.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name (last modified in: V1.0.0)	FimDataFixed {FIM_DATA_FIXED} [FIM008]		
Description	Enable or disable calibration of inhibit relations The scope of the parameter is to meet the requirement (FIM008) to have the option to calibrate inhibit data on the one hand side and also to provide the option to protect inhibit data for consistency reasons.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name (last modified in: V1.0.0)	FimDevErrorDetect {FIM_DEV_ERROR_DETECT} [FIM040]		
Description	This configuration parameter is used to switch on or to switch off the detection of development errors during development.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FimMaxEventFidLinks {FIM_MAXIMUM_EVENT_FID_LINKS} [FIM040]		
Description	This configuration parameter specifies the total maximum number of links between EventIds and FIDs.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 65535		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FimMaxEventsPerFid {FIM_MAXIMUM_EVENTS_PER_FID} [FIM040]		
Description	This configuration parameter specifies the maximum number of EventIds that can be linked to a single FID.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 65535		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FimMaxFidsPerEvent {FIM_MAXIMUM_FIDS_PER_EVENT} [FIM040]		
Description	This configuration parameter specifies the maximum number of FIDs that can be linked to a single event.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 65535		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FimMaxSummaryEvents {FIM_MAXIMUM_SUMMARY_EVENTS} [FIM040]		
Description	This configuration parameter specifies the maximum number of summarized events that can be configured.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FimMaxSummaryLinks {FIM_MAXIMUM_SUMMARY_LINKS} [FIM040]		
Description	This configuration parameter specifies the total maximum number of links between EventIds and FIDs plus the number of links between EventIds and summarized events.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 65535		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FimMaxTotalLinks {FIM_MAXIMUM_TOTAL_LINKS} [FIM040]		
Description	This configuration parameter specifies the total maximum number of links between EventIds and FIDs plus the number of links between EventIds and summarized events.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 65535		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FimVersionInfoApi {FIM_VERSION_INFO_API} [FIM040]		
Description	This configuration parameter is used to switch on or to switch off the API to get the version information.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

FimFid

SWS Item	[FIM039]
Container Name	FimFid {FimFID}
Description	This container includes symbolic names of all FIDs.
Configuration Parameters	

Name last modified in: V1.0.0	FimFuntionId {FIM_FUNCTION_ID} [FIM039]		
Description	The configuration parameter is used as an ID which represents a functionality. FIM_FUNCTION_ID is the unique identifier assigned during FIM configuration.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

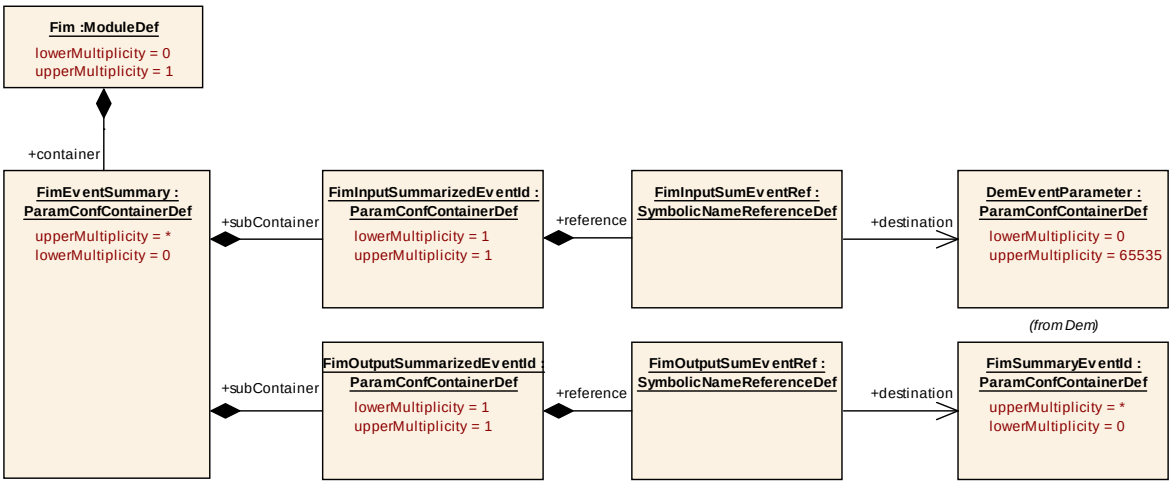


Figure 5.24: FIM - Event Summary

5.3.1.1 FimEventSummaryConfiguration

FimEventSummary

SWS Item	[FIM037]	
Container Name	FimEventSummary {FimEventSummary}	
Description	The summarized EventId definition record consists of a summarized event ID and a specific EventId. This record means that a particular FID that has to be disabled in case of summarized event (defined above) is to be disabled in any of the specific events. A possible solution could be assigning events as summarized events along with a list of specific events. During the configuration process the summarized event substitutes the referenced single events. However, it is not outlined how this requirement is solved - whether by configuration process or by implementation within the FIM. The FIM configuration tool could also build up a suitable data structure for summarized events and deal with it in the FIM implementation.	
Configuration Parameters Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
FimInputSummarized EventId	1	
FimOutputSummarized EventId	1	

FimInputSummarizedEventId

SWS Item	[FIM037]		
Container Name	FimInputSummarizedEventId {FIM_INPUT_SUMMARIZED_EVENT}		
Description	The configuration parameter is used as an input for a summarized event.		
Configuration Parameters			
Name last modified in: V1.0.0	FimInputSumEventRef		
Description			
Multiplicity	1		
Type	Symbolic name reference to DemEventParameter		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

FimOutputSummarizedEventId

SWS Item	[FIM037]		
Container Name	FimOutputSummarizedEventId {FIM_OUTPUT_SUMMARIZED_EVENT}		
Description	The configuration parameter is used as an output for a summarized event.		
Configuration Parameters			
Name last modified in: V1.0.0	FimOutputSumEventRef		
Description			
Multiplicity	1		
Type	Symbolic name reference to FimSummaryEventId		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

5.3.1.2 FimInhibitionConfiguration

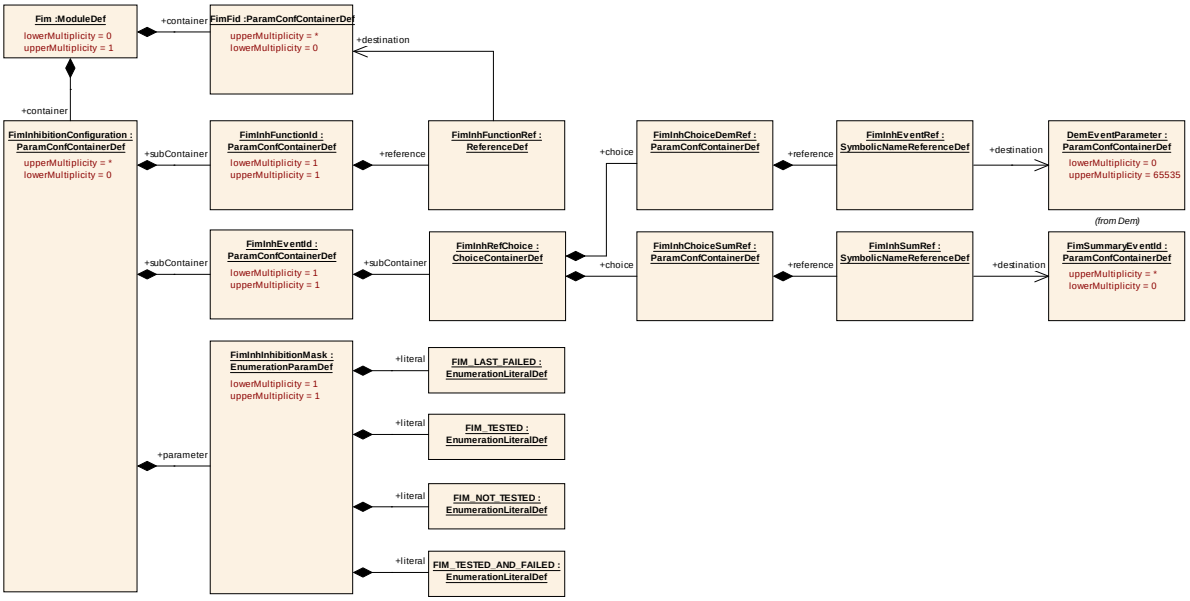


Figure 5.25: FIM - Inhibition Configuration

FimInhibitionConfiguration

SWS Item	[FIM038]		
Container Name	FimInhibitionConfiguration {FimInhibitionConfiguration}		
Description	This container includes all configuration parameters concerning the relationship between event and FID.		
Configuration Parameters			

Name last modified in: V1.0.0	FimInhInhibitionMask {FIM_INH_INHIBITION_MASK} [FIM038]		
Description	The configuration parameter is used to specify the inhibition mask for an event - FID relation.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	FIM_LAST_FAILED	Last Failed - Use case: Re-configuration, avoiding follow-up errors	
	FIM_NOT_TESTED	Not Tested this cycle - Use case: Scheduling of monitors.	
	FIM_TESTED	Tested - Use case: Self-deactivation, check during driving cycle.	
	FIM_TESTED_AND_FAILED	Tested and Failed - Use case: Avoiding deadlocks, repeated monitoring.	
Configuration Class	**Pre-compile time**	X	Pre-compile only
	Link time	–	
	Post-build time	L	Pre-compile and post-build
Scope / Dependency			

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
FimInhEventId	1	
FimInhFunctionId	1	

FimInhEventId

SWS Item	[FIM038]		
Container Name	FimInhEventId {FIM_INH_EVENT_ID}		
Description	The configuration parameter is used for an existing DEM event and summarized events as well.		
Configuration Parameters **Included Containers**			
Container Name	**Multiplicity**	**Scope / Depedency**	
FimInhRefChoice [FimInhChoiceDemRef, FimInhChoiceSumRef]	1		

FimInhFunctionId

SWS Item	[FIM038]		
Container Name	FimInhFunctionId {FIM_INH_FUNCTION_ID}		
Description	The configuration parameter is used as an ID which represents a functionality. FIM_INH_FUNCTION_ID is the FID to be considered in a particular inhibit relation EventId-FID-Mask.		
Configuration Parameters			

Name last modified in: V1.0.0	FimInhFunctionRef		
Description			
Multiplicity	1		
Type	Reference to FimFid		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency **No Included Containers**			

FimInhChoiceDemRef

SWS Item			
Container Name	FimInhChoiceDemRef		
Description			
Configuration Parameters			

Name last modified in: V1.0.0	FimInhEventRef		
Description			
Multiplicity	1		
Type	Symbolic name reference to DemEventParameter		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency **No Included Containers**			

FimInhChoiceSumRef

SWS Item			
Container Name	FimInhChoiceSumRef		
Description			
Configuration Parameters			
Name last modified in: V1.0.0	FimInhSumRef		
Description			
Multiplicity	1		
Type	Symbolic name reference to FimSummaryEventId		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

FimSummaryEventId

SWS Item	[FIM037]		
Container Name	FimSummaryEventId {FimSummaryEventId}		
Description	The summarized EventId definition record defines the existence of a summarized event with a specific name. This summarized event can be referenced in the EventSummary (as FIM_OUTPUT_SUMMARIZED_EVENT) and Inhibition configuration (as FIM_INH_EVENT_ID).		
Configuration Parameters			
Name last modified in: V1.1.0	FimEventSumId {FIM_FUNCTION_ID} [FIM037]		
Description	The summarized EventId definition record defines the existence of a summarized event with a specific name. This summarized event can be referenced in the EventSummary (as FIM_OUTPUT_SUMMARIZED_EVENT) and Inhibition configuration (as FIM_INH_EVENT_ID).		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

5.3.2 DEM - Diagnostics Event Manager

The Diagnostic Event Manager DEM is a sub-component, like the Diagnostic Communication Manager (DCM) and Function Inhibition Manager (FIM) of the diagnostic module within AUTOSAR. It is responsible for processing and storing diagnostic events (errors) and associated FreezeFrame data. Further, the DEM provides fault information to the DCM (e.g. read all stored DTCs from the event memory). The DEM offers interfaces to the application layer, the DCM and the FIM. Optional filter services are defined.

The following sections descibe the configuration parameters of DEM in details. Further details may be obtained from [9].

5.3.2.1 DemConfiguration

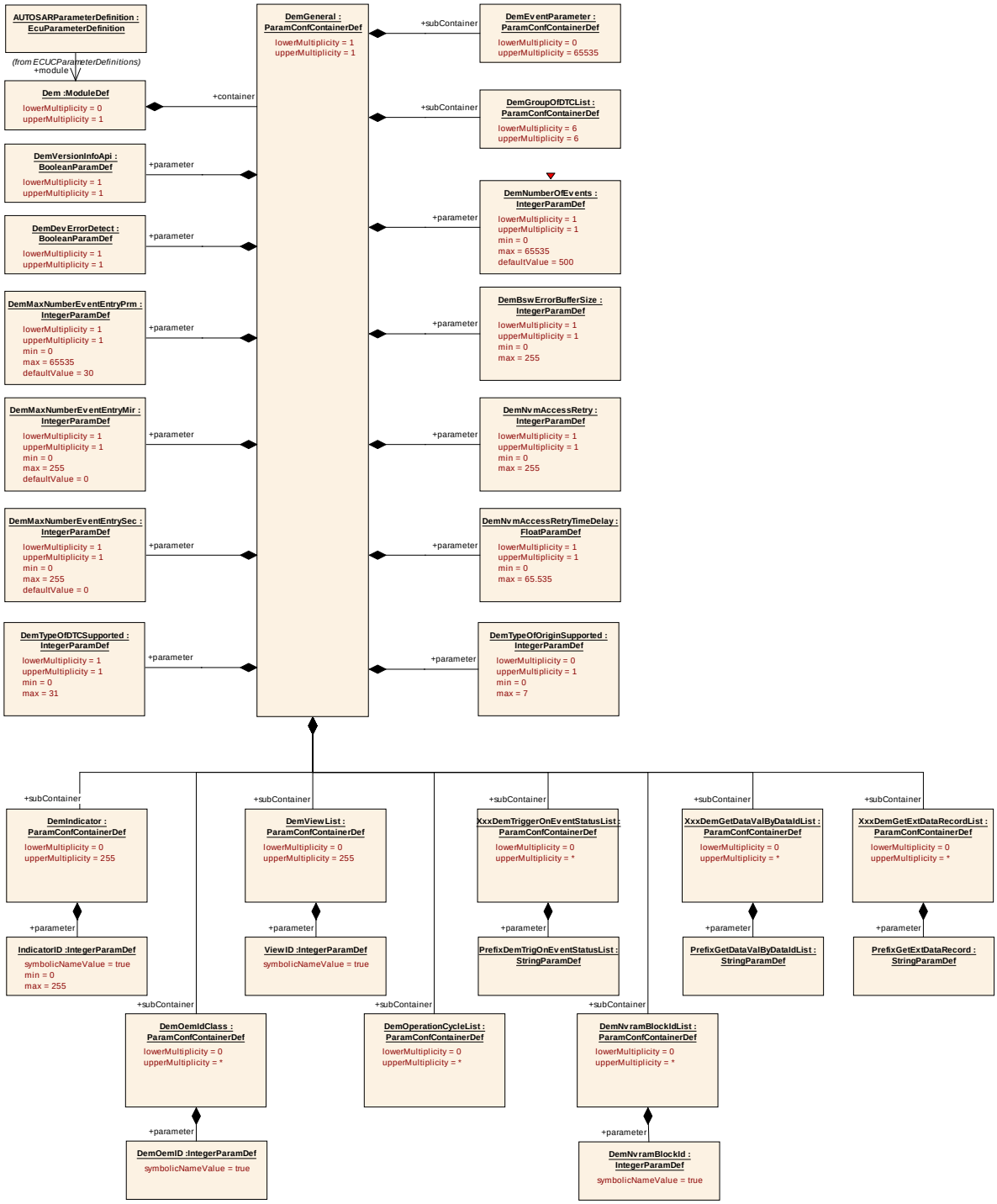


Figure 5.26: DEM - Diagnostic Event Manager

Module Name	Dem	
Module Description	Configuration of the Dem (Diagnostic Event Manager) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
DemGeneral	1	

DemGeneral

SWS Item	[Dem128]
Container Name	DemGeneral {DemConfiguation}
Description	
Configuration Parameters	

Name last modified in: V1.0.0	DemBswErrorBufferSize {DEM_BSW_ERROR_BUFFER_SIZE} [Dem128]		
Description	Maximum number of elements in buffer for handling of BSW errors (ref. to Dem107).		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	DemDevErrorDetect {DEM_DEV_ERROR_DETECT} [Dem128]		
Description	Activate/Deactivate the Development Error Detection and Notification.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	DemMaxNumberEventEntryMir {DEM_MAX_NUMBER_EVENT_ENTRY_MIR} [Dem128]		
Description	Maximum number of events which can be stored in the mirror memory (typically up to 30) (ref. to example ch. 7)		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	DemMaxNumberEventEntryPrm {DEM_MAX_NUMBER_EVENT_ENTRY_PRM} [Dem128]		
Description	Maximum number of events which can be stored in the primary memory (typically up to 20) (ref. to example ch. 7)		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	DemMaxNumberEventEntrySec {DEM_MAX_NUMBER_EVENT_ENTRY_SEC} [Dem128]		
Description	Maximum number of events which can be stored in the secondary memory (typically up to 30) (ref. to example ch. 7)		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	DemNumberOfEvents {DEM_NUMBER_OF_EVENTS} [Dem128]		
Description	Maximum of defined events which are present in the system (typically 500).		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	DemNvmAccessRetry {DEM_NVM_ACCESS_RETRY} [Dem128]		
Description	Maximum number of retries to access NVRAM Manager.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	DemNvmAccessRetryTimeDelay {DEM_NVM_ACCESS_RETRY_TIME_DELAY} [Dem128]		
Description	Time between to retries to NVRAM Manager in case of access failure.		
Multiplicity	1		
Type	FloatParamDef		
Range	0 .. 65.535		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	DemTypeOfDTCSupported {DEM_TYPE_OF_DTC_SUPPORTED} [Dem128]		
Description	DEM_TYPE_OF_DTC_SUPPORTED is defined in the DEM SWS as uint8. However, it is used in a way similar to an enum. Bit 0: 2 byte ISO15031-6 DTC Bit 1: 3 byte ISO14229-1 DTC Bit 2: Customer specific DTC Bit 3: SAEJ1939 Bit 4: WWH-OBD-format Set the according Bit to '1' means DTC format is supported. Combination of different DTC formats is possible (e.g. ISO 15031-6 and ISO14229-1 is coded by 0x0011b).		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 31		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	DemTypeOfOriginSupported {DEM_TYPE_OF_ORIGIN_SUPPORTED} [Dem128]		
Description	Definition of origins available in ECU. Bit 0: DEM_DTC_ORIGIN_PRIMARY_MEMORY Bit 1: DEM_DTC_ORIGIN_SECONDARY_MEMORY Bit 2: DEM_DTC_ORIGIN_MIRROR_MEMORY Set the according Bit to '1' means origin is available. Comination of different origins is possible (e.g. DEM_DTC_ORIGIN_MIRROR_MEMORY and DEM_DTC_ORIGIN_PRIMARY_MEMORY supported is coded by 0x0101b) This parameter is optional, because multiple origins are not supported by all automotive manufacturer.		
Multiplicity	0..1		
Type	IntegerParamDef		
Range	0 .. 7		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	DemVersionInfoApi {DEM_VERSION_INFO_API} [Dem128]		
Description	Activate/Deactivate the version information API.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
DemEventParameter	0..65535	
DemGroupOfDTCList	6..6	
DemIndicator	0..255	
DemNvramBlockIdList	0..*	
DemOemIdClass	0..*	
DemOperationCycleList	0..*	
DemViewList	0..255	
XxxDemGetDataValBy DataIdList	0..*	
XxxDemGetExtData RecordList	0..*	
XxxDemTriggerOnEvent StatusList	0..*	

DemIndicator

SWS Item	[Dem129]
Container Name	DemIndicator {IndicatorList}
Description	This container contains the configuaration (parameters) for Indicators. Note hat this container definition does not explicitly define a symbolic name parameter. Instead, the short name of the container will be used in the Ecu Configuration Description to specify the symbolic name of the INDICATOR_NAME.
Configuration Parameters	

Name last modified in: V1.0.0	IndicatorID {IndicatorID} [Dem129]		
Description	Unique Identifier of a Indicator		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

DemViewList

SWS Item	[Dem138]
Container Name	DemViewList {IndicatorList}
Description	This container contains the configuaration (parameters) for Views. Note hat this container definition does not explicitly define a symbolic name parameter. Instead, the short name of the container will be used in the Ecu Configuration Description to specify the symbolic name of the VIEW_NAME.
Configuration Parameters	

Name last modified in: V1.0.0	ViewID {ViewID} [Dem138]		
Description	Unique Identifier of a View		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

XxxDemTriggerOnEventStatusList

SWS Item	[Dem139]
Container Name	XxxDemTriggerOnEventStatusList {Xxx_DemTriggerOnEventStatus List}
Description	This container contains the configuaration (parameters) for DemTriggerOnEventStatus functions.
Configuration Parameters	

Name last modified in: V1.0.0	PrefixDemTrigOnEventStatusList {PREFIX_DemTriggerOnEventStatus List} [Dem139]		
Description	Unique prefix of a Xxx_DemTriggerOnEventStatus function. The prefix shall be replace the Xxx in function name. The prefixes shall be matching to available SC_C and BSW.		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

XxxDemGetExtDataRecordList

SWS Item	[Dem139]
Container Name	XxxDemGetExtDataRecordList {Xxx_DemGetExtendedDataRecord List}
Description	This container contains the configuaration (parameters) for GetDataValueByDataIdentifier functions.
Configuration Parameters	

Name last modified in: V1.1.0	PrefixGetExtDataRecord {PREFIX_GetExtendedDataRecord} [Dem139]		
Description	Unique prefix of a Xxx_DemTriggerOnEventStatus function. The prefix shall be replace the Xxx in function name. The prefixes shall be matching to available SC_C and BSW.		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

XxxDemGetDataValByDataIdList

SWS Item	[Dem139]		
Container Name	XxxDemGetDataValByDataIdList {Xxx_DemGetDataValueByDataIdentifierList}		
Description	This container contains the configuaration (parameters) for GetDataValueByDataIdentifier functions.		
Configuration Parameters			
Name last modified in: V1.1.0	PrefixGetDataValByDataIdList {PREFIX_GetDataValueByDataIdentifierList} [Dem139]		
Description	Unique prefix of a Xxx_DemTriggerOnEventStatus function. The prefix shall be replace the Xxx in function name. The prefixes shall be matching to available SC_C and BSW.		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

DemOemIdClass

SWS Item	[Dem141]		
Container Name	DemOemIdClass {OEMIdClass}		
Description	This container contains the configuaration (parameters) for OEMIdClass. Note hat this container definition does not explicitly define a symbolic name parameter. Instead, the short name of the container will be used in the Ecu Configuration Description to specify the symbolic name of the VALUE_NAME.		
Configuration Parameters			
Name last modified in: V1.0.0	DemOemID {OemID} [Dem141]		
Description	Defines a unique ID of a data value.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	pre-compile time variant
	Link time	–	
	Post-build time	X	pre-compile and post-build time variant
Scope / Dependency			
No Included Containers			

DemOperationCycleList

SWS Item	[Dem142]
Container Name	DemOperationCycleList {Dem_OperationCycleList}
Description	Note hat this container definition does not explicitly define a symbolic name parameter. Instead, the short name of the container will be used in the Ecu Configuration Description to specify the symbolic name of the OperationCycleName.
Configuration Parameters **No Included Containers**	

DemNvramBlockIdList

SWS Item	[Dem147]
Container Name	DemNvramBlockIdList {NVRAMBlockIDList}
Description	This container contains the configuaration (parameters) for Dem_OperationCycleList. Note hat this container definition does not explicitly define a symbolic name parameter. Instead, the short name of the container will be used in the Ecu Configuration Description to specify the symbolic name of the VALUE_NAME.
Configuration Parameters	

Name last modified in: V1.0.0	DemNvramBlockId {BlockID} [Dem147]		
Description	Defines a unique ID of a data value.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency **No Included Containers**			

5.3.2.1.1 DemEventParameter

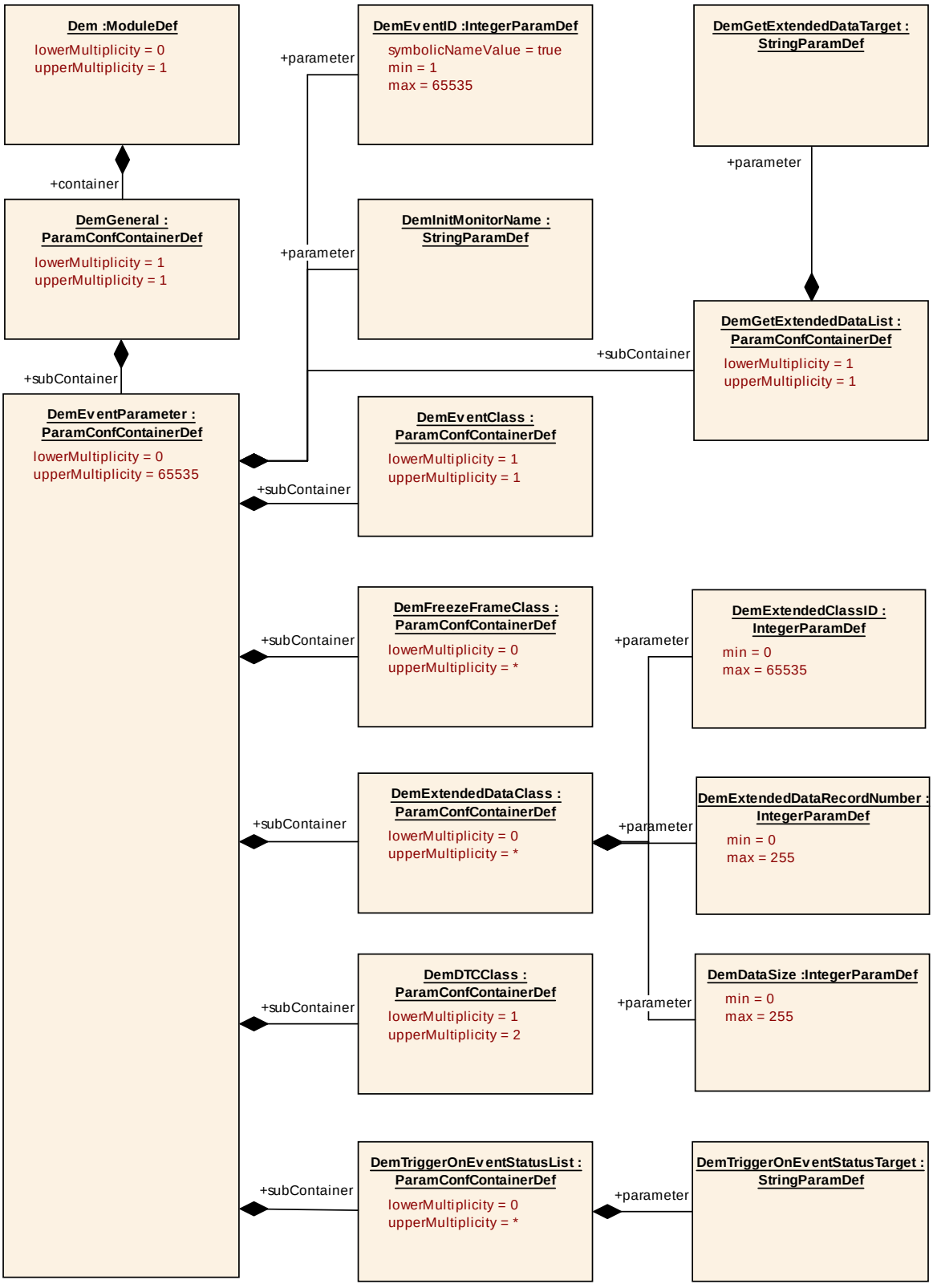


Figure 5.27: DEM - Event Manager Parameter

DemEventParameter

SWS Item	[Dem130]
Container Name	DemEventParameter {EventParameter}
Description	This container contains the configuaration (parameters) for events. Note that this container definition does not explicitly define a symbolic name parameter. Instead, the short name of the container will be used in the Ecu Configuration Description to specify the symbolic name of the DEM_EVENT_NAME.
Configuration Parameters	

Name last modified in: V1.0.0	DemEventID {EventId} [Dem130]		
Description	Unique identifier of an event 1..255 for small PDUs with <255 different events and a limited RAM 1..65535 for other PDUs The events should be sequential ordered beginning with 1 and no gaps in between.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Range	1 .. 65535		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	DemInitMonitorName {InitMonitorName} [Dem130]		
Description	Monitor function which has to be initialized for the event (ref. to Xxx_DemInitMonitor(EventId)		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Included Containers

Container Name	Multiplicity	Scope / Depedency
DemDTCClass	1..2	
DemEventClass	1	
DemExtendedDataClass	0..*	
DemFreezeFrameClass	0..*	
DemGetExtendedData List	1	
DemTriggerOnEvent StatusList	0..*	

DemExtendedDataClass

SWS Item	[Dem135]		
Container Name	DemExtendedDataClass {ExtendedDataClass}		
Description	This container contains the configuration for ExtendedClass.		
Configuration Parameters			

Name last modified in: V1.0.0	DemDataSize {ExtendedDataRecordNumber} [Dem135]		
Description	Defines the size of the extended Data Record in Bytes.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	pre-compile time variant
	Link time	–	
	Post-build time	X	pre-compile and post-build time variant
Scope / Dependency			

Name last modified in: V1.0.0	DemExtendedClassID {ExtendedDataClass} [Dem135]		
Description	Unique identifier of an extended data class.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	pre-compile time variant
	Link time	–	
	Post-build time	X	pre-compile and post-build time variant
Scope / Dependency			

Name last modified in: V1.0.0	DemExtendedDataRecordNumber {ExtendedDataRecordNumber} [Dem135]		
Description	Unique identifier of an extended data class.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	pre-compile time variant
	Link time	–	
	Post-build time	X	pre-compile and post-build time variant
Scope / Dependency			
No Included Containers			

DemGetExtendedDataList

SWS Item	[Dem131]
Container Name	DemGetExtendedDataList {EventClass}
Description	This container contains the configuaration for GetExtendedData functions
Configuration Parameters	

Name last modified in: V1.0.0	DemGetExtendedDataTarget {DemTriggerOnEventStatusTarget} [Dem140]		
Description	Real name of the Xxx_GetExtendedData function.		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

DemTriggerOnEventStatusList

SWS Item	[Dem140]
Container Name	DemTriggerOnEventStatusList {TriggerOnEventStatusList}
Description	This container contains the configuration for TriggerOnEvent functions.
Configuration Parameters	

Name last modified in: V1.0.0	DemTriggerOnEventStatusTarget {DemTriggerOnEventStatusTarget} [Dem140]		
Description	Real name of Xxx_DemTriggerOnEventStatus function. The possible selection depends on Xxx_DemTriggerOnEventStatusList.		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

5.3.2.1.2 DemEventClass

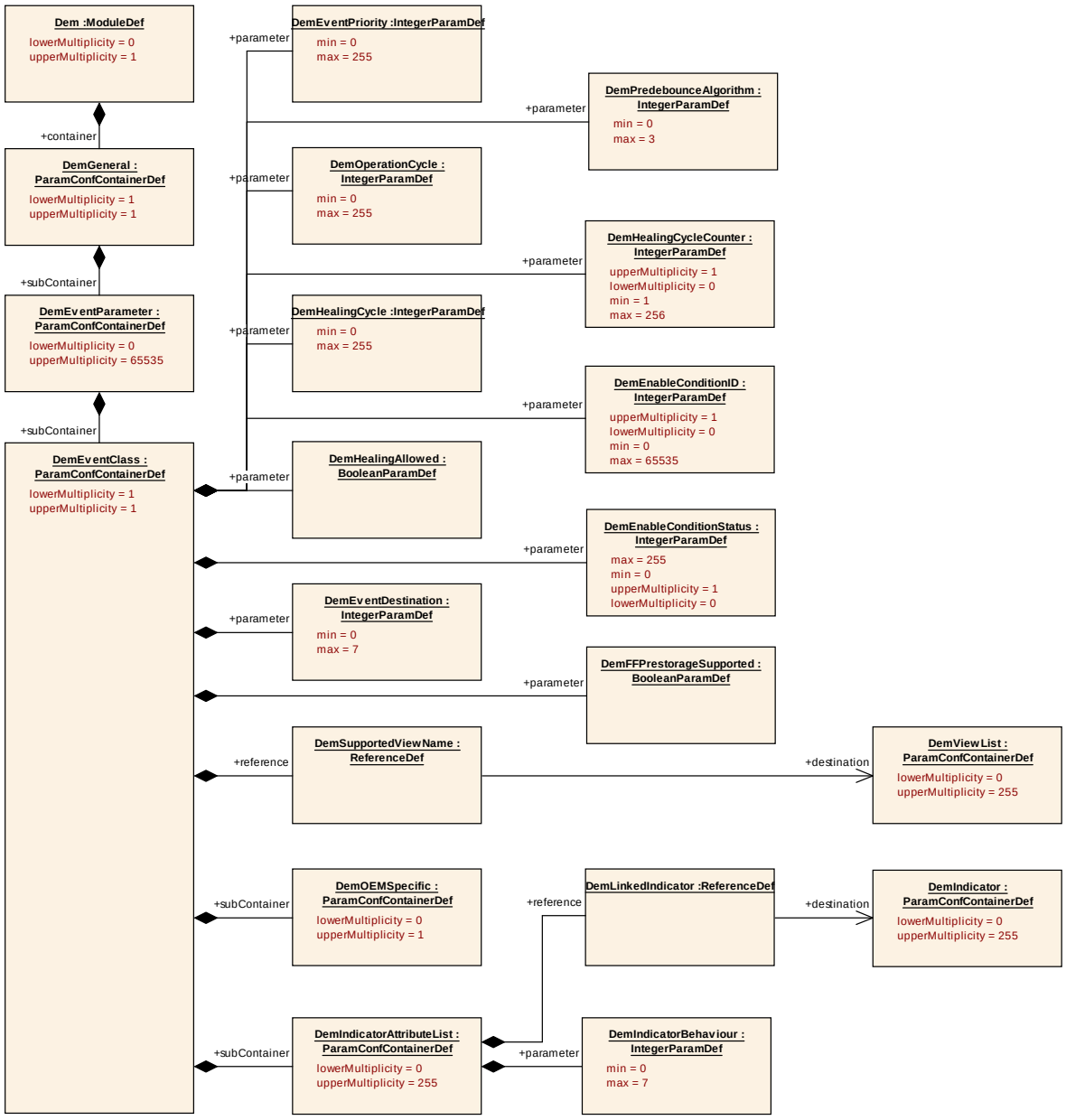


Figure 5.28: DEM - Event CLass

DemEventClass

SWS Item	[Dem131]
Container Name	DemEventClass {EventClass}
Description	This container contains the configuaration (parameters) for EventClass
Configuration Parameters	

Name last modified in: V1.1.0	DemEnableConditionID {EnableConditionID} [Dem131]		
Description	Defines a condition ID.		
Multiplicity	0..1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	pre-compile time variant
	Link time	–	
	Post-build time	X	pre-compile and post-build time variant
Scope / Dependency			

Name last modified in: V1.0.0	DemEnableConditionStatus {EnableConditionStatus} [Dem131]		
Description	Defines a status for enable or disable of storage of a event.		
Multiplicity	0..1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	pre-compile time variant
	Link time	–	
	Post-build time	X	pre-compile and post-build time variant
Scope / Dependency			

Name last modified in: V1.0.0	DemEventDestination {EventDestination} [Dem131]		
Description	Dem104 identification of the destination of an event/Dem_DTCOriginType.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 7		
Configuration Class	**Pre-compile time**	X	pre-compile time variant
	Link time	–	
	Post-build time	X	pre-compile and post-build time variant
Scope / Dependency			

Name last modified in: V1.0.0	DemEventPriority {EventPriority} [Dem131]		
Description	Priority of an event, in view of full event buffer (ref. to Dem104).		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	pre-compile time variant
	Link time	–	
	Post-build time	X	pre-compile and post-build time variant
Scope / Dependency			

Name last modified in: V1.1.0	DemFFPrestorageSupported {FF_Prestorage_Supported} [Dem131]		
Description	If this parameter is set to true, then the Prestorage of freeze frames is supported by the assigned event. This parameter is useful to calculate the buffer size.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	pre-compile time variant
	Link time	–	
	Post-build time	X	pre-compile and post-build time variant
Scope / Dependency			

Name last modified in: V1.0.0	DemHealingAllowed {HealingAllowed} [Dem131]		
Description	(Dem104) general switch to allow healing/unlearning or not.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	pre-compile time variant
	Link time	–	
	Post-build time	X	pre-compile and post-build time variant
Scope / Dependency			

Name last modified in: V1.0.0	DemHealingCycle {HealingCycle} [Dem131]		
Description	cycles needed to heal/erase event (ref. Dem104). This parameter is optional (depends on OEM).		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	pre-compile time variant
	Link time	–	
	Post-build time	X	pre-compile and post-build time variant
Scope / Dependency			

Name last modified in: V1.1.0	DemHealingCycleCounter {HealingCycleCounter} [Dem131]		
Description	cycles needed to heal/erase event (ref. Dem104). This parameter is optional (depends on OEM).		
Multiplicity	0..1		
Type	IntegerParamDef		
Range	1 .. 256		
Configuration Class	**Pre-compile time**	X	pre-compile time variant
	Link time	–	
	Post-build time	X	pre-compile and post-build time variant
Scope / Dependency			

Name last modified in: V1.0.0	DemOperationCycle {OperationCycle} [Dem131]		
Description	Kind of operation cycle for the event (e.g. power cycle, driving cycle, ...)		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	pre-compile time variant
	Link time	–	
	Post-build time	X	pre-compile and post-build time variant
Scope / Dependency			

Name last modified in: V1.1.0	DemPredebounceAlgorithm {PredebounceAlgorithm} [Dem131]		
Description	Selects a Predebounce Algorithm		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 3		
Configuration Class	**Pre-compile time**	X	pre-compile time variant
	Link time	–	
	Post-build time	X	pre-compile and post-build time variant
Scope / Dependency			

Name last modified in: V1.0.0	DemSupportedViewName {SUPPORTED_VIEW_NAME} [Dem130]		
Description	view name of supported view A view describes a functional group like a wiper system or a window lifter for the access of corresponding DTCs and related data. SUPPORTED_VIEW_NAME selects a view in which the event is visible. Example: WIPERSYSTEM refers to functionality wiper system, WINDOWLIFTER refers to functionality window lifter, ... This parameter is optional.		
Multiplicity	1		
Type	Reference to DemViewList		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency **Included Containers**			

Container Name	**Multiplicity**	**Scope / Depedency**
DemIndicatorAttributeList	0..255	
DemOEMSpecific	0..1	

DemIndicatorAttributeList

SWS Item	[Dem133]
Container Name	DemIndicatorAttributeList {IndicatorAttributeList}
Description	This container contains the event specific configuration of Indicators.
Configuration Parameters	

Name last modified in: V1.0.0	DemIndicatorBehaviour {IndicatorBehaviour} [Dem133]		
Description	Behaviour of the linked indicator Bit 0: Indicator is active if event in status FAILED Bit 1: Indicator is passive if event in status FAILED Bit 2: Indicator is blinking if event in status FAILED		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 7		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	DemLinkedIndicator {LinkedIndicator} [Dem133]		
Description	indicator name of the used indicator		
Multiplicity	1		
Type	Reference to DemIndicator		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency **No Included Containers**			

DemOEMSpecific

SWS Item	[Dem134]
Container Name	DemOEMSpecific {OEMSpecific}
Description	This container contains the configuaration for OEM specific additional parameter.
Configuration Parameters **No Included Containers**	

5.3.2.1.3 DemDTCClass

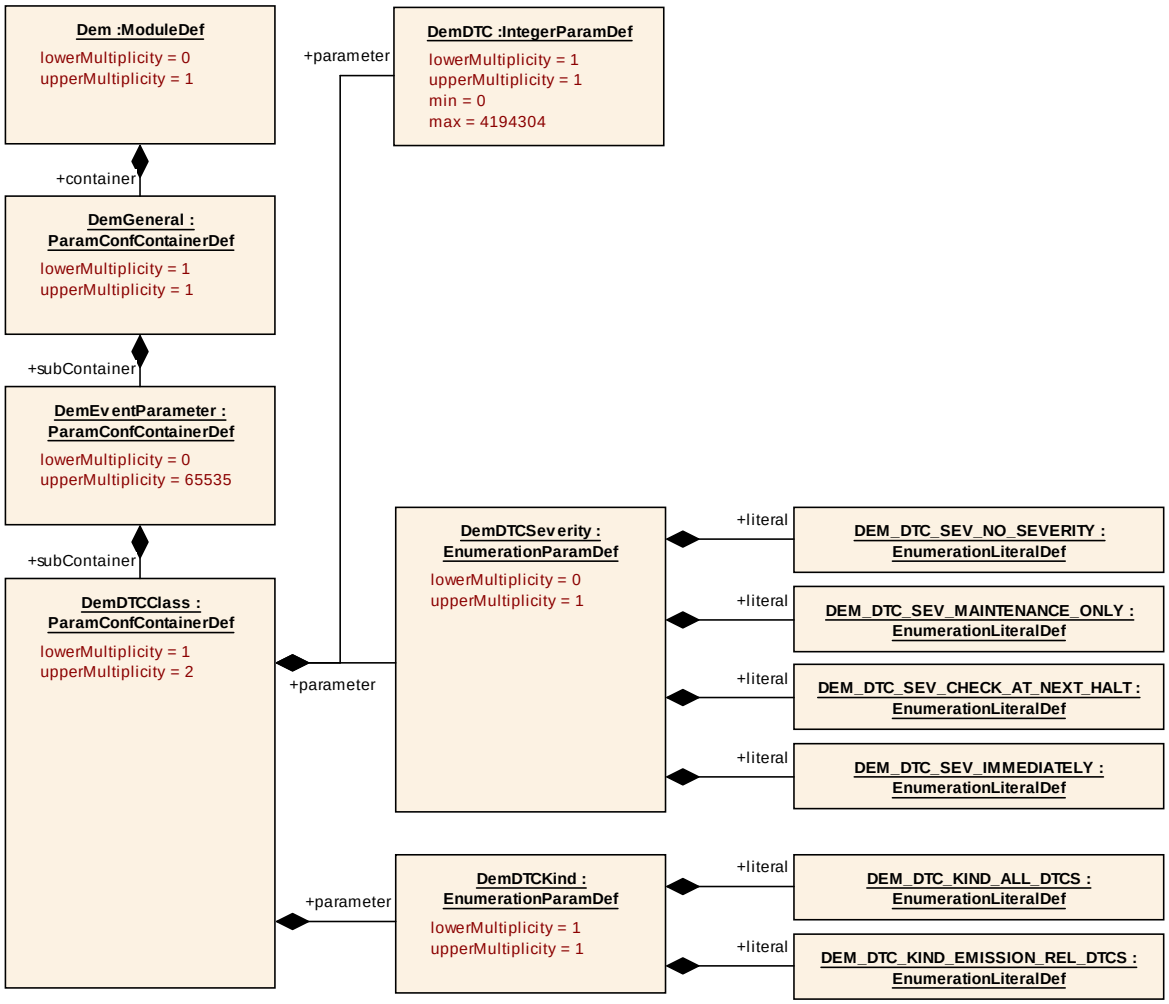


Figure 5.29: DEM - DTC Class

DemDTCClass

SWS Item	[Dem132]		
Container Name	DemDTCClass {DTCClass}		
Description	This container contains the configuration for DTCClass.		
Configuration Parameters			

Name last modified in: V1.0.0	DemDTC {DTC} [Dem132]		
Description	Diagnostic Trouble Code		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 4194304		
Configuration Class	**Pre-compile time**	X	pre-compile time variant
	Link time	–	
	Post-build time	X	pre-compile and post-build time variant
Scope / Dependency			

Name last modified in: V1.0.0	DemDTCKind {DTCKind} [Dem132]		
Description	Kind of DTC (OBD relevant or not)		
Multiplicity	1		
Type	EnumerationParamDef		
Range	DEM_DTC_KIND_ALL_DTCS		
	DEM_DTC_KIND_EMISSION_REL_DTCS		
Configuration Class	**Pre-compile time**	X	pre-compile time variant
	Link time	–	
	Post-build time	X	pre-compile and post-build time variant
Scope / Dependency			

Name last modified in: V1.0.0	DemDTCSeverity {DTCSeverity} [Dem132]		
Description	This parameter depends on automotive manufacturer and is optional.		
Multiplicity	0..1		
Type	EnumerationParamDef		
Range	DEM_DTC_SEV_CHECK_AT_NEXT_HALT		
	DEM_DTC_SEV_IMMEDIATELY		
	DEM_DTC_SEV_MAINTENANCE_ONLY		
	DEM_DTC_SEV_NO_SEVERITY		
Configuration Class	**Pre-compile time**	X	pre-compile time variant
	Link time	–	
	Post-build time	X	pre-compile and post-build time variant
Scope / Dependency			
No Included Containers			

5.3.2.1.4 DemFreezeFrameClass

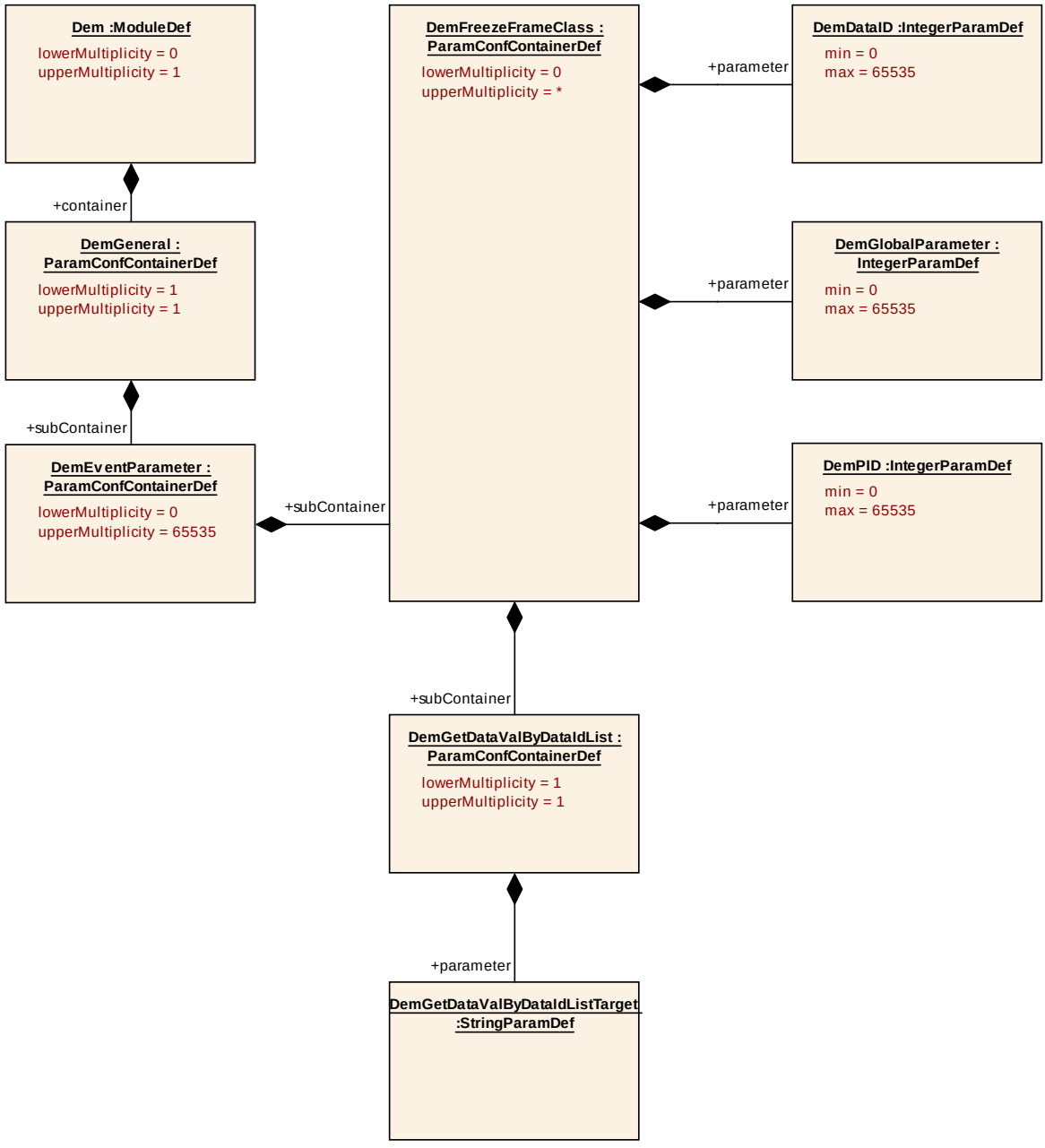


Figure 5.30: DEM - Freeze Frame Class

DemFreezeFrameClass

SWS Item	[Dem136]
Container Name	DemFreezeFrameClass {FreezeFrameClass}
Description	This container contains the configuration for FreezeFrameClass.
Configuration Parameters	

Name last modified in: V1.0.0	DemDataID {DataId} [Dem136]		
Description	For enhanced diagnostics Multiple DataIDs can be relevant per FreezeFrame. This parameter is optional!		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	pre-compile time variant
	Link time	–	
	Post-build time	X	pre-compile and post-build time variant
Scope / Dependency			

Name last modified in: V1.0.0	DemGlobalParameter {EventId} [Dem136]		
Description	Mandatory parameters for OBD and common parameter for all events		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	pre-compile time variant
	Link time	–	
	Post-build time	X	pre-compile and post-build time variant
Scope / Dependency			

Name last modified in: V1.0.0	DemPID {EventId} [Dem136]		
Description	For OBD relevant data Multiple PIDs can be relevant per Freezeframe. This parameter is optional!		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	pre-compile time variant
	Link time	–	
	Post-build time	X	pre-compile and post-build time variant
Scope / Dependency			

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
DemGetDataValByDataId List	1	

DemGetDataValByDataIdList

SWS Item	[Dem140]
Container Name	DemGetDataValByDataIdList {GetDataValueByDataIdentifierList}
Description	This container contains the configuration for GetDataValueByDataIdentifierList.
Configuration Parameters	

Name last modified in: V1.1.0	DemGetDataValByDataIdListTarget {GetDataValueByDataIdentifierList Target} [Dem140]		
Description	Real name of Xxx_GetDataValueByDataIdentifier function.		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

5.3.2.1.5 DemGroupOfDTCList

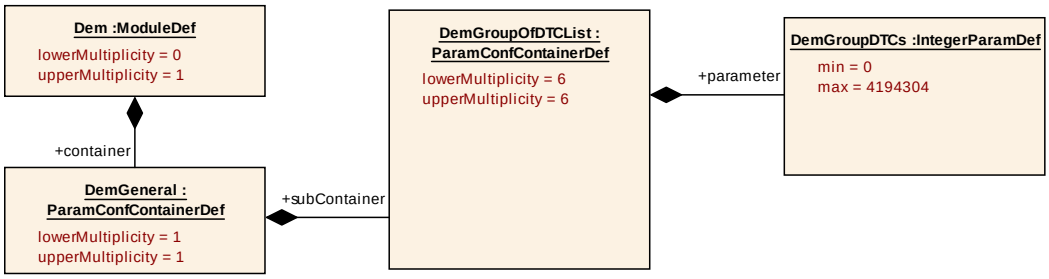


Figure 5.31: DEM - Group Of DTC List

DemGroupOfDTCList

SWS Item	[Dem137]		
Container Name	DemGroupOfDTCList {GroupOfDTCList}		
Description	This container contains the configuaration (parameters) for DTC Groups.		
Configuration Parameters			

Name last modified in: V1.0.0	DemGroupDTCs {DTCGroup} [Dem137]		
Description	Range of DTC numbers of the selected group of DTC (according to ISO14229-1[10] Annex D1)		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 4194304		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

5.3.2.1.6 DemPredebounce

REMARK: Modeling of Predebounce algorithm not finished! Additional information to SWS needed!

5.3.3 DCM - Diagnostic Communication Manager

The Diagnostic Communication Manager (DCM) provides a common API in AUTOSAR BSW for diagnostic services. DCM functionality is used by external diagnostic tools throughout the product life-cycle, including development, manufacturing and service. The DCM handles different diagnostic protocols, for legislated OBD (ISO 15031-5) and for enhanced diagnostics (ISO 14229-1). The DCM also handles the network-independent sections (OSI-Layers 5 to 7) of diagnostic sessions for ISO 15765-3 (implementation of unified diagnostic services (UDS) on CAN) and ISO 15765-4 (requirements for emissions-related systems). The DCM ensures diagnostic data flow and manages the diagnostic states, especially diagnostic sessions and security states. The DCM provides for all diagnostic service interfaces (full-set of ISO 14229-1 and ISO 15031-5) to the AUTOSAR-RTE. Additionally the DCM processes some basic diagnostic services and checks whether diagnostic service requests are supported and whether services may be executed in the current session state. Further details may be obtained from [26].

5.3.3.1 DcmConfiguration

The top level configuration container and parameters of a DCM are shown in Figure 5.32. Table `Dcm` identifies the contents in textual form. It is assumed that there may be no DCM configuration present if there is no DCM in the ECU (or no configuration of a default DCM is required). It is also possible for multiple DCMs to exist in an AUTOSAR ECU Configuration. Each would manage separate subsets of the ECU's diagnostic service population.

Within the top level configuration container are a number of sub-containers to handle aspects of the DCM functionality. There must be exactly one of each of the following in `DcmConfiguration`:

The detail in each sub-container in `DcmConfiguration` is described in following sections.

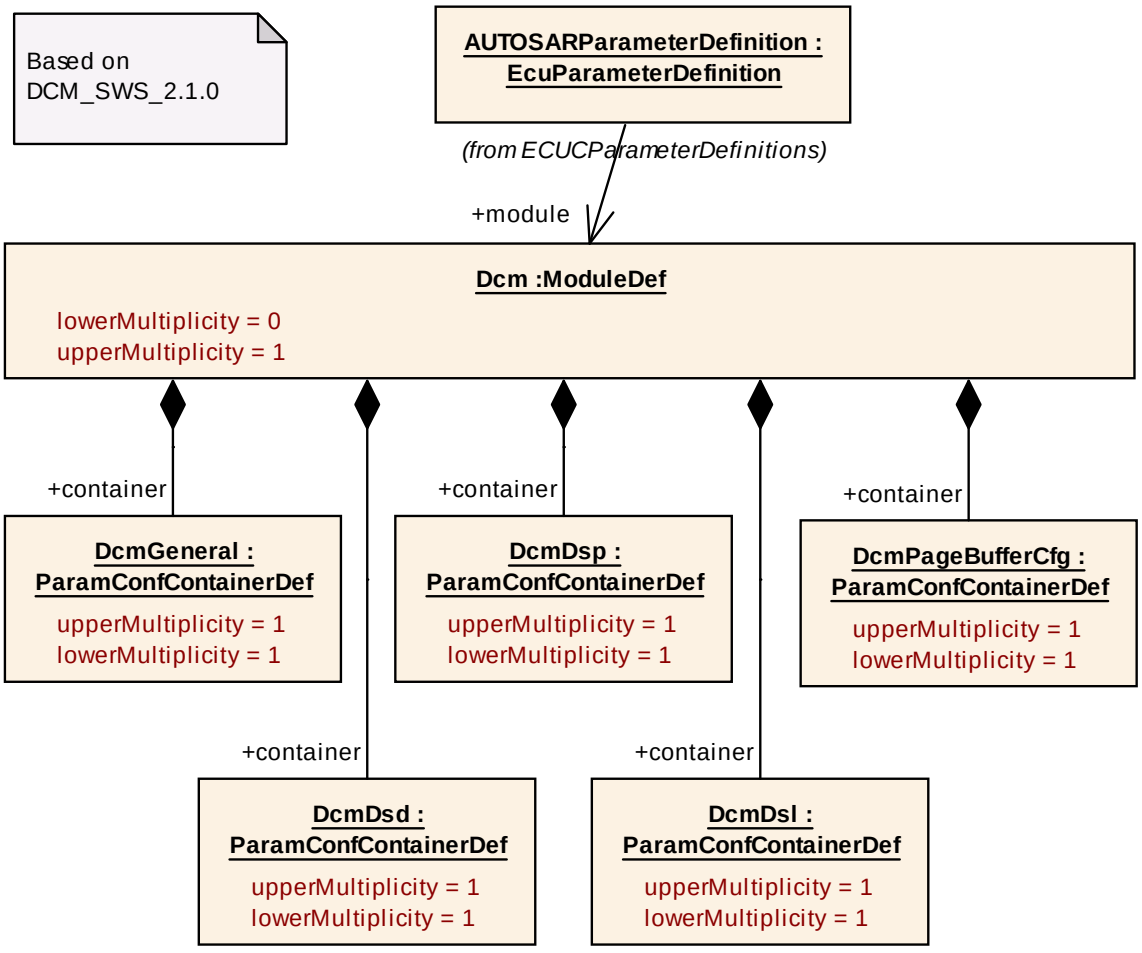


Figure 5.32: DCM - Diagnostic Communication Manager - Top Level

Module Name	Dcm	
Module Description	Configuration of the Dcm (Diagnostic Communications Manager) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
DcmDsd	1	These parameters apply to Diagnostic Service Dispatcher. All parameters for all service dispatchers are included in this one configuration container. upperMultiplicity: Multiple service dispatchers are allowed in this one configuration container. lowerMultiplicity: There must always be one service dispatcher in a DCM.

Container Name	Multiplicity	Scope / Dependency
DcmDsl	1	These parameters apply to a Diagnostic Session Layer. There may be a parameter set (DSL Configuration) per protocol. upperMultiplicity: Each DCM configuration must have exactly one DSL configuration. lowerMultiplicity: Each DCM configuration must have exactly one DSL configuration.
DcmDsp	1	These parameters apply to Diagnostic Service Processing. There will always be one set of these parameters per DCM.
DcmGeneral	1	This container contains the configuration (parameters) for Component wide parameters
DcmPageBufferCfg	1	This container contains the configuration (parameters) for Page Buffer handling

5.3.3.2 DcmGeneralConfiguration

`DcmGlobalConfiguration` holds parameters that are set for the whole of a DCM. Only one set of these parameters is required per instance of DCM.

