

<b>Document Title</b>	Specification of FlexRay Interface
<b>Document Owner</b>	AUTOSAR
<b>Document Responsibility</b>	AUTOSAR
<b>Document Identification No</b>	027

<b>Document Status</b>	Final
<b>Part of AUTOSAR Standard</b>	Classic Platform
<b>Part of Standard Release</b>	4.4.0

<b>Document Change History</b>			
<b>Date</b>	<b>Release</b>	<b>Changed by</b>	<b>Change Description</b>
2018-10-31	4.4.0	AUTOSAR Release Management	<ul style="list-style-type: none"> <li>Added bus mirroring support</li> <li>Changed behavior for TxConflict</li> <li>Minor corrections</li> </ul>
2017-12-08	4.3.1	AUTOSAR Release Management	<ul style="list-style-type: none"> <li>Runtime error rollout</li> <li>UL_TxConfirmation replaced with UL_TriggerTransmit in affected requirements</li> </ul>
2016-11-30	4.3.0	AUTOSAR Release Management	<ul style="list-style-type: none"> <li>New feature to get the "TxConflictState"</li> <li>Introduce reliable TxConfirmation</li> <li>Unused bit handling reworked</li> <li>Several bug fixes</li> </ul>
2015-07-31	4.2.2	AUTOSAR Release Management	<ul style="list-style-type: none"> <li>Minor corrections</li> <li>Editorial changes</li> </ul>
2014-10-31	4.2.1	AUTOSAR Release Management	<ul style="list-style-type: none"> <li>Support for GlobalTimeSynchronization added</li> <li>Minor corrections</li> </ul>
2014-03-31	4.1.3	AUTOSAR Release Management	<ul style="list-style-type: none"> <li>Added Chapter for Production Errors</li> <li>Editorial Changes</li> </ul>
2013-10-31	4.1.2	AUTOSAR Release Management	<ul style="list-style-type: none"> <li>Minor corrections</li> <li>Editorial changes</li> <li>Removed chapter(s) on change documentation</li> </ul>
2013-03-15	4.1.1	AUTOSAR Administration	<ul style="list-style-type: none"> <li>Traceability requirements added</li> <li>Several Bug fixes</li> <li>Editorial Changes</li> </ul>
2011-12-22	4.0.3	AUTOSAR Administration	<ul style="list-style-type: none"> <li>Added User-defined communication operations</li> </ul>

<b>Document Change History</b>			
<b>Date</b>	<b>Release</b>	<b>Changed by</b>	<b>Change Description</b>
2010-09-30	3.1.5	AUTOSAR Administration	<ul style="list-style-type: none"> <li>• API “Frlf_GetCycleLength” added</li> <li>• API “Frlf_ReadCCConfig” added</li> <li>• APIs Frlf_EnableTransceiverWakeup / Frlf_DisableTransceiverWakeup removed</li> <li>• Configuration parameter “FrlfByteOrder” added</li> </ul>
2010-02-02	3.1.4	AUTOSAR Administration	<ul style="list-style-type: none"> <li>• Added support for FlexRay 3.0 hardware (CCs and transceivers)</li> <li>• Added functionalities to get detailed (error) information of the communications bus</li> <li>• Added support for single/key-slot mode</li> <li>• Added “cancel transmission” support</li> <li>• Legal disclaimer revised</li> </ul>
2008-08-13	3.1.1	AUTOSAR Administration	<ul style="list-style-type: none"> <li>• Legal disclaimer revised</li> </ul>
2008-02-01	3.0.2	AUTOSAR Administration	<ul style="list-style-type: none"> <li>• Correction of Figure 5.1</li> </ul>
2007-12-21	3.0.1	AUTOSAR Administration	<ul style="list-style-type: none"> <li>• Simplification of the FlexRay Interface State Machine due to the introduction of the new AUTOSAR SWS FlexRay State Manager</li> <li>• Cluster-based APIs were removed due to the introduced AUTOSAR SWS FlexRay State Manager</li> <li>• The FlexRay Interface does not initialize any other modules any more due to the introduction of the "flat initialization" for AUTOSAR release 3.0 Document meta information extended</li> <li>• Small layout adaptations made</li> </ul>
2007-01-24	2.1.15	AUTOSAR Administration	<ul style="list-style-type: none"> <li>• “Advice for users” revised</li> <li>• Legal disclaimer added</li> <li>• “Revision Information” added</li> <li>• Release Notes added</li> </ul>

<b>Document Change History</b>			
<b>Date</b>	<b>Release</b>	<b>Changed by</b>	<b>Change Description</b>
2006-05-16	2.0	AUTOSAR Administration	<ul style="list-style-type: none"><li>• Second Release</li></ul>
2005-05-31	1.0	AUTOSAR Administration	<ul style="list-style-type: none"><li>• Initial Release</li></ul>

## Disclaimer

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

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

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

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

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

## Table of Content

1	Introduction and Functional Overview .....	9
2	Information about this Document.....	10
2.1	General Hints .....	10
2.2	Acronyms and Abbreviations.....	11
3	Related Documentation .....	12
3.1	Input Documents .....	12
3.2	Related Standards and Norms .....	13
3.3	Related specification .....	13
4	Constraints and Assumptions.....	14
4.1	Limitations .....	14
4.2	Applicability to Car Domains .....	15
5	Dependencies to Other Modules .....	16
5.1	AUTOSAR Operating System .....	16
5.2	All Upper Layer AUTOSAR BSW Modules.....	16
5.3	AUTOSAR PDU-Router .....	16
5.4	AUTOSAR FlexRay Network Management.....	17
5.5	AUTOSAR FlexRay Transport Protocol .....	17
5.6	AUTOSAR Bus Mirroring.....	17
5.7	AUTOSAR FlexRay Driver .....	17
5.8	AUTOSAR FlexRay Transceiver Driver.....	17
5.9	File Structure.....	18
5.9.1	Header File Structure .....	18
6	Requirements Traceability .....	19
6.1	Specification Items .....	20
7	Functional Specification.....	22
7.1	FlexRay BSW Stack.....	22
7.2	Indexing Scheme.....	22
7.2.1	Principle .....	22
7.2.2	Supported Indexed Resources.....	26
7.3	FlexRay Interface State Machine .....	26
7.3.1	FlexRay Interface Main Function.....	28
7.4	Implementation Requirements .....	30
7.5	Configuration description.....	31
7.6	Data Communication via FlexRay .....	31
7.6.1	PDU Packing, PDU update bits, and Frame Construction Plans.....	32
7.6.2	Dynamic PDU length .....	34
7.6.3	AlwaysTransmit.....	34
7.6.4	Realization of the Time-Driven FlexRay Schedule .....	35
7.6.5	Communication Operations.....	37
7.6.6	Transmission with Immediate Buffer Access.....	45
7.7	Error Classification .....	46
7.7.1	Development Errors .....	46

7.7.2	Runtime Errors .....	46
7.7.3	Transient Faults .....	46
7.7.4	Production Errors .....	47
7.7.5	Extended Production Errors .....	50
8	API Service Specification .....	51
8.1	Imported types.....	51
8.2	Type Definitions.....	51
8.2.1	Frlf_ConfigType .....	51
8.2.2	Frlf_StateType .....	52
8.2.3	Frlf_StateTransitionType.....	52
8.3	Function Definitions.....	53
8.3.1	Frlf_Init.....	53
8.3.2	Frlf_ControllerInit .....	54
8.3.3	Frlf_SetAbsoluteTimer .....	54
8.3.4	Frlf_EnableAbsoluteTimerIRQ .....	56
8.3.5	Frlf_AckAbsoluteTimerIRQ .....	56
8.3.6	Frlf_StartCommunication .....	57
8.3.7	Frlf_HaltCommunication .....	58
8.3.8	Frlf_AbortCommunication .....	59
8.3.9	Frlf_GetState.....	60
8.3.10	Frlf_SetState .....	61
8.3.11	Frlf_SetWakeupChannel .....	62
8.3.12	Frlf_SendWUP .....	63
8.3.13	Frlf_GetPOCStatus .....	64
8.3.14	Frlf_GetGlobalTime.....	66
8.3.15	Frlf_AllowColdstart.....	67
8.3.16	Frlf_GetMacroticksPerCycle .....	67
8.3.17	Frlf_GetMacrotickDuration .....	68
8.3.18	Frlf_Transmit.....	69
8.3.19	Frlf_SetTransceiverMode.....	70
8.3.20	Frlf_GetTransceiverMode .....	71
8.3.21	Frlf_GetTransceiverWUReason .....	73
8.3.22	Frlf_ClearTransceiverWakeup .....	74
8.3.23	Frlf_CancelAbsoluteTimer.....	75
8.3.24	Frlf_GetAbsoluteTimerIRQStatus .....	77
8.3.25	Frlf_DisableAbsoluteTimerIRQ .....	78
8.3.26	Frlf_GetCycleLength .....	78
8.4	Optional Function Definitions .....	80
8.4.1	Frlf_AllSlots.....	80
8.4.2	Frlf_GetChannelStatus .....	81
8.4.3	Frlf_GetClockCorrection .....	81
8.4.4	Frlf_GetSyncFrameList.....	82
8.4.5	Frlf_GetNumOfStartupFrames .....	84
8.4.6	Frlf_GetWakeupRxStatus .....	85
8.4.7	Frlf_CancelTransmit.....	85
8.4.8	Frlf_DisableLPdu .....	87
8.4.9	Frlf_GetTransceiverError .....	87
8.4.10	Frlf_EnableTransceiverBranch.....	89
8.4.11	Frlf_DisableTransceiverBranch.....	90

8.4.12	Frlf_ReconfigLPdu .....	91
8.4.13	Frlf_GetNmVector .....	93
8.4.14	Frlf_GetVersionInfo .....	94
8.4.15	Frlf_ReadCCConfig .....	94
8.4.16	Frlf_EnableBusMirroring .....	95
8.5	Interrupt Service Routines .....	97
8.5.1	Frlf_JobListExec_<FrlfCluster.ShortName> .....	97
8.6	Call-back Notifications .....	98
8.6.1	Frlf_CheckWakeupByTransceiver .....	98
8.7	Scheduled Functions .....	99
8.7.1	Frlf_MainFunction_<FrlfCluster.ShortName> .....	99
8.8	Expected Interfaces .....	100
8.8.1	Mandatory Interfaces .....	100
8.8.2	Optional Interfaces .....	101
8.8.3	Configurable Interfaces .....	104
9	Sequence Diagrams .....	110
9.1	Data Transmission .....	110
9.1.1	TransmitWithImmediateBufferAccess .....	110
9.1.2	TransmitWithDecoupledBufferAccess .....	111
9.1.3	ProvideTxConfirmation .....	112
9.2	Data Reception .....	113
9.2.1	ReceiveAndIndicate .....	113
9.2.2	ReceiveAndStore .....	114
9.2.3	ProvideRxIndication .....	115
9.2.4	Cancel Transmission .....	116
9.3	Prepare LPDU .....	117
10	Configuration Specification .....	118
10.1	How to Read this Chapter .....	118
10.2	Containers and Configuration Parameters .....	118
10.2.1	Frlf .....	119
10.2.2	FrlfGeneral .....	120
10.2.3	FrlfCluster .....	128
10.2.4	FrlfController .....	138
10.2.5	FrlfTransceiver .....	141
10.2.6	FrlfLPdu .....	142
10.2.7	FrlfFrameTriggering .....	143
10.2.8	FrlfJobList .....	147
10.2.9	FrlfJob .....	148
10.2.10	FrlfCommunicationOperation .....	150
10.2.11	FrlfFrameStructure .....	151
10.2.12	FrlfPduInFrame .....	152
10.2.13	FrlfPdu .....	153
10.2.14	FrlfTxPdu .....	154
10.2.15	FrlfRxPdu .....	157
10.2.16	FrlfPduDirection .....	159
10.2.17	FrlfConfig .....	159
10.2.18	FrlfClusterDemEventParameterRefs .....	160
10.2.19	FrlfFrameTriggeringDemEventParameterRefs .....	162
10.3	Published Information .....	163

11 Not applicable requirements ..... 164



# 1 Introduction and Functional Overview

This specification specifies the functionality, API and the configuration of the AUTOSAR Basic Software module "FlexRay Interface".

In the AUTOSAR Layered Software Architecture Layered Software Architecture, the FlexRay Interface belongs to the *ECU Abstraction Layer*, or more precisely, to the *Communication Hardware Abstraction*. This indicates the main task of the FlexRay Interface:

Provide to upper layers an abstract interface to the FlexRay Communication System. At least as far as data transmission (i.e. data sending and reception) is concerned, this interface shall be uniform for all bus systems in Autosar (FlexRay, CAN, LIN). Thus, the upper layer (Communication Services like PDU Router, Transport Protocol, and Network Management and others) may access all underlying bus systems for data transmission in a uniform manner. The configuration of the FlexRay Interface however is bus-specific, since it takes into account the specific features of the communication system.

The FlexRay Interface does not directly access the FlexRay hardware (FlexRay Communication Controller and FlexRay Transceiver), but by means of one or more hardware-specific Driver modules.

In order to access the FlexRay Communication Controller(s), the FlexRay Interface uses one or multiple FlexRay Driver modules, which abstract the specific features and interfaces ([CHI](#)) of the respective FlexRay Communication Controller(s).

Likewise, in order to access the FlexRay Transceiver(s), the FlexRay Interface shall use one or multiple FlexRay Transceiver Driver module(s), which abstract the specific features and interfaces of the respective FlexRay Transceiver(s)

Therefore, the FlexRay Interface executable code (however, not the configuration used during runtime) shall be completely independent of the FlexRay Communication Controller(s) and the FlexRay Transceiver(s).

Note: The FlexRay Interface is specified in a way that allows for object code delivery of the code module, following the "one-fits-all" principle, i.e. the entire configuration of the FlexRay Interface can be carried out without modifying any source code. Thus, the configuration of the FlexRay Interface can be carried out largely without detailed knowledge of the underlying hardware.

The FlexRay Interface provides to upper layer AUTOSAR [BSW](#) modules the following groups of functions:

- initialization
- data transmission (sending and reception)
- start/halt/abort communication
- FlexRay specific functions (e.g. send wake-up pattern)
- set operation mode
- get status information
- various timer functions

## 2 Information about this Document

### 2.1 General Hints

In general, the FlexRay Interface has no knowledge of the origin of a PDU passed to it in an API service call.

Therefore, throughout this document, the term "PDU" is being used for PDUs originating from or sent to:

- AUTOSAR Com (I-PDU) via the PDU-Router, or
- AUTOSAR FlexRay TP (N-PDU), or
- AUTOSAR FlexRay NM
- AUTOSAR XCP

In addition to the above-mentioned AUTOSAR BSW modules, the Frlf shall, with the functionality described within the specification in hand, also support other non-AUTOSAR upper layer software modules (Complex Drivers), provided that these modules interact with the Frlf in the same manner as the upper layer AUTOSAR BSW modules.

Throughout this document, several scenarios for changing configuration data are mentioned. They are being used as follows:

- "**pre compile time**" = carried out *before* compiling the code of the FlexRay Interface, since the code generation depends on this setting.
- "**at system configuration time**" = static configuration parameters stored in the FlexRay Interface; may be defined *after* compilation of the code of the FlexRay Interface ("**link time**" or "**post build time**"), but have to be defined *before* the first execution of the FlexRay Interface code.
- "**during runtime**" = dynamically switching (in [POC](#):*normal active* state of the FlexRay [CC](#), if supported) between different configuration parameter sets stored in the static configuration of the FlexRay Interface, or the FlexRay Driver, respectively.

Everything not explicitly mentioned in this document, should be considered as implementation-specific.

## 2.2 Acronyms and Abbreviations

The following acronyms and abbreviations are used throughout this document:

Acronym:	Description:
BSW	(AUTOSAR) Basic Software
CAS	Collision Avoidance Symbol
CC	(FlexRay) Communication Controller
CDD	Complex Driver
CHI	Controller Host Interface of a FlexRay <a href="#">CC</a>
COM	Communication (AUTOSAR BSW module)
ComM	Communication Manager (AUTOSAR BSW module)
DEM	Diagnostic Event Manager (AUTOSAR BSW module)
DET	Default Error Tracer (AUTOSAR BSW module)
FrIf	FlexRay Interface (AUTOSAR BSW module)
FrNm	FlexRay Network Management (AUTOSAR BSW module)
FrTp	FlexRay Transport Layer (AUTOSAR BSW module)
ISR	Interrupt Service Routine
MCG	Module Configuration Generator
PduR	PDU Router (AUTOSAR BSW module)
POC	Protocol Operation Control
WUDOP	Wake-Up During Operation
WUP	Wake-Up Pattern
WUS	Wake-Up Symbol
System Designer	The person responsible for the configuration of all system parameters that do not influence the <b>executable code</b> itself (i.e. the sequence of instructions executed during runtime), but the <b>data</b> used to <b>configure</b> <i>which operations</i> this executable code performs on <i>which data</i> and at <i>which points in time</i> .

Abbreviation:	Description:
i.e.	[lat.] id est = [eng.] that is
e.g.	[lat.] exempli gratia = [eng.] for example
N/A	not applicable

## 3 Related Documentation

### 3.1 Input Documents

- [1] List of Basic Software Modules  
AUTOSAR\_TR\_BSWModuleList.pdf
  
- [2] Layered Software Architecture  
AUTOSAR\_EXP\_LayeredSoftwareArchitecture.pdf
  
- [3] General Requirements on Basic Software Modules  
AUTOSAR\_SRS\_BSWGeneral.pdf
  
- [4] Input for API Specification of AUTOSAR COM Stack
  
- [5] Specification of Communication Stack Types  
AUTOSAR\_SWS\_CommunicationStackTypes.pdf
  
- [6] Requirements on FlexRay  
AUTOSAR\_SRS\_FlexRay.pdf
  
- [7] Specification of FlexRay Driver  
AUTOSAR\_SWS\_FlexRay.pdf
  
- [8] Specification of FlexRay State Manager  
AUTOSAR\_SWS\_FlexRayStateManager.pdf
  
- [9] Specification of FlexRay Transceiver Driver  
AUTOSAR\_SWS\_FlexRayTransceiverDriver.pdf
  
- [10] Specification of FlexRay Transport Layer  
AUTOSAR\_SWS\_FlexRayTransportLayer.pdf
  
- [11] Specification of FlexRay Network Management  
AUTOSAR\_SWS\_FlexRayNetworkManagement.pdf
  
- [12] Specification of PDU Router  
AUTOSAR\_SWS\_PDURouter
  
- [13] Specification of [BSW](#) Scheduler  
AUTOSAR\_SWS\_BSW\_Scheduler
  
- [14] Specification of ECU Configuration  
AUTOSAR\_TPS\_ECUConfiguration
  
- [15] Specification of Memory Mapping  
AUTOSAR\_SWS\_MemoryMapping

- [16] General Specification of Basic Software Modules  
AUTOSAR\_SWS\_BSWGeneral.pdf

### **3.2 Related Standards and Norms**

- [17] FlexRay Communications System Protocol Specification Version 2.1  
Revision A
- [18] FlexRay Communications System Electrical Physical Layer Specification  
Version 2.1 Revision A
- [19] FlexRay Communications System Protocol Specification Version 3.0
- [20] Flexray Communications System Electrical Physical Layer Specification 3.0

### **3.3 Related specification**

AUTOSAR provides a General Specification on Basic Software modules [16] (SWS BSW General), which is also valid for FlexRay Interface.

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

## 4 Constraints and Assumptions

### 4.1 Limitations

The FlexRay [BSW](#) modules are only able to handle a single thread of execution per Cluster. The execution for a particular Cluster must not be pre-empted by itself for the same Cluster. The same applies to the execution of the FlexRay Job List Execution Function.

It is not possible to transmit signals, PDUs, and/or L-SDUs, which exceed the available buffer size of the used FlexRay [CC](#) during normal operation. Longer signals, PDUs, and/or L-SDUs have to be transmitted using the FlexRay Transport Protocol.

Note: The FlexRay Interface does not make any PDU payload-dependent routing decisions.

Note: In order for the AUTOSAR FlexRay [BSW](#) ([Erlf](#) and FlexRay Driver) modules to be able to control a FlexRay [CC](#), this [CC](#) must allow for configuring its transmit/receive buffers to support the Cycle Counter Filter Criterion / (Support of Slot/Cycle Multiplexing)

For 2.1 FlexRay Hardware, the following Cycle Counter Filtering is possible

**Cycle Number = (B + n \* 2R)mod64**

with **exactly one tuple** of values for **B** and **2R**, where:

- Base Cycle **B** ∈ [0 ... 63]
- Cycle Repetition **2R** ; R ∈ [0 ... 6]
- Variable **n** = 0 ... 63
- **B < 2R**

For 3.0 FlexRay Hardware, the Cycle Counter Filtering shall be possible as described in [19]

## 4.2 Applicability to Car Domains

The FlexRay BSW Stack can be used wherever high data rates and fault tolerant communication (in conjunction with [AUTOSAR COM](#)) are required. Of course, it can also be used for less-demanding use cases, i.e. for low data rates or non-fault-tolerant communication. Furthermore, it enables the synchronized operation of several ECUs within a car.

## 5 Dependencies to Other Modules

### 5.1 AUTOSAR Operating System

**[SWS\_FrIf\_05099]** [There is one dedicated FlexRay Job List Execution Function for each FlexRay Cluster. ] ()

**[SWS\_FrIf\_05100]** [The FlexRay Interface module shall execute the Flexray Job List Execution Function. ] ()

Note: It is up to the implementer whether the FlexRay Job List Execution Functions run in a task context or in an ISR.

### 5.2 All Upper Layer AUTOSAR BSW Modules

**[SWS\_FrIf\_05050]** [The calling of the FlexRay Job List Execution Function by the FlexRay Interface module synchronously to the FlexRay Global Time shall ensure that both the indication (to an upper layer [BSW](#) module) of received data and the request (to an upper layer [BSW](#) module) for data to be sent occur synchronously to the FlexRay Global Time. ] (SRS\_Fr\_05000)

**[SWS\_FrIf\_05148]** [The FlexRay Interface module shall ensure data consistency in its buffers. ] ()

Rationale for [SWS\\_FrIf\\_05148](#): If the respective upper layer [BSW](#) module does not operate synchronously to the FlexRay Global Time, these occurrences are asynchronous to the code execution of this [BSW](#) module.

### 5.3 AUTOSAR PDU-Router

The [FrIf](#) module declares and calls some callback functions of the PDU-Router in order to confirm transmission and notify reception of PDUs.



## 5.4 AUTOSAR FlexRay Network Management

The [Frlf](#) module declares and calls some callback functions of the FlexRay Network Management in order to confirm transmission and notify reception of PDUs.

## 5.5 AUTOSAR FlexRay Transport Protocol

The [Frlf](#) module declares and calls some callback functions of the FlexRay Transport Protocol in order to confirm transmission and notify reception of PDUs.

## 5.6 AUTOSAR Bus Mirroring

The [Frlf](#) module calls a callback function of the Bus Mirroring module in order to report received and transmitted frames, which in turn calls some service functions of the [Frlf](#) module to acquire the network state.

## 5.7 AUTOSAR FlexRay Driver

The [Frlf](#) module has a tight relation to the FlexRay Driver since many of the FlexRay-related services offered by the [Frlf](#) module to upper layer [BSW](#) modules are actually carried out by the FlexRay Driver [BSW](#) module. For those services, the [Frlf](#) module mainly performs only an abstraction of the communication hardware specific information (e.g. the topology of the FlexRay Communication System) and then calls the respective FlexRay Driver with the appropriate parameters.

The FlexRay Driver module has to be the only BSW module which has to run necessarily synchronous to the FlexRay Interface.

## 5.8 AUTOSAR FlexRay Transceiver Driver

The [Frlf](#) module has a tight relation to the FlexRay Transceiver Driver since calls of API services of the FlexRay Transceiver Driver are also routed through the [Frlf](#) module in order to abstract the communication hardware specific information (e.g. the topology of the FlexRay Communication System).

## 5.9 File Structure

### 5.9.1 Header File Structure

Please refer to the chapter 5.1.7 Header file structure in “SWS\_BSWGeneral” [16].

**[SWS\_Frlf\_05087]** [The Frlf module source code file(s) shall include *SchM\_Frlf.h* if data consistency mechanisms of the BSW scheduler are required as described in [13]. ] 0

**[SWS\_Frlf\_05090]** [The header file *Frlf.h* shall contain a software and specification version number. ] (SRS\_BSW\_00004)

**[SWS\_Frlf\_05095]** [ *Mirror.h* contains the declaration of the API service the Bus Mirroring module offers to the FlexRay Interface. This header is only included if Bus Mirroring is enabled (see *FrlfBusMirroringSupport*). ] (SRS\_BSW\_00004)

## 6 Requirements Traceability

Requirement	Description	Satisfied by
SRS_BSW_00004	All Basic SW Modules shall perform a pre-processor check of the versions of all imported include files	SWS_Frlf_05090, SWS_Frlf_05095
SRS_BSW_00101	The Basic Software Module shall be able to initialize variables and hardware in a separate initialization function	SWS_Frlf_05003
SRS_BSW_00170	The AUTOSAR SW Components shall provide information about their dependency from faults, signal qualities, driver demands	SWS_Frlf_05089
SRS_BSW_00171	Optional functionality of a Basic-SW component that is not required in the ECU shall be configurable at pre-compile-time	SWS_Frlf_05089
SRS_BSW_00304	All AUTOSAR Basic Software Modules shall use the following data types instead of native C data types	SWS_Frlf_05001
SRS_BSW_00334	All Basic Software Modules shall provide an XML file that contains the meta data	SWS_Frlf_05089
SRS_BSW_00336	Basic SW module shall be able to shutdown	SWS_Frlf_05006
SRS_BSW_00342	It shall be possible to create an AUTOSAR ECU out of modules provided as source code and modules provided as object code, even mixed	SWS_Frlf_05078
SRS_BSW_00345	BSW Modules shall support pre-compile configuration	SWS_Frlf_05069
SRS_BSW_00348	All AUTOSAR standard types and constants shall be placed and organized in a standard type header file	SWS_Frlf_05001
SRS_BSW_00353	All integer type definitions of target and compiler specific scope shall be placed and organized in a single type header	SWS_Frlf_05001
SRS_BSW_00358	The return type of init() functions implemented by AUTOSAR Basic Software Modules shall be void	SWS_Frlf_05003
SRS_BSW_00361	All mappings of not standardized keywords of compiler specific scope shall be placed and organized in a compiler specific type and keyword header	SWS_Frlf_05001
SRS_BSW_00375	Basic Software Modules shall report wake-up reasons	SWS_Frlf_05036
SRS_BSW_00378	AUTOSAR shall provide a boolean type	SWS_Frlf_05001
SRS_BSW_00404	BSW Modules shall support post-build configuration	SWS_Frlf_05069
SRS_BSW_00405	BSW Modules shall support multiple configuration sets	SWS_Frlf_05003
SRS_BSW_00407	Each BSW module shall provide a function to read out the version information of a dedicated module implementation	SWS_Frlf_05002
SRS_BSW_00411	All AUTOSAR Basic Software Modules shall apply a naming rule for enabling/disabling the existence of the API	SWS_Frlf_05002
SRS_BSW_00414	Init functions shall have a pointer to a configuration structure as single parameter	SWS_Frlf_05003
SRS_Fr_05000	Synchronous SW Modules shall be supported	SWS_Frlf_05050
SRS_Fr_05007	The FlexRay Interface shall be able to communicate with at least four FlexRay CCs via the appropriate FlexRay	SWS_Frlf_05053

	Driver(s)	
SRS_Fr_05010	Each PDU shall have one PDU-ID	SWS_Frlf_05052
SRS_Fr_05013	The local Memory Space shall be initialized	SWS_Frlf_05003
SRS_Fr_05015	The FlexRay Interface shall provide a software interface to start-up a specific FlexRay CC	SWS_Frlf_05005
SRS_Fr_05016	A FlexRay CC Communication shall be aborted when wanted	SWS_Frlf_05007
SRS_Fr_05018	The FlexRay Interface shall provide a software interface to send a wake-up pattern on a channel or CC	SWS_Frlf_05011
SRS_Fr_05022	FlexRay CC POC Status shall be available	SWS_Frlf_05014
SRS_Fr_05027	A PDU shall be transmitted via the FlexRay communication system	SWS_Frlf_05063
SRS_Fr_05031	A FlexRay CC shall be initialized and configured	SWS_Frlf_05004
SRS_Fr_05039	The Operation Mode of a FlexRay Transceiver shall be set	SWS_Frlf_05034
SRS_Fr_05042	The FlexRay Interface shall allow switching from one configuration to another one in Normal Active Mode	SWS_Frlf_05061
SRS_Fr_05056	Configuration of the FlexRay Interface shall be done at System Configuration Time	SWS_Frlf_05054
SRS_Fr_05063	A FlexRay CC Communication shall be halted when wanted	SWS_Frlf_05006
SRS_Fr_05096	Communication controllers shall be assigned to FlexRay Driver.	SWS_Frlf_05060
SRS_Fr_05097	The FlexRay Interface shall be able to communicate with at least four FlexRay Drivers	SWS_Frlf_05057
SRS_Fr_05126	PDU Update/Valid Information shall be handled	SWS_Frlf_05056
SRS_Fr_05130	The FlexRay Interface shall support PDU transmission buffer queues	SWS_Frlf_05058
SRS_Fr_05157	The Operation Mode of a FlexRay Transceiver shall be available	SWS_Frlf_05035
SRS_Fr_05158	The wake-up reason of a specific FlexRay Transceiver device shall be available	SWS_Frlf_05036
SRS_Fr_05161	Pending Wake-up Events of a Transceiver shall be cleared if necessary	SWS_Frlf_05039
SRS_Fr_05170	PDUs received via the FlexRay communication system shall be retrieved	SWS_Frlf_05062

## 6.1 Specification Items

The following Items shall be seen as implementation hints only!