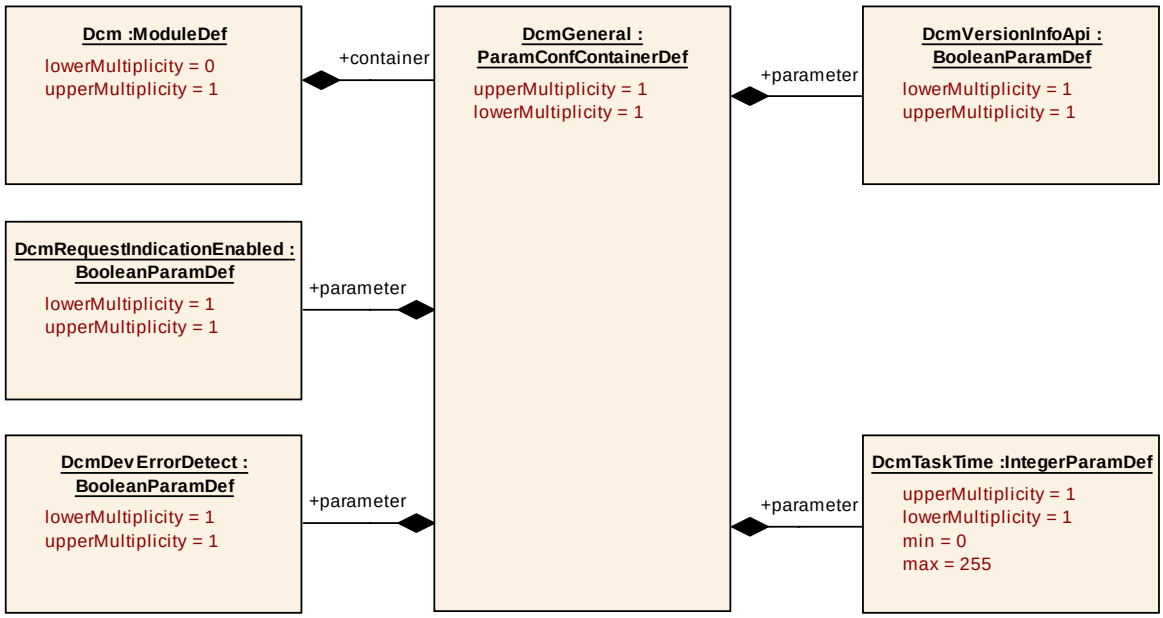


Figure 5.33: DCM Global Settings

DcmGeneral

SWS Item	[DCM075, DCM076]		
Container Name	DcmGeneral {COMPONENT_WIDE_PARAMETERS}		
Description	This container contains the configuration (parameters) for Component wide parameters		
Configuration Parameters			

Name last modified in: V1.0.0	DcmDevErrorDetect {DCM_DEV_ERROR_DETECT} [DCM075, DCM076]		
Description	Preprocessor switch to enable or disable the Development Error Detection (DET) mechanism.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	DcmRequestIndicationEnabled {DCM_REQUEST_INDICATION_ENABLED} [DCM075, DCM076]		
Description	Allows to enable or disable the requested indication mechanism.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	DcmTaskTime {DCM_TASK_TIME} [DCM075, DCM076]		
Description	Allow to configure the time for the periodic cyclic task (in ms). Please note: This configuration value shall be equal to the value in the ScheduleManager module.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	DcmVersionInfoApi {DCM_VERSION_INFO_API} [DCM075, DCM076]		
Description	Preprocessor switch to enable or disable the output Version info of the functionality.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

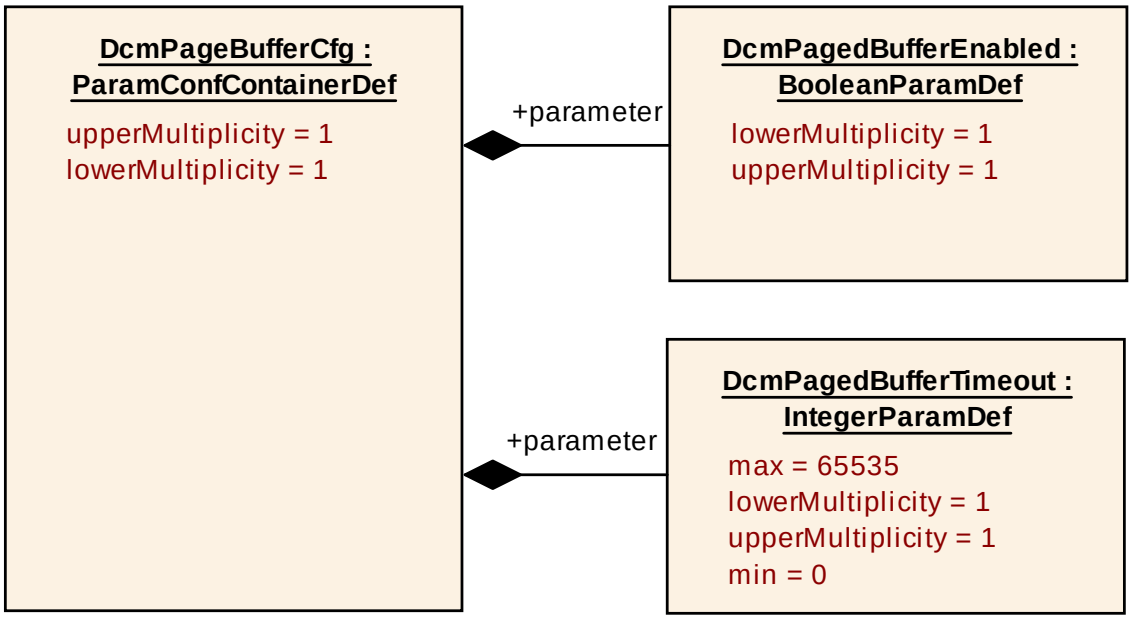


Figure 5.34: DCM Page Buffer Config

DcmPageBufferCfg

SWS Item	[DCM068]		
Container Name	DcmPageBufferCfg {PAGE_BUFFER_CFG}		
Description	This container contains the configuration (parameters) for Page Buffer handling		
Configuration Parameters			
Name last modified in: V1.0.0	DcmPagedBufferEnabled {DCM_PAGEDBUFFER_ENABLED} [DCM068]		
Description	Allow to enable or disable the Page buffer mechanism.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
Name last modified in: V1.0.0	DcmPagedBufferTimeout {DCM_PAGEDBUFFER_TIMEOUT} [DCM068]		
Description	Allow to configure the Timeout (in ms) towards the application for filling the next page.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	–	
Scope / Dependency			
No Included Containers			

5.3.3.3 DcmDslConfiguration

The Diagnostic Session Layer (DSL) controls data flow of diagnostic requests and responses. It manages transmit and receive flows for each protocol (e.g. UDS, OBD) and links to message buffers. The DSL also supervises and guarantees diagnostic protocol timing, particularly for Response On Events and for Periodic Transmissions. There is one DSL Configuration per DCM.

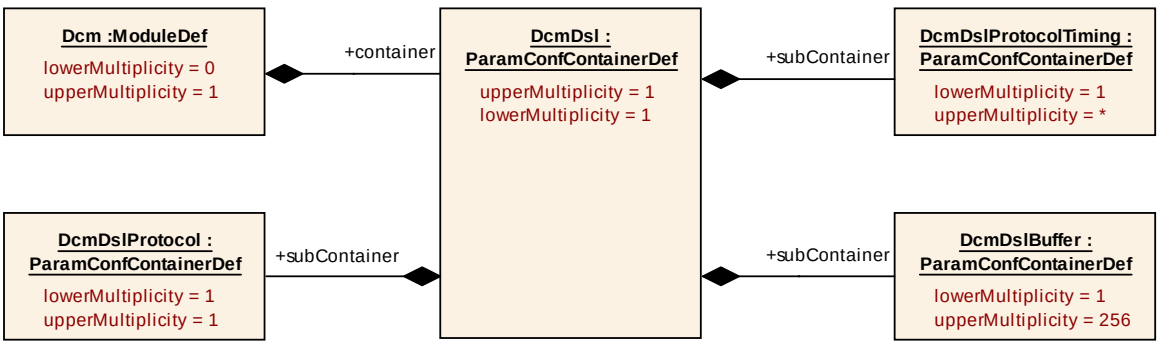


Figure 5.35: DCM DSL - Diagnostic Session Layer

DcmDsl

SWS Item		
Container Name	DcmDsl	
Description	These parameters apply to a Diagnostic Session Layer. There may be a parameter set (DSL Configuration) per protocol. upperMultiplicity: Each DCM configuration must have exactly one DSL configuration. lowerMultiplicity: Each DCM configuration must have exactly one DSL configuration.	
Configuration Parameters Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
DcmDslBuffer	1..256	
DcmDslPeriodic Transmission	1	
DcmDslProtocol	1	
DcmDslProtocolTiming	1..*	
DcmDslResponseOn Event	1	

5.3.3.3.1 DcmDslBufferConfiguration

A `DcmDslBufferConfiguration` holds parameters to configure a diagnostic buffer used by the DSL. One set of these parameters is required per DCM buffer. These buffers are referenced by DSL Protocol definitions.

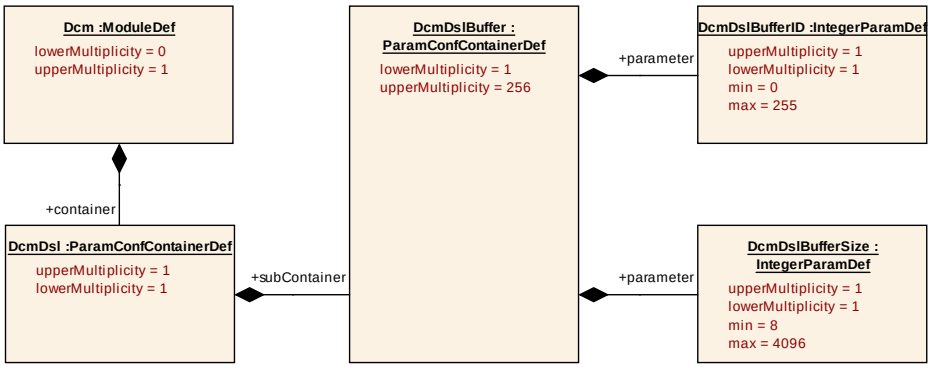


Figure 5.36: `DCM DSL` Buffer

DcmDslBuffer

SWS Item	[DCM032]
Container Name	DcmDslBuffer {DIAGNOSTIC_BUFFER_CFG}
Description	This container contains the configuration (parameters) for the diagnostic buffer.
Configuration Parameters	

Name last modified in: V1.0.0	DcmDslBufferID {DCM_BUFFER_ID} [DCM032]		
Description	Identifier of Buffer.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	L	–
Scope / Dependency			

Name last modified in: V1.0.0	DcmDslBufferSize {DCM_BUFFER_SIZE} [DCM032]		
Description	Size of Diagnostic Buffer (in Bytes) For a linear buffer: size of the buffer shall be as large as the longest message (request or response) For a paged buffer: size has impacts on the application performance Please note: max. range is the valid range for a CAN network. We assume a FlexRay (or other networks) implementation will work with this range (and the page buffer mechanism) without any problems.		
Multiplicity	1		
Type	IntegerParamDef		
Range	8 .. 4096		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

5.3.3.3.2 DcmDslProtocolTimingConfiguration

A `DcmDslProtocolTimingConfiguration` holds a set of parameters that together define the timing performance of a diagnostic protocol. One set of these parameters is required for each distinct DSL protocol timing behaviour. There must always be one timing configuration and there can be multiple configurations defined in a DCM DSL. The upper multiplicity is unlimited as there are virtually ”infinite” different structure names possible. It is possible to define timing configurations that are not used in the DSL.

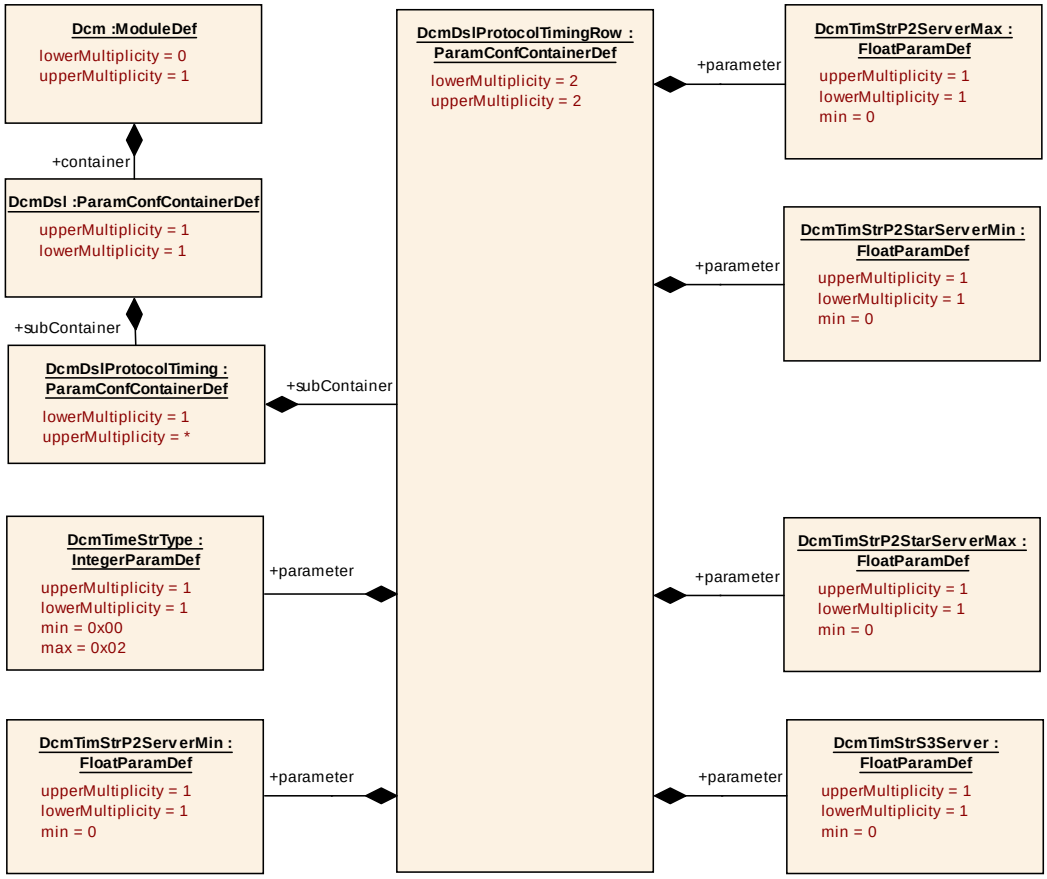


Figure 5.37: `DCM DSL` Protocol Timing

The protocol timing configuration structure consists of a name string and a set of five times (in milliseconds). A single value for each must be defined to complete a timing configuration.

DcmDslProtocolTiming

SWS Item	[Dcm031]	
Container Name	DcmDslProtocolTiming {PROTOCOL_TIMING_STRUCTURE}	
Description	This container contains the configuration (parameters) for Protocol timing. Please notice that per protocol 2 Time tables shall be configured (DCM_DEFAULT and DCM_LIMITS) . This container contains rows of PROTOCOL_TIMING_STRUCTURE_ROW.	
Configuration Parameters Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
DcmDslProtocolTiming Row	2..2	

DcmDslProtocolTimingRow

SWS Item	[Dcm031]		
Container Name	DcmDslProtocolTimingRow {PROTOCOL_TIMING_STRUCTURE_ROW}		
Description	Definition of a single row of configuration for protocol timing		
Configuration Parameters			

Name last modified in: V1.0.0	DcmTimStrP2ServerMax {DCM_TIMSTR_P2SERVER_MAX} [Dcm031]		
Description	This is the value for P2ServerMin in milliseconds. This is defined in the AUTOSAR SWS for DCM as a uint16. The AUTOSAR configuration standard is to use SI units, so this parameter is defined as a float value in seconds. DCM configuration tools must convert this float value to the appropriate value format for the use in the software implementation of DCM.		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	L	–
Scope / Dependency			

Name last modified in: V1.0.0	DcmTimStrP2ServerMin {DCM_TIMSTR_P2SERVER_MIN} [Dcm031]		
Description	This is the value for P2ServerMin in milliseconds. This is defined in the AUTOSAR SWS for DCM as a uint16. The AUTOSAR configuration standard is to use SI units, so this parameter is defined as a float value in seconds. DCM configuration tools must convert this float value to the appropriate value format for the use in the software implementation of DCM.		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	L	–
Scope / Dependency			

Name last modified in: V1.0.0	DcmTimStrP2StarServerMax {DCM_TIMSTR_P2STARSERVER_MAX} [Dcm031]		
Description	This is the value for P2ServerMin in milliseconds. This is defined in the AUTOSAR SWS for DCM as a uint16. The AUTOSAR configuration standard is to use SI units, so this parameter is defined as a float value in seconds. DCM configuration tools must convert this float value to the appropriate value format for the use in the software implementation of DCM.		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	L	–
Scope / Dependency			

Name last modified in: V1.0.0	DcmTimStrP2StarServerMin {DCM_TIMSTR_P2STARSERVER_MIN} [Dcm031]		
Description	This is the value for P2ServerMin in milliseconds. This is defined in the AUTOSAR SWS for DCM as a uint16. The AUTOSAR configuration standard is to use SI units, so this parameter is defined as a float value in seconds. DCM configuration tools must convert this float value to the appropriate value format for the use in the software implementation of DCM.		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	L	–
Scope / Dependency			

Name last modified in: V1.0.0	DcmTimStrS3Server {DCM_TIMSTR_S3SERVER} [Dcm031]		
Description	This is the value for P2ServerMin in milliseconds. This is defined in the AUTOSAR SWS for DCM as a uint16. The AUTOSAR configuration standard is to use SI units, so this parameter is defined as a float value in seconds. DCM configuration tools must convert this float value to the appropriate value format for the use in the software implementation of DCM.		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	X	All Variants
	Post-build time	L	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	DcmTimeStrType {DCM_TIMESTR_TYPE} [DCM031]		
Description	Type of the timing structure Note: The Dcm_TimerModeType DCM_CURRENT is not configurable		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 2		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	L	–
Scope / Dependency			
No Included Containers			

5.3.3.3.3 DcmDslPeriodicTransmissionConfiguration

`DcmDslPeriodicTransmissionConfiguration` defines how the DSL protocol (UDS, OBD, ...) will behave for periodic transmission mode. It is assumed that this will always be defined once as part of the DSL protocol configuration, even if only to indicate that periodic transmissions are not enabled.

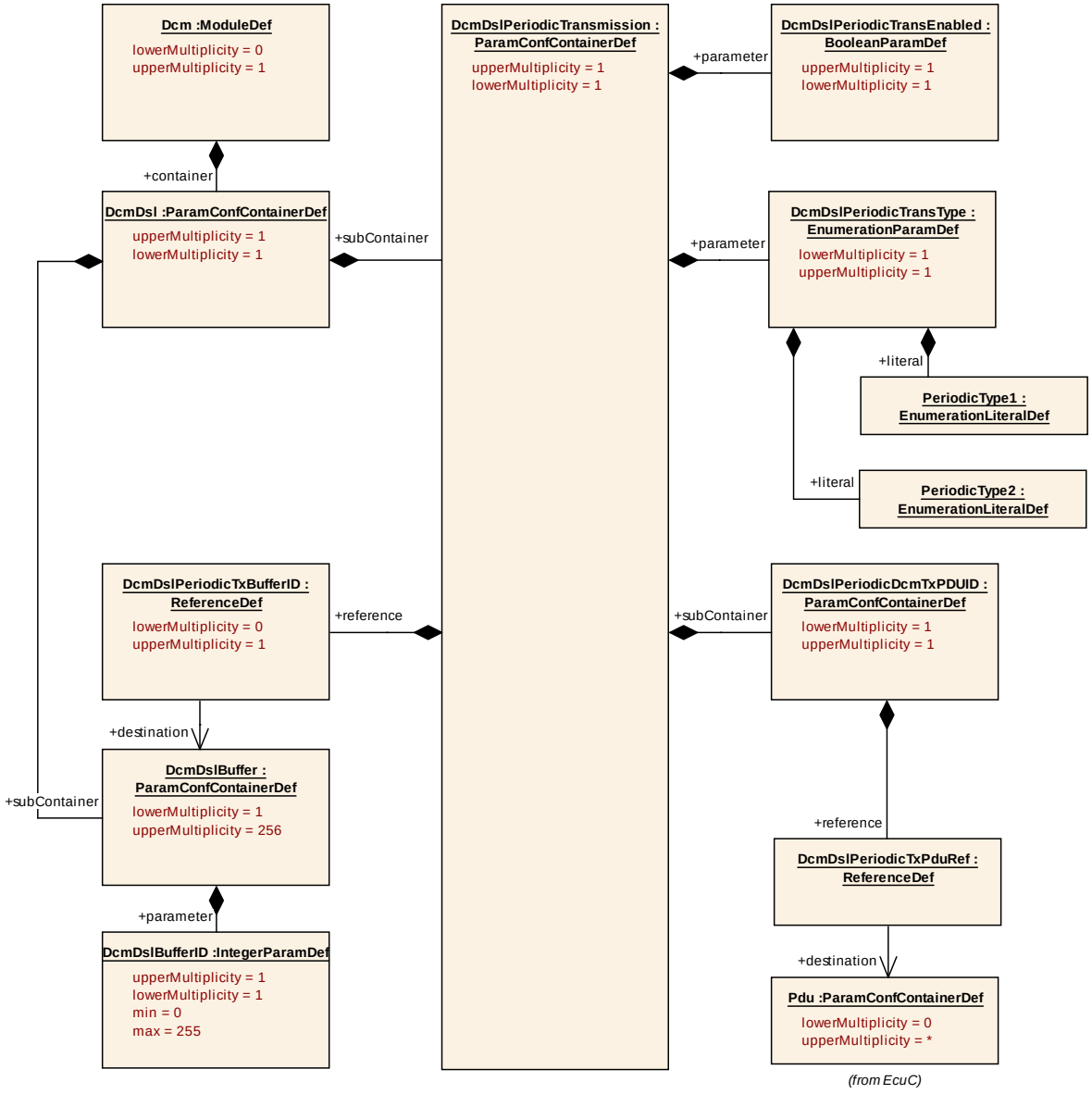


Figure 5.38: `DCM DSL` Periodic Transmission

DcmDslPeriodicTransmission

SWS Item	[Dcm070]
Container Name	DcmDslPeriodicTransmission {PERIODIC_TRANSMISSION_PARAMETERS}
Description	The parameters within this container are required per protocol (DSL Configuration).
Configuration Parameters	

Name last modified in: V1.0.0	DcmDslPeriodicTransEnabled {DCM_PERIODIC_TRANS_ENABLED} [Dcm070]		
Description	Allow to enable or disable a periodic transmission (CFG_ON, CFG_OFF).		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	–
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name last modified in: V1.0.0	DcmDslPeriodicTransType {DCM_PERIODIC_TRANS_TYPE} [Dcm070]		
Description	Defines the transmission type for Periodic Transmissions. The SWS for DCM defines it as enumerated with two values: TYPE1 = normal diagnostic responses are transmitted on the DSL Tx PDU ID for this protocol (DCM_PROTOCOL_DCMTXPDUID). TYPE2 = normal diagnostic responses are transmitted on a separate DSL Tx PDU ID, the value of which is held in this configuration. upperMultiplicity: Exactly one value of PeriodicTransmission Type must be defined if Periodic Transmission is enabled. lowerMultiplicity: PeriodicTransmission Type does not need to be defined if Periodic Transmission is not enabled. origin: Standard AUTOSAR configuration parameter.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	PeriodicType1	transmission on the DcmTxPduId used for normal diagnostic responses (DCM_PROTOCOL_DCMTXPDUID).	
	PeriodicType2	transmission on a separate DcmTxPduId	
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	L	–
Scope / Dependency	scope: ECU dependency: DCM_PERIODIC_TRANS_ENABLED		

Name last modified in: V1.0.0	DcmDslPeriodicTxBufferID {DCM_PERIODIC_TRANS_TX_BUFFER_ID} [Dcm070]		
Description	This identifies the buffer configuration to be used for the Periodic Tx PDU ID for this DSL. upperMultiplicity: One buffer ID required if Periodic Transmission Type is TYPE2. lowerMultiplicity: No buffer is required if Periodic Transmission Type is TYPE1.		
Multiplicity	0..1		
Type	Reference to DcmDslBuffer		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	L	–
Scope / Dependency **Included Containers**			

Container Name	**Multiplicity**	**Scope / Depedency**
DcmDslPeriodicDcmTxPDUID	1	

DcmDslPeriodicDcmTxPDUID

SWS Item	[Dcm070]
Container Name	DcmDslPeriodicDcmTxPDUID {DCM_PERIODIC_TRANS_DCMTXPDUID}
Description	This is the transmission PDU ID to be used by this DCM when sending Periodic Transmissions. It is only needed for Type2 Periodic Transmissions configurations.
Configuration Parameters	

Name last modified in: V1.0.0	DcmDslPeriodicTxPduRef		
Description			
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency **No Included Containers**			

5.3.3.3.4 DcmDslResponseOnEventConfiguration

`DcmDslResponseOnEventConfiguration` defines how the DSL protocol (UDS, OBD, ...) will behave for response on event transmission mode. It is assumed that this will always be defined once as part of the DSL protocol configuration, even if only to indicate that response on event transmissions are not enabled.

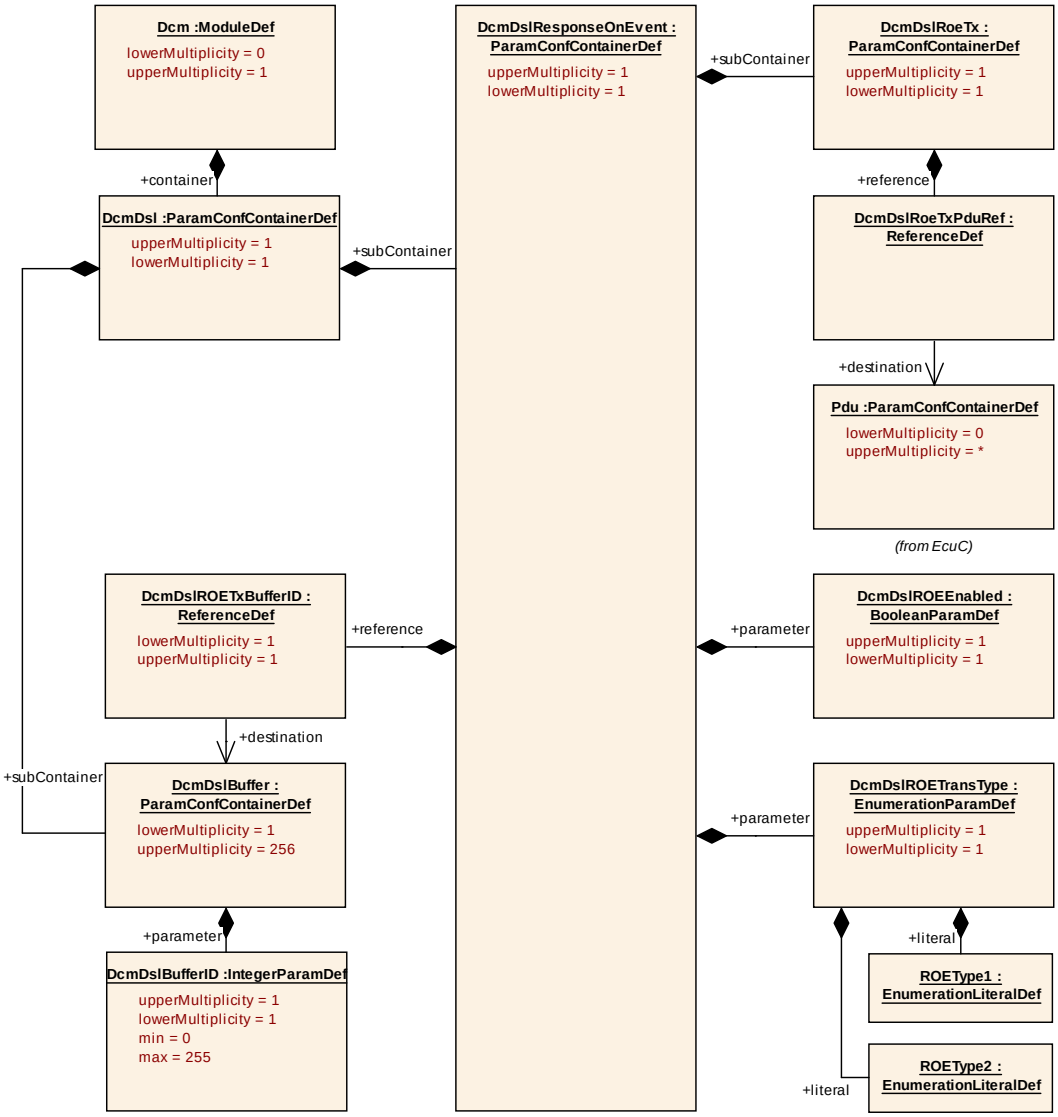


Figure 5.39: DCM DSL Response On Event

DcmDslResponseOnEvent

SWS Item	[Dcm069]
Container Name	DcmDslResponseOnEvent {RESPONSE_ON_EVENT_PARAMETERS}
Description	This container contains the configuration (parameters) for ResponseOnEvent service
Configuration Parameters	

Name last modified in: V1.0.0	DcmDslROEEnabled {DCM_ROE_ENABLED} [Dcm069]		
Description	Allow to enable or disable a ROE transmission		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	–
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name last modified in: V1.0.0	DcmDslROETransType {DCM_ROE_TRANS_TYPE} [Dcm069]		
Description	Defines the transmission type for ResponseOnEvent. The SWS for DCM defines it as enumerated with two values: TYPE1 = normal diagnostic responses are transmitted on the DSL Tx PDU ID for this protocol (DCM_PROTOCOL_DCMTXPDUID). TYPE2 = normal diagnostic responses are transmitted on a separate DSL Tx PDU ID, the value of which is held in this configuration.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	ROEType1	transmission on the DcmTxPduId used for normal diagnostic responses (DCM_PROTOCOL_DCMTXPDUID)	
	ROEType2	transmission on a separate DcmTxPduId	
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	L	–
Scope / Dependency	scope: ECU dependency: DCM_ROE_DCMTXPUID		

Name last modified in: V1.0.0	DcmDslROETxBufferID		
Description	This identifies the buffer configuration to be used for the ROE Tx PDU ID for this DSL. upperMultiplicity: One buffer ID required if ROE Transmission Type is TYPE2. lowerMultiplicity: No bufer required if ROE Transmission Type is TYPE1.		
Multiplicity	1		
Type	Reference to DcmDslBuffer		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Included Containers

Container Name	Multiplicity	Scope / Depedency
DcmDslRoeTx	1	

DcmDslRoeTx

SWS Item			
Container Name	DcmDslRoeTx		
Description	These parameters apply to a Diagnostic Session Layer. There may be a parameter set (DSL Configuration) per protocol. upperMultiplicity: Each DCM configuration must have exactly one DSL configuration. lowerMultiplicity: Each DCM configuration must have exactly one DSL configuration.		
Configuration Parameters			

Name last modified in: V1.0.0	DcmDslRoeTxPduRef		
Description	Reference to the PDU that is transmitted in this frame.		
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency **No Included Containers**			

5.3.3.3.5 DcmDslProtocolConfiguration

`DcmDslProtocolConfiguration` is the master configuration container used to configure a protocol (UDS, OBD, ...) for diagnostic service communications. There can be multiple protocols per DSL so there can be multiple protocol configurations per DSL configuration. It is assumed that each protocol can only have one associated configuration block and so the theoretical upper multiplicity is 256 (the maximum number of different protocols), although only two values are currently defined by AUTOSAR.

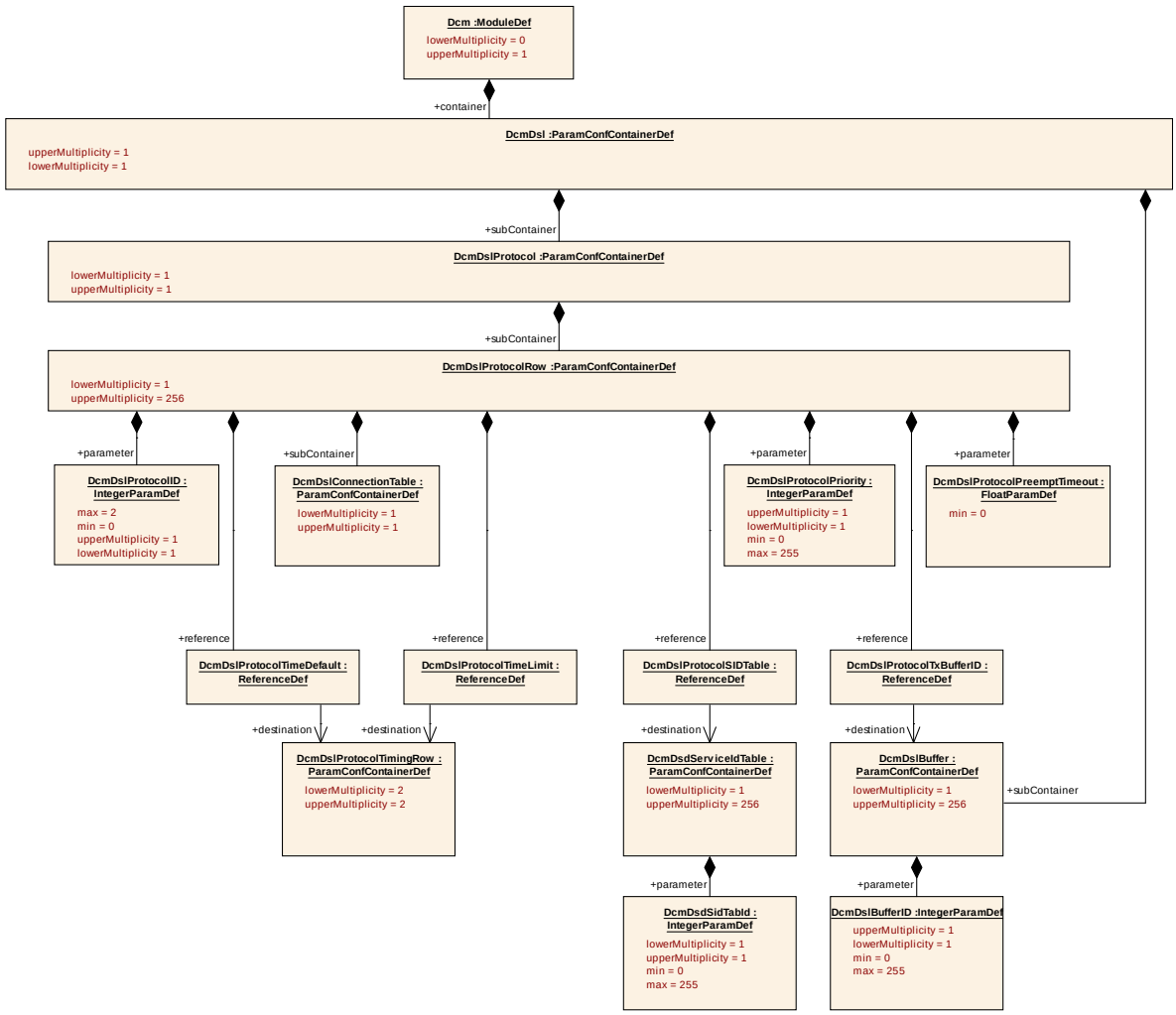


Figure 5.40: `DCM DSL` Protocol

DcmDslProtocol

SWS Item		
Container Name	DcmDslProtocol {DIAGNOSTIC_PROTOCOL_TABLE_ROW}	
Description	This container contains the configuration (parameters) for the protocol configuration (for each protocol) The following parameters needs to be configured per protocol.	
Configuration Parameters Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
DcmDslProtocolRow	1..256	

DcmDslProtocolRow

SWS Item			
Container Name	DcmDslProtocolRow {DIAGNOSTIC_PROTOCOL_TABLE_ROW}		
Description	This container contains the configuration (parameters) for the protocol configuration (for each protocol) The following parameters needs to be configured per protocol.		
Configuration Parameters			
Name last modified in: V1.0.0	DcmDslProtocolID {DCM_PROTOCOL_ID}		
Description	The SWS for DCM defines this as enum type of maxsize uint8 with values {OBD, UDS, ...}. It is effectively the name of the diagnostic protocol type for the DCM DSL protocol that is being configured. 0x00 : DCM_OBD_ON_CAN OBD on CAN (ISO15765-4; ISO15031-5) 0x01 : DCM_UDS_ON_CAN UDS on CAN (ISO15765-3; ISO14229-1) 0x02 : DCM_UDS_ON_FLEXRAY UDS on FlexRay (Manufacturer specific; ISO14229-1) 0x03..0xEF : Reserved for further AUTOSAR implementation 0xF0..0xFF : Reserved for SW supplier specific		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 2		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	L	–
Scope / Dependency	scope: ECU		

Name last modified in: V1.0.0	DcmDslProtocolPreemptTimeout {DCM_PROTOCOL_PREEMPT_TIMEOUT}		
Description	This is the value for the timeout (in milliseconds) of preempting protocol until protocol needs to be started. This is defined in the AUTOSAR SWS for DCM as a uint16. The AUTOSAR configuration standard is to use SI units, so this parameter is defined as a float value in seconds. DCM configuration tools must convert this float value to the appropriate value format for the use in the software implementation of DCM. min: A negative value is not allowed. upperMultiplicity: Exactly one P2ServerMin value per DSL protocol timing structure. lowerMultiplicity: Exactly one P2ServerMin value per DSL timing structure. origin: Standard AUTOSAR configuration parameter.		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	L	–
Scope / Dependency			

Name last modified in: V1.0.0	DcmDslProtocolPriority {DCM_PROTOCOL_PRIO}		
Description	The SWS for DCM defines this item as uint8. This is the protocol priority that is used during protocol preemption handling. 0 = Highest priority (protocol may not be preempted by other protocols) 1, 2, 3... = Reducing priority. Protocol may be preempted by other protocols with lower priority value. upperMultiplicity: Exactly one priority must be provided per Tx Protocol. lowerMultiplicity: Exactly one priority must be provided per Tx Protocol. origin: Standard AUTOSAR configuration parameter.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	L	–
Scope / Dependency			

Name last modified in: V1.0.0	DcmDslProtocolSIDTable {DCM_PROTOCOL_IDENTIFIER_TABLE_ID}		
Description	Link to the diagnostic service table for this protocol. upperMultiplicity: Must have exactly one service table for the protocol. lowerMultiplicity: Must have exactly one service table for the protocol.		
Multiplicity	1		
Type	Reference to DcmDsdServiceIdTable		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	DcmDslProtocolTimeDefault {DCM_DEFAULT}		
Description	Link to the default timing structure (valid for a default session) upperMultiplicity: Exactly one default timing structure must be provided. lowerMultiplicity: Exactly one default timing structure must be provided.		
Multiplicity	1		
Type	Reference to DcmDslProtocolTimingRow		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	DcmDslProtocolTimeLimit {DCM_LIMITS}		
Description	Link to the extended / limit timing structure. Note: limits are checked when using AccessTimingParameter service. upperMultiplicity: Exactly one extended timing structure must be provided. lowerMultiplicity: Exactly one extended timing structure must be provided.		
Multiplicity	1		
Type	Reference to DcmDslProtocolTimingRow		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	DcmDslProtocolTxBufferID {DCM_PROTOCOL_TX_BUFFER_ID}		
Description	Link to the buffer configuration for transmissions for this protocol. upperMultiplicity: Exactly one transmit buffer required for a DSL protocol. lowerMultiplicity: Exactly one transmit buffer required for a DSL protocol.		
Multiplicity	1		
Type	Reference to DcmDslBuffer		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	L	–
Scope / Dependency			
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
DcmDslConnectionTable	1		

DcmDslConnectionTable

SWS Item		
Container Name	DcmDslConnectionTable {DIAGNOSTIC_CONNECTION_TABLE}	
Description	This container contains links between DIAGNOSTIC_PROTOCOL_TABLE and the included containers.	
Configuration Parameters		
Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
DcmDslProtocolRx	1..*	
DcmDslProtocolTx	1..*	

5.3.3.3.6 DcmDslConnectionTableConfiguration

`DcmDslConnectionTableConfiguration` contains a set of parameters for PDU IDs and buffers to be used by the protocol for message reception. Reception of messages may be from multiple sources so multiple Rx Configurations may be defined per protocol.

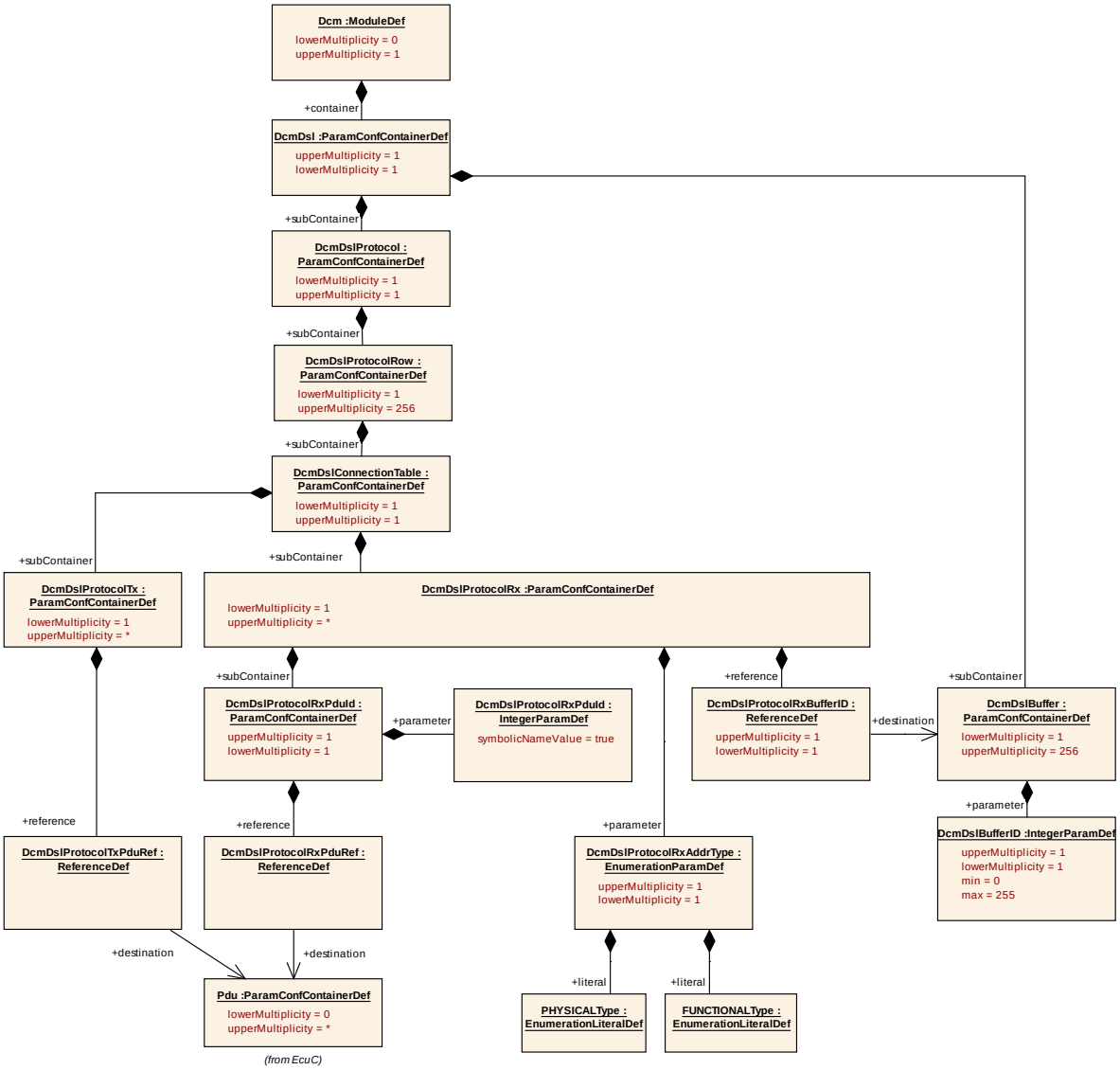


Figure 5.41: DCM DSL Protocol Rx

DcmDslProtocolRx

SWS Item			
Container Name	DcmDslProtocolRx {DIAGNOSTIC_PROTOCOL_RX_TABLE}		
Description	This holds the PDU IDs at which this protocol will receive requests to be used by this DCM. Multiple receive PDU IDs are allowed (e.g. one to receive func requests, one to receive phys requests). Each ID has an associate buffer and address type. upperMultiplicity: More than one receive PDU ID may be configured for reception (e.g. one for func requests, one for phys requests). lowerMultiplicity: At least one receive PDU ID configuration must be provided per protocol.		
Configuration Parameters			

Name last modified in: V1.0.0	DcmDslProtocolRxAddrType {DCM_PROTOCOL_RX_ADDR_TYPE}		
Description	The SWS for DCM defines this as a two value enumeration: PHYSICAL = 1 to 1 communications using physical addressing FUNCTIONAL = 1 to n communication		
Multiplicity	1		
Type	EnumerationParamDef		
Range	FUNCTIONALType	FUNCTIONAL = 1 to n communication	
	PHYSICALType	PHYSICAL = 1 to 1 communications using physical addressing	
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	L	–
Scope / Dependency			

Name last modified in: V1.0.0	DcmDslProtocolRxBufferID {DCM_PROTOCOL_RX_BUFFER_}		
Description	Link to the buffer to be used by this protocol for receiving . upperMultiplicity: Exactly one receive buffer is required for protocol reception. lowerMultiplicity: Exactly one receive buffer is required for protocol reception.		
Multiplicity	1		
Type	Reference to DcmDslBuffer		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	L	–
Scope / Dependency			

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
DcmDslProtocolRxPduId	1	

DcmDslProtocolRxPduId

SWS Item			
Container Name	DcmDslProtocolRxPduId {DCM_PROTOCOL_DCMRXPDUID}		
Description	DcmTxPduId value for transmission of responses.		
Configuration Parameters			

Name last modified in: V1.0.0	DcmDslProtocolRxPduId {DCM_PROTOCOL_DCMRXPDUID}		
Description	DcmTxPduId value for transmission of responses.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	L	–
Scope / Dependency			

Name last modified in: V1.0.0	DcmDslProtocolRxPduRef		
Description			
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

DcmDslProtocolTx

SWS Item	
Container Name	DcmDslProtocolTx {DIAGNOSTIC_PROTOCOL_RX_TABLE}
Description	This holds the PDU IDs at which this protocol will receive requests to be used by this DCM. Multiple receive PDU IDs are allowed (e.g. one to receive func requests, one to receive phys requests). Each ID has an associate buffer and address type. upperMultiplicity: More than one receive PDU ID may be configured for reception (e.g. one for func requests, one for phys requests). lowerMultiplicity: At least one receive PDU ID configuration must be provided per protocol.
Configuration Parameters	

Name last modified in: V1.0.0	DcmDslProtocolTxPduRef		
Description			
Multiplicity	1		
Type	Reference to Pdu		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

5.3.3.4 DcmDsdConfiguration

`DcmDsdConfiguration` configures the Diagnostic Service Dispatcher in the DCM instance. Only one DSD, and hence one set of these parameters, is present in each instance of DCM.

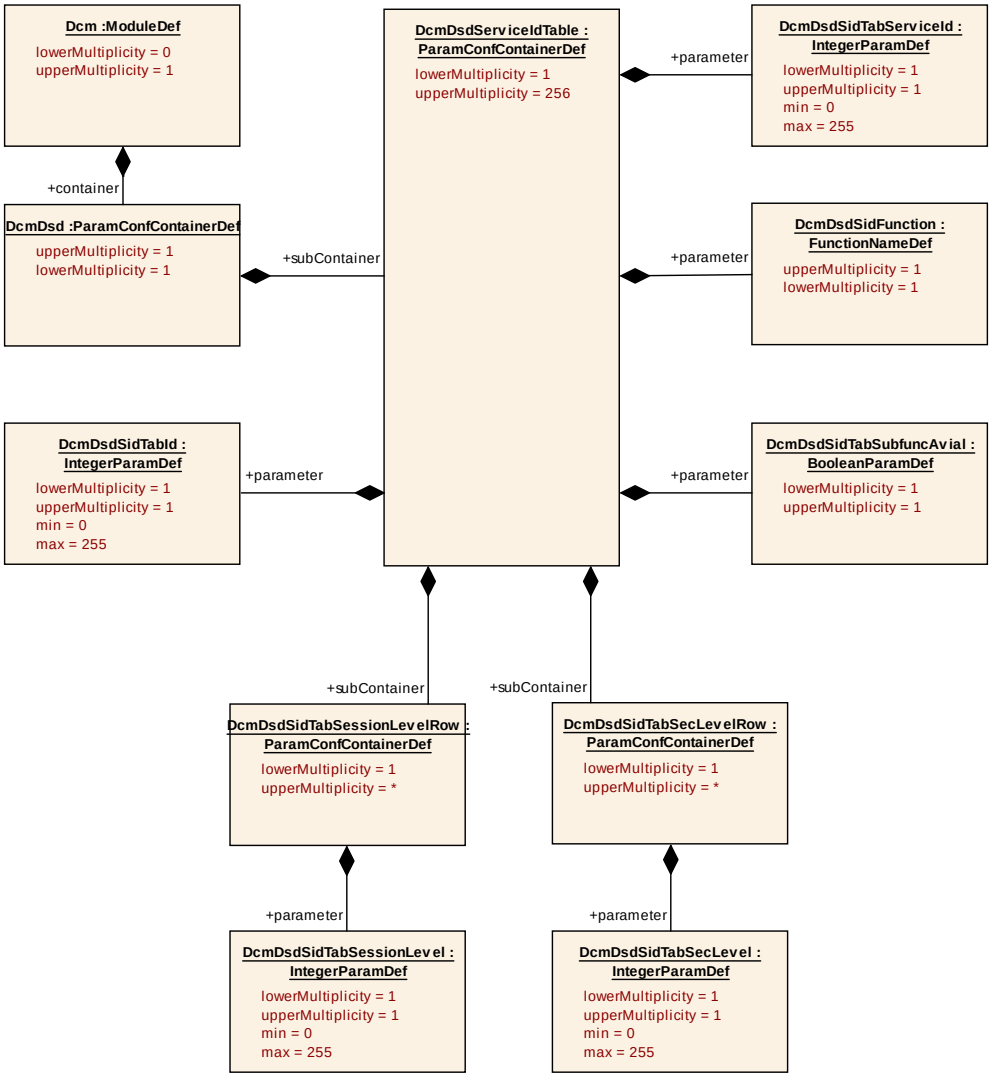


Figure 5.42: DCM DSD - Diagnostic Service Dispatcher

DcmDsd

SWS Item		
Container Name	DcmDsd	
Description	These parameters apply to Diagnostic Service Dispatcher. All parameters for all service dispatchers are included in this one configuration container. upperMultiplicity: Multiple service dispatchers are allowed in this one configuration container. lowerMultiplicity: There must always be one service dispatcher in a DCM.	
Configuration Parameters Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
DcmDsdServiceIdTable	1..256	

DcmDsdServiceIdTable

SWS Item	[Dcm071]		
Container Name	DcmDsdServiceIdTable {SERVICE_IDENTIFIER_TABLE}		
Description	This container contains the configuration (DSD parameters) for Service Identifier Table.		
Configuration Parameters			

Name last modified in: V1.0.0	DcmDsdSidFunction {DCM_SIDTAB_FUNCTIONPOINTER} [Dcm071]		
Description	Function Pointer for the callback function of the DSP or/and the Autosar SW component for the particular DCM_SIDTAB_SERVICEID. Please refer to the ISO 14229-1, ISO 15031-5 and ”Distribution of Diagnostic Message to data processor” (chapter 7.2.2.4). Type: void XXX_Dcm<DiagnosticService>(Dcm_MsgContextType*)		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name last modified in: V1.0.0	DcmDsdSidTabId {DCM_SIDTAB_ID} [Dcm071]		
Description	Due the fact of using one or more service tables the member of the Service Identifier Table includes a unique id for the Service Identifier Table.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name last modified in: V1.0.0	DcmDsdSidTabServiceId {DCM_SIDTAB_SERVICEID} [Dcm071]		
Description	Id of the Service identifier in hex. The possible Service identifier are predefined in the ISO 14229-1 and ISO 15031-5 and in Table 5 and Table 6.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name last modified in: V1.0.0	DcmDsdSidTabSubfuncAvial {DCM_SIDTAB_SUBFUNC_AVAIL} [Dcm071]		
Description	Information whether the DCM_SIDTAB_SERVICEID includes Sub functions or not. Used for the Handling of ”suppressPosRspMsgIndicationBit” (chapter 7.2.2.4) ISO14229-1 can be referenced here, as this specification gives fix definition, if an SID includes Subfunction or not.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	–	
Scope / Dependency	scope: ECU		

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
DcmDsdSidTabSecLevel Row	1..*	
DcmDsdSidTabSession LevelRow	1..*	

DcmDsdSidTabSecLevelRow

SWS Item	
Container Name	DcmDsdSidTabSecLevelRow {Service Table Row}
Description	Within each Service Table there may be one or more services, each with a configuration record.
Configuration Parameters	

Name last modified in: V1.0.0	DcmDsdSidTabSecLevel {DCM_SIDTAB_SEC_LEVEL}		
Description	Information about the Security Access Levels needed for execution of the DCM_SIDTAB_SERVICEID. Please refer to the ISO 14229-1, ISO 15031-5 and ”Verification of the Security Access levels” (chapter 7.2.2.4). For DCM_SEC_LEVEL please refer to chapter 10.5 DSP (Diagnostic Service Processing)Please note, that it shall be provided to configure several DCM_SIDTAB_SEC_LEVEL per DCM_SIDTAB_SERVICEID. 0x00, 0x01,0x03...0x7F: <configuration dependent - only odd values (and 0) are allowed (according to ”securityAccessType” parameter of SecurityAccess request (see [6]))> 0xFF: all Sec. Access Levels		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	–	
Scope / Dependency	scope: ECU		
No Included Containers			

DcmDsdSidTabSessionLevelRow

SWS Item			
Container Name	DcmDsdSidTabSessionLevelRow {DCM_SIDTAB_SESSION_LEVEL_ROW}		
Description	Within each Service Table there may be one or more services, each with a configuration record.		
Configuration Parameters			

Name last modified in: V1.0.0	DcmDsdSidTabSessionLevel {DCM_SIDTAB_SESSION_LEVEL}		
Description	Information about the Session Control needed for execution of the DCM_SIDTAB_SERVICEID. Please refer to the ISO 14229-1, ISO 15031-5 and ”Verification of the Diagnostic Session” (chapter 7.2.2.4). For DCM_SESSION_LEVEL please refer to chapter 10.5 DSP (Diagnostic Service Processing)Please note, that it shall be provided to configure several DCM_SIDTAB_SESSION_LEVEL per DCM_SIDTAB_SERVICEID.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	–	
Scope / Dependency	scope: ECU		
No Included Containers			

5.3.3.5 DcmDspConfiguration

`DcmDspConfiguration` contains Diagnostic Service Processing configuration parameters for all the DCM sessions, security levels and DTC sub-functions. Only one set of these parameters is present per instance of DCM.

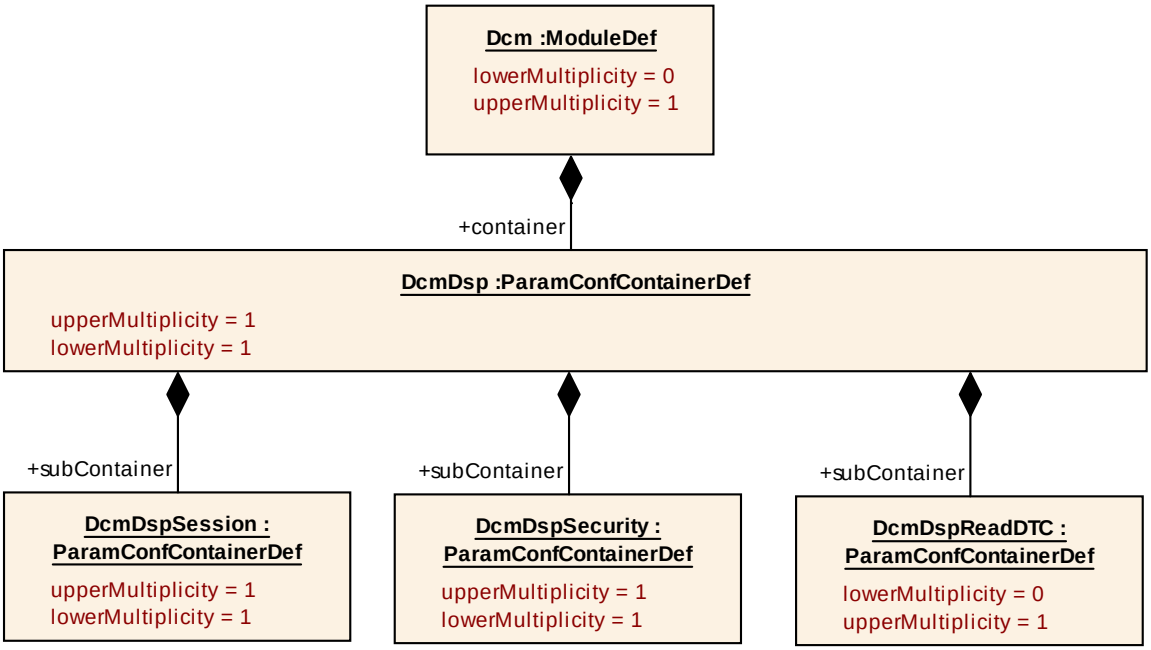


Figure 5.43: DCM DSP - Diagnostic Service Processing

DcmDsp

SWS Item		
Container Name	DcmDsp	
Description	These parameters apply to Diagnostic Service Processing. There will always be one set of these parameters per DCM.	
Configuration Parameters Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
DcmDspReadDTC	0..1	
DcmDspSecurity	1	
DcmDspSession	1	

5.3.3.5.1 DcmDspReadDTCConfiguration

The ”Read DTC” service may support sub-functions to access different sets of fault data. A configuration block with three mandatory parameters is required for each sub-function. It is assumed that there can be a virtual ”infinite” number of sub-functions and the name string acts as the key.

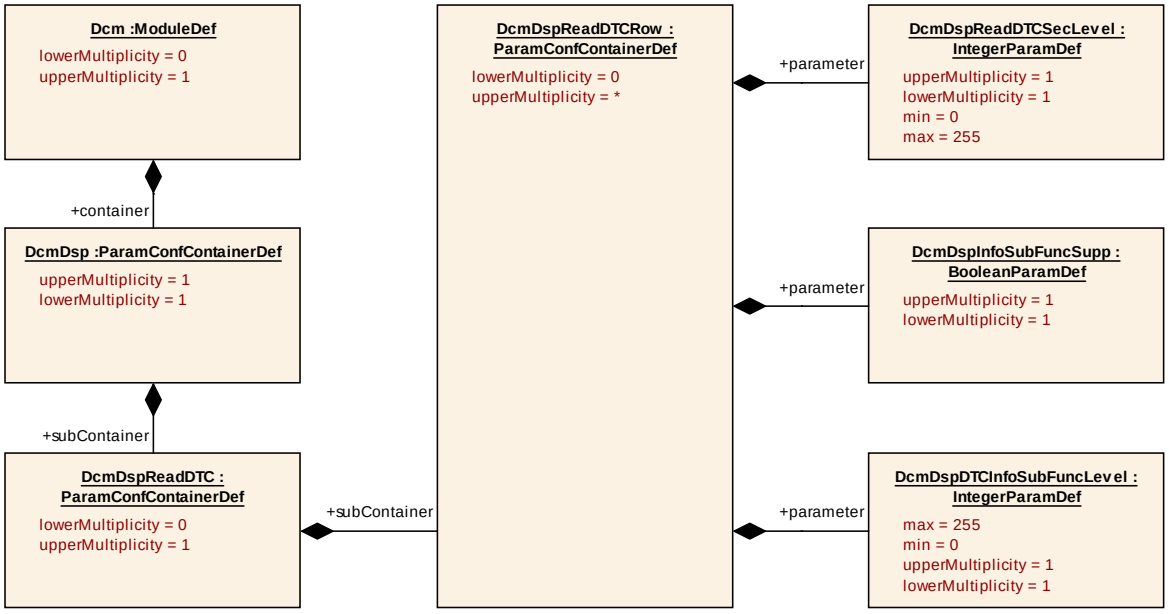


Figure 5.44: `DcmDspReadDTC`

DcmDspReadDTC

SWS Item	[Dcm074]	
Container Name	DcmDspReadDTC {DCM_READDTC_SUB_FUNCTION_TABLE}	
Description	This container contains the configuration (parameters) of the service Read DTC Information - 0x19 (per sub-function type)	
Configuration Parameters Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
DcmDspReadDTCRow	0..*	

DcmDspReadDTCRow

SWS Item	[Dcm074]		
Container Name	DcmDspReadDTCRow {DCM_READDTC_SUB_FUNCTION_TABLE_ROW}		
Description	This container contains the configuration (parameters) of the service Read DTC Information - 0x19 (per sub-function type)		
Configuration Parameters			
Name last modified in: V1.0.0	DcmDspDTCInfoSubFuncLevel {DCM_DTC_INFO_SUB_FUNCTION_LEVEL} [Dcm074]		
Description	hex value of the sub-function.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	DcmDspInfoSubFuncSupp {DCM_DTC_INFO_SUB_FUNCTION_SUPPORT} [Dcm074]		
Description	This parameter is defined in the DCM SWS as a Boolean that indicates whether the respective sub-function is supported. TRUE = sub-function supported. FALSE = sub-function not supported.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	DcmDspReadDTCSecLevel {DCM_SECURITY_LEVEL} [Dcm074]		
Description	Security level value, indicates which security level is necessary to execute a specific sub-function 0x00, 0x01,0x03...0x7F: configuration dependent - only odd values (and 0) are allowed (according to "securityAccessType" parameter of SecurityAccess request		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	–	
Scope / Dependency	scope: Module		
No Included Containers			

5.3.3.5.2 DcmDspSecurityConfiguration

A security configuration block is required per security level that is to be supported by the DSP (the security level is the key that selects the active configuration). It defines the size of the security seeds and keys to be transferred and the number of attempts that are allowed before delays and, finally, lockout are introduced. Exactly one of each configuration value is required for a valid security level configuration.

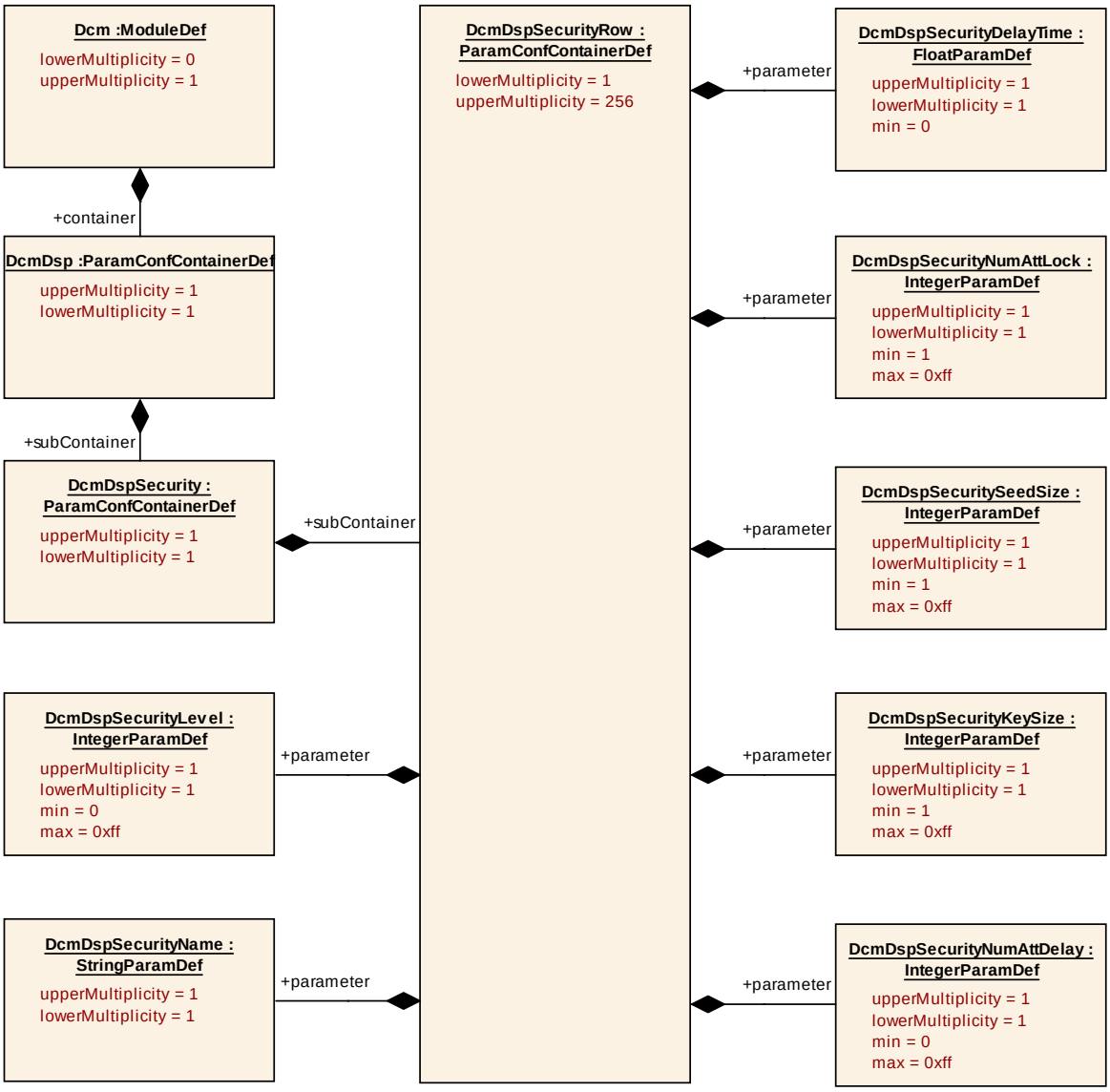


Figure 5.45: `DcmDspSecurity`

DcmDspSecurity

SWS Item	[Dcm073]	
Container Name	DcmDspSecurity {DCM_SEC_LEVEL_TABLE}	
Description	These parameters apply to Diagnostic Service Processing. There will always be one set of these parameters per DCM.	
Configuration Parameters Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
DcmDspSecurityRow	1..256	

DcmDspSecurityRow

SWS Item	[Dcm073]		
Container Name	DcmDspSecurityRow {DCM_SEC_LEVEL_TABLE_ROW}		
Description	This container contains the configuration (DSP parameter) for security level configuration (per security level)		
Configuration Parameters			
Name last modified in: V1.0.0	DcmDspSecurityDelayTime {DCM_SEC_DELAY_INV_KEY} [Dcm073]		
Description	Defined in the DCM SWS as uint16, this parameter contains a value in milliseconds for the delay time introduced after a failed security access once DcmDspSecurityNumAttDelay failed attempts have been made. The AUTOSAR configuration standard is to use SI units, so this parameter is defined as float value in seconds. DCM configuration tools must convert this float value to the appropriate value format for the use in the software implementation of DCM. min: A negative value is not allowed. upperMultiplicity: Exactly one delay tie must be specified per security levbel configuration. lowerMultiplicity: Exactly one delay tie must be specified per security levbel configuration. origin: Standard AUTOSAR configuration parameter.		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	DcmDspSecurityKeySize {DCM_SEC_NUM_KEY} [Dcm073]		
Description	This parameter is defined in the DCM SWS as uint8. It holds a value indicating the size of the security key (in Bytes).		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 255		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	DcmDspSecurityLevel {DCM_SEC_LEVEL} [Dcm073]		
Description	Defined in the DCM SWS as a uint8 containing the security level value. 0 is the highest security level. 0x00, 0x01,0x03...0x7F: configuration dependent - only odd values (and 0) are allowed (according to ”securityAccessType” parameter of SecurityAccess request		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	DcmDspSecurityName {DCM_SEC_NAME} [Dcm073]		
Description	Defined in the DCM SWS as a string containing the name of the security level.		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	DcmDspSecurityNumAttDelay {DCM_SEC_NUM_MAX_ATT_DELAY} [Dcm073]		
Description	Defined in the DCM SWS as uint8, this parameter contains the number of failed security accesses after which the delay time is activated.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	–	
Scope / Dependency	scope: Module dependency: DCM_SEC_NUM_MAX_ATT_DELAY < DCM_SEC_NUM_MAX_ATT_LOCK		

Name last modified in: V1.0.0	DcmDspSecurityNumAttLock {DCM_SEC_NUM_MAX_ATT_LOCK} [Dcm073]		
Description	Defined in the DCM SWS as uint8, this parameter contains the number of failed security accesses after which security access is locked.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 255		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	All Variants
	Post-build time	–	
Scope / Dependency	scope: Module dependency: DCM_SEC_NUM_MAX_ATT_DELAY < DCM_SEC_NUM_MAX_ATT_LOCK		

Name last modified in: V1.0.0	DcmDspSecuritySeedSize {DCM_SEC_NUM_SEED} [Dcm073]		
Description	This parameter is defined in the DCM SWS as uint8. It holds a value indicating the size of the security seed (in Bytes).		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 255		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	–	
Scope / Dependency	scope: Module		
No Included Containers			

5.3.3.5.3 DcmDspSessionConfiguration

A session configuration block is required per session level (session control value) that is to be supported by the DSP (the session level is the key that selects the active configuration). The configuration block defines the name and server timeouts to be used at that session level. Exactly one of each configuration value is required for a valid session level configuration.

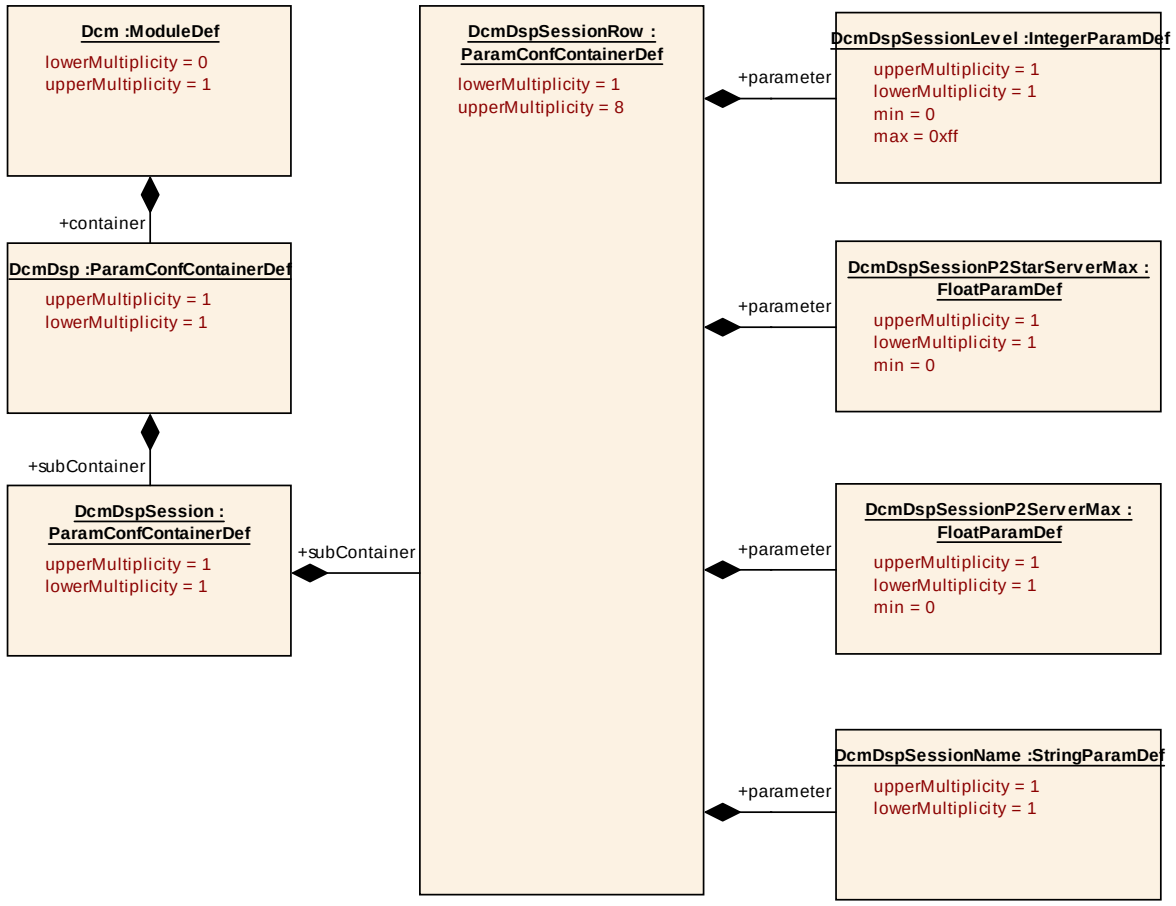


Figure 5.46: `DcmDspSession`

DcmDspSession

SWS Item	[Dcm072]	
Container Name	DcmDspSession {DCM_SESSION_LEVEL_TABLE}	
Description	These parameters apply to Diagnostic Service Processing. There will always be one set of these parameters per DCM.	
Configuration Parameters Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
DcmDspSessionRow	1..8	

DcmDspSessionRow

SWS Item	[Dcm072]
Container Name	DcmDspSessionRow {DCM_SESSION_LEVEL_TABLE_ROW}
Description	This container contains the configuration (DSP parameter) session control configuration (per session control)
Configuration Parameters	

Name last modified in: V1.0.0	DcmDspSessionLevel {DCM_SESSION_LEVEL} [Dcm072]		
Description	This is defined as uint8 in the SWS for DCM. This is the session control value. .		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	DcmDspSessionName {DCM_SESSION_NAME} [Dcm072]		
Description	Name of the Session control.		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	DcmDspSessionP2ServerMax {DCM_SESSION_P2SERVER_MAX} [Dcm072]		
Description	This is the session value for P2ServerMax in milliseconds. This is defined in the AUTOSAR SWS for DCM as a uint16. The AUTOSAR configuration standard is to use SI units, so this parameter is defined as a float value in seconds. DCM configuration tools must convert this float value to the appropriate value format for the use in the software implementation of DCM.		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	DcmDspSessionP2StarServerMax {DCM_SESSION_P2STRSERVER_MAX} [Dcm072]		
Description	This is the session value for P2ServerMax in milliseconds. This is defined in the AUTOSAR SWS for DCM as a uint16. The AUTOSAR configuration standard is to use SI units, so this parameter is defined as a float value in seconds. DCM configuration tools must convert this float value to the appropriate value format for the use in the software implementation of DCM.		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	–
	Link time	X	–
	Post-build time	–	
Scope / Dependency	scope: Module		
No Included Containers			

5.4 ECU State Manager

The Ecu State Manager (EcuM) is specified in [17]. The top level configuration container and its including subcontainers of the ECU State Manager are shown in Figure 5.47.

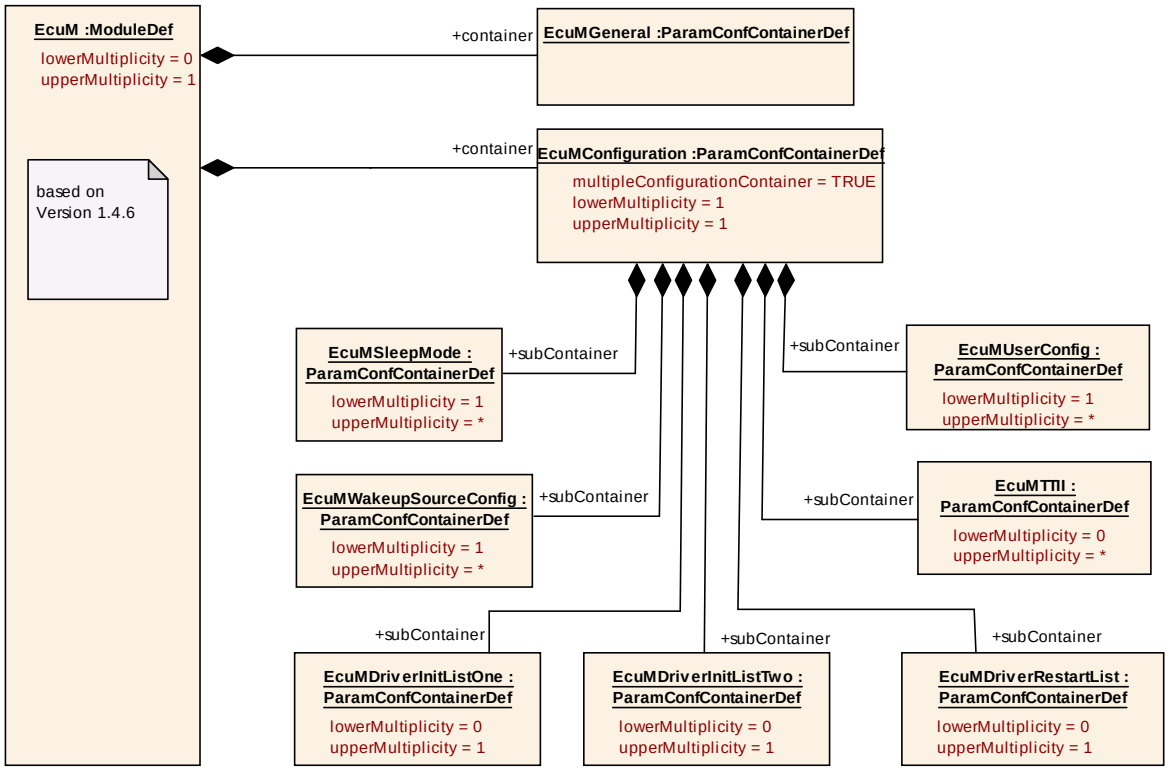


Figure 5.47: Overview of the module EcuM

The top level configuration container EcuMConfiguration also contains parameters which are shown in Figures 5.48 and 5.49

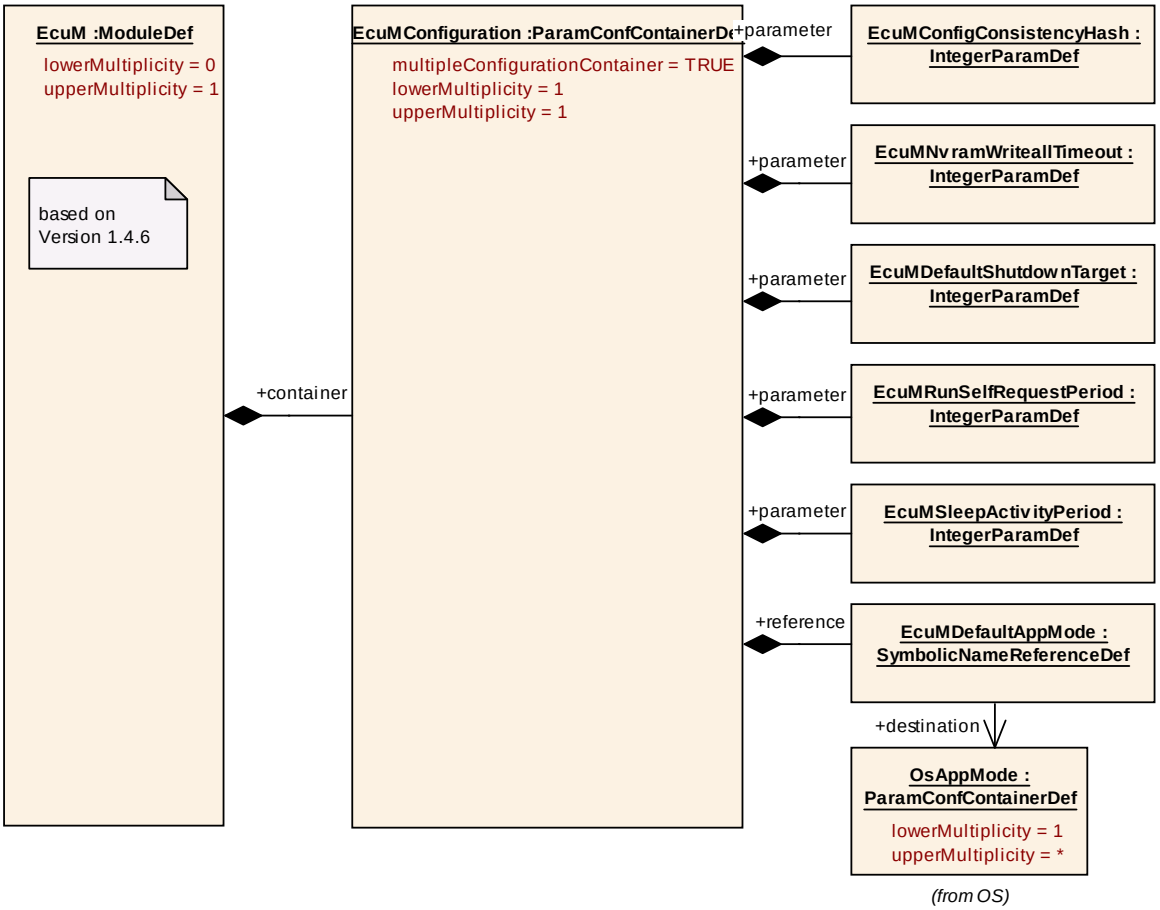


Figure 5.48: Parameters of the container EcuMConfiguration

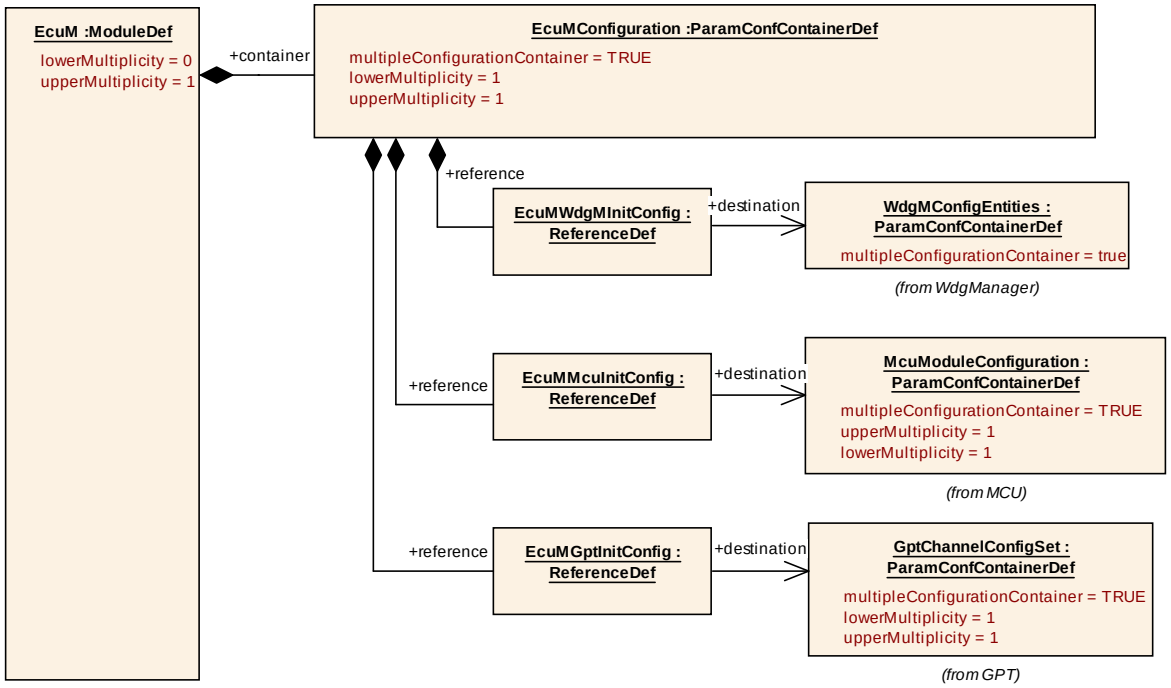


Figure 5.49: Init Config references in EcuMConfiguration

EcuMConfiguration

SWS Item			
Container Name	EcuMConfiguration {EcuM_Configuration}		
Description	This container contains the configuration (parameters) of the ECU State Manager		
Configuration Parameters			

Name last modified in: V1.1.0	EcuMConfigConsistencyHash {ECUM_CONFIGCONSISTENCY_HASH}		
Description	A hash value generated across all pre-compile and link-time parameters of all BSW modules. This hash value is compared against a field in the EcuM_ConfigType and hence allows checking the consistency of the entire configuration.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	X	PC
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	EcuMDefaultAppMode {ECUM_DEFAULT_APP_MODE}		
Description	The default application mode loaded when the ECU comes out of reset.		
Multiplicity	1		
Type	Symbolic name reference to OsAppMode		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	PC
Scope / Dependency			

Name last modified in: V1.0.0	EcuMDefaultShutdownTarget {ECUM_DEFAULT_SHUTDOWN_TARGET}		
Description	Structure of - EcuM_StateType - uint8 The integer number is the mode qualifier for the default target. It is an index into the ECUM_SLEEP_MODES list. See also parameters of the EcuM_SelectShutdownTarget service.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	PC
Scope / Dependency			

Name last modified in: V1.1.0	EcuMGptInitConfig {ECUM_MCU_INIT_CONFIG}		
Description	Init configuration for the General Purpose Timer		
Multiplicity	1		
Type	Reference to GptChannelConfigSet		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	EcuMMcuInitConfig {ECUM_MCU_INIT_CONFIG}		
Description	Init configuration for the MCU Driver		
Multiplicity	1		
Type	Reference to McuModuleConfiguration		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	EcuMNvramWriteallTimeout {ECUM_NVRAM_WRITEALL_TIMEOUT}		
Description	Period given in milliseconds for which the ECU State Manager will wait until it considers a write all job of the NVRAM Manager as failed		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	PC
Scope / Dependency			

Name last modified in: V1.0.0	EcuMRunSelfRequestPeriod {ECUM_RUN_SELF_REQUEST_PERIOD}		
Description	Period given in milliseconds for which the ECU State Manager will request itself to stay in RUN state when RUN state just has been entered.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	PC
Scope / Dependency			

Name last modified in: V1.1.0	EcuMSleepActivityPeriod {ECUM_SLEEP_ACTIVITY_PERIOD}		
Description	Period of the EcuM_SleepActivity callout. The period is given in milliseconds		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	PC
Scope / Dependency			

Name last modified in: V1.0.0	EcuMWdgMInitConfig {ECUM_WDGM_INIT_CONFIG}		
Description	EcuM2722: Init configuration for the Watchdog Manager EcuM2681: The configuration shall reference the startup mode of the Watchdog Manager		
Multiplicity	1		
Type	Reference to WdgMConfigEntities		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Included Containers

Container Name	Multiplicity	Scope / Depedency
EcuMDriverInitListOne	0..1	
EcuMDriverInitListTwo	0..1	
EcuMDriverRestartList	0..1	
EcuMSleepMode	1..*	
EcuMTTII	0..*	
EcuMUserConfig	1..*	
EcuMWakeupSource Config	1..*	

The configuration parameters of the subcontainer EcuMInitLists are shown in figure 5.50

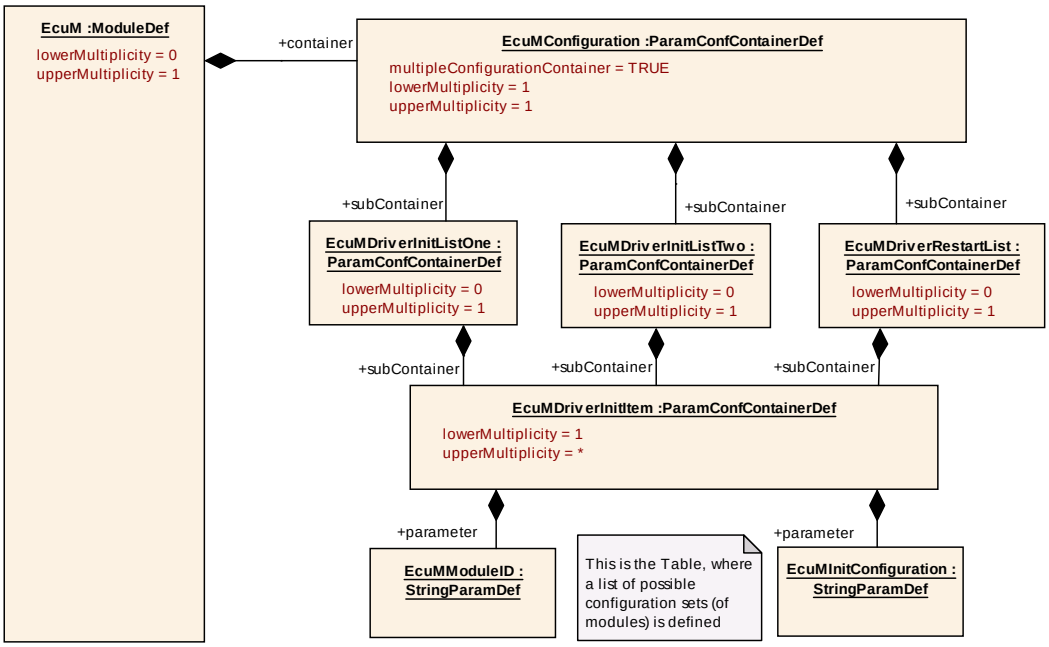


Figure 5.50: Overview of the container EcuMInitLists

EcuMDriverInitListOne

SWS Item		
Container Name	EcuMDriverInitListOne	
Description	Container of module Initialisation List One. This container holds a list of module IDs that will be initialised under a predefined set of circumstances. When the predefined conditions are met, each module in the list will be called for initialisation in the list order. All modules in this list are initilialised before the OS is started and so these modules require no OS support.	
Configuration Parameters **Included Containers**		
Container Name	**Multiplicity**	**Scope / Depedency**
EcuMDriverInitItem	1..*	

EcuMDriverInitListTwo

SWS Item		
Container Name	EcuMDriverInitListTwo	
Description	Container of module Initialisation List One. This container holds a list of module IDs that will be initialised under a predefined set of circumstances. When the predefined conditions are met, each module in the list will be called for initialisation in the list order. All modules in this list are initilialised after the OS is started and so these modules may use OS support.	
Configuration Parameters Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
EcuMDriverInitItem	1..*	

EcuMDriverRestartList

SWS Item		
Container Name	EcuMDriverRestartList	
Description	List of module IDs. EcuM2719: A configuration tool shall fill the callout EcuM_AL_DriverRestart with initialization calls to the listed drivers in the order in which they occur in the list. EcuM2720: Entries in this list must appear in the same order as in the combined list of EcuM_DriverInitListOne and EcuM_DriverInitListTwo. This list may be a real subset though. In all other cases, the generation tool shall report an error. The included container has the same structure as EcuM_DriverInitItem	
Configuration Parameters Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
EcuMDriverInitItem	1..*	

EcuMDriverInitItem

SWS Item	
Container Name	EcuMDriverInitItem
Description	This container describes a structure with the following items as its elements. EcuM_DriverInitItems are concatenated to build up a list.
Configuration Parameters	

Name last modified in: V1.0.0	EcuMInitConfiguration {InitConfiguration}		
Description	This paramter contains pointer to the init structure of the corresponding BSW module		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	EcuMModuleID {ModuleID}		
Description	An item in an init list is referenced by its module ID.		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

The configuration parameters of the subcontainer EcuMSleepMode are shown in figure 5.51 The subcontainer EcuMSleepMode is shown in figure 5.51

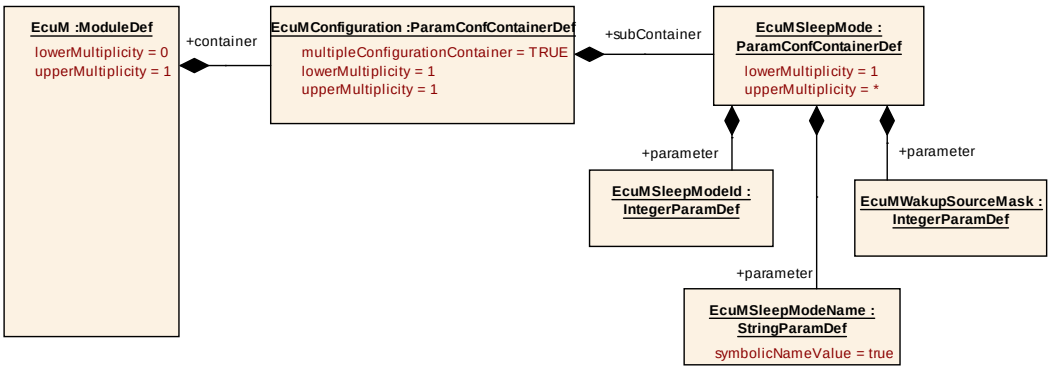


Figure 5.51: Overview of the container EcuMSleepMode

EcuMSleepMode

SWS Item	
Container Name	EcuMSleepMode
Description	List of the configured sleep modes
Configuration Parameters	

Name last modified in: V1.1.0	EcuMSleepModeId {SleepModeConfiguration}		
Description	This parameter is part of the SleepModeConfiguration as defined in the SWS. It represents the index to configuration tables of MCU driver of the EcuM_SleepModeConfigType.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	EcuMSleepModeName		
Description	This item allows to give symbolic names to the different sleep modes. Multiple names can be given to one sleep mode. This definition is application dependent. In combination with the hardware specific EcuM_SleepMode, this allows to port SW-Cs across different ECUs.		
Multiplicity	1		
Type	StringParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	EcuMWakupSourceMask {SleepModeConfiguration}		
Description	This parameter is part of the SleepModeConfiguration as defined in the SWS. This mask defines the wakeup sources to be enabled for the following sleep period.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

The configuration parameters of the subcontainer EcuMTTII are shown in figure 5.52

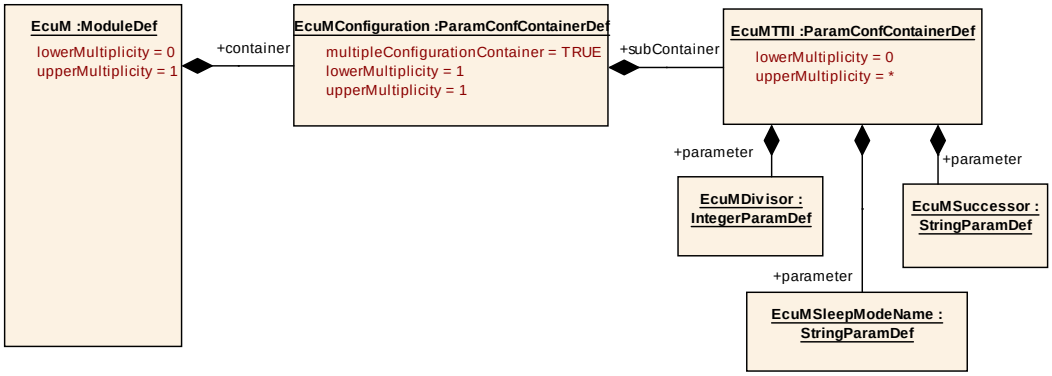


Figure 5.52: Overview of the container EcuMTTII

EcuMTTII

SWS Item			
Container Name	EcuMTTII		
Description	This container describes a structure and the following configuration items describe its elements. This structures are concatenated to build a list as indicated by Figure 27 - Configuration Container Diagram. The list must contain at least on element when ECUM_TTII_ENABLED is set to true.		
Configuration Parameters			

Name last modified in: V1.0.0	EcuMDivisor {Divisor}		
Description	A lookup table to find the divisor preload value. The key is the associated sleep mode.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	EcuMSleepModeName {WakeupSourceName}		
Description	This configuration parameter is a human readable name for a sleep mode which may only be used by the configuration tool. For imlementation on the ECU, this parameter may be dropped and replaced by a generated list index of EcuM_SleepMode. The SleepModeName (or the index, if more suitable for an ECU implementation) is used to reference the EcuM_SleepMode.		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	EcuMSuccessor {Successor}		
Description	This item is a reference to the next sleep mode in the TTII protocol. The representation here is human readable but may be replaced by an index for efficient implementation on an ECU		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

The configuration parameters of the subcontainer EcuMUserConfig are shown in figure 5.53

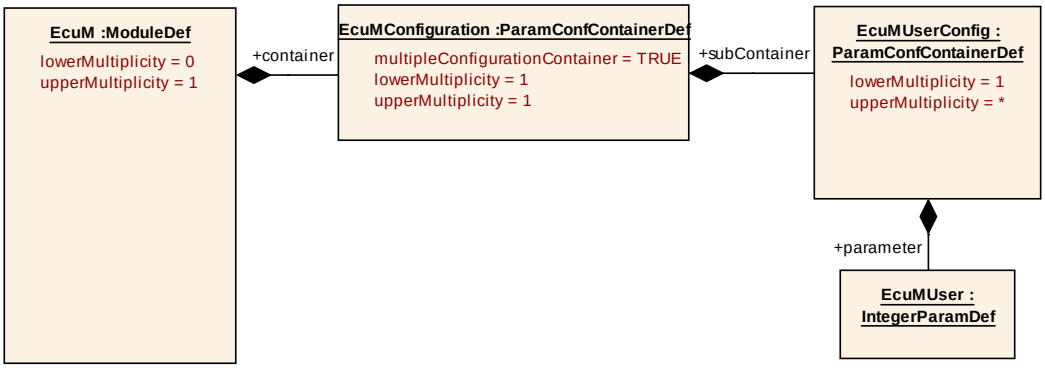


Figure 5.53: Overview of the container EcuMUserConfig

EcuMUserConfig

SWS Item			
Container Name	EcuMUserConfig {EcuM_User}		
Description	A list of identifiers that are needed to refer to a software component or another appropriate entity in the system which is designated to request the RUN state. Application requestors refer to entities above RTE, system requestors to entities below RTE (e.g. Communication Manager).		
Configuration Parameters			

Name last modified in: V1.0.0	EcuMUser		
Description			
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

5.5 Communication Manager

The ComM (Communication Manager) parameters are structured according to figure 5.54 and will be described in detail in the following sections:

- ComMGeneral in section 5.5.1
- ComMChannel in section 5.5.2
- ComMNetworkManagement in section 5.5.3
- ComMUser in section 5.5.4

The ComM support one configuration variant that uses a mixed set of Pre Compile and Post Build configurable parameters. Hence the statement ”All Variants” refers to this single configuration variant.

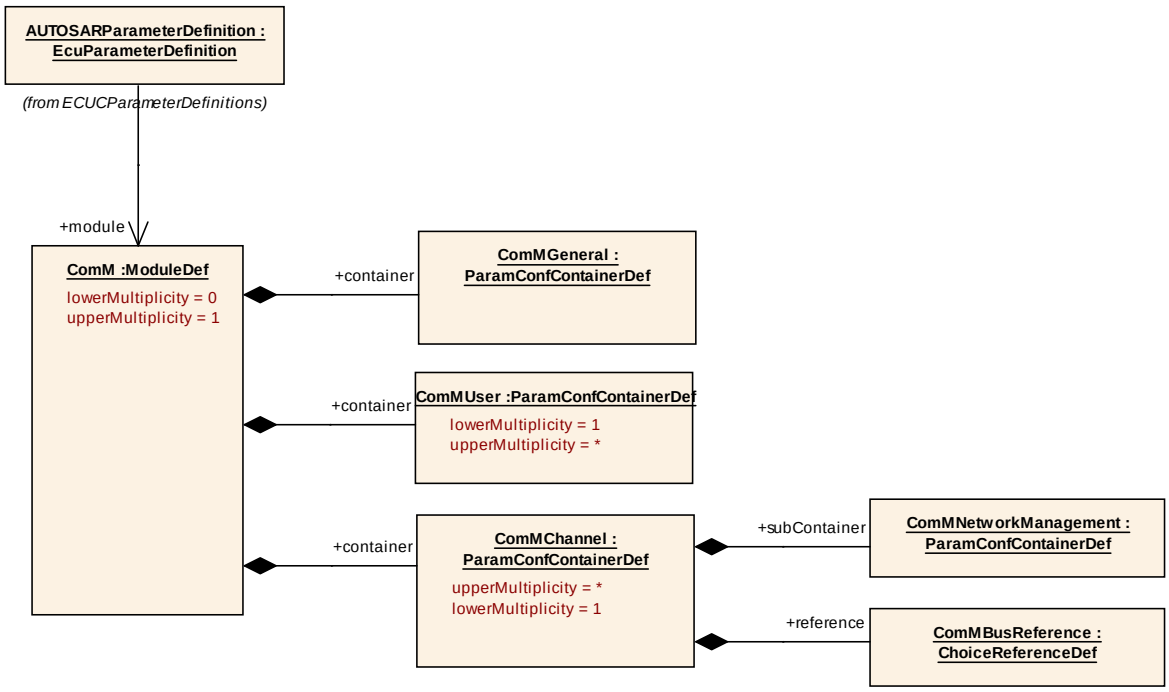


Figure 5.54: ComM configuration overview

Module Name	ComM	
Module Description	Configuration of the ComM (Communications Manager) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
ComMChannel	1..*	This container contains the configuration (parameters) of the bus channel(s). The channel parameters shall be harmonized within the whole communication stack.
ComMGeneral	1	Container for Communication Manager Configuration
ComMUser	1..*	This container contains a list of identifiers that are needed to refer to a user in the system which is designated to request Communication modes.
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.

5.5.1 ComMGeneral

The `ComMGeneral` container is used for parameters that do not belong to a specific user or communication channel.

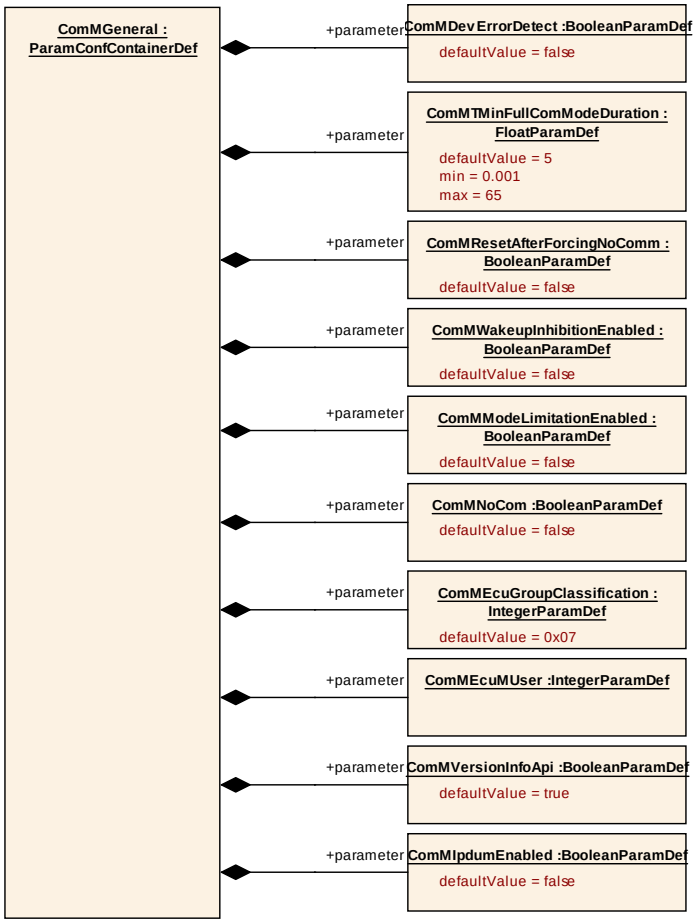


Figure 5.55: ComMGeneral configuration parameters

ComMGeneral

SWS Item	[ComM554]
Container Name	ComMGeneral {CommunicationManagerConfiguration}
Description	Container for Communication Manager Configuration
Configuration Parameters	

Name last modified in: V1.0.0	ComMDevErrorDetect {COMM_DEV_ERROR_DETECT} [ComM555]		
Description	Switches the Development Error Detection and Notification ON or OFF.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	ComMEcuGroupClassification {COMM_ECU_GROUP_CLASSIFICATION} [ComM563]		
Description	Defines whether a mode inhibition affects the ECU or not. Bit 1 Wake Up inhibition (default = 1) Bit 2 Forcing into ”Silent Communication” mode (default = 1) Bit 3 Forcing into ”No Communication” mode (default = 1) Examples: 000: No mode inhibition can be activated 001: Wake up inhibition can be enabled		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: module dependency: Shall be stored none volatile (value must be kept during a reset.). Can be changed during runtime with ComM_SetECUGroupClassification() thus the default values shall be set only once (first ECU initialization).		

Name last modified in: V1.0.0	ComMEcuMUser {COMM_ECUM_USER} [ComM586]		
Description	User handle of ComM for requesting EcuM modes.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	ComMIpdumEnabled {COMM_IDPUM_ENABLED} [ComM643]		
Description	Switches the IPDUM functionality ON or OFF.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	ComMModeLimitationEnabled {COMM_MODE_LIMITATION_ENABLED} [ComM560]		
Description	TRUE if mode limitation functionality shall be enabled.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	ComMNoCom {COMM_NO_COM} [ComM561]		
Description	TRUE if ECU is not allowed to change state of the ECU to ”Silent Communication” or ”Full communication”. TRUE - Enabled (not allowed to switch to ”Silent Communication mode” or ”Full Communication mode”) FALSE - Disabled (default)		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: module dependency: Shall not be stored none volatile (shall be reseted to default after initialization). Can be changed during runtime with ComM_LimitECUToNoComMode().		

Name last modified in: V1.0.0	ComMResetAfterForcingNoComm {COMM_RESET_AFTER_FORCING_NO_COMM} [ComM558]		
Description	ComM shall perform a reset after entering ”No Communication” mode because of an active mode limitation to ”No Communication” mode.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	ComMTMinFullComModeDuration {COMM_T_MIN_FULL_COM_MODE_DURATION} [ComM557]		
Description	Minimum time duration in seconds, spent in the Full Communication mode.		
Multiplicity	1		
Type	FloatParamDef		
Range	0.001 .. 65		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	ComMVersionInfoApi {COMM_VERSION_INFO_API} [ComM622]		
Description	Switches the possibility to read the published information with the service ComM_GetPublishedInformation().		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	ComMWakeupInhibitionEnabled {COMM_WAKEUP_INHIBITION_ENABLED} [ComM559]		
Description	TRUE if wake up inhibition functionality enabled.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

No Included Containers

5.5.2 ComMChannel

There shall be one `ComMChannel` sub-container for each channel for the ECU. Each `ComMChannel` contains a choice reference `ComMBusReference`, which refers to a CAN, LIN or Flexray channel.

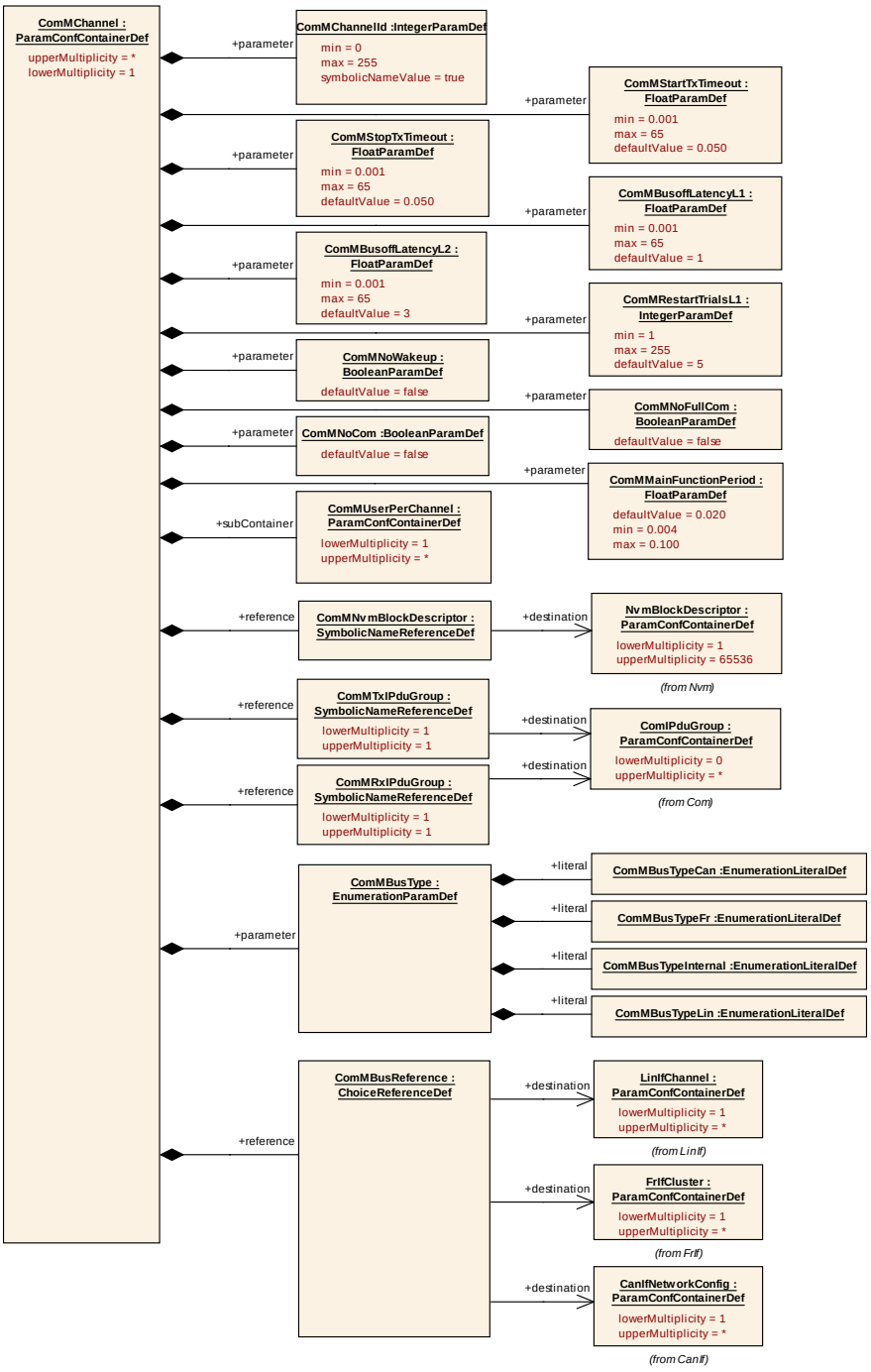


Figure 5.56: ComMChannel configuration parameters

ComMChannel

SWS Item	[ComM565]
Container Name	ComMChannel {Channel}
Description	This container contains the configuration (parameters) of the bus channel(s). The channel parameters shall be harmonized within the whole communication stack.
Configuration Parameters	

Name last modified in: V1.1.0	ComMBusReference		
Description	Subcontainer for reference to one channel.		
Multiplicity	1		
Type	Choice Reference to [FrIfCluster, LinIfChannel, CanIfNetworkConfig]		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency	dependency: ComMBusType		

Name last modified in: V1.1.0	ComMBusType {COMM_BUS_TYPE} [ComM567]		
Description			
Multiplicity	1		
Type	EnumerationParamDef		
Range	ComMBusTypeCan	Contains reference to CAN bus.	
	ComMBusTypeFr	Contains reference to Flexray bus.	
	ComMBusTypeInternal		
	ComMBusTypeLin	Contains reference to LIN bus.	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name last modified in: V1.0.0	ComMBusoffLatencyL1 {COMM_BUSOFF_LATENCY_L1} [ComM574]		
Description	Time in seconds between a bus off indication and a reconnect attempt or two reconnect attempts during level 1.		
Multiplicity	1		
Type	FloatParamDef		
Range	0.001 .. 65		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	ComMBusoffLatencyL2 {COMM_BUSOFF_LATENCY_L2} [ComM575]		
Description	Time in seconds between two reconnect attempts during level 2.		
Multiplicity	1		
Type	FloatParamDef		
Range	0.001 .. 65		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	ComMChannelId {COMM_CHANNEL_ID} [ComM635]		
Description	Channel identification number of the corresponding channel. Shall be harmonized with channel IDs of networkmanagement and the bus interfaces.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: Shall be harmonized with channel IDs of networkmanagement and the bus interfaces.		

Name last modified in: V1.0.0	ComMMainFunctionPeriod {COMM_MAIN_FUNCTION_PERIOD} [ComM556]		
Description	Specifies the period the MainFunction has to be triggered in seconds. Comment: ComM scheduling shall be at lest as fast as the communication stack and a schedule longer then 100ms makes no sense for communication.		
Multiplicity	1		
Type	FloatParamDef		
Range	0.004 .. 0.100		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name last modified in: V1.0.0	ComMNoCom {COMM_NO_COM} [ComM571]		
Description	Defines if a ECU is not allowed to change state of the channel to ”Silent Communication” or ”Full Communication”. TRUE - Enabled (not allowed to switch to ”Silent Communication mode” or ”Full Communication mode”) Range FALSE - disabled (default)		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: module dependency: Shall not be stored none volatile (shall be reseted to default after initialization). Can be changed during runtime with ComM_LimitChannelToNoComMode().		

Name last modified in: V1.0.0	ComMNoFullCom {COMM_NO_FULL_COM} [ComM570]		
Description	Defines if a ECU is not allowed to change state of the channel to ”Full Communication”. TRUE - Enabled (not allowed to switch to ”Full Communication mode”) Range FALSE - disabled (default)		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: module dependency: Shall not be stored none volatile (shall be reseted to default after initialization). Can be changed during runtime with ComM_LimitChannelToSilentComMode().		

Name last modified in: V1.0.0	ComMNoWakeup {COMM_NO_WAKEUP} [ComM569]		
Description	Defines if a ECU is not allowed to wake-up the channel. TRUE - Enabled (not allowed to wake-up)) FALSE - disabled (default)		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: module dependency: Shall be stored none volatile (value must be kept during a reset). Can be changed during runtime with ComM_PreventWakeUp().		

Name last modified in: V1.1.0	ComMNvmBlockDescriptor {COMM_NvmBlockDescriptor} [ComM670]		
Description	Reference to NVRAM block containing the none volatile data		
Multiplicity	1		
Type	Symbolic name reference to NvmBlockDescriptor		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: Derived from Nvm configuration		

Name last modified in: V1.0.0	ComMRestartTrialsL1 {COMM_RESTART_TRIALS_L1} [ComM576]		
Description	Amount of restarts during level1 until switching to level 2.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 255		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	ComMRxIPduGroup {COMM_RX_IPDU_GROUP} [ComM614]		
Description	ComIPduGroupHandleId of COM IPDU group containing all received IPDUs of the corresponding channel. In milliseconds.		
Multiplicity	1		
Type	Symbolic name reference to ComIPduGroup		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: ECU		

Name last modified in: V1.0.0	ComMStartTxTimeout {COMM_START_TX_TIMEOUT} [ComM572]		
Description	Maximum time in seconds between start transmission request and the confirmation.		
Multiplicity	1		
Type	FloatParamDef		
Range	0.001 .. 65		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	ComMStopTxTimeout {COMM_STOP_TX_TIMEOUT} [ComM573]		
Description	Maximum time in seconds between stop transmission request and the confirmation.		
Multiplicity	1		
Type	FloatParamDef		
Range	0.001 .. 65		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	ComMTxIPduGroup {COMM_TX_IPDU_GROUP} [ComM613]		
Description	ComIPduGroupHandleId of COM IPDU group containing all transmitted IPDUs of the corresponding channel. In milliseconds.		
Multiplicity	1		
Type	Symbolic name reference to ComIPduGroup		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	X	All Variants
Scope / Dependency	scope: ECU		

Included Containers

Container Name	Multiplicity	Scope / Depedency
ComMNetwork Management	1	
ComMUserPerChannel	1..*	

5.5.3 ComMNetworkManagement

The `ComMNetworkManagement` subcontainer contains parameters for the network management for each channel.

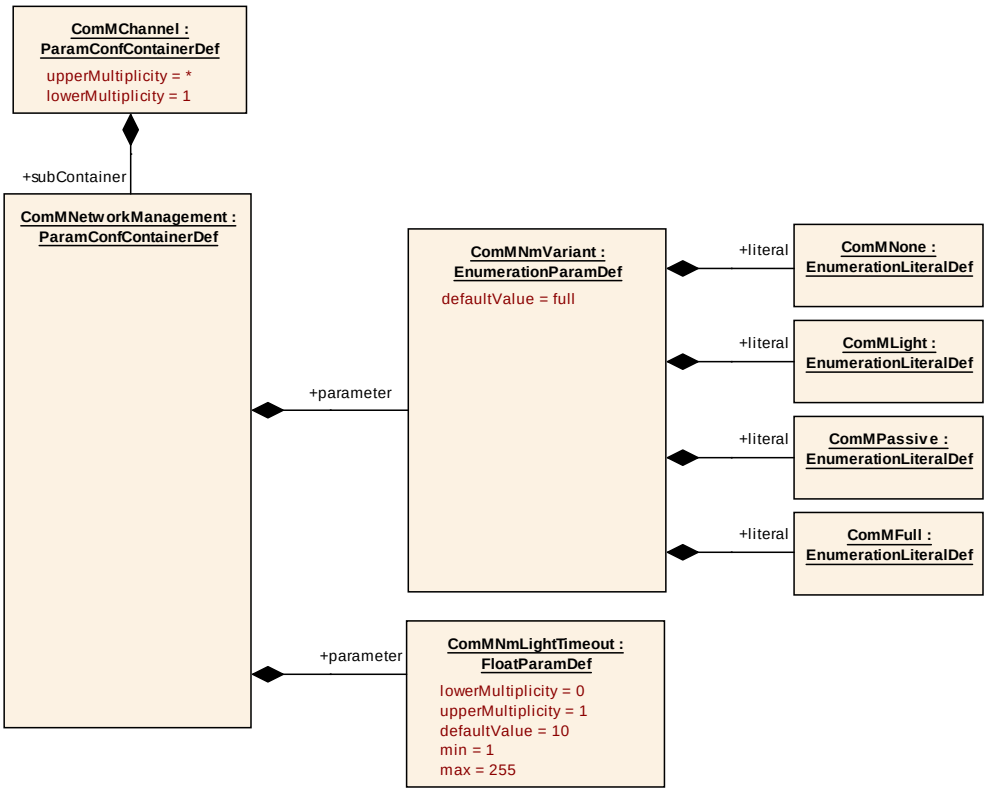


Figure 5.57: ComMNetworkManagement configuration parameters

ComMNetworkManagement

SWS Item	[ComM578]		
Container Name	ComMNetworkManagement {Networkmanagement}		
Description	This container contains the configuration parameters of the networkmanagement.		
Configuration Parameters			

Name last modified in: V1.0.0	ComMNmLightTimeout {COMM_NM_LIGHT_TIMEOUT} [ComM606]		
Description	Defines the timeout (in seconds) after state ”ready sleep” is left.		
Multiplicity	0..1		
Type	FloatParamDef		
Range	1 .. 255		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module dependency: ComMNmVariant configured as ComMLight		

Name last modified in: V1.1.0	ComMNmVariant {COMM_NM_VARIANT} [ComM568]		
Description	Defines the functionality of the networkmanagement.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	ComMFull	AUTOSAR NM available (default).	
	ComMLight	No AUTOSAR NM available but functionality to shut down a channel.	
	ComMNone	No NM available	
	ComMPassive	AUTOSAR NM available (default).	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module dependency: ComMNmVariant shall be ComMNone if ComMBusType = ComMBusTypeInternal		
No Included Containers			

5.5.4 ComMUser

The Com Manager configuration contains a global set of `ComMUser`s and a set of `ComMUser`s per channel. A `ComMUser` is someone (normally a SW-C) that uses the Communication manager. What users that are related to each channel is defined using references between global and channel specific user definitions.