### Functional Specification

Abstraction of FlexRay Transceivers	Frlf05105, Frlf05106
-------------------------------------	-------------------------

Usage of Controller and Channel Index	Frlf05106
Usage of zero-based index	SWS_Frlf_05107
Usage of FR Cluster Index	Frlf05108
Configuration Data	Frlf05109
Usage of PDU index	SWS_Frlf_05110
Support one of both or both FlexRay Channels	SWS_Frlf_05111
Support of at least four FlexRay Clusters	SWS_Frlf_05112
Support of at least one absolute timer per FlexRay CCs	SWS_Frlf_05113

### FlexRay Interface State Machine

One State Machine per Cluster	SWS_Frlf_05115
Frlf_State offline during initialization	SWS_Frlf_05117

### FlexRay Interface Main Function

One Main Function for each FlexRay Cluster	SWS_Frlf_05119
Main Function tasks	Frlf05120

### Data Communication via FlexRay

Packaging of multiple PDUs in one FR Frame	SWS_Frlf_05121
Frame construction plan (layout)	SWS_Frlf_05122
Frame construction plan (config)	SWS_Frlf_05123
Transmission rule	SWS_Frlf_05124
Update Information per PDU	SWS_Frlf_05125
Location of Update Information	SWS_Frlf_05126
Configuration of Update Information	SWS_Frlf_05127
Indication in case of no update information	SWS_Frlf_05128
Transmission with Immediate Buffer Access	SWS_Frlf_05129
Ensure synchronous buffer access	SWS_Frlf_05130
Sortation of Communication Job	SWS_Frlf_05131
Communication Job properties	<a href="#">Frlf05368</a>
Communication Job execution start time	SWS_Frlf_05133
Actions specified by Communication Operation	SWS_Frlf_05134
Communication Operation properties	<a href="#">Frlf05369</a>
Job List Execution Function nameing	SWS_Frlf_05136
Job List synchronously to global time	SWS_Frlf_05137
Job List Execution Function actions	SWS_Frlf_05138

## 7 Functional Specification

### 7.1 FlexRay BSW Stack

As part of the AUTOSAR Layered Software Architecture according to [2], the FlexRay [BSW](#) modules also form a layered software stack.

Figure 7-1 depicts the basic structure of this FlexRay [BSW](#) stack. The [Frlf](#) module accesses several [CCs](#) using the FlexRay Driver layer, which can be made up of several FlexRay Drivers modules. The FlexRay Transceivers are not shown in this figure; however, the structure that applies to the FlexRay Drivers and the FlexRay [CCs](#) analogously applies to the FlexRay Transceiver Drivers and the FlexRay Transceivers.

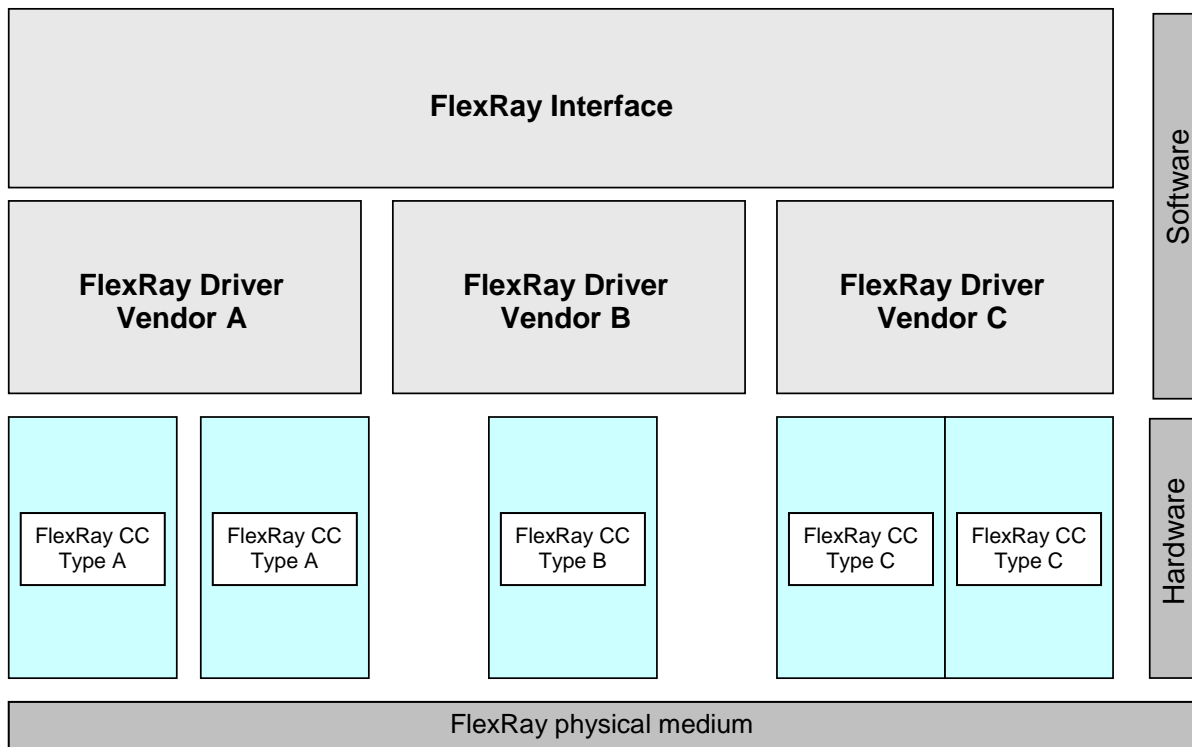


Figure 7-1: Basic Structure of the FlexRay BSW Stack

## 7.2 Indexing Scheme

### 7.2.1 Principle

Most of the [Frlf](#) module's API services used for accessing the numerous (hardware and software) resources<sup>1</sup> map to corresponding API services of the underlying FlexRay Driver(s), or FlexRay Transceiver Driver(s), respectively.

In order to select those resources spread over the various entities<sup>2</sup> accessed via the [Frlf](#) module, the FlexRay-related AUTOSAR [BSW](#) modules use an indexing scheme that is exemplarily described in Figure 7-2 and Figure 7-3.

**Definition** ControllerIndex: The ControllerIndex is an abstract, unique, zero-based consecutive index to achieve the abstraction of the FlexRay Communication Controllers, independent of their type, location, and access method.

**Definition** ClusterIndex: The ClusterIndex is an abstract, unique, zero-based consecutive index to achieve the abstraction of the FlexRay Clusters, independent of their type, location, and access method.

**Definition** ChannelIndex: The ChannelIndex has either the value FR\_CHANNEL\_A or FR\_CHANNEL\_B. In combination with the ControllerIndex, the corresponding FlexRay Transceiver is identified.

**[SWS\_Frlf\_05052]** [The [Frlf](#) module shall achieve the abstraction (of the CCs and Drivers) by providing to the upper layer [BSW](#) modules an abstract, unique, zero-based consecutive index for each sort of resource, independent of their type, location, and access method. ] (SRS\_Fr\_05010)

**Rationale:** The Frlf module achieves the abstraction (of the CCs and Drivers) by providing these abstract indices to the upper layer BSW modules.

The [Frlf](#) module API service uses the abstract index passed to it by the upper layer [BSW](#) module to retrieve:

1. **the function pointer to a corresponding lower layer BSW module's API service** from a static configuration data table containing function pointers to all API services of all lower layer [BSW](#) modules called by the [Frlf](#) module, and
2. **the translated index used in the call to the lower layer BSW module's API service** from a static configuration data table.

Since this static configuration data table contains function pointers to the lower layer BSW module's API services, it obviously has to be linked against the linked and located code of the lower layer BSW modules.

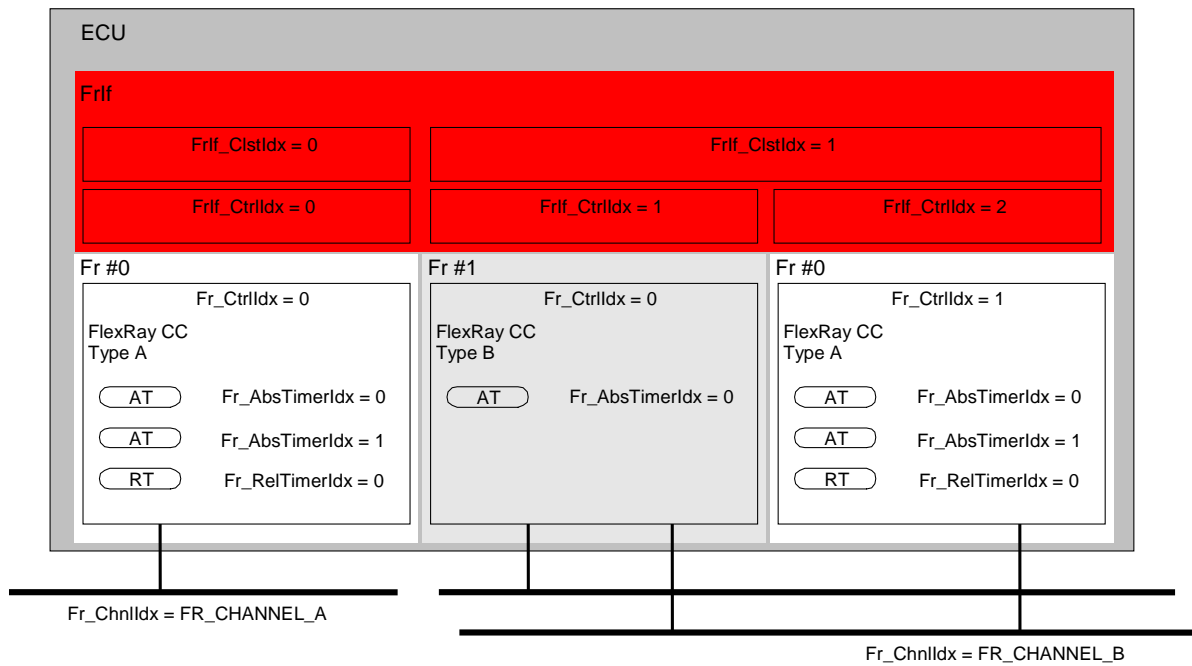
The [Frlf](#) module then calls the corresponding lower layer [BSW](#) module's API service via the function pointer and passes the translated index in the API call.

The function descriptions in chapter 8 specify the required calls of corresponding lower layer [BSW](#) module's API services in detail.

---

<sup>1</sup> E.g. timers, configuration data sets, etc.

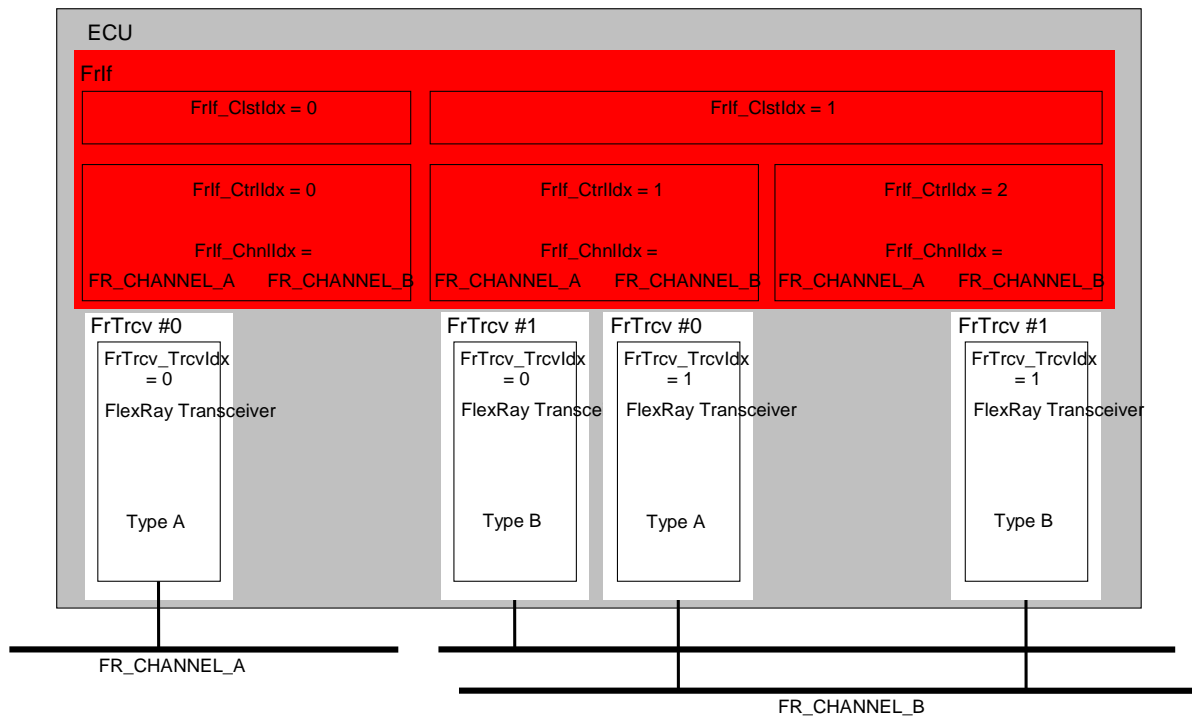
<sup>2</sup> FlexRay Drivers, FlexRay Communication Controllers, FlexRay Transceiver Drivers, and FlexRay Transceivers



**Figure 7-2: CC Indexing Scheme of the FlexRay Interface**

**[SWS\_FrIf\_05060]** [In order to abstract for upper layer [BSW](#) modules the various CCs, which the [FrIf](#) module controls via the FlexRay Driver modules, the [FrIf](#) module offers an abstract, unique, zero-based consecutive index `FrIfCtrlIdx` as configuration parameter, which maps to a tuple of FlexRay Driver API Service function pointer and CC index `Fr_CtrlIdx`. ] (SRS\_Fr\_05096)





**Figure 7-3: Flexray Transceiver Indexing Scheme of the FlexRay Interface**

In order to abstract for upper layer [BSW](#) modules the various FlexRay Transceiver modules, which the [FrIf](#) module accesses via the FlexRay Transceiver Driver modules, the [FrIf](#) module takes advantage of the fact that each FlexRay Transceiver module is unambiguously assigned to a specific Channel on a specific FlexRay [CC](#).

Therefore, the [FrIf](#) module abstracts the various FlexRay Transceivers by a **combination** of the two indices **FrIf\_CtrlIdx** (Controller Index) and **FrIf\_ChnlIdx** (Channel Index) and maps this to a tuple of FlexRay Transceiver Driver API Service function pointer and FlexRay Transceiver index **FrTrcv\_Idx**. (Transceiver Index)

The function descriptions in chapter 8 specify the required mapping of upper layer BSW module's parameters to corresponding lower layer [BSW](#) module's API services in detail.”

**[SWS\_FrIf\_05107]** [Besides hardware and software resources, the [FrIf](#) module also numbers the logical structure elements presented by FlexRay with an abstract, unique, zero-based consecutive index.

The static configuration data of the [FrIf](#) module contains a data structure that specifies which FlexRay [CC](#) modules and which FlexRay Transceiver modules are connected to which Clusters, or in other words, that maps each value of **FrIf\_ClstIdx** to (one, or in general) a set of values for **FrIf\_CtrlIdx** and tuples of (**FrIf\_CtrlIdx**, **FrIf\_ChnlIdx**). ] ()

[SWS\_Frlf\_05110] [The [Frlf](#) module shall number all PDUs to be transmitted with an abstract, unique, zero-based consecutive index TxPduld. ] ()

Note: This index is used in the [Frlf](#) API service Frlf\_Transmit() and allows the [Frlf](#) module to quickly identify (e.g. by a table look-up) the PDU that is passed to it by an upper layer [BSW](#) module, and to process it accordingly.

### 7.2.2 Supported Indexed Resources

[SWS\_Frlf\_05057] [It shall be possible that the [Frlf](#) module can be configured to support at least four (possibly different) **FlexRay Drivers** to access the FlexRay Communication Controllers. ] (SRS\_Fr\_05097)

[SWS\_Frlf\_05053] [It shall be possible that the [Frlf](#) module can be configured using the parameter FRIF\_CTRL\_IDX to support at least four (possibly different) **FlexRay CCs**. ] (SRS\_Fr\_05007)

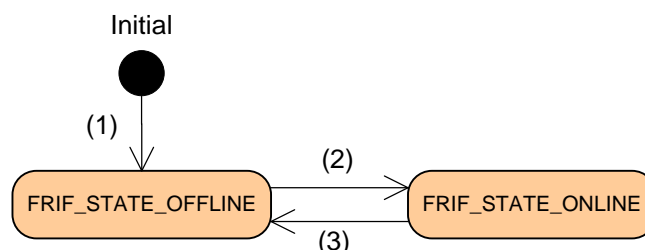
[SWS\_Frlf\_05111] [It shall be possible that the [Frlf](#) module can be configured to support one of both or both **FlexRay Channels** as specified in [17]. ] ()

[SWS\_Frlf\_05112] [It shall be possible that the [Frlf](#) module can be configured using the parameter FRIF\_CLST\_IDX to support at least four **FlexRay Clusters**. ] ()

[SWS\_Frlf\_05113] [It shall be possible that the [Frlf](#) module can be configured using the parameter FRIF\_ABS\_TIMER\_IDX to support at least one **absolute timer** per FlexRay [CCs](#). ] ()

### 7.3 FlexRay Interface State Machine

[SWS\_Frlf\_05115] [ In order to allow to control the communication operations of the FlexRay system, the [Frlf](#) module shall implement a behavior, which is defined using a simple state machine (one per FlexRay cluster), called FlexRay Interface State Machine



**Figure 7-4: FlexRay Interface State Machine**

Figure 7-4 shows the states and transitions that are visible to the user of a [Frlf](#) module. The two different states, which are defined as Frlf type Frlf\_StateType (see 8.2.2), represent the communication capabilities of a Frlf module.

State	Description
FRIF_STATE_OFFLINE	No communication services are executed (see chapter 7.6 for details)
FRIF_STATE_ONLINE	All communication services (reception, transmission, transmission confirmation) are executed (see chapter 7.6 for details).

] ()

**[SWS\_Frlf\_05117]** [During initialization of the Frlf by executing Frlf\_Init() the Frlf\_State for each cluster shall be initialized with state 'FRIF\_STATE\_OFFLINE'.

The transitions are requested by an API service Frlf\_SetState() which takes the Cluster to process on and the Transition name to invoke. ] ()

**[SWS\_Frlf\_05118]** [If the Frlf module's environment calls the function Frlf\_SetState with parameter Frlf\_StateTransition = FRIF\_GOTO\_ONLINE and if the current state for the requested cluster is FRIF\_STATE\_OFFLINE, the Frlf module shall take the current state of the requested cluster to FRIF\_STATE\_ONLINE." (refer to figure 7-4 transition (2)).

If the Frlf module's environment calls the function Frlf\_SetState with parameter Frlf\_StateTransition = FRIF\_GOTO\_OFFLINE and if the current state for the requested cluster is FRIF\_STATE\_ONLINE, the Frlf module shall take the current state of the requested cluster to FRIF\_STATE\_OFFLINE." (refer to figure 7-4 transition (3)).

Otherwise, do not perform a state transition.

Transition Name	Transitions (see Figure 7-4)	Description
FRIF_GOTO_ONLINE	(2)	Transition resulting in Frlf_State FRIF_STATE_ONLINE
FRIF_GOTO_OFFLINE	(3)	Transition resulting in Frlf_State FRIF_STATE_OFFLINE

] ()

**[SWS\_Frlf\_05501]** ¶ If the API `Frlf_SetState` with parameter `FRIF_STATE_OFFLINE` is called, the FlexRay Interface module shall check the parameter "TxConfCounter" for every PDU. If the value for the corresponding PDU is greater than 0, the FlexRay Interface shall call the upper layer using the API `<UL>_TxConfirmation(id, E_NOT_OK)`. ¶ ()

Note: It has to be ensured that the FlexRay Interface does not lose the TxConfCounter values at the point in time the API `Frlf_SetState` with parameter `FRIF_STATE_OFFLINE` is called.

### 7.3.1 FlexRay Interface Main Function

The FlexRay Interface Main Function needs to be called cyclically from a task body provided by the [BSW](#) Scheduler with a calling period (`FRIF_MAINFUNCTION_PERIOD`) depending on the FlexRay Cycle length and configurable [at system configuration time](#).

Since the Cycle length of each Cluster is independent, the desired calling period of the FlexRay Interface Main Function might differ from Cluster to Cluster, except for "Transmission with Immediate Buffer Access".

**[SWS\_Frlf\_05119]** ¶ The Frlf module shall provide one dedicated FlexRay Interface Main Function for each FlexRay Cluster that is controlled by that Frlf module. ¶ ()

**[SWS\_Frlf\_05283]** ¶ The API names of the FlexRay Interface Main Functions shall obey the following pattern:

`Frlf_MainFunction_<FrlfCluster.ShortName>` where `FrlfCluster.ShortName` is the Short Name of the corresponding `FrlfCluster`. ¶ ()

**[SWS\_Frlf\_15120]** ¶ The Main Function monitors and controls the continuous execution of the FlexRay Job List Execution Function including the (re)synchronization if the current FlexRay Interface State Machine is `FRIF_STATE_ONLINE`. ¶ ()

**[SWS\_Frlf\_01124]** ¶ If Bus Mirroring is enabled globally (see `FrlfBusMirroringSupport`), then call `Fr_GetChannelStatus` for all controllers of each FlexRay cluster for which mirroring has been activated with a call to `Frlf_EnableBusMirroring()`, merge the states reported for the controllers of one cluster with a binary OR, and then call `Mirror_ReportFlexRayChannelStatus()` with the cluster, `Fr_ChannelAStatusPtr`, and `Fr_ChannelBStatusPtr` to report the aggregated channel states to the Bus Mirroring module. ¶ ()

**[SWS\_Frlf\_25120]** ¶ If one of the optional cluster-specific configuration parameters `FRIF_E_NIT_CH_A`, `FRIF_E_NIT_CH_B`, `FRIF_E_SW_CH_A`, `FRIF_E_SW_CH_B` or `FRIF_E_ACS_CH_A`, `FRIF_E_ACS_CH_B` exists, then call

FrIf\_GetChannelStatus for each FlexRay controller of the cluster and report the status to DEM as described below. ] ()

**[SWS\_FrIf\_35120]** If the optional configuration parameter FRIF\_E\_NIT\_CH\_A exists, then the channel status information shall be reported to DEM as Dem\_SetEventStatus (FRIF\_E\_NIT\_CH\_A, DEM\_EVENT\_STATUS\_PREFAILED) when any of the error bits of a single controller (Channel A NIT status data vSS!SyntaxError, vSS!Bviolation) is set or as Dem\_SetEventStatus (FRIF\_E\_NIT\_CH\_A, DEM\_EVENT\_STATUS\_PREPASSED) when none of these error bits is set. ] ()

**[SWS\_FrIf\_45120]** If the optional configuration parameter FRIF\_E\_NIT\_CH\_B exists, then the channel status information shall be reported to DEM as Dem\_SetEventStatus (FRIF\_E\_NIT\_CH\_B, DEM\_EVENT\_STATUS\_PREFAILED) when any of the error bits of a single controller (Channel B NIT status data vSS!SyntaxError, vSS!Bviolation) is set or as Dem\_SetEventStatus (FRIF\_E\_NIT\_CH\_B, DEM\_EVENT\_STATUS\_PREPASSED) when none of these error bits is set. ] ()

**[SWS\_FrIf\_55120]** If the optional configuration parameter FRIF\_E\_SW\_CH\_A exists, then the channel status information shall be reported to DEM as Dem\_SetEventStatus (FRIF\_E\_SW\_CH\_A, DEM\_EVENT\_STATUS\_PREFAILED) when any of the error bits of a single controller (Channel A symbol window status data vSS!SyntaxError, vSS!Bviolation, vSS!TxConflict) is set or as Dem\_SetEventStatus (FRIF\_E\_SW\_CH\_A, DEM\_EVENT\_STATUS\_PREPASSED) when none of these error bits is set. ] ()

**[SWS\_FrIf\_65120]** If the optional configuration parameter FRIF\_E\_SW\_CH\_B exists, then the channel status information shall be reported to DEM as Dem\_SetEventStatus (FRIF\_E\_SW\_CH\_B, DEM\_EVENT\_STATUS\_PREFAILED) when any of the error bits of a single controller (Channel B symbol window status data vSS!SyntaxError, vSS!Bviolation vSS!TxConflict) is set or as Dem\_SetEventStatus (FRIF\_E\_SW\_CH\_B, DEM\_EVENT\_STATUS\_PREPASSED) when none of these error bits is set. ] ()

**[SWS\_FrIf\_75120]** If the optional configuration parameter FRIF\_E\_ACS\_CH\_A exists, then the channel status information shall be reported to DEM as Dem\_SetEventStatus (FRIF\_E\_ACS\_CH\_A , DEM\_EVENT\_STATUS\_PREFAILED) when any of the error bits of a single controller (Channel A aggregated channel status vSS!SyntaxError, vSS!ContentError, vSS!Bviolation, vSS!TxConflict) is set or as Dem\_SetEventStatus (FRIF\_E\_ACS\_CH\_A, DEM\_EVENT\_STATUS\_PREPASSED) when none of these error bits is set. ] ()

**[SWS\_FrIf\_85120]** If the optional configuration parameter FRIF\_E\_ACS\_CH\_B exists, then the channel status information shall be reported to DEM as Dem\_SetEventStatus (FRIF\_E\_ACS\_CH\_B, DEM\_EVENT\_STATUS\_PREFAILED) when any of the error bits of a single controller (Channel B aggregated channel status vSS!SyntaxError, vSS!ContentError, vSS!Bviolation, vSS!TxConflict) is set or

as Dem\_SetEventStatus (FRIF\_E\_ACS\_CH\_B, DEM\_EVENT\_STATUS\_PREPASSED) when none of these error bits is set. ] ()

**[SWS\_Frlf\_95120]** [If a loss of the JobList's synchronization (see [JobListAsyncFlag](#)) or a miss of execution was detected, the following steps shall be performed:

1. Get the global time (Frlf\_GetGlobalTime())
    - If Frlf\_GetGlobalTime() returns E\_NOT\_OK, stop here
    - If Frlf\_GetGlobalTime() returns E\_OK, continue with step 2
  2. add some 'time buffer' (i.e. some timespan which takes jitter into account)
  3. search the FlexRay Job List for the next job, i.e. that job with an invocation time greater than the current global time + 'time buffer'.
  4. set the JobListPointer to that job and program the absolute timer with this job's invocation time (now the FlexRay Job List is synchronized again)
  5. clear the JobListAsyncFlag
  6. Enable the absolute timer interrupt
- ] ()

## 7.4 Implementation Requirements

**[SWS\_Frlf\_05096]** [The FlexRay Interface executable code (however, not the configuration used during runtime) shall be completely independent of the FlexRay Communication Controller(s) and the FlexRay Transceiver(s). ] ( )

**[SWS\_Frlf\_05069]** [The Frlf module shall support pre-compile time, link-time and post-build-time configuration. ] (SRS\_BSW\_00404, SRS\_BSW\_00345)

**[SWS\_Frlf\_05284]** [The Frlf module shall implement link-time and post-build-time configuration data as read-only data structures. ] ( )

**[SWS\_Frlf\_05285]** [The Frlf module shall immediately reference link-time configuration data by the implementation, ] ( )

**[SWS\_Frlf\_05078]** [The Frlf module shall implement the API functions specified by the Frlf SWS as real C code functions and shall not implement the API functions as macros. ] (SRS\_BSW\_00342)

**Note:**The rationale of [SWS\\_Frlf\\_05078](#) is to allow object code module integration.

**[SWS\_Frlf\_05092]** [The Frlf module shall support dynamic payload length for LPdus whose associated parameter FrlfAllowDynamicLSduLength (see [Frlf06049](#)) is set to true.

FrlfAllowDynamicLSduLength shall only be used for PDUs

- which are the only ones within the Frame Construction Plans, or
- for the last PDU within the Frame Construction Plans

] ()

## 7.5 Configuration description

**[SWS\_Frlf\_05089]** [The Frlf module shall provide an XML file that contains the data which is required for the SW identification (it shall contain the vendor identification, module ID and software version information), configuration and integration process. This file should describe vendor specific configuration parameters as well as it should contain recommended configuration parameter values.

The description of the configuration and initialization data itself is not part of this specification but very implementation specific. ] (SRS\_BSW\_00171, SRS\_BSW\_00170, SRS\_BSW\_00334)

## 7.6 Data Communication via FlexRay

FlexRay in general is a deterministic time-driven communication system.

Each datum that should be transmitted or received has to be scheduled [at system configuration time](#).

This even holds true for data that - from the application's point of view - are considered *event-driven*.

Note: When looking only at specific instances of the AUTOSAR FlexRay software modules running on a specific ECU it is not possible to "anticipate" the **exact point in time** when a certain FlexRay frame is being sent (or received, respectively) in the Dynamic Segment of the FlexRay Cycle.

**[SWS\_Frlf\_05054]** [The Frlf module shall define the resources (e.g. a buffer in the FlexRay Communication Controller or FlexRay Driver) needed for data transmission (or reception, respectively) [at system configuration time](#) specifically for data transmission (or reception, respectively). ] (SRS\_Fr\_05056)

Note: There is no true spontaneous event-driven data communication on FlexRay. Even application data that occur at unpredictable points in time (i.e. "event-driven"), and that should be transmitted via FlexRay, have to be scheduled for transmission [at system configuration time](#).

### 7.6.1 PDU Packing, PDU update bits, and Frame Construction Plans

In accordance with basic AUTOSAR rules, the API services that the [Frlf](#) module provides to upper layer [BSW](#) modules for data transmission and data reception are PDU-based.

**[SWS\_Frlf\_05121]** [The [Frlf](#) module shall be capable of packing multiple PDUs into one FlexRay Frame. ] ()

Rationale for [SWS\\_Frlf\\_05121](#): Bus-independent AUTOSAR PDUs have a maximal length of 8 bytes, but according to [17] a FlexRay Frame can contain as many as 254 bytes of payload data.

Note: It is also allowed to define PDUs which are larger than 8 bytes. Please be aware that PDUs greater than 8 bytes are not bus independent any more!