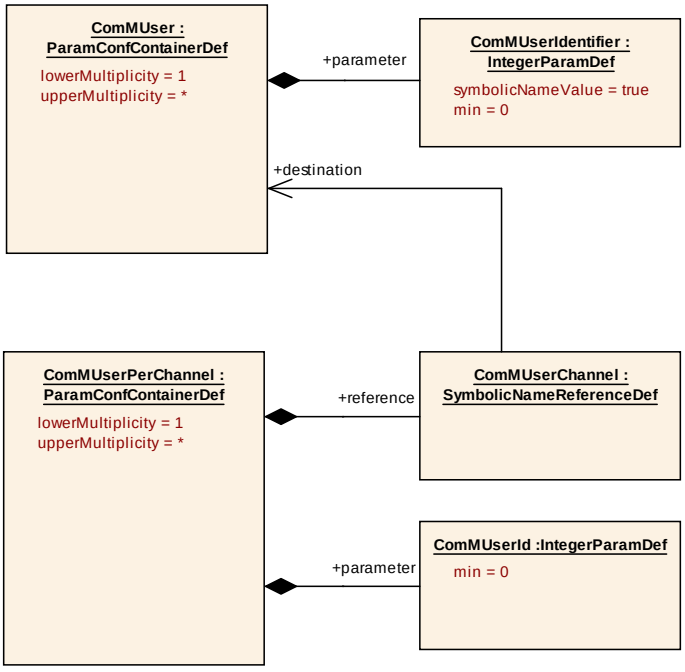


Figure 5.58: ComMUser configuration parameters

ComMUser

SWS Item	[ComM653]
Container Name	ComMUser {CommunicationManagerUser}
Description	This container contains a list of identifiers that are needed to refer to a user in the system which is designated to request Communication modes.
Configuration Parameters	

Name last modified in: V1.1.0	ComMUserIdentifier {COMM_USER} [ComM654]		
Description	A identifier that is needed to refer to a user in the system which is designated to request Communication modes.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module dependency: The concept of users is very similar to the concept of requestors in the ECU State Manager specification. These two parameters shall be harmonized during the configuration process.		
No Included Containers			

5.6 CRC Routines

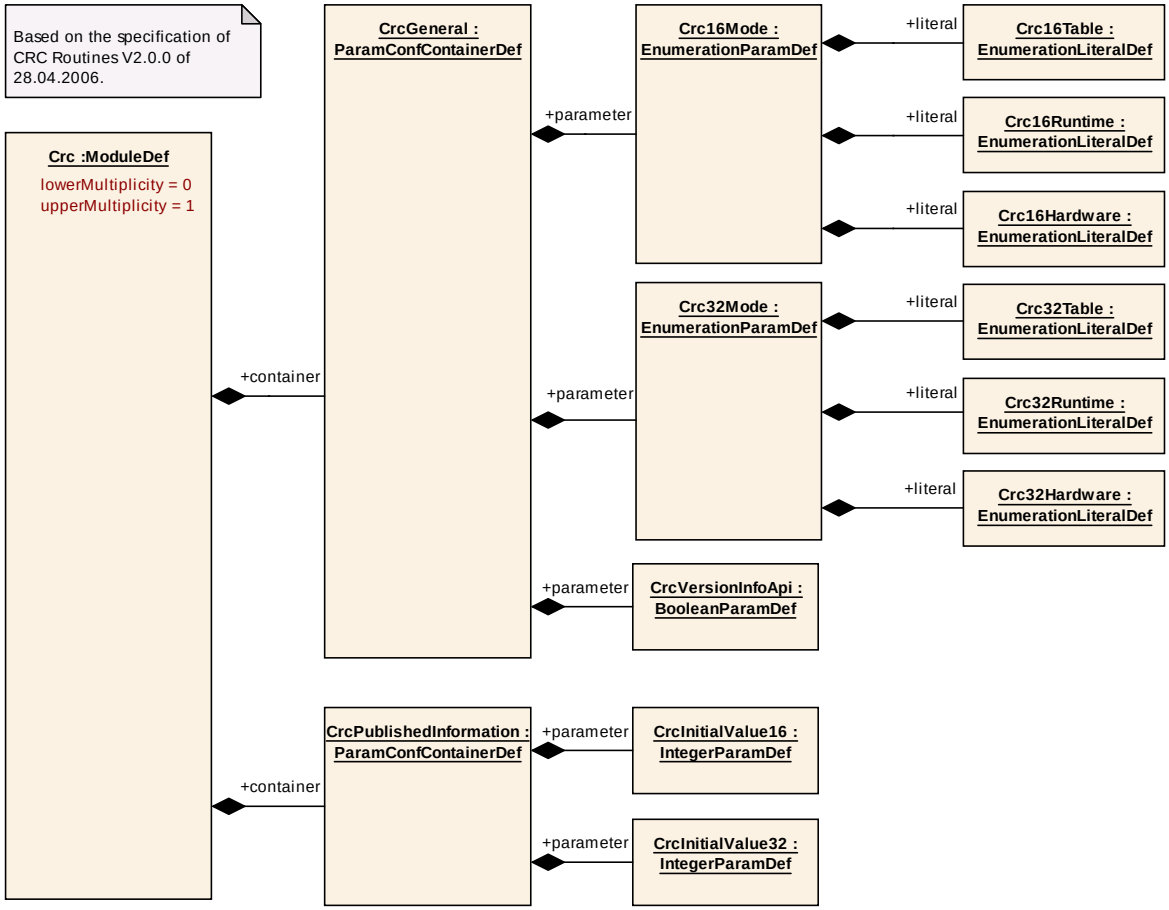


Figure 5.59: CRC routines configuration parameters

Module Name	Crc	
Module Description	Configuration of the Crc (Crc routines) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
CrcGeneral	1	General configuration of CRC module
CrcPublishedInformation	1	Additional published parameters not covered by CommonPublishedInformation container. Note that these parameters do not have any configuration class setting, since they are published information.

CrcGeneral

SWS Item	[CRC006]		
Container Name	CrcGeneral {CRC_COMMON}		
Description	General configuration of CRC module		
Configuration Parameters			

Name last modified in: V1.0.0	Crc16Mode {CRC_16_MODE}		
Description	Switch to select one of the available CRC16 calculation methods.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	Crc16Hardware	hardware based CRC16 calculation	
	Crc16Runtime	runtime based CRC16 calculation	
	Crc16Table	table based CRC16 calculation (default selection)	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	Crc32Mode {CRC_32_MODE}		
Description	Switch to select one of the available CRC32 calculation methods.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	Crc32Hardware	hardware based CRC32 calculation	
	Crc32Runtime	runtime based CRC32 calculation	
	Crc32Table	table based CRC32 calculation (default selection)	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CrcVersionInfoApi {CRC_VERSION_INFO_API}		
Description	Pre-processor switch to enable / disable the API to read out the modules version information. ON Version info API enabled. OFF Version info API disabled,		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

CrcPublishedInformation

SWS Item	[CRC004]		
Container Name	CrcPublishedInformation		
Description	Additional published parameters not covered by CommonPublishedInformation container. Note that these parameters do not have any configuration class setting, since they are published information.		
Configuration Parameters			

Name last modified in: V1.0.0	CrcInitialValue16 {CRC_INITIAL_VALUE16}		
Description	Initial value for the 16-bit CRC calculation.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	CrcInitialValue32 {CRC_INITIAL_VALUE32}		
Description	Initial value for the 32-bit CRC calculation.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

5.7 Crypto Routines

TODO *This BSW module is not part of the current release review cycle.*

5.8 Interpolation Routines

TODO *This BSW module is not part of the current release review cycle.*

5.9 Development Error Tracer

Due to the specification of the Det (see [11]) only the API of the Det is specified. Therefore there are no standardized configuration parameters available for the Det.

To allow the introduction of vendor-specific configuration parameters, only the module definition (`ModuleDef`) of the Det is specified in the AUTOSAR standardized configuration parameter definition. Inside this module definition vendors of the Det have to put their configuration parameter definitions to be compatible with the ECU Configuration Specification (see [7]).

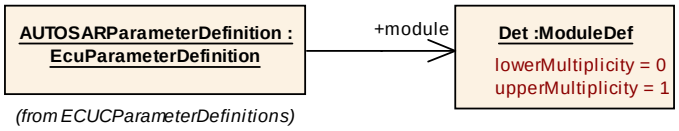


Figure 5.60: Det configuration module definition

Det

Module Name	Det	
Module Description	Configuration of the Det (Development Error Tracer) module. There are NO standardized configuration parameters defined. All the configuration of the Det is done via vendor-specific configuration parameters which need to be defined inside this ModuleDef.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.

5.10 External Time Service

TODO *This BSW module is not part of the current release review cycle.*

6 I/O Services

This chapter defines the configuration parameters of the AUTOSAR BSW I/O modules. These modules include the I/O Drivers and the IO hardware abstraction, as shown in figure 6.1.

The chapter currently also includes the PORT driver. The top level configuration model is shown in figure 6.2

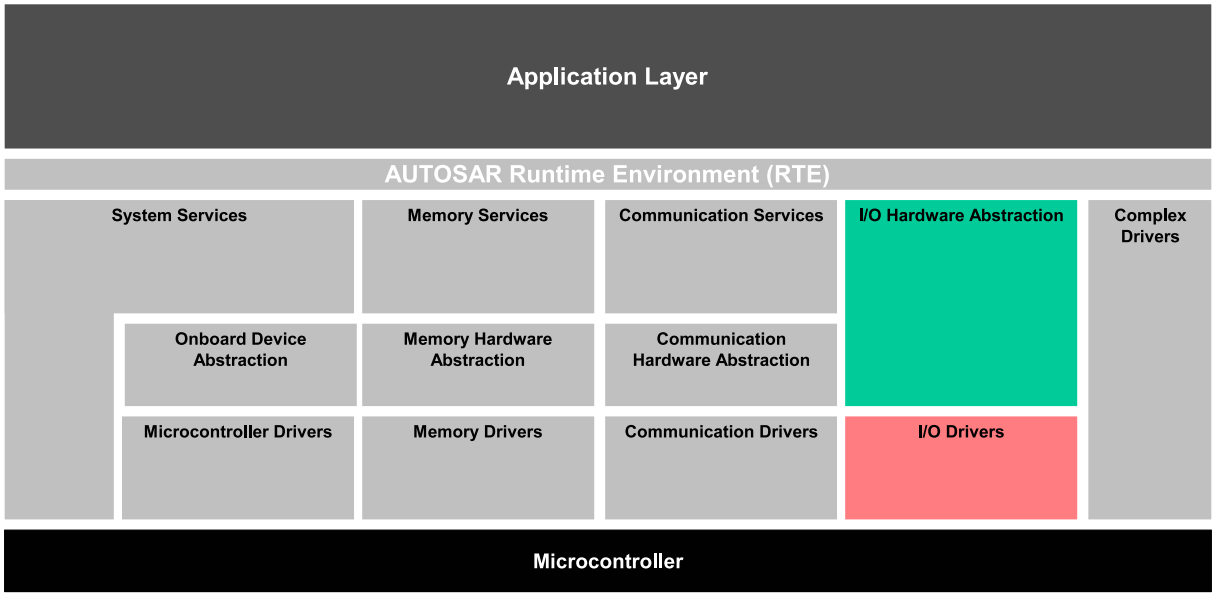


Figure 6.1: IO modules in the BSW architecture

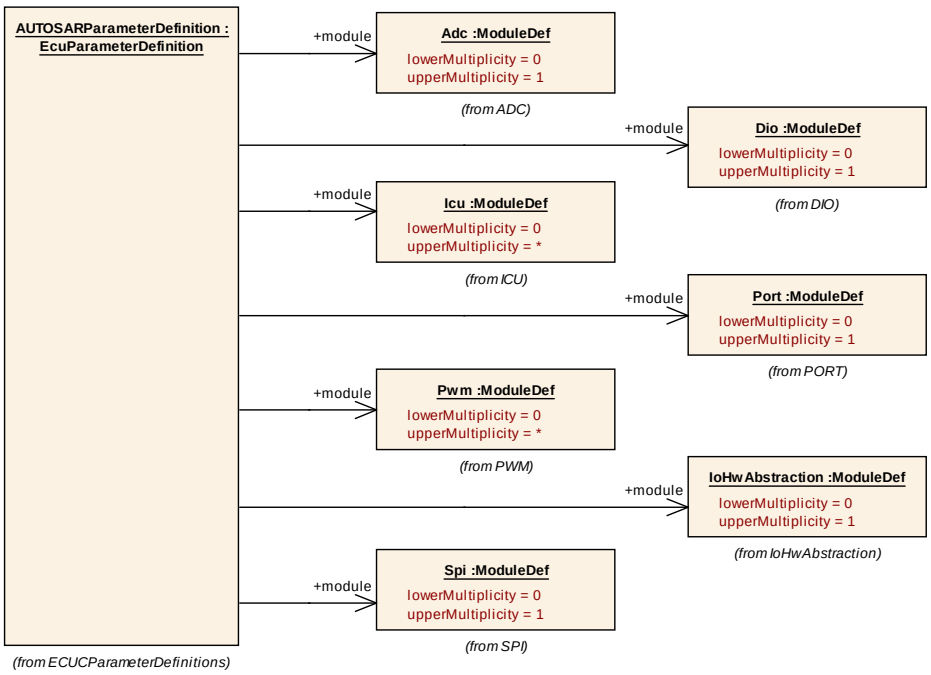


Figure 6.2: Overview: IO configuration

6.1 I/O Hardware Abstraction

IO HW Abstraction is specified in [22]. Aim of the IO Hardware Abstraction module is to make data transiting through the RTE fully independent of the ECU hardware. This means that the Software Component designer doesn't need anymore to have the know-how how signals are affected on the physical level. Thus, this module is ECU specific.

An overview of the configuration model with all containers for the IO HW Abstraction module is shown in figure 6.3.

Module Name	IoHwAbstraction	
Module Description	Configuration of IO HW Abstraction.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
IoHwAbEcuSignalGroup	0..*	This container contains the configuration (parameters) of all ECU signals groups implemented in the IO Hardware Abstraction Module
IoHwAbEcuSignals	0..*	This container contains the configuration (parameters) of all ECU signals implemented in the IO Hardware Abstraction Module. Remarks: - The IO Hardware Abstraction module is an ECU firmware module. That means that the implementation shall fit only to the ECU hardware. The implementer has to develop only the functionality needed (eg: if there is no pulse width ECU signal configurated, the pulse width abstraction will not be available. - At least one ECU signal shall be configurated to have the module.
IoHwAbGeneral	1	This container contains common description for the IO Hardware Abstraction module

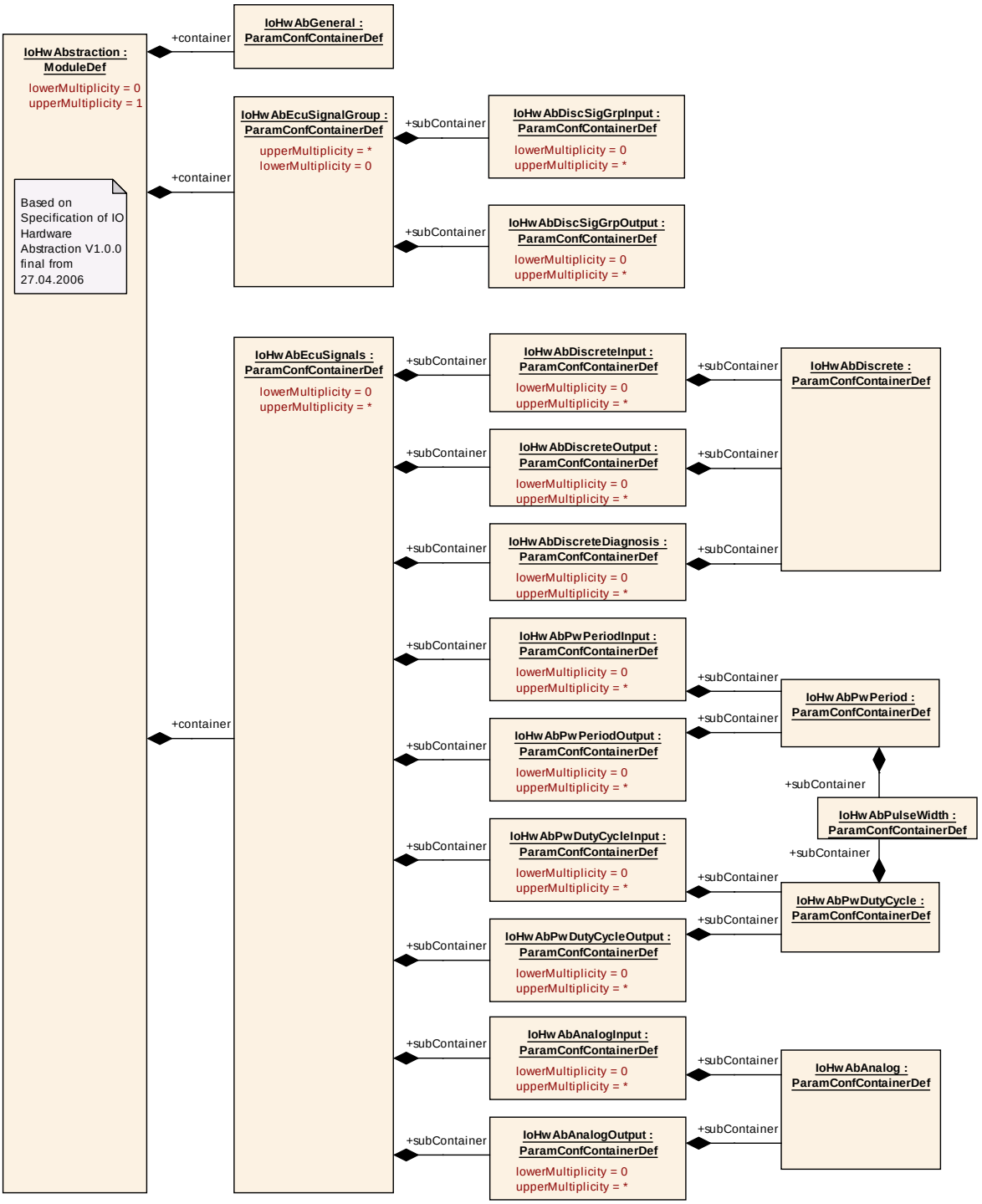


Figure 6.3: IO HW abstraction overview

6.1.1 Generic configuration

Generic configuration of the module are collected in container `IoHwAbGeneral`, which is shown in figure 6.4.

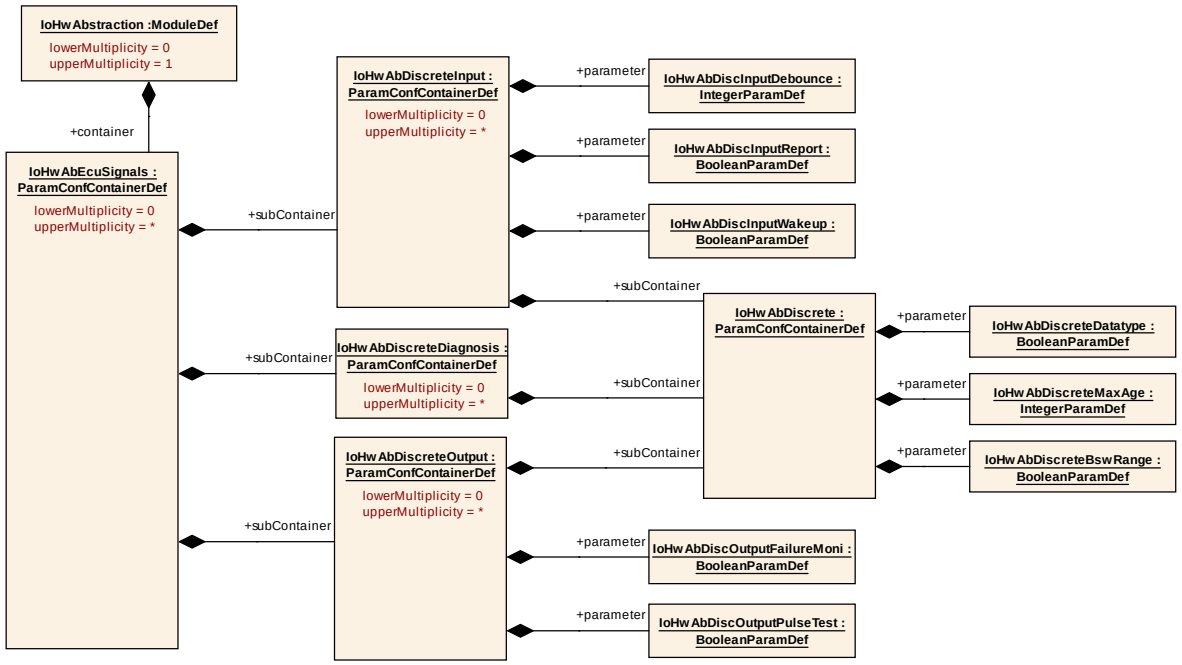


Figure 6.4: General IO HW abstraction module configuration

IoHwAbGeneral

SWS Item	[IoHwAb096]
Container Name	IoHwAbGeneral {IoHwAb_Configuration}
Description	This container contains common description for the IO Hardware Abstraction module
Configuration Parameters	

Name last modified in: V1.0.0	IoHwAbDevErrorDetect		
Description	Switches the Development Error Detection and Notification ON or OFF		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	IoHwAbVersionInfoApi {IOHWAB_VERSION_INFO_API}		
Description	Pre-processor switch to enable / disable the API to read out the modules version information.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

6.1.2 Signal configuration

All signal related configuration is done in container `IoHwAbEcuSignals`.

IoHwAbEcuSignals

SWS Item	[IoHwAb087]	
Container Name	IoHwAbEcuSignals {IoHwAb_ECUSignals_Configuration}	
Description	This container contains the configuration (parameters) of all ECU signals implemented in the IO Hardware Abstraction Module. Remarks: - The IO Hardware Abstraction module is an ECU firmware module. That means that the implementation shall fit only to the ECU hardware. The implementer has to develop only the functionality needed (eg: if there is no pulse width ECU signal configurated, the pulse width abstraction will not be available. - At least one ECU signal shall be configurated to have the module.	
Configuration Parameters		
Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
IoHwAbAnalogInput	0..*	
IoHwAbAnalogOutput	0..*	
IoHwAbDiscrete Diagnosis	0..*	
IoHwAbDiscreteInput	0..*	
IoHwAbDiscreteOutput	0..*	
IoHwAbPwDutyCycle Input	0..*	
IoHwAbPwDutyCycle Output	0..*	
IoHwAbPwPeriodInput	0..*	
IoHwAbPwPeriodOutput	0..*	

6.1.2.1 Signal groups

Several discrete signals can be grouped to form a signal group, as shown in figure 6.5

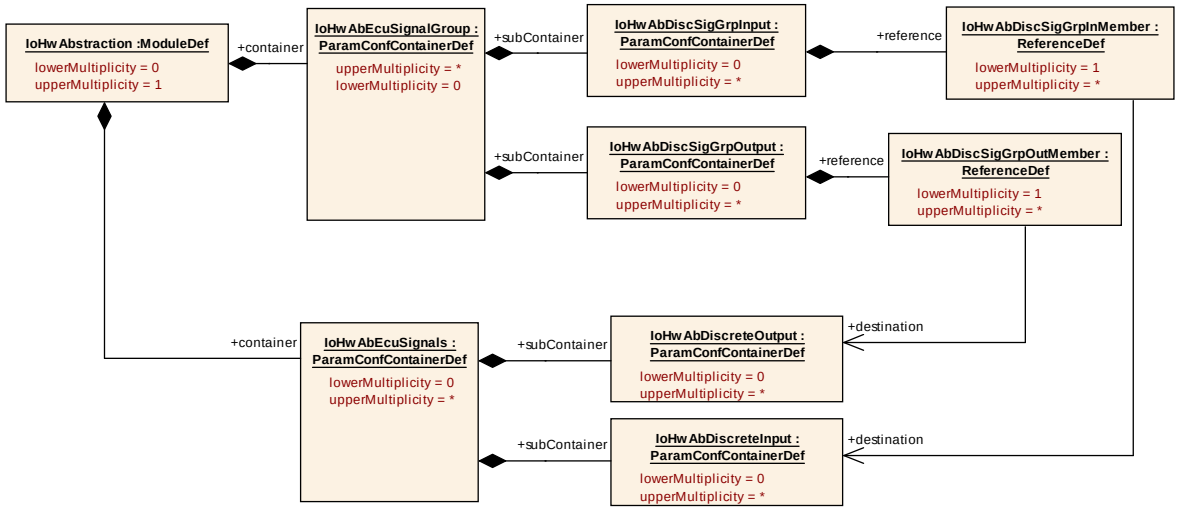


Figure 6.5: Configuration of signal groups

IoHwAbEcuSignalGroup

SWS Item	[IoHwAb087]	
Container Name	IoHwAbEcuSignalGroup {IoHwAb_ECUSignalGroups_Configuration}	
Description	This container contains the configuration (parameters) of all ECU signals groups implemented in the IO Hardware Abstraction Module	
Configuration Parameters Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
IoHwAbDiscSigGrpInput	0..*	
IoHwAbDiscSigGrp Output	0..*	

Input signal group specific parameters are collected in container `IoHwAbDiscSigGrpInput`.

IoHwAbDiscSigGrpInput

SWS Item	[IoHwAb101]		
Container Name	IoHwAbDiscSigGrpInput {IoHwAb_Discrete_SignalGroup_Inputs}		
Description	This container contains the description of a discrete ECU signal group. The container IoHwAb_Discrete_SignalGroup contains several IoHwAb_Class_Discrete_Input sub-containers		
Configuration Parameters			

Name last modified in: V1.0.0	IoHwAbDiscSigGrpInMember		
Description	Each member of the signal group is referenced by an instance of tthis element. In the IO HW Abstraction SWS, individual signals are aggregated by the signal group instead of referenced, but by using references it is still possible to have the same signal being par of several signal groups, as intended in the SWS.		
Multiplicity	1..*		
Type	Reference to IoHwAbDiscreteInput		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
No Included Containers			

Output signal group specific parameters are collected in container `IoHwAbDiscSigGrpOutput`.

IoHwAbDiscSigGrpOutput

SWS Item	[IoHwAb103]		
Container Name	IoHwAbDiscSigGrpOutput {IoHwAb_Discrete_SignalGroup_Outputs}		
Description	This container contains the description of a discrete ECU signal group. The container IoHwAb_Discrete_SignalGroup contains several IoHwAb_Class_Discrete_Output sub-containers.		
Configuration Parameters			

Name last modified in: V1.0.0	IoHwAbDiscSigGrpOutMember		
Description	Each member of the signal group is referenced by an instance of tthis element. In the IO HW Abstraction SWS, individual signals are aggregated by the signal group instead of referenced, but by using references it is still possible to have the same signal being par of several signal groups, as intended in the SWS.		
Multiplicity	1..*		
Type	Reference to IoHwAbDiscreteOutput		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
No Included Containers			

6.1.2.2 Discrete signals

Details for the configuration of discrete signals are shown in figure 6.6.

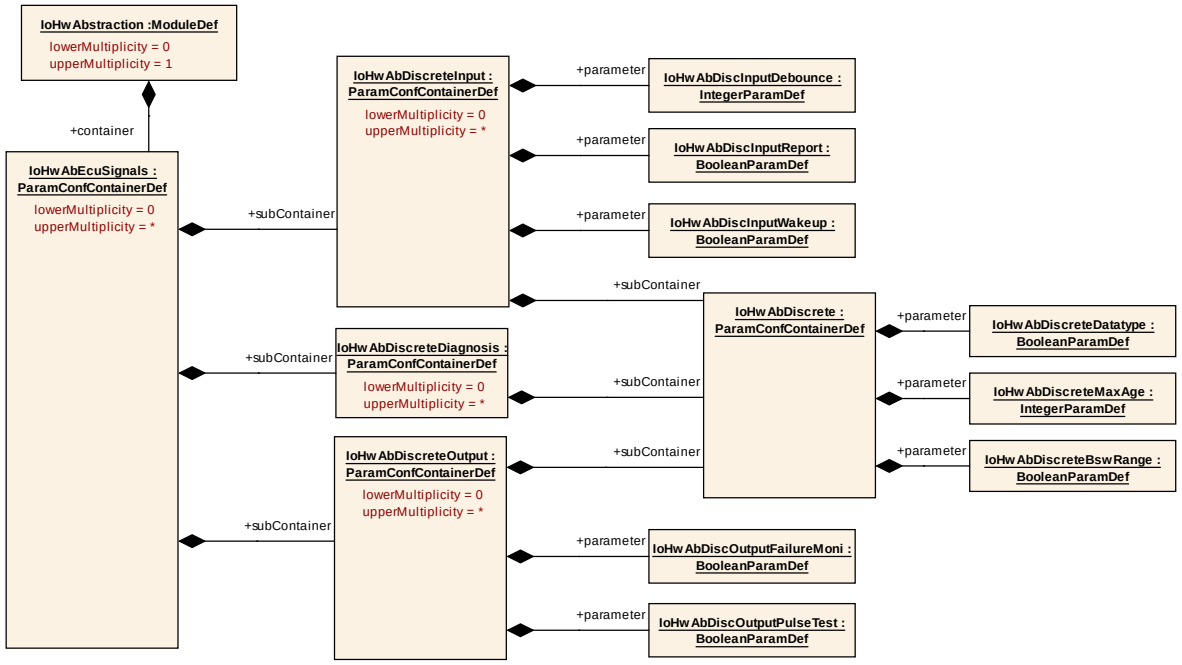


Figure 6.6: Configuration of discrete signals

There is one container `IoHwAbDiscrete` with all common parameters for input, diagnosis, and output signals.

IoHwAbDiscrete

SWS Item	[IoHwAb081]
Container Name	IoHwAbDiscrete {IoHwAb_Class_Discrete}
Description	This container contains common description of a all ECU signals from class discrete
Configuration Parameters	

Name last modified in: V1.0.0	IoHwAbDiscreteBswRange {IOHWAB_CLASS_DISCRETE_BSW_RANGE}		
Description	This parameter gives all possible values of an ECU discrete signal		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	IoHwAbDiscreteDatatype {IOHWAB_CLASS_DISCRETE_DATATYPE}		
Description	This parameter gives the DataType used for a discrete signal		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	IoHwAbDiscreteMaxAge {IOHWAB_CLASS_DISCRETE_MAX_AGE}		
Description	This parameter gives the maximum age of an ECU discrete signal		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
No Included Containers			

Input specific parameters are collected in container `IoHwAbDiscreteInput`.

IoHwAbDiscreteInput

SWS Item	[IoHwAb082]
Container Name	IoHwAbDiscreteInput {IoHwAb_Discrete_Input}
Description	This container contains the description of a discrete ECU signal used as input. The operation available through the Port is an OP_GET.
Configuration Parameters	

Name last modified in: V1.0.0	IoHwAbDiscInputDebounce {IOHWAB_DISCRETE_INPUT_DEBOUNCE}		
Description	This parameter specifies if the ECU discrete signal is filtered / debounced 0 represents Debouncing deactivated N represents Debouncing activated, N values sampled before value is available		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	IoHwAbDiscInputReport {IOHWAB_DISCRETE_INPUT_REPORT}		
Description	This parameter specifies if the report features is activated FALSE represents no report if a significant change occurs TRUE represents repot if a significant change occurs		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	IoHwAbDiscInputWakeup {IOHWAB_DISCRETE_INPUT_WAKEUP}		
Description	This parameter specifies if the ECU pin is a wakeup input TRUE represents a wakeup Input. During sleep mode, a change is reported to the module. In case of a valid reason, the ECU state manager will be restarted to validate the wakeup. FALSE represents a simple discrete Input.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
IoHwAbDiscrete	1		

Output specific parameters are collected in container `IoHwAbDiscreteOutput`.

IoHwAbDiscreteOutput

SWS Item	[IoHwAb083]		
Container Name	IoHwAbDiscreteOutput {IoHwAb_Discrete_Output}		
Description	This container contains the description of a discrete ECU signal used as Output. The operation available through the Port is a OP_SET.		
Configuration Parameters			

Name last modified in: V1.0.0	IoHwAbDiscOutputFailureMoni {IOHWAB_DISCRETE_OUTPUT_FAILURE_MONITORING}		
Description	This parameter specifies if the ECU discrete Output is monitored, to detect failure TRUE represents that the ECU Signal is a monitored Output. Failures on this Output are detected and managed by the module FALSE represents that the ECU Signal is a simple discrete Output.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	IoHwAbDiscOutputPulseTest {IOHWAB_DISCRETE_OUTPUT_PULSE_TEST}		
Description	This parameter specifies if an test pulse is used to detect failure on the ECU discrete Output TRUE represents that a Test Pulse is used for failure monitoring FALSE represents that no Test Pulse is used for failure monitoring		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
IoHwAbDiscrete	1		

Diagnosis specific parameters are collected in container `IoHwAbDiscreteDiagnosis`.

IoHwAbDiscreteDiagnosis

SWS Item	[IoHwAb099]	
Container Name	IoHwAbDiscreteDiagnosis {IoHwAb_Discrete_Diagnosis}	
Description	This container contains the description of a discrete ECU signal used as diagnosis.	
Configuration Parameters		
Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
IoHwAbDiscrete	1	

6.1.2.3 Analog signals

Details for the configuration of analog signals are shown in figure 6.7.

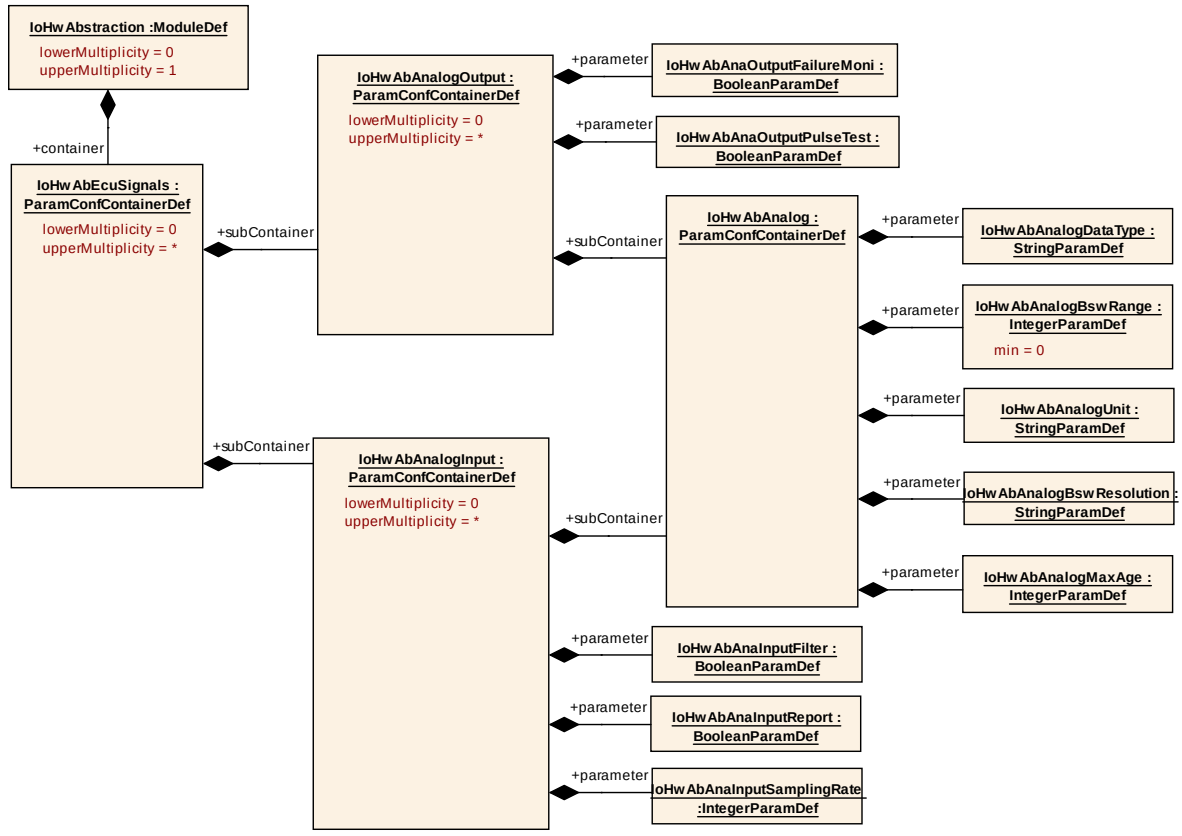


Figure 6.7: Configuration of analog signals

`IoHwAbAnalog` is a container used by both `IoHwAbAnalogInput` and `IoHwAbAnalogOutput`.

IoHwAbAnalog

SWS Item	[IoHwAb084]
Container Name	IoHwAbAnalog {IoHwAb_Class_Analog}
Description	This container contains common description of a all ECU signals from class analog.
Configuration Parameters	

Name last modified in: V1.0.0	IoHwAbAnalogBswRange {IOHWAB_CLASS_ANALOG_BSW_RANGE}		
Description	This parameter gives the BSW Range used for an analog signal, according to the datatype		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	IoHwAbAnalogBswResolution {IOHWAB_CLASS_ANALOG_BSW_RESOLUTION}		
Description	This parameter gives the resolution used for an analog signal		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	IoHwAbAnalogDataType {IOHWAB_CLASS_ANALOG_DATATYPE}		
Description	This parameter gives the DataType used for an analog signal		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	IoHwAbAnalogMaxAge {IOHWAB_CLASS_ANALOG_MAX_AGE}		
Description	This parameter gives the maximum age of an ECU analog signal		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	IoHwAbAnalogUnit {IOHWAB_CLASS_ANALOG_UNIT}		
Description	This parameter gives the unit used for an analog signal		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
No Included Containers			

Input specific parameters are collected in container `IoHwAbAnalogInput`.

IoHwAbAnalogInput

SWS Item	[IoHwAb085]
Container Name	IoHwAbAnalogInput {IoHwAb_Analog_Input}
Description	This container contains the description of an analog ECU signal used as input. The operation available through the Port is a OP_GET.
Configuration Parameters	

Name last modified in: V1.0.0	IoHwAbAnaInputFilter {IOHWAB_ANALOG_INPUT_FILTER}		
Description	This parameter specifies if the ECU analog signal is filtered		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	IoHwAbAnaInputReport {IOHWAB_ANALOG_INPUT_SAMPLING_RATE}		
Description	This parameter specifies if the report features is activated		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	IoHwAbAnaInputSamplingRate {IOHWAB_ANALOG_INPUT_SAMPLING_RATE}		
Description	This parameter specifies the sampling rate for an analog input		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
IoHwAbAnalog	1		

Output specific parameters are collected in container `IoHwAbAnaloOutput`.

IoHwAbAnalogOutput

SWS Item	[IoHwAb086]		
Container Name	IoHwAbAnalogOutput {IoHwAb_Analog_Output}		
Description	This container contains the description of an analog ECU signal used as output. The operation available through the Port is a OP_SET.		
Configuration Parameters			

Name last modified in: V1.0.0	IoHwAbAnaOutputFailureMoni {IOHWAB_ANALOG_OUTPUT_FAILURE_MONITORING}		
Description	This parameter specifies if the ECU analog Output is monitored, to detect failure		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	IoHwAbAnaOutputPulseTest {IOHWAB_ANALOG_OUTPUT_PULSETEST}		
Description	This parameter specifies if an test pulse is used to detect failure on the ECU analog Output		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
IoHwAbAnalog	1	

6.1.2.4 Pulse Width signals

The configuration of pulse width signals is shown in figure 6.8.

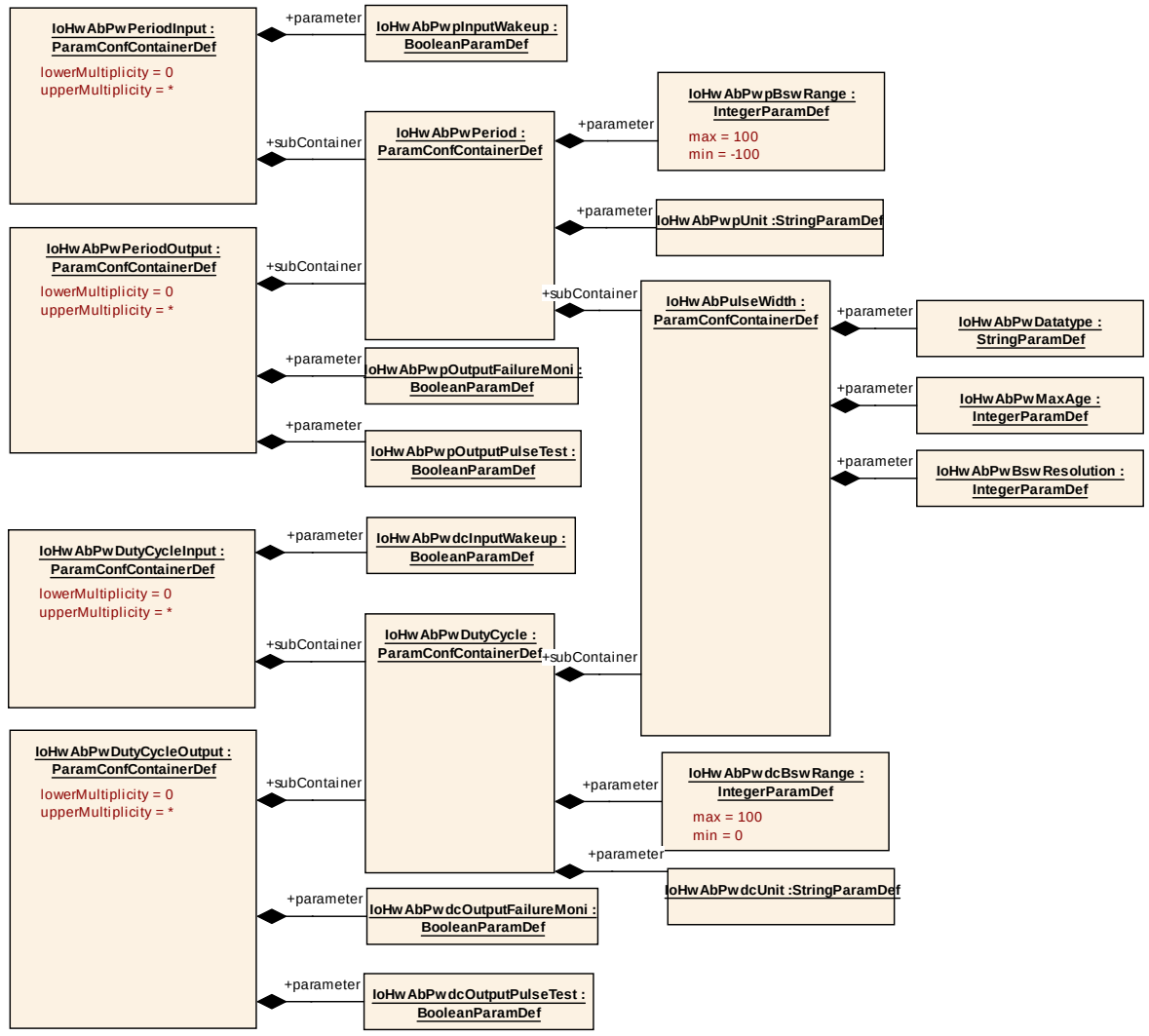


Figure 6.8: Configuration of pulse width signals

Parameters common to both period and pulse width signals are collected in `IoHwAbPulseWidth`, which is aggregated by both `IoHwAbPwPeriod` and `IoHwAbPwDutyCycle`.

IoHwAbPulseWidth

SWS Item	[IoHwAb088]		
Container Name	IoHwAbPulseWidth {IoHwAb_Class_PulseWidth}		
Description	This container contains common description of a all ECU signals from class pulse width		
Configuration Parameters			
Name last modified in: V1.0.0	IoHwAbPwBswResolution {IOHWAB_CLASS_PULSEWIDTH_BSW_RESOLUTION}		
Description	This parameter gives the resolution used for a pulse width ECU signal		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
Name last modified in: V1.0.0	IoHwAbPwDatatype {IOHWAB_CLASS_PULSEWIDTH_BSW_RESOLUTION}		
Description	This parameter gives the DataType used for a pulse width signal		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
Name last modified in: V1.0.0	IoHwAbPwMaxAge {IOHWAB_CLASS_PULSEWIDTH_MAX_AGE}		
Description	This parameter gives the maximum age for a pulse width ECU signal		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
No Included Containers			

Parameters common to period input and output signals are collected in `IoHwAbPwPeriod` which is included in containers `IoHwAbPwPeriodInput` and `IoHwAbPwPeriodOutput`.

IoHwAbPwPeriod

SWS Item	[IoHwAb089]		
Container Name	IoHwAbPwPeriod {IoHwAb_Class_PulseWidthPeriod}		
Description	This container contains common description of a all ECU signals from class pulse width, handling especially the characteristic period		
Configuration Parameters			
Name last modified in: V1.0.0	IoHwAbPwpBswRange {IOHWAB_CLASS_PULSEWIDTHPERIOD_BSW_RANGE}		
Description	This parameter gives the BSW Range for a pulse width signal (in percent)		
Multiplicity	1		
Type	IntegerParamDef		
Range	-100 .. 100		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	IoHwAbPwpUnit {IOHWAB_CLASS_PULSEWIDTHPERIOD_UNIT}		
Description	This parameter gives the unit for an ECU signal, handling especially the characteristic period		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
IoHwAbPulseWidth	1		

Period Input specific parameters are collected in container `IoHwAbPwPeriodInput`.

IoHwAbPwPeriodInput

SWS Item	[IoHwAb090]		
Container Name	IoHwAbPwPeriodInput {IoHwAb_PulseWidthPeriodInput}		
Description	This container contains specific description for an input ECU signal from class pulse width, handling especially the characteristic period		
Configuration Parameters			
Name last modified in: V1.0.0	IoHwAbPwpInputWakeup {IOHWAB_PulseWidthPeriod_Wakeup}		
Description	This parameter specifies if the ECU pin is a wakeup input		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
IoHwAbPwPeriod	1	

Period Output specific parameters are collected in container `IoHwAbPwPeriodOutput`.

IoHwAbPwPeriodOutput

SWS Item	[IoHwAb091]		
Container Name	IoHwAbPwPeriodOutput {IoHwAb_PulseWidthPeriodOutput}		
Description	This container contains specific description for an output ECU signal from class pulse width, handling especially the characteristic period The operation available through the Port is a OP_SET.		
Configuration Parameters			
Name last modified in: V1.0.0	IoHwAbPwpOutputFailureMoni {IOHWAB_PulseWidthPeriod_Output_Failure_Monitoring}		
Description	This parameter specifies if the ECU Output is monitored, to detect failure		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	IoHwAbPwpOutputPulseTest {IOHWAB_PulseWidthPeriod_Output_PulseTest}		
Description	This parameter specifies if an test pulse is used to detect failure on the ECU Output		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
IoHwAbPwPeriod	1	

Parameters common to duty cycle input and output signals are collected in `IoHwAbPwDutyCycle` which is included in containers `IoHwAbPwDutyCycleInput` and `IoHwAbPwDutyCycleOutput`.

IoHwAbPwDutyCycle

SWS Item	[IoHwAb092]		
Container Name	IoHwAbPwDutyCycle {IoHwAb_Class_PulseWidthDutyCycle}		
Description	This container contains common description of a all ECU signals from class pulse width, handling especially the characteristic duty cycle		
Configuration Parameters			
Name last modified in: V1.0.0	IoHwAbPwdcBswRange {IOHWAB_CLASS_PULSEWIDTHDUTYCYCLE_BSW_RANGE}		
Description	This parameter gives the BSW Range for a pulse width signal		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 100		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	IoHwAbPwdcUnit {IOHWAB_CLASS_PULSEWIDTHDUTYCYCLE_UNIT}		
Description	This parameter gives the unit for an ECU signal, handling especially the characteristic period		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
IoHwAbPulseWidth	1		

Duty Cycle Input specific parameters are collected in container `IoHwAbPwDutyCycleInput`.

IoHwAbPwDutyCycleInput

SWS Item	[IoHwAb093]		
Container Name	IoHwAbPwDutyCycleInput {IoHwAb_PulseWidthDutyCycleInput}		
Description	This container contains specific description for an input ECU signal from class pulse width, handling especially the characteristic DutyCycle The operation available through the Port is a OP_GET		
Configuration Parameters			
Name last modified in: V1.0.0	IoHwAbPwdcInputWakeup {IOHWAB_PULSEWIDTHDUTYCYCLE_INPUT_WAKEUP}		
Description	This parameter specifies if the ECU pin is a wakeup input		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
IoHwAbPwDutyCycle	1		

Duty Cycle Output specific parameters are collected in container `IoHwAbPwDutyCycleInput`.

IoHwAbPwDutyCycleOutput

SWS Item	[IoHwAb094]		
Container Name	IoHwAbPwDutyCycleOutput {IoHwAb_PulseWidthDutyCycleOutput}		
Description	This container contains specific description for an output ECU signal from class pulse width, handling especially the characteristic DutyCycle. The operation available through the Port is a OP_SET.		
Configuration Parameters			
Name last modified in: V1.0.0	IoHwAbPwdcOutputFailureMoni {IOHWAB_CLASS_PULSEWIDTHDUTYCYCLE_OUTPUT_FAILURE_MONITORING}		
Description	This parameter specifies if the ECU Output is monitored, to detect failure		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
Name last modified in: V1.0.0	IoHwAbPwdcOutputPulseTest {IOHWAB_CLASS_PULSEWIDTHDUTYCYCLE_OUTPUT_PULSETEST}		
Description	This parameter specifies if an test pulse is used to detect failure on the ECU Output		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
IoHwAbPwDutyCycle	1	

6.2 Port driver

The PORT driver module is specified in [35]. It provides the service for initializing the whole PORT structure of the microcontroller. In most microcontroller architectures, ports and port pins can be assigned to various functionalities, e.g. DIO, ADC, PWM, etc.

To avoid initialization conflicts, the AUTOSAR architecture defines a single module that is used to configure and initialize these ports. The base structure of configuration model for the PORT module is shown in figure 6.9.

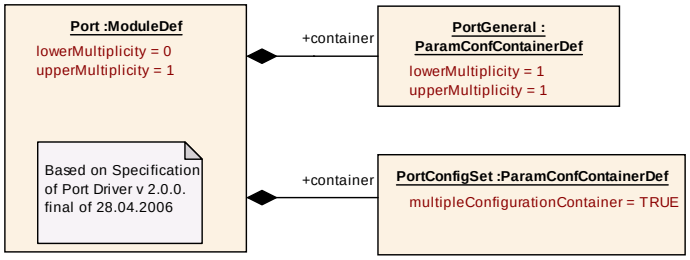


Figure 6.9: Configuration container structure for the PORT module

Module Name	Port	
Module Description	Configuration of the Port module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
PortConfigSet	1	This container is the base of a multiple configuration set
PortGeneral	1	Module wide configuration parameters of the PORT driver.

In container `PortGeneral` (shown in figure 6.10 the general configuration parameters are collected.

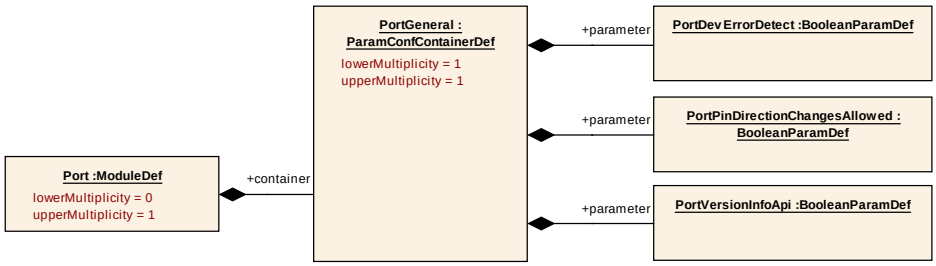


Figure 6.10: General configuration parameters for the PORT module

PortGeneral

SWS Item	[PORT117]		
Container Name	PortGeneral {PORT General configuration}		
Description	Module wide configuration parameters of the PORT driver.		
Configuration Parameters			
Name last modified in: V1.0.0	PortDevErrorDetect {PORT_DEV_ERROR_DETECT}		
Description	Switches the Development Error Detection and Notification ON or OFF.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
Name last modified in: V1.0.0	PortPinDirectionChangesAllowed {PORT_PIN_DIRECTION_CHANGES_ALLOWED}		
Description	Pre-processor switch to enable / disable the use of the function Port_SetPinDirection().		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
Name last modified in: V1.0.0	PortVersionInfoApi {PORT_VERSION_INFO_API}		
Description	Pre-processor switch to enable / disable the API to read out the modules version information.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
No Included Containers			

All port pin related configuration is collected in container `PortPin`, as shown in figure 6.11.

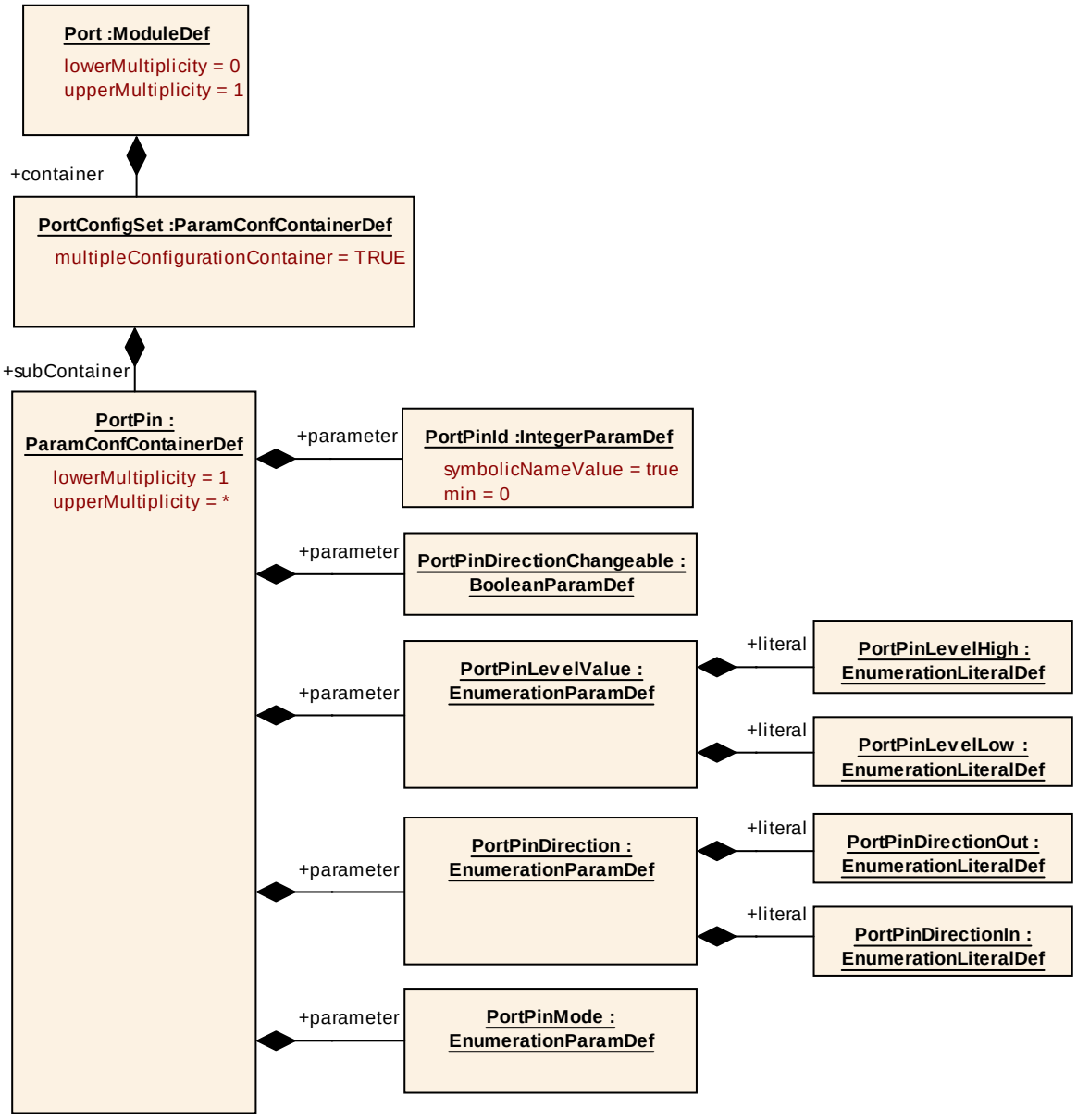


Figure 6.11: Configuration of port pins in the PORT module

PortPin

SWS Item	[PORT118]		
Container Name	PortPin		
Description	Configuration of the individual port pins.		
Configuration Parameters			

Name last modified in: V1.0.0	PortPinDirection {PORT_PIN_DIRECTION}		
Description	The initial direction of the pin (IN or OUT). If the direction is not changeable, the value configured here is fixed. The direction must match the pin mode. E.g. a pin used for an ADC must be configured to be an in port.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	PortPinDirectionIn	Port Pin direction set as input	
	PortPinDirectionOut	Port Pin direction set as output	
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PortPinDirectionChangeable {PORT_PIN_DIRECTION_CHANGEABLE}		
Description	Parameter to indicate if the direction is changeable on a port pin during runtime.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PortPinId		
Description	Pin Id of the port pin. This value will be assigned to the symbolic name derived from the port pin container short name.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PortPinLevelValue {PORT_PIN_LEVEL_VALUE}		
Description	Port Pin Level value from Port pin list.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	PortPinLevelHigh	Port Pin level is High	
	PortPinLevelLow	Port Pin level is LOW	
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PortPinMode {PORT_PIN_MODE}		
Description	Port pin mode from mode list. Note that more than one mode is allowed by default. That way it is e.g. possible to combine DIO with another mode such as ICU. More HW specific literals may be added by module implementor. E.g. SPI may be further broken down into SpiCsHw, SpiCsSw, etc. if necessary.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	PortPinModeAdc	Port Pin used by ADC	
	PortPinModeCan	Port Pin used for CAN	
	PortPinModeDio	Port Pin configured for DIO. It shall be used under control of the DIO driver.	
	PortPinModeDioGpt	Port Pin configured for DIO. It shall be used under control of the general purpose timer driver.	
	PortPinModeDioWdg	Port Pin configured for DIO. It shall be used under control of the watchdog driver.	
	PortPinModeFlexray	Port Pin used for FlexRay	
	PortPinModeIcu	Port Pin used by ICU	
	PortPinModeLin	Port Pin used for LIN	
	PortPinModeMem	Port Pin used for external memory under control of a memory driver.	
	PortPinModePwm	Port Pin used by PWM	
	PortPinModeSpi	Port Pin used by SPI	
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency **No Included Containers**	scope: Module		

Since the configuration and mode of the port pins is microcontroller and ECU dependent, the configuration abstracts from the concrete microcontroller implementation. Instead, the user(s) of the port pin may be selected from a enumeration. The PORT module generator must then translate this information to the concrete microcontroller specific code. The microcontroller hardware will in most cases restrict which users can be chosen for a certain pin. In many cases, several users may be chosen simultaneously. For example, for a port pin used for ICU, DIO may be added as well to perform error checking.

6.3 DIO Driver

The DIO Driver is specified in [16]. It provides services for reading from and writing to

- DIO Ports
- DIO Channel Groups
- DIO Channels.