**[SWS\_Frlf\_05122]** [The Frlf module shall take the information on how to pack PDUs into FlexRay Frames from the so-called Frame Construction Plans. The rules defining how to pack PDUs into FlexRay Frames are defined [at system configuration time](#) ] ()

**[SWS\_Frlf\_05123]** [The Frame Construction Plan shall be stored in the static configuration of the [Frlf](#) module (configuration parameter FrlfFrameStructure, see [Frlf05370](#)). ] ()

**[SWS\_Frlf\_05124]** [If multiple PDUs are packed into a single FlexRay Frame and if the Frlf module recognizes the update of at least one of the contained PDUs, then the Frlf module shall transmit this FlexRay Frame. ] ()

Note: As a result, the space associated with PDUs in this FlexRay Frame that have not been updated by the upper layer BSW module will also be transmitted. This does not necessarily mean that the previous values of those PDUs are transmitted. On the contrary, in case the parameter 'FrlfUnusedBitValue' does not exist, arbitrary values for those PDUs will be transmitted.

**[SWS\_Frlf\_05723]** [In case the parameter 'FrlfUnusedBitValue' exists, all the unused bits within the Frame Construction Plan shall be set to the configured value 'FrlfUnusedBitValue' while assembling the frame on sender side. ] ()

**[SWS\_Frlf\_05725]** [The FlexRayInterface shall ensure that unused spaces within the frame construction plan only contain deterministic values (instead of possible random data).

For this purpose, the value given by the parameter 'FrlfUnusedBitValue' shall be used to fill unused spaces with this value.] ()

**[SWS\_Frlf\_05125]** [It shall be possible to configure (configuration parameter FrlfPduUpdateBitOffset, see Frlf06071) for each PDU a dedicated PDU update bits in the FlexRay Frame. The Frlf module shall identify the position of the PDU update bits



for each PDU using the information stored in configuration parameter  
FrlfPduUpdateBitOffset. ] ()

**[SWS\_Frlf\_05056]** [The receiving Frlf module shall evaluate the PDU Update-bit (if configured) to recognize the update of the PDU associated with this PDU update bits  
] (SRS\_Fr\_05126)

Rationale: In order for the receiving [Frlf](#) module to be able to determine which of the PDUs in a received FlexRay Frame have actually been updated by the upper layer BSW module (by a call of Frlf\_Transmit()) on the transmitter side, additional update information, so called **PDU update bits** within the FlexRay Frame, shall be transmitted to the receiving [Frlf](#) module.

Note: A details description of the update bits handling is described in the Communication Operation, chapter 7.6.3.1 “TransmitWithDecoupledBufferAccess”

**[SWS\_Frlf\_05126]** [This PDU update bits shall be located at an arbitrary bit position in the Frame Construction Plan that is not occupied by any PDU. ] ()

**[SWS\_Frlf\_05127]** [The configuration of update bitss for the PDUs and the definition of the location of the update bitss within the FlexRay Frame are performed [at system configuration time](#) [Configuration Parameter FrlfPduUpdateBitOffset, see [Frlf06071](#)] ] ()

**[SWS\_Frlf\_05128]** [If no update bit is configured for a specific PDU, the Frlf module shall assume this PDU to be always valid and the Frlf module shall always indicate its reception to the upper layer BSW module on the receiver side. ] ()

**[SWS\_Frlf\_05758]** [In case the parameter ‘FrlfAllowDynamicLSduLength’ exists and is set to TRUE for the associated frame triggering for reception, PDUs in non-received areas (PDU offset > actual L-SDU length) shall not be indicated to upper layer(s).  
] ()

**[SWS\_Frlf\_05129]** [If Transmission with Immediate Buffer Access is used, only one PDU is allowed per FlexRay Frame (L-SDU). ] ()

Note: Therefore, PDU update bits can be omitted for Transmission with Immediate Buffer Access.

## 7.6.2 Dynamic PDU length

**[SWS\_Frlf\_05093]** ¶ In case the parameter ‘FrlfAllowDynamicLSduLength’ (see Frlf06049) is set to true for the associated frame triggering, the Frlf module passes the actual used L-PDU length to the driver (Fr\_TransmitTxLPdu()), taking into account the following parameters for each PDU:

- the position of the PDU within the L-PDU
- the position of the update-bit information (if configured)

If FrlfImmediate equals TRUE, the actual length of the respective PDU shall be as passed via Frlf\_Transmit().

If FrlfImmediate equals FALSE, the actual length of the respective PDU shall be as passed via <UL\_TriggerTransmit>()

¶ ()

Note: If FrlfAllowDynamicLSduLength is set to false, the Frlf module just passes the length information according to the frame construction plan to the FlexRay driver.

**[SWS\_Frlf\_05094]** ¶ The Frlf shall only indicate PDUs in received areas (PDU offset <= actual L-PDU length) to upper layer(s). ¶ ()

## 7.6.3 AlwaysTransmit

Note: According to [17], a FlexRay [CC](#) might **only** support the so-called “continuous transmission mode” where a message is transmitted continuously until the host explicitly invalidates the transmit buffer. If such a FlexRay [CC](#) is being used for transmission, and the receiving [Frlf](#) should still be able to determine which of the PDUs in a received FlexRay Frame have actually been updated by an upper layer [BSW](#) module on the transmitter side, a special mechanism is needed in the transmitting [Frlf](#), called **AlwaysTransmit** (configuration parameter FrlfAlwaysTransmit, see ECUC\_Frlf\_06050). If AlwaysTransmit is enabled for an L-PDU that is transmitted using the Communication Operation DECOUPLED\_TRANSMISSION, the FlexRay Driver’s API service Fr\_TransmitTxLPdu() is always called for this L-PDU, independent from any PDUs in this L-PDU having been updated by an upper layer [BSW](#) module. This enables resetting the PDU update bits in the FlexRay [CC](#)’s transmit buffer, even if none of the PDUs in the FlexRay Frame have actually been updated by an upper layer [BSW](#) module, and thus ensures the correct interpretation of the received Frame contents by the receiving [Frlf](#).

**Note:** Since:

- in general, the transmit mode of a FlexRay [CC](#) can be configured (“continuous mode” / “single shot mode”), and

- [AlwaysTransmit](#) can be configured independently per L-PDU, and
- update bits can be configured independently per PDU,

the [Frlf](#) module can be tailored to exhibit exactly the behavior required by a certain use case,

however, it is the responsibility of the [System Designer](#) to select the correct configuration of all these parameters. An incorrect configuration will lead to undesired results.

#### 7.6.4 Realization of the Time-Driven FlexRay Schedule

According to [17], a FlexRay [CC](#) is **not** required to provide mechanisms in hardware to ensure asynchronous access to its transmit and receive buffers e.g. by providing shadow buffers that may be accessed asynchronously by the AUTOSAR FlexRay software modules.

**[SWS\_Frlf\_05130]** [The Frlf module shall call all functions accessing the transmit and receive buffers (i.e. performing data transmission or reception, respectively) synchronously (i.e. synchronized to the FlexRay Global Time) ] ()

Rationale for [SWS\\_Frlf\\_05130](#): The access of Frlf module functions to transmit and receive buffers only at well-defined points in time<sup>3</sup> avoids concurrent access to the buffers by the hardware and the software.

Note: In order to provide this necessary synchronicity, the [Frlf](#) module defines for each Cluster a FlexRay Job List [Configuration Parameter FrlfJobList, see [Frlf05367](#)].

The Cluster's FlexRay Job List is executed by its Job List Execution Function (see 8.5.1) using an absolute timer [Configuration Parameter FrlfAbsTimerRef, see [Frlf06063](#)] of a FlexRay [CC](#) connected to the respective Cluster.

##### 7.6.4.1 FlexRay Job List

**[SWS\_Frlf\_05131]** [Definition: A FlexRay Job List is a list of (maybe different) Communication Jobs sorted according to their respective execution start time.

Each Communication Job [Configuration Parameter FrlfJob, see [Frlf05368](#)] contains the following properties:

- Job start time by means of
  - FlexRay Communication Cycle [Configuration Parameter FrlfCycle, see [Frlf06064](#)]
  - Macrotick Offset within the Communication Cycle [Configuration Parameter FrlfMacrotick, see [Frlf06065](#)].

---

<sup>3</sup> In FlexRay Global Time  
35 of 164

- A list of Communication Operations [Configuration Parameter `FrlfCommunicationOperation`, see [Frlf05369](#)] sorted according to a configurable Communication operation index [Configuration Parameter `FrlfCommunicationOperationIdx`, see [Frlf06068](#)]. The sorting order defines the order of execution of the Communication Operations within a FlexRay Communication Job.

] ()

**[SWS\_Frlf\_05133]** [The Frlf module shall call the respective Cluster's FlexRay Job List Execution Function to execute each FlexRay Communication Job at the execution start time assigned to that Communication Job] ()

**[SWS\_Frlf\_05134]** [The Frlf module shall process the actions determined by the Communication Operations assigned to each FlexRay Communication Job

Each Communication Operation (see [Frlf05369](#)) contains the following properties:

- Communication Operation Index [Configuration Parameter `FrlfCommunicationOperationIdx`, see `ECUC_Frlf_06068`], which determines the execution order of the Communication Operations.
- Communication Action [Configuration Parameter `FrlfCommunicationAction`, see [Frlf06067](#)], which specifies the actual action to perform (see 7.6.5):
  - `DECOUPLED_TRANSMISSION`
  - `TX_CONFIRMATION`
  - `RECEIVE_AND_STORE`
  - `RX_INDICATION`
  - `RECEIVE_AND_INDICATE`
  - `PREPARE_LPDU`
- A reference to a frame triggering (L-PDU) which is associated with the Communication Action to perform [Configuration parameter `FrlfLPduldx`, see [Frlf06058](#)]<sup>4</sup>. ] ()

#### 7.6.4.2 FlexRay Job List Execution Function

Since the Communication Schedule of each FlexRay Cluster is independent, there is one dedicated FlexRay Job List and one dedicated FlexRay Job List Execution Function for each FlexRay Cluster that is controlled by the FlexRay Interface.

The Copy Operation into/from the FlexRay CCs are scheduled within the FlexRay JobLists' communication operations

**[SWS\_Frlf\_05136]** [The API names of the FlexRay Job List Execution Functions shall obey the following pattern:

`Frlf_JobListExec_<FrlfCluster.ShortName>` where `FrlfCluster.ShortName` is the Short Name of the corresponding `FrlfCluster`. ] ()

<sup>4</sup> The LPDU is identified by a LPdu Index, which has a 1:1 association to a frame triggering for historical reasons. To obtain compatibility this configuration structure is not changed here. The L-PDU index is identified with a zero-based and dense index, which shall be used as the parameter `Fr_LPduldx` passed to the AUTOSAR FlexRay Driver when processing LPdus.

**[SWS\_FrIf\_05137]** [The FlexRay Job List Execution Function shall execute the Cluster's FlexRay Job List Jobs synchronously to the Cluster's global time (i.e. at well-defined points in time). ] ()

**[SWS\_FrIf\_05138]** [Upon invocation, the FlexRay Job List Execution Function shall perform the following steps:

1. Retrieve the FlexRay Global Time from the FlexRay [CC](#) providing the Cluster's absolute timer interrupt.
2. If the FlexRay Global Time cannot be retrieved or the global time delay compared to the jobs start time is larger than a maximum delay [Configuration Parameter FrIfMaxIsrDelay, see FrIf06004], the execution of the FlexRay Job List is considered to be asynchronous to the FlexRay Global Time and thus the following actions are performed:
  - Either set a flag (JobListAsyncFlag) indicating that the execution of the FlexRay Job List of this Cluster is asynchronous or directly resynchronize the Joblist as described in SWS\_FrIf\_95120
  - If the JobListAsyncFlag was set, call the Runtime error FRIF\_E\_JLE\_SYNC
  - Disable absolute Timer Interrupt
  - Terminate the execution of this FlexRay Job.

Otherwise, the FlexRay Job List Execution Function continues with step 3.

3. Retrieve the ordered list of Communication Operations of the current Job pointed to by the current job-pointer.
4. Forward the current job-pointer to the next job-list entry. If the job-pointer was pointed at the end of the job-list, wrap around and set it to the first job-list entry.
5. Retrieve the execution start time of the job marked by the job-pointer and set the absolute timer to this job's start time in order to invoke the FlexRay Job List Execution Function again.
6. Execute the retrieved Communication Operations.

] ()

Note: In order to keep the runtime of the JLEF short, it is acceptable to implement the described functionality of the JLEF into a separate, high priority task which has to be activated immediately in the JLEF.

### 7.6.5 Communication Operations

This chapter describes each Communication Operation that is executed within the Job List Execution Function.

### 7.6.5.1 TransmitWithDecoupledBufferAccess

**[SWS\_Frlf\_05058]** [The Frlf module shall be capable of Transmit Request queuing by using the TrigTxCounter. ] (SRS\_Fr\_05130)

Note: Only the amount of transmit requests are stored, not the data itself.

**[SWS\_Frlf\_05063]** [If the related CC is in Frlf\_State FRIF\_STATE\_ONLINE for a Communication Operation DECOUPLED\_TRANSMISSION, then the Job List Execution Function shall execute this Communication Operation. Otherwise, the Job List Execution Function shall ignore this Communication Operation. ] (SRS\_Fr\_05027)

**[SWS\_Frlf\_05287]** [For a Communication Operation DECOUPLED\_TRANSMISSION the Job List Execution Function shall perform the following steps

1. Iterate over all PDUs contained in the FrlfFrameStructure (see Frlf05370) of the associated frame triggering of this Communication Operation and
  - a. Check whether TrigTxCounter is > 0 or FrlfNoneMode == true for the PDU. If not, clear the update-bit for this PDU [Configuration Parameter FrlfPduUpdateBitOffset, see Frlf06071] and proceed with the next PDU, otherwise continue with the following steps:
    - i. Decrement TrigTxCounter only if TrigTxCounter > 0. If the value of TrigTxCounter = 0, do not decrement.
    - ii. Call the upper layer's function <UL>\_TriggerTransmit() with the associated PDUId (defined by the upper layer) and pass a pointer to a temporary buffer within the Frlf that assembles the L-SDU. The pointer shall consider the byte offset [Configuration Parameter FrlfPduOffset, see Frlf06070]] of the PDU within the frame. If <UL>\_TriggerTransmit() returns E\_NOT\_OK, the TrigTxCounter value has to be rolled back to the previous value.
    - iii. Remember that a transmission for this PDU is pending if a transmission confirmation is needed for this PDU [Configuration Parameter FrlfConfirm, see Frlf06075] increment TxConfCounter, where the maximum value is limited by static configuration [Configuration Parameter FrlfCounterLimit, see Frlf06076]. If the FrlfCounterLimit has been reached, the FrlfCounterLimit value is kept and not incremented any more.
    - iv. Set the update-bit if configured for this PDU [Configuration Parameter FrlfPduUpdateBitOffset, see Frlf06071]. In case the API <UL>\_TriggerTransmit() does not return E\_OK, or the API Frlf\_CancelTransmit ()for the corresponding PDU has been called, reset the update-bit to "not updated".
2. If at least one PDU was requested for transmission or for at least one PDU FrlfNoneMode == true and <UL>\_TriggerTransmit returned E\_OK or the frame is configured to be always transmitted [Configuration Parameter FrlfAlwaysTransmit == true] then the FlexRay Driver's API service Fr\_TransmitTxLPdu() is called:
  - a. Fr\_CtrlIdx is derived according to the indexing scheme described in 7.2
  - b. Fr\_LPduIdx is set to the configured L-PDU index [Configuration Parameter

- c. Fr\_LSduPtr is set to the temporary FrLf L-SDU assembling buffer.
  - d. Fr\_LSduLength is set to the L-SDU length [Configuration Parameter FrLfLSduLength, see FrLf06054]
  - e. Fr\_SlotAssignmentPtr is set to a temporary slot assignment buffer if Bus Mirroring is enabled globally (see FrLfBusMirroringSupport), otherwise to the NULL\_PTR.
3. If Bus Mirroring is enabled globally (see FrLfBusMirroringSupport) and has been activated with a call to FrLf\_EnableBusMirroring() for the Fr\_CtrlIdx and Fr\_TransmitTxLPdu() returned E\_OK (indicating that the transmission succeeded), call Mirror\_ReportFlexRayFrame() with “controllerId” set to Fr\_CtrlIdx, “slotId”, “cycle”, and “channel” taken from Fr\_SlotAssignmentPtr, “frame” constructed from Fr\_LSduPtr and Fr\_LSduLength, and “txConflict” set to false.
  4. In case the Driver’s API Fr\_TransmitTxLPdu() returned E\_NOT\_OK (indicating that the transmission failed) changes on TrigTxCounter and TxConfCounter must be rolled back (see 4. and 5.) for each PDU contained in the FlexRay L-SDU.

All described actions in [SWS FrLf\\_05287](#) are depicted in detail in the sequence chart in chapter 9.1.2.

In case the parameter ‘FrLfAllowDynamicLSduLength’ exists and is set to TRUE for the associated frame triggering, the actual L-SDU length, that is passed to the driver (Fr\_TransmitTxLPdu()), shall be determined (i.e. shortened as much as possible) taking into account the following for those PDUs only, which have been indicated via <UL\_TriggerTransmit>():

- the position of the respective PDU within the L-SDU
- the actual length of the respective PDU as passed via <UL\_TriggerTransmit>()
- the position of the update-bit of the respective PDU (if configured)

This ensures that on one hand all the needed information for disassembling the L-SDU is available on receiver side (PDU(s) itself and the corresponding update-bit(s) if configured), and on the other hand that the payload can be reduced as much as possible by taking the position of all the required data for disassembling contained in the frame construction plan into account when shortening the L-SDU to be passed to the driver. ] ()

### 7.6.5.2 ProvideTxConfirmation

This Communication Operation provides a Tx confirmation and optionally checks the occurrence of a Tx conflict.

**[SWS\_FrLf\_05064]** [If the related CC is in FrLf\_State FRIF\_STATE\_ONLINE for a Communication Operation TX\_CONFIRMATION, then the Job List Execution

Function shall execute this Communication Operation. Otherwise, the Job List Execution Function shall ignore this Communication Operation. ] ()

**[SWS\_Frlf\_05288]** [ “For a Communication Operation TX\_CONFIRMATION the Job List Execution Function shall perform the following steps:

1. Call the FlexRay Driver’s API function Fr\_CheckTxLPduStatus():
  - a. Fr\_CtrlIdx is derived according to the indexing scheme described in 7.2
  - b. Fr\_LPduldx is set to the configured L-PDU buffer index [Configuration Parameter FrlfLPduldx, see [Frlf06058](#)] associated with the Communication Operation.
  - c. Fr\_SlotAssignmentPtr is set to a temporary slot assignment buffer if Bus Mirroring is enabled globally (see FrlfBusMirroringSupport), otherwise to the NULL\_PTR.
2. If the transmission was performed (output parameter \*Fr\_TxLPduStatusPtr is set to FR\_TRANSMITTED) then iterate over all PDUs contained in the FrlfFrameStructure (see [Frlf05370](#)) of the associated frame triggering. If [TxConfCounter](#) for a PDU is 0 proceed with the next PDU, otherwise
  - a. If FrlfConfirm == true, call the upper layer’s function <UL\_TxConfirmation(E\_OK)> with the associated PDUId (defined by the upper layer).
  - b. If FrlfConfirm == true , decrement [TxConfCounter](#).
3. If the transmission was performed but a TxConflict occurred (output parameter \*Fr\_TxLPduStatusPtr is set to FR\_TRANSMITTED\_CONFLICT) then iterate over all PDUs contained in the FrlfFrameStructure (see [Frlf05370](#)) of the associated frame triggering. If [TxConfCounter](#) for a PDU is 0 proceed with the next PDU, otherwise
  - a. If FrlfConfirm == true, call the upper layer’s function <UL\_TxConfirmation(E\_NOT\_OK)> with the associated PDUId (defined by the upper layer).
  - b. If FrlfConfirm == true , decrement [TxConfCounter](#).
4. If Bus Mirroring is enabled globally (see FrlfBusMirroringSupport) and has been activated with a call to Frlf\_EnableBusMirroring() for the Fr\_CtrlIdx and the API Fr\_CheckTxLpduStatus() returns "FR\_TRANSMITTED\_CONFLICT", call Mirror\_ReportFlexRayFrame() with “controllerId” set to Fr\_CtrlIdx, “slotId”, “cycle”, and “channel” taken from Fr\_SlotAssignmentPtr, “frame” set to the NULL\_PTR, and “txConflict” set to true.
5. If the API Fr\_CheckTxLpduStatus() returns "FR\_TRANSMITTED\_CONFLICT" and the <UL\_TxConflictNotification> is configured via FrlfTxConflictNotificationName (ECUC\_Frlf\_06122), call this function for the same LPduldx. ] ()

### 7.6.5.3 ReceiveAndStore

**[SWS\_Frlf\_05289]** [ If the related CC is in Frlf\_State FRIF\_STATE\_ONLINE for a Communication Operation RECEIVE\_AND\_STORE, then the Job List Execution Function shall execute this Communication Operation. Otherwise, the Job List Execution Function shall ignore this Communication Operation. ] ()



**[SWS\_Frlf\_05290]** ¶ For a Communication Operation RECEIVE\_AND\_STORE the Job List Execution Function shall perform the following steps:

1. Call the FlexRay Driver's API function Fr\_ReceiveRxLPdu():
  - a. Fr\_CtrlIdx is derived according to the indexing scheme described in 7.2
  - b. Fr\_LPduldx is set to the configured L-PDU index [Configuration Parameter FrlfLPduldx, see [Frlf06058](#)] associated with the Communication Operation.
  - c. Fr\_LSduPtr is set to a temporary buffer.
  - d. Fr\_SlotAssignmentPtr is set to a temporary slot assignment buffer if Bus Mirroring is enabled globally (see FrlfBusMirroringSupport), otherwise to the NULL\_PTR.
2. If Bus Mirroring is enabled globally (see FrlfBusMirroringSupport) and has been activated with a call to Frlf\_EnableBusMirroring() for the Fr\_CtrlIdx and an L-PDU was received (Output parameter \*Fr\_LPduStatusPtr != FR\_NOT\_RECEIVED), call Mirror\_ReportFlexRayFrame() with "controllerId" set to Fr\_CtrlIdx, "slotId", "cycle", and "channel" taken from Fr\_SlotAssignmentPtr, "frame" constructed from Fr\_LSduPtr and Fr\_LSduLengthPtr, and "txConflict" set to false.
3. If a L-PDU was received (Output parameter \*Fr\_LPduStatusPtr != FR\_NOT\_RECEIVED) iterate over all PDUs contained in the FrlfFrameStructure (see [Frlf05370](#)) of the associated frame triggering and:
  - a. If an update bit was configured for the PDU [Configuration Parameter FrlfPduUpdateBitOffset, see [Frlf06071](#)] and the update bit for the PDU is not set, continue with the next PDU. Otherwise,
  - b. Copy the PDU Payload from the temporary buffer considering the PDU offset within the L-SDU [Configuration Parameter FrlfPduOffset, see [Frlf06070](#)] into a Frlf PDU-related static buffer.
  - c. Store the actual received PDU length
  - d. Mark the PDU-related static buffer as up-to-date.
4. if \*Fr\_LPduStatusPtr == FR\_RECEIVED\_MORE\_DATA\_AVAILABLE restart at number 1 again. Otherwise the communication operation has finished. ] ()

#### 7.6.5.4 ProvideRxIndication

**[SWS\_Frlf\_05062]** ¶ If the related CC is in Frlf\_State FRIF\_STATE\_ONLINE for a Communication Operation RX\_INDICATION, then the Job List Execution Function shall execute this Communication Operation. Otherwise, the Job List Execution Function shall ignore this Communication Operation. ] (SRS\_Fr\_05170)

**[SWS\_Frlf\_05291]** ¶ For a Communication Operation RX\_INDICATION the Job List Execution Function shall perform the following steps:

1. Iterate over all PDU-related static buffers of PDUs contained in the FrlfFrameStructure (see [Frlf05370](#)) of the associated frame triggering
2. If the PDU-related static buffer is marked as outdated, continue with the next PDU. Otherwise if the buffer is marked up-to-date,

- a. Call the upper layer's function <UL>\_RxIndication() with the PDU Id the receiving module expects and PduInfoPtr which contains the received data address and received data length.
- b. Mark the PDU-related static buffer as outdated. ] ()

### 7.6.5.5 ReceiveAndIndicate

**[SWS\_FrIf\_05292]** [If the related CC is in FrIf\_State FRIF\_STATE\_ONLINE for a Communication Operation RECEIVE\_AND\_INDICATE, then the Job List Execution Function shall execute this Communication Operation. Otherwise, the Job List Execution Function shall ignore this Communication Operation. ] ()

**[SWS\_FrIf\_05293]** [For a Communication Operation RECEIVE\_AND\_INDICATE the Job List Execution Function shall perform the following steps:

- 1) Calculate values for input parameters:
  - a) Fr\_CtrlIdx is derived according to the indexing scheme described in 7.2
  - b) Fr\_LPdulIdx is set to the configured L-PDU index [Configuration Parameter FrIfLPdulIdx, see FrIf06058] associated with the Communication Operation.
  - c) Fr\_LSduPtr is set to a temporary buffer.
  - d) Fr\_SlotAssignmentPtr is set to a temporary slot assignment buffer if Bus Mirroring is enabled globally (see FrIfBusMirroringSupport), otherwise to the NULL\_PTR.
  
- 2) Initialize ComOpLoopCounter to 0.
  
- 3) As long as ComOpLoopCounter < FrIfRxComOpMaxLoop do
  - a) Call Fr\_ReceiveRxLPdu with the parameters calculated in 1)
  - b) If \*Fr\_LPduStatusPtr != FR\_NOT\_RECEIVED then continue at 3)c), otherwise the communication operation has finished.
  - c) If Bus Mirroring is enabled globally (see FrIfBusMirroringSupport) and has been activated with a call to FrIf\_EnableBusMirroring() for the Fr\_CtrlIdx, call Mirror\_ReportFlexRayFrame() with "controllerId" set to Fr\_CtrlIdx, "slotId", "cycle", and "channel" taken from Fr\_SlotAssignmentPtr, "frame" constructed from Fr\_LSduPtr and Fr\_LSduLengthPtr, and "txConflict" set to false. Otherwise, continue at 3)d).
  
- d) For each Pdu contained in the FrIfFrameStructure (see FrIf05370) of the associated frame triggering do
  - ) If an update bit was configured for the PDU [Configuration Parameter FrIfPduUpdateBitOffset, see FrIf06071] and the update bit for the PDU is not set, continue with the next PDU. Otherwise
  - ) Call the upper layer's function <UL>\_RxIndication() with the PDU Id the receiving module expects and a pointer to the Pdu-Info structure containing the Pdu length and a reference to the temporary buffer considering the PDU offset within the L-SDU [Configuration Parameter FrIfPduOffset, see FrIf06070]] as parameters.

- e) if \*Fr\_LPduStatusPtr == FR\_RECEIVED\_MORE\_DATA\_AVAILABLE then increment ComOpLoopCounter and restart at 3)a), otherwise the communication operation has finished.

] 0

### 7.6.5.6 PREPARE\_LPDU

The Communication Operation PREPARE\_LPDU enables hardware optimization purposes (hardware buffer re-configuration)

**[SWS\_FrIf\_05294]** [The Communication Operation PREPARE\_LPDU performs the following steps:

1. Call the FlexRay Driver's API function Fr\_PrepareLPdu():
  - a. Fr\_CtrlIdx is derived according to the indexing scheme described in 7.2
  - b. Fr\_LPduldx is set to the configured L-PDU index [Configuration Parameter FrIfLPduldx, see [FrIf06058](#)] associated with the Communication Operation. ] ()

**[SWS\_FrIf\_05061]** [

The Communication Operation PREPARE\_LPDU enables hardware optimization purposes. Its purpose is to enable certain FlexRay CC hardware resources (e.g. a CC's message buffer) to be prepared (configured) for the transmission/reception of a certain L-PDU.

This Communication Operation enables the FlexRay Driver to optimize the usage of hardware resources if available at appropriate point of times. However, it is the responsibility of the FlexRay Driver to decide and validate resource allocation optimizations based on the PREPARE\_LPDU Communication Operations. Practically the usage of this Communication Operation will introduce some runtime-overhead even if the FlexRay Driver does not use the opportunity for reconfiguration. ]

(SRS\_Fr\_05042)

### 7.6.3.7 FREE\_OP\_A

User-defined communication operation in order to support hardware specific or additional communication controller features to increase performance. Use cases are communication controllers with serial connection or DMA-transfers.

### 7.6.3.8 FREE\_OP\_B

User-defined communication operation in order to support hardware specific or additional communication controller features to increase performance. Use cases are communication controllers with serial connection or DMA-transfers.

## 7.6.6 Transmission with Immediate Buffer Access

### [SWS\_Frlf\_15295] ¶

The FlexRay Job List Execution Function does not initiate transmission with immediate buffer access. Instead, the actions described here are carried out in the context of the `Frlf_Transmit()` API service, which in turn is called by an upper layer [BSW](#) module. ¶ ()

[SWS\_Frlf\_05295] ¶ The FlexRay Interface shall perform a PDU transmission with immediate buffer access (see 9.1), only if the following restriction regarding static configuration apply:

- The PDU must be **the only** PDU in a FlexRay Frame (L-SDU). It is **not** packed into a FlexRay Frame together with other PDUs (i.e., the mapping between this PDU and the respective L-SDU is a 1:1 association).
- The PDU must be located **at the beginning** of the L-SDU.
- There is no update-bit for immediate PDUs configured. ¶ ()

[SWS\_Frlf\_05296] ¶ If an upper layer module calls `Frlf_Transmit()` with `TxPdul` being configured for an immediate PDU, the AUTOSAR module FlexRay Interface shall perform the following steps for an immediate PDU transmission within the context of the `Frlf_Transmit()` API service Driver's API function `Fr_TransmitTxLPdu()`:

- a. `Fr_CtrlIdx` is derived according to the indexing scheme described in 7.2
- b. `Fr_LPdulIdx` is set to the configured L-PDU index [Configuration Parameter `FrlfLPdulIdx`, see [Frlf06058](#)] associated with the `TxPdul`.
- c. `Fr_LSduPtr` is set to the Pdu Payload pointer contained in the `PdulInfoPtr` passed as parameter to `Frlf_Transmit`.
- d. If the parameter `FrlfAllowDynamicLSduLength=TRUE`, the actual length of the respective PDU shall be as passed via `Frlf_Transmit()`.
- e. `Fr_SlotAssignmentPtr` is set to a temporary slot assignment buffer if Bus Mirroring is enabled globally (see `FrlfBusMirroringSupport`), otherwise to the `NULL_PTR`.

In case the Driver's API `Fr_TransmitTxLPdu()` returned `E_OK` (indicating that the transmission request succeeded) the [TxConfCounter](#) is incremented for the respective PDU. The maximum value of [TxConfCounter](#) is limited by static configuration [Configuration Parameter `FrlfCounterLimit`, see [Frlf06076](#)]. If Bus Mirroring is enabled globally (see `FrlfBusMirroringSupport`) and has been activated with a call to `Frlf_EnableBusMirroring()` for the `Fr_CtrlIdx`, call `Mirror_ReportFlexRayFrame()` with "controllerId" set to `Fr_CtrlIdx`, "slotId", "cycle", and "channel" taken from `Fr_SlotAssignmentPtr`, "frame" constructed from `Fr_LSduPtr` and `Fr_LSduLength`, and "txConflict" set to false.

In case the Driver's API `Fr_TransmitTxLPdu()` returned `E_NOT_OK` do not modify the current counter value of [TxConfCounter](#). ¶ ()

## 7.7 Error Classification

### 7.7.1 Development Errors

[SWS\_Frlf\_05145] [

<i>Type or error</i>	<i>Related error code</i>	<i>Value [hex]</i>
Invalid pointer	FRIF_E_PARAM_POINTER	0x01
Invalid Controller index	FRIF_E_INV_CTRL_IDX	0x02
Invalid Cluster index	FRIF_E_INV_CLST_IDX	0x03
Invalid Channel index	FRIF_E_INV_CHNL_IDX	0x04
Invalid timer index	FRIF_E_INV_TIMER_IDX	0x05
Invalid Frlf_TxPdu Index	FRIF_E_INV_TXPDUID	0x06
Invalid LPdu Index	FRIF_E_INV_LPDU_IDX	0x07
Frlf not initialized	FRIF_E_UNINIT	0x08
Invalid parametFrlf state	FRIF_E_INV_FRIF_STATE	0x0A
Invalid Frame ID	FRIF_E_INV_FRAME_ID	0x0B
Initialization failed	FRIF_E_INIT_FAILED	0x0C
Invalid Pdu length	FRIF_E_INV_PDULENGTH	0x0D

Table 7-1: Definition of Development Errors

] ()

### 7.7.2 Runtime Errors

[SWS\_Frlf\_05432] [

<i>Type or error</i>	<i>Related error code</i>	<i>Value [hex]</i>
Job List Execution lost synchronization to the FlexRay Global Time	FRIF_E_JLE_SYNC	0x01

] ()

Table 7-2: Definition of Runtime Errors

### 7.7.3 Transient Faults

There are no transient faults.

### 7.7.4 Production Errors

[SWS\_Frlf\_05146] |

Type or error	Related error code	Value [hex]
error detection in NIT on channel A	FRIF_E_NIT_CH_A	Assigned by <a href="#">DEM</a>
error detection in NIT on channel B	FRIF_E_NIT_CH_B	Assigned by <a href="#">DEM</a>
error detection in SW on channel A	FRIF_E_SW_CH_A	Assigned by <a href="#">DEM</a>
error detection in SW on channel B	FRIF_E_SW_CH_B	Assigned by <a href="#">DEM</a>
error detection in ACS on channel A	FRIF_E_ACS_CH_A	Assigned by <a href="#">DEM</a>
error detection in ACS on channel B	FRIF_E_ACS_CH_B	Assigned by <a href="#">DEM</a>

| ()

Table 7-3: Definition of Production Errors

[SWS\_Frlf\_05426] |

<b>Error Name:</b>	FRIF_E_NIT_CH_A	
<b>Short Description:</b>	Error detection in NIT on channel A	
<b>Long Description:</b>	This production error shall be issued when an error in NIT on channel A was detected	
<b>Recommended DTC:</b>	N/A	
<b>Detection Criteria:</b>	Fail	The channel status information shall be reported to DEM as Dem_SetEventStatus (FRIF_E_NIT_CH_A, DEM_EVENT_STATUS_PREFAILED) when any of the error bits of a single controller (Channel A NIT status data vSS!SyntaxError, vSS!Bviolation) is set ( <a href="#">SWS_Frlf_35120</a> )
	Pass	The channel status information shall be reported to DEM as Dem_SetEventStatus (FRIF_E_NIT_CH_A, DEM_EVENT_STATUS_PREPASSED) when none of the error bits of a single controller (Channel A NIT status data vSS!SyntaxError, vSS!Bviolation) is set ( <a href="#">SWS_Frlf_35120</a> )
<b>Secondary Parameters:</b>	N/A	
<b>Time Required:</b>	N/A	
<b>Monitor Frequency</b>	continuous	
<b>MIL illumination:</b>	N/A	

| ()

[SWS\_Frlf\_05427] |

<b>Error Name:</b>	FRIF_E_NIT_CH_B
<b>Short Description:</b>	Error detection in NIT on channel B
<b>Long Description:</b>	This production error shall be issued when an error in NIT on

	channel B was detected	
<b>Recommended DTC:</b>	N/A	
<b>Detection Criteria:</b>	Fail	The channel status information shall be reported to DEM as Dem_SetEventStatus (FRIF_E_NIT_CH_B, DEM_EVENT_STATUS_PREFAILED) when any of the error bits of a single controller (Channel B NIT status data vSS!SyntaxError, vSS!Bviolation) is set ( <a href="#">SWS FrIf_45120</a> )
	Pass	The channel status information shall be reported to DEM as Dem_SetEventStatus (FRIF_E_NIT_CH_B, DEM_EVENT_STATUS_PREPASSED) when none of the error bits of a single controller (Channel B NIT status data vSS!SyntaxError, vSS!Bviolation) is set ( <a href="#">SWS FrIf_45120</a> )
<b>Secondary Parameters:</b>	N/A	
<b>Time Required:</b>	N/A	
<b>Monitor Frequency</b>	continuous	
<b>MIL illumination:</b>	N/A	

] ()

[SWS\_FrIf\_05428]

<b>Error Name:</b>	FRIF_E_SW_CH_A	
<b>Short Description:</b>	Error detection in SW on channel A	
<b>Long Description:</b>	This production error shall be issued when an error in SW on channel A was detected.	
<b>Recommended DTC:</b>	N/A	
<b>Detection Criteria:</b>	Fail	The channel status information shall be reported to DEM as Dem_SetEventStatus (FRIF_E_SW_CH_A, DEM_EVENT_STATUS_PREFAILED) when any of the error bits of a single controller (Channel A symbol window status data vSS!SyntaxError, vSS!Bviolation, vSS!TxConflict) is set ( <a href="#">SWS FrIf_55120</a> )
	Pass	The channel status information shall be reported to DEM as Dem_SetEventStatus (FRIF_E_SW_CH_A, DEM_EVENT_STATUS_PREPASSED) when none of the error bits of a single controller (Channel A symbol window status data vSS!SyntaxError, vSS!Bviolation, vSS!TxConflict) is set ( <a href="#">SWS FrIf_55120</a> )
<b>Secondary Parameters:</b>	N/A	
<b>Time Required:</b>	N/A	
<b>Monitor Frequency</b>	continuous	
<b>MIL illumination:</b>	N/A	

] ()



[SWS\_Frlf\_05429]

<b>Error Name:</b>	FRIF_E_SW_CH_B	
<b>Short Description:</b>	Error detection in SW on channel B	
<b>Long Description:</b>	This production error shall be issued when an error in SW on channel B was detected.	
<b>Recommended DTC:</b>	N/A	
<b>Detection Criteria:</b>	Fail	The channel status information shall be reported to DEM as Dem_SetEventStatus (FRIF_E_SW_CH_B, DEM_EVENT_STATUS_PREFAILED) when any of the error bits of a single controller (Channel B symbol window status data vSS!SyntaxError, vSS!Bviolation, vSS!TxConflict) is set ( <a href="#">SWS Frlf 65120</a> )
	Pass	The channel status information shall be reported to DEM as Dem_SetEventStatus (FRIF_E_SW_CH_B, DEM_EVENT_STATUS_PREPASSED) when none of the error bits of a single controller (Channel B symbol window status data vSS!SyntaxError, vSS!Bviolation, vSS!TxConflict) is set ( <a href="#">SWS Frlf 65120</a> )
<b>Secondary Parameters:</b>	N/A	
<b>Time Required:</b>	N/A	
<b>Monitor Frequency</b>	continuous	
<b>MIL illumination:</b>	N/A	

] ()

[SWS\_Frlf\_05431]

<b>Error Name:</b>	FRIF_E_ACS_CH_A	
<b>Short Description:</b>	Error detection in ACS on channel A	
<b>Long Description:</b>	This production error shall be issued when an error in ACS on channel A was detected	
<b>Recommended DTC:</b>	N/A	
<b>Detection Criteria:</b>	Fail	The channel status information shall be reported to DEM as Dem_SetEventStatus (FRIF_E_ACS_CH_A , DEM_EVENT_STATUS_PREFAILED) when any of the error bits of a single controller (Channel A aggregated channel status vSS!SyntaxError, vSS!ContentError, vSS!Bviolation, vSS!TxConflict) is set ( <a href="#">SWS Frlf 75120</a> )
	Pass	The channel status information shall be reported to DEM as Dem_SetEventStatus (FRIF_E_ACS_CH_A , DEM_EVENT_STATUS_PREPASSED) when none of the error bits of a single controller (Channel A aggregated channel status

		vSS!SyntaxError, vSS!ContentError, vSS!Bviolation, vSS!TxConflict) is set ( <a href="#">SWS Frlf 75120</a> )
<b>Secondary Parameters:</b>	N/A	
<b>Time Required:</b>	N/A	
<b>Monitor Frequency</b>	continuous	
<b>MIL illumination:</b>	N/A	

] ()

[SWS\_Frlf\_05430]

<b>Error Name:</b>	FRIF_E_ACS_CH_B	
<b>Short Description:</b>	Error detection in ACS on channel B	
<b>Long Description:</b>	This production error shall be issued when an error in ACS on channel B was detected	
<b>Recommended DTC:</b>	N/A	
<b>Detection Criteria:</b>	Fail	The channel status information shall be reported to DEM as Dem_SetEventStatus (FRIF_E_SW_CH_B, DEM_EVENT_STATUS_PREFAILED) when any of the error bits of a single controller (Channel B symbol window status data vSS!SyntaxError, vSS!Bviolation, vSS!TxConflict) is set ( <a href="#">SWS Frlf 85120</a> )
	Pass	The channel status information shall be reported to DEM as Dem_SetEventStatus (FRIF_E_ACS_CH_B, DEM_EVENT_STATUS_PREPASSED) when none of the error bits of a single controller (Channel B aggregated channel status vSS!SyntaxError, vSS!ContentError, vSS!Bviolation, vSS!TxConflict) is set ( <a href="#">SWS Frlf 85120</a> )
<b>Secondary Parameters:</b>	N/A	
<b>Time Required:</b>	N/A	
<b>Monitor Frequency</b>	continuous	
<b>MIL illumination:</b>	N/A	

] ()

### 7.7.5 Extended Production Errors

There are no extended production errors.

## 8 API Service Specification

### 8.1 Imported types

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

[SWS\_Frlf\_05001] [

<b>Module</b>	<b>Header File</b>	<b>Imported Type</b>
ComStack_Types	ComStackTypes.h	PdulIdType
	ComStackTypes.h	PdulInfoType
Dem	Rte_Dem_Type.h	Dem_EventIdType
	Rte_Dem_Type.h	Dem_EventStatusType
Fr	Fr_GeneralTypes.h	Fr_ChannelType
	Fr_GeneralTypes.h	Fr_POCSStatusType
	Fr_GeneralTypes.h	Fr_RxLPduStatusType
	Fr_GeneralTypes.h	Fr_SlotAssignmentType
	Fr_GeneralTypes.h	Fr_TxLPduStatusType
FrTrcv	Fr_GeneralTypes.h	FrTrcv_TrvcModeType
	Fr_GeneralTypes.h	FrTrcv_TrvcWUReasonType
Std_Types	StandardTypes.h	Std_ReturnType
	StandardTypes.h	Std_VersionInfoType

] (SRS\_BSW\_00348, SRS\_BSW\_00353, SRS\_BSW\_00361, SRS\_BSW\_00304, SRS\_BSW\_00378)

### 8.2 Type Definitions

This chapter lists the data types that the FlexRay Interface defines.

#### 8.2.1 Frlf\_ConfigType

[SWS\_Frlf\_05301] [

<b>Name:</b>	Frlf_ConfigType	
<b>Type:</b>	Structure	
<b>Range:</b>	Implementation specific	--
<b>Description:</b>	This type contains the implementation-specific post build time configuration structure. Only pointers of this type are allowed.	
<b>Available via:</b>	Frlf.h	

] ()

### 8.2.2 FrIf\_StateType

[SWS\_FrIf\_05755] [

<b>Name:</b>	FrIf_StateType	
<b>Type:</b>	Enumeration	
<b>Range:</b>	FRIF_STATE_OFFLINE	-- The FlexRay CC is not ready for communication, the FlexRay cluster is not synchronized.
	FRIF_STATE_ONLINE	-- The FlexRay CC is ready for communication, the FlexRay cluster is synchronized.
<b>Description:</b>	Variables of this type are used to represent the FrIf_State of a FlexRay CC.	
<b>Available via:</b>	FrIf.h	

] ()

### 8.2.3 FrIf\_StateTransitionType

[SWS\_FrIf\_05303] [

<b>Name:</b>	FrIf_StateTransitionType	
<b>Type:</b>	Enumeration	
<b>Range:</b>	FRIF_GOTO_OFFLINE	-- Literal for requesting transition into FRIF_STATE_OFFLINE
	FRIF_GOTO_ONLINE	-- Literal for requesting transition into FRIF_STATE_ONLINE state.
<b>Description:</b>	Variables of this type are used to represent the FrIf_State of a FlexRay CC.	
<b>Available via:</b>	FrIf.h	

] ()

### 8.3 Function Definitions

This is a list of API services (functions) the [Frlf](#) module provides to upper layer [BSW](#) modules.

#### 8.3.1 Frlf\_Init

**[SWS\_Frlf\_05003]** [

<b>Service name:</b>	Frlf_Init
<b>Syntax:</b>	void Frlf_Init( const Frlf_ConfigType* Frlf_ConfigPtr )
<b>Service ID[hex]:</b>	0x02
<b>Sync/Async:</b>	Synchronous
<b>Reentrancy:</b>	Non Reentrant
<b>Parameters (in):</b>	Frlf_ConfigPtr   Base pointer to the configuration structure of the FlexRay Interface.
<b>Parameters (inout):</b>	None
<b>Parameters (out):</b>	None
<b>Return value:</b>	void   --
<b>Description:</b>	Initializes the FlexRay Interface.
<b>Available via:</b>	Frlf.h

] (SRS\_BSW\_00405, SRS\_BSW\_00101, SRS\_BSW\_00358, SRS\_BSW\_00414, SRS\_Fr\_05013)

**Note:**

The AUTOSAR ECU StateManager calls this FlexRay Interface API service with the address of the static configuration structure of the [Frlf](#) module in parameter Frlf\_ConfigPtr.

**[SWS\_Frlf\_05156]** [The function Frlf\_Init shall carry out the following actions:

- 1) Configure the FlexRay Interface module: initialize the local memory space used to store the PDU data and the PDU properties and state variables and the FlexRay Interface State Machine.
- 2) The initialization of the memory space has to make sure that the PDU-related static buffer status is set to "outdated"] ()

### 8.3.2 FrIf\_ControllerInit

**[SWS\_FrIf\_05004]** [

<b>Service name:</b>	FrIf_ControllerInit	
<b>Syntax:</b>	Std_ReturnType FrIf_ControllerInit( uint8 FrIf_CtrlIdx )	
<b>Service ID[hex]:</b>	0x03	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	non reentrant for identical values of FrIf_CtrlIdx, reentrant for different values of FrIf_CtrlIdx	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of the FlexRay CC to address.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: The call of the FlexRay Driver's API service has returned E_OK. E_NOT_OK: The call of the FlexRay Driver's API service has returned E_NOT_OK, or an error has been detected in development mode.
<b>Description:</b>	Initialized a FlexRay CC.	
<b>Available via:</b>	FrIf.h	

] (SRS\_Fr\_05031)

**[SWS\_FrIf\_05158]** [If parameter FrIf\_CtrlIdx of FrIf\_ControllerInit has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_ControllerInit shall report development error code FRIF\_E\_INV\_CTRL\_IDX to the Det\_ReportError service of the DET module. ] ()

**[SWS\_FrIf\_05159]** [The function FrIf\_ControllerInit shall wrap the FlexRay Driver API function Fr\_ControllerInit() by:

- 1) Translating (based on static [FrIf](#) module configuration) the FlexRay [CC](#) index FrIf\_CtrlIdx into a tuple (FlexRay Driver | Driver-specific [CC](#) index Fr\_CtrlIdx).
- 2) Calling Fr\_ControllerInit() of the determined FlexRay Driver module with the parameters determined as described above. ] ()

**[SWS\_FrIf\_05160]** [Caveats of FrIf\_ControllerInit: The FlexRay Interface module has to be initialized with a call of FrIf\_Init() before this API service may be called, see SWS\_FrIf\_05003] ()

### 8.3.3 FrIf\_SetAbsoluteTimer

**[SWS\_FrIf\_05021]** [

<b>Service name:</b>	FrIf_SetAbsoluteTimer
----------------------	-----------------------

<b>Syntax:</b>	Std_ReturnType FrIf_SetAbsoluteTimer ( uint8 FrIf_CtrlIdx, uint8 FrIf_AbsTimerIdx, uint8 FrIf_Cycle, uint16 FrIf_Offset )	
<b>Service ID[hex]:</b>	0x19	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	non reentrant for the same FlexRay CC, reentrant for different FlexRay CCs	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of the FlexRay CC to address.
	FrIf_AbsTimerIdx	Index of the absolute timer to address.
	FrIf_Cycle	FlexRay Cycle number to be set.
	FrIf_Offset	Number of Macroticks to be set.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: The call of the FlexRay Driver's API service has returned E_OK. E_NOT_OK: The call of the FlexRay Driver's API service has returned E_NOT_OK, or an error has been detected in development mode.
<b>Description:</b>	Wraps the FlexRay Driver API function Fr_SetAbsoluteTimer().	
<b>Available via:</b>	FrIf.h	

] ()

**[SWS\_FrIf\_05234]** [If parameter FrIf\_CtrlIdx of FrIf\_SetAbsoluteTimer has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_SetAbsoluteTimer shall report development error code FRIF\_E\_INV\_CTRL\_IDX to the Det\_ReportError service of the DET module. ] ()

**[SWS\_FrIf\_05235]** [The function FrIf\_SetAbsoluteTimer shall wrap This API service of the FlexRay Interface wraps the FlexRay Driver API function Fr\_SetAbsoluteTimer() by:

- 1) Translating (based on static FrIf module configuration) the FlexRay CC index FrIf\_CtrlIdx into a tuple (FlexRay Driver | Driver-specific CC index Fr\_CtrlIdx).
- 2) Setting parameters
- 3) Fr\_AbsTimerIdx to FrIf\_AbsTimerIdx
- 4) Fr\_Cycle to FrIf\_Cycle
- 5) Fr\_Offset to FrIf\_Offset
- 6) Calling Fr\_SetAbsoluteTimer() of the determined FlexRay Driver module with the parameters determined as described above. ] ()

**[SWS\_FrIf\_05236]** [Caveats of FrIf\_SetAbsoluteTimer: The FlexRay Interface module has to be initialized with a call of FrIf\_Init() before this API service may be called, see SWS\_FrIf\_05003. ] ()

### 8.3.4 FrIf\_EnableAbsoluteTimerIRQ

#### [SWS\_FrIf\_05025] |

<b>Service name:</b>	FrIf_EnableAbsoluteTimerIRQ	
<b>Syntax:</b>	<pre>Std_ReturnType FrIf_EnableAbsoluteTimerIRQ(     uint8 FrIf_CtrlIdx,     uint8 FrIf_AbsTimerIdx )</pre>	
<b>Service ID[hex]:</b>	0x1d	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	non reentrant for the same FlexRay CC, reentrant for different FlexRay CCs	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of the FlexRay CC to address.
	FrIf_AbsTimerIdx	Index of the absolute timer to address.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: The call of the FlexRay Driver's API service has returned E_OK. E_NOT_OK: The call of the FlexRay Driver's API service has returned E_NOT_OK, or an error has been detected in development mode.
<b>Description:</b>	Wraps the FlexRay Driver API function Fr_EnableAbsoluteTimerIRQ().	
<b>Available via:</b>	FrIf.h	

| ()

**[SWS\_FrIf\_05246]** | If parameter FrIf\_CtrlIdx of FrIf\_EnableAbsoluteTimerIRQ has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_EnableAbsoluteTimerIRQ shall report development error code FRIF\_E\_INV\_CTRL\_IDX to the Det\_ReportError service of the DET module. | ()

**[SWS\_FrIf\_05247]** | The function FrIf\_EnableAbsoluteTimerIRQ shall wrap the FlexRay Driver API function Fr\_EnableAbsoluteTimerIRQ() by:

1. Translating (based on static FrIf module configuration) the FlexRay CC index FrIf\_CtrlIdx into a tuple (FlexRay Driver | Driver-specific CC index Fr\_CtrlIdx).
2. Setting parameters
  - Fr\_AbsTimerIdx to FrIf\_AbsTimerIdx
3. Calling Fr\_EnableAbsoluteTimerIRQ() of the determined FlexRay Driver module with the parameters determined as described above. | ()

**[SWS\_FrIf\_05248]** | Caveats of FrIf\_EnableAbsoluteTimerIRQ: The FlexRay Interface module has to be initialized with a call of FrIf\_Init() before this API service may be called, see SWS\_FrIf\_05003. | ()

### 8.3.5 FrIf\_AckAbsoluteTimerIRQ

#### [SWS\_FrIf\_05029] |



<b>Service name:</b>	FrIf_AckAbsoluteTimerIRQ	
<b>Syntax:</b>	Std_ReturnType FrIf_AckAbsoluteTimerIRQ( uint8 FrIf_CtrlIdx, uint8 FrIf_AbsTimerIdx )	
<b>Service ID[hex]:</b>	0x21	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	non reentrant for the same FlexRay CC, reentrant for different FlexRay CCs	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of the FlexRay CC to address.
	FrIf_AbsTimerIdx	Index of the absolute timer to address.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: The call of the FlexRay Driver's API service has returned E_OK. E_NOT_OK: The call of the FlexRay Driver's API service has returned E_NOT_OK, or an error has been detected in development mode.
<b>Description:</b>	Wraps the FlexRay Driver API function Fr_AckAbsoluteTimerIRQ()	
<b>Available via:</b>	FrIf.h	

] ()

**[SWS\_FrIf\_05258]** [If parameter FrIf\_CtrlIdx of FrIf\_AckAbsoluteTimerIRQ has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_AckAbsoluteTimerIRQ shall report development error code FRIF\_E\_INV\_CTRL\_IDX to the Det\_ReportError service of the DET module. ] ()

**[SWS\_FrIf\_05259]** [The function FrIf\_AckAbsoluteTimerIRQ shall wrap the FlexRay Driver API function Fr\_AckAbsoluteTimerIRQ() by:

1. Translating (based on static FrIf module configuration) the FlexRay CC index FrIf\_CtrlIdx into a tuple (FlexRay Driver | Driver-specific CC index Fr\_CtrlIdx).
2. Setting parameters
  - Fr\_AbsTimerIdx to FrIf\_AbsTimerIdx
3. Calling Fr\_AckAbsoluteTimerIRQ() of the determined FlexRay Driver module with the parameters determined as described above. ] ()

**[SWS\_FrIf\_05260]** [Caveats of FrIf\_AckAbsoluteTimerIRQ: The FlexRay Interface module has to be initialized with a call of FrIf\_Init() before this API service may be called, see SWS\_FrIf\_05003. ] ()

### 8.3.6 FrIf\_StartCommunication

**[SWS\_FrIf\_05005]** [

<b>Service name:</b>	FrIf_StartCommunication	
<b>Syntax:</b>	Std_ReturnType FrIf_StartCommunication( uint8 FrIf_CtrlIdx )	
<b>Service ID[hex]:</b>	0x04	

<b>Sync/Async:</b>	Asynchronous	
<b>Reentrancy:</b>	non reentrant for identical values of FrIf_CtrlIdx, reentrant for different values of FrIf_CtrlIdx	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of the FlexRay CC to address.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: The call of the FlexRay Driver's API service has returned E_OK. E_NOT_OK: The call of the FlexRay Driver's API service has returned E_NOT_OK, or an error has been detected in development mode.
<b>Description:</b>	Wraps the FlexRay Driver API function Fr_StartCommunication().	
<b>Available via:</b>	FrIf.h	

] (SRS\_Fr\_05015)

**[SWS\_FrIf\_05161]** [If parameter FrIf\_CtrlIdx of FrIf\_StartCommunication has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_StartCommunication shall report development error code FRIF\_E\_INV\_CTRL\_IDX to the Det\_ReportError service of the DET module. ] ()

**[SWS\_FrIf\_05162]** [The function FrIf\_StartCommunication shall wrap the FlexRay Driver API function Fr\_StartCommunication() by:  
-1) Translating (based on static FrIf module configuration) the FlexRay CC index FrIf\_CtrlIdx into a tuple (FlexRay Driver | Driver-specific CC index Fr\_CtrlIdx).  
-2) Calling Fr\_StartCommunication() of the determined FlexRay Driver module with the parameters determined as described above. ] ()

**[SWS\_FrIf\_05163]** [Caveats of FrIf\_StartCommunication: The FlexRay Interface module has to be initialized with a call of FrIf\_Init() before this API service may be called, see SWS\_FrIf\_05003] ()

### 8.3.7 FrIf\_HaltCommunication

**[SWS\_FrIf\_05006]** [

<b>Service name:</b>	FrIf_HaltCommunication	
<b>Syntax:</b>	Std_ReturnType FrIf_HaltCommunication( uint8 FrIf_CtrlIdx )	
<b>Service ID[hex]:</b>	0x05	
<b>Sync/Async:</b>	Asynchronous	
<b>Reentrancy:</b>	non reentrant for identical values of FrIf_CtrlIdx, reentrant for different values of FrIf_CtrlIdx	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of the FlexRay CC to address.
<b>Parameters (inout):</b>	None	

<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: The call of the FlexRay Driver's API service has returned E_OK. E_NOT_OK: The call of the FlexRay Driver's API service has returned E_NOT_OK, or an error has been detected in development mode.
<b>Description:</b>	Wraps the FlexRay Driver API function Fr_HaltCommunication().	
<b>Available via:</b>	FrIf.h	

] (SRS\_BSW\_00336, SRS\_Fr\_05063)

**[SWS\_FrIf\_05164]** ] If parameter FrIf\_CtrlIdx of FrIf\_HaltCommunication has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_HaltCommunication shall report development error code FRIF\_E\_INV\_CTRL\_IDX to the Det\_ReportError service of the DET module. ] ()

**[SWS\_FrIf\_05165]** ] The function FrIf\_HaltCommunication shall wrap the FlexRay Driver API function Fr\_HaltCommunication() by:

- 1) Translating (based on static FrIf module configuration) the FlexRay CC index FrIf\_CtrlIdx into a tuple (FlexRay Driver | Driver-specific CC index Fr\_CtrlIdx).
- 2) Calling Fr\_HaltCommunication() of the determined FlexRay Driver module with the parameters determined as described above. ] ()

**[SWS\_FrIf\_05166]** ] Caveats of FrIf\_HaltCommunication: The FlexRay Interface module has to be initialized with a call of FrIf\_Init() before this API service may be called, see SWS\_FrIf\_05003] ()

### 8.3.8 FrIf\_AbortCommunication

**[SWS\_FrIf\_05007]** ]

<b>Service name:</b>	FrIf_AbortCommunication	
<b>Syntax:</b>	Std_ReturnType FrIf_AbortCommunication( uint8 FrIf_CtrlIdx )	
<b>Service ID[hex]:</b>	0x06	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	non reentrant for identical values of FrIf_CtrlIdx, reentrant for different values of FrIf_CtrlIdx	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of the FlexRay CC to address.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: The call of the FlexRay Driver's API service has returned E_OK. E_NOT_OK: The call of the FlexRay Driver's API service has returned E_NOT_OK, or an error has been detected in

	development mode.
<b>Description:</b>	Wraps the FlexRay Driver API function Fr_AbortCommunication().
<b>Available via:</b>	FrIf.h

] (SRS\_Fr\_05016)

**[SWS\_Frlf\_05167]** [If parameter Frlf\_CtrlIdx of Frlf\_AbortCommunication has an invalid value and if development error detection is enabled (i.e. FrlfDevErrorDetect equals ON), the function Frlf\_AbortCommunication shall report development error code FRIF\_E\_INV\_CTRL\_IDX to the Det\_ReportError service of the DET module. ] ()

**[SWS\_Frlf\_05168]** [The function Frlf\_AbortCommunication shall wrap the FlexRay Driver API function Fr\_AbortCommunication() by:

- 1) Translating (based on static Frlf module configuration) the FlexRay CC index Frlf\_CtrlIdx into a tuple (FlexRay Driver | Driver-specific CC index Fr\_CtrlIdx).
- 2) Calling Fr\_AbortCommunication() of the determined FlexRay Driver module with the parameters determined as described above. ] ()

**[SWS\_Frlf\_05169]** [Caveats of Frlf\_AbortCommunication: The FlexRay Interface module has to be initialized with a call of Frlf\_Init() before this API service may be called, see SWS\_Frlf\_05003] ()