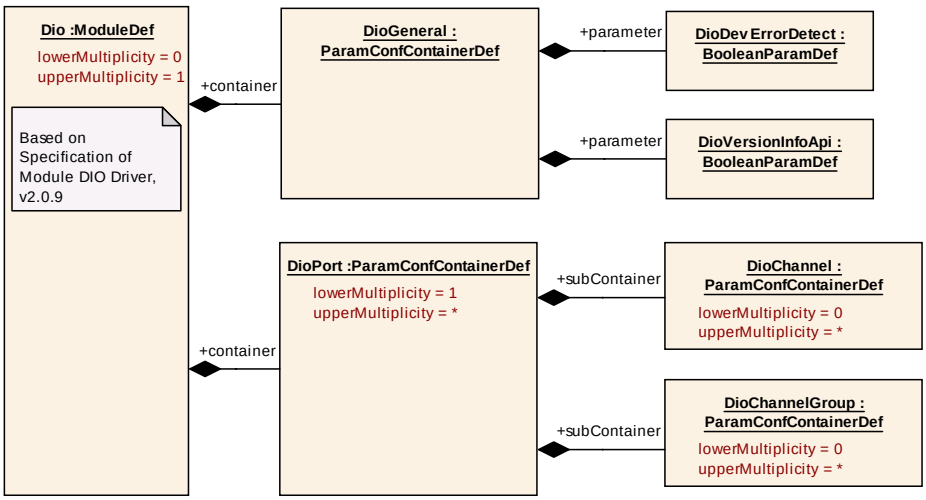


Figure 6.12: Parameter description for the DIO module - overview

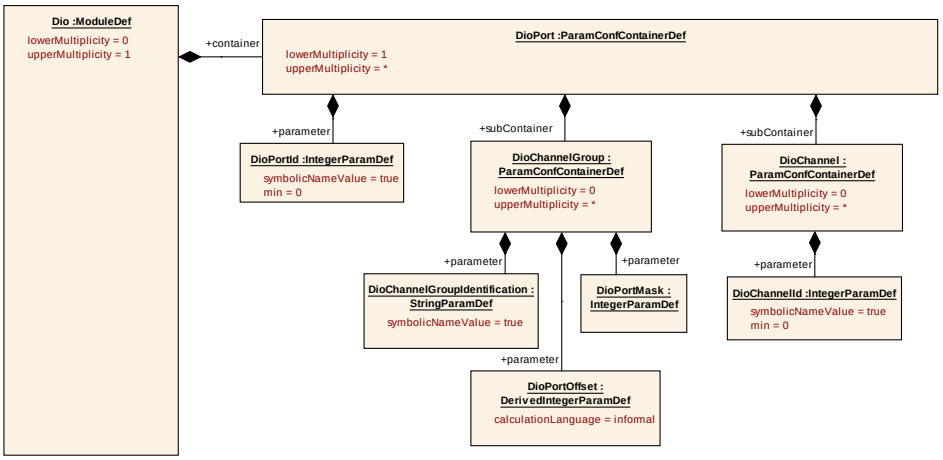


Figure 6.13: Parameter description for the DIO module - port and port pin details

The configuration model for the DIO module is shown in figures 6.12 and 6.13. Several containers are used to model the configuration. The following tables show the detailed descriptions of these containers and their parameters.

The `Dio` module holds on top level the containers `DioGeneral` and `DioOptionalApis` with parameters that apply to the complete DIO configuration, and the container `DioPort` that holds all port, channel and channel group specific configuration.

Module Name	Dio	
Module Description	Configuration of the Dio (Digital IO) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
DioGeneral	1	General DIO module configuration parameters.
DioPort	1..*	Configuration of individual DIO ports, consisting of channels and possible channel groups. Nothe that this container definition does not explicitly define a symbolic name parameter. Instead, the container's short name will be used in the Ecu Configuration Description to specify the symbolic name of the port.

Container `DioGeneral` containers general configuration parameters that affect the complete DIO module.

DioGeneral

SWS Item			
Container Name	DioGeneral		
Description	General DIO module configuration parameters.		
Configuration Parameters			
Name last modified in: V1.0.0	DioDevErrorDetect {DIO_DEV_ERROR_DETECT}		
Description	Switches the Development Error Detection and Notification ON or OFF		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
Name last modified in: V1.0.0	DioVersionInfoApi {DIO_VERSION_INFO_API}		
Description	Adds / removes the service Dio_ GetVersionInfo() from the code.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
No Included Containers			

With the parameters collected in container `DioOptionalApis` it is possible to select which APIs shall be included in the generated driver code.

The DioPort container holds configuration parameters that apply to complete ports, plus the channels and channel groups defined within that port (as subcontainers). Which individual channels are part of a port is typically defined by hardware.

DioPort

SWS Item	
Container Name	DioPort
Description	Configuration of individual DIO ports, consisting of channels and possible channel groups. Nothe that this container definition does not explicitly define a symbolic name parameter. Instead, the container's short name will be used in the Ecu Configuration Description to specify the symbolic name of the port.
Configuration Parameters	

Name (last modified in: V1.0.0)	DioPortId {DIO_PORT_ID}		
Description	Numeric identifier of the DIO port. Not all MCU ports may be used for DIO, thus there may be "gaps" in the list of all IDs. This value will be assigned to the DIO port symbolic name (i.e. the SHORT-NAME of the DioPort container).		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	Pre-compile time	X	All Variants
	Link time	X	LT
	Post-build time	–	
Scope / Dependency	scope: Module		

Included Containers		
Container Name	Multiplicity	Scope / Depedency
DioChannel	0..*	
DioChannelGroup	0..*	

The parameters defined per channel are aggregated by the DioChannel container.

DioChannel

SWS Item	
Container Name	DioChannel
Description	Configuration of an individual DIO channel. Besides a HW specific channel name which is typically fixed for a specific micro controller, additional symbolic names can be defined per channel. Note hat this container definition does not explicitly define a symbolic name parameter. Instead, the container's short name will be used in the Ecu Configuration Description to specify the symbolic name of the channel.
Configuration Parameters	

Name last modified in: V1.0.0	DioChannelId {DIO_CHANNEL_ID}		
Description	Channel Id of the DIO channel. This value will be assigned to the symbolic names.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	X	LT
	Post-build time	–	
Scope / Dependency	scope: Module		
No Included Containers			

In addition to reading and writing individual channels, it is also possible to group adjoining channels within a port to channel groups. This allows reading and writing the whole group simultaneously with a single API call. Channel groups are typically user defined and not fixed by hardware. All parameters necessary to define the channel group are aggregated by container `DioChannelGroup`.

Whether a channel is part of a channel group is defined by the references `DioChannelGroupChannelLow` and `DioChannelGroupChannelHigh`, which refer to the lowest and highest channel in the channel group, respectively.

DioChannelGroup

SWS Item			
Container Name	DioChannelGroup		
Description	Definition and configuration of DIO channel groups. A channel group represents several adjoining DIO channels represented by a logical group. Note hat this container definition does not explicitly define a symbolic name parameter. Instead, the container's short name will be used in the Ecu Configuration Description to specify the symbolic name of the channel group.		
Configuration Parameters			

Name last modified in: V1.0.0	DioChannelGroupIdentification		
Description	The DIO channel group is identified in DIO API by a pointer to a data structure (of type Dio_ChannelGroupType). That data structure contains the channel group information. This parameter contains the code fragment that has to be inserted in the API call of the calling module to get the address of the variable in memory which holds the channel group information. Example values are "&MyDioGroup1" or "&MyDioGroupArray[0]"		
Multiplicity	1		
Type	StringParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	X	LT
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.1.0	DioPortMask {DIO_PORT_MASK}		
Description	This shall be the mask which defines the positions of the channel group. The data type depends on the port width		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	X	LT
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	DioPortOffset {DIO_PORT_OFFSET}		
Description	The position of the Channel Group on the port, counted from the LSB. This value can be derived from DioPortMask. calculationFormula = Position of the first bit of DioPortMask which is set to '1' counted from LSB		
Multiplicity	1		
Type	DerivedIntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	X	LT
	Post-build time	–	
Scope / Dependency			
No Included Containers			

6.4 ADC Driver

The ADC driver is specified in [12]. It initializes and controls the internal Analog Digital Converter unit of the microcontroller. The structure of configuration model for the ADC module is shown in figure 6.14. Several containers are used to describe the configuration. On top level, `AdcPublishedInformation` container holds additional, module specific published information. The `AdcGeneral` container holds gerneral, pre-comiple parameter settings and `AdcConfigSet` holds the post-build changeable configuration data of the module.

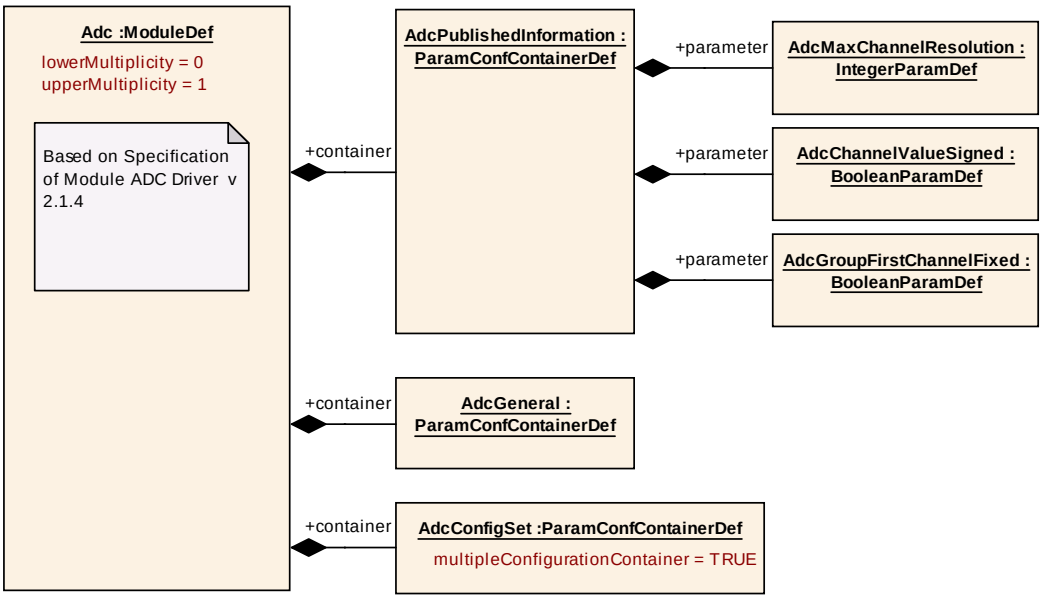


Figure 6.14: ADC module configuration overview

Module Name	Adc	
Module Description	Configuration of the Adc (Analog Digital Conversion) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
AdcConfigSet	1	This is the base container that contains the post-build selectable configuration parameters
AdcGeneral	1	General configuration (parameters) of the ADC Driver software module.
AdcPublishedInformation	1	Additional published parameters not covered by CommonPublishedInformation container. Note that these parameters do not have any configuration class setting, since they are published information.
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.

AdcPublishedInformation

SWS Item	[ADC030]		
Container Name	AdcPublishedInformation		
Description	Additional published parameters not covered by CommonPublishedInformation container. Note that these parameters do not have any configuration class setting, since they are published information.		
Configuration Parameters			

Name last modified in: V1.0.0	AdcChannelValueSigned {ADC_CHANNEL_VALUESIGNED} [ADC030]		
Description	Information whether the result value of the ADC driver has sign information (true) or not (false). If the result shall be interpreted as signed value it shall apply to C-language rules.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	AdcGroupFirstChannelFixed {ADC_GROUP_FIRST_CHANNEL_FIXED} [ADC030]		
Description	Information whether the first channel of an ADC Channel group can be configured (false) or is fixed (true) to a value determined by the ADC HW Unit.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	AdcMaxChannelResolution {ADC_MAX_CHANNEL_RESOLUTION} [ADC030]		
Description	Maximum Channel resolution in bits (does not specify accuracy).		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			
No Included Containers			

AdcGeneral

SWS Item	[ADC027]		
Container Name	AdcGeneral {AdcDriverGeneralConfiguration}		
Description	General configuration (parameters) of the ADC Driver software module.		
Configuration Parameters			

Name last modified in: V1.0.0	AdcDeInitApi {ADC_DEINIT_API}		
Description	Adds / removes the service Adc_DeInit() from the code.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	AdcDevErrorDetect {ADC_DEV_ERROR_DETECT}		
Description	Switches the Development Error Detection and Notification ON or OFF.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	AdcEnableStartStopGroupApi {ADC_ENABLE_START_STOP_GROUP_API}		
Description	Adds / removes the services Adc_StartGroupConversion() and Adc_StopGroupConversion() from the code.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	AdcGrpNotifCapability {ADC_GRP_NOTIF_CAPABILITY} [ADC105]		
Description	Determines, if the group notification mechanism is available at runtime.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.1.0	AdcGrpPriorityImpLevel {ADC_GRP_PRIORITY_IMP_LEVEL}		
Description	Determines the number of available priority levels (255 means a full support of the priority mechanism support, 0 means that the priority mechanism is not available at all, any other value means a partial support of the priority mechanism).		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	AdcHwTriggerApi {ADC_HW_TRIGGER_API}		
Description	Adds / removes the services Adc_EnableHardwareTrigger() and Adc_DisableHardwareTrigger() from the code.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module dependency: ADC_HW_TRIGGER_CAPABILITY		

Name last modified in: V1.0.0	AdcVersionInfoApi {ADC_VERSION_INFO_API}		
Description	Adds / removes the service Adc_GetVersionInfo() from the code.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
No Included Containers			

In order to support post-build configuration, the container `AdcConfigSet` is defined to be the base container that holds the post-build configuartion of the module. The ADC driver can handle several hardware units simultaneously. The unit specific configuration is gathered in container `AdcHwUnit`, as shown in figure 6.15.

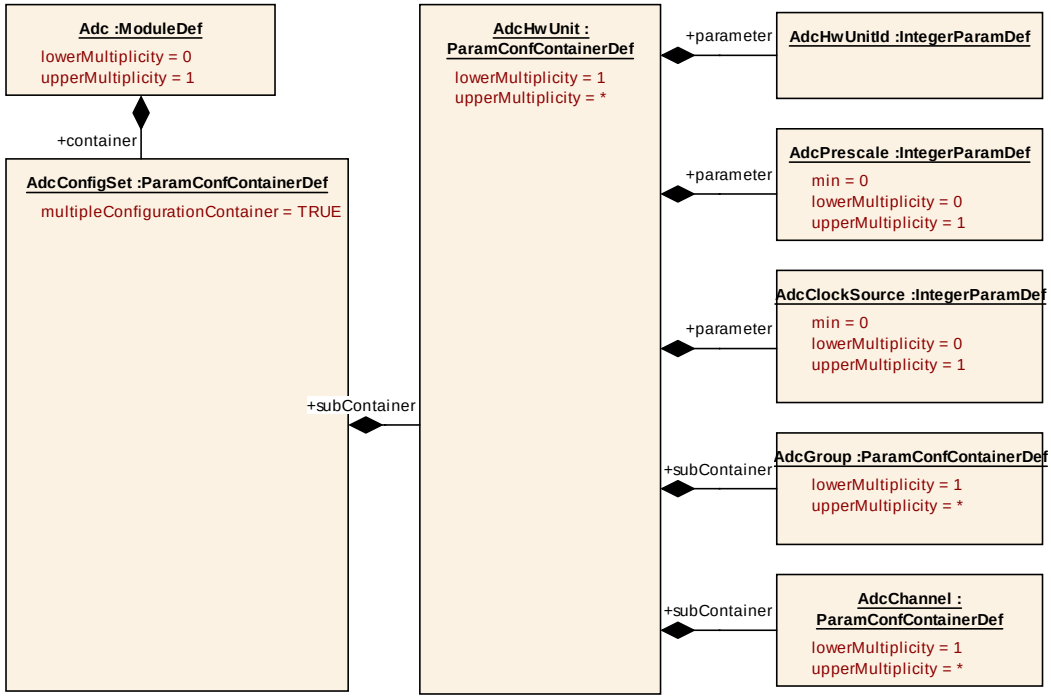


Figure 6.15: Configuration set of one or several ADC HW units

AdcConfigSet

SWS Item		
Container Name	AdcConfigSet	
Description	This is the base container that contains the post-build selectable configuration parameters	
Configuration Parameters		
Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
AdcHwUnit	1..*	

AdcHwUnit

SWS Item	[ADC242]		
Container Name	AdcHwUnit {AdcHWUnitConfiguration}		
Description	This container contains the Driver configuration (parameters) depending on grouping of channels This container could contain HW specific parameters which are not defined in the Standardized Module Definition. They must be added in the Vendor Specific Module Definition.		
Configuration Parameters			

Name last modified in: V1.0.0	AdcClockSource {ADC_CLK_SRC}		
Description	The ADC module specific clock input for the conversion unit can statically be configured to select different clock sources if provided by hardware.		
Multiplicity	0..1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module		

Name last modified in: V1.1.0	AdcHwUnitId {ADC_HWUNIT_ID}		
Description	Numeric ID of the HW Unit. This symbolic nam allows accessing Hw Unit data.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	AdcPrescale {ADC_PRESCALE}		
Description	Optional ADC module specific clock prescale factor, if supported by hardware.		
Multiplicity	0..1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module		

Included Containers

Container Name	Multiplicity	Scope / Depedency
AdcChannel	1..*	
AdcGroup	1..*	

All channel specific configuration is collected in container `AdcChannel` (see figure 6.16.

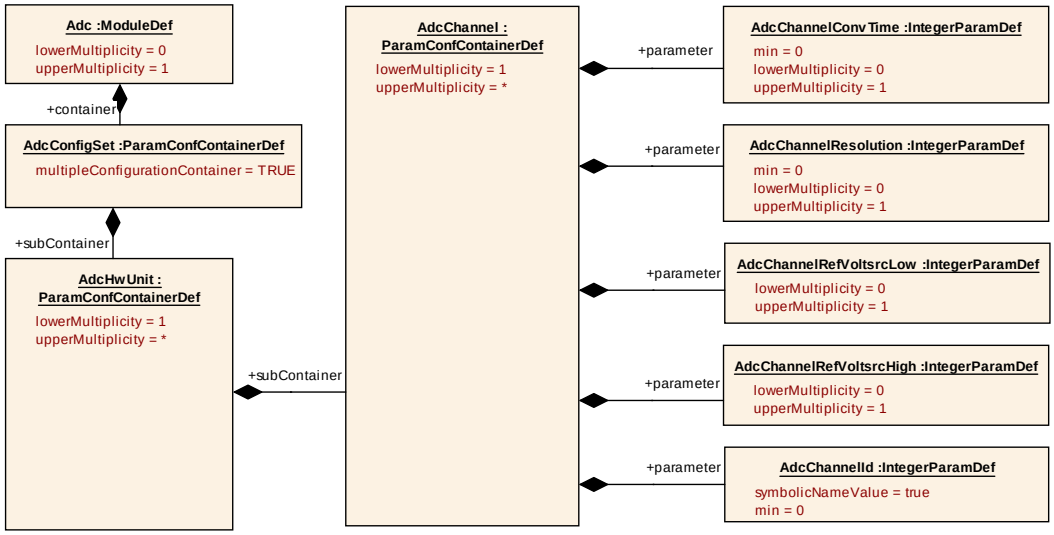


Figure 6.16: Configuration of ADC channels

AdcChannel

SWS Item	[ADC268]
Container Name	AdcChannel {AdcChannelConfiguration}
Description	This container contains the channel configuration (parameters) depending on the hardware capability. The organization of this data structure could contain dependencies to the microcontroller so this is left up to the implementer and its location is left up to the configuration. Note: Since a AdcChannel can be part of several AdcGroups, this container is not realized as a subcontainer of AdcGroup but instead as a subcontainer of AdcHwUnit.
Configuration Parameters	

Name last modified in: V1.0.0	AdcChannelConvTime {ADC_CHANNEL_CONV_TIME}		
Description	Configuration of conversion time, including sample time (in clockcycles) for each channel, if supported by hardware .		
Multiplicity	0..1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	AdcChannelId		
Description	Channel Id of the ADC channel. This value will be assigned to the symbolic name derived of the AdcChannel container short name.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	AdcChannelRefVoltsrcHigh {ADC_CHANNEL_REF_VOLTSRC_HIGH}		
Description	Upper reference voltage source for each channel (optional)		
Multiplicity	0..1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	AdcChannelRefVoltsrcLow {ADC_CHANNEL_REF_VOLTSRC_LOW}		
Description	Lower reference voltage source for each channel (optional)		
Multiplicity	0..1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	AdcChannelResolution {ADC_CHANNEL_RESOLUTION}		
Description	Channel resolution in bits (optional)		
Multiplicity	0..1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module		
No Included Containers			

Several ADC channels can be grouped to a channel group. This is necessary since AD conversion is always triggered for a complete channel group. If conversion of a single channel is requested, a group with only one channel must be defined.

An ADC channel may be part of several channel groups. This may e.g. be necessary if a channel can be triggered in different trigger modes, since every channel group has a fixed trigger.

The `AdcGroup` container holds all parameters necessary to define an ADC channel group, as shown in figure 6.17. ADC conversion for all channels within a group is triggered simultaneously.

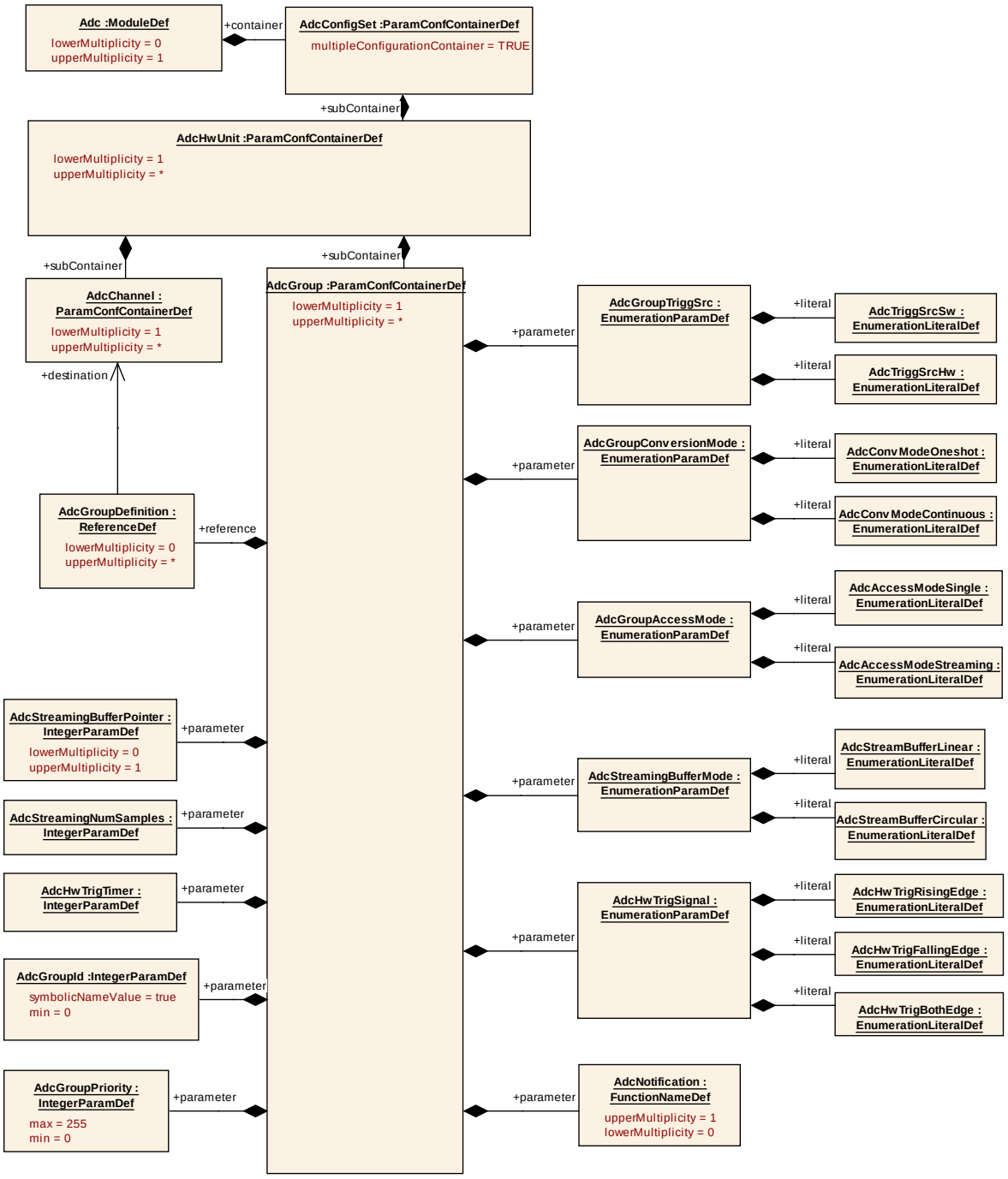


Figure 6.17: Configuration of ADC channel groups

Parameter `AdcGroupTriggSrcOneshot` may be extended with vendor specific literals depending on HW support. E.g., several HW triggers may be supported by hardare which may be added to the list of literals.

AdcGroup

SWS Item	[ADC028]		
Container Name	AdcGroup {AdcGroupConfiguration}		
Description	This container contains the Group configuration (parameters).		
Configuration Parameters			

Name last modified in: V1.1.0	AdcGroupAccessMode {ADC_GROUP_ACCESS_MODE} [ADC317]		
Description	Type of access mode to group conversion results.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	AdcAccessModeSingle	Single value access mode	
	AdcAccessMode Streaming	Streaming access mode	
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency			

Name last modified in: V1.0.0	AdcGroupConversionMode {ADC_GROUP_CONV_MODE}		
Description	Conversion mode of the channel group.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	AdcConvModeContinuous	: Conversions of an ADC channel group are performed continuously after a software API call (start). The conversions itself are running automatically (no additional software or hardware trigger needed).	
	AdcConvModeOneshot	The conversion of an ADC channel group is performed once after a trigger.	
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	AdcGroupDefinition {ADC_GROUP_FIRST_CHANNEL}		
Description	Assignment of channels to a channel group. For each AdcChannel that should belong to the group, a reference needs to be defined.		
Multiplicity	0..*		
Type	Reference to AdcChannel		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module dependency: ADC_GROUP_FIRST_CHANNEL_FIXED		

Name last modified in: V1.0.0	AdcGroupId		
Description	Channel Id of the ADC channel group This value will be assigned to the symbolic name derived of the AdcChannelGroup container short name.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module		

Name last modified in: V1.1.0	AdcGroupPriority {ADC_GROUP_PRIORITY} [ADC287]		
Description	Priority level of the group		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	dependency: ADC_GRP_PRIORITY_IMP_LEVEL		

Name last modified in: V1.0.0	AdcGroupTriggSrc {ADC_GROUP_TRIGG_SRC}		
Description	Triggering source for oneshot conversion:		
Multiplicity	1		
Type	EnumerationParamDef		
Range	AdcTriggSrcHw	Group is triggered by a hardware event.	
	AdcTriggSrcSw	Group is triggered by a software API call.	
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module		

Name last modified in: V1.1.0	AdcHwTrigSignal {ADC_HW_TRIG_SIGNAL}		
Description	Configures on which edge of the hardware trigger signal the driver should reach, i.e. start the conversion (only if supported by the ADC hardware)		
Multiplicity	1		
Type	EnumerationParamDef		
Range	AdcHwTrigBothEdge	React on both edges of the hardware trigger signal (only if supported by the ADC hardware)	
	AdcHwTrigFallingEdge		
	AdcHwTrigRisingEdge		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency			

Name last modified in: V1.1.0	AdcHwTrigTimer {ADC_HW_TRIG_TIMER}		
Description	Reload value of the ADC module embedded timer (only if supported by hardware)		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency			

Name last modified in: V1.0.0	AdcNotification {Adc_Notification}		
Description	Callback function for each group		
Multiplicity	0..1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module dependency: This parameter is only available, if notification capability is configured available by AdcGrpNotifCapability		

Name last modified in: V1.1.0	AdcStreamingBufferMode {ADC_STREAMING_BUFFER_MODE}		
Description	Configure streaming buffer as ”linear buffer” (i.e. the ADC Driver stops the conversion as soon as the stream buffer is full) or as ”ring buffer” (wraps around if the end of the stream buffer is reached).		
Multiplicity	1		
Type	EnumerationParamDef		
Range	AdcStreamBufferCircular	The ADC Driver continues the conversion even if the stream buffer is full (number of samples reached) by wrapping around the stream buffer itself	
	AdcStreamBufferLinear	The ADC Driver stops the conversion as soon as sthe stream buffer is full (number of samples reached).	
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency			

Name last modified in: V1.1.0	AdcStreamingBufferPointer {ADC_STREAMING_BUFFER_POINTER} [ADC291]		
Description	Pointer to data buffer (destination for conversion results) in case of streaming access mode		
Multiplicity	0..1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	dependency: AdcGroupAccessMode		

Name last modified in: V1.1.0	AdcStreamingNumSamples {ADC_STREAMING_NUM_SAMPLES} [ADC292]		
Description	Number of ADC values to be acquired per channel in streaming access mode.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency			
No Included Containers			

6.5 ICU Driver

The Input Capture Unit (ICU) driver is specified in [21]. It provides services for signal edge and level notification. Furthermore it provides services to control Wake-up interrupts.

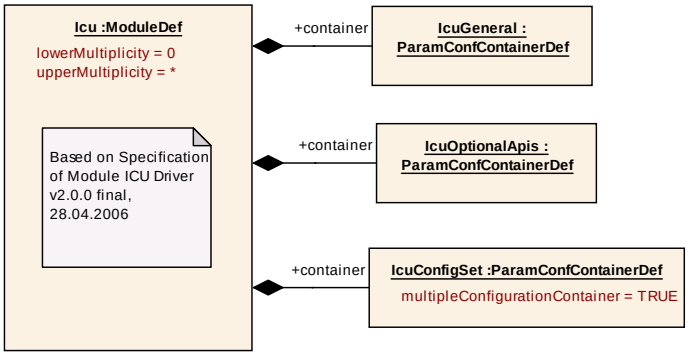


Figure 6.18: Parameter description for the ICU module

Configuration of the ICU driver is shown in figure 6.18.

Module Name	Icu	
Module Description	Configuration of the Icu (Input Capture Unit) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
IcuConfigSet	1	This container is the base for a multiple configuration set
IcuGeneral	1	Configuration of general ICU parameters.
IcuOptionalApis	1	This container contains all configuration switches for configuring optional API services of the ICU driver.

General parameters are collected in the `IcuGeneral` container, shown in figure 6.19.

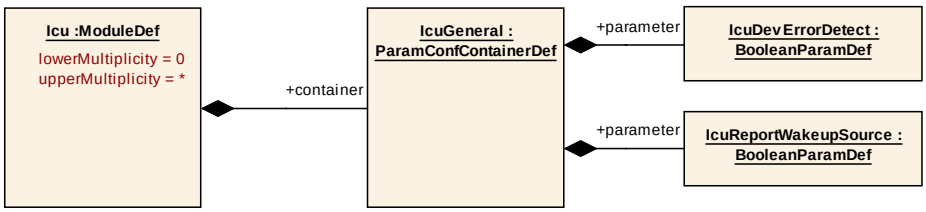


Figure 6.19: ICU module wide parameters

IcuGeneral

SWS Item	[ICU026]		
Container Name	IcuGeneral {General Configuration}		
Description	Configuration of general ICU parameters.		
Configuration Parameters			

Name last modified in: V1.0.0	IcuDevErrorDetect {ICU_DEV_ERROR_DETECT}		
Description	Switches the Development Error Detection and Notification ON or OFF		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	IcuReportWakeupSource {ICU_REPORT_WAKEUP_SOURCE}		
Description	Switch for enabling Wakeup source reporting.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
No Included Containers			

With the parameters collected in container `IcuOptionalApis` (as shown in figure 6.20) it is possible to select which APIs shall be included in the generated driver code.

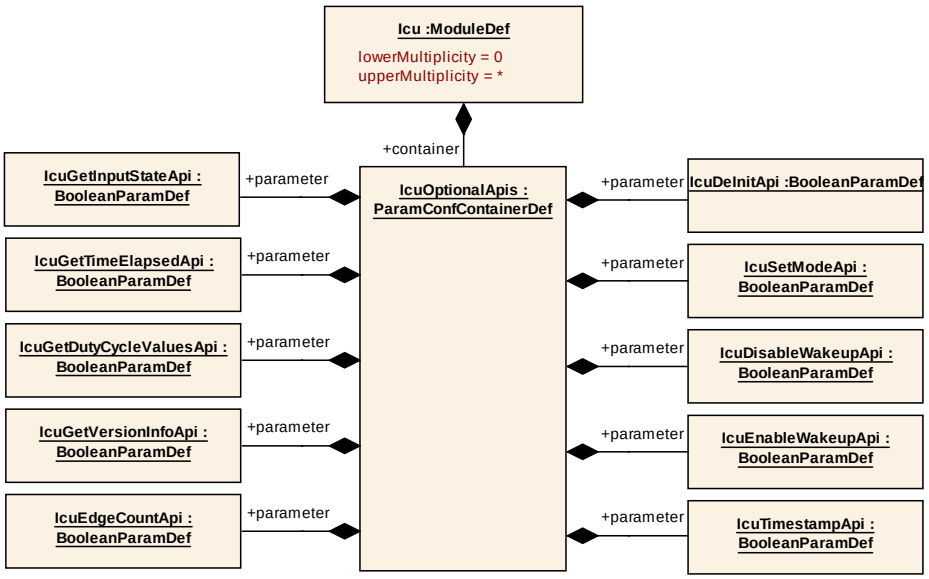


Figure 6.20: Optional APIs in the ICU driver

IcuOptionalApis

SWS Item	[ICU114]		
Container Name	IcuOptionalApis {Configuration of optional API services}		
Description	This container contains all configuration switches for configuring optional API services of the ICU driver.		
Configuration Parameters			
Name last modified in: V1.0.0	IcuDeInitApi {ICU_DE_INIT_API}		
Description	Adds / removes the service Icu_DeInit() from the code		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
Name last modified in: V1.0.0	IcuDisableWakeupApi {ICU_DISABLE_WAKEUP_API}		
Description	Adds / removes the service Icu_DisableWakeup() from the code.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
Name last modified in: V1.0.0	IcuEdgeCountApi {ICU_EDGE_COUNT_API} [ICU124]		
Description	Adds / removes all services related to the edge counting functionality as listed below, from the code: Icu_ResetEdgeCount(), Icu_EnableEdgeCount(), Icu_DisableEdgeCount(), Icu_GetEdgeNumbers()		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
Name last modified in: V1.0.0	IcuEnableWakeupApi {ICU_ENABLE_WAKEUP_API}		
Description	Adds / removes the service Icu_EnableWakeup() from the code.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
Name last modified in: V1.0.0	IcuGetDutyCycleValuesApi {ICU_GET_DUTY_CYCLE_VALUES_API}		
Description	Adds / removes the service Icu_GetDutyCycleValues() from the code.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	IcuGetInputStateApi {ICU_GET_INPUT_STATE_API}		
Description	Adds / removes the service Icu_GetInputState() from the code.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	IcuGetTimeElapsedApi {ICU_GET_TIME_ELAPSED_API}		
Description	Adds / removes the service Icu_GetTimeElapsed() from the code.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	IcuGetVersionInfoApi {ICU_GET_VERSION_INFO_API}		
Description	Adds / removes the service Icu_ GetVersionInfo() from the code.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	IcuSetModeApi {ICU_SET_MODE_API}		
Description	Adds / removes the service Icu_SetMode() from the code		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	IcuTimestampApi {ICU_TIMESTAMP_API} [ICU123]		
Description	Adds / removes all services related to the timestamping functionality as listed below from the code: Icu_StartTimestamp(), Icu_StopTimestamp(), Icu_GetTimestampIndex()		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
No Included Containers			

The `Icu` modules includes `IcuChannel` subcontainers, one for each ICU channel provided by the microcontroller. ICU channel specific configuration is collected in the `IcuChannel` containers (shown in figure 6.21).

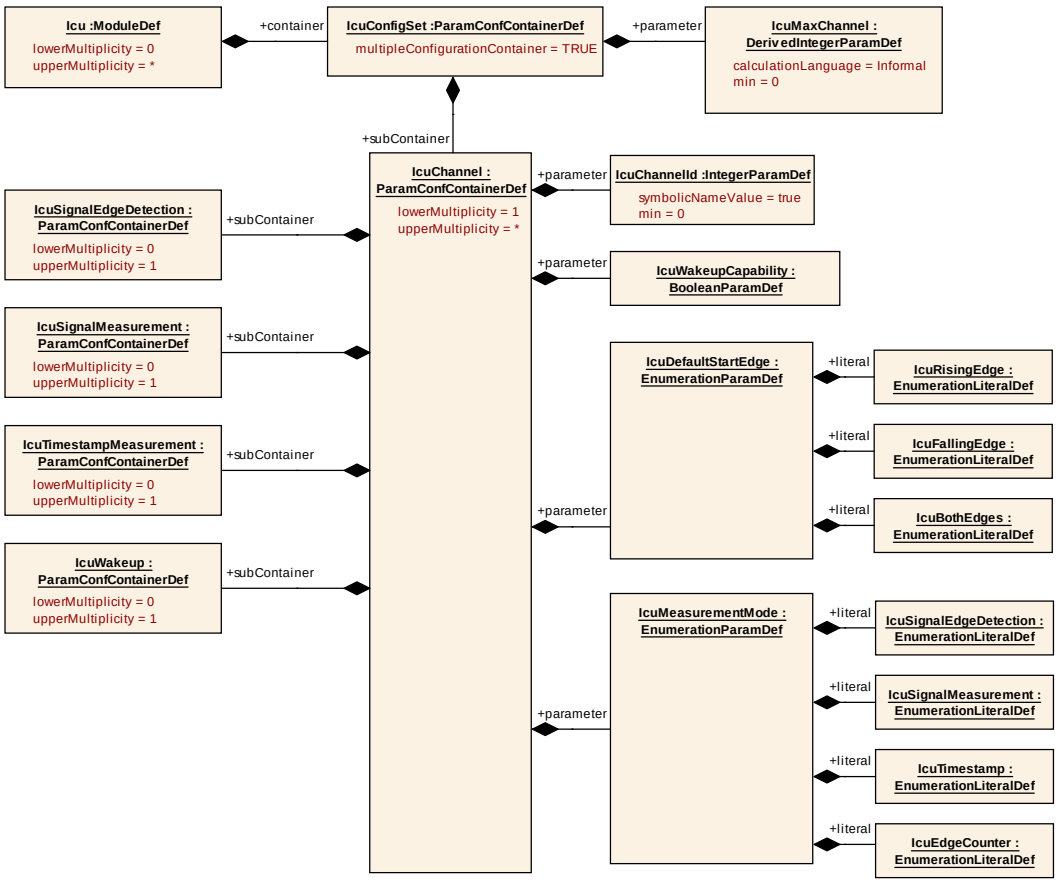


Figure 6.21: ICU channel specific parameters

IcuChannel

SWS Item	[ICU027]
Container Name	IcuChannel
Description	Configuration of an individual ICU channel.
Configuration Parameters	

Name last modified in: V1.0.0	IcuChannelId		
Description	Channel Id of the ICU channel. This value will be assigned to the symbolic name derived of the IcuChannel container short name.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	IcuDefaultStartEdge {Icu_DefaultStartEdge}		
Description	Configures the default-activation-edge which shall be used for this channel if there was no activation-edge configured by the call of service Icu_SetActivationCondition(). In case the Measurement Mode is ”SignalMeasurement” and the properties DutyCycle or Period are set, the edge configured here is used as Default Period Start Edge.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	IcuBothEdges	As default, both edges are used.	
	IcuFallingEdge	As default, falling edge is the used.	
	IcuRisingEdge	As default, rising edge is the used.	
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	IcuMeasurementMode {Icu_MeasurementMode}		
Description	Configures the measurement mode of this channel.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	IcuEdgeCounter	The channnel is used to count the edges which are configured by the call of the service Icu_SetActivationCondition(). The following API services support this mode: - Icu_EnableEdgeCount() - Icu_DisableEdgeCount() - Icu_GetEdgeNumbers() - Icu_ResetEdgeCount() This mode can only be configured if IcuEdgeVountApi is switched on.	
	IcuSignalEdgeDetection	The channel is used for detecting the edges which are configured by the call of the service Icu_SetActivationCondition(). The following API services support this mode: - Icu_EnableNotification() - Icu_DisableNotification() - Icu_GetInputState()	
	IcuSignalMeasurement	The channel is used to measure different times between various configurable edges. The configuration of the period-start edges are done by configuration and cannot be changed during runtime. The following API services support this mode: - Icu_GetTimeElapsed() - Icu_GetDutyCycleValues() - Icu_GetInputState() This mode can only be configured if at least one of the following switches are set to ”on”: - IcugetdutyCycleValuesApi - IIcuGetTimeElapsedApi	
	IcuTimestamp	The channel is used to capture timer values on the edges which are configured by the call of the service Icu_SetActivationCondition(). The following API services support this mode: - Icu_StartTimestamp() - Icu_StopTimestamp() - Icu_GetTimestampIndex() This mode can only be configured if IcuTimeStampApi is switched on.	
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module dependency: The possible measurement modes are depending on the pre-processor switches, which enable/disable optional API services.		

Name last modified in: V1.0.0	IcuWakeupCapability {Icu_WakeupCapability}		
Description	Boolean flag whether wake-up on this channel is supported or not.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module		
Included Containers			

Container Name	Multiplicity	Scope / Depedency
IcuEdgeCounter Measurement	0..1	
IcuSignalEdgeDetection	0..1	
IcuSignalMeasurement	0..1	
IcuTimestamp Measurement	0..1	
IcuWakeup	0..1	

The channel container contains several optional subcontainers with parameters for specific purpose, shown in figure 6.22.

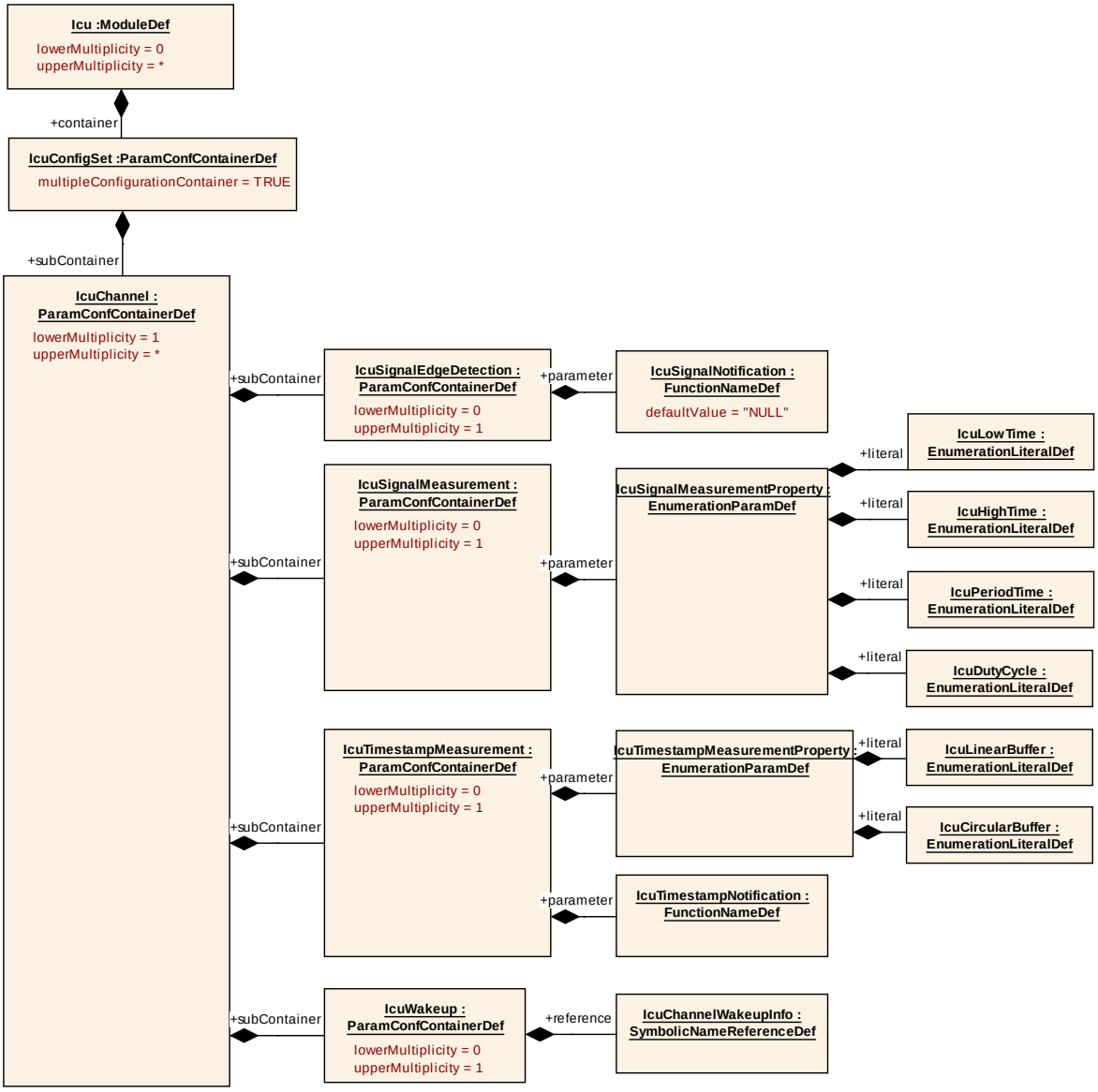


Figure 6.22: ICU channel subcontainers and their parameters

IcuSignalEdgeDetection

SWS Item	[ICU021]		
Container Name	IcuSignalEdgeDetection {Configuration of Signal Edge Detection}		
Description	This container contains the configuration (parameters) in case the measurement mode is ”IcuSignalEdgeDetection”		
Configuration Parameters			
Name last modified in: V1.0.0	IcuSignalNotification {Icu_SignalNotification_<Channel>}		
Description	Notification function for signal notification.		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module dependency: IcuMeasurementMode		
No Included Containers			

IcuSignalMeasurement

SWS Item	[ICU039]		
Container Name	IcuSignalMeasurement {Configuration of Signal Measurement}		
Description	This container contains the configuration (parameters) in case the measurement mode is ”IcuSignalMeasurement”		
Configuration Parameters			
Name last modified in: V1.0.0	IcuSignalMeasurementProperty {Icu_SignalMeasurementProperty}		
Description	Configures the property that could be measured in case the mode is ”IcuSignalMeasurement”. This property can not be changed during runtime.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	IcuDutyCycle	The channel is configured to read values which are needed for calculating the duty cycle (coherent Active and Period Time).	
	IcuHighTime	The channel is configured for reading the elapsed Signal High Time	
	IcuLowTime	The channel is configured for reading the elapsed Signal Low Time	
	IcuPeriodTime	The channel is configured for reading the elapsed Signal Period Time	
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module dependency: IcuMeasurementMode, IcuGetDutyCycleValuesApi, IcuGetTimeElapsedApi		
No Included Containers			

IcuTimestampMeasurement

SWS Item	[ICU039]
Container Name	IcuTimestampMeasurement {Configuration of Timestamp Measurement}
Description	This container contains the configuration (parameters) in case the measurement mode is ”IcuTimestamp”
Configuration Parameters	

Name last modified in: V1.0.0	IcuTimestampMeasurementProperty {Icu_TimestampMeasurement Property}		
Description	Configures the handling of the buffer in case the mode is ”Timestamp”		
Multiplicity	1		
Type	EnumerationParamDef		
Range	IcuCircularBuffer	After reaching the end of the buffer, the driver restarts at the beginning of the buffer	
	IcuLinearBuffer	The buffer will just be filled once	
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module dependency: IcuMeasurementMode		

Name last modified in: V1.0.0	IcuTimestampNotification {Icu_TimestampNotification_<Channel>}		
Description	Notification function if the number of requested timestamps (Notification interval $>$ 0) are acquired.		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module dependency: IcuTimestampApi		
No Included Containers			

IcuEdgeCounterMeasurement

SWS Item	[ICU076]
Container Name	IcuEdgeCounterMeasurement {Configuration of Edge Counter Measurement}
Description	This container contains the configuration (parameters) in case the measurement mode is ”IcuEdgeCounter”
Configuration Parameters **No Included Containers**	

IcuWakeup

SWS Item	[ICU126]		
Container Name	IcuWakeup {Wakeup Configuration}		
Description	This container contains the configuration (parameters) needed to configure a wakeup capable channel		
Configuration Parameters			
Name last modified in: V1.0.0	IcuChannelWakeupInfo {Icu_ChannelWakeupInfo}		
Description	If the wakeup-capability is true the wakeup source referenced is transmitted to the ECU State Manager (EcuM) .		
Multiplicity	1		
Type	Symbolic name reference to EcuMWakeupSourceConfig		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: ECU dependency: IcuWakeupCapability and IcuReportWakeupSource		
No Included Containers			

6.6 PWM Driver

The pulse width modulation (PWM) driver is specified in [36]. It provides services for initialization and control of the microcontroller internal PWM stage. The PWM module generates pulses with variable pulse width. It allows the selection of the duty cycle and the signal period time.

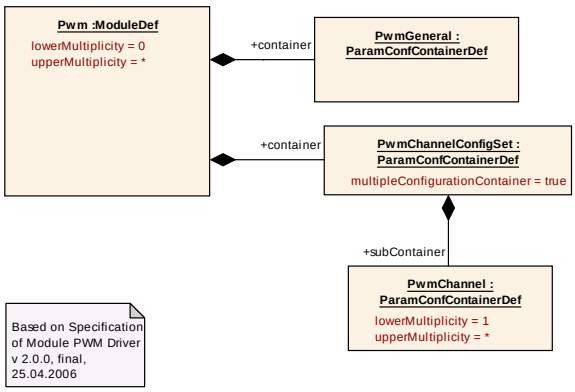


Figure 6.23: Overview PWM module

The configuration model for the PWM module is shown in figure 6.23.

Module Name	Pwm	
Module Description	Configuration of Pwm (Pulse Width Modulation) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
PwmChannelConfigSet	1	
PwmConfigurationOfOptApi Services	1	
PwmGeneral	1	

Configuration of the PWM driver is straightforward. General parameters are collected in the `PwmGeneral` container.

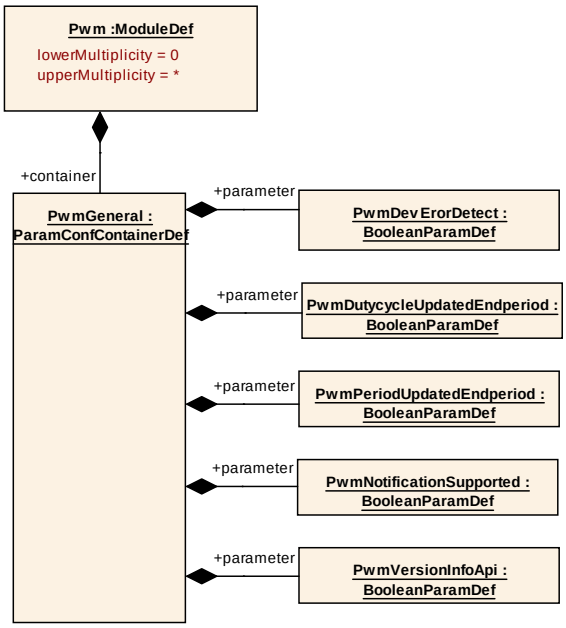


Figure 6.24: General PWM module configuration

PwmGeneral

SWS Item	[PWM004]
Container Name	PwmGeneral {PwmModuleConfiguration}
Description	
Configuration Parameters	

Name last modified in: V1.0.0	PwmDevErorDetect {PWM_DEV_ERROR_DETECTED}		
Description	Switch for enabling the development error detection.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PwmDutycycleUpdatedEndperiod {PWM_DUTYCYCLE_UPDATED_ENDPERIOD}		
Description	Switch for enabling the update of the duty cycle parameter at the end of the current period		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PwmNotificationSupported {PWM_NOTIFICATION_SUPPORTED}		
Description	Switch to indicate that the notifications are supported		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PwmPeriodUpdatedEndperiod {PWM_PERIOD_UPDATED_ENDPERIOD}		
Description	Switch for enabling the update of the period parameter at the end of the current period		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PwmVersionInfoApi {PWM_VERSION_INFO_API}		
Description	Switch to indicate that the Pwm_ GetVersionInfo is supported		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
No Included Containers			

The PWM module includes `PwmChannel` subcontainers, one for each PWM channel provided by the microcontroller.

PWM channel specific configuration is collected in the `PwmChannel` containers.

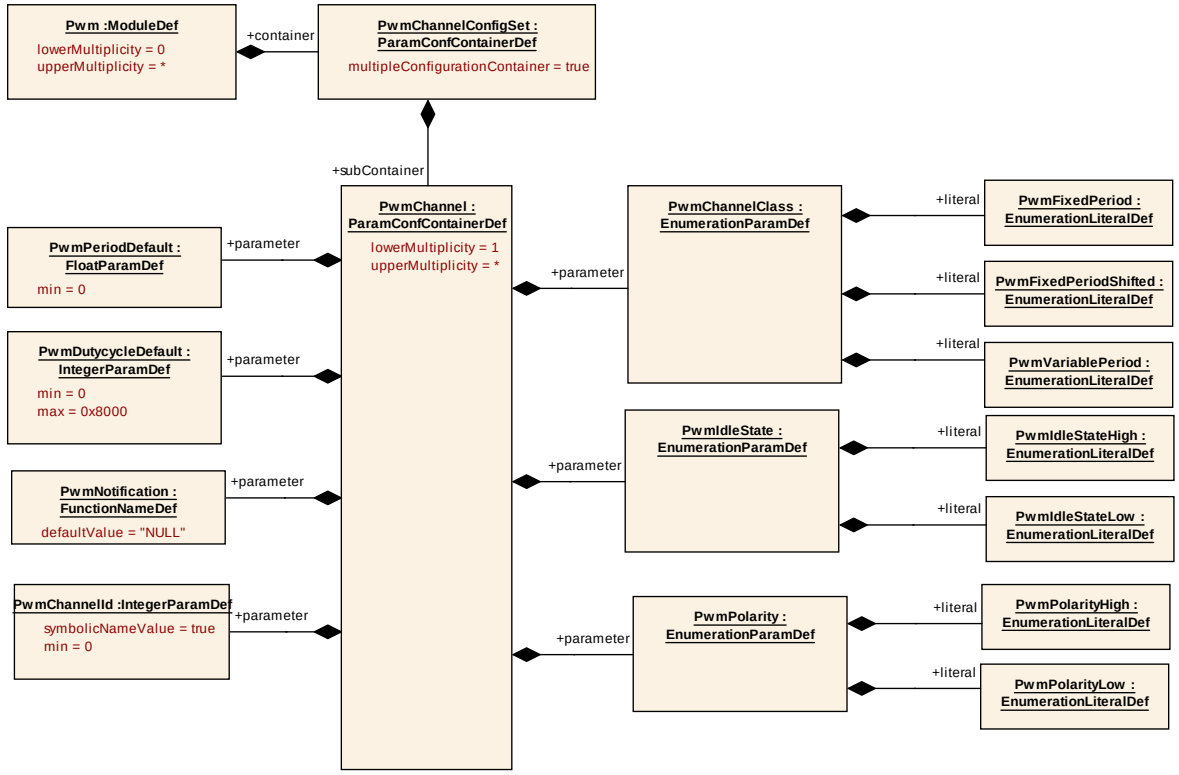


Figure 6.25: PWM channel specific configuration

PwmChannel

SWS Item	[PWM027]
Container Name	PwmChannel {PwmChannelConfiguration}
Description	Configuration of an individual PWM channel.
Configuration Parameters	

Name last modified in: V1.0.0	PwmChannelClass {PWM_CHANNEL_CLASS}		
Description	Class of PWM Channel.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	PwmFixedPeriod	Only the duty cycle can be changed.	
	PwmFixedPeriodShifted	Only the duty cycle can be changed. The period is shifted (only if supported by hardware)	
	PwmVariablePeriod	Duty Cycle and period can be changed.	
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: ECU		

Name last modified in: V1.0.0	PwmChannelId		
Description	Channel Id of the PWM channel. This value will be assigned to the symbolic name derived of the PwmChannel container short name.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	PwmDutycycleDefault {PWM_DUTYCYLE_DEFAULT}		
Description	Value of duty cycle used for Initialization 0, represents 0% 0x8000 represents 100%		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 32768		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: ECU		

Name last modified in: V1.0.0	PwmIdleState {PWM_IDLE_STATE}		
Description	Value of PWM output signal used for Initialization.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	PwmIdleStateHigh	The PWM channel output will be set to high (3 or 5 V) in idle state.	
	PwmIdleStateLow	Support for fixed period shifted is hardware dependent.	
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: ECU		

Name last modified in: V1.0.0	PwmNotification {Pwm_Notification}		
Description	Definition of the Callback function.		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: ECU		

Name last modified in: V1.0.0	PwmPeriodDefault {PWM_PERIOD_DEFAULT}		
Description	Value of period used for Initialization.(in seconds).		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: ECU		

Name last modified in: V1.0.0	PwmPolarity {PWM_POLARITY}		
Description	Defines the starting polarity of each PWM channel.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	PwmPolarityHigh	The PWM channel output is high at the beginning of the cycle and then goes low when the duty count is reached.	
	PwmPolarityLow	The PWM channel output will be set to low (0 V) in idle state	
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: ECU		
No Included Containers			

6.7 SPI

The configuration parameters of the SPI (Serial peripheral interface) handler/driver is described in this subchapter. The module SPI contains configuration elements for one handler and one or more driver. The structure is shown in figure 6.26.

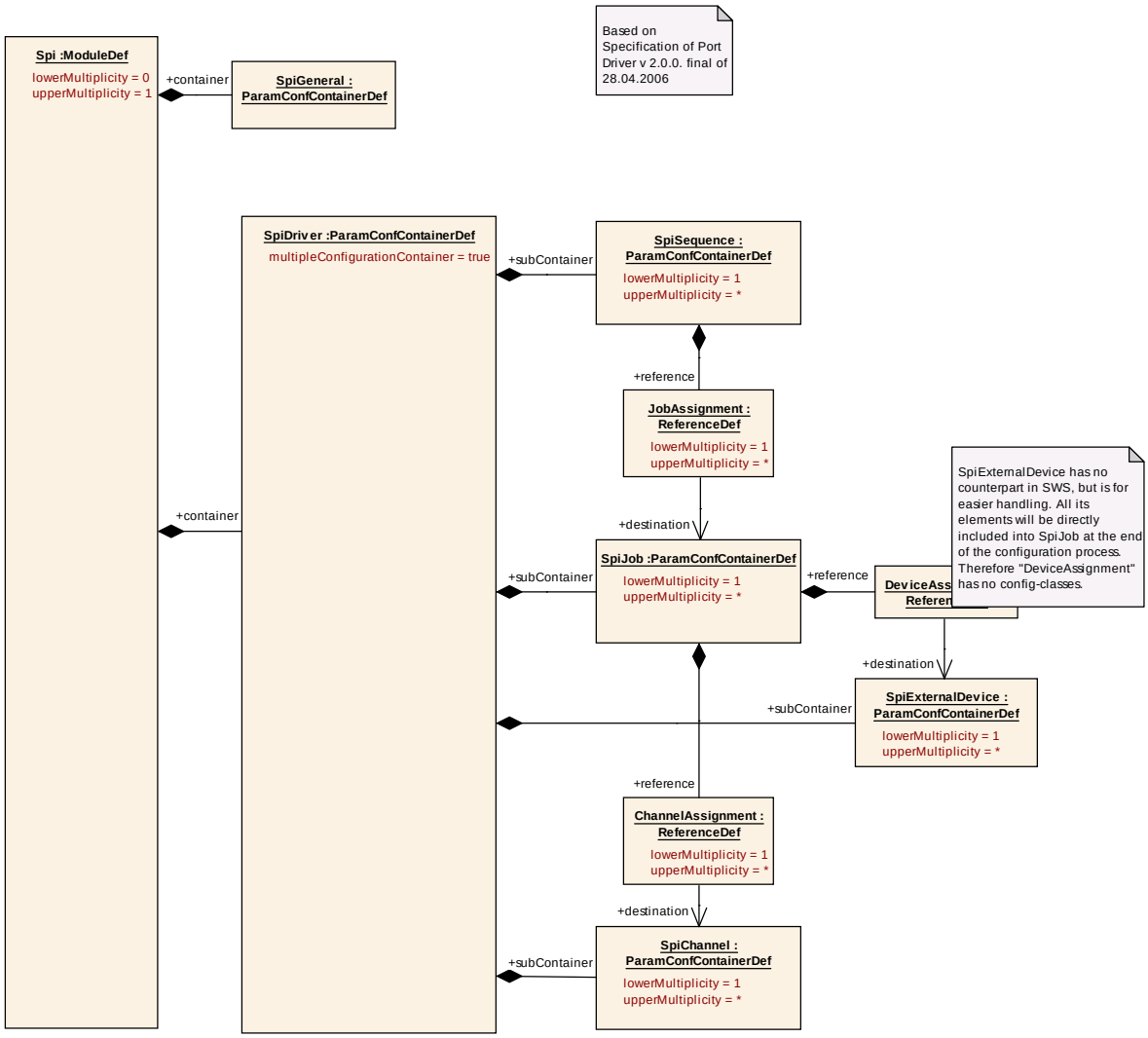


Figure 6.26: Overview: SPI

The handler has only general settings, which configure some general capabilities of the handler/driver. That means here it's possible to influence the need of resources from the base.