### 8.3.9 Frlf\_GetState

**[SWS\_Frlf\_05170]** [

<b>Service name:</b>	Frlf_GetState	
<b>Syntax:</b>	Std_ReturnType Frlf_GetState( uint8 Frlf_ClstIdx, Frlf_StateType* Frlf_StatePtr )	
<b>Service ID[hex]:</b>	0x07	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant	
<b>Parameters (in):</b>	Frlf_ClstIdx	Index of the cluster addressed.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	Frlf_StatePtr	Pointer to a memory location where the retrieved FrlfState will be stored
<b>Return value:</b>	Std_ReturnType	E_OK: Function was successfully executed. State transition request was accepted. E_NOT_OK: Function execution failed due to detected errors. State transition request was not accepted.
<b>Description:</b>	Get current Frlf state.	
<b>Available via:</b>	FrIf.h	

] ()

**[SWS\_FrIf\_05171]** ¶ If parameter `FrIf_ClstIdx` of `FrIf_GetState` has an invalid value and if development error detection is enabled (i.e. `FrIfDevErrorDetect` equals ON), the function `FrIf_GetState` shall report development error code `FRIF_E_INV_CLST_IDX` to the `Det_ReportError` service of the DET module. ] ()

**[SWS\_FrIf\_05172]** ¶ If parameter `FrIf_StatePtr` of `FrIf_GetState` equals `NULL_PTR` and if development error detection is enabled (i.e. `FrIfDevErrorDetect` equals ON), the function `FrIf_GetState` shall report development error code `FRIF_E_PARAM_POINTER` to the `Det_ReportError` service of the DET module. ] ()

**[SWS\_FrIf\_05173]** ¶ Caveats of `FrIf_GetState`: The FlexRay Interface module has to be initialized with a call of `FrIf_Init()` before this API service may be called, see `SWS_FrIf_05003`] ()

### 8.3.10 `FrIf_SetState`

**[SWS\_FrIf\_05174]** [

<b>Service name:</b>	<code>FrIf_SetState</code>	
<b>Syntax:</b>	<pre>Std_ReturnType FrIf_SetState(     uint8 FrIf_ClstIdx,     FrIf_StateTransitionType FrIf_StateTransition )</pre>	
<b>Service ID[hex]:</b>	0x08	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	<code>FrIf_ClstIdx</code>	Index of the cluster addressed.
	<code>FrIf_StateTransition</code>	Requested <code>FrIf</code> state transition.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	<code>Std_ReturnType</code>	<code>E_OK</code> : Function was successfully executed. State transition request was accepted. <code>E_NOT_OK</code> : Function execution failed due to detected errors. State transition request was not accepted.
	<b>Description:</b> Requests <code>FrIf</code> state machine transition.	
<b>Available via:</b>	<code>FrIf.h</code>	

] ()

**[SWS\_FrIf\_05175]** ¶ If parameter `FrIf_ClstIdx` of `FrIf_SetState` has an invalid value and if development error detection is enabled (i.e. `FrIfDevErrorDetect` equals ON), the function `FrIf_SetState` shall report development error code `FRIF_E_INV_CLST_IDX` to the `Det_ReportError` service of the DET module. ] ()

**[SWS\_FrIf\_05037]** ¶ If parameter `FrIf_StateTransition` of `FrIf_SetState` has an invalid value and if development error detection is enabled (i.e. `FrIfDevErrorDetect` equals ON), the function `FrIf_SetState` shall report development error code `FRIF_E_INV_FRIF_STATE` to the `Det_ReportError` service of the DET module.] ()

[SWS\_Frlf\_05176] [Caveats of Frlf\_SetState: The FlexRay Interface module has to be initialized with a call of Frlf\_Init() before this API service may be called, see SWS\_Frlf\_05003] ()

### 8.3.11 Frlf\_SetWakeupChannel

[SWS\_Frlf\_05010] [

<b>Service name:</b>	Frlf_SetWakeupChannel	
<b>Syntax:</b>	Std_ReturnType Frlf_SetWakeupChannel ( uint8 Frlf_CtrlIdx, Frlf_ChannelType Frlf_ChnlIdx )	
<b>Service ID[hex]:</b>	0x09	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	non reentrant for identical values of Frlf_CtrlIdx, reentrant for different values of Frlf_CtrlIdx	
<b>Parameters (in):</b>	Frlf_CtrlIdx	Index of the FlexRay CC to address.
	Frlf_ChnlIdx	Index of the FlexRay Channel to address in scope of the FlexRay controller Frlf_CtrlIdx.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: The call of the FlexRay Driver's API service has returned E_OK. E_NOT_OK: The call of the FlexRay Driver's API service has returned E_NOT_OK, or an error has been detected in development mode.
<b>Description:</b>	Wraps the FlexRay Driver API function Fr_SetWakeupChannel(). The enum value "FR_CHANNEL_AB" shall not be used.	
<b>Available via:</b>	Frlf.h	

] ()

[SWS\_Frlf\_05500] [If parameter Frlf\_CtrlIdx of Frlf\_SetWakeupChannel has an invalid value and if development error detection is enabled (i.e. FrlfDevErrorDetect equals ON), the function Frlf\_SetWakeupChannel shall report development error code FRIF\_E\_INV\_CTRL\_IDX to the Det\_ReportError service of the DET module. ] ()

[SWS\_Frlf\_05177] [If parameter Frlf\_ChnlIdx of Frlf\_SetWakeupChannel has an invalid value and if development error detection is enabled (i.e. FrlfDevErrorDetect equals ON), the function Frlf\_SetWakeupChannel shall report development error code FRIF\_E\_INV\_CHNL\_IDX to the Det\_ReportError service of the DET module. ] ()

**[SWS\_Frlf\_05178]** [The function `Frlf_SetWakeupChannel` shall wrap the FlexRay Driver API function `Fr_SetWakeupChannel()` by:

- 1) Translating (based on static `Frlf` module configuration) the FlexRay CC index `Frlf_CtrlIdx` into a tuple (FlexRay Driver | Driver-specific CC index `Fr_CtrlIdx`).
- 2) Setting parameters `Fr_ChnlIdx` to `Frlf_ChnlIdx`
- 3) Calling `Fr_SetWakeupChannel()` of the determined FlexRay Driver module with the parameters determined as described above. ] ()

**[SWS\_Frlf\_05179]** [Caveats of `Frlf_SetWakeupChannel`: The FlexRay Interface module has to be initialized with a call of `Frlf_Init()` before this API service may be called, see `SWS_Frlf_05003`. ] ()

### 8.3.12 `Frlf_SendWUP`

**[SWS\_Frlf\_05011]** [

<b>Service name:</b>	<code>Frlf_SendWUP</code>	
<b>Syntax:</b>	<pre>Std_ReturnType Frlf_SendWUP(     uint8 Frlf_CtrlIdx )</pre>	
<b>Service ID[hex]:</b>	0x0a	
<b>Sync/Async:</b>	Asynchronous	
<b>Reentrancy:</b>	non reentrant for identical values of <code>Frlf_CtrlIdx</code> , reentrant for different values of <code>Frlf_CtrlIdx</code>	
<b>Parameters (in):</b>	<code>Frlf_CtrlIdx</code>	Index of the FlexRay CC to address.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	<code>Std_ReturnType</code>	<p><code>E_OK</code>: The call of the FlexRay Driver's API service has returned <code>E_OK</code>.</p> <p><code>E_NOT_OK</code>: The call of the FlexRay Driver's API service has returned <code>E_NOT_OK</code>, or an error has been detected in development mode.</p>
<b>Description:</b>	Wraps the FlexRay Driver API function <code>Fr_SendWUP()</code> .	
<b>Available via:</b>	<code>Frlf.h</code>	

] (`SRS_Fr_05018`)

**[SWS\_Frlf\_05180]** [If parameter `Frlf_CtrlIdx` of `Frlf_SendWUP` has an invalid value and if development error detection is enabled (i.e. `FrlfDevErrorDetect` equals ON), the function `Frlf_SendWUP` shall report development error code `FRIF_E_INV_CTRL_IDX` to the `Det_ReportError` service of the DET module. ] ()

**[SWS\_Frlf\_05181]** [The function `Frlf_SendWUP` shall wrap the FlexRay Driver API function `Fr_SendWUP()` by:

- 1) Translating (based on static `Frlf` module configuration) the FlexRay CC index `Frlf_CtrlIdx` into a tuple (FlexRay Driver | Driver-specific CC index `Fr_CtrlIdx`).

- 2) Calling Fr\_SendWUP() of the determined FlexRay Driver module with the parameters determined as described above.
- ] ()

**[SWS\_FrIf\_05182]** [Caveats of FrIf\_SendWUP: The FlexRay Interface module has to be initialized with a call of FrIf\_Init() before this API service may be called, see SWS\_FrIf\_05003. ] ()

### 8.3.13 FrIf\_GetPOCStatus

**[SWS\_FrIf\_05014]** [

<b>Service name:</b>	FrIf_GetPOCStatus	
<b>Syntax:</b>	Std_ReturnType FrIf_GetPOCStatus ( uint8 FrIf_CtrlIdx, Fr_POCStatusType* FrIf_POCStatusPtr )	
<b>Service ID[hex]:</b>	0x0d	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	non reentrant for identical values of FrIf_CtrlIdx, reentrant for different values of FrIf_CtrlIdx	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of the FlexRay CC to address.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	FrIf_POCStatusPtr	Pointer to a memory location where output value will be stored.
<b>Return value:</b>	Std_ReturnType	E_OK: The call of the FlexRay Driver's API service has returned E_OK. E_NOT_OK: The call of the FlexRay Driver's API service has returned E_NOT_OK, or an error has been detected in development mode.
<b>Description:</b>	Wraps the FlexRay Driver API function Fr_GetPOCStatus().	
<b>Available via:</b>	FrIf.h	

] (SRS\_Fr\_05022)

**[SWS\_FrIf\_05190]** [If parameter FrIf\_CtrlIdx of FrIf\_GetPOCStatus has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_GetPOCStatus shall report development error code FRIF\_E\_INV\_CTRL\_IDX to the Det\_ReportError service of the DET module. ] ()

**[SWS\_FrIf\_05192]** [The function FrIf\_GetPOCStatus shall wrap the FlexRay Driver API function Fr\_GetPOCStatus() by:

- 1) Translating (based on static FrIf module configuration) the FlexRay CC index FrIf\_CtrlIdx into a tuple (FlexRay Driver | Driver-specific CC index Fr\_CtrlIdx).
- 2) Setting parameters Fr\_POCStatusPtr to FrIf\_POCStatusPtr
- 3) Calling Fr\_GetPOCStatus() of the determined FlexRay Driver module with the parameters determined as described above. ] ()



**[SWS\_Frlf\_05193]** [Caveats of Frlf\_GetPOCStatus: The FlexRay Interface module has to be initialized with a call of Frlf\_Init() before this API service may be called, see SWS\_Frlf\_05003. ] ()

### 8.3.14 FrIf\_GetGlobalTime

**[SWS\_FrIf\_05015]** [

<b>Service name:</b>	FrIf_GetGlobalTime	
<b>Syntax:</b>	<pre>Std_ReturnType FrIf_GetGlobalTime (     uint8 FrIf_CtrlIdx,     uint8* FrIf_CyclePtr,     uint16* FrIf_MacroTickPtr )</pre>	
<b>Service ID[hex]:</b>	0x0e	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	non reentrant for identical values of FrIf_CtrlIdx, reentrant for different values of FrIf_CtrlIdx	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of the FlexRay CC to address.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	FrIf_CyclePtr	Pointer to a memory location where output value will be stored.
	FrIf_MacroTickPtr	Pointer to a memory location where output value will be stored.
<b>Return value:</b>	Std_ReturnType	<p>E_OK: The call of the FlexRay Driver's API service has returned E_OK.</p> <p>E_NOT_OK: The call of the FlexRay Driver's API service has returned E_NOT_OK, or an error has been detected in development mode.</p>
<b>Description:</b>	<p>Wraps the FlexRay Driver API function Fr_GetGlobalTime().</p> <p>Important Note: FrIf_GetGlobalTime may be called within an exclusive area.</p>	
<b>Available via:</b>	FrIf.h	

] ()

**[SWS\_FrIf\_05194]** [If parameter FrIf\_CtrlIdx of FrIf\_GetGlobalTime has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_GetGlobalTime shall report development error code FRIF\_E\_INV\_CTRL\_IDX to the Det\_ReportError service of the DET module. ] ()

**[SWS\_FrIf\_05195]** [The function FrIf\_GetGlobalTime shall wrap the FlexRay Driver API function Fr\_GetGlobalTime() by:

- 1) Translating (based on static FrIf module configuration) the FlexRay CC index FrIf\_CtrlIdx into a tuple (FlexRay Driver | Driver-specific CC index Fr\_CtrlIdx).
- 2) Setting parameters
- 3) Fr\_CylcePtr to FrIf\_CyclePtr  
Fr\_MacroTickPtr to FrIf\_MacroTickPtr
- 4) Calling Fr\_GetGlobalTime() of the determined FlexRay Driver module with the parameters determined as described above. ] ()

**[SWS\_FrIf\_05196]** [Caveats of FrIf\_GetGlobalTime: The FlexRay Interface module has to be initialized with a call of FrIf\_Init() before this API service may be called, see SWS\_FrIf\_05003. ] ()

### 8.3.15 FrIf\_AllowColdstart

[SWS\_FrIf\_05017] [

<b>Service name:</b>	FrIf_AllowColdstart	
<b>Syntax:</b>	Std_ReturnType FrIf_AllowColdstart ( uint8 FrIf_CtrlIdx )	
<b>Service ID[hex]:</b>	0x10	
<b>Sync/Async:</b>	Asynchronous	
<b>Reentrancy:</b>	non reentrant for identical values of FrIf_CtrlIdx, reentrant for different values of FrIf_CtrlIdx	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of the FlexRay CC to address.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: The call of the FlexRay Driver's API service has returned E_OK. E_NOT_OK: The call of the FlexRay Driver's API service has returned E_NOT_OK, or an error has been detected in development mode.
<b>Description:</b>	Wraps the FlexRay Driver API function Fr_AllowColdstart().	
<b>Available via:</b>	FrIf.h	

] ()

[SWS\_FrIf\_05200] [If parameter FrIf\_CtrlIdx of FrIf\_AllowColdstart has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_AllowColdstart shall report development error code FRIF\_E\_INV\_CTRL\_IDX to the Det\_ReportError service of the DET module. ] ()

[SWS\_FrIf\_05201] [The function FrIf\_AllowColdstart shall wrap the FlexRay Driver API function Fr\_AllowColdstart() by:

- 1) Translating (based on static FrIf module configuration) the FlexRay CC index FrIf\_CtrlIdx into a tuple (FlexRay Driver | Driver-specific CC index Fr\_CtrlIdx).
- 2) Calling Fr\_AllowColdstart() of the determined FlexRay Driver module with the parameters determined as described above. ] ()

[SWS\_FrIf\_05202] [Caveats: The FlexRay Interface module has to be initialized with a call of FrIf\_Init() before this API service may be called, see SWS\_FrIf\_05003. ] ()

### 8.3.16 FrIf\_GetMacroticksPerCycle

[SWS\_FrIf\_05018] [

<b>Service name:</b>	FrIf_GetMacroticksPerCycle
----------------------	----------------------------

<b>Syntax:</b>	uint16 FrIf_GetMacroticksPerCycle ( uint8 FrIf_CtrlIdx )
<b>Service ID[hex]:</b>	0x11
<b>Sync/Async:</b>	Synchronous
<b>Reentrancy:</b>	Reentrant
<b>Parameters (in):</b>	FrIf_CtrlIdx   Index of the FlexRay CC to address.
<b>Parameters (inout):</b>	None
<b>Parameters (out):</b>	None
<b>Return value:</b>	uint16   Number of Macroticks per Cycle
<b>Description:</b>	Retrieves the amount of Macroticks per Cycle
<b>Available via:</b>	FrIf.h

] ()

**[SWS\_FrIf\_05203]** | If parameter FrIf\_CtrlIdx of FrIf\_GetMacroticksPerCycle has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_GetMacroticksPerCycle shall report development error code FRIF\_E\_INV\_CTRL\_IDX to the Det\_ReportError service of the DET module.

This API service of the FlexRay Interface retrieves the number of Macroticks per FlexRay Cycle of the FlexRay Cluster with index FrIf\_CtrlIdx out of the static configuration. ] ()

**[SWS\_FrIf\_05204]** | Caveats of FrIf\_GetMacroticksPerCycle: The FlexRay Interface module has to be initialized with a call of FrIf\_Init() before this API service may be called, see SWS\_FrIf\_05003. ] ()

### 8.3.17 FrIf\_GetMacrotickDuration

**[SWS\_FrIf\_05019]** |

<b>Service name:</b>	FrIf_GetMacrotickDuration
<b>Syntax:</b>	uint16 FrIf_GetMacrotickDuration ( uint8 FrIf_CtrlIdx )
<b>Service ID[hex]:</b>	0x31
<b>Sync/Async:</b>	Synchronous
<b>Reentrancy:</b>	Reentrant
<b>Parameters (in):</b>	FrIf_CtrlIdx   Index of the FlexRay CC to address.
<b>Parameters (inout):</b>	None
<b>Parameters (out):</b>	None
<b>Return value:</b>	uint16   Duration of one Macrotick in ns
<b>Description:</b>	Retrieves the Duration of a Macrotick in ns
<b>Available via:</b>	FrIf.h

] ()

**[SWS\_FrIf\_05191]** | If parameter FrIf\_CtrlIdx of FrIf\_GetMacrotickDuration: has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect

equals ON), the function `Frlf_GetMacrotickDuration`: shall report development error code `FRIF_E_INV_CTRL_IDX` to the `Det_ReportError` service of the DET module.

This API service of the FlexRay Interface retrieves duration of one Macrotick in nanoseconds of the FlexRay Cluster with index `Frlf_CtrlIdx` out of the static configuration. `] ()`

**[SWS\_Frlf\_05754]** [Caveats of `Frlf_GetMacrotickDuration`: The FlexRay Interface module has to be initialized with a call of `Frlf_Init()` before this API service may be called, see `SWS_Frlf_05003`] `()`

### 8.3.18 `Frlf_Transmit`

**[SWS\_Frlf\_05033]** [

<b>Service name:</b>	<code>Frlf_Transmit</code>	
<b>Syntax:</b>	<pre>Std_ReturnType Frlf_Transmit(     PduIdType TxPduId,     const PduInfoType* PduInfoPtr )</pre>	
<b>Service ID[hex]:</b>	0x49	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant for different PduIds. Non reentrant for the same PduId.	
<b>Parameters (in):</b>	<code>TxPduId</code>	Identifier of the PDU to be transmitted
	<code>PduInfoPtr</code>	Length of and pointer to the PDU data and pointer to MetaData.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	<code>Std_ReturnType</code>	<code>E_OK</code> : Transmit request has been accepted. <code>E_NOT_OK</code> : Transmit request has not been accepted.
	<b>Description:</b> Requests transmission of a PDU.	
<b>Available via:</b>	<code>Frlf.h</code>	

`] ()`

**[SWS\_Frlf\_05318]**

`Frlf_Transmit()` shall return `E_NOT_OK` in case the Frlf's state is `FRIF_STATE_OFFLINE`.

**[SWS\_Frlf\_05205]** [If parameter `TxPduId` of `Frlf_Transmit` has an invalid value and if development error detection is enabled (i.e. `FrlfDevErrorDetect` equals ON), the function `Frlf_Transmit` shall report development error code `FRIF_E_INV_TXPDUID` to the `Det_ReportError` service of the DET module. `] ()`

**[SWS\_Frlf\_05207]** [If the parameter `FrlfAllowedDynamicSduLength` is set to false and/or if the parameter `FrlfImmediate` is set to true for the passed `TxPduId`, the passed `SduDataPtr` in parameter `PduInfoPtr` of `Frlf_Transmit` shall be checked for `NULL_PTR` in case development error detection is enabled (i.e. `FrlfDevErrorDetect` equals ON). If in this case the passed `SduDataPtr` equals `NULL_PTR`, the function

FrIf\_Transmit shall report the development error code FRIF\_E\_PARAM\_POINTER to the Det\_ReportError service of the DET module.

In case of decoupled transmission the PDU with index TxPduld is **not yet** passed to the underlying FlexRay Driver module for transmission. FrIf only remembers the PDU's transmission request (increment TrigTxCounter<sup>5</sup>). This decoupling mechanism between the call of FrIf\_Transmit() and the execution of the FrIfCommunicationAction (see [FrIf06067](#)) has some implications:

- The upper layer BSW module may operate asynchronously to the FlexRay Communication System and thus may call FrIf\_Transmit() at any point in time.
- The upper layer [BSW](#) module must permanently buffer the PDU's payload data and must be able to handle a call of its <UL\_TriggerTransmit>() API service at (from the [BSW](#)'s point of view) any arbitrary point in time. ] ()

**[SWS\_FrIf\_05208]** [In case of immediate transmission the function FrIf\_Transmit shall pass the PDU (single PDU, no Update bit) to the underlying FlexRay Driver module immediately for transmission. ] ()

**[SWS\_FrIf\_05757]** [

"If parameter TxPduld is configured for an immediate PDU, and if configuration parameter FrIfAllowDynamicLSduLength is set to FALSE, the provided length in PdulInfoPtr shall be compared with the static configured length (see ECUC\_FrIf\_06054).

If the length information does not match, FrIf\_Transmit() shall return E\_NOT\_OK and shall not perform the immediate PDU transmission. If development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_Transmit() shall report development error code FRIF\_E\_INV\_PDULENGTH to the Det\_ReportError service of the DET module. ] ()

**[SWS\_FrIf\_05209]** [Caveats of FrIf\_Transmit: The FlexRay Interface module has to be initialized with a call of FrIf\_Init() before this API service may be called, see SWS\_FrIf\_05003] ()

### 8.3.19 FrIf\_SetTransceiverMode

**[SWS\_FrIf\_05034]** [

<b>Service name:</b>	FrIf_SetTransceiverMode
<b>Syntax:</b>	Std_ReturnType FrIf_SetTransceiverMode( uint8 FrIf_CtrlIdx, Fr_ChannelType FrIf_ChnlIdx, FrTrcv_TrcevModeType FrIf_TrcevMode )
<b>Service ID[hex]:</b>	0x13
<b>Sync/Async:</b>	Synchronous
<b>Reentrancy:</b>	Reentrant

<sup>5</sup> Limited by static configuration [Configuration Parameter [FrIfCounterLimit](#), see [FrIf06076](#)]

<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of the FlexRay CC to address.
	FrIf_ChnlIdx	Index of the FlexRay Channel to address in scope of the FlexRay controller FrIf_CtrlIdx.
	FrIf_TrcvMode	Transceiver mode to be set.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: The call of the FlexRay Transceiver Driver's API service has returned E_OK. E_NOT_OK: The call of the FlexRay Transceiver Driver's API service has returned E_NOT_OK.
<b>Description:</b>	Wraps the FlexRay Transceiver Driver API function FrTrcv_SetTransceiverMode(). The enum value "FR_CHANNEL_AB" shall not be used.	
<b>Available via:</b>	FrIf.h	

] (SRS\_Fr\_05039)

**[SWS\_FrIf\_05210]** [If parameter FrIf\_CtrlIdx of FrIf\_SetTransceiverMode has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_SetTransceiverMode shall report development error code FRIF\_E\_INV\_CTRL\_IDX to the Det\_ReportError service of the DET module. ] ()

**[SWS\_FrIf\_05211]** [If parameter FrIf\_ChnlIdx of FrIf\_SetTransceiverMode has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_SetTransceiverMode shall report development error code FRIF\_E\_INV\_CHNL\_IDX to the Det\_ReportError service of the DET module. ] ()

**[SWS\_FrIf\_05212]** [The function FrIf\_SetTransceiverMode shall wrap the FlexRay Transceiver Driver API function FrTrcv\_SetTransceiverMode() by:

1. Translating (based on static [FrIf](#) module configuration) the tuple (FlexRay [CC](#) index FrIf\_CtrlIdx | FlexRay Channel index FrIf\_ChnlIdx) into a tuple (FlexRay Transceiver Driver | Driver-specific Transceiver index FrTrcv\_TrcvIdx).
2. Setting parameters
  - FrTrcv\_TrcvMode to FrIf\_TrcvMode
3. Calling FrTrcv\_SetTransceiverMode() of the determined FlexRay Driver module with the parameters determined as described above. ] ()

**[SWS\_FrIf\_05213]** [Caveats of FrIf\_SetTransceiverMode: The FlexRay Interface module has to be initialized with a call of FrIf\_Init() before this API service may be called, see SWS\_FrIf\_05003. ] ()

### 8.3.20 FrIf\_GetTransceiverMode

**[SWS\_FrIf\_05035]** [

<b>Service name:</b>	FrIf_GetTransceiverMode
<b>Syntax:</b>	Std_ReturnType FrIf_GetTransceiverMode( uint8 FrIf CtrlIdx,

	Fr_ChannelType FrIf_ChnlIdx, FrTrcv_TrvcModeType* FrIf_TrvcModePtr )	
<b>Service ID[hex]:</b>	0x14	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of the FlexRay CC to address.
	FrIf_ChnlIdx	Index of the FlexRay Channel to address in scope of the FlexRay controller FrIf_CtrlIdx.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	FrIf_TrvcModePtr	Pointer to a memory location where output value will be stored.
<b>Return value:</b>	Std_ReturnType	E_OK: The call of the FlexRay Transceiver Driver's API service has returned E_OK.
		E_NOT_OK: The call of the FlexRay Transceiver Driver's API service has returned E_NOT_OK.
<b>Description:</b>	Wraps the FlexRay Transceiver Driver API function FrTrcv_GetTransceiverMode(). The enum value "FR_CHANNEL_AB" shall not be used.	
<b>Available via:</b>	FrIf.h	

] (SRS\_Fr\_05157)

**[SWS\_Frlf\_05214]** [If parameter FrIf\_CtrlIdx of FrIf\_GetTransceiverMode has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_GetTransceiverMode shall report development error code FRIF\_E\_INV\_CTRL\_IDX to the Det\_ReportError service of the DET module. ] ()

**[SWS\_Frlf\_05215]** [If parameter FrIf\_ChnlIdx of FrIf\_GetTransceiverMode has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_GetTransceiverMode shall report development error code FRIF\_E\_INV\_CHNL\_IDX to the Det\_ReportError service of the DET module. ] ()

**[SWS\_Frlf\_05216]** [The function FrIf\_GetTransceiverMode shall wrap the FlexRay Transceiver Driver API function FrTrcv\_GetTransceiverMode() by:

1. Translating (based on static [Frlf](#) module configuration) the tuple (FlexRay [CC](#) index FrIf\_CtrlIdx | FlexRay Channel index FrIf\_ChnlIdx) into a tuple (FlexRay Transceiver Driver | Driver-specific Transceiver index FrTrcv\_TrvcIdx).
2. Setting parameters
  - FrTrcv\_TrvcModePtr to FrIf\_TrvcModePtr
3. Calling FrTrcv\_GetTransceiverMode() of the determined FlexRay Driver module with the parameters determined as described above. ] ()

**[SWS\_Frlf\_05217]** [Caveats of FrIf\_GetTransceiverMode: The FlexRay Interface module has to be initialized with a call of FrIf\_Init() before this API service may be called, see SWS\_Frlf\_05003. ] ()



### 8.3.21 FrIf\_GetTransceiverWUReason

#### [SWS\_FrIf\_05036] [

<b>Service name:</b>	FrIf_GetTransceiverWUReason	
<b>Syntax:</b>	<pre>Std_ReturnType FrIf_GetTransceiverWUReason(     uint8 FrIf_CtrlIdx,     Fr_ChannelType FrIf_ChnlIdx,     FrTrcv_TrcvWUReasonType* FrIf_TrcvWUReasonPtr )</pre>	
<b>Service ID[hex]:</b>	0x15	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of the FlexRay CC to address.
	FrIf_ChnlIdx	Index of the FlexRay Channel to address in scope of the FlexRay controller FrIf_CtrlIdx.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	FrIf_TrcvWUReasonPtr	Pointer to a memory location where output value will be stored.
<b>Return value:</b>	Std_ReturnType	E_OK: The call of the FlexRay Transceiver Driver's API service has returned E_OK. E_NOT_OK: The call of the FlexRay Transceiver Driver's API service has returned E_NOT_OK.
<b>Description:</b>	Wraps the FlexRay Transceiver Driver API function FrTrcv_GetTransceiverWUReason(). The enum value "FR_CHANNEL_AB" shall not be used.	
<b>Available via:</b>	FrIf.h	

] (SRS\_BSW\_00375, SRS\_Fr\_05158)

**[SWS\_FrIf\_05218]** [If parameter FrIf\_CtrlIdx of FrIf\_GetTransceiverWUReason has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_GetTransceiverWUReason shall report development error code FRIF\_E\_INV\_CTRL\_IDX to the Det\_ReportError service of the DET module. ] ()

**[SWS\_FrIf\_05219]** [If parameter FrIf\_ChnlIdx of FrIf\_GetTransceiverWUReason has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_GetTransceiverWUReason shall report development error code FRIF\_E\_INV\_CHNL\_IDX to the Det\_ReportError service of the DET module. ] ()

**[SWS\_FrIf\_05220]** [The function FrIf\_GetTransceiverWUReason shall wrap the FlexRay Transceiver Driver API function FrTrcv\_GetTransceiverWUReason() by:

1. Translating (based on static [FrIf](#) module configuration) the tuple (FlexRay [CC](#) index FrIf\_CtrlIdx | FlexRay Channel index FrIf\_ChnlIdx) into a tuple (FlexRay Transceiver Driver | Driver-specific Transceiver index FrTrcv\_TrcvIdx).
2. Setting parameters
  - FrTrcv\_TrcvWUReasonPtr to FrIf\_WUReasonPtr
3. Calling FrTrcv\_GetTransceiverWUReason() of the determined FlexRay Driver module with the parameters determined as described above. ] ()

**[SWS\_Frlf\_05221]** [Caveats of Frlf\_GetTransceiverWUReason: The FlexRay Interface module has to be initialized with a call of Frlf\_Init() before this API service may be called, see SWS\_Frlf\_05003. ] ()

### 8.3.22 Frlf\_ClearTransceiverWakeup

**[SWS\_Frlf\_05039]** [

<b>Service name:</b>	Frlf_ClearTransceiverWakeup	
<b>Syntax:</b>	Std_ReturnType Frlf_ClearTransceiverWakeup( uint8 Frlf_CtrlIdx, Frlf_ChannelType Frlf_ChnlIdx )	
<b>Service ID[hex]:</b>	0x18	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant	
<b>Parameters (in):</b>	Frlf_CtrlIdx	Index of the FlexRay CC to address.
	Frlf_ChnlIdx	Index of the FlexRay Channel to address in scope of the FlexRay controller Frlf_CtrlIdx.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: The call of the FlexRay Transceiver Driver's API service has returned E_OK. E_NOT_OK: The call of the FlexRay Transceiver Driver's API service has returned E_NOT_OK.
<b>Description:</b>	Wraps the FlexRay Transceiver Driver API function FrTrcv_ClearTransceiverWakeup(). The enum value "FR_CHANNEL_AB" shall not be used.	
<b>Available via:</b>	Frlf.h	

] (SRS\_Fr\_05161)

**[SWS\_Frlf\_05230]** [If parameter Frlf\_CtrlIdx of Frlf\_ClearTransceiverWakeup has an invalid value and if development error detection is enabled (i.e. FrlfDevErrorDetect equals ON), the function Frlf\_ClearTransceiverWakeup shall report development error code FRIF\_E\_INV\_CTRL\_IDX to the Det\_ReportError service of the DET module. ] ()

**[SWS\_Frlf\_05231]** [If parameter Frlf\_ChnlIdx of Frlf\_ClearTransceiverWakeup has an invalid value and if development error detection is enabled (i.e. FrlfDevErrorDetect equals ON), the function Frlf\_ClearTransceiverWakeup shall report development error code FRIF\_E\_INV\_CHNL\_IDX to the Det\_ReportError service of the DET module. ] ()

**[SWS\_Frlf\_05232]** [The function Frlf\_ClearTransceiverWakeup shall wrap the FlexRay Transceiver Driver API function FrTrcv\_ClearTransceiverWakeup() by:  
-1) Translating (based on static [Frlf](#) module configuration) the tuple (FlexRay [CC](#) index Frlf\_CtrlIdx | FlexRay Channel index Frlf\_ChnlIdx) into a tuple (FlexRay Transceiver Driver | Driver-specific Transceiver index FrTrcv\_TrcvIdx).