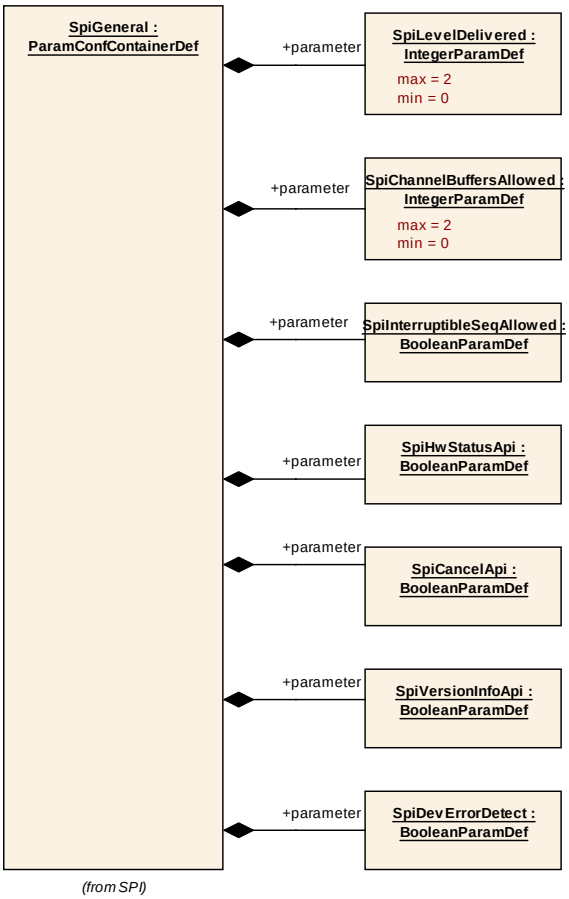


Figure 6.27: General configuration parameters

The configuration elements for a SPI-driver can be divided into four major parts: the sequences, the jobs, the channels and the external-device-settings.

The sequences are the top interface to the user of the SPI, like the EEPROM-driver. Sequences can reference a number of jobs which should be worked on in one communication cycle.

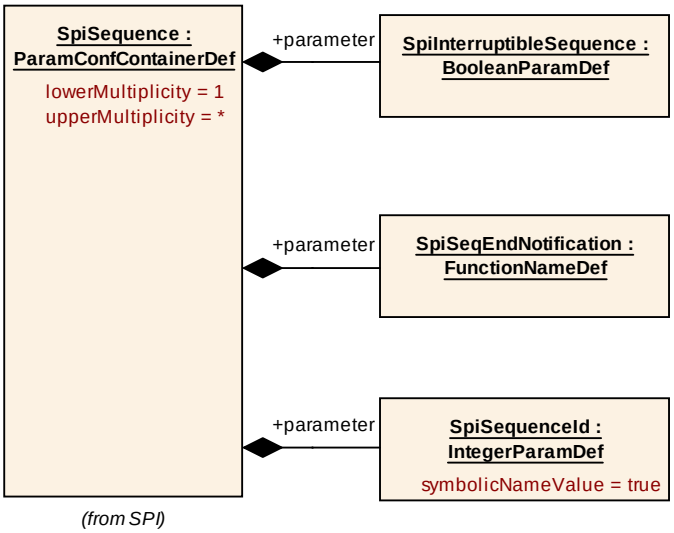


Figure 6.28: SPI-Sequence

The jobs aggregate all the information about the destination of a data transfer. Jobs reference first to a number of channels. Second jobs reference to a SpiExternalDevice for the information that is needed to communicate with a special device connected to the SPI bus. The SpiExternalDevice has no counterpart in the SWS of the SPI, but is used to group the parameter of a job in an hardware depended part and a part which contains more communication related paramters (like the related channels). At the end of configuration process, when all parameters have been set, the tools will copy the parameters from SpiExternalDevice to the belonging job.

The container SpiExternalDevice contains communication information about an external device like baudrate, chipselects and special timings.

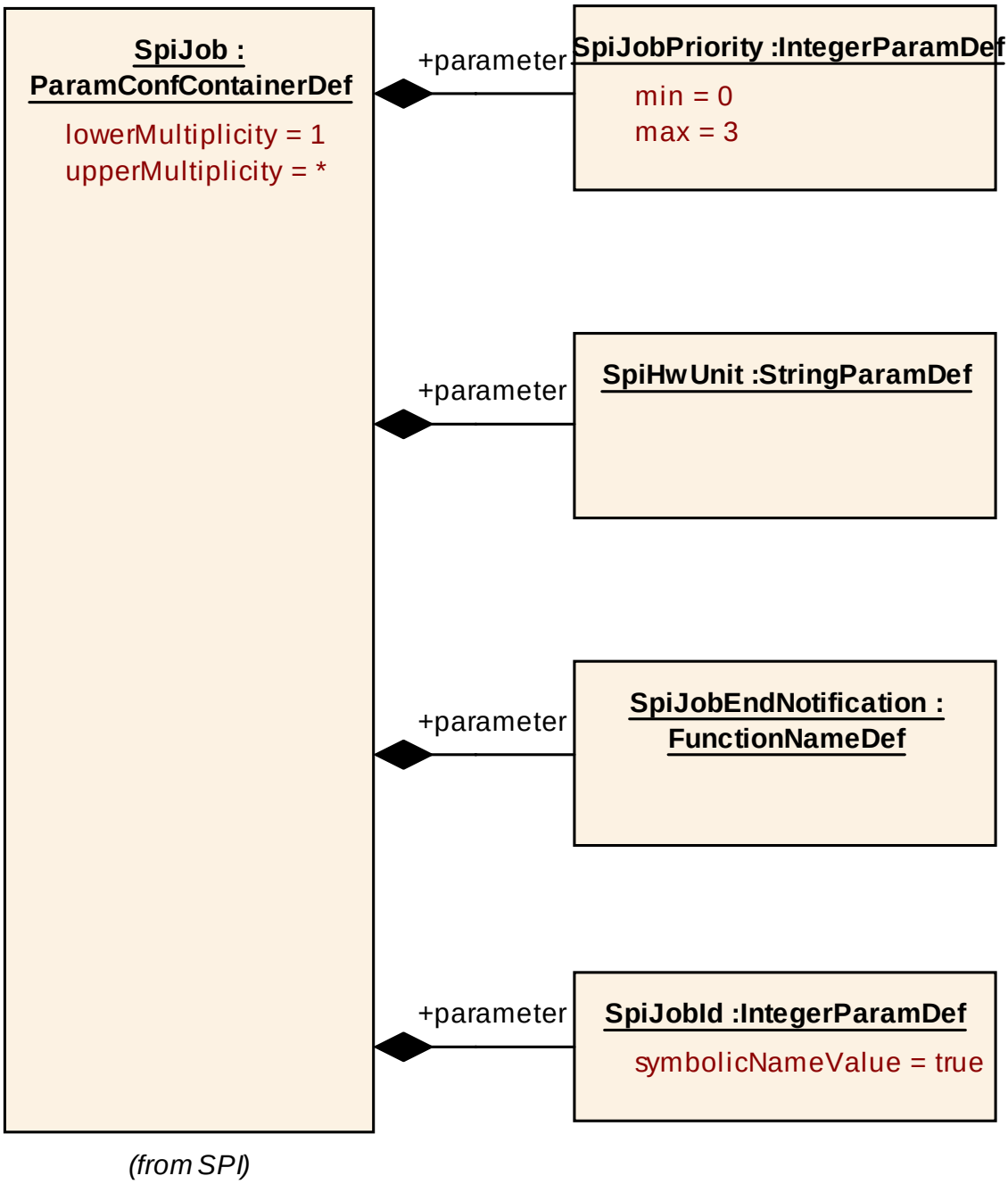


Figure 6.29: SPI-Job

A channel contains information about one data item which is transported between the hardware communication partner and the SPI-chip.

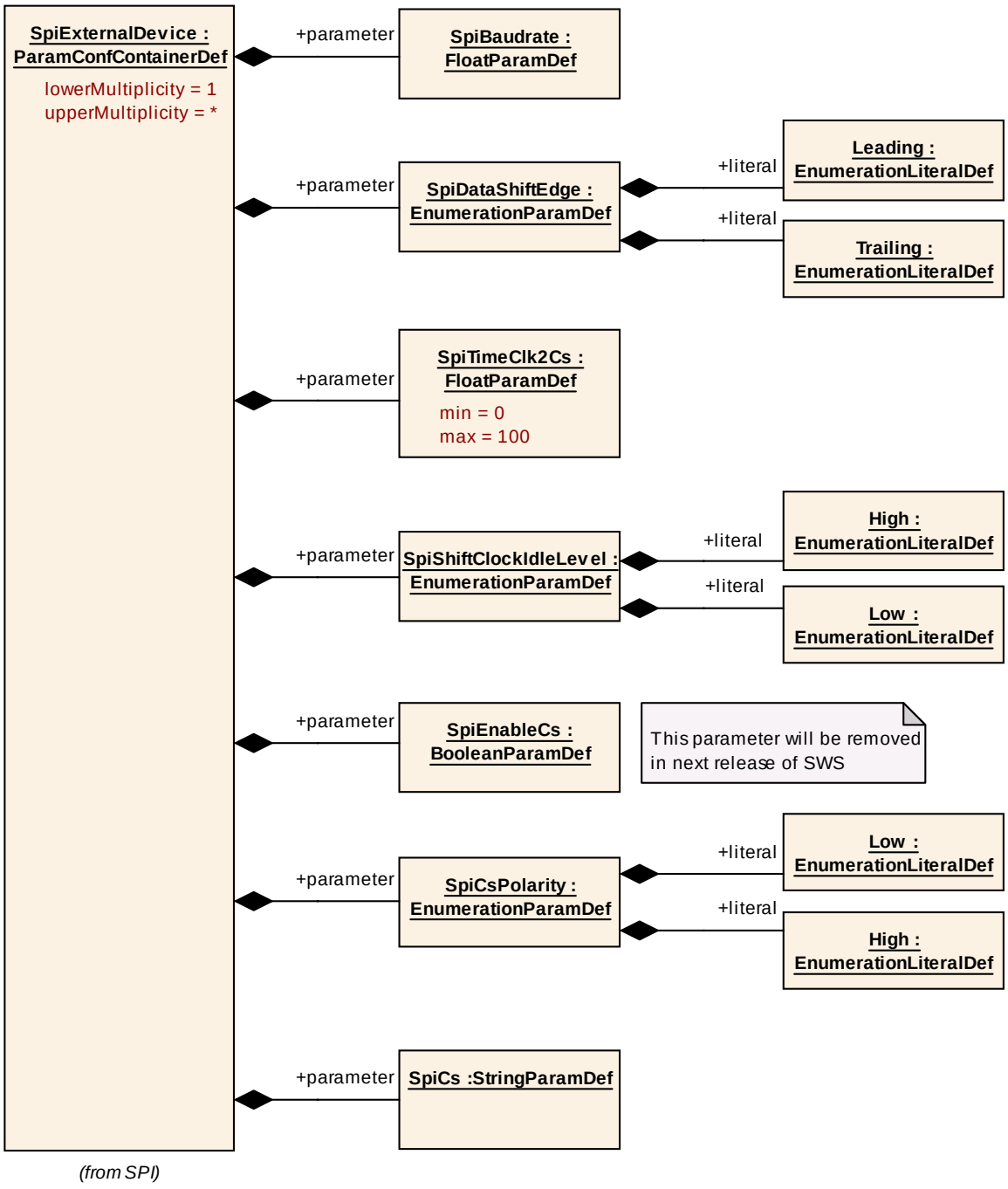


Figure 6.30: SPI-External Device: part of a job

Remark: Buffers for reading or writing data will not be configured! This information has to be set via the API dynamically.

For a detailed description look at the SPI-SWS as specified in [38].

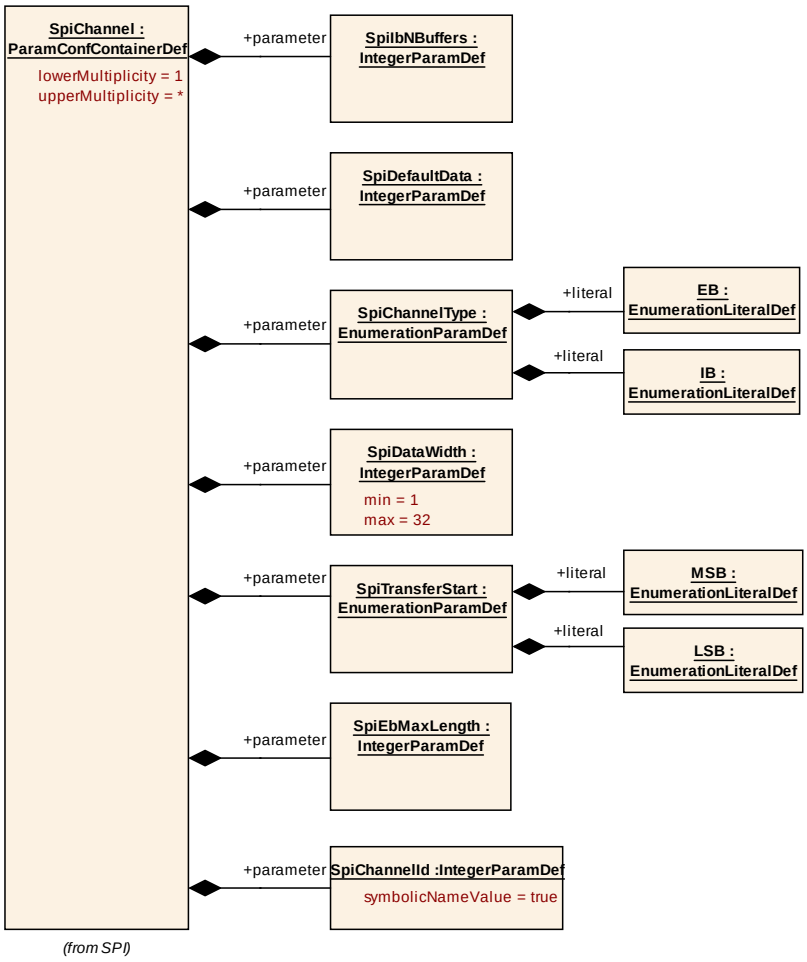


Figure 6.31: SPI-Channel

6.7.1 Interfaces to other driver

Concerning the configuration the SPI driver/handler has logical interfaces to the port driver, because in many cases the communication partner of an SPI has to be addressed via a digital pin, called chip-select.

But the SPI does not use the API of the DIO module, it does the handling and initialisation on its on. So there is no reference to a dedicated port pin in configuration. Only during configuration of a port pin there can be defined that a pin is reserved for use by SPI. The parameter SpiCs in SpiExternalDevice contains the port-pin the SPI uses, with the format of a symbolic name. The list of possible names has to be provided by the deliverer of the SPI driver.

The user of the SPI driver/handler, like the EEPROM-driver, reference only to a SpiSequence, like shown in figure 6.32. The parameter SpiSequenceName is used for "'implementing"' the reference - it contains the id, which will be used in the end by the BSW-software.

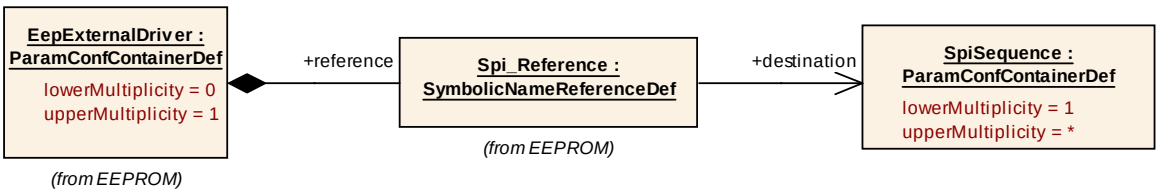


Figure 6.32: Interfaces to 'User' of SPI

6.7.2 Description of the parameter

The following tables contain the definition of all parameters concerning the SPI handler/driver.

Module Name	Spi	
Module Description	Configuration of the Spi (Serial Peripheral Interface) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
SpiDriver	1	
SpiGeneral	1	General configuration settings for SPI-Handler

SpiGeneral

SWS Item			
Container Name	SpiGeneral		
Description	General configuration settings for SPI-Handler		
Configuration Parameters			

Name last modified in: V1.0.0	SpiCancelApi {SPI_CANCEL_API}		
Description	Switches the Spi_Cancel function ON or OFF.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	SpiChannelBuffersAllowed {SPI_CHANNEL_BUFFERS_ALLOWED}		
Description	Selects the SPI Handler/Driver Channel Buffers usage allowed and delivered.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 2		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	SpiDevErrorDetect {SPI_DEV_ERROR_DETECT}		
Description	Switches the Development Error Detection and Notification ON or OFF.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	SpiHwStatusApi {SPI_HW_STATUS_API}		
Description	Switches the Spi_GetHWUnitStatus function ON or OFF.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	SpiInterruptibleSeqAllowed {SPI_INTERRUPTIBLE_SEQ_ALLOWED}		
Description	Switches the Interruptible Sequences handling functionality ON or OFF.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module dependency: This parameter depends on SPI_LEVEL_DELIVERED value. It is only used for SPI_LEVEL_DELIVERED configured to 1 or 2.		

Name last modified in: V1.0.0	SpiLevelDelivered {SPI_LEVEL_DELIVERED}		
Description	Selects the SPI Handler/Driver level of scalable functionality that is available and delivered.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 2		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	SpiVersionInfoApi {SPI_VERSION_INFO_API}		
Description	Switches the Spi_GetVersionInfo function ON or OFF.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		
No Included Containers			

SpiSequence

SWS Item	[SPI106]
Container Name	SpiSequence {SpiSequenceConfiguration}
Description	All data needed to configure one SPI-sequence
Configuration Parameters	

Name last modified in: V1.0.0	JobAssignment {SPI_JOB_LINKING}		
Description	A sequence references several jobs, which are executed during a communication sequence		
Multiplicity	1..*		
Type	Reference to SpiJob		
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	X	PB
Scope / Dependency	scope: ECU		

Name last modified in: V1.0.0	SpiInterruptibleSequence {SPI_INTERRUPTIBLE_SEQUENCE}		
Description	This parameter allows or not this Sequence to be suspended by another one.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	X	PB
Scope / Dependency	scope: module dependency: This SPI_INTERRUPTIBLE_SEQ_ALLOWED parameter as to be configured as ON.		

Name last modified in: V1.0.0	SpiSeqEndNotification {SPI_SEQ_END_NOTIFICATION}		
Description	This parameter is a reference to a notification function.		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	X	PB
Scope / Dependency	scope: ECU		

Name last modified in: V1.0.0	SpiSequenceId {SPI_SEQUENCE_NAME}		
Description			
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		
No Included Containers			

SpiChannel

SWS Item	[SPI104]
Container Name	SpiChannel {SpiChannelConfiguration}
Description	All data needed to configure one SPI-channel
Configuration Parameters	

Name last modified in: V1.0.0	SpiChannelId {SPI_CHANNEL_NAME}		
Description			
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name last modified in: V1.0.0	SpiChannelType {SPI_CHANNEL_TYPE}		
Description	Buffer usage with EB/IB channel		
Multiplicity	1		
Type	EnumerationParamDef		
Range	EB	External Buffer	
	IB	SPI Handler/Driver Internal Buffer	
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	X	PB
Scope / Dependency	scope: ECU dependency: SPI_CHANNEL_BUFFERS_ALLOWED		

Name last modified in: V1.0.0	SpiDataWidth {SPI_DATA_WIDTH}		
Description	This parameter is the width of a transmitted data unit.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 32		
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	X	PB
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	SpiDefaultData {SPI_DEFAULT_DATA}		
Description	This parameter is the default value to transmit.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	X	PB
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	SpiEbMaxLength {SPI_EB_MAX_LENGTH}		
Description	This parameter contains the maximum size of data buffers in case of EB Channels and only.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	X	PB
Scope / Dependency	scope: module dependency: The SPI_CHANNEL_TYPE parameter has to be configured as EB for this Channel. The SPI_CHANNEL_BUFFERS_ALLOWED parameter has to be configured as 1 or 2.		

Name last modified in: V1.0.0	SpiIbNBuffers {SPI_IB_N_BUFFERS}		
Description	This parameter contains the maximum number of data buffers in case of IB Channels and only.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	X	PB
Scope / Dependency	scope: module dependency: The SPI_CHANNEL_TYPE parameter has to be configured as IB for this Channel. The SPI_CHANNEL_BUFFERS_ALLOWED parameter has to be configured as 0 or 2.		

Name last modified in: V1.0.0	SpiTransferStart {SPI_TRANSFER_START}		
Description	This parameter defines the first starting bit for transmission.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	LSB	Transmission starts with the Least Significant Bit first	
	MSB	Transmission starts with the Most Significant Bit first	
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	X	PB
Scope / Dependency	scope: module		
No Included Containers			

SpiJob

SWS Item	[SPI105]
Container Name	SpiJob {SpiJobConfiguration}
Description	All data needed to configure one SPI-Job, amongst others the connection between the internal SPI unit and the special settings for an external device is done.
Configuration Parameters	

Name last modified in: V1.0.0	ChannelAssignment {SPI_CHANNEL_LINKING}		
Description	A job references several channels.		
Multiplicity	1..*		
Type	Reference to SpiChannel		
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	X	PB
Scope / Dependency	scope: ECU		

Name last modified in: V1.0.0	DeviceAssignment		
Description	Reference to the external device used by this job		
Multiplicity	1		
Type	Reference to SpiExternalDevice		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	SpiHwUnit {SPI_HW_UNIT}		
Description	This parameter is the symbolic name to identify the HW SPI Hardware microcontroller peripheral allocated to this Job.		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	X	PB
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	SpiJobEndNotification {SPI_JOB_END_NOTIFICATION}		
Description	This parameter is a reference to a notification function.		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	X	PB
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	SpiJobId {SPI_JOB_NAME}		
Description			
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name last modified in: V1.0.0	SpiJobPriority {SPI_JOB_PRIORITY}		
Description	Priority of the Job		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 3		
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	X	PB
Scope / Dependency	scope: module		
No Included Containers			

SpiExternalDevice

SWS Item			
Container Name	SpiExternalDevice		
Description	The communication settings of an external device. Closely linked to SpiJob.		
Configuration Parameters			

Name last modified in: V1.0.0	SpiBaudrate {SPI_BAUDRATE}		
Description	This parameter is the communication baudrate - This parameter allows using a range of values, from the point of view of configuration tools, from Hz up to MHz.		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	X	PB
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	SpiCs		
Description			
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	SpiCsPolarity {SPI_CS_POLARITY}		
Description	This parameter defines the active polarity of Chip Select.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	High		
	Low		
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	X	PB
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	SpiDataShiftEdge {SPI_DATA_SHIFT_EDGE}		
Description	This parameter defines the SPI data shift edge.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	Leading		
	Trailing		
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	X	PB
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	SpiEnableCs {SPI_ENABLE_CS}		
Description	This parameter enables or not the Chip Select handling functions. It will be removed in next release of SWS		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	X	PB
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	SpiShiftClockIdleLevel {SPI_SHIFT_CLOCK_IDLE_LEVEL}		
Description	This parameter defines the SPI shift clock idle level.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	High		
	Low		
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	X	PB
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	SpiTimeClk2Cs {SPI_TIME_CLK2CS}		
Description	Timing between clock and chip select - This parameter allows to use a range of values from 0 up to 100 microSec. the real configuration-value used in software BSW-SPI is calculated out of this by the generator-tools		
Multiplicity	1		
Type	FloatParamDef		
Range	0 .. 100		
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	X	PB
Scope / Dependency	scope: module		
No Included Containers			

7 Memory

7.1 AUTOSAR Architecture

This chapter defines the configuration parameters of the different AUTOSAR Basic Software memory modules. These modules include the Memory Services, the Memory Hardware Abstraction and the Memory Drivers as shown in figure 7.1.

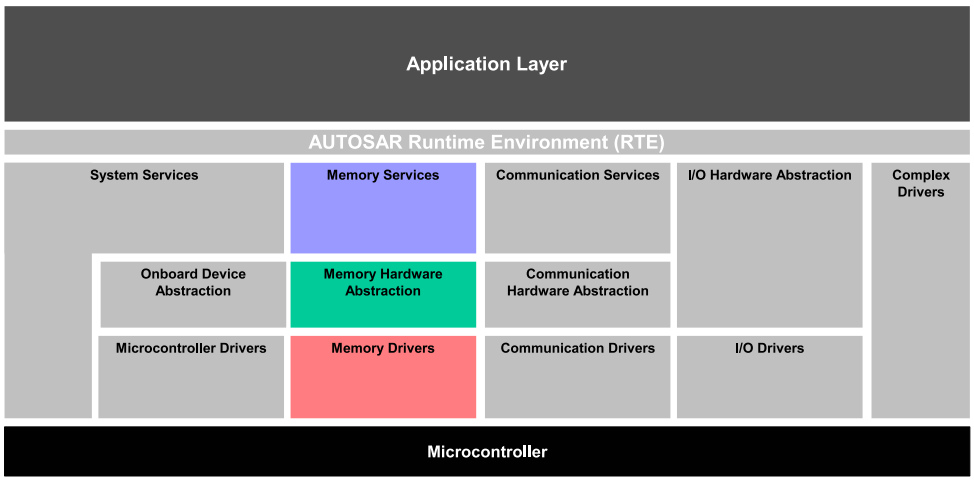


Figure 7.1: ECU Architecture Memory

Figure 7.2 shows the components and hierarchy of the memory stack. A detailed description of these modules is found in the following sections.

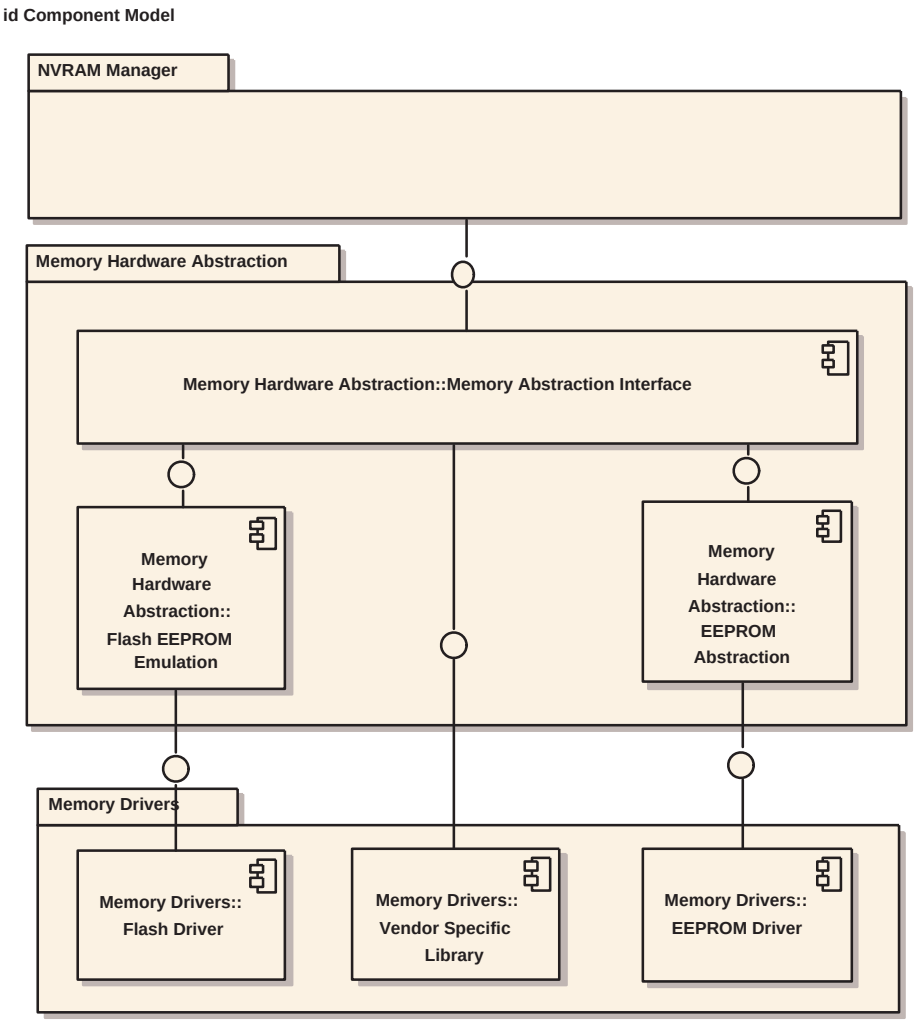


Figure 7.2: Module overview of memory hardware abstraction layer

7.2 Memory Services

This chapter defines the configuration parameters of the AUTOSAR NVRAM Manager module.

7.2.1 NVRAM Manager

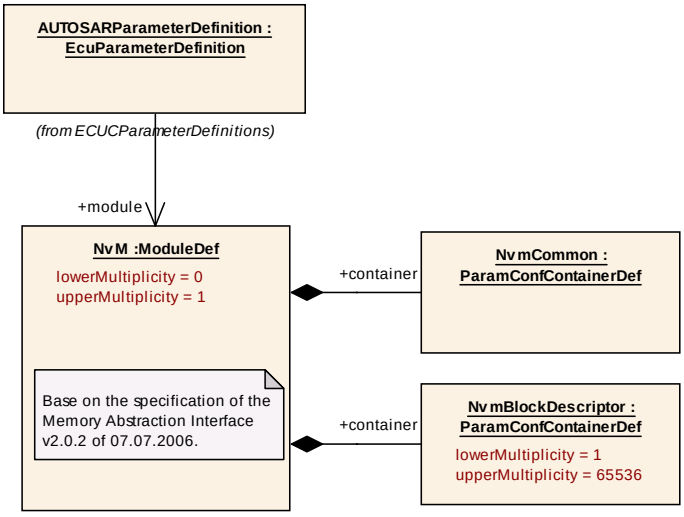


Figure 7.3: Root parameter description for the NVM module

Module Name	NvM	
Module Description	Configuration of the NvM (NvRam Manager) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
NvmBlockDescriptor	1..65536	Container for a management structure to configure the composition of a given NVRAM Block Management Type. Its multiplicity describes thenumber of configured NVRAM blocks, one block is required to be configured. The NVRAM block descriptors are condensed in the NVRAM block descriptor table. The start address of the table is configured in the NVMCommon parameters.
NvmCommon	1	Container for common configuration options.

The included containers `NvmCommon` and `NvmBlockDescriptor` will be described in detail (see figures 7.4 and 7.5).

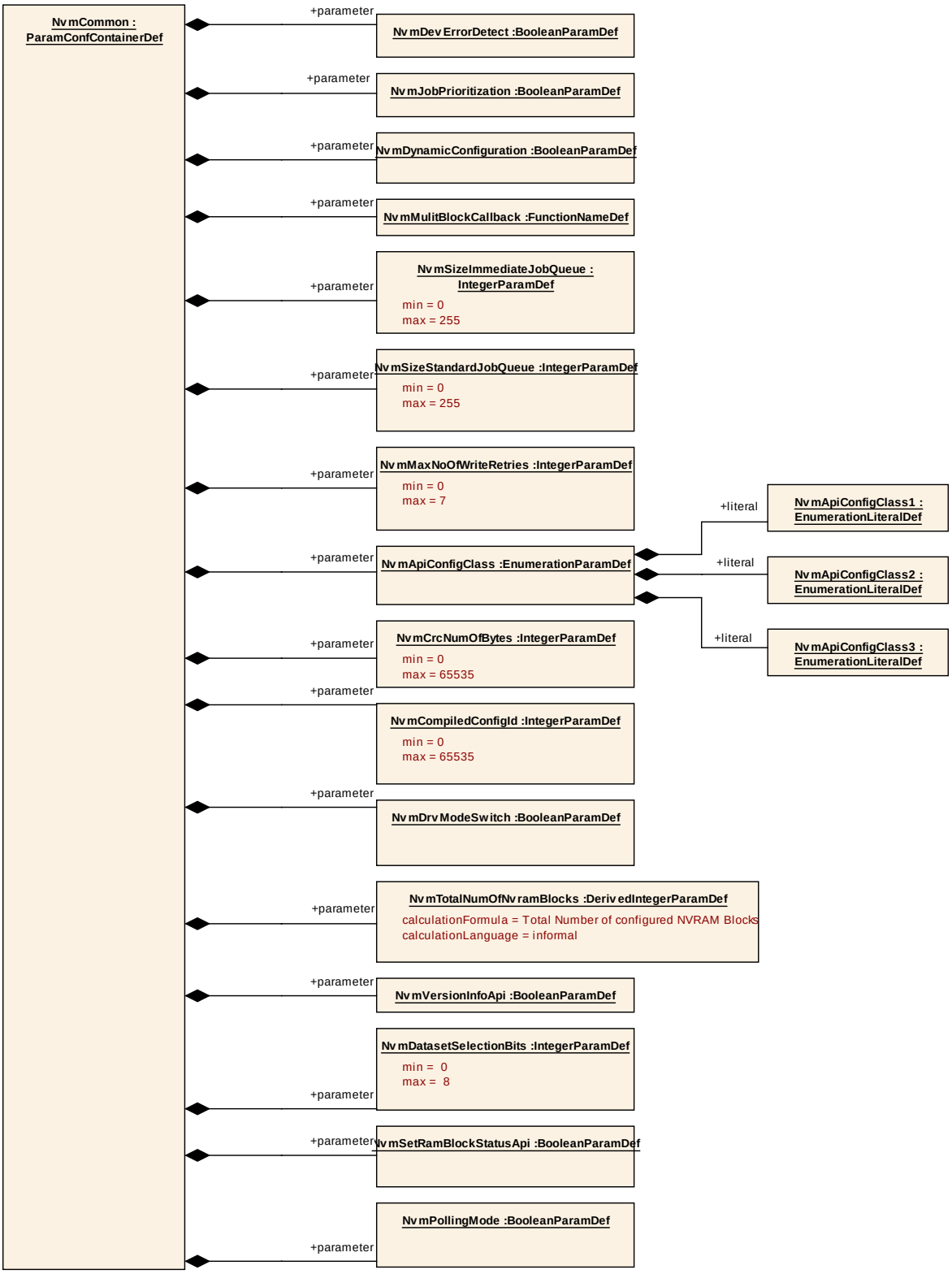


Figure 7.4: Common parameter description for the NVM module

NvmCommon

SWS Item	[NVM028]
Container Name	NvmCommon
Description	Container for common configuration options.
Configuration Parameters	

Name last modified in: V1.0.0	NvmApiConfigClass {NVM_API_CONFIG_CLASS}		
Description	Preprocessor switch to enable some API calls which are related to NVM API configuration classes.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	NvmApiConfigClass1	All API calls belonging to configuration class 1 are available.	
	NvmApiConfigClass2	All API calls belonging to configuration class 2 are available.	
	NvmApiConfigClass3	All API calls belonging to configuration class 3 are available.	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	NvmCompiledConfigId {NVM_COMPILED_CONFIG_ID}		
Description	Configuration ID regarding the NV memory layout. This configuration ID shall be published as e.g. a SW-C shall have the possibility to write it to NV memory.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	NvmCrcNumOfBytes {NVM_CRC_NUM_OF_BYTES}		
Description	If CRC is configured for at least one NVRAM block, this parameter defines the maximum number of bytes which shall be processed within one cycle of job processing.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module dependency: CRC library		

Name last modified in: V1.0.0	NvmDatasetSelectionBits {NVM_DATASET_SELECTION_BITS}		
Description	Defines the number of least significant bits which shall be used to address a certain dataset of a NVRAM block within the interface to the memory hardware abstraction. 0..8: Number of bits which are used for dataset or redundant block addressing. 0: No dataset or redundant NVRAM blocks are configured at all, no selection bits required. 1: In case of redundant NVRAM blocks are configured, but no dataset NVRAM blocks.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 8		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: MemHwA, NVM_NV_BLOCK_IDENTIFIER, NVM_BLOCK_MANAGEMENT_TYPE		

Name last modified in: V1.0.0	NvmDevErrorDetect {NVM_DEV_ERROR_DETECT}		
Description	Preprocessor switch to enable the development error detection		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	NvmDrvModeSwitch {NVM_DRV_MODE_SWITCH}		
Description	Preprocessor switch to enable switching memory drivers to fast mode during performing NvM_ReadAll and NvM_WriteAll		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	NvmDynamicConfiguration {NVM_DYNAMIC_CONFIGURATION}		
Description	Preprocessor switch to enable the dynamic configuration management handling by the NvM_ReadAll request		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	NvmJobPrioritization {NVM_JOB_PRIORITIZATION}		
Description	Preprocessor switch to enable job prioritization handling		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	NvmMaxNoOfWriteRetries {NVM_MAX_NUM_OF_WRITE_RETRIES}		
Description	Defines the maximum number of write retries for a NVRAM block with [NVM061]. Regardless of configuration a consistency check (and maybe write retries) are always forced for each block which is processed by the request NvM_WriteAll.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 7		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	NvmMulitBlockCallback {NVM_MULTI_BLOCK_CALLBACK}		
Description	Entry address of the common callback routine which shall be invoked on termination of each asynchronous multi block request		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	NvmPollingMode {NVM_POLLING_MODE}		
Description	Preprocessor switch to enable/disable the polling mode in the NVRAM Manager and at the same time disable/enable the callback functions useable by lower layers ON Polling mode enabled/ callbacks disabled OFF Polling mode disabled/callbacks enabled		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	NvmSetRamBlockStatusApi {NVM_SET_RAM_BLOCK_STATUS_API}		
Description	Preprocessor switch to enable the API NvM_SetRamBlockStatus.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	NvmSizeImmediateJobQueue {NVM_SIZE_IMMEDIATE_JOB_QUEUE}		
Description	Defines the number of queue entries for the immediate priority job queue. If NVM_JOB_PRIORITIZATION is switched OFF this parameter shall be out of scope.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module dependency: NVM_JOB_PRIORITIZATION		

Name last modified in: V1.0.0	NvmSizeStandardJobQueue {NVM_SIZE_STANDARD_JOB_QUEUE}		
Description	Defines the number of queue entries for the standard job queue.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	NvmTotalNumOfNvramBlocks {NVM_TOTAL_NUM_OF_NVRAM_BLOCKS}		
Description	This parameter shall publish the total number of configured NVRAM blocks to offer the possibility to do an over-all block handling, e.g. during start-up by a single SW-C.		
Multiplicity	1		
Type	DerivedIntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	NvmVersionInfoApi {NVM_VERSION_INFO_API}		
Description	Preprocessor switch to enable the version info API [NVM285], [NVM286].		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		
No Included Containers			

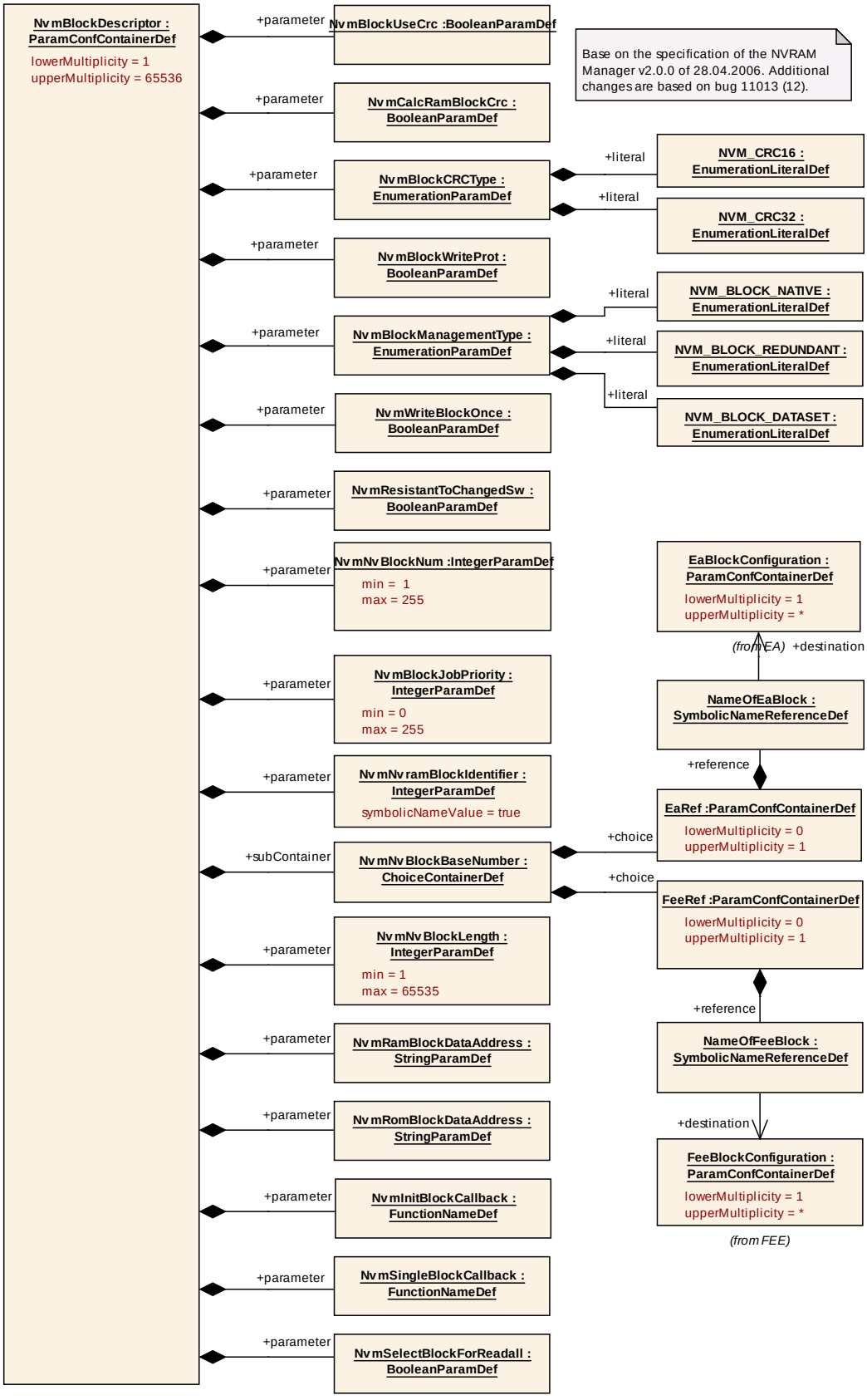


Figure 7.5: Block parameter description for the NVM module

NvmBlockDescriptor

SWS Item	[NVM061]
Container Name	NvmBlockDescriptor
Description	Container for a management structure to configure the composition of a given NVRAM Block Management Type. Its multiplicity describes thenumber of configured NVRAM blocks, one block is required to be configured. The NVRAM block descriptors are condensed in the NVRAM block descriptor table. The start address of the table is configured in the NVMCommon parameters.
Configuration Parameters	

Name last modified in: V1.0.0	NvmBlockCRCType {NVM_BLOCK_CRC_TYPE}		
Description	Defines CRC data width for the NVRAM block. Default: NVM_CRC16, i.e. CRC16 will be used if NVM_BLOCK_USE_CRC==TRUE		
Multiplicity	1		
Type	EnumerationParamDef		
Range	NVM_CRC16	(Default) CRC16 will be used if NVM_BLOCK_USE_CRC==TRUE.	
	NVM_CRC32	CRC32 is selected for this NVRAM block.	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	NvmBlockJobPriority {NVM_BLOCK_JOB_PRIORITY}		
Description	Defines the job priority for a NVRAM block		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	NvmBlockManagementType {NVM_BLOCK_MANAGEMENT_TYPE} [NVM062]		
Description	Defines the block management type for the NVRAM block.[NVM137]		
Multiplicity	1		
Type	EnumerationParamDef		
Range	NVM_BLOCK_DATASET	NVRAM block is configured to be of dataset type.	
	NVM_BLOCK_NATIVE	NVRAM block is configured to be of native type.	
	NVM_BLOCK_REDUNDANT	NVRAM block is configured to be of redundant type.	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	NvmBlockUseCrc {NVM_BLOCK_USE_CRC} [NVM036]		
Description	Defines CRC usage for the NVRAM block, i.e. memory space for CRC is reserved in RAM and NV memory.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	NvmBlockWriteProt {NVM_BLOCK_WRITE_PROT} [NVM033]		
Description	Defines an initial write protection of the NV block		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	NvmCalcRamBlockCrc {NVM_CALC_RAM_BLOCK_CRC} [NVM119]		
Description	Defines CRC (re)calculation for the permanent RAM block		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module dependency: NVM_BLOCK_GEN_CRC		

Name last modified in: V1.0.0	NvmInitBlockCallback {NVM_INIT_BLOCK_CALLBACK} [NVM116]		
Description	Entry address of a block specific callback routine which shall be called if no ROM data is available for initialization of the NVRAM block. If a NULL pointer is configured, no specific callback routine shall be called for initialization of the NVRAM block with default data.		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	NvmNvBlockLength {NVM_NV_BLOCK_LENGTH}		
Description	Defines the NV block data length in bytes		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 65535		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	NvmNvBlockNum {NVM_NV_BLOCK_NUM}		
Description	Defines the number of multiple NV blocks in a contiguous area according to the given block management type. 1-255 For NVRAM blocks to be configured of block management type NVM_BLOCK_DATASET 1 For NVRAM blocks to be configured of block management type NVM_BLOCK_NATIVE 2 For NVRAM blocks to be configured of block management type NVM_BLOCK_REDUNDANT		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 255		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module dependency: NVM_BLOCK_MANAGEMENT_TYPE		

Name last modified in: V1.0.0	NvmNvramBlockIdentifier		
Description	The SW-C invoke the NVRAM manager with a handle denoting a certain NVRAM block.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	NvmRamBlockDataAddress {NVM_RAM_BLOCK_DATA_ADDRESS}		
Description	Defines the start address of the RAM block data. If a NULL pointer is configured, no permanent RAM data block is available for the selected block management type.		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	NvmResistantToChangedSw {NVM_RESISTANT_TO_CHANGED_SW}		
Description	Defines whether a NVRAM block shall be treated resistant to configuration changes or not. If there is no default data available at configuration time then the application shall be responsible for providing the default initialization data. In this case the application has to use NvM_GetErrorStatus()to be able to distinguish between first initialization and corrupted data.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	NvmRomBlockDataAddress {NVM_ROM_BLOCK_DATA_ADDRESS}		
Description	Defines the start address of the ROM block data. If a NULL pointer is configured, no ROM block is available for the selected block management type.		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	NvmSelectBlockForReadall {NVM_SELECT_BLOCK_FOR_READALL} [NVM117]		
Description	Defines whether a NVRAM block shall be processed during NvM_ReadAll or not. This configuration parameter has only influence on those NVRAM blocks which are configured to have a permanent RAM block.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module dependency: NVM_RAM_BLOCK_DATA_ADDRESS		

Name last modified in: V1.0.0	NvmSingleBlockCallback {NVM_SINGLE_BLOCK_CALLBACK} [NVM113]		
Description	Entry address of the block specific callback routine which shall be invoked on termination of each asynchronous single block request.		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Name last modified in: V1.0.0	NvmWriteBlockOnce {NVM_WRITE_BLOCK_ONCE} [NVM072]		
Description	Defines write protection after first write. The NVRAM manager sets the write protection bit after the NV block was written the first time. This means that some of the NV blocks in the NVRAM should never be erased nor be replaced with the default ROM data after first initialization. [NVM276]		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: module		

Included Containers

Container Name	Multiplicity	Scope / Depedency
NvmNvBlockBase Number [EaRef, FeeRef]	1	

EaRef

SWS Item			
Container Name	EaRef		
Description	EEPROM Abstraction		
Configuration Parameters			
Name last modified in: V1.0.0	NameOfEaBlock		
Description	reference to EaBlock		
Multiplicity	1		
Type	Symbolic name reference to EaBlockConfiguration		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

FeeRef

SWS Item			
Container Name	FeeRef		
Description	Flash EEPROM Emulation		
Configuration Parameters			
Name last modified in: V1.0.0	NameOfFeeBlock		
Description	reference to FeeBlock		
Multiplicity	1		
Type	Symbolic name reference to FeeBlockConfiguration		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

7.3 Memory Hardware Abstraction

This chapter defines the configuration parameters of the Memory Abstraction Interface, the Flash EEPROM Emulation module and the EEPROM Abstraction module. The interface module is used in order to allow integration of multiple flash or EEPROM drivers into a single ECU. Flash drivers are always accessed through the Flash EEPROM Emulation module whereas EEPROM drivers are always accessed through the EEPROM Abstraction module.

7.3.1 Memory Abstraction Interface

In figure 7.6 the structure of the configuration object `MemIf` is depicted. It describes the configuration parameters which are available at the memory abstraction interface level.

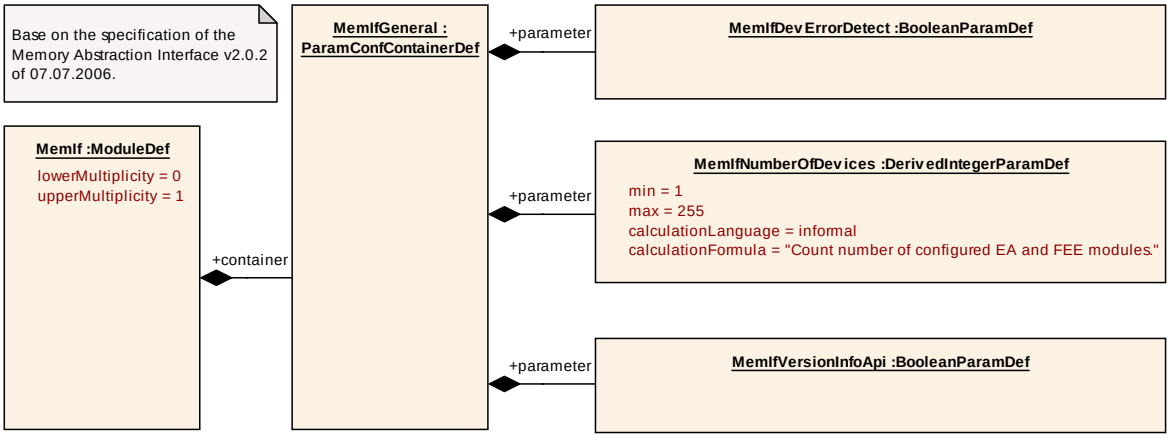


Figure 7.6: `MemIf` details

Module Name	MemIf	
Module Description	Configuration of the MemIf (Memory Abstraction Interface) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
MemIfGeneral	1	Container for general configuration parameters of the module which are always pre-compile.

MemIfGeneral

SWS Item	[MemIf025]		
Container Name	MemIfGeneral {MemIf_Configuration}		
Description	Container for general configuration parameters of the module which are always pre-compile.		
Configuration Parameters			

Name last modified in: V1.0.0	MemIfDevErrorDetect {MEMIF_DEV_ERROR_DETECT}		
Description	Pre-processor switch to enable or disable development error detection.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	MemIfNumberOfDevices {MEMIF_NUMBER_OF_DEVICES}		
Description	Actual number of underlying memory memory abstraction modules.		
Multiplicity	1		
Type	DerivedIntegerParamDef		
Range	1 .. 255		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	MemIfVersionInfoApi {MEMIF_VERSION_INFO_API} [MemIf032]		
Description	Pre-processor switch to enable or disable the API to read out the modules version information.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

7.3.2 FLASH EEPROM Emulation Module

Flash memory is accessed through the `Flash EEPROM Emulation Module` which abstracts from the underlying flash device by providing an EEPROM-like interface. The `Flash EEPROM Emulation Module` is configured by the `Fee` configuration object. In figure 7.7 the structure of the configuration object `Fee` is depicted.

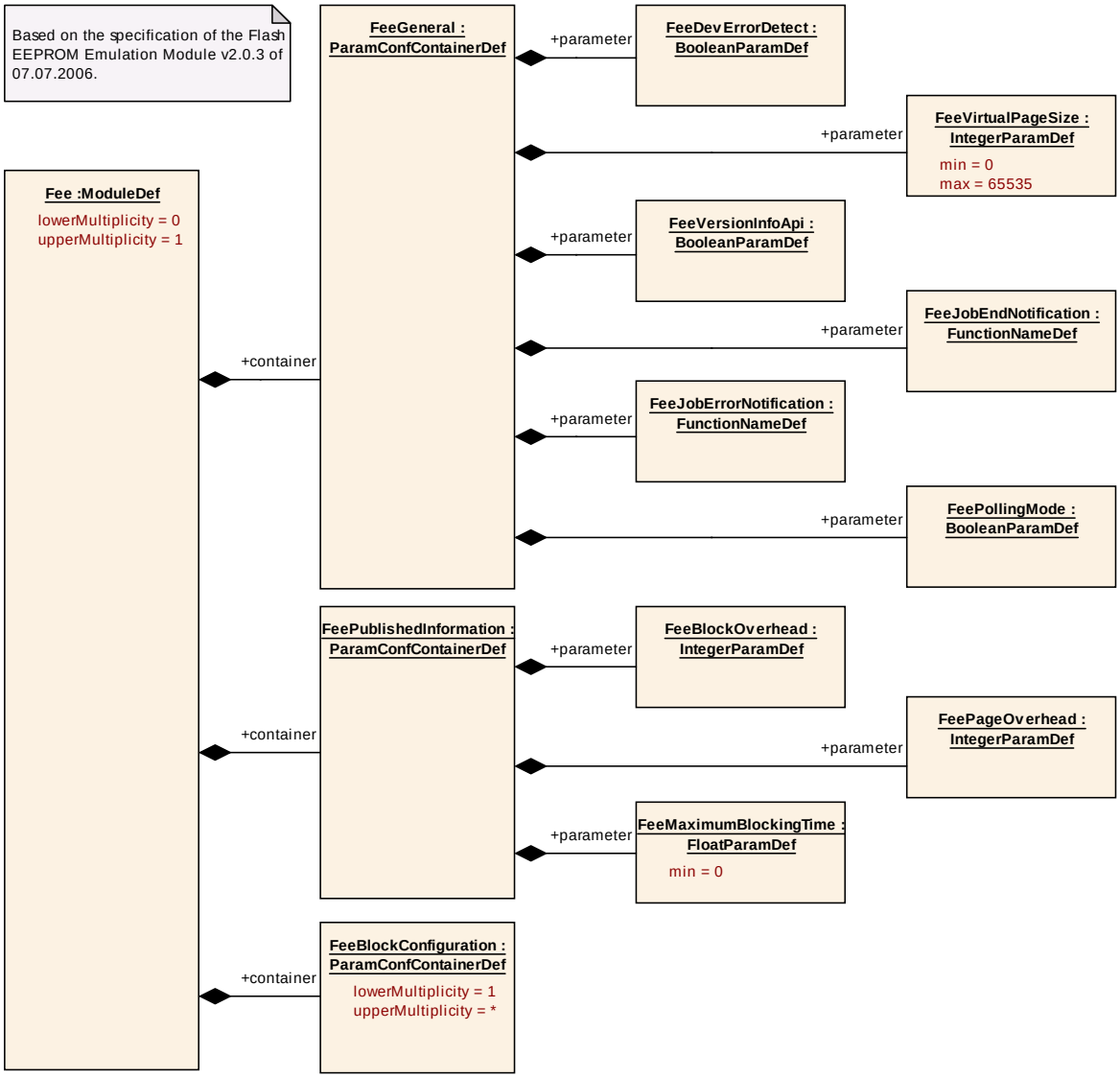


Figure 7.7: `Fee` details

Module Name	Fee	
Module Description	Configuration of the Fee (Flash EEPROM Emulation) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
FeeBlockConfiguration	1..*	Configuration of block specific parameters for the Flash EEPROM Emulation module.
FeeGeneral	1	Container for general parameters. These parameters are not specific to a block.
FeePublishedInformation	1	Additional published parameters not covered by CommonPublishedInformation container. Note that these parameters do not have any configuration class setting, since they are published information.

FeeGeneral

SWS Item	[FEE039]		
Container Name	FeeGeneral {FEE_ModuleConfiguration}		
Description	Container for general parameters. These parameters are not specific to a block.		
Configuration Parameters			
Name last modified in: V1.0.0	FeeDevErrorDetect {FEE_DEV_ERROR_DETECT}		
Description	Pre-processor switch to enable and disable development error detection		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FeeJobEndNotification {FEE_JOB_END_NOTIFICATION}		
Description	Mapped to the job end notification routine provided by the upper layer module (NvM_JobEndNotification).		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FeeJobErrorNotification {FEE_JOB_ERROR_NOTIFICATION}		
Description	Mapped to the job error notification routine provided by the upper layer module (NvM_JobErrorNotification).		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FeePollingMode {FEE_POLLING_MODE}		
Description	Pre-processor switch to enable and disable the polling mode for this module.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FeeVersionInfoApi {FEE_VERSION_INFO_API}		
Description	Pre-processor switch to enable and disable the API to read out the modules version information.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FeeVirtualPageSize {FEE_VIRTUAL_PAGE_SIZE}		
Description	The size in bytes to which logical blocks shall be aligned.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

FeePublishedInformation

SWS Item	[FEE043]		
Container Name	FeePublishedInformation		
Description	Additional published parameters not covered by CommonPublishedInformation container. Note that these parameters do not have any configuration class setting, since they are published information.		
Configuration Parameters			

Name last modified in: V1.0.0	FeeBlockOverhead {FEE_BLOCK_OVERHEAD}		
Description	Management overhead per logical block in bytes. Note: If the management overhead depends on the block size or block location a formula has to be provided that allows the configurator to calculate the management overhead correctly.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FeeMaximumBlockingTime {FEE_MAXIMUM_BLOCKING_TIME} [FEE070]		
Description	The maximum time the FEE module's API routines shall be blocked (delayed) by internal operations. Note: Internal operations in that case means operations that are not explicitly invoked from the upper layer module but need to be handled for proper operation of this module or the underlying memory driver. unit: [s] Configuration generators may need to convert this value into a hardware specific representation (e.g. timer ticks).		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FeePageOverhead {FEE_PAGE_OVERHEAD}		
Description	Management overhead per page in bytes. Note: If the management overhead depends on the block size or block location a formula has to be provided that allows the configurator to calculate the management overhead correctly.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

In figure 7.8 the structure of the configuration object `FeeBlockConfiguration` is depicted. This block contains the configuration parameters for one abstracted flash memory block.