-2) Calling FrTrcv\_ClearTransceiverWakeup() of the determined FlexRay Driver module with the parameters determined as described above. ] ()

**[SWS\_FrIf\_05233]** [Caveats of FrIf\_ClearTransceiverWakeup: The FlexRay Interface module has to be initialized with a call of FrIf\_Init() before this API service may be called, see SWS\_FrIf\_05003. ] ()

### 8.3.23 FrIf\_CancelAbsoluteTimer

**[SWS\_FrIf\_05023]** [

<b>Service name:</b>	FrIf_CancelAbsoluteTimer	
<b>Syntax:</b>	Std_ReturnType FrIf_CancelAbsoluteTimer ( uint8 FrIf_CtrlIdx, uint8 FrIf_AbsTimerIdx )	
<b>Service ID[hex]:</b>	0x1b	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	non reentrant for the same FlexRay CC, reentrant for different FlexRay CCs	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of the FlexRay CC to address.
	FrIf_AbsTimerIdx	Index of the absolute timer to address.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: The call of the FlexRay Driver's API service has returned E_OK. E_NOT_OK: The call of the FlexRay Driver's API service has returned E_NOT_OK, or an error has been detected in development mode.
<b>Description:</b>	Wraps the FlexRay Driver API function Fr_CancelAbsoluteTimer() .	
<b>Available via:</b>	FrIf.h	

] ()

**[SWS\_FrIf\_05240]** [If parameter FrIf\_CtrlIdx of FrIf\_CancelAbsoluteTimer has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_CancelAbsoluteTimer shall report development error code FRIF\_E\_INV\_CTRL\_IDX to the Det\_ReportError service of the DET module. ] ()

**[SWS\_FrIf\_05241]** [The function FrIf\_CancelAbsoluteTimer shall wrap the FlexRay Driver API function Fr\_CancelAbsoluteTimer() by:

- 1) Translating (based on static FrIf module configuration) the FlexRay CC index FrIf\_CtrlIdx into a tuple (FlexRay Driver | Driver-specific CC index Fr\_CtrlIdx).
- 2) Setting parameters Fr\_AbsTimerIdx to FrIf\_AbsTimerIdx
- 3) Calling Fr\_CancelAbsoluteTimer() of the determined FlexRay Driver module with the parameters determined as described above. ] ()

**[SWS\_Frlf\_05242]** [Caveats of Frlf\_CancelAbsoluteTimer: The FlexRay Interface module has to be initialized with a call of Frlf\_Init() before this API service may be called, see SWS\_Frlf\_05003. ] ()

### 8.3.24 FrIf\_GetAbsoluteTimerIRQStatus

**[SWS\_FrIf\_05027]** |

<b>Service name:</b>	FrIf_GetAbsoluteTimerIRQStatus	
<b>Syntax:</b>	<pre>Std_ReturnType FrIf_GetAbsoluteTimerIRQStatus (     uint8 FrIf_CtrlIdx,     uint8 FrIf_AbsTimerIdx,     boolean* FrIf_IRQStatusPtr )</pre>	
<b>Service ID[hex]:</b>	0x1f	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	non reentrant for the same FlexRay CC, reentrant for different FlexRay CCs	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of the FlexRay CC to address.
	FrIf_AbsTimerIdx	Index of the absolute timer to address.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	FrIf_IRQStatusPtr	Pointer to a memory location where output value will be stored.
<b>Return value:</b>	Std_ReturnType	<p>E_OK: The call of the FlexRay Driver's API service has returned E_OK.</p> <p>E_NOT_OK: The call of the FlexRay Driver's API service has returned E_NOT_OK, or an error has been detected in development mode.</p>
<b>Description:</b>	Wraps the FlexRay Driver API function Fr_GetAbsoluteTimerIRQStatus()	
<b>Available via:</b>	FrIf.h	

| ()

**[SWS\_FrIf\_05252]** | If parameter FrIf\_CtrlIdx of FrIf\_GetAbsoluteTimerIRQStatus has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_GetAbsoluteTimerIRQStatus shall report development error code FRIF\_E\_INV\_CTRL\_IDX to the Det\_ReportError service of the DET module. | ()

**[SWS\_FrIf\_05253]** | The function FrIf\_GetAbsoluteTimerIRQStatus shall wrap the FlexRay Driver API function Fr\_GetAbsoluteTimerIRQStatus() by:

1. Translating (based on static FrIf module configuration) the FlexRay CC index FrIf\_CtrlIdx into a tuple (FlexRay Driver | Driver-specific CC index Fr\_CtrlIdx).
2. Setting parameters
  - Fr\_AbsTimerIdx to FrIf\_AbsTimerIdx
  - Fr\_IRQStatusPtr to FrIf\_IRQStatusPtr
3. Calling Fr\_GetAbsoluteTimerIRQStatus() of the determined FlexRay Driver module with the parameters determined as described above. | ()

**[SWS\_FrIf\_05254]** | Caveats of FrIf\_GetAbsoluteTimerIRQStatus: The FlexRay Interface module has to be initialized with a call of FrIf\_Init() before this API service may be called, see SWS\_FrIf\_05003. | ()

### 8.3.25 FrIf\_DisableAbsoluteTimerIRQ

#### [SWS\_FrIf\_05031] [

<b>Service name:</b>	FrIf_DisableAbsoluteTimerIRQ	
<b>Syntax:</b>	Std_ReturnType FrIf_DisableAbsoluteTimerIRQ( uint8 FrIf_CtrlIdx, uint8 FrIf_AbsTimerIdx )	
<b>Service ID[hex]:</b>	0x23	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	non reentrant for the same FlexRay CC, reentrant for different FlexRay CCs	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of the FlexRay CC to address.
	FrIf_AbsTimerIdx	Index of the absolute timer to address.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: The call of the FlexRay Driver's API service has returned E_OK. E_NOT_OK: The call of the FlexRay Driver's API service has returned E_NOT_OK, or an error has been detected in development mode.
<b>Description:</b>	Wraps the FlexRay Driver API function Fr_DisableAbsoluteTimerIRQ().	
<b>Available via:</b>	FrIf.h	

] ()

**[SWS\_FrIf\_05264]** [If parameter FrIf\_CtrlIdx of FrIf\_DisableAbsoluteTimerIRQ has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_DisableAbsoluteTimerIRQ shall report development error code FRIF\_E\_INV\_CTRL\_IDX to the Det\_ReportError service of the DET module. ] ()

**[SWS\_FrIf\_05266]** [Caveats of FrIf\_DisableAbsoluteTimerIRQ: The FlexRay Interface module has to be initialized with a call of FrIf\_Init() before this API service may be called, see SWS\_FrIf\_05003. ] ()

### 8.3.26 FrIf\_GetCycleLength

#### [SWS\_FrIf\_05239] [

<b>Service name:</b>	FrIf_GetCycleLength	
<b>Syntax:</b>	uint32 FrIf_GetCycleLength( uint8 FrIf_CtrlIdx )	
<b>Service ID[hex]:</b>	0x3a	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant for the same FlexRay CC, reentrant for different FlexRay CCs	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of the FlexRay CC to address.
<b>Parameters</b>	None	

<b>(inout):</b>	
<b>Parameters (out):</b>	None
<b>Return value:</b>	uint32   Time in unit of nanoseconds
<b>Description:</b>	This API returns the configured time of the configuration parameter "GdCycle" in nanoseconds for the FlexRay controller with index Frlf_CtrlIdx.
<b>Available via:</b>	FrIf.h

] ()

**[SWS\_Frlf\_05237]** [If parameter Frlf\_CtrlIdx of Frlf\_GetCycleLength has an invalid value and if development error detection is enabled (i.e. FrlfDevErrorDetect equals ON), the function Frlf\_GetCycleLength shall report development error code FRIF\_E\_INV\_CTRL\_IDX to the Det\_ReportError service of the DET module. ] ()

**[SWS\_Frlf\_05238]** [Caveats of Frlf\_GetCycleLength: The FlexRay Interface module has to be initialized with a call of Frlf\_Init() before this API service may be called, see SWS\_Frlf\_05003. ] ()

## 8.4 Optional Function Definitions

### 8.4.1 FrIf\_AllSlots

[SWS\_FrIf\_05020] |

<b>Service name:</b>	FrIf_AllSlots	
<b>Syntax:</b>	Std_ReturnType FrIf_AllSlots( uint8 FrIf_CtrlIdx )	
<b>Service ID[hex]:</b>	0x33	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	non reentrant	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of the FlexRay CC to address.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: The call of the FlexRay Driver's API service has returned E_OK. E_NOT_OK: The call of the FlexRay Driver's API service has returned E_NOT_OK, or an error has been detected in development mode.
<b>Description:</b>	Wraps the FlexRay Driver API function Fr_AllSlots	
<b>Available via:</b>	FrIf.h	

| ()

[SWS\_FrIf\_05412] | The function FrIf\_AllSlots shall be pre compile time configurable ON/OFF by the configuration parameter FrIfAllSlotsSupport (derived from configuration parameter FrIfAllSlotsSupport, see ECUC\_FrIf\_06108) |

()

[SWS\_FrIf\_05706] | If development error detection for the FrIf module is enabled: if the function FrIf\_AllSlots is called before the FrIf was initialized successfully, the function FrIf\_AllSlots shall raise the development error FRIF\_E\_UNINIT and return E\_NOT\_OK. | ()

[SWS\_FrIf\_05707] | If development error detection for the Fr module is enabled: the function FrIf\_AllSlots shall check the parameter FrIf\_CtrlIdx for being valid. If FrIf\_CtrlIdx is invalid, the function FrIf\_AllSlots shall raise the development error FRIF\_E\_INV\_CTRL\_IDX and return E\_NOT\_OK. | ()



### 8.4.2 FrIf\_GetChannelStatus

**[SWS\_FrIf\_05030]** |

<b>Service name:</b>	FrIf_GetChannelStatus	
<b>Syntax:</b>	<pre>Std_ReturnType FrIf_GetChannelStatus (     uint8 FrIf_CtrlIdx,     uint16* FrIf_ChannelAStatusPtr,     uint16* FrIf_ChannelBStatusPtr )</pre>	
<b>Service ID[hex]:</b>	0x26	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant for the same device	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of FlexRay CC within the context of the FlexRay Interface.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	FrIf_ChannelAStatusPtr	Address where the bitcoded channel A status information shall be stored.
	FrIf_ChannelBStatusPtr	Address where the bitcoded channel B status information shall be stored.
<b>Return value:</b>	Std_ReturnType	E_OK: API call finished successfully. E_NOT_OK: API call aborted due to errors.
<b>Description:</b>	Wraps the FlexRay Driver API function Fr_GetChannelStatus() and gets the channel status information.	
<b>Available via:</b>	FrIf.h	

| ()

**[SWS\_FrIf\_05413]** | The function FrIf\_GetChannelStatus shall be pre compile time configurable ON/OFF by the configuration parameter FrIfGetGetChannelStatusSupport (derived from configuration parameter FrIfGetGetChannelStatusSupport, see ECUC\_FrIf\_06105) | ()

**[SWS\_FrIf\_05708]** | If development error detection for the FrIf module is enabled: if the function FrIf\_GetChannelStatus is called before the FrIf module was initialized successfully, the function FrIf\_GetChannelStatus shall raise the development error FRIF\_E\_UNINIT and return E\_NOT\_OK. | ()

**[SWS\_FrIf\_05709]** | If development error detection for the FrIf module is enabled: the function FrIf\_GetChannelStatus shall check the parameter FrIf\_CtrlIdx for being valid. If FrIf\_CtrlIdx is invalid, the function FrIf\_GetChannelStatus shall raise the development error FRIF\_E\_INV\_CTRL\_IDX and return E\_NOT\_OK. | ()

### 8.4.3 FrIf\_GetClockCorrection

**[SWS\_FrIf\_05071]** |

<b>Service name:</b>	FrIf_GetClockCorrection
----------------------	-------------------------

<b>Syntax:</b>	<pre>Std_ReturnType FrIf_GetClockCorrection(     uint8 FrIf_CtrlIdx,     sint16* FrIf_RateCorrectionPtr,     sint32* FrIf_OffsetCorrectionPtr )</pre>	
<b>Service ID[hex]:</b>	0x29	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant for the same device	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of FlexRay CC within the context of the FlexRay Interface.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	FrIf_RateCorrectionPtr	Address where the current rate correction value shall be stored.
	FrIf_OffsetCorrectionPtr	Address where the current offset correction value shall be stored.
<b>Return value:</b>	Std_ReturnType	E_OK: API call finished successfully. E_NOT_OK: API call aborted due to errors.
<b>Description:</b>	Wraps the FlexRay Driver API function Fr_GetClockCorrection () and gets the current clock correction values.	
<b>Available via:</b>	FrIf.h	

] ()

**[SWS\_FrIf\_05414]** [The function FrIf\_GetClockCorrection shall be pre compile time configurable ON/OFF by the configuration parameter FrIfGetClockCorrectionSupport (derived from configuration parameter FrIfGetClockCorrectionSupport, see ECUC\_FrIf\_06106) ] ()

**[SWS\_FrIf\_05711]** [If development error detection for the FrIf module is enabled: if the function FrIf\_GetClockCorrection is called before the FrIf was initialized successfully, the function FrIf\_GetClockCorrection shall raise the development error FRIF\_E\_UNINIT and return E\_NOT\_OK. ] ()

**[SWS\_FrIf\_05712]** [If development error detection for the FrIf module is enabled: the function FrIf\_GetClockCorrection shall check the parameter FrIf\_CtrlIdx for being valid. If FrIf\_CtrlIdx is invalid, the function FrIf\_GetClockCorrection shall raise the development error FRIF\_E\_INV\_CTRL\_IDX and return E\_NOT\_OK. ] ()

#### 8.4.4 FrIf\_GetSyncFrameList

**[SWS\_FrIf\_05072]** [

<b>Service name:</b>	FrIf_GetSyncFrameList	
<b>Syntax:</b>	<pre>Std_ReturnType FrIf_GetSyncFrameList(     uint8 FrIf_CtrlIdx,     uint8 FrIf_ListSize,     uint16* FrIf_ChannelAEvenListPtr,     uint16* FrIf_ChannelBEvenListPtr,     uint16* FrIf_ChannelAOddListPtr,     uint16* FrIf_ChannelBOddListPtr )</pre>	

	)	
<b>Service ID[hex]:</b>	0x2a	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant for the same device	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of FlexRay CC within the context of the FlexRay Interface.
	FrIf_ListSize	Size of the arrays passed via parameters: FrIf_ChannelAEvenListPtr FrIf_ChannelBEvenListPtr FrIf_ChannelAOddListPtr FrIf_ChannelBOddListPtr. The service must ensure to not write more entries into those arrays than granted by this parameter.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	FrIf_ChannelAEvenListPtr	Address the list of syncframes on channel A within the even communication cycle is written to. The exact number of elements written to the list is limited by parameter FrIf_ListSize. Unused list elements are filled with the value '0' to indicate that no more syncframe has been seen.
	FrIf_ChannelBEvenListPtr	Address the list of syncframes on channel B within the even communication cycle is written to. The exact number of elements written to the list is limited by parameter FrIf_ListSize. Unused list elements are filled with the value '0' to indicate that no more syncframe has been seen.
	FrIf_ChannelAOddListPtr	Address the list of syncframes on channel A within the odd communication cycle is written to. The exact number of elements written to the list is limited by parameter FrIf_ListSize. Unused list elements are filled with the value '0' to indicate that no more syncframe has been seen.
	FrIf_ChannelBOddListPtr	Address the list of syncframes on channel B within the odd communication cycle is written to. The exact number of elements written to the list is limited by parameter FrIf_ListSize. Unused list elements are filled with the value '0' to indicate that no more syncframe has been seen.
<b>Return value:</b>	Std_ReturnType	E_OK: API call finished successfully. E_NOT_OK: API call aborted due to errors.
<b>Description:</b>	Wraps the FlexRay Driver API function Fr_GetSyncFrameList and gets a list of syncframes received or transmitted on channel A and channel B via the even and odd communication cycle.	
<b>Available via:</b>	FrIf.h	

] ()

**[SWS\_FrIf\_05415]** [The function FrIf\_GetSyncFrameList shall be pre compile time configurable ON/OFF by the configuration parameter FrIfGetSyncFrameListSupport (derived from configuration parameter FrIfGetSyncFrameListSupport, see ECUC\_FrIf\_06107) ] ()

**[SWS\_FrIf\_05715]** [If development error detection for the FrIf module is enabled: if the function FrIf\_GetSyncFrameList is called before the Fr was initialized successfully, the function FrIf\_GetSyncFrameList shall raise the development error FRIF\_E\_UNINIT and return E\_NOT\_OK. ] ()

**[SWS\_Frlf\_05716]** [If development error detection for the Frlf module is enabled: the function Frlf\_GetSyncFrameList shall check the parameter Frlf\_CtrlIdx for being valid. If Frlf\_CtrlIdx is invalid, the function Frlf\_GetSyncFrameList shall raise the development error FRIF\_E\_INV\_CTRL\_IDX and return E\_NOT\_OK. ] ()

#### 8.4.5 Frlf\_GetNumOfStartupFrames

**[SWS\_Frlf\_05073]** [

<b>Service name:</b>	Frlf_GetNumOfStartupFrames	
<b>Syntax:</b>	Std_ReturnType Frlf_GetNumOfStartupFrames ( uint8 Frlf_CtrlIdx, uint8* Frlf_NumOfStartupFramesPtr )	
<b>Service ID[hex]:</b>	0x34	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant for the same device	
<b>Parameters (in):</b>	Frlf_CtrlIdx	Index of FlexRay CC within the context of the FlexRay Interface.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	Frlf_NumOfStartupFramesPtr	Address where the number of startup frames seen within the last even/odd cycle pair shall be stored.
<b>Return value:</b>	Std_ReturnType	E_OK: API call finished successfully. E_NOT_OK: API call aborted due to errors.
<b>Description:</b>	Wraps the FlexRay Driver API function Fr_GetNumOfStartupFrames and gets a list of the current number of startup frames seen on the cluster. See variable vStartupPairs of [12] for details.	
<b>Available via:</b>	Frlf.h	

] ()

**[SWS\_Frlf\_05416]** [The function Frlf\_GetNumOfStartupFrames shall be pre compile time configurable ON/OFF by the configuration parameter FrlfGetNumOfStartupFramesSupport (derived from configuration parameter FrlfGetNumOfStartupFramesSupport, see ECUC\_Frlf\_06104) ] ()

**[SWS\_Frlf\_05721]** [If development error detection for the Frlf module is enabled: if the function Frlf\_GetNumOfStartupFrames is called before the Frlf was initialized successfully, the function Frlf\_GetNumOfStartupFrames shall raise the development error FRIF\_E\_UNINIT and return E\_NOT\_OK. ] ()

**[SWS\_Frlf\_05722]** [If development error detection for the Frlf module is enabled: the function Frlf\_GetNumOfStartupFrames shall check the parameter Frlf\_CtrlIdx for being valid. If Frlf\_CtrlIdx is invalid, the function Frlf\_GetNumOfStartupFrames shall raise the development error FRIF\_E\_INV\_CTRL\_IDX and return E\_NOT\_OK. ] ()

### 8.4.6 FrIf\_GetWakeupRxStatus

[SWS\_FrIf\_05102] [

<b>Service name:</b>	FrIf_GetWakeupRxStatus	
<b>Syntax:</b>	Std_ReturnType FrIf_GetWakeupRxStatus ( uint8 FrIf_CtrlIdx, uint8* FrIf_WakeupRxStatusPtr )	
<b>Service ID[hex]:</b>	0x2b	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant for the same device	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of FlexRay CC within the context of the FlexRay Driver.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	FrIf_WakeupRxStatusPtr	Address where bitcoded wakeup reception status shall be stored. Bit 0: Wakeup received on channel A indicator Bit 1: Wakeup received on channel B indicator Bit 2-7: Unused
<b>Return value:</b>	Std_ReturnType	E_OK: API call finished successfully. E_NOT_OK: API call aborted due to errors.
<b>Description:</b>	Wraps the FlexRay Driver API function Fr_GetWakeupRxStatus and gets the wakeup received information from the FlexRay controller.	
<b>Available via:</b>	FrIf.h	

] ()

[SWS\_FrIf\_05417] [The function FrIf\_GetWakeupRxStatus shall be pre compile time configurable ON/OFF by the configuration parameter FrIfGetWakeupRxStatusSupport (derived from configuration parameter FrIfGetWakeupRxStatusSupport, see ECUC\_FrIf\_06111) ] ()

[SWS\_FrIf\_05700] [If development error detection for the FrIf module is enabled: if the function FrIf\_GetWakeupRxStatus is called before the Fr was initialized successfully, the function FrIf\_GetWakeupRxStatus shall raise the development error FRIF\_E\_UNINIT and return E\_NOT\_OK. ] ()

[SWS\_FrIf\_05701] [If development error detection for the FrIf module is enabled: the function FrIf\_GetWakeupRxStatus shall check the parameter FrIf\_CtrlIdx for being valid. If FrIf\_CtrlIdx is invalid, the function FrIf\_GetWakeupRxStatus shall raise the development error FRIF\_E\_INV\_CTRL\_IDX and return E\_NOT\_OK. ] ()

### 8.4.7 FrIf\_CancelTransmit

[SWS\_FrIf\_05070] [

<b>Service name:</b>	FrIf_CancelTransmit	
<b>Syntax:</b>	Std_ReturnType FrIf_CancelTransmit ( PduIdType TxPduId )	

<b>Service ID[hex]:</b>	0x4a	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant for different Pdulds. Non reentrant for the same Pdulds.	
<b>Parameters (in):</b>	TxPdulds	Identification of the PDU to be cancelled.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: Cancellation was executed successfully by the destination module. E_NOT_OK: Cancellation was rejected by the destination module.
<b>Description:</b>	Requests cancellation of an ongoing transmission of a PDU in a lower layer communication module.	
<b>Available via:</b>	FrIf.h	

] ()

**[SWS\_FrIf\_05713]** ¶The function FrIf\_CancelTransmit shall be pre compile time configurable ON/OFF by the configuration parameter FrIfCancelTransmitSupport (derived from configuration parameter FrIfCancelTransmitSupport, see ECUC\_FrIf\_00002) ] ()

**[SWS\_FrIf\_05703]** ¶If development error detection for the FrIf module is enabled: if the function FrIf\_CancelTransmit is called before the FrIf was initialized successfully, the function FrIf\_CancelTransmit shall raise the development error FRIF\_E\_UNINIT and return E\_NOT\_OK. ] ()

**[SWS\_FrIf\_05704]** ¶If development error detection for the FrIf module is enabled: the function FrIf\_CancelTransmit shall check the parameter TxPdulds for being valid. If TxPdulds is invalid, the function FrIf\_CancelTransmit shall raise the development error FRIF\_E\_INV\_TXPDUID and return E\_NOT\_OK. ] ()

**[SWS\_FrIf\_05705]** ¶For Transmit Cancellation, the following steps are performed:

1. Decrement TrigTxCounter for the IPDU that shall be canceled.
2. If TxConfCounter > 0 for this PDU, continue with step 3). Else, stop here.
3. Call FlexRay Driver's API function Fr\_CancelTxLPdu():
  - a. Fr\_CtrlIdx is derived according to the indexing scheme described in 7.2
  - b. Fr\_LPduldx is set to the configured L-PDU buffer index [Configuration Parameter FrIfLPduldx, see [FrIf06058](#)] associated with the Communication Operation.
4. Increment [TrigTxCounter](#) (limited by FrIfCounterLimit) for all other I-PDUs within that L-PDU that have a TxConfCounter > 0.
5. Decrement TxConfCounter for all other I-PDUs within that L-PDU that have a TxConfCounter > 0.
6. Decrement the TxConfCounter for the IPDU that has been initiated by the CancelTransmit API call. ] ()

### 8.4.8 FrIf\_DisableLPdu

#### [SWS\_FrIf\_05710] [

<b>Service name:</b>	FrIf_DisableLPdu	
<b>Syntax:</b>	<pre>Std_ReturnType FrIf_DisableLPdu(     uint8 FrIf_CtrlIdx,     uint16 FrIf_LPduIdx )</pre>	
<b>Service ID[hex]:</b>	0x28	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant for the same device	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of FlexRay CC within the context of the FlexRay Interface.
	FrIf_LPduIdx	This index is used to uniquely identify a FlexRay frame
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: API call finished successfully. E_NOT_OK: API call aborted due to errors.
<b>Description:</b>	Wraps the FlexRay Driver Function Fr_DisableLPdu. It disables the hardware resource of an LPdu for transmission/reception.	
<b>Available via:</b>	FrIf.h	

] ()

**[SWS\_FrIf\_05418]** [The function FrIf\_DisableLPdu shall be pre compile time configurable ON/OFF by the configuration parameter FrIfDisableLPduSupport (derived from configuration parameter FrIfDisableLPduSupport, see ECUC\_FrIf\_06110) ] ()

**[SWS\_FrIf\_05717]** [If development error detection for the FrIf module is enabled: if the function FrIf\_DisableLPdu is called before the FrIf was initialized successfully, the function FrIf\_DisableLPdu shall raise the development error FRIF\_E\_UNINIT and return E\_NOT\_OK. ] ()

**[SWS\_FrIf\_05714]** [If development error detection for the FrIf module is enabled: the function FrIf\_DisableLPdu shall check the parameter FrIf\_CtrlIdx for being valid. If FrIf\_CtrlIdx is invalid, the function FrIf\_DisableLPdu shall raise the development error FRIF\_E\_INV\_CTRL\_IDX and return E\_NOT\_OK. ] ()

### 8.4.9 FrIf\_GetTransceiverError

#### [SWS\_FrIf\_05032] [

<b>Service name:</b>	FrIf_GetTransceiverError	
<b>Syntax:</b>	<pre>Std_ReturnType FrIf_GetTransceiverError(     uint8 FrIf_CtrlIdx,     Fr_ChannelType FrIf_ChnlIdx,     uint8 FrIf_BranchIdx,     uint32* FrIf_BusErrorState )</pre>	

<b>Service ID[hex]:</b>	0x35	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Function is non reentrant for the same channel of the same controller.	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of the FlexRay CC to address.
	FrIf_ChnlIdx	Index of the FlexRay Channel to address in scope of the FlexRay controller FrIf_CtrlIdx.
	FrIf_BranchIdx	This zero based index identifies the branch of the (active star) transceiver to which the API call has to be applied.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	FrIf_BusErrorState	Address where the transceiver error state is stored.
<b>Return value:</b>	Std_ReturnType	E_OK: API call finished successfully. E_NOT_OK: API call aborted due to errors
	<b>Description:</b> Wraps the FlexRay Transceiver Driver API function FrTrcv_GetTransceiverError. The enum value "FR_CHANNEL_AB" shall not be used.	
<b>Available via:</b>	FrIf.h	

⌋ ()

**[SWS\_FrIf\_05419]** ⌈The function FrIf\_GetTransceiverError shall be pre compile time configurable ON/OFF by the configuration parameter FrIfGetTransceiverErrorSupport (derived from configuration parameter FrIfGetTransceiverErrorSupport, see ECUC\_FrIf\_06101) ⌋ ()

**[SWS\_FrIf\_05718]** ⌈If development error detection for the FrIf module is enabled: if the function FrIf\_GetTransceiverError is called before the FrIf was initialized successfully, the function FrIf\_GetTransceiverError shall raise the development error FRIF\_E\_UNINIT and return E\_NOT\_OK. ⌋ ()