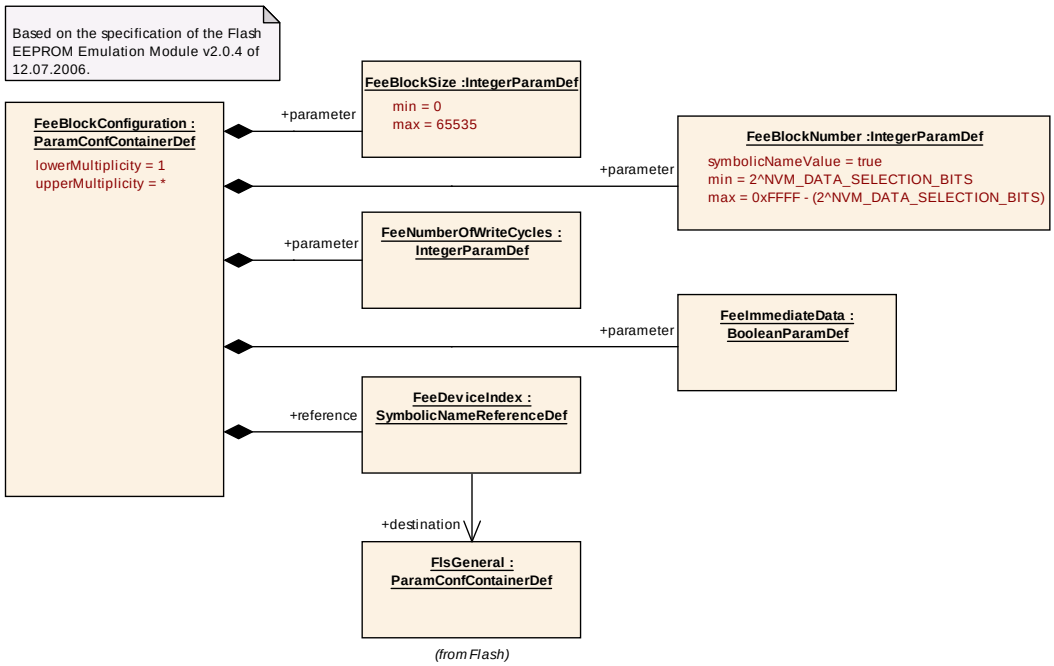


Figure 7.8: `FeeBlockConfiguration` details

FeeBlockConfiguration

SWS Item	[FEE040]		
Container Name	FeeBlockConfiguration {FEE_BlockConfiguration}		
Description	Configuration of block specific parameters for the Flash EEPROM Emulation module.		
Configuration Parameters			

Name last modified in: V1.0.0	FeeBlockNumber {FEE_BLOCK_NUMBER}		
Description	Number of logical block.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Range	2^ NVM_DATA_SELECTION_BITS .. 0xFFFF - (2^ NVM_DATA_SELECTION_BITS)		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FeeBlockSize {FEE_BLOCK_SIZE}		
Description	Size of a logical block in bytes.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FeeDeviceIndex {FEE_DEVICE_INDEX}		
Description			
Multiplicity	1		
Type	Symbolic name reference to FlsGeneral		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FeeImmediateData {FEE_IMMEDIATE_DATA}		
Description	Marker for high priority data.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FeeNumberOfWriteCycles {FEE_NUMBER_OF_WRITE_CYCLES}		
Description	Number of write cycles required for this block.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

7.3.3 EEPROM Abstraction Module

EEPROM memory is accessed through the `EEPROM Abstraction Module`. The `EEPROM Abstraction Module` is configured by the `Ea` configuration object. In figure 7.9 the structure of the configuration object `Ea` is depicted.

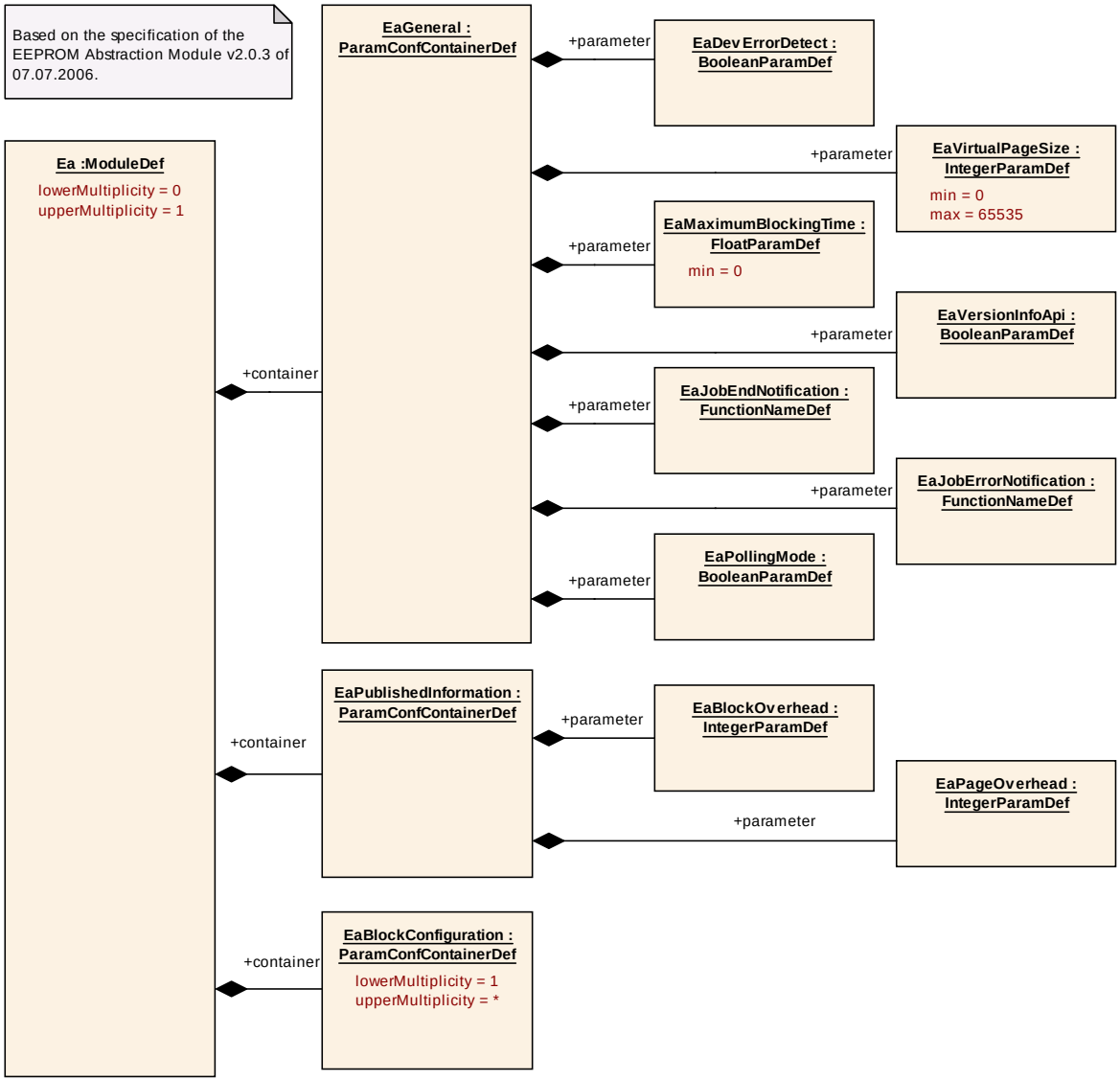


Figure 7.9: `Ea` details

Module Name	Ea	
Module Description	Configuration of the Ea (EEPROM Abstraction) module. The module shall abstract from the device specific addressing scheme and segmentation and provide the upper layers with a virtual addressing scheme and segmentation as well as a ”virtually” unlimited number of erase cycles.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**

Container Name	Multiplicity	Scope / Dependency
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
EaBlockConfiguration	1..*	Configuration of block specific parameters for the EEPROM abstraction module.
EaGeneral	1	General configuration of the EEPROM abstraction module. This container lists block independent configuration parameters.
EaPublishedInformation	1	Additional published parameters not covered by CommonPublishedInformation container. Note that these parameters do not have any configuration class setting, since they are published information.

EaGeneral

SWS Item	[EA039]		
Container Name	EaGeneral {EA_ModuleConfiguration}		
Description	General configuration of the EEPROM abstraction module. This container lists block independent configuration parameters.		
Configuration Parameters			

Name last modified in: V1.0.0	EaDevErrorDetect {EA_DEV_ERROR_DETECT}		
Description	Pre-processor switch to enable and disable development error detection.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	EaJobEndNotification {EA_JOB_END_NOTIFICATION}		
Description	Mapped to the job end notification routine provided by the upper layer module (NvM_JobEndNotification).		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	EaJobErrorNotification {EA_JOB_ERROR_NOTIFICATION}		
Description	Mapped to the job error notification routine provided by the upper layer module (NvM_JobErrorNotification).		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	EaMaximumBlockingTime {EA_MAXIMUM_BLOCKING_TIME} [EA070]		
Description	The maximum time the EA module's API routines shall be blocked (delayed) by internal operations. (EA070) unit: [s] Note: Configuration generators need to convert this value into a hardware specific representation (ticks) when generating the modules configuration header file.		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	EaPollingMode {EA_POLLING_MODE}		
Description	Pre-processor switch to enable and disable the polling mode for this module.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	EaVersionInfoApi {EA_VERSION_INFO_API}		
Description	Pre-processor switch to enable and disable the API to read out the modules version information.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	EaVirtualPageSize {EA_VIRTUAL_PAGE_SIZE}		
Description	The size in bytes to which logical blocks shall be aligned.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

EaPublishedInformation

SWS Item	[EA043]
Container Name	EaPublishedInformation
Description	Additional published parameters not covered by CommonPublishedInformation container. Note that these parameters do not have any configuration class setting, since they are published information.
Configuration Parameters	

Name last modified in: V1.0.0	EaBlockOverhead {EA_BLOCK_OVERHEAD}		
Description	Management overhead per logical block in bytes. Note: If the management overhead depends on the block size or block location a formula has to be provided that allows the configurator to calculate the management overhead correctly.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	EaPageOverhead {EA_PAGE_OVERHEAD}		
Description	Management overhead per page in bytes. Note: If the management overhead depends on the block size or block location a formula has to be provided that allows the configurator to calculate the management overhead correctly.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	–	
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

In figure 7.10 the structure of the configuration object `EaBlockConfiguration` is depicted. This block contains the configuration parameters for one abstracted EEPROM block.

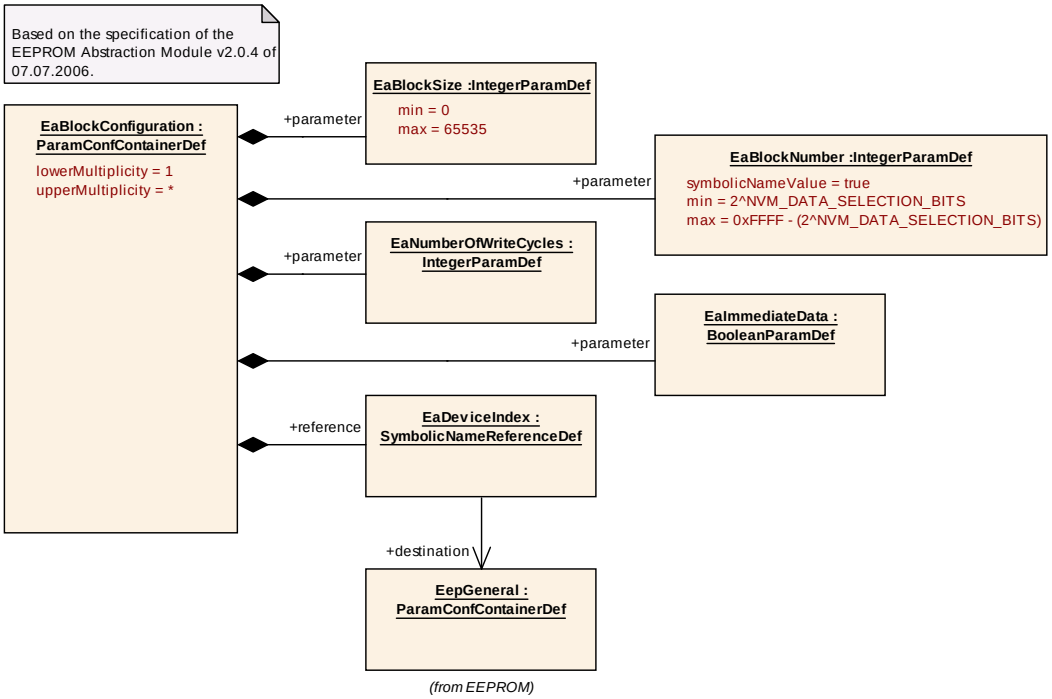


Figure 7.10: `EaBlockConfiguration` details

EaBlockConfiguration

SWS Item	[EA040]		
Container Name	EaBlockConfiguration {EA_BlockConfiguration}		
Description	Configuration of block specific parameters for the EEPROM abstraction module.		
Configuration Parameters			
Name last modified in: V1.0.0	EaBlockNumber {EA_BLOCK_NUMBER}		
Description	Number of logical block.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Range	2^ NVM_DATA_SELECTION_BITS .. 0xFFFF - (2^ NVM_DATA_SELECTION_BITS)		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	EaBlockSize {EA_BLOCK_SIZE}		
Description	Size of a logical block in bytes.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 65535		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	EaDeviceIndex {EA_DEVICE_INDEX}		
Description			
Multiplicity	1		
Type	Symbolic name reference to EepGeneral		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	EaImmediateData {EA_IMMEDIATE_DATA}		
Description	Marker for high priority data.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	EaNumberOfWriteCycles {EA_NUMBER_OF_WRITE_CYCLES}		
Description	Number of write cycles required for this block.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

7.4 Memory Drivers

This chapter defines the configuration parameters of the AUTOSAR Basic Software flash and EEPROM modules.

7.4.1 Flash Driver

Each flash driver is configured by one `Fls` configuration object. In figure 7.11 the structure of the configuration object `Fls` is depicted.

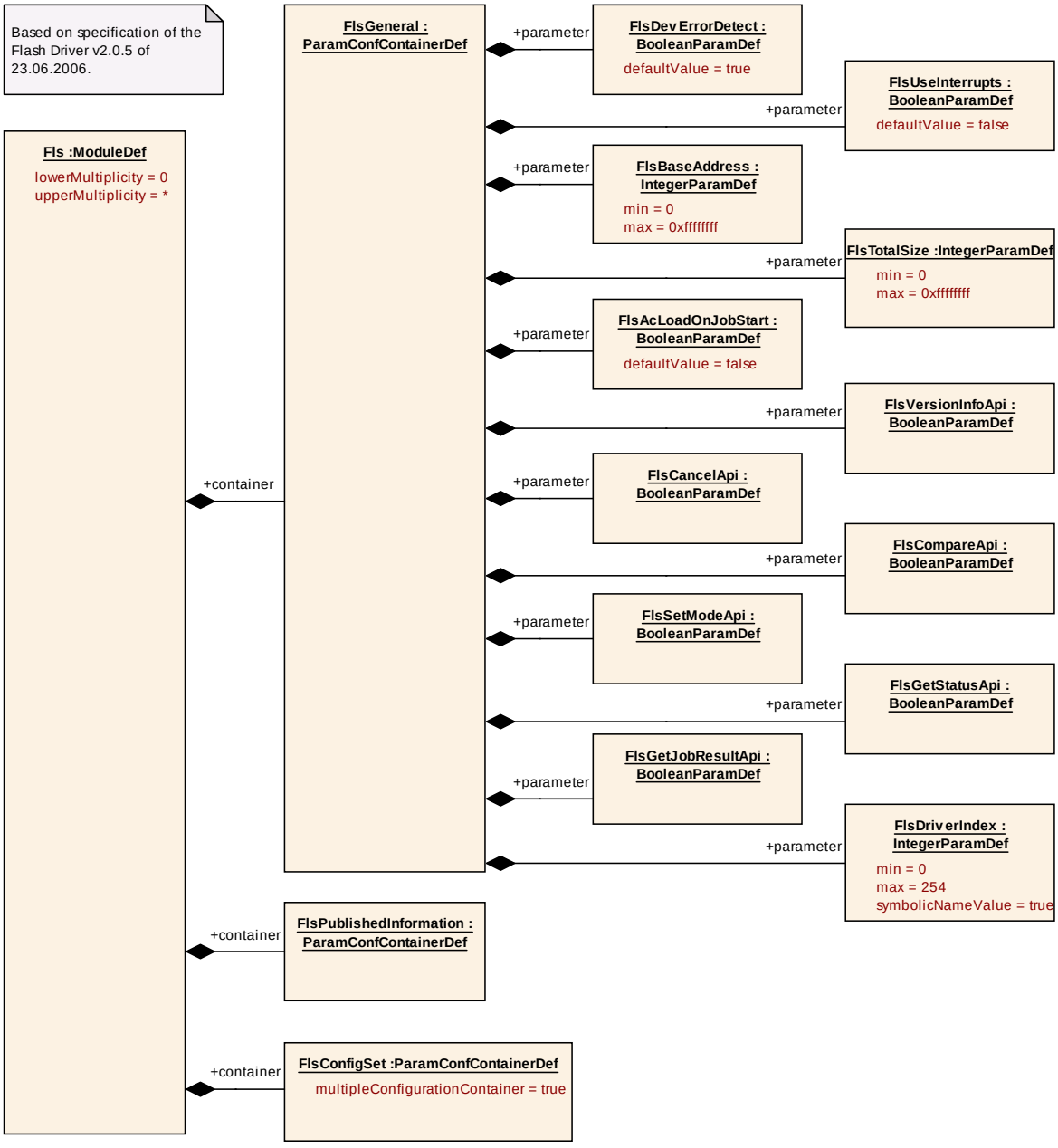


Figure 7.11: `Fls` details

Module Name	Fls	
Module Description	Configuration of the Fls (internal or external flash driver) module. Its multiplicity describes the number of flash drivers present, so there will be one container for each flash driver in the ECUC template. When no flash driver is present then the multiplicity is 0.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
FlsConfigSet	1	Container for runtime configuration parameters of the flash driver.
FlsGeneral	1	Container for general parameters of the flash driver. These parameters are always pre-compile.
FlsPublishedInformation	1	Additional published parameters not covered by CommonPublishedInformation container. Note that these parameters do not have any configuration class setting, since they are published information.

FlsGeneral

SWS Item	[FLS172]		
Container Name	FlsGeneral {Fls_ModuleConfiguration}		
Description	Container for general parameters of the flash driver. These parameters are always pre-compile.		
Configuration Parameters			
Name last modified in: V1.0.0	FlsAcLoadOnJobStart {FLS_AC_LOAD_ON_JOB_START}		
Description	The flash driver shall load the flash access code to RAM whenever an erase or write job is started and unload (overwrite) it after that job has been finished or canceled.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FlsBaseAddress {FLS_BASE_ADDRESS} [FLS169]		
Description	The flash memory start address (see also FLS118). FLS169: This parameter defines the lower boundary for read / write / erase and compare jobs.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 4294967295		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FlsCancelApi {FLS_CANCEL_API}		
Description	Compile switch to enable and disable the Fls_Cancel function.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FlsCompareApi {FLS_COMPARE_API}		
Description	Compile switch to enable and disable the Fls_Compare function.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FlsDevErrorDetect {FLS_DEV_ERROR_DETECT}		
Description	Pre-processor switch for enabling the development error detection and reporting (see FLS077).		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FlsDriverIndex		
Description	Index of the driver, used by FEE.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 254		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FlsGetJobResultApi {FLS_GET_JOB_RESULT_API}		
Description	Compile switch to enable and disable the Fls_GetJobResult function.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FlsGetStatusApi {FLS_GET_STATUS_API}		
Description	Compile switch to enable and disable the Fls_GetStatus function.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FlsSetModeApi {FLS_SET_MODE_API}		
Description	Compile switch to enable and disable the Fls_SetMode function.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FlsTotalSize {FLS_TOTAL_SIZE} [FLS170]		
Description	The total amount of flash memory in bytes (see also FLS118). FLS170: This parameter in conjunction with FLS_BASE_ADDRESS defines the upper boundary for read / write / erase and compare jobs.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 4294967295		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	FlsUseInterrupts {FLS_USE_INTERRUPTS}		
Description	Job processing triggered by hardware interrupt (only available if supported by underlying flash hardware).		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	dependency: Only available if supported by underlying flash hardware		

Name last modified in: V1.0.0	FlsVersionInfoApi {FLS_VERSION_INFO_API}		
Description	Compile switch to enable and disable the version information function.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
No Included Containers			

In figure 7.12 the structure of the additional published information object `FlsPublishedInformation` is depicted.

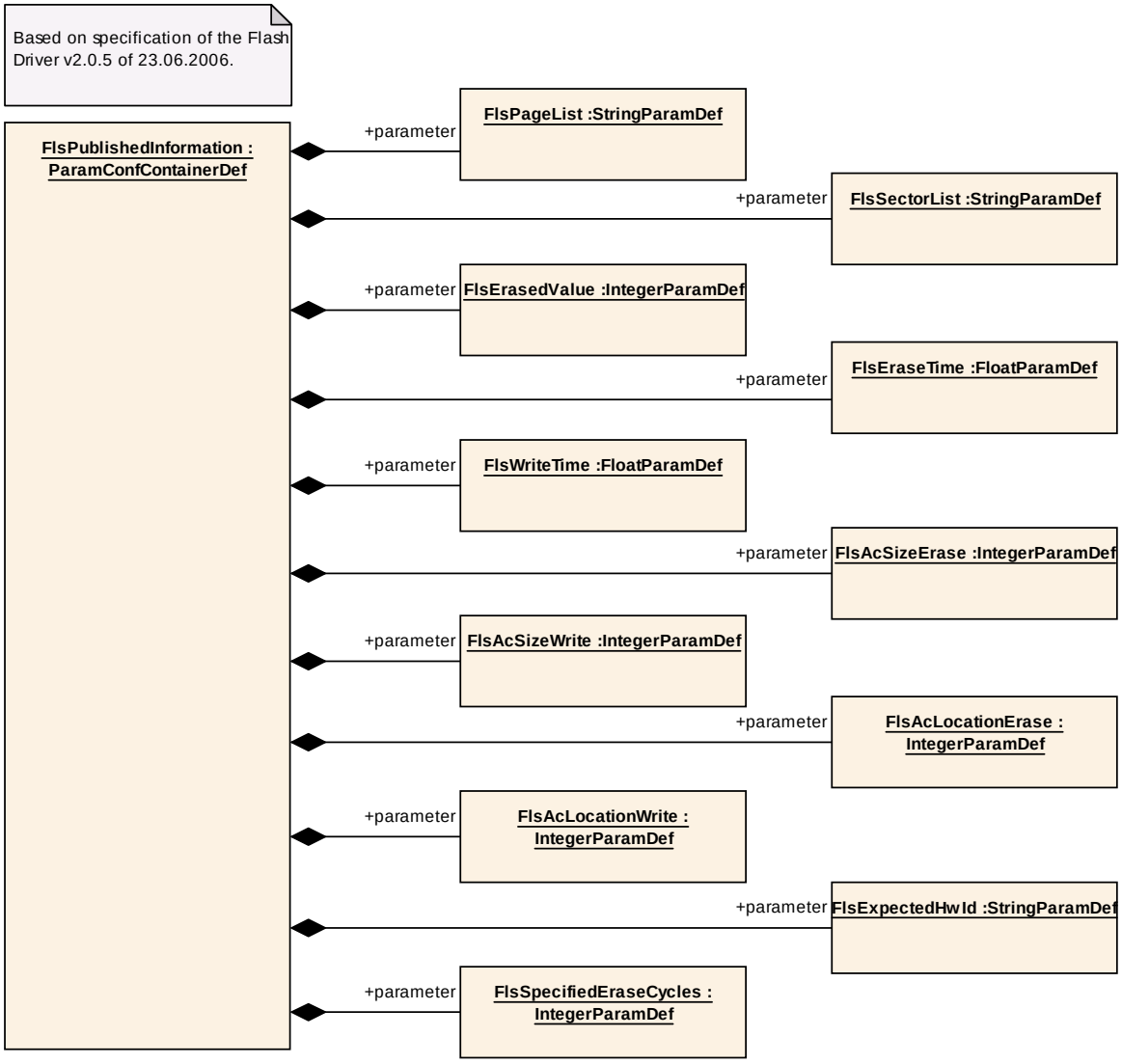


Figure 7.12: `FlsPublishedInformation` details

FlsPublishedInformation

SWS Item	[FLS178]		
Container Name	FlsPublishedInformation		
Description	Additional published parameters not covered by CommonPublishedInformation container. Note that these parameters do not have any configuration class setting, since they are published information.		
Configuration Parameters			

Name last modified in: V1.0.0	FlsAcLocationErase {FLS_AC_LOCATION_ERASE}		
Description	Position in RAM, to which the erase flash access code has to be loaded. Only relevant if the erase flash access code is not position independent. If this information is not provided it is assumed that the erase flash access code is position independent and that therefore the RAM position can be freely configured.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	FlsAcLocationWrite {FLS_AC_LOCATION_WRITE}		
Description	Position in RAM, to which the write flash access code has to be loaded. Only relevant if the write flash access code is not position independent. If this information is not provided it is assumed that the write flash access code is position independent and that therefore the RAM position can be freely configured.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	FlsAcSizeErase {FLS_AC_SIZE_ERASE}		
Description	Number of bytes in RAM needed for the erase flash access code.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	FlsAcSizeWrite {FLS_AC_SIZE_WRITE}		
Description	Number of bytes in RAM needed for the write flash access code.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	FlsEraseTime {FLS_ERASE_TIME}		
Description	Maximum time to erase one complete flash sector.		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	FlsErasedValue {FLS_ERASED_VALUE}		
Description	The contents of an erased flash memory cell.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	FlsExpectedHwId {FLS_EXPECTED_HW_ID}		
Description	Unique identifier of the hardware device that is expected by this driver (the device for which this driver has been implemented). Only relevant for external flash drivers.		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	FlsPageList {FLS_PAGE_LIST}		
Description	List of flash pages (smallest writable flash memory units) specifying start address (for checking of address alignment) and size (for checking write job sizes) for all different flash page sizes.		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	FlsSectorList {FLS_SECTOR_LIST}		
Description	List of flash sectors (smallest erasable flash memory units) specifying start address (for checking of address alignment) and size (for checking erase job sizes) for all different flash sector sizes.		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	FlsSpecifiedEraseCycles {FLS_SPECIFIED_ERASE_CYCLES} [FLS198]		
Description	Number of erase cycles specified for the flash device (usually given in the device data sheet). FLS198: If the number of specified erase cycles depends on the operating environment (temperature, voltage, ...) during reprogramming of the flash device, the minimum number for which a data retention of at least 15 years over the temperature range from -40°C .. +125°C can be guaranteed shall be given. Note: If there are different numbers of specified erase cycles for different flash sectors of the device this parameter has to be extended to a parameter list (similar to the sector list above).		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	FlsWriteTime {FLS_WRITE_TIME}		
Description	Maximum time to program one complete flash page.		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

No Included Containers

In figure 7.13 the structure of the configuration object `FlsConfigSet` is depicted.

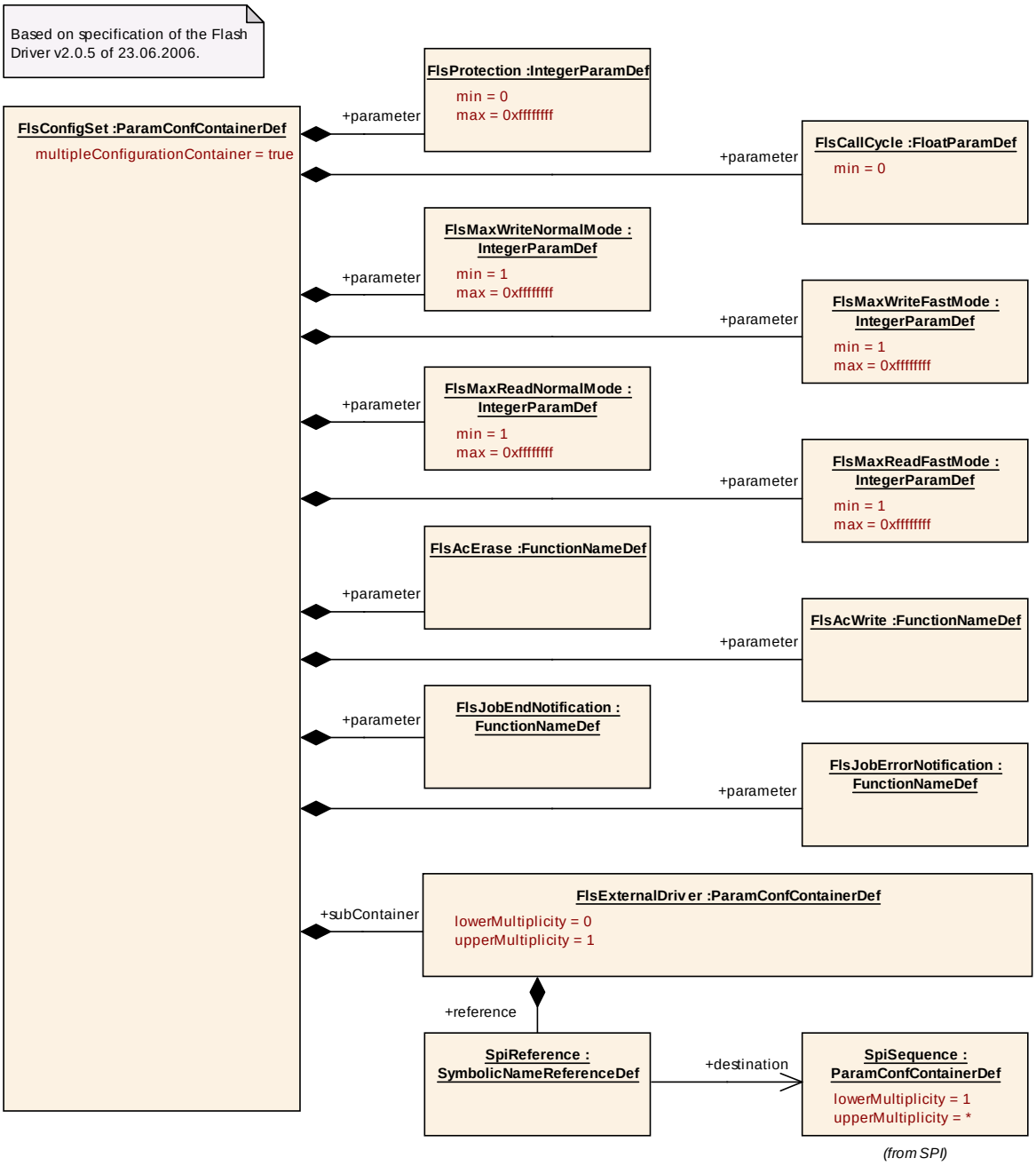


Figure 7.13: `FlsConfigSet` details

FlsConfigSet

SWS Item	[FLS174]		
Container Name	FlsConfigSet		
Description	Container for runtime configuration parameters of the flash driver.		
Configuration Parameters			

Name last modified in: V1.0.0	FlsAcErase {FLS_AC_ERASE}		
Description	Function pointer to erase flash access code. Address offset in RAM to which the erase flash access code shall be loaded.		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	Variant PC
	Link time	–	
	Post-build time	X	Variant PB
Scope / Dependency			

Name last modified in: V1.0.0	FlsAcWrite {FLS_AC_WRITE}		
Description	Function pointer to write flash access code. Address offset in RAM to which the write flash access code shall be loaded.		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	Variant PC
	Link time	–	
	Post-build time	X	Variant PB
Scope / Dependency			

Name last modified in: V1.0.0	FlsCallCycle {FLS_CALL_CYCLE}		
Description	Cycle time of calls of the flash driver's main function. This value is given in seconds. A tool generating the flash driver's configuration header file converts this value to a HW specific representation (e.g. timer ticks).		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	Variant PC
	Link time	–	
	Post-build time	X	Variant PB
Scope / Dependency	dependency: Only relevant if deadline monitoring for internal functionality has to be done in software (e.g. erase / write timings)		

Name last modified in: V1.0.0	FlsJobEndNotification {FLS_JOB_END_NOTIFICATION}		
Description	Mapped to the job end notification routine provided by some upper layer module, typically the Fee module.		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	Variant PC
	Link time	–	
	Post-build time	X	Variant PB
Scope / Dependency			

Name last modified in: V1.0.0	FlsJobErrorNotification {FLS_JOB_ERROR_NOTIFICATION}		
Description	Mapped to the job error notification routine provided by some upper layer module, typically the Fee module.		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	Variant PC
	Link time	–	
	Post-build time	X	Variant PB
Scope / Dependency			

Name last modified in: V1.0.0	FlsMaxReadFastMode {FLS_MAX_READ_FAST_MODE}		
Description	The maximum number of bytes to read or compare in one cycle of the flash driver's job processing function in fast mode.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 4294967295		
Configuration Class	**Pre-compile time**	X	Variant PC
	Link time	–	
	Post-build time	X	Variant PB
Scope / Dependency	dependency: The minimum number might depend on the underlying flash device or communication driver, e.g. if the access to an external flash device is done via SPI and the minimum transfer size on SPI is four bytes.		

Name last modified in: V1.0.0	FlsMaxReadNormalMode {FLS_MAX_READ_NORMAL_MODE}		
Description	The maximum number of bytes to read or compare in one cycle of the flash driver's job processing function in normal mode.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 4294967295		
Configuration Class	**Pre-compile time**	X	Variant PC
	Link time	–	
	Post-build time	X	Variant PB
Scope / Dependency	dependency: The minimum number might depend on the underlying flash device or communication driver, e.g. if the access to an external flash device is done via SPI and the minimum transfer size on SPI is four bytes.		

Name last modified in: V1.0.0	FlsMaxWriteFastMode {FLS_MAX_WRITE_FAST_MODE}		
Description	The maximum number of bytes to write in one cycle of the flash driver's job processing function in fast mode.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 4294967295		
Configuration Class	**Pre-compile time**	X	Variant PC
	Link time	–	
	Post-build time	X	Variant PB
Scope / Dependency	dependency: FLS182: This value has to correspond to the settings in FLS_PAGE_LIST. The minimum number is defined by the size of one flash page and therefore depends on the underlying flash device.		

Name last modified in: V1.0.0	FlsMaxWriteNormalMode {FLS_MAX_WRITE_NORMAL_MODE}		
Description	The maximum number of bytes to write in one cycle of the flash driver's job processing function in normal mode.		
Multiplicity	1		
Type	IntegerParamDef		
Range	1 .. 4294967295		
Configuration Class	**Pre-compile time**	X	Variant PC
	Link time	–	
	Post-build time	X	Variant PB
Scope / Dependency	dependency: FLS176: This value has to correspond to the settings in FLS_PAGE_LIST. The minimum number is defined by the size of one flash page and therefore depends on the underlying flash device.		

Name last modified in: V1.0.0	FlsProtection {FLS_PROTECTION}		
Description	Erase/write protection settings. Only relevant if supported by hardware.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 4294967295		
Configuration Class	**Pre-compile time**	X	Variant PC
	Link time	–	
	Post-build time	X	Variant PB
Scope / Dependency	dependency: Only relevant if supported by hardware.		
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
FlsExternalDriver	0..1		

Differences between the configuration of internal and external flash drivers are related to the `FlsExternalDriver` subcontainer. The parameters within this container are not required for an internal flash drivers. Hence this container is not present for internal flash drivers - its multiplicity may be 0. Whether the flash driver applies to an internal or external flash device can be determined through published information in the module's BSWMD template.

The details of `FlsExternalDriver` container are described in the following table.

FlsExternalDriver

SWS Item	[FLS189]		
Container Name	FlsExternalDriver		
Description	This container is present for external flash drivers only. Internal flash drivers do not use the parameter listed in this container, hence its multiplicity is 0 for internal drivers.		
Configuration Parameters			

Name last modified in: V1.0.0	FlsMaxSuspendTime		
Description	The maximum read access blocking time (suspend time) in milliseconds. A tool generating the flash driver configuration header file may need to convert the value of this parameter to a HW specific representation. Parameter is present for external drivers only.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	Variant PC
	Link time	–	
	Post-build time	X	Variant PB
Scope / Dependency			

Name last modified in: V1.0.0	SpiReference		
Description	Reference to SPI sequence (required for external flash drivers).		
Multiplicity	1		
Type	Symbolic name reference to SpiSequence		
Configuration Class	**Pre-compile time**	X	Variant PC
	Link time	–	
	Post-build time	X	Variant PB
Scope / Dependency **No Included Containers**			

7.4.2 EEPROM Driver

In figure 7.14 the structure of the configuration object `Eep` is depicted.

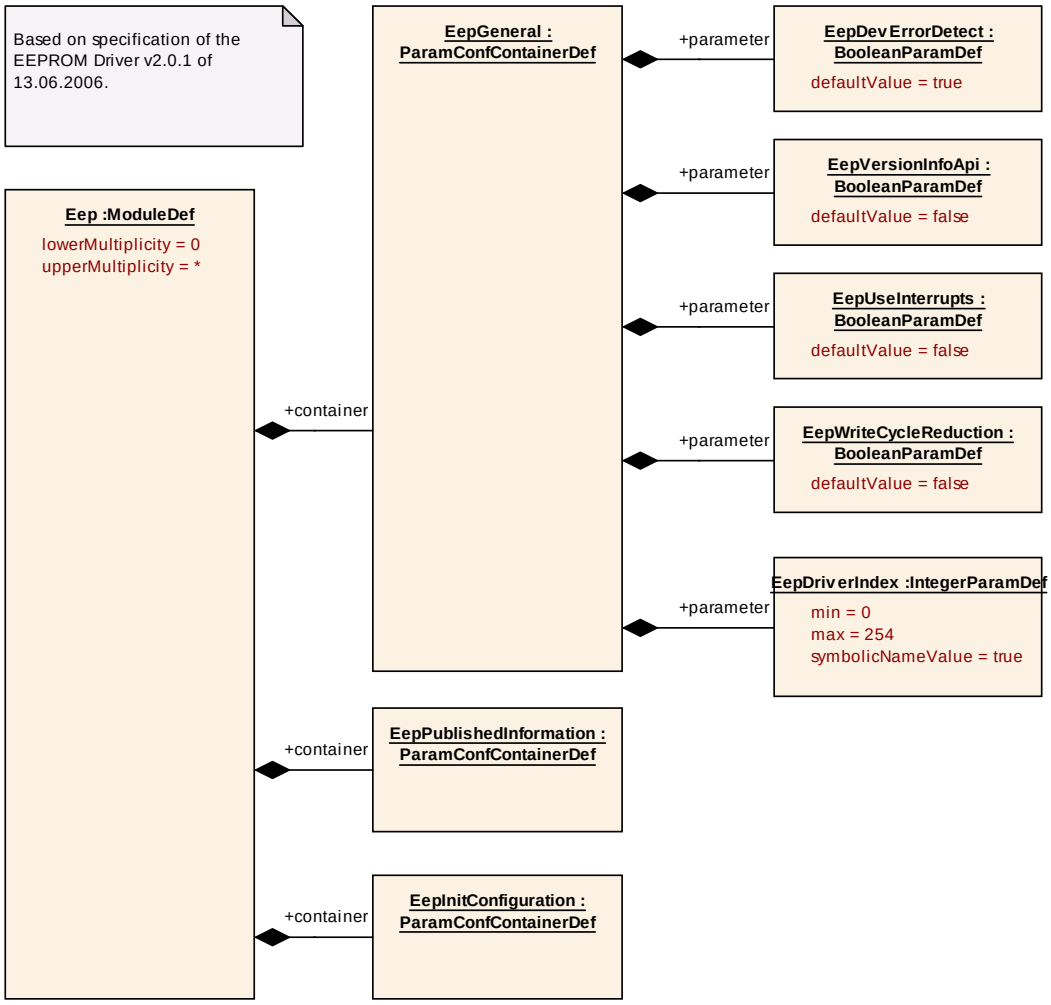


Figure 7.14: `Eep` details

Module Name	Eep	
Module Description	Configuration of the Eep (internal or external EEPROM driver) module. Its multiplicity describes the number of EEPROM drivers present, so there will be one container for each EEPROM driver in the ECUC template. When no EEPROM driver is present then the multiplicity is 0.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
EepGeneral	1	Container for general configuration parameters of the EEPROM driver. These parameters are always pre-compile.

Container Name	Multiplicity	Scope / Dependency
EepInitConfiguration	1	Container for runtime configuration parameters of the EEPROM driver.
EepPublishedInformation	1	Additional published parameters not covered by CommonPublishedInformation container. Note that these parameters do not have any configuration class setting, since they are published information.

EepGeneral

SWS Item	[EEP085]		
Container Name	EepGeneral {EepGeneralConfiguration}		
Description	Container for general configuration parameters of the EEPROM driver. These parameters are always pre-compile.		
Configuration Parameters			
Name last modified in: V1.0.0	EepDevErrorDetect {EEP_DEV_ERROR_DETECT}		
Description	Switches the Development Error Detection and Notification ON or OFF.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
Name last modified in: V1.0.0	EepDriverIndex		
Description	Index of the driver, used by EA.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 254		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
Name last modified in: V1.0.0	EepUseInterrupts {EEP_USE_INTERRUPTS}		
Description	Switches to activate or deactivate interrupt controlled job processing.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	dependency: Usually, this is only supported by some internal EEPROM peripherals.		
Name last modified in: V1.0.0	EepVersionInfoApi {EEP_VERSION_INFO_API}		
Description	Switches the Eep_GetVersionInfo function ON or OFF.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	EepWriteCycleReduction {EEP_WRITE_CYCLE_REDUCTION}		
Description	Switches to activate or deactivate write cycle reduction (EEPROM value is read and compared before being overwritten).		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency **No Included Containers**			

In figure 7.15 the structure of the additional published information object `EepPublishedInformation` is depicted.

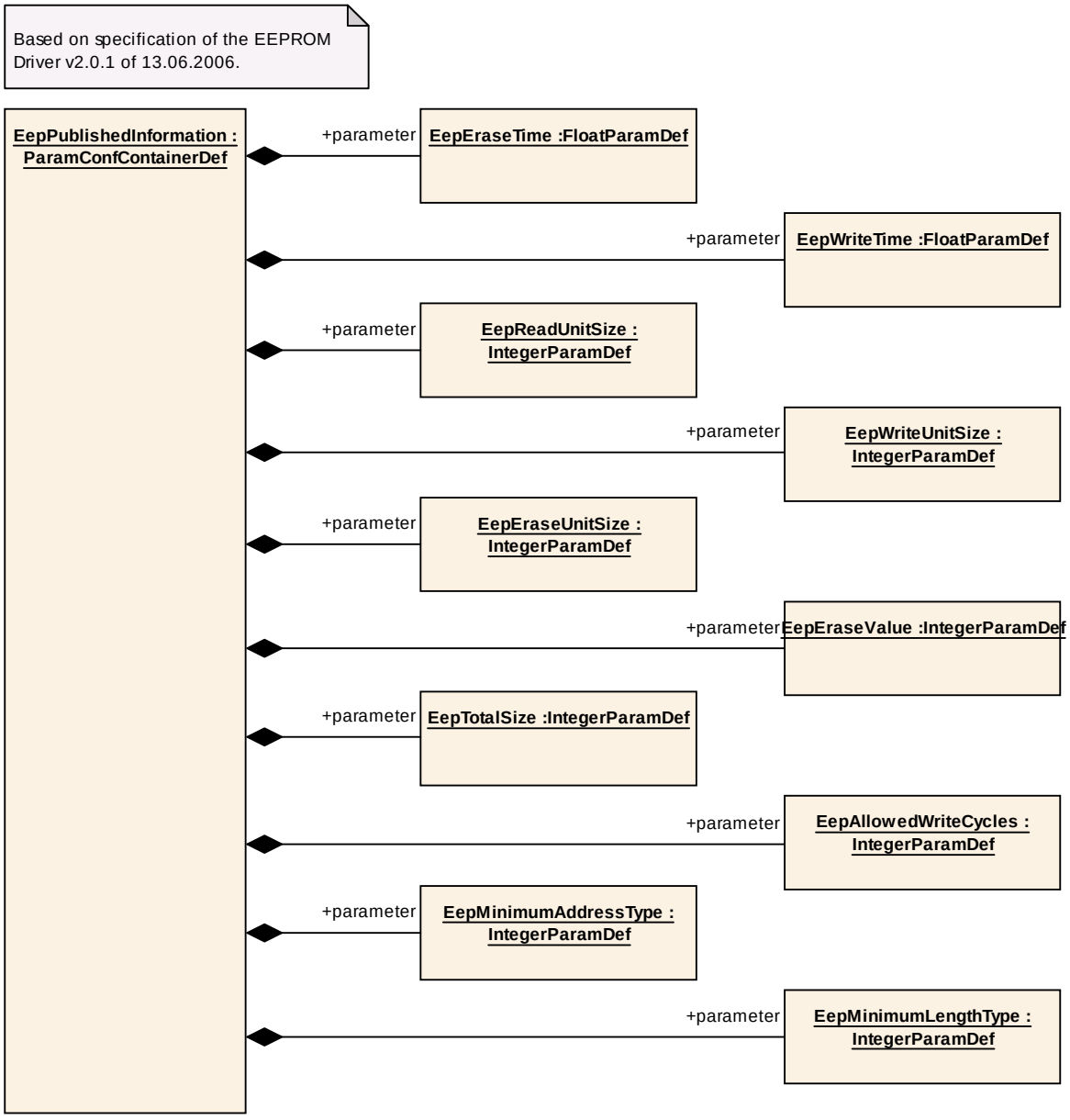


Figure 7.15: `EepPublishedInformation` details

EepPublishedInformation

SWS Item	[EEP111]		
Container Name	EepPublishedInformation		
Description	Additional published parameters not covered by CommonPublishedInformation container. Note that these parameters do not have any configuration class setting, since they are published information.		
Configuration Parameters			

Name last modified in: V1.0.0	EepAllowedWriteCycles {EEP_ALLOWED_WRITE_CYCLES}		
Description	Specified maximum number of write cycles under worst case conditions of specific EEPROM hardware (e.g. +90°C)		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	EepEraseTime {EEP_ERASE_TIME}		
Description	Time for erasing one EEPROM data unit.		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	EepEraseUnitSize {EEP_ERASE_UNIT_SIZE}		
Description	Size of smallest erasable EEPROM data unit in bytes.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	EepEraseValue {EEP_ERASE_VALUE}		
Description	Value of an erased EEPROM cell.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	EepMinimumAddressType {EEP_MINIMUM_ADDRESS_TYPE}		
Description	Minimum expected size of Eep_AddressType.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	EepMinimumLengthType {EEP_MINIMUM_LENGTH_TYPE}		
Description	Minimum expected size of Eep_LengthType.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	EepReadUnitSize {EEP_READ_UNIT_SIZE}		
Description	Size of smallest readable EEPROM data unit in bytes.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	EepTotalSize {EEP_TOTAL_SIZE}		
Description	Total size of EEPROM in bytes.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	EepWriteTime {EEP_WRITE_TIME}		
Description	Time for writing one EEPROM data unit.		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			

Name last modified in: V1.0.0	EepWriteUnitSize {EEP_WRITE_UNIT_SIZE}		
Description	Size of smallest writeable EEPROM data unit in bytes.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Published Information**	X	All Variants
Scope / Dependency			
No Included Containers			

In figure 7.16 the structure of the configuration object `EepInitConfiguration` is depicted.

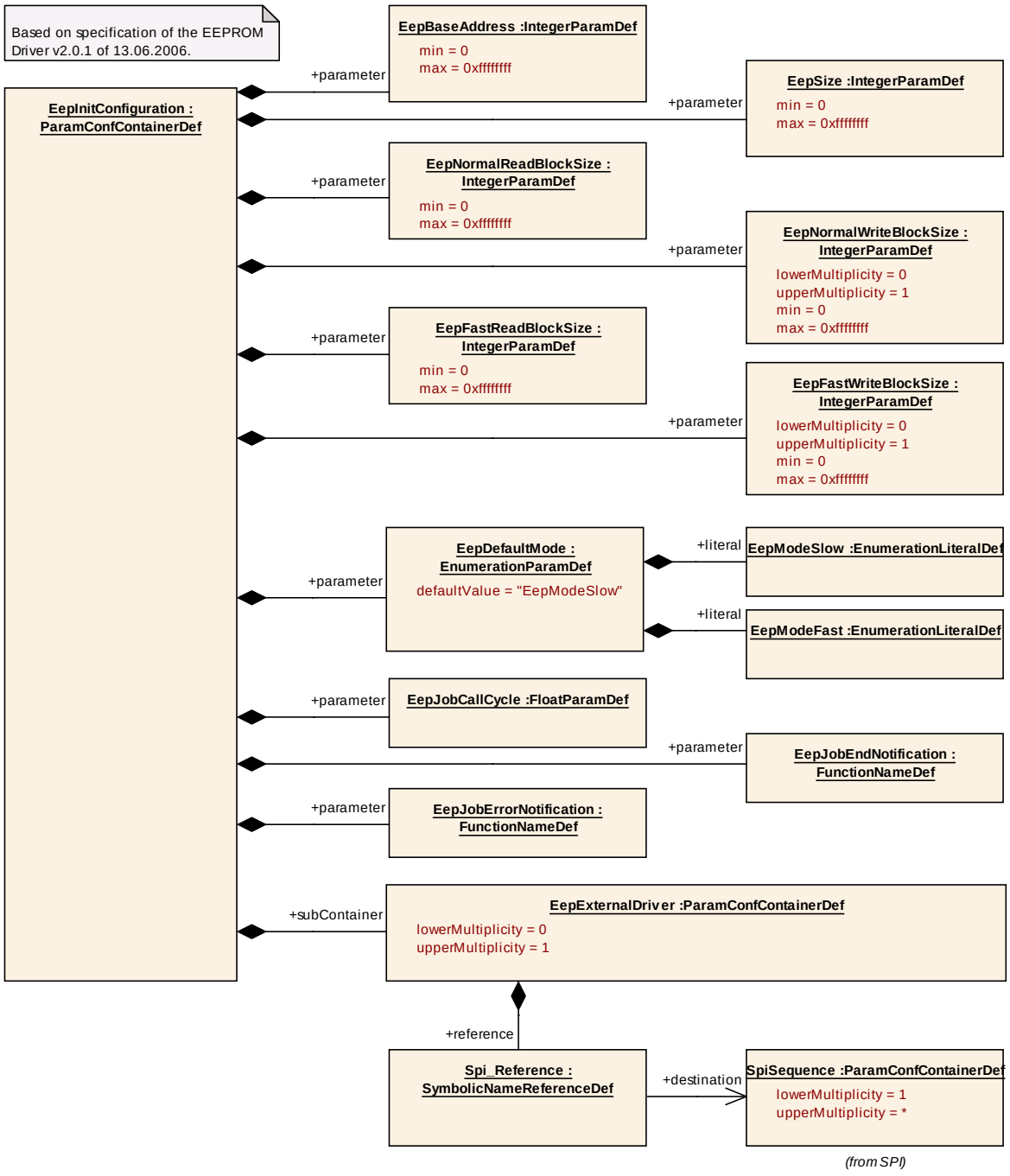


Figure 7.16: `EepInitConfiguration` details

EepInitConfiguration

SWS Item	[EEP039]		
Container Name	EepInitConfiguration {EepInitConfiguration}		
Description	Container for runtime configuration parameters of the EEPROM driver.		
Configuration Parameters			

Name last modified in: V1.0.0	EepBaseAddress {EEP_BASE_ADDRESS}		
Description	This parameter is the EEPROM device base address.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 4294967295		
Configuration Class	**Pre-compile time**	X	Variant PC
	Link time	X	Variant LT
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	EepDefaultMode {EEP_DEFAULT_MODE}		
Description	This parameter is the default EEPROM device mode after initialization.		
Multiplicity	1		
Type	EnumerationParamDef		
Range	EepModeFast	The driver is working in fast mode (fast read access / SPI burst access).	
	EepModeSlow	The driver is working in slow mode.	
Configuration Class	**Pre-compile time**	X	Variant PC
	Link time	X	Variant LT
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	EepFastReadBlockSize {EEP_FAST_READ_BLOCK_SIZE}		
Description	This parameter is the number of bytes read within one job processing cycle in fast mode.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 4294967295		
Configuration Class	**Pre-compile time**	X	Variant PC
	Link time	X	Variant LT
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	EepFastWriteBlockSize {EEP_FAST_WRITE_BLOCK_SIZE}		
Description	This parameter is the number of bytes written within one job processing cycle in fast mode.		
Multiplicity	0..1		
Type	IntegerParamDef		
Range	0 .. 4294967295		
Configuration Class	**Pre-compile time**	X	Variant PC
	Link time	X	Variant LT
	Post-build time	–	
Scope / Dependency	dependency: This parameter is optional and only available if the hardware allows writing several bytes in one step (e.g. external EEPROMs with burst mode capability).		

Name last modified in: V1.0.0	EepJobCallCycle {EEP_JOB_CALL_CYCLE}		
Description	This parameter is the call cycle of the job processing function during write/erase operations. Unit: [s]		
Multiplicity	1		
Type	FloatParamDef		
Configuration Class	**Pre-compile time**	X	Variant PC
	Link time	X	Variant LT
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	EepJobEndNotification {EEP_JOB_END_NOTIFICATION}		
Description	This parameter is a reference to a callback function for positive job result (see EEP045).		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	Variant PC
	Link time	X	Variant LT
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	EepJobErrorNotification {EEP_JOB_ERROR_NOTIFICATION}		
Description	This parameter is a reference to a callback function for negative job result (see EEP046).		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	Variant PC
	Link time	X	Variant LT
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	EepNormalReadBlockSize {EEP_NORMAL_READ_BLOCK_SIZE}		
Description	This parameter is the number of bytes read within one job processing cycle in normal mode.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 4294967295		
Configuration Class	**Pre-compile time**	X	Variant PC
	Link time	X	Variant LT
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	EepNormalWriteBlockSize {EEP_NORMAL_WRITE_BLOCK_SIZE}		
Description	Number of bytes written within one job processing cycle in normal mode. This parameter is optional and only available if the hardware allows configuration. Whether this feature is supported by the EEP driver can be determined through published information in the BSW Module Description Template. The multiplicity of this parameter is 0 if this option is not supported by the driver.		
Multiplicity	0..1		
Type	IntegerParamDef		
Range	0 .. 4294967295		
Configuration Class	**Pre-compile time**	X	Variant PC
	Link time	X	Variant LT
	Post-build time	–	
Scope / Dependency	dependency: This parameter is optional and only available if the hardware allows configuration.		

Name last modified in: V1.0.0	EepSize {EEP_SIZE}		
Description	This parameter is the used size of EEPROM device in bytes.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 4294967295		
Configuration Class	**Pre-compile time**	X	Variant PC
	Link time	X	Variant LT
	Post-build time	–	
Scope / Dependency			
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
EepExternalDriver	0..1		

Differences between the configuration of internal and external EEPROM drivers are related to the `EepExternalDriver` subcontainer. The parameters within this container are not required for an internal EEPROM drivers. Hence this container is not present for internal EEPROM drivers - its multiplicity may be 0. Whether the EEPROM driver applies to an internal or external EEPROM device can be determined through published information in the module's BSWMD template.

The details of `EepExternalDriver` container are described in the following table.

EepExternalDriver

SWS Item	[EEP094]
Container Name	EepExternalDriver
Description	This container is present for external EEPROM drivers only. Internal EEPROM drivers do not use the parameter listed in this container, hence its multiplicity is 0 for internal drivers.
Configuration Parameters	

Name last modified in: V1.0.0	Spi_Reference		
Description	Reference to SPI sequence (required for external EEPROM drivers).		
Multiplicity	1		
Type	Symbolic name reference to SpiSequence		
Configuration Class	**Pre-compile time**	X	Variant PC
	Link time	X	Variant LT
	Post-build time	–	
Scope / Dependency			
No Included Containers			

8 Miscellaneous

8.1 BSW Scheduler Module

TODO *This chapter needs to be reworked when the BSWMD-T is released. There are several potential references from the SchM module into the BSWMD-T that needs to be defined.*

The configuration model for the BSW Scheduler module is shown in figure 8.1.

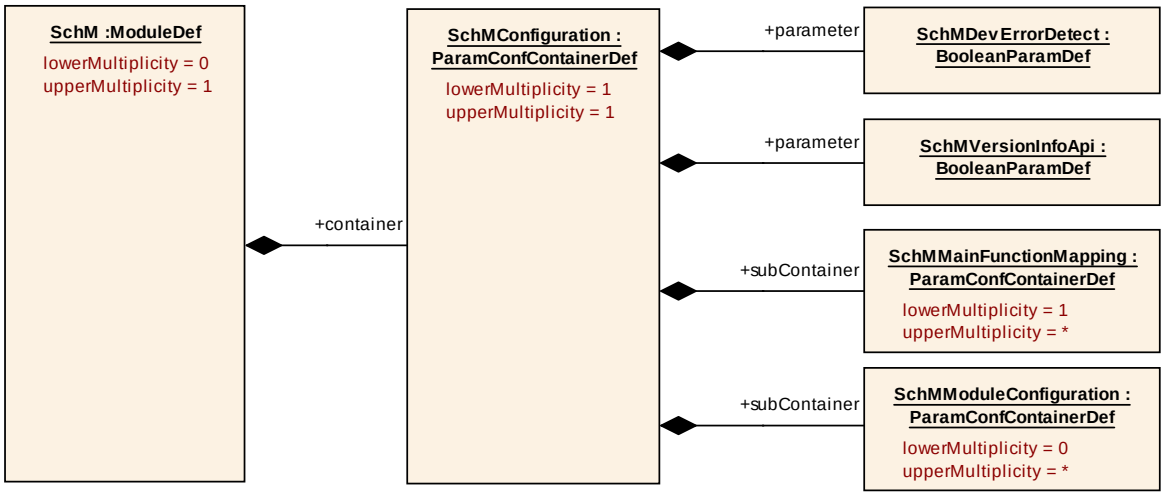


Figure 8.1: BSW Scheduler Module

Module Name	SchM	
Module Description	Configuration of the SchM (BSW Scheduler) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
SchMConfiguration	1	Top container for the entire configuration set.