**[SWS\_FrIf\_05719]** ⌈If development error detection for the FrIf module is enabled: the function FrIf\_GetTransceiverError shall check the parameter FrIf\_CtrlIdx for being valid. If FrIf\_CtrlIdx is invalid, the function FrIf\_GetTransceiverError shall raise the development error FRIF\_E\_INV\_CTRL\_IDX and return E\_NOT\_OK. ⌋ ()

**[SWS\_FrIf\_05720]** ⌈If parameter FrIf\_ChnlIdx of FrIf\_GetTransceiverError has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_GetTransceiverError shall report development error code FRIF\_E\_INV\_CHNL\_IDX to the Det\_ReportError service of the DET module. ⌋ ()

**[SWS\_FrIf\_05728]** ⌈The function FrIf\_GetTransceiverError shall wrap the FlexRay Transceiver Driver API function FrTrcv\_GetTransceiverError by:

1. Translating (based on static [FrIf](#) module configuration) the tuple (FlexRay [CC](#) index FrIf\_CtrlIdx | FlexRay Channel index FrIf\_ChnlIdx) into a tuple (FlexRay Transceiver Driver | Driver-specific Transceiver index FrTrcv\_TrcvIdx).
2. Setting parameters
  - FrTrcv\_BranchIdx to FrIf\_BranchIdx
  - FrTrcv\_BusErrorState to FrIf\_BusErrorState
3. Calling FrTrcv\_GetTransceiverError of the determined FlexRay Transceiver module with the parameters determined as described above. ⌋ ()



### 8.4.10 FrIf\_EnableTransceiverBranch

[SWS\_FrIf\_05085] |

<b>Service name:</b>	FrIf_EnableTransceiverBranch	
<b>Syntax:</b>	<pre>Std_ReturnType FrIf_EnableTransceiverBranch (     uint8 FrIf_CtrlIdx,     Fr_ChannelType FrIf_ChnlIdx,     uint8 FrIf_BranchIdx )</pre>	
<b>Service ID[hex]:</b>	0x36	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of the FlexRay CC to address.
	FrIf_ChnlIdx	Index of the FlexRay Channel to address in scope of the FlexRay controller FrIf_CtrlIdx.
	FrIf_BranchIdx	This zero based index identifies the branch of the (active star) transceiver to which the API call has to be applied.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: The call of the FlexRay Transceiver Driver's API service has returned E_OK. E_NOT_OK: The call of the FlexRay Transceiver Driver's API service has returned E_NOT_OK.
<b>Description:</b>	Wraps the FlexRay Transceiver Driver API function FrTrcv_EnableTransceiverBranch. The enum value "FR_CHANNEL_AB" shall not be used.	
<b>Available via:</b>	FrIf.h	

| ()

[SWS\_FrIf\_05420] | The function FrIf\_EnableTransceiverBranch shall be pre compile time configurable ON/OFF by the configuration parameter FrIfEnableTransceiverBranchSupport (derived from configuration parameter FrIfEnableTransceiverBranchSupport, see ECUC\_FrIf\_06103) | ()

[SWS\_FrIf\_05302] | If parameter FrIf\_CtrlIdx of FrIf\_EnableTransceiverBranch has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_EnableTransceiverBranch shall report development error code FRIF\_E\_INV\_CTRL\_IDX to the Det\_ReportError service of the DET module. | ()

[SWS\_FrIf\_05304] | If parameter FrIf\_ChnlIdx of FrIf\_EnableTransceiverBranch has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_EnableTransceiverBranch shall report development error code FRIF\_E\_INV\_CHNL\_IDX to the Det\_ReportError service of the DET module. | ()

- [SWS\_Frlf\_05306]** [The function `Frlf_EnableTransceiverBranch` shall wrap the FlexRay Transceiver Driver API function `Frlf_EnableTransceiverBranch` by:
1. Translating (based on static [Frlf](#) module configuration) the tuple (FlexRay [CC](#) index `Frlf_CtrlIdx` | FlexRay Channel index `Frlf_ChnlIdx`) into a tuple (FlexRay Transceiver Driver | Driver-specific Transceiver index `FrTrcv_TrcvIdx`).
  - 2) Setting parameter: `FrTrcv_BranchIdx` to `Frlf_BranchIdx`
  - 3) Calling `FrTrcv_EnableTransceiverBranch` of the determined FlexRay Driver module with the parameters determined as described above. ] ()

**[SWS\_Frlf\_05307]** [If development error detection for the `Frlf` module is enabled: if the function `Frlf_EnableTransceiverBranch` is called before the `Fr` was initialized successfully, the function `Frlf_EnableTransceiverBranch` shall raise the development error `FRIF_E_UNINIT` and return `E_NOT_OK`. ] ()

#### 8.4.11 `Frlf_DisableTransceiverBranch`

**[SWS\_Frlf\_05028]** [

<b>Service name:</b>	<code>Frlf_DisableTransceiverBranch</code>	
<b>Syntax:</b>	<pre>Std_ReturnType Frlf_DisableTransceiverBranch(     uint8 Frlf_CtrlIdx,     Fr_ChannelType Frlf_ChnlIdx,     uint8 Frlf_BranchIdx )</pre>	
<b>Service ID[hex]:</b>	0x37	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant	
<b>Parameters (in):</b>	<code>Frlf_CtrlIdx</code>	Index of the FlexRay CC to address.
	<code>Frlf_ChnlIdx</code>	Index of the FlexRay Channel to address in scope of the FlexRay controller <code>Frlf_CtrlIdx</code> .
	<code>Frlf_BranchIdx</code>	This zero based index identifies the branch of the (active star) transceiver to which the API call has to be applied.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	<code>Std_ReturnType</code>	<code>E_OK</code> : The call of the FlexRay Transceiver Driver's API service has returned <code>E_OK</code> . <code>E_NOT_OK</code> : The call of the FlexRay Transceiver Driver's API service has returned <code>E_NOT_OK</code> .
<b>Description:</b>	Wraps the FlexRay Transceiver Driver API function <code>FrTrcv_DisableTransceiverBranch</code> . The enum value "FR_CHANNEL_AB" shall not be used.	
<b>Available via:</b>	<code>FrIf.h</code>	

] ()

**[SWS\_Frlf\_05421]** [The function `Frlf_DisableTransceiverBranch` shall be pre compile time configurable ON/OFF by the configuration parameter `FrlfDisableTransceiverBranchSupport` (derived from configuration parameter `FrlfDisableTransceiverBranchSupport`, see ECUC\_Frlf\_06102) ] ()

**[SWS\_Frlf\_05425]** [The function `Frlf_DisableTransceiverBranch` shall be pre compile time configurable ON/OFF by the configuration parameter `FrlfDisableTransceiverBranchSupport` (derived from configuration parameter `FrlfDisableTransceiverBranchSupport`, see ECUC\_Frlf\_06102) ] ()

**[SWS\_Frlf\_05756]** [If parameter `Frlf_CtrlIdx` of `Frlf_DisableTransceiverBranch` has an invalid value and if development error detection is enabled (i.e. `FrlfDevErrorDetect` equals ON), the function `Frlf_DisableTransceiverBranch` shall report development error code `FRIF_E_INV_CTRL_IDX` to the `Det_ReportError` service of the DET module. ] ()

**[SWS\_Frlf\_05243]** [If parameter `Frlf_ChnlIdx` of `Frlf_DisableTransceiverBranch` has an invalid value and if development error detection is enabled (i.e. `FrlfDevErrorDetect` equals ON), the function `Frlf_DisableTransceiverBranch` shall report development error code `FRIF_E_INV_CHNL_IDX` to the `Det_ReportError` service of the DET module. ] ()

**[SWS\_Frlf\_05305]** [The function `Frlf_DisableTransceiverBranch` shall wrap the FlexRay Transceiver Driver API function `Frlf_DisableTransceiverBranch` by:

- 1) Translating (based on static [Frlf](#) module configuration) the tuple (FlexRay [CC](#) index `Frlf_CtrlIdx` | FlexRay Channel index `Frlf_ChnlIdx`) into a tuple (FlexRay Transceiver Driver | Driver-specific Transceiver index `FrTrcv_TrcvIdx`)
- 2) Setting parameter: `FrTrcv_BranchIdx` to `Frlf_BranchIdx`
- 3) Calling `FrTrcv_DisableTransceiverBranch()` of the determined FlexRay Driver module with the parameters determined as described above. ] ()

**[SWS\_Frlf\_05308]** [Caveats of `Frlf_DisableTransceiverBranch`: The FlexRay Interface module has to be initialized with a call of `Frlf_Init()` before this API service may be called, see SWS\_Frlf\_05003. ] ()

## 8.4.12 `Frlf_ReconfigLPdu`

**[SWS\_Frlf\_05048]** [

<b>Service name:</b>	<code>Frlf_ReconfigLPdu</code>
<b>Syntax:</b>	<code>Std_ReturnType FrIf_ReconfigLPdu(     uint8 FrIf_CtrlIdx,     uint16 FrIf_LPduIdx,     uint16 FrIf_FrameId,</code>

	Fr_ChannelType FrIf_ChnlIdx, uint8 FrIf_CycleRepetition, uint8 FrIf_CycleOffset, uint8 FrIf_PayloadLength, uint16 FrIf_HeaderCRC )	
<b>Service ID[hex]:</b>	0	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of FlexRay CC within the context of the FlexRay Driver.
	FrIf_LPduldx	This index is used to uniquely identify a FlexRay frame.
	FrIf_FrameId	FlexRay Frame ID the FrIf_LPdu shall be configured to.
	FrIf_ChnlIdx	FlexRay Channel the FrIf_LPdu shall be configured to.
	FrIf_CycleRepetition	Cycle Repetition part of the cycle filter mechanism FrIf_LPdu shall be configured to.
	FrIf_CycleOffset	Cycle Offset part of the cycle filter mechanism FrIf_LPdu shall be configured to.
	FrIf_PayloadLength	Payloadlength in units of bytes the FrIf_LPduldx shall be configured to.
	FrIf_HeaderCRC	Header CRC the FrIf_LPdu shall be configured to.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: API call finished successfully. E_NOT_OK: API call aborted due to errors.
<b>Description:</b>	Calls the FlexRay Driver's API Fr_ReconfigLPdu. The enum value "FR_CHANNEL_AB" shall not be used.	
<b>Available via:</b>	FrIf.h	

┆ ()

**[SWS\_FrIf\_05422]** ┆ The function FrIf\_ReconfigLPdu shall be pre compile time configurable ON/OFF by the configuration parameter FrIfReconfigLPduSupport (derived from configuration parameter FrIfReconfigLPduSupport, see ECUC\_FrIf\_06109) ┆ ()

**[SWS\_FrIf\_05309]** ┆ If parameter FrIf\_CtrlIdx of FrIf\_ReconfigLPdu has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_ReconfigLPdu shall report development error code FRIF\_E\_INV\_CTRL\_IDX to the Det\_ReportError service of the DET module. ┆ ()

**[SWS\_FrIf\_05310]** ┆ If parameter FrIf\_ChnlIdx of FrIf\_ReconfigLPdu has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_ReconfigLPdu shall report development error code FRIF\_E\_INV\_CHNL\_IDX to the Det\_ReportError service of the DET module. ┆ ()

**[SWS\_FrIf\_05311]** ┆ If parameter FrIf\_LPduldx of FrIf\_ReconfigLPdu has an invalid value (i.e. outside of LPdu range or if FrIfReconfigurable of this LPdu is not set to TRUE) and development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the FrIf\_ReconfigLPdu shall report development error code FRIF\_E\_INV\_LPDU\_IDX to the Det\_ReportError service of the DET module. ┆ ()

**[SWS\_FrIf\_05312]** ¶ If parameter `FrIf_FrameId` of `FrIf_ReconfigLPdu` has an invalid value and if development error detection is enabled (i.e. `FrIfDevErrorDetect` equals ON), the `FrIf_ReconfigLPdu` shall report development error code `FRIF_E_INV_FRAME_ID` to the `Det_ReportError` service of the DET module. ¶ ()

### 8.4.13 `FrIf_GetNmVector`

**[SWS\_FrIf\_05016]** ¶

<b>Service name:</b>	<code>FrIf_GetNmVector</code>	
<b>Syntax:</b>	<pre>Std_ReturnType FrIf_GetNmVector(     uint8 FrIf_CtrlIdx,     uint8* FrIf_NmVectorPtr )</pre>	
<b>Service ID[hex]:</b>	0x0f	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	non reentrant for identical values of <code>FrIf_CtrlIdx</code> , reentrant for different values of <code>FrIf_CtrlIdx</code>	
<b>Parameters (in):</b>	<code>FrIf_CtrlIdx</code>	Index of the FlexRay CC to address.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	<code>FrIf_NmVectorPtr</code>	Pointer to a memory location where output value will be stored.
<b>Return value:</b>	<code>Std_ReturnType</code>	<code>E_OK</code> : The call of the FlexRay Driver's API service has returned <code>E_OK</code> . <code>E_NOT_OK</code> : The call of the FlexRay Driver's API service has returned <code>E_NOT_OK</code> , or an error has been detected in development mode.
<b>Description:</b>	Derives the FlexRay NM Vector.	
<b>Available via:</b>	<code>FrIf.h</code>	

¶ ()

**[SWS\_FrIf\_05423]** ¶ The function `FrIf_GetNmVector` shall be pre compile time configurable ON/OFF by the configuration parameter `FrIfGetNmVectorSupport` (derived from configuration parameter `FrIfGetNmVectorSupport`, see `FrIf06100_Conf`) ¶ ()

**[SWS\_FrIf\_05197]** ¶ If parameter `FrIf_CtrlIdx` of `FrIf_GetNmVector` has an invalid value and if development error detection is enabled (i.e. `FrIfDevErrorDetect` equals ON), the function `FrIf_GetNmVector` shall report development error code `FRIF_E_INV_CTRL_IDX` to the `Det_ReportError` service of the DET module. ¶ ()

**[SWS\_FrIf\_05198]** ¶ The function `FrIf_GetNmVector` wraps the FlexRay Driver API `Fr_GetNmVector` function. ¶ ()

**[SWS\_FrIf\_05199]** ¶ Caveats of `FrIf_GetNmVector`: The FlexRay Interface module has to be initialized with a call of `FrIf_Init()` before this API service may be called, see `SWS_FrIf_05003` ¶ ()

### 8.4.14 FrIf\_GetVersionInfo

[SWS\_Frlf\_05002] |

<b>Service name:</b>	Frlf_GetVersionInfo	
<b>Syntax:</b>	<pre>void FrIf_GetVersionInfo(     Std_VersionInfoType* FrIf_VersionInfoPtr )</pre>	
<b>Service ID[hex]:</b>	0x01	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant	
<b>Parameters (in):</b>	None	
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	Frlf_VersionInfoPtr	Pointer to a memory location where the FlexRay Interface version information shall be stored.
<b>Return value:</b>	void	--
<b>Description:</b>	Returns the version information of this module.	
<b>Available via:</b>	FrIf.h	

| (SRS\_BSW\_00407, SRS\_BSW\_00411)

[SWS\_Frlf\_05424] | The function Frlf\_GetVersionInfo shall be pre compile time configurable ON/OFF by the configuration parameter FrlfVersionInfoApi (derived from configuration parameter FrlfVersionInfoApi, see ECUC\_Frlf\_06083) | ()

[SWS\_Frlf\_05151] | If parameter Frlf\_VersionInfoPtr of Frlf\_GetVersionInfo equals NULL\_PTR and if development error detection is enabled (i.e. FrlfDevErrorDetect equals ON), the function Frlf\_GetVersionInfo shall report development error code FRIF\_E\_PARAM\_POINTER to the Det\_ReportError service of the DET module. | ()

### 8.4.15 Frlf\_ReadCCConfig

[SWS\_Frlf\_05313] |

<b>Service name:</b>	Frlf_ReadCCConfig	
<b>Syntax:</b>	<pre>Std_ReturnType FrIf_ReadCCConfig(     uint8 FrIf_CtrlIdx,     uint8 FrIf_ConfigParamIdx,     uint32* FrIf_ConfigParamValuePtr )</pre>	
<b>Service ID[hex]:</b>	0x3b	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant for the same FlexRay CC, reentrant for different FlexRay CCs	
<b>Parameters (in):</b>	Frlf_CtrlIdx	Index of the FlexRay CC to address.
	Frlf_ConfigParamIdx	Index of the configuration parameter to read.
<b>Parameters (inout):</b>	None	

<b>Parameters (out):</b>	FrIf_ConfigParamValuePtr	Pointer to a memory location where output value will be stored.
<b>Return value:</b>	Std_ReturnType	E_OK: The call of the FlexRay Driver's API service has returned E_OK. E_NOT_OK: The call of the FlexRay Driver's API service has returned E_NOT_OK, or an error has been detected in development mode.
<b>Description:</b>	Wraps the FlexRay Driver API function Fr_ReadCCConfig().	
<b>Available via:</b>	FrIf.h	

] ()

**[SWS\_FrIf\_05314]** [The function FrIf\_ReadCCConfig wraps the FlexRay Driver API Fr\_ReadCCConfig function. ] ()

**[SWS\_FrIf\_05315]** [If parameter FrIf\_CtrlIdx of FrIf\_ReadCCConfig has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_ReadCCConfig shall report development error code FRIF\_E\_INV\_CTRL\_IDX to the Det\_ReportError service of the DET module. ] ()

#### 8.4.16 FrIf\_EnableBusMirroring

**[SWS\_FrIf\_05726]** [

<b>Service name:</b>	FrIf_EnableBusMirroring	
<b>Syntax:</b>	Std_ReturnType FrIf_EnableBusMirroring( uint8 FrIf_ClstIdx, boolean FrIf_MirroringActive )	
<b>Service ID[hex]:</b>	0x4b	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant	
<b>Parameters (in):</b>	FrIf_ClstIdx	Index of the FlexRay cluster to address.
	FrIf_MirroringActive	TRUE: Mirror_ReportFlexRayFrame will be called for each frame received or transmitted on the addressed FlexRay CC. FALSE: Mirror_ReportFlexRayFrame will not be called for the addressed FlexRay CC.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: Mirroring mode was changed. E_NOT_OK: Wrong FrIf_CtrlIdx, or mirroring is globally disabled (see FrIfBusMirroringSupport).
<b>Description:</b>	Enables or disables mirroring for all FlexRay controllers connected to the addressed FlexRay cluster.	
<b>Available via:</b>	FrIf.h	

] ()

**[SWS\_FrIF\_05727]** [The function FrIf\_EnableBusMirroring shall be pre compile time configurable ON/OFF by the configuration parameter FrIfBusMirroringSupport (see ECUC\_FrIf\_06124).] ()





## 8.5 Interrupt Service Routines

### 8.5.1 FrIf\_JobListExec\_<FrIfCluster.ShortName>

[SWS\_FrIf\_05040] [

<b>Service name:</b>	FrIf_JobListExec_<FrIfCluster.ShortName>
<b>Syntax:</b>	void FrIf_JobListExec_<FrIfCluster.ShortName>( void )
<b>Service ID[hex]:</b>	0x32
<b>Sync/Async:</b>	Synchronous
<b>Reentrancy:</b>	Non Reentrant
<b>Parameters (in):</b>	None
<b>Parameters (inout):</b>	None
<b>Parameters (out):</b>	None
<b>Return value:</b>	None
<b>Description:</b>	Processes the FlexRay Job List of the FlexRay Cluster with index ClstIdx.
<b>Available via:</b>	FrIf.h

] ()

**Note:**

For a detailed description of this API service, please refer to chapter 7.6.4.2.

[SWS\_FrIf\_05270] [The function FrIf\_JobListExec\_<FrIfCluster.ShortName> shall exist once per FlexRay Cluster of a FlexRay Interface module. ] ()

[SWS\_FrIf\_05271] [The function name of each instance of FrIf\_JobListExec\_<FrIfCluster.ShortName> shall contain the short name of the respective FlexRay Cluster (FrIfCluster).

For each FlexRay Cluster (identified by index ClstIdx), the respective API service FrIf\_JobListExec\_<FrIfCluster.ShortName> must be registered in the AUTOSAR OS as the [ISR](#) of an absolute timer of a FlexRay [CC](#) connected to the FlexRay Cluster with index ClstIdx, if the CC does **not guarantee asynchronous buffer access**. ] ()

Note: If the CC guarantees asynchronous buffer access, the execution of FrIf\_JobListExec\_<FrIfCluster.ShortName> can run in a regular OS task.

[SWS\_FrIf\_05272] [Caveats of FrIf\_JobListExec\_<FrIfCluster.ShortName>: The FlexRay Interface module has to be initialized with a call of FrIf\_Init() before this API service may be called, see SWS\_FrIf\_05003. ] ()

## 8.6 Call-back Notifications

This is a list of functions provided for other modules.

### 8.6.1 FrIf\_CheckWakeupByTransceiver

#### [SWS\_FrIf\_05041] |

<b>Service name:</b>	FrIf_CheckWakeupByTransceiver	
<b>Syntax:</b>	<pre>void FrIf_CheckWakeupByTransceiver(     uint8 FrIf_CtrlIdx,     Fr_ChannelType FrIf_ChnlIdx )</pre>	
<b>Service ID[hex]:</b>	0x39	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of the FlexRay CC to address.
	FrIf_ChnlIdx	Index of the FlexRay Channel to address in scope of the FlexRay controller FrIf_CtrlIdx.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	None	
<b>Description:</b>	Wraps the FlexRay Transceiver Driver API function FrTrcv_CheckWakeupByTransceiver(). The enum value "FR_CHANNEL_AB" shall not be used.	
<b>Available via:</b>	FrIf.h	

| ()

**[SWS\_FrIf\_05274]** | If parameter FrIf\_CtrlIdx of FrIf\_CheckWakeupByTransceiver has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_CheckWakeupByTransceiver shall report development error code FRIF\_E\_INV\_CTRL\_IDX to the Det\_ReportError service of the DET module. | ()

**[SWS\_FrIf\_05275]** | If parameter FrIf\_ChnlIdx of FrIf\_CheckWakeupByTransceiver has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf\_CheckWakeupByTransceiver shall report development error code FRIF\_E\_INV\_CHNL\_IDX to the Det\_ReportError service of the DET module. | ()

**[SWS\_FrIf\_05276]** | The function FrIf\_CheckWakeupByTransceiver shall wrap the FlexRay Transceiver Driver API function FrTrcv\_CheckWakeupByTransceiver() by:

- 1) Translating (based on static [FrIf](#) module configuration) the tuple (FlexRay [CC](#) index FrIf\_CtrlIdx | FlexRay Channel index FrIf\_ChnlIdx) into a tuple (FlexRay Transceiver Driver | Driver-specific Transceiver index FrTrcv\_TrcvIdx).

-2) Calling `FrTrcv_CheckWakeupByTransceiver()` of the determined FlexRay Driver module with the parameters determined as described above. `] ()`

**[SWS\_Frlf\_05277]** [Caveats of `Frlf_CheckWakeupByTransceiver`: The FlexRay Interface module has to be initialized with a call of `Frlf_Init()` before this API service may be called, see `SWS_Frlf_05003`. `] ()`

## 8.7 Scheduled Functions

### 8.7.1 `Frlf_MainFunction_<FrlfCluster.ShortName>`

**[SWS\_Frlf\_05042]** [

<b>Service name:</b>	<code>Frlf_MainFunction_&lt;FrlfCluster.ShortName&gt;</code>
<b>Syntax:</b>	<code>void Frlf_MainFunction_&lt;FrlfCluster.ShortName&gt;(void)</code>
<b>Service ID[hex]:</b>	<code>0x27</code>
<b>Description:</b>	This function will be called cyclically by a task body provided by the BSW Scheduler.
<b>Available via:</b>	<code>SchM_FrIf.h</code>

`] ()`

#### Note:

This cyclically executed API service of the FlexRay Interface serves the following purposes:

- Program the absolute timer interrupt in order to start the execution of `Frlf_JobListExec_<FrlfCluster.ShortName>()` if the CC does not support asynchronous buffer access.
- Monitoring the proper (in time) execution of the `Frlf_JobListExec_<FrlfCluster.ShortName>()` and resynchronize the Joblist if necessary.

Please refer to chapter 7.3 for a detailed description.

Pre condition: The function `Frlf_MainFunction_<FrlfCluster.ShortName>` is cyclically called from a task body provided by the [BSW](#) Scheduler module.

Since the duration of a FlexRay Cycle may be different for two Clusters of an ECU, the calling period (parameter `FrlfMainFunctionPeriod`) of this API service shall be configurable independently for each Cluster [at system configuration time](#).

The parameter `FrlfMainFunctionPeriod` determines for each FlexRay cluster of a FlexRay Interface module the calling period, which is provided for the BSW scheduler module.

**[SWS\_Frlf\_05278]** [The function `Frlf_MainFunction_<FrlfCluster.ShortName>` shall exist once per FlexRay Cluster of a FlexRay Interface module. ] ()

**[SWS\_Frlf\_05279]** [The function name of each instance of `Frlf_MainFunction_<FrlfCluster.ShortName>` shall contain the short name of the respective FlexRay Cluster (`FrlfCluster`). ] ()

## 8.8 Expected Interfaces

This chapter lists all API services required from other [BSW](#) modules.

### 8.8.1 Mandatory Interfaces

This chapter defines all API services which are required from other [BSW](#) modules to fulfill the core functionality of the FlexRay Interface.

**[SWS\_Frlf\_05043]** [

<i>API function</i>	<i>Header File</i>	<i>Description</i>
<code>Det_ReportRuntimeError</code>	<code>Det.h</code>	Service to report runtime errors. If a callout has been configured then this callout shall be called.
<code>Fr_AbortCommunication</code>	<code>Fr.h</code>	Invokes the CC CHI command 'FREEZE'.
<code>Fr_AckAbsoluteTimerIRQ</code>	<code>Fr.h</code>	Resets the interrupt condition of an absolute timer.
<code>Fr_AllowColdstart</code>	<code>Fr.h</code>	Invokes the CC CHI command 'ALLOW_COLDSTART'.
<code>Fr_CancelAbsoluteTimer</code>	<code>Fr.h</code>	Stops an absolute timer.
<code>Fr_CheckTxLPduStatus</code>	<code>Fr.h</code>	Checks the transmit status of the LSdu.
<code>Fr_ControllerInit</code>	<code>Fr.h</code>	Initializes a FlexRay CC.
<code>Fr_DisableAbsoluteTimerIRQ</code>	<code>Fr.h</code>	Disables the interrupt line of an absolute timer.
<code>Fr_EnableAbsoluteTimerIRQ</code>	<code>Fr.h</code>	Enables the interrupt line of an absolute timer.
<code>Fr_GetAbsoluteTimerIRQStatus</code>	<code>Fr.h</code>	Gets IRQ status of an absolute timer.
<code>Fr_GetGlobalTime</code>	<code>Fr.h</code>	Gets the current global FlexRay time.  Important Note: <code>Fr_GetGlobalTime</code> may be called within an exclusive area.
<code>Fr_GetPOCStatus</code>	<code>Fr.h</code>	Gets the POC status.
<code>Fr_HaltCommunication</code>	<code>Fr.h</code>	Invokes the CC CHI command 'DEFERRED_HALT'.
<code>Fr_ReceiveRxLPdu</code>	<code>Fr.h</code>	Receives data from the FlexRay network.
<code>Fr_SendWUP</code>	<code>Fr.h</code>	Invokes the CC CHI command 'WAKEUP'.
<code>Fr_SetAbsoluteTimer</code>	<code>Fr.h</code>	Sets the absolute FlexRay timer.
<code>Fr_SetWakeupChannel</code>	<code>Fr.h</code>	Sets a wakeup channel.
<code>Fr_StartCommunication</code>	<code>Fr.h</code>	Starts communication.
<code>Fr_TransmitTxLPdu</code>	<code>Fr.h</code>	Transmits data on the FlexRay network.
<code>FrTrcv_CheckWakeupByTransceiver</code>	<code>FrTrcv.h</code>	--
<code>FrTrcv_ClearTransceiverWakeup</code>	<code>FrTrcv.h</code>	This function clears a pending wake up event.
<code>FrTrcv_GetTransceiverMode</code>	<code>FrTrcv.h</code>	This function returns the actual state of the transceiver.
<code>FrTrcv_GetTransceiverWUReason</code>	<code>FrTrcv.h</code>	This function returns the wakeup reason.
<code>FrTrcv_SetTransceiverMode</code>	<code>FrTrcv.h</code>	This service sets the transceiver mode.