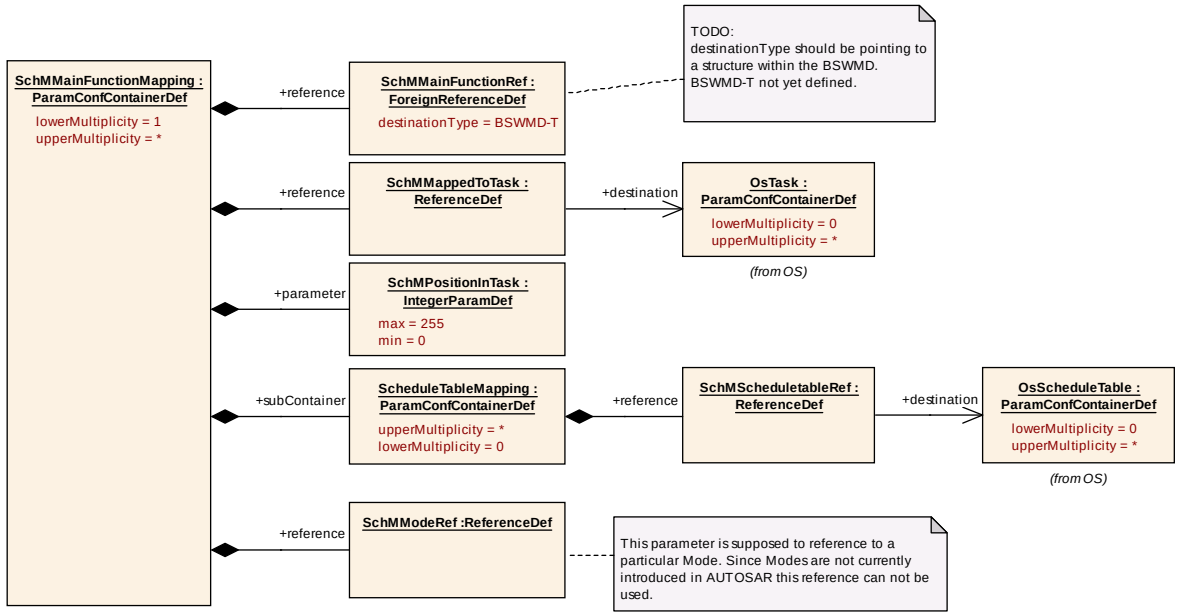


Figure 8.2: SchM Main Function Mapping

SchMMainFunctionMapping

SWS Item	[INTEGR068]
Container Name	SchMMainFunctionMapping {MainFunctionMapping}
Description	Maps a MainFunction onto one OS Task.
Configuration Parameters	

Name last modified in: V1.1.0	SchMMainFunctionRef {SCHM_MAINFUNCTION_REF}		
Description	The parameter depicts the reference to the MainFunction. TODO: destinationType should be pointing to a structure within the BSWMD. BSWMD-T not yet defined.		
Multiplicity	1		
Type	Foreign reference to BSWMD-T		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	SchMMappedToTask {SCHM_MAPPED_TO_TASK}		
Description	The parameter depicts the reference to the OS Task the MainFunction is mapped to.		
Multiplicity	1		
Type	Reference to OsTask		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	SchMModeRef {SCHM_MODE_REF}		
Description	This parameter is supposed to reference to a particular Mode. Since Modes are not currently introduced in AUTOSAR this reference can not be used.		
Multiplicity	1		
Type	!!! Wrong specified reference destination !!!		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.1.0	SchMPositionInTask {SCHM_POSITION_IN_TASK}		
Description	The parameter depicts the specific Position of the MainFunction within the OS Task. The dependencies between the main processing functions mapped to the same task have been resolved by the configuration (process) beforehand. The result of this resolution is the order of the main processing functions within this task.		
Multiplicity	1		
Type	IntegerParamDef		
Range	0 .. 255		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Included Containers

Container Name	Multiplicity	Scope / Depedency
ScheduleTableMapping	0..*	

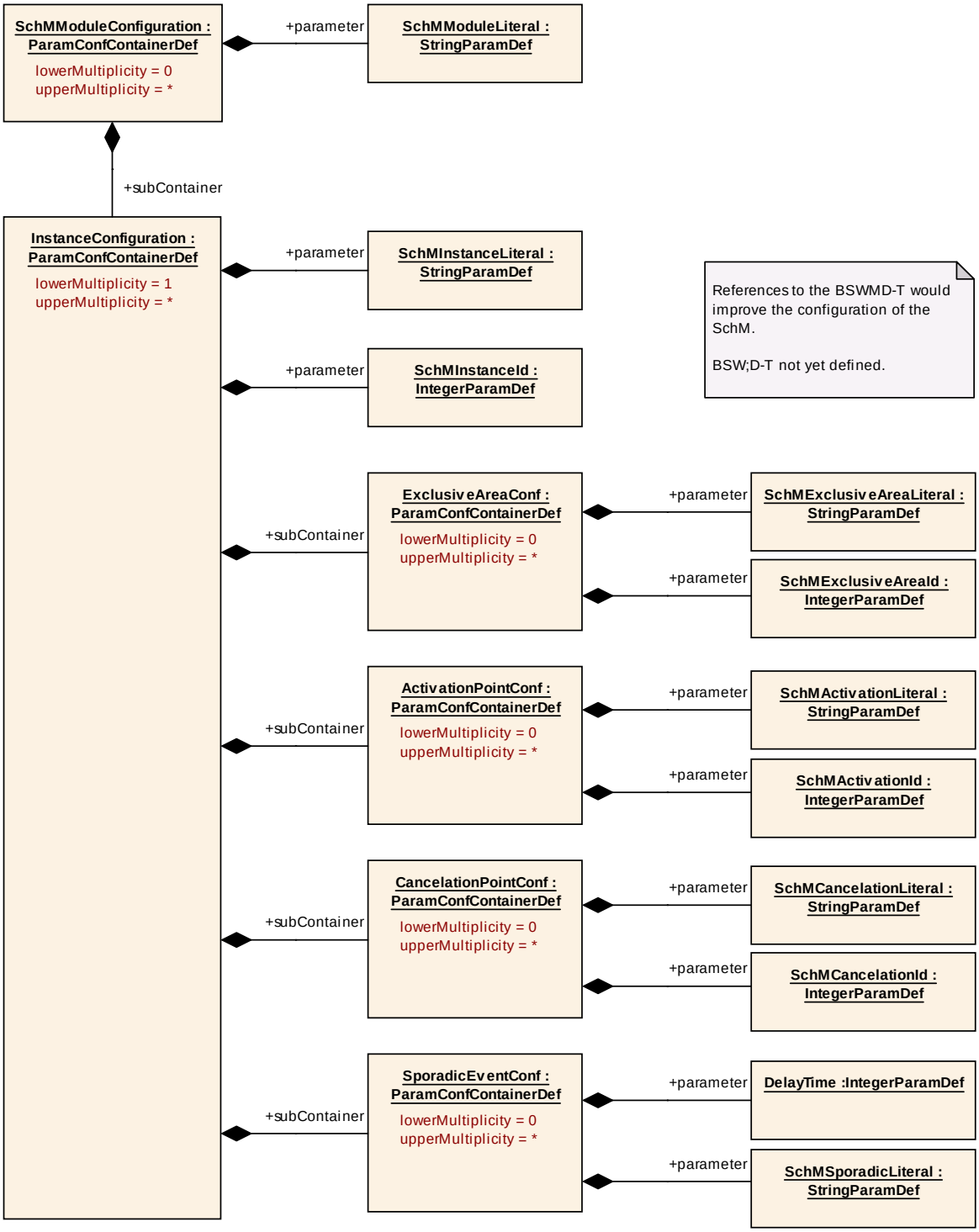


Figure 8.3: SchM Module and Instance configuration

SchMModuleConfiguration

SWS Item	[INTEGR087]
Container Name	SchMModuleConfiguration {ModuleConfiguration}
Description	Contains module specific configuration items.
Configuration Parameters	

Name last modified in: V1.1.0	SchMModuleLiteral {SCHM_MODULE_LITERAL}		
Description	The parameter depicts a symbolic name (a literal) for the unambigous identification of a BSW module; in fact it is the <ModulePrefix> specified for each BSW module in the Basic Software Module List.		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
InstanceConfiguration	1..*		

8.2 MCU Driver

The Microcontroller Unit Driver (MCU Driver) is specified in [25]. The MCU driver provides services for basic microcontroller initialization, power down functionality, reset and microcontroller specific functions required from other MCAL software modules. The general configuration parameters are collected in the container `McuGeneralConfiguration`, whereas the module configuration is summerized in the container `McuModuleConfiguration` which is also the base of a multiple configuration set. The structure of the module's configuration and it's including subcontainers is shown in figure 8.4.

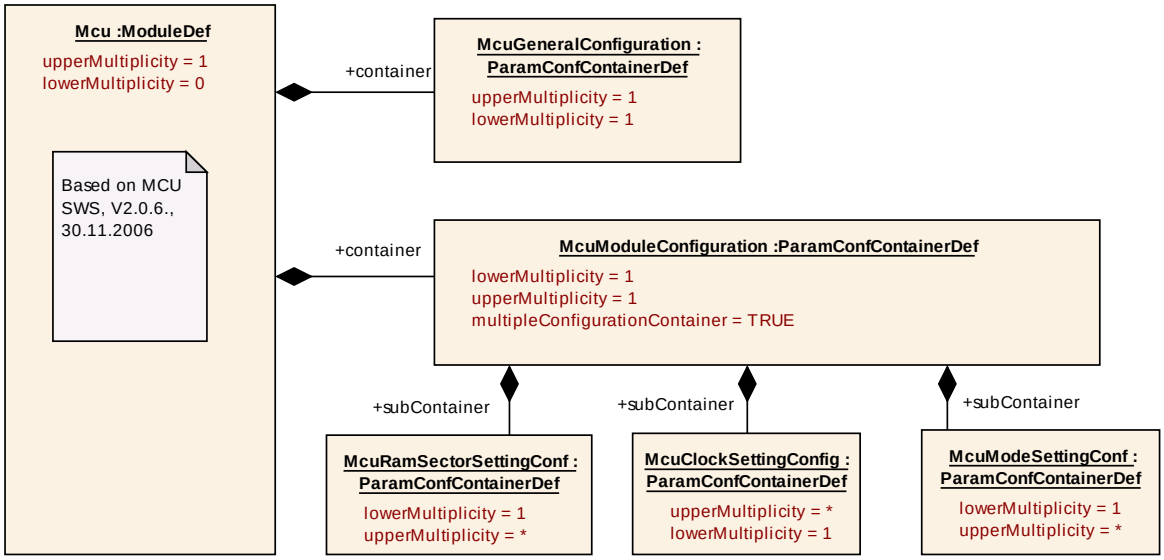


Figure 8.4: Overview MCU module

Module Name	Mcu	
Module Description	Configuration of the Mcu (Microcontroler Unit) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
McuGeneralConfiguration	1	This container contains the configuration (parameters) of the MCU driver.
McuModuleConfiguration	1	This container contains the configuration (parameters) of the MCU driver

The general, pre-compile parameters are collected in the container `McuGeneralConfiguration`, which is shown in figure 8.5.

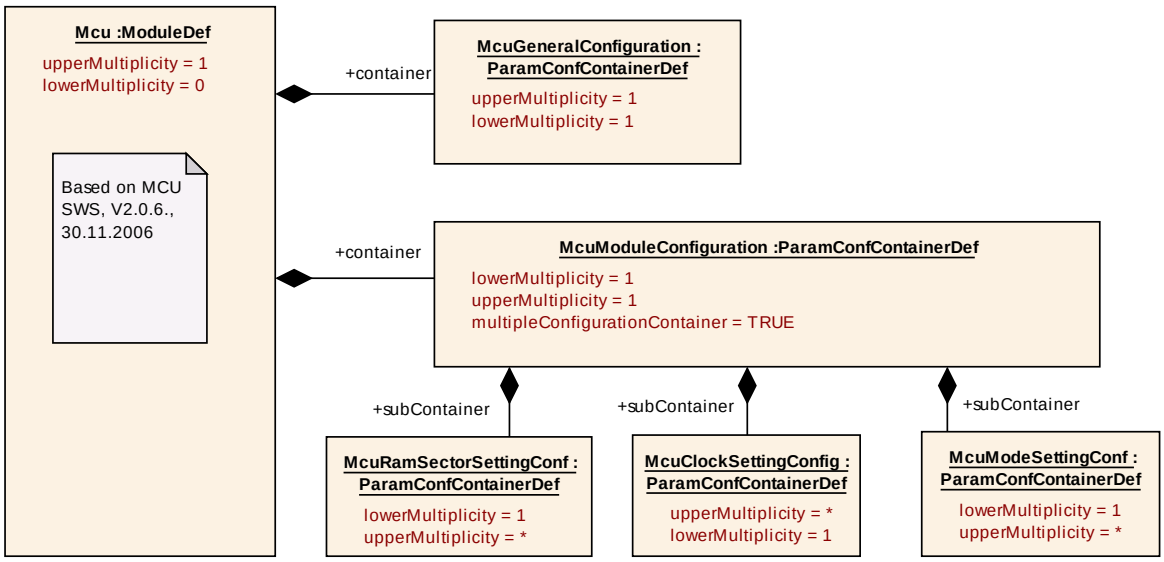


Figure 8.5: Parameters of the container McuGeneralConfiguration

McuGeneralConfiguration

SWS Item	[MCU118]		
Container Name	McuGeneralConfiguration {MCU General Configuration}		
Description	This container contains the configuration (parameters) of the MCU driver.		
Configuration Parameters			

Name last modified in: V1.0.0	McuDevErrorDetect {MCU_DEV_ERROR_DETECT}		
Description	Pre-processor switch for enabling the development error detection and reporting.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.1.0	McuPerformResetApi {MCU_PERFORM_RESET_API}		
Description	TRUE: Mcu_PerformReset() API is used FALSE: Mcu_PerformReset() API is not used		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	McuReportWakeupReason {MCU_REPORT_WAKE_UP_REASON}		
Description	Pre-processor switch for enabling Wake-up reason notification.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	McuVersionInfoApi {MCU_VERSION_INFO_API}		
Description	Pre-processor switch to enable / disable the API to read out the modules version information.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
No Included Containers			

The container `McuModuleConfiguration` contains module specific configuration parameters as well as further subcontainers (not included in the figure 8.6).

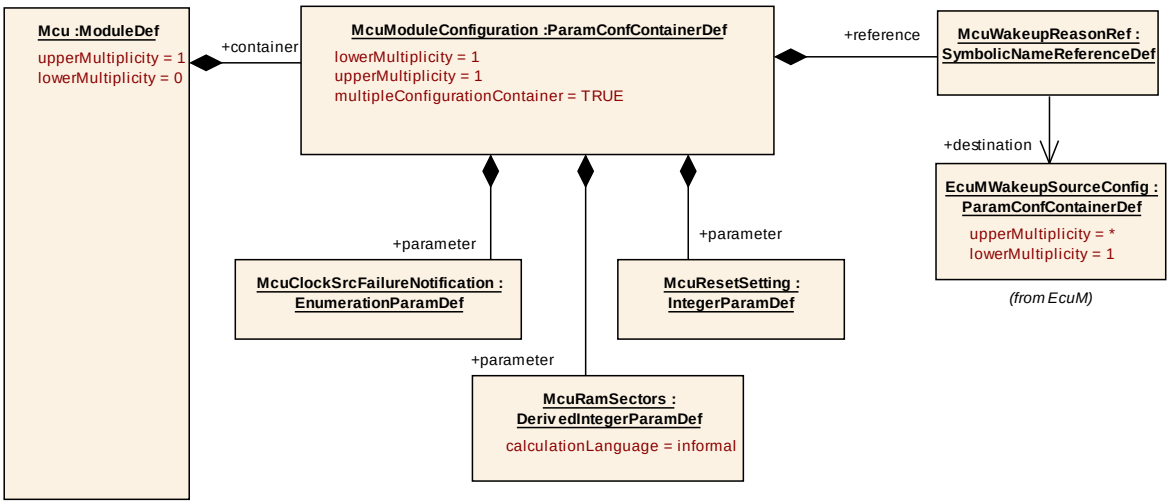


Figure 8.6: Overview of the MCU module configuration parameters

McuModuleConfiguration

SWS Item	[MCU119]		
Container Name	McuModuleConfiguration {MCU Module Configuration}		
Description	This container contains the configuration (parameters) of the MCU driver		
Configuration Parameters			

Name last modified in: V1.0.0	McuClockSrcFailureNotification {MCU_CLOCK_SOURCE_FAILURE_NOTIFICATION}		
Description	Enables/Disables clock failure notification		
Multiplicity	1		
Type	EnumerationParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	McuRamSectors {MCU_RAM_SECTORS}		
Description	This parameter shall represent the number of RAM sectors available for the MCU. calculationFormula = Number of configured McuRamSectorSettingConf		
Multiplicity	1		
Type	DerivedIntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	McuResetSetting {MCU_RESET_SETTING}		
Description	This parameter relates to the MCU specific reset configuration. This applies to the function Mcu_PerformReset, which performs a microcontroller reset using the hardware feature of the microcontroller.		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	McuWakeupReasonRef		
Description			
Multiplicity	1		
Type	Symbolic name reference to EcuMWakeupSourceConfig		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency			

Included Containers

Container Name	Multiplicity	Scope / Depedency
McuClockSettingConfig	1..*	
McuModeSettingConf	1..*	
McuRamSectorSetting Conf	1..*	

Figure 8.7 shows the subcontainers of `McuModuleConfiguration` and their including parameters. This includes `McuClockSettingConf`, `McuModeSettingConf` and `McuRamSectorSettingConf`. The container `McuClockSeetingConf` contains a subcontainer `McuClockReferencePoint` which holds a resulting frequency of the configured clock tree, that can then be used by other modules as a reference point. This enables the clock-tree depend BSWs to use this frequency in order to establish the right register settings for the required behaviour (e.g. baudrate setting).

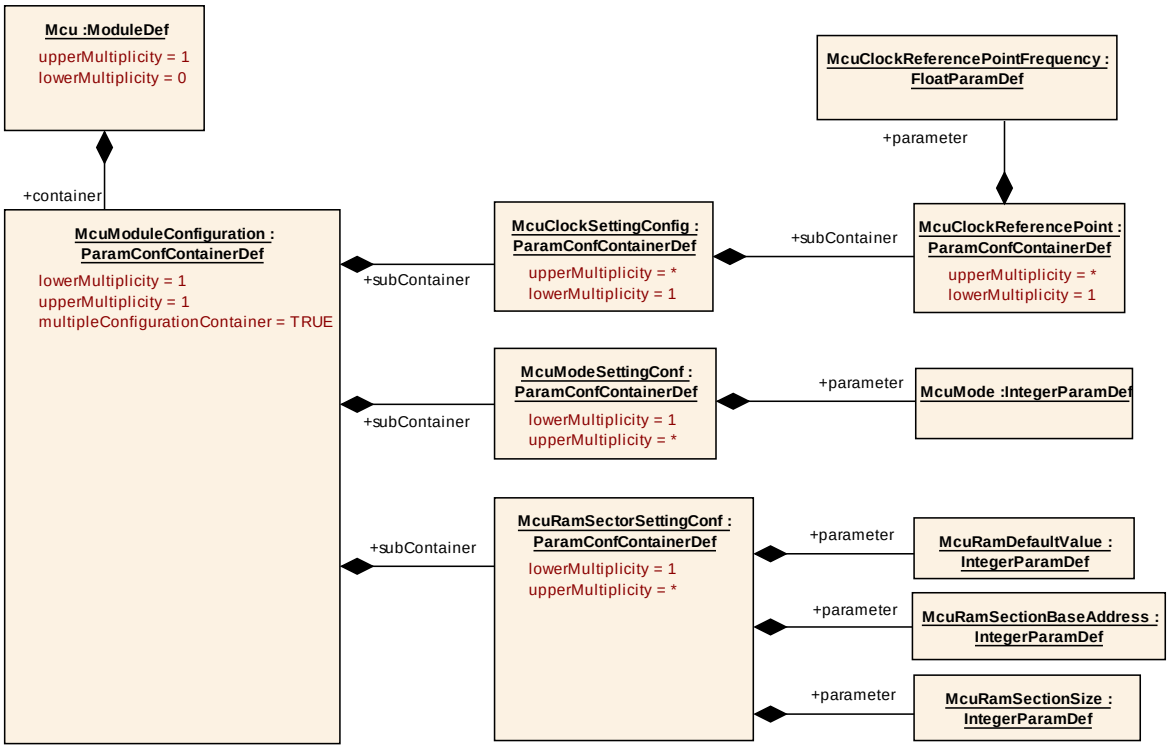


Figure 8.7: Overview of the subcontainers McuRamSectorSettingConf, McuClockSettingConfig and MCUModeSettingConf

McuClockSettingConfig

SWS Item		
Container Name	McuClockSettingConfig {MCU Clock Setting Configuration}	
Description	This container contains the configuration (parameters) for the Clock settings of the MCU. Please see MCU031 for more information on the MCU clock settings.	
Configuration Parameters Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
McuClockReferencePoint	1..*	

McuModeSettingConf

SWS Item	[MCU123]		
Container Name	McuModeSettingConf {MCU Mode Setting Configuration}		
Description			
Configuration Parameters			

Name last modified in: V1.1.0	McuMode {MCU_MODE}		
Description	The parameter is to represent the different MCU Modes available to an MCU		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module		
No Included Containers			

McuRamSectorSettingConf

SWS Item	[MCU120]		
Container Name	McuRamSectorSettingConf {MCU RAM Sector Setting Configuration}		
Description			
Configuration Parameters			

Name last modified in: V1.0.0	McuRamDefaultValue {MCU_RAM_DEFAULT_VALUE}		
Description	This parameter shall represent the Data pre-setting to be initialized		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	McuRamSectionBaseAddress {MCU_RAM_SECTION_BASE_ADDRESS}		
Description	This parameter shall represent the MCU RAM section base address		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	McuRamSectionSize {MCU_RAM_SECTION_SIZE}		
Description	This parameter shall represent the MCU RAM Section size		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Module		
No Included Containers			

8.3 GPT Driver

The General Purpose Timer Driver (GPT Driver) is specified in [19] and only uses the hardware timer channels of the general-purpose timer unit and thus provides exact and short-term timings for use in the Operating system or within other basic software modules where an OS Alarm service has too much overhead.

The GPT Driver module is structured in the container `GptDriverConfiguration` as shown in figure 8.8.

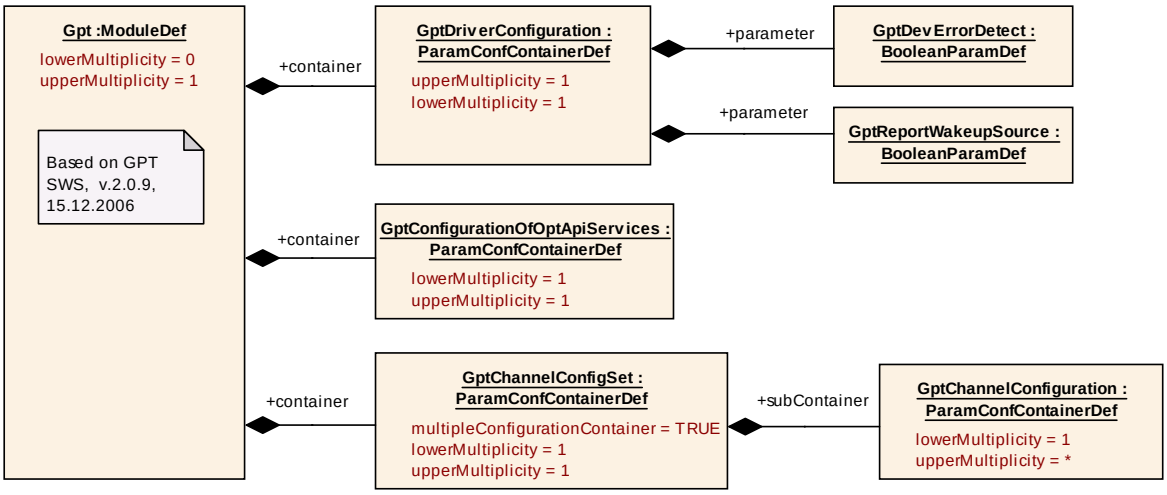


Figure 8.8: Parameter definition for the GPT module

Module Name	Gpt	
Module Description	Configuration of the Gpt (General Purpose Timer) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
GptChannelConfigSet	1	This container is the base of an Configuration Set which contains the configured GPT channels. This way, different configuration sets can be defined for post-build process.
GptConfigurationOfOptApi Services	1	This container contains all configuration switches for configuring optional API services of the GPT driver
GptDriverConfiguration	1	This container contains the module-wide configuration (parameters) of the GPT Driver

GptDriverConfiguration

SWS Item	[GPT183]		
Container Name	GptDriverConfiguration		
Description	This container contains the module-wide configuration (parameters) of the GPT Driver		
Configuration Parameters			

Name last modified in: V1.0.0	GptDevErrorDetect {GPT_DEV_ERROR_DETECT}		
Description	Enables/Disables development error detection		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	GptReportWakeupSource {GPT_REPORT_WAKEUP_SOURCE}		
Description	Enables/Disables wakeup source reporting		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
No Included Containers			

The channel specific configuration parameters are located in the subcontainer `GptChannelConfiguration` which contains an additional subcontainer `GptWakeupConfiguration` as shown in figure 8.9

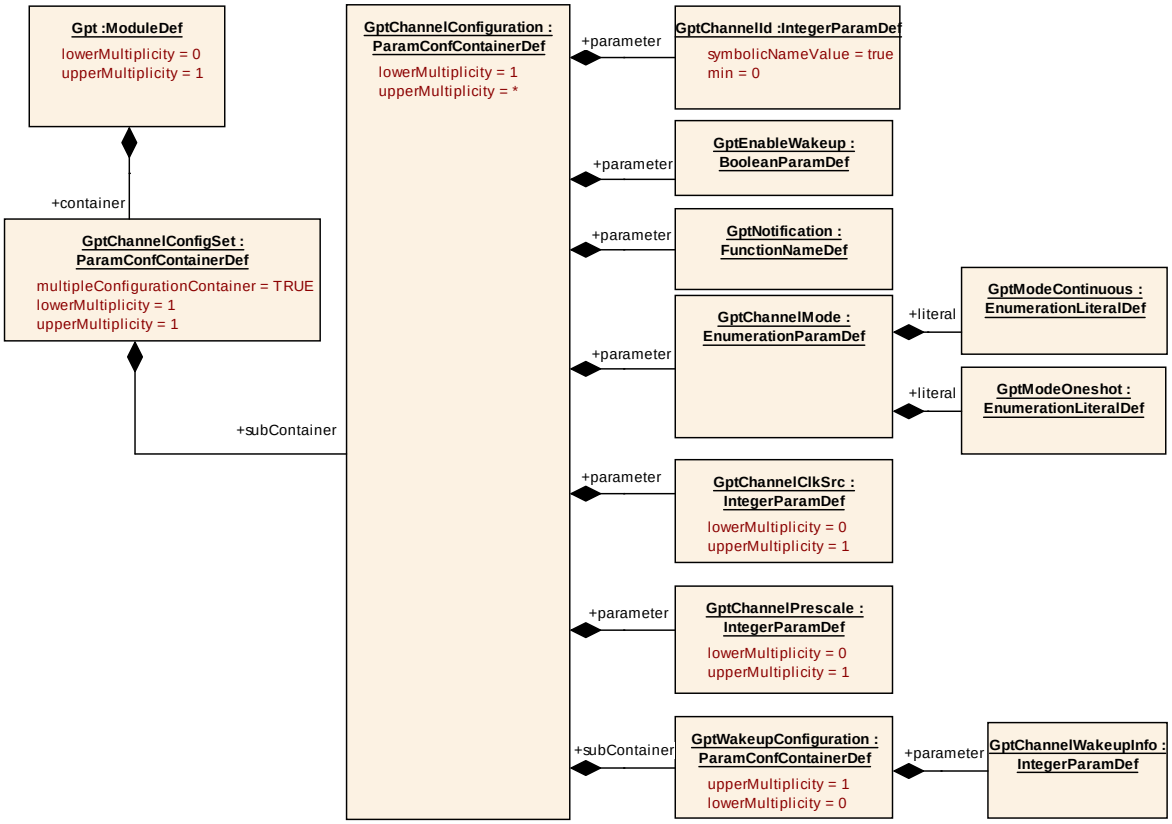


Figure 8.9: Parameter definition for the GPT channel configuration

GptChannelConfiguration

SWS Item	[GPT184]		
Container Name	GptChannelConfiguration		
Description	This container contains the channel-wide configuration (parameters) of the GPT Driver		
Configuration Parameters			

Name last modified in: V1.0.0	GptChannelClkSrc {GPT_CHANNEL_CLKSRC}		
Description	The GPT module specific clock input for the timer unit can statically be configured and allows to select different clock sources (external clock, internal GPT specific clock) per channel		
Multiplicity	0..1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Instance		

Name last modified in: V1.0.0	GptChannelId		
Description	Channel Id of the GPT channel. This value will be assigned to the symbolic name derived of the GptChannelConfiguration container short name.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	GptChannelMode {GPT_CHANNEL_MODE}		
Description	Specifies the behaviour of the timerchannel after the timeout has expired		
Multiplicity	1		
Type	EnumerationParamDef		
Range	GptModeContinuous	Timerchannel is restarted automatically after reaching its end value	
	GptModeOneshot	Timerchannel stops after reaching its end value	
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Instance		

Name last modified in: V1.0.0	GptChannelPrescale {GPT_CHANNEL_PRESCALE}		
Description	GPT module specific prescaler factor per channel		
Multiplicity	0..1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Instance		

Name last modified in: V1.0.0	GptEnableWakeup {GPT_ENABLE_WAKEUP}		
Description	Enables wakeup capability of CPU for a channel when timeout period expires. This might be different to enabling the notification depending on hardware capabilities		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Instance		

Name last modified in: V1.1.0	GptNotification {Gpt_Notification}		
Description	Function pointer to callback function		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: Instance		
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
GptWakeupConfiguration	0..1		

GptWakeupConfiguration

SWS Item	[GPT235]		
Container Name	GptWakeupConfiguration		
Description			
Configuration Parameters			

Name last modified in: V1.0.0	GptChannelWakeupInfo {WakeupInfo}		
Description	In case the wakeup-capability is true this value is transmitted to the Ecu State Manager		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	–	
	Post-build time	M	PB
Scope / Dependency	scope: ECU		
No Included Containers			

The configuration of the optional APIs is located in the subcontainer `GptConfigurationOfOptApiServices` as shown in figure 8.10.

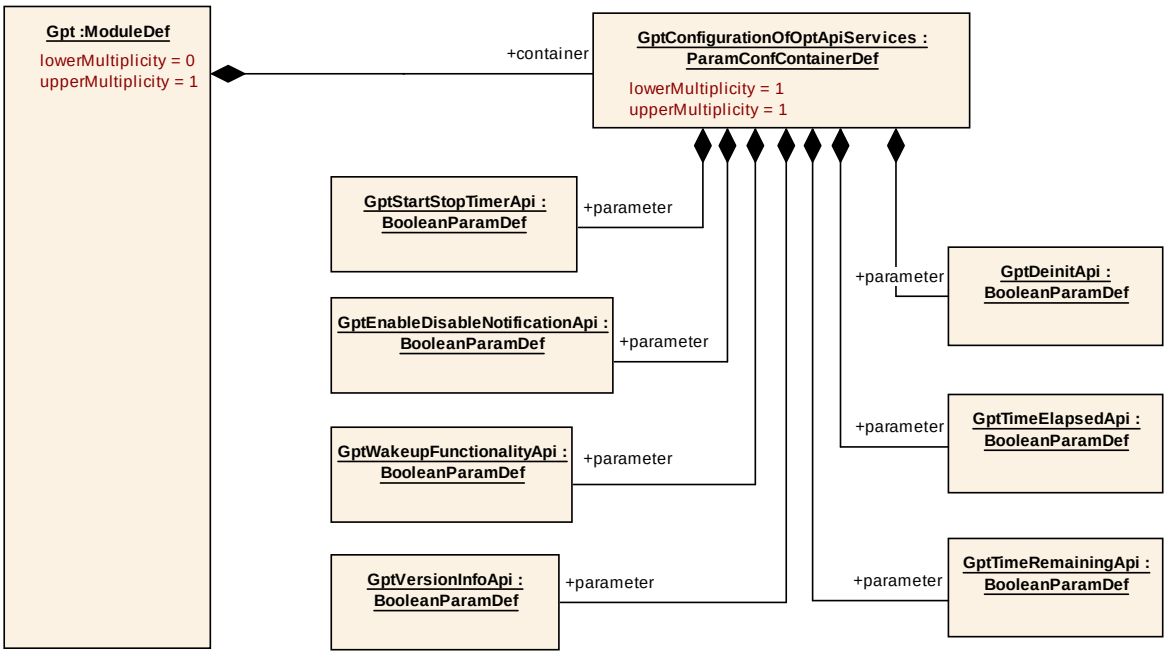


Figure 8.10: Parameter definition for configuration of optional API Services

GptConfigurationOfOptApiServices

SWS Item	[GPT193]		
Container Name	GptConfigurationOfOptApiServices {Configuration of optional API services}		
Description	This container contains all configuration switches for configuring optional API services of the GPT driver		
Configuration Parameters			

Name last modified in: V1.0.0	GptDeinitApi {GPT_DEINIT_API}		
Description	Adds / removes the service Gpt_DeInit() from the code.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	GptEnableDisableNotificationApi {GPT_ENABLE_DISABLE_NOTIFICATION_API}		
Description	Adds / removes the services Gpt_EnableNotification() and Gpt_DisableNotification from the code.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	GptStartStopTimerApi {GPT_START_STOP_TIMER_API}		
Description	Adds / removes the services Gpt_StartTimer() and Gpt_StopTimer() from the code.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	GptTimeElapsedApi {GPT_TIME_ELAPSED_API}		
Description	Adds / removes the service Gpt_GetTimeElapsed() from the code		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	GptTimeRemainingApi {GPT_TIME_REMAINING_API}		
Description	Adds / removes the service Gpt_GetTimeRemaining() from the code.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	GptVersionInfoApi {GPT_VERSION_INFO_API}		
Description	Adds / removes the service Gpt_GetVersionInfo() from the code.		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.1.0	GptWakeupFunctionalityApi {GPT_WAKEUP_FUNCTIONALITY_API}		
Description	Adds / removes the services Gpt_SetMode(), Gpt_EnableWakeup() and Gpt_DisableWakeup() from the code		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		
No Included Containers			

8.4 RAM Test

The Ram Test driver is specified in [37]. The configuration model for the RamTst module is structured into several containers. Figure 8.11 shows the module overview.

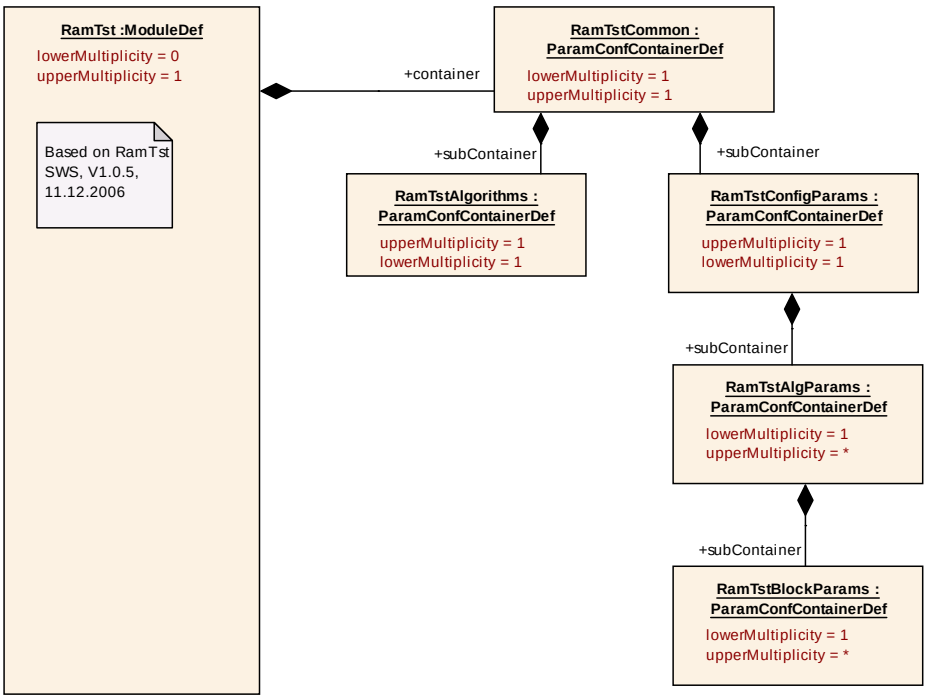


Figure 8.11: Module overview

Module Name	RamTst	
Module Description	Configuration of the RamTst module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CommonPublished Information	1	Common container, aggregated by all modules. It contains published information about vendor and versions.
RamTstCommon	1	This container contains all configuration parameter for the RamTest selectable at pre-compile time.

The container `RamTstCommon` which contains the common configuration parameters of the module is shown in figure 8.12.

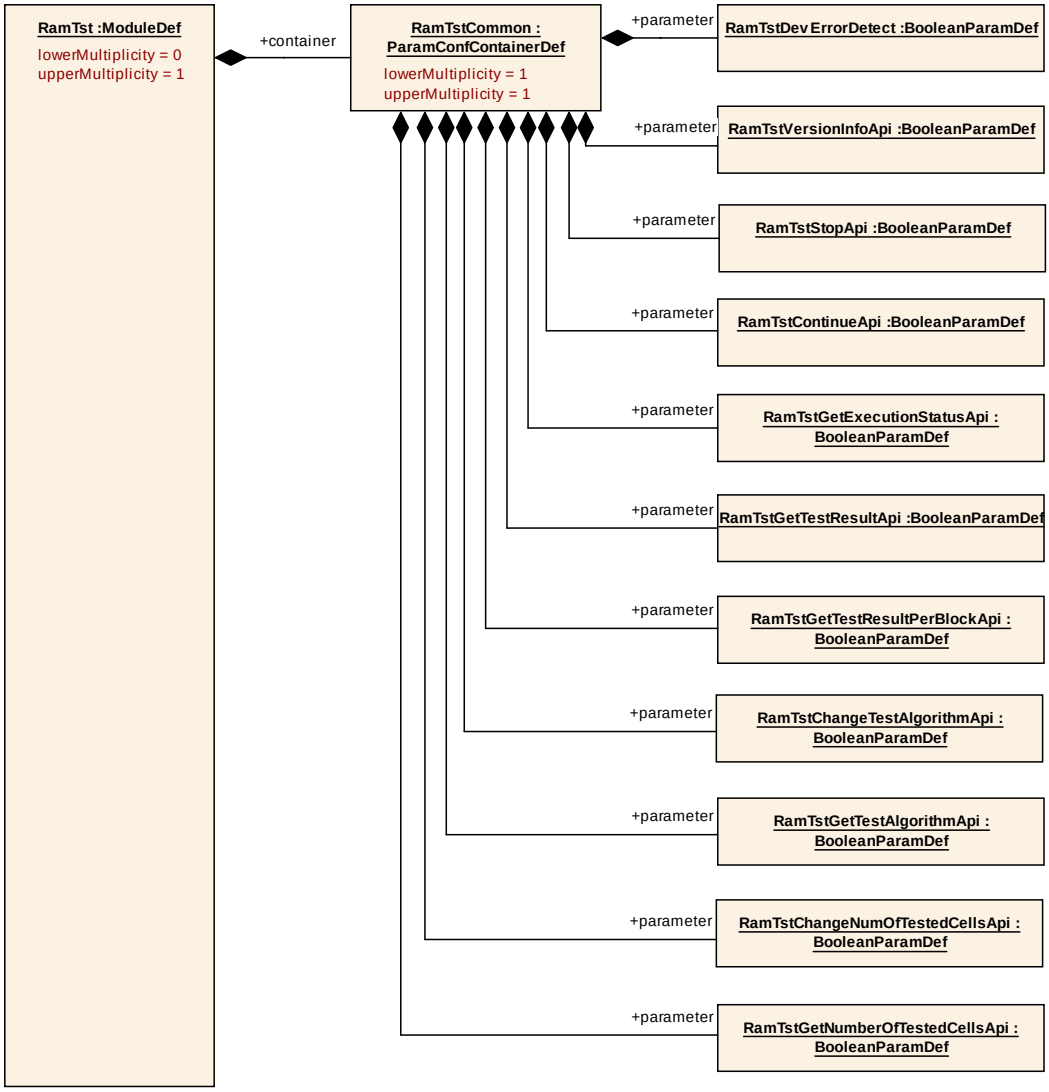


Figure 8.12: Parameter description for the container RamTstcommon

RamTstCommon

SWS Item	[RamTst070]		
Container Name	RamTstCommon {RamTst_Common}		
Description	This container contains all configuration parameter for the RamTest selectable at pre-compile time.		
Configuration Parameters			

Name last modified in: V1.0.0	RamTstChangeNumOfTestedCellsApi {RAMTST_CHANGE_NUMBER_OF_TESTED_CELLS_API}		
Description	Preprocessor switch to disable / enable the function "RamTst_ChangeNumberOfTestedCells"		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	RamTstChangeTestAlgorithmApi {RAMTST_CHANGE_TEST_ALGORITHM_API}		
Description	Preprocessor switch to disable / enable the function "RamTst_ChangeTestAlgorithm"		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	RamTstContinueApi {RAMTST_CONTINUE_API}		
Description	Preprocessor switch to disable / enable the function "RamTst_Continue"		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	RamTstDevErrorDetect {RAMTST_DEV_ERROR_DETECT}		
Description	Preprocessor switch to select the development error tracer (DET) ON or OFF		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	RamTstGetExecutionStatusApi {RAMTST_GET_EXECUTION_STATUS_API}		
Description	Preprocessor switch to disable / enable the function "RamTst_GetExecutionStatus"		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	RamTstGetNumberOfTestedCellsApi {RAMTST_GET_NUMBER_OF_TESTED_CELLS_API}		
Description	Preprocessor switch to disable / enable the function "RamTst_GetNumberOfTestedCells"		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	RamTstGetTestAlgorithmApi {RAMTST_GET_TEST_ALGORITHM_API}		
Description	Preprocessor switch to disable / enable the function "RamTst_GetTestAlgorithm"		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	RamTstGetTestResultApi		
Description	Preprocessor switch to disable / enable the function "RamTst_GetTestResult"		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	RamTstGetTestResultPerBlockApi {RAMTST_GET_TEST_RESULT_PER_BLOCK_API}		
Description	Preprocessor switch to disable / enable the function "RamTst_GetTestResultPerBlock"		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	RamTstStopApi {RAMTST_STOP_API}		
Description	Preprocessor switch to disable / enable the function "RamTst_Stop"		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	RamTstVersionInfoApi {RAMTST_VERSION_INFO_API}		
Description	Preprocessor switch to disable / enable the function "RamTst_GetVersionInfo"		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
RamTstAlgorithms	1	
RamTstConfigParams	1	

The configuration regarding the included Algorithms of the driver is content of the sub-container `RamTstAlgorithms` and is shown in figure 8.13.

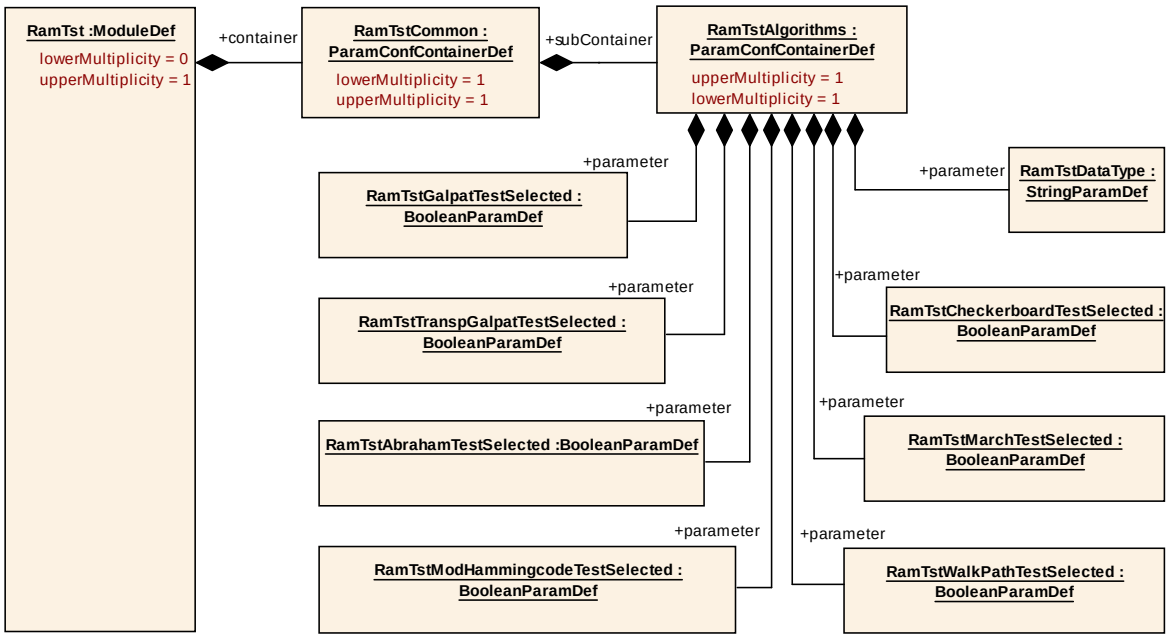


Figure 8.13: Parameter description for the RamTst Algorithms

RamTstAlgorithms

SWS Item	[RamTst065]		
Container Name	RamTstAlgorithms {RamTst_Algorithms}		
Description	A subset of supported RAM Test algorithms could be chosen in this container		
Configuration Parameters			

Name last modified in: V1.0.0	RamTstAbrahamTestSelected {RAMTST_ABRAHAM_TEST_SELECTED}		
Description	Preprocessor switch to select the abraham test ON or OFF		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	RamTstCheckerboardTestSelected {RAMTST_CHECKERBOARD_TEST_SELECTED}		
Description	Preprocessor switch to select the checkerboard test ON or OFF		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	RamTstDataType {RAM_TEST_DATA_TYPE}		
Description	Data type of tested RAM cells		
Multiplicity	1		
Type	StringParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name last modified in: V1.0.0	RamTstGalpatTestSelected {RAMTST_GALPAT_TEST_SELECTED}		
Description	Preprocessor switch to select the galpat test ON or OFF		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	RamTstMarchTestSelected {RAMTST_MARCH_TEST_SELECTED}		
Description	Preprocessor switch to select the march test ON or OFF		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	RamTstModHammingcodeTestSelected {RAMTST_MOD_HAMMINGCODE_TEST_SELECTED}		
Description	Preprocessor switch to select the modified hamming code test ON or OFF		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	RamTstTranspGalpatTestSelected {RAMTST_TRANSP_GALPAT_TEST_SELECTED}		
Description	Preprocessor switch to select the transparent galpat test ON or OFF		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	RamTstWalkPathTestSelected {RAMTST_WALK_PATH_TEST_SELECTED}		
Description	Preprocessor switch to select the walking path test ON or OFF		
Multiplicity	1		
Type	BooleanParamDef		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency **No Included Containers**	scope: Module		

The subcontainer `RamTstConfigParams` contains parameters and further subcontainers `RamTstAlgParams` and `RamTstBlockParams` to describe the specific algorithms and settings such as which algorithm is used as a default. The structure of these subcontainers and their parameters is shown in the figure 8.14.

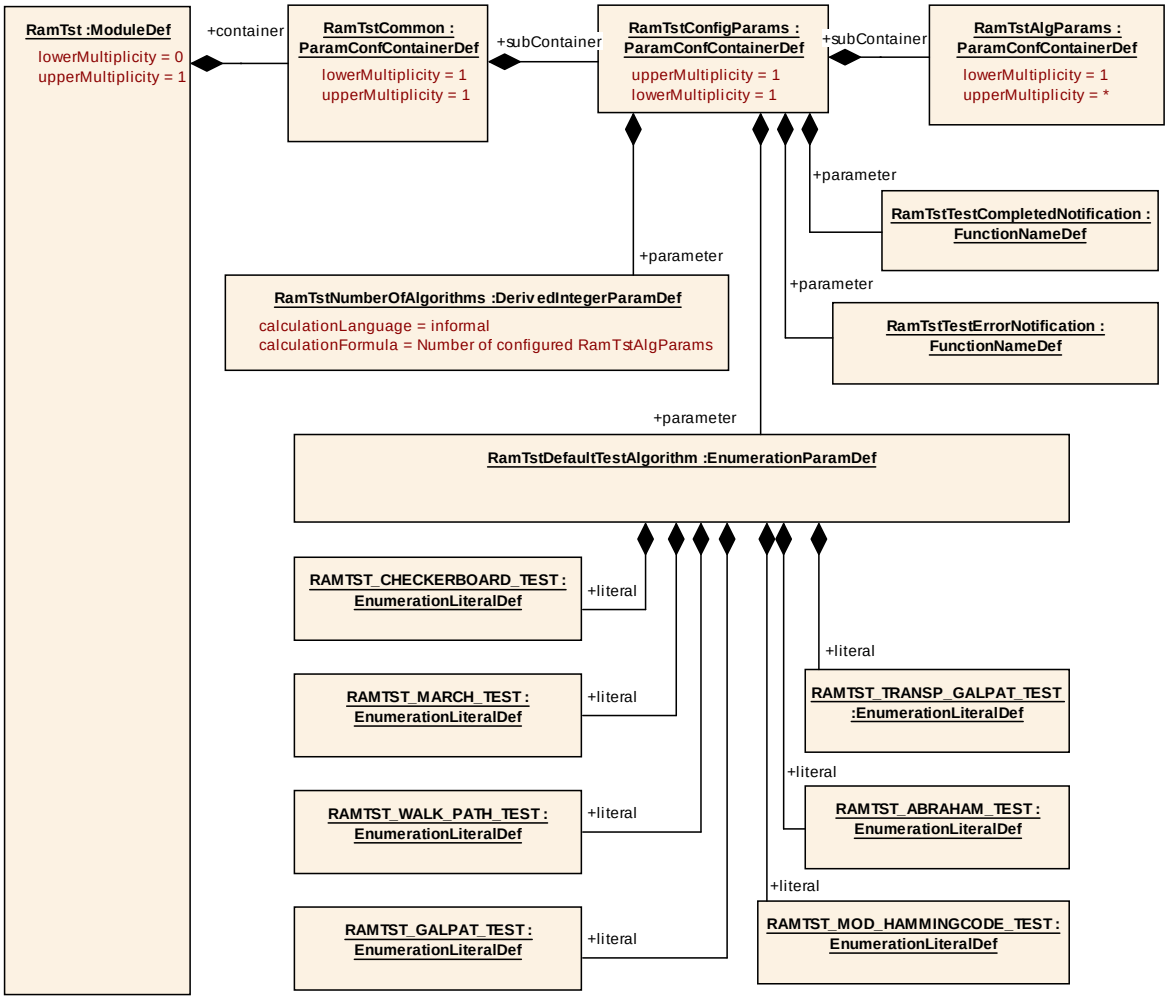


Figure 8.14: Parameter definition for the RamTst Configuration Parameters

RamTstConfigParams

SWS Item	[RamTst066]		
Container Name	RamTstConfigParams {RamTst_ConfigParams}		
Description	The multiplicity of the included container depends on the pre-compile configuration in the container ”RamTst_Algorithms”		
Configuration Parameters			
Name last modified in: V1.0.0	RamTstDefaultTestAlgorithm {RAMTST_DEFAULT_TEST_ALGORITHM}		
Description	This is the default algorithm after the ”RamTst_Init(..)” function. This is the initial value for a RAM variable which could be changed by the function ”RamTst_ChangeTestAlgorithm”		
Multiplicity	1		
Type	EnumerationParamDef		
Range	RAMTST_ABRAHAM_TEST		
	RAMTST_CHECKERBOARD_TEST		
	RAMTST_GALPAT_TEST		
	RAMTST_MARCH_TEST		
	RAMTST_MOD_HAMMINGCODE_TEST		
	RAMTST_TRANSP_GALPAT_TEST		
	RAMTST_WALK_PATH_TEST		
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	–	
Scope / Dependency	scope: Module		
Name last modified in: V1.1.0	RamTstNumberOfAlgorithms {RAMTST_NUMBER_OF_ALGORITHMS}		
Description	Number of configured algorithms in the Container ”RamTst_Algorithms”		
Multiplicity	1		
Type	DerivedIntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	–	
Scope / Dependency	scope: Module dependency: ”RamTstNumberOfAlgorithms” is derived by the count of ”RamTstAlgParams” which is part of the same subContainer and has a multiplicity of 1 to *.		
Name last modified in: V1.0.0	RamTstTestCompletedNotification {RAMTST_TEST_COMPOLETED_NOTIFICATION}		
Description	This function will be called after finishing the ram test without an error		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	RamTstTestErrorNotification {RAMTST_TEST_ERROR_NOTIFICATION}		
Description	This function will be called if an error occurs during the ram test		
Multiplicity	1		
Type	FunctionNameDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	–	
Scope / Dependency	scope: Module		
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
RamTstAlgParams	1..*		

RamTstAlgParams

SWS Item	[RamTst090]		
Container Name	RamTstAlgParams {RamTst_AlgParams}		
Description	This is the parameter container for an algorithm.		
Configuration Parameters			

Name last modified in: V1.0.0	RamTstAlgorithmId {RAMTST_ALGORITHM_ID}		
Description	This is the algorithm ID		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	–	
Scope / Dependency	scope: Module		

Name last modified in: V1.0.0	RamTstNumberOfBlocks {RAMTST_NUMBER_OF_BLOCKS}		
Description	Number of ram blocks configured in the container ”RamTst_BlockParams” calculationFormula = Count of enabled RamTstBlockParams		
Multiplicity	1		
Type	DerivedIntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	–	
Scope / Dependency	scope: Module dependency: ”RamTstNumberOfBlocks” is derived by the count of ”RamTstBlockParams” which is part of the same subContainer and has a multiplicity of 1 to *.		

Name last modified in: V1.0.0	RamTstNumberOfTestedCells {RAMTST_NUMBER_OF_TESTED_CELLS}		
Description	This is the initial value for a RAM variable which could be changed by the function ”RamTst_ChangeNumberOfTestedCells”		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	–	
Scope / Dependency	scope: Module		
Included Containers			
Container Name	**Multiplicity**	**Scope / Depedency**	
RamTstBlockParams	1..*		

RamTstBlockParams

SWS Item	[RamTst091]		
Container Name	RamTstBlockParams {RamTst_BlockParams}		
Description	This container is used in the ” RamTst_AlgParams” container		
Configuration Parameters			
Name last modified in: V1.0.0	RamTstBlockId {RAMTST_BLOCK_ID}		
Description	ID of the RAM block		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	–	
Scope / Dependency	scope: Module		
Name last modified in: V1.0.0	RamTstEndAddress {RAMTST_END_ADDRESS}		
Description	End Address of the RAM block		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	–	
Scope / Dependency	scope: Module		
Name last modified in: V1.0.0	RamTstStartAddress {RAMTST_START_ADDRESS}		
Description	Start Address of the RAM block		
Multiplicity	1		
Type	IntegerParamDef		
Configuration Class	**Pre-compile time**	X	PC
	Link time	X	LT
	Post-build time	–	
Scope / Dependency	scope: Module		
No Included Containers			

A Parameter Change History

This section of the document contains a detailed list of all the changes performed compared to the previous released version of this document.

A.1 Release 1.1.0

Module	Parameter/Container	Chapter	RfC Id	Description
Adc	AdcPublishedInformation	6.4	12390	Moved ”AdcPublishedInformation” to PublishedInformation configuration class
Can	CanVendorId	4.5.2	?????	todo: added ”CanVendorId”
CanIf	CanIfDriverRef	4.5.1	?????	todo: added ”CanIfDriverRef”
CanIf	CanIfDriverVendorId	4.5.1	?????	todo: removed ”CanIfDriverVendorId”
CanIf	CanIfDriverRef	4.5.1	?????	todo: added parameter ”CanIfDriverRef”
CanTp	CanTp	4.5.3	15488	Configuration classes for references
CanTrcv	CanTrcv	4.5.4	16745	Missing Configuration Parameters
CanNm	CanNm	4.8.3	16228	Updated to R2.1
CNm	CNm	4.8.2	n/a	New in this release
Nm	Nm	4.8.1	n/a	Nm Interface is new in this release
FrNm	FrNm	4.8.4	n/a	FrNm is new in this release
Com	ComSignalType	4.2.2	15521	Changed variants 2 and 3 from `postBuild` to `link`
ComM	ComM	5.5	15683	R2.1 Adoption
EcuM	EcuMNvramWriteallTimeout			todo: added ”EcuMNvramWriteallTimeout”
EcuM	EcuMTTIIWakupSource			todo: added ”EcuMTTIIWakupSource”
EcuM	EcuMSleepModeId			todo: added ”EcuMSleepModeId”
EcuM	EcuMWakupSourceMask			todo: added ”EcuMWakupSourceMask”
EcuM	EcuMSleepModeConfiguration			todo: removed ”EcuMSleep-ModeConfiguration”
Eep	EepPublishedInformation	7.4.2	12390	Moved ”EepPublishedInformation” to PublishedInformation configuration class
Fls	FlsPublishedInformation	7.4.1	12390	Moved ”FlsPublishedInformation” to PublishedInformation configuration class
FrTp	FrTp	4.6.3	15295	R2.1 Adoption
FrTp	FrTp	4.6.3	17689	Multiple Configuration
FrTrcv	FrTrcv	4.6.4	15490	R2.1 Adoption

Table A.1: Release 1.1.0 changes

Module	Parameter/Container	Chapter	RfC Id	Description
Gpt	GptNotification	8.3	13706	Changed configuration class from ”Ecuc.variant.preCompile:*” to ”Ecuc.variant.postBuild:2 - Ecuc.variant.preCompile:1”
Gpt	GptWakeupConfiguration	8.3	13509	Changed UPPER-MULTIPLICITY from ”*” to ”1”
Gpt	GptWakeupFunctionalityApi	8.3	13509	Typo correction. Changed ”GptWakeupFuntionalityApi” to ”GptWakeupFunctionalityApi”
LinTp	LinTpRxNSduPduRef			todo: added ”LinTpRxNSduPduRef”
Mcu	McuMode	8.2	12474	Description changed as previous description was misleading
Mcu	McuPerformResetApi	8.2	13705	Added ”McuPerformResetApi”
Mcu	McuClockReferencePoint	8.2		Added ”McuClockReferencePoint” and ”McuClockReference-PointFrequency”
Os	OsEventMask	5.1.9	n/a	Changed LOWER-MULTIPLICITY from ”1” to ”0”
Os	OsScheduleTblSyncStrategy	5.1.3.2	n/a	Added defaultValue = ”HARD”
Port	PortPinMode	6.2	n/a	Changed UPPER-MULTIPLICITY from ”*” to ”1”
RamTst	RamTstNumberOfAlgorithms	8.4	13978	Changed ”INTEGER-PARAM-DEF” to ”DERIVED-INTEGER-PARAM-DEF” and included dependency description
Rte	Rte	3	14866	General rework of the Rte configuration
Wdg	WdgDefaultMode	5.2.3.1	15807	Moved ”WdgDefaultMode” to container WdgModeConfig to support multiple configuration sets
Wdg	WdgIndex	5.2.3	n/a	Added a missing configuration class *pre-compile* to the reference
Wdg	WdgModeConfig	5.2.3.1	15807	Set attribute *multipleConfigu-rationContainer* to *true* to support multiple configuration sets
Wdg	WdgPublishedInformation	5.2.3.2	12390	Moved ”WdgPublishedInformation” to PublishedInformation configuration class
WdgIf	WdgRef	5.2.2	n/a	Added a missing configuration class *pre-compile time* to the parameter

Table A.1: Release 1.1.0 changes

Module	Parameter/Container	Chapter	RfC Id	Description
WdgM	WdgIfRef	5.2.1.1	n/a	Added a missing configuration class *pre-compile time* to the parameter
WdgM	WdgMConfigEntities	5.2.1.1	15807	Introduced new container to support multiple configuration sets
WdgM	WdgMMode	5.2.1.1	15807	Moved parameter to newly introduced container WdgMConfigEntities due to multiple configuration set support
WdgM	WdgMNumberOfSupervised-Entities	5.2.1.2	16754	Fixed an inconsistency in configuration class between SWS and ECUC

Table A.1: Release 1.1.0 changes