] ()

## 8.8.2 Optional Interfaces

This chapter defines all API services which are required from other [BSW](#) modules to fulfill an optional functionality of the FlexRay Interface

### [SWS\_Frlf\_05044] |

API function	Header File	Description
Dem_SetEventStatus	Dem.h	Called by SW-Cs or BSW modules to report monitor status information to the Dem. BSW modules calling Dem_SetEventStatus can safely ignore the return value.
Det_ReportError	Det.h	Service to report development errors.
Fr_AllSlots	Fr.h	Invokes the CC CHI command 'ALL_SLOTS'.
Fr_CancelTxLPdu	Fr.h	Cancels the already pending transmission of a LPdu contained in a controllers physical transmit resource (e.g. message buffer).
Fr_DisableLPdu	Fr.h	Disables the hardware resource of a LPdu for transmission/reception.
Fr_GetChannelStatus	Fr.h	Gets the channel status information.
Fr_GetClockCorrection	Fr.h	Gets the current clock correction values. See variables vInterimRateCorrection and vInterimOffsetCorrection of [12] for details.
Fr_GetNmVector	Fr.h	Gets the network management vector of the last communication cycle.
Fr_GetNumOfStartupFrames	Fr.h	Gets the current number of startup frames seen on the cluster. See variable vStartupPairs of [12] for details.
Fr_GetSyncFrameList	Fr.h	Gets a list of syncframes received or transmitted on channel A and channel B via the even and odd communication cycle. See variables vsSyncIdListA and vsSyncIdListB of [12] for details.
Fr_GetWakeupRxStatus	Fr.h	Gets the wakeup received information from the FlexRay controller.
Fr_PrepareLPdu	Fr.h	Prepares a LPdu.
Fr_ReadCCConfig	Fr.h	Reads a FlexRay protocol configuration parameter for a particular FlexRay controller out of the module's configuration.
Fr_ReconfigLPdu	Fr.h	Reconfigures a given LPdu according to the parameters (FrameId, Channel, CycleRepetition, CycleOffset, PayloadLength, HeaderCRC) at runtime.
FrArTp_RxIndication	FrNm_Frlf.h	Indication of a received PDU from a lower layer communication interface module.
FrArTp_TriggerTransmit	FrNm_Frlf.h	Within this API, the upper layer module (called module) shall check whether the available data fits into the buffer size reported by PduInfoPtr->SduLength. If it fits, it shall copy its data into the buffer provided by PduInfoPtr->SduDataPtr and update the length of the actual copied data in PduInfoPtr->SduLength. If not, it returns E_NOT_OK without changing PduInfoPtr.
FrArTp_TxConfirmation	FrTp.h	The lower layer communication interface module confirms the transmission of a PDU, or the failure to transmit a PDU.
FrNm_RxIndication	FrNm_Frlf.h	Indication of a received PDU from a lower layer

		communication interface module.
FrNm_TriggerTransmit	FrNm_Frlf.h	Within this API, the upper layer module (called module) shall check whether the available data fits into the buffer size reported by PduInfoPtr->SduLength. If it fits, it shall copy its data into the buffer provided by PduInfoPtr->SduDataPtr and update the length of the actual copied data in PduInfoPtr->SduLength. If not, it returns E_NOT_OK without changing PduInfoPtr.
FrNm_TxConfirmation	FrTp.h	The lower layer communication interface module confirms the transmission of a PDU, or the failure to transmit a PDU.
FrTp_RxIndication	FrNm_Frlf.h	Indication of a received PDU from a lower layer communication interface module.
FrTp_TriggerTransmit	FrNm_Frlf.h	Within this API, the upper layer module (called module) shall check whether the available data fits into the buffer size reported by PduInfoPtr->SduLength. If it fits, it shall copy its data into the buffer provided by PduInfoPtr->SduDataPtr and update the length of the actual copied data in PduInfoPtr->SduLength. If not, it returns E_NOT_OK without changing PduInfoPtr.
FrTp_TxConfirmation	FrTp.h	The lower layer communication interface module confirms the transmission of a PDU, or the failure to transmit a PDU.
FrTrcv_DisableTransceiverBranch	FrTrcv.h	This function disables the specified branch on the addressed (active star) transceiver.
FrTrcv_EnableTransceiverBranch	FrTrcv.h	This function enables the specified branch on the addressed (active star) transceiver.
FrTrcv_GetTransceiverError	FrTrcv.h	All mandatory errors defined by the FlexRay EPL [5] which are supported by the FlexRay transceiver hardware can be accessed via this API: In addition to errors on the physical layer and local to the ECU hardware, a global error flag is provided.
Mirror_ReportFlexRayFrame	Mirror.h	Reports a received or transmitted FlexRay frame or a Tx conflict.
PduR_FrlfRxIndication	PduR_Frlf.h	Indication of a received PDU from a lower layer communication interface module.
PduR_FrlfTriggerTransmit	PduR_Frlf.h	Within this API, the upper layer module (called module) shall check whether the available data fits into the buffer size reported by PduInfoPtr->SduLength. If it fits, it shall copy its data into the buffer provided by PduInfoPtr->SduDataPtr and update the length of the actual copied data in PduInfoPtr->SduLength. If not, it returns E_NOT_OK without changing PduInfoPtr.
PduR_FrlfTxConfirmation	PduR_Frlf.h	The lower layer communication interface module confirms the transmission of a PDU, or the failure to transmit a PDU.
Xcp_FrlfRxIndication	Xcp.h	Indication of a received PDU from a lower layer communication interface module.
Xcp_FrlfTriggerTransmit	Xcp.h	Within this API, the upper layer module (called module) shall check whether the available data fits into the buffer size reported by PduInfoPtr->SduLength. If it fits, it shall copy its data into the buffer provided by PduInfoPtr->SduDataPtr and update the length of the actual copied data in PduInfoPtr->SduLength. If not, it returns E_NOT_OK without changing

		PduInfoPtr.
Xcp_FrlfTxConfirmation	Xcp.h	The lower layer communication interface module confirms the transmission of a PDU, or the failure to transmit a PDU.

] ()

### 8.8.3 Configurable Interfaces

This chapter lists all interfaces where the target API service of any upper layer, which require one or more of these mentioned interfaces to be called has to be set up by static configuration of the FlexRay Interface. The target function is usually a call-back function. The names of these kinds of interfaces are not fixed because they are configurable.

These call-back services are specified and implemented in the upper layer BSW modules, which use the FlexRay Interface according to [2]. The specific call-back notification is specified in the corresponding AUTOSAR SWS document (see chapter 3).

In addition to upper layer AUTOSAR BSW modules, the FrIf can, with the functionality described within this specification, also support other non-AUTOSAR upper layer software modules (CDs), provided that these modules interact with the FrIf in the same manner as the upper layer AUTOSAR BSW modules. In particular, those non-AUTOSAR modules need to provide APIs as described in this chapter.

**[SWS\_FrIf\_05729]** [Configuration of <UL\_RxIndication>: If the parameter FrIfUserRxIndicationUL is set to FR\_AR\_TP, <UL\_RxIndication> must be FrArTp\_RxIndication.] ()

**[SWS\_FrIf\_05730]** [Configuration of <UL\_RxIndication>: If the parameter FrIfUserRxIndicationUL is set to FR\_NM, <UL\_RxIndication> must be FrNm\_RxIndication.] ()

**[SWS\_FrIf\_05731]** [Configuration of <UL\_RxIndication>: If the parameter FrIfUserRxIndicationUL is set to FR\_TP, <UL\_RxIndication> must be FrTp\_RxIndication.] ()

**[SWS\_FrIf\_05732]** [Configuration of <UL\_RxIndication>: If the parameter FrIfUserRxIndicationUL is set to PDUR, <UL\_RxIndication> must be PduR\_FrIfRxIndication.] ()

**[SWS\_FrIf\_05733]** [Configuration of <UL\_RxIndication>: If the parameter FrIfUserRxIndicationUL is set to XCP, <UL\_RxIndication> must be Xcp\_FrIfRxIndication.] ()

**[SWS\_FrIf\_05434]** [Configuration of <UL\_RxIndication>: If the parameter FrIfUserRxIndicationUL is set to FR\_TSYN, <UL\_RxIndication> must be FrTSyn\_RxIndication.] ()



**[SWS\_Frlf\_05734]** [Configuration of <UL\_TxConfirmation>: If the parameter FrlfUserTxUL is set to FR\_AR\_TP, <UL\_TxConfirmation> must be FrArTp\_TxConfirmation.] ()

**[SWS\_Frlf\_05735]** [Configuration of <UL\_TxConfirmation>: If the parameter FrlfUserTxUL is set to FR\_NM, <UL\_TxConfirmation> must be FrNm\_TxConfirmation.] ()

**[SWS\_Frlf\_05736]** [Configuration of <UL\_TxConfirmation>: If the parameter FrlfUserTxUL is set to FR\_TP, <UL\_TxConfirmation> must be FrTp\_TxConfirmation.] ()

**[SWS\_Frlf\_05737]** [Configuration of <UL\_TxConfirmation>: If the parameter FrlfUserTxUL is set to PDUR, <UL\_TxConfirmation> must be PduR\_FrlfTxConfirmation.] ()

**[SWS\_Frlf\_05738]** [Configuration of <UL\_TxConfirmation>: If the parameter FrlfUserTxUL is set to XCP, <UL\_TxConfirmation> must be Xcp\_FrlfTxConfirmation.] ()

**[SWS\_Frlf\_05739]** [Configuration of <UL\_TriggerTransmit>: If the parameter FrlfUserTxUL is set to FR\_AR\_TP, <UL\_TriggerTransmit> must be FrArTp\_TriggerTransmit. ] ()

**[SWS\_Frlf\_05740]** [Configuration of <UL\_TriggerTransmit>: If the parameter FrlfUserTxUL is set to FR\_NM, <UL\_TriggerTransmit> must be FrNm\_TriggerTransmit. ] ()

**[SWS\_Frlf\_05741]** [Configuration of <UL\_TriggerTransmit>: If the parameter FrlfUserTxUL is set to FR\_TP, <UL\_TriggerTransmit> must be FrTp\_TriggerTransmit. ] ()

**[SWS\_Frlf\_05742]** [Configuration of <UL\_TriggerTransmit>: If the parameter FrlfUserTxUL is set to PDUR, <UL\_TriggerTransmit> must be PduR\_TriggerTransmit. ] ()

**[SWS\_Frlf\_05743]** [Configuration of <UL\_TriggerTransmit>: If the parameter FrlfUserTxUL is set to XCP, <UL\_TriggerTransmit> must be Xcp\_TriggerTransmit. ] ()

**[SWS\_Frlf\_05759]** [Configuration of <UL\_TriggerTransmit>: If the parameter FrlfUserTxUL is set to FR\_TSYN, <UL\_TriggerTransmit> must be FrTSyn\_TriggerTransmit. ] ()

### 8.8.3.1 <UL\_RxIndication>

[SWS\_Frlf\_05045] [

<b>Service name:</b>	<User_RxIndication>	
<b>Syntax:</b>	<pre>void &lt;User_RxIndication&gt;(     PduIdType RxPduId,     const PduInfoType* PduInfoPtr )</pre>	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant for different PduIds. Non reentrant for the same PduId.	
<b>Parameters (in):</b>	RxPduId	ID of the received PDU.
	PduInfoPtr	Contains the length (SduLength) of the received PDU, a pointer to a buffer (SduDataPtr) containing the PDU, and the MetaData related to this PDU.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	None	
<b>Description:</b>	Indication of a received PDU from a lower layer communication interface module.	
<b>Available via:</b>	configurable	

] ()

#### Note:

During the execution of this API service, the upper layer BSW module that is the final recipient of this PDU is expected to retrieve (i.e. copy) the SDU (i.e. the payload of the PDU) by means of the pointer PduInfoPtr which contains the received data address and received data length.

Caveats of <UL\_RxIndication>: This API service is called during the execution of the FlexRay Job List Execution Function.

### 8.8.3.2 <UL\_TxConfirmation>

[SWS\_Frlf\_05046] [

<b>Service name:</b>	<User_TxConfirmation>	
<b>Syntax:</b>	<pre>void &lt;User_TxConfirmation&gt;(     PduIdType TxPduId,     Std_ReturnType result )</pre>	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant for different PduIds. Non reentrant for the same PduId.	
<b>Parameters (in):</b>	TxPduId	ID of the PDU that has been transmitted.
	result	E_OK: The PDU was transmitted. E_NOT_OK: Transmission of the PDU failed.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	None	
<b>Description:</b>	The lower layer communication interface module confirms the transmission of a PDU, or the failure to transmit a PDU.	
<b>Available via:</b>	configurable	

] ()

Caveats of <UL\_TxConfirmation>: This API service is called during the execution of the FlexRay Job List Execution Function.

### 8.8.3.3 <UL\_TriggerTransmit>

[SWS\_Frlf\_05047] |

<b>Service name:</b>	<User_TriggerTransmit>	
<b>Syntax:</b>	Std_ReturnType <User_TriggerTransmit>( PduIdType TxPduId, PduInfoType* PduInfoPtr )	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant for different Pdulds. Non reentrant for the same Pdulld.	
<b>Parameters (in):</b>	TxPdulld	ID of the SDU that is requested to be transmitted.
<b>Parameters (inout):</b>	PduInfoPtr	Contains a pointer to a buffer (SduDataPtr) to where the SDU data shall be copied, and the available buffer size in SduLength. On return, the service will indicate the length of the copied SDU data in SduLength.
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: SDU has been copied and SduLength indicates the number of copied bytes. E_NOT_OK: No SDU data has been copied. PduInfoPtr must not be used since it may contain a NULL pointer or point to invalid data.
<b>Description:</b>	Within this API, the upper layer module (called module) shall check whether the available data fits into the buffer size reported by PduInfoPtr->SduLength. If it fits, it shall copy its data into the buffer provided by PduInfoPtr->SduDataPtr and update the length of the actual copied data in PduInfoPtr->SduLength. If not, it returns E_NOT_OK without changing PduInfoPtr.	
<b>Available via:</b>	configurable	

| ()

Caveats of <UL\_TriggerTransmit>: This API service is called during the execution of the FlexRay Job List Execution Function.

### 8.8.3.4 <Free\_Op\_A>

[SWS\_Frlf\_05316] |

<b>Service name:</b>	<Free_Op_A>	
<b>Syntax:</b>	void <Free_Op_A>( uint8 FrIf_CtrlIdx, uint16 FrIf_LPdulldx )	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant for different FrIf_LPdulldx, non reentrant for same FrIf_LPdulldx	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of the FlexRay CC to address.
	FrIf_LPdulldx	This index is used to uniquely identify a FlexRay frame.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	None	
<b>Description:</b>	User defined communication operation in order to support hardware specific or additional communication controller features to increase performance.	
<b>Available via:</b>	FrIf_Externals.h	

| ()

Caveats of <Free\_Op\_A>: This API service is called during the execution of the FlexRay Job List Execution Function.

### 8.8.3.5 <Free\_Op\_B>

#### [SWS\_FrIf\_05317] |

<b>Service name:</b>	<Free_Op_B>	
<b>Syntax:</b>	void <Free_Op_B>( uint8 FrIf_CtrlIdx, uint16 FrIf_LPduIdx )	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant for different FrIf_LPduIdx, non reentrant for same FrIf_LPduIdx	
<b>Parameters (in):</b>	FrIf_CtrlIdx	Index of the FlexRay CC to address.
	FrIf_LPduIdx	This index is used to uniquely identify a FlexRay frame.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	None	
<b>Description:</b>	User defined communication operation in order to support hardware specific or additional communication controller features to increase performance.	
<b>Available via:</b>	FrIf_Externals.h	

| ()

Caveats of <Free\_Op\_B>: This API service is called during the execution of the FlexRay Job List Execution Function.

### 8.8.3.6 <UL\_TxConflictNotification>

#### [SWS\_FrIf\_91001] |

<b>Service name:</b>	<UL_TxConflictNotification>	
<b>Syntax:</b>	void <UL_TxConflictNotification>( uint8 FrIf_CtrlIdx, uint16 FrIf_LPduIdx )	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant for different FrIf_LPduIdx. Non reentrant for the same FrIf_LPduIdx.	
<b>Parameters (in):</b>	FrIf_CtrlIdx	ID of the addressed FlexRay CC
	FrIf_LPduIdx	ID of the transmitted FlexRay frame
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	None	
<b>Description:</b>	Notification in case a TxConflict has been detected.	
<b>Available via:</b>	FrIf_Externals.h	

| ()

## 9 Sequence Diagrams

The sequence diagrams in this chapter show the basic operations carried out in a FlexRay Cluster's FlexRay Job List Execution Function when executing the various Communication Operations. They also show the interaction of the [Frlf](#) with the upper layer [BSW](#) module and with the underlying FlexRay Driver.

Please note that the sequence diagrams are an extension for illustrational purposes to ease understanding of the specification.

### 9.1 Data Transmission

#### 9.1.1 TransmitWithImmediateBufferAccess

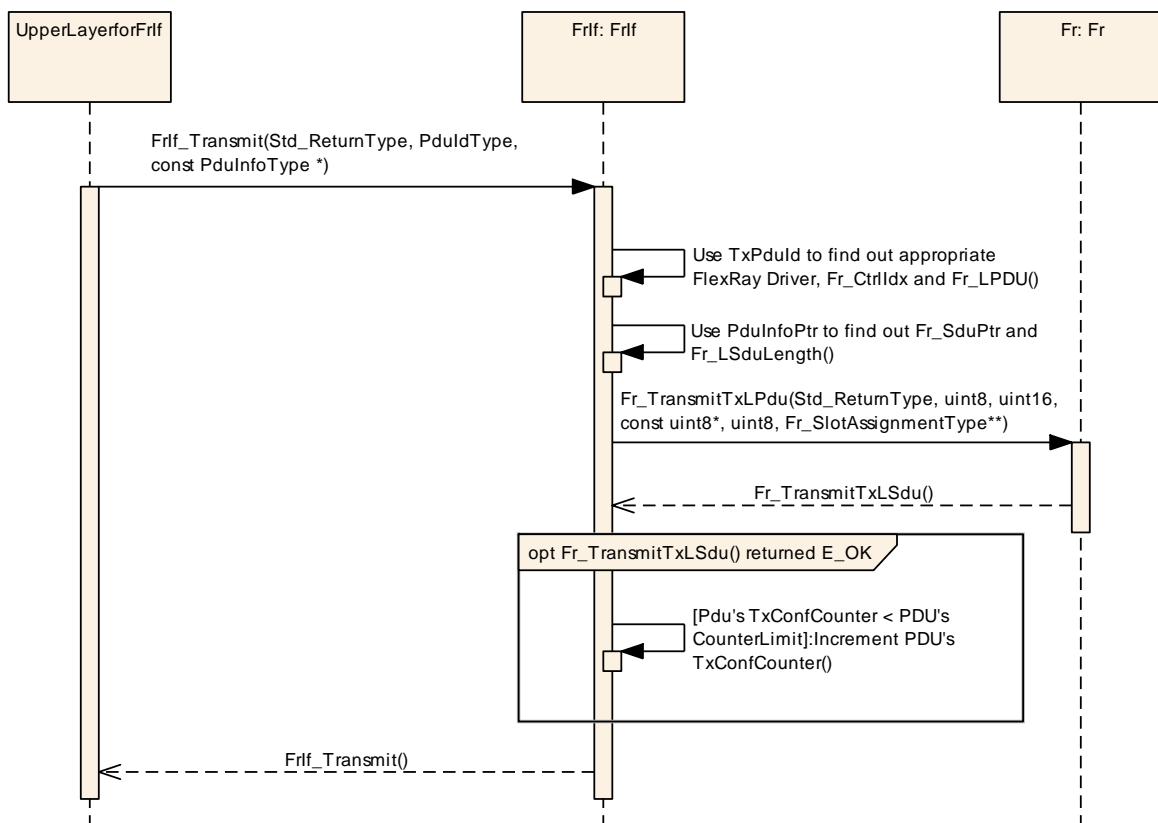


Figure 9-1: TransmitWithImmediateBufferAccess

**9.1.2 TransmitWithDecoupledBufferAccess**

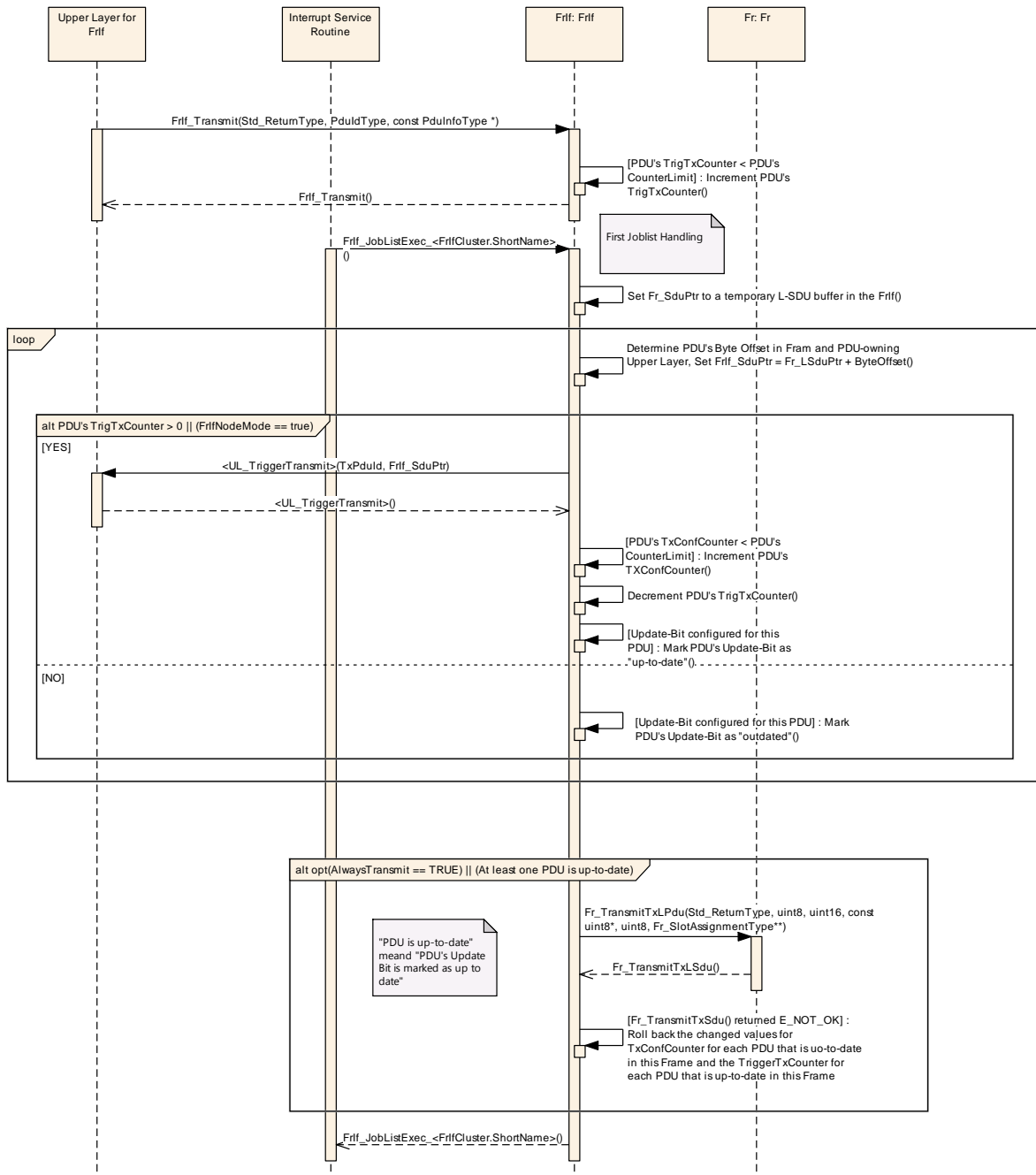


Figure 9-2: TransmitWithDecoupledBufferAccess

**9.1.3 ProvideTxConfirmation**

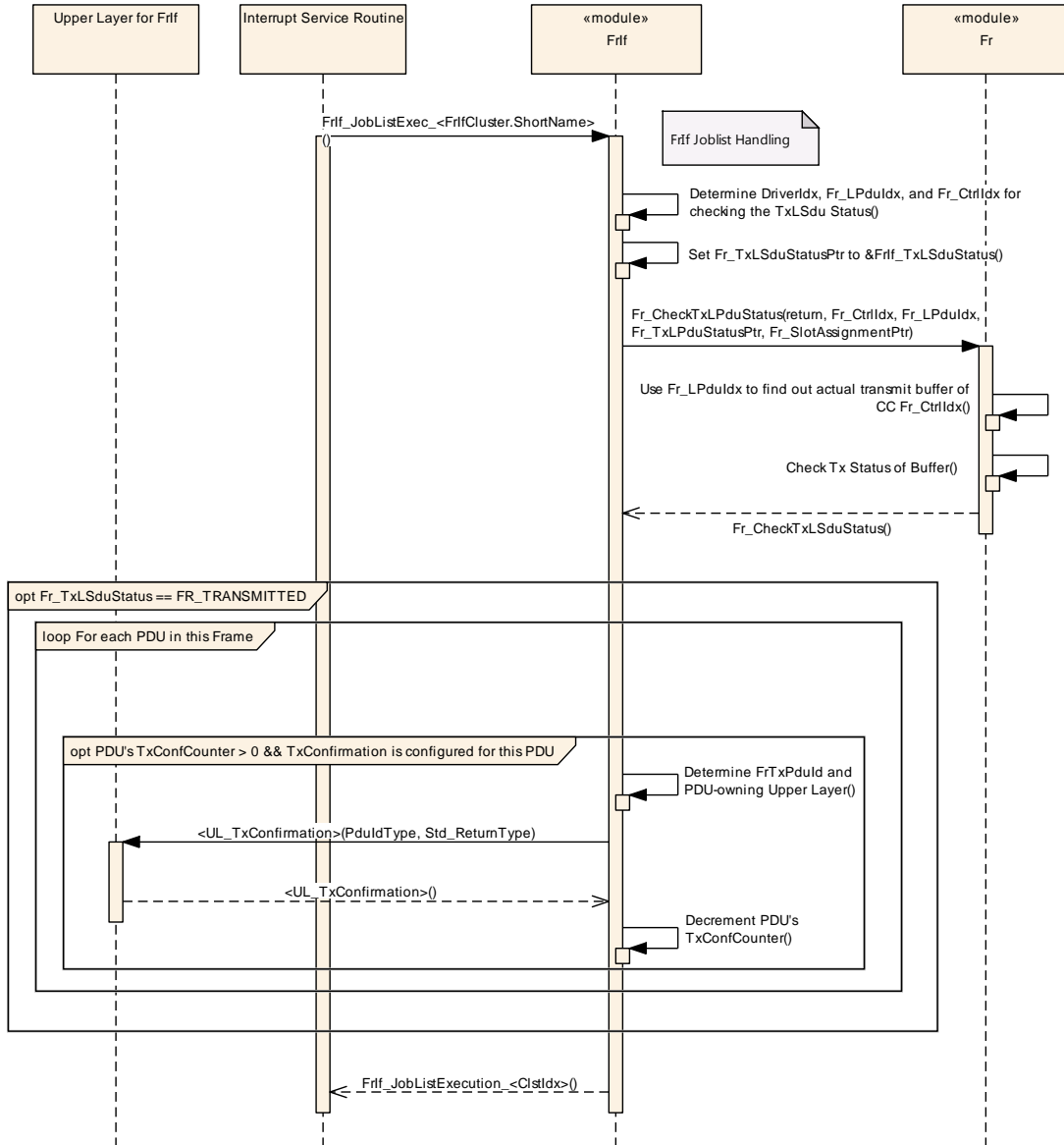


Figure 9-3: ProvideTxConfirmation



## 9.2 Data Reception

### 9.2.1 ReceiveAndIndicate

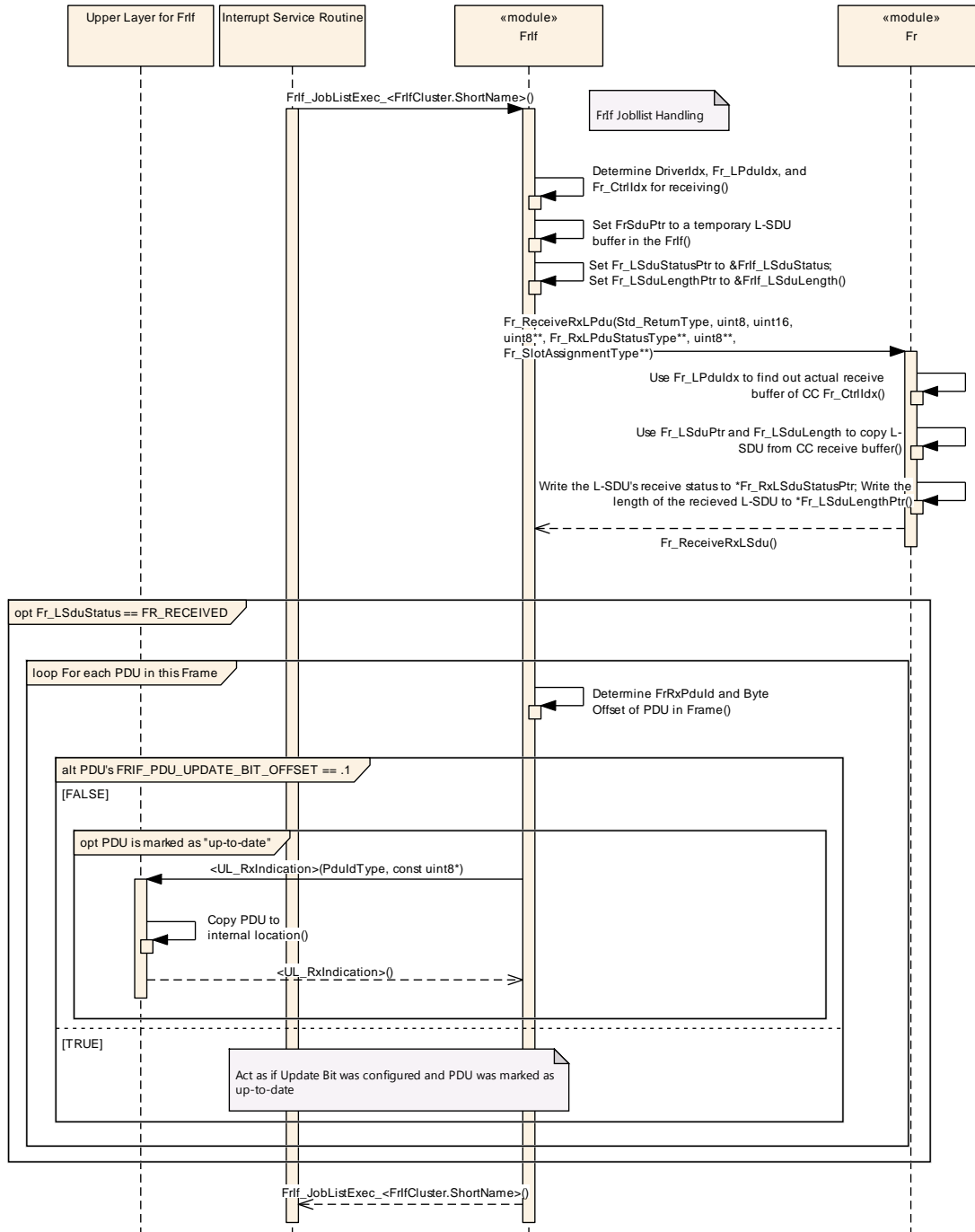


Figure 9-4: ReceiveAndIndicate

9.2.2 ReceiveAndStore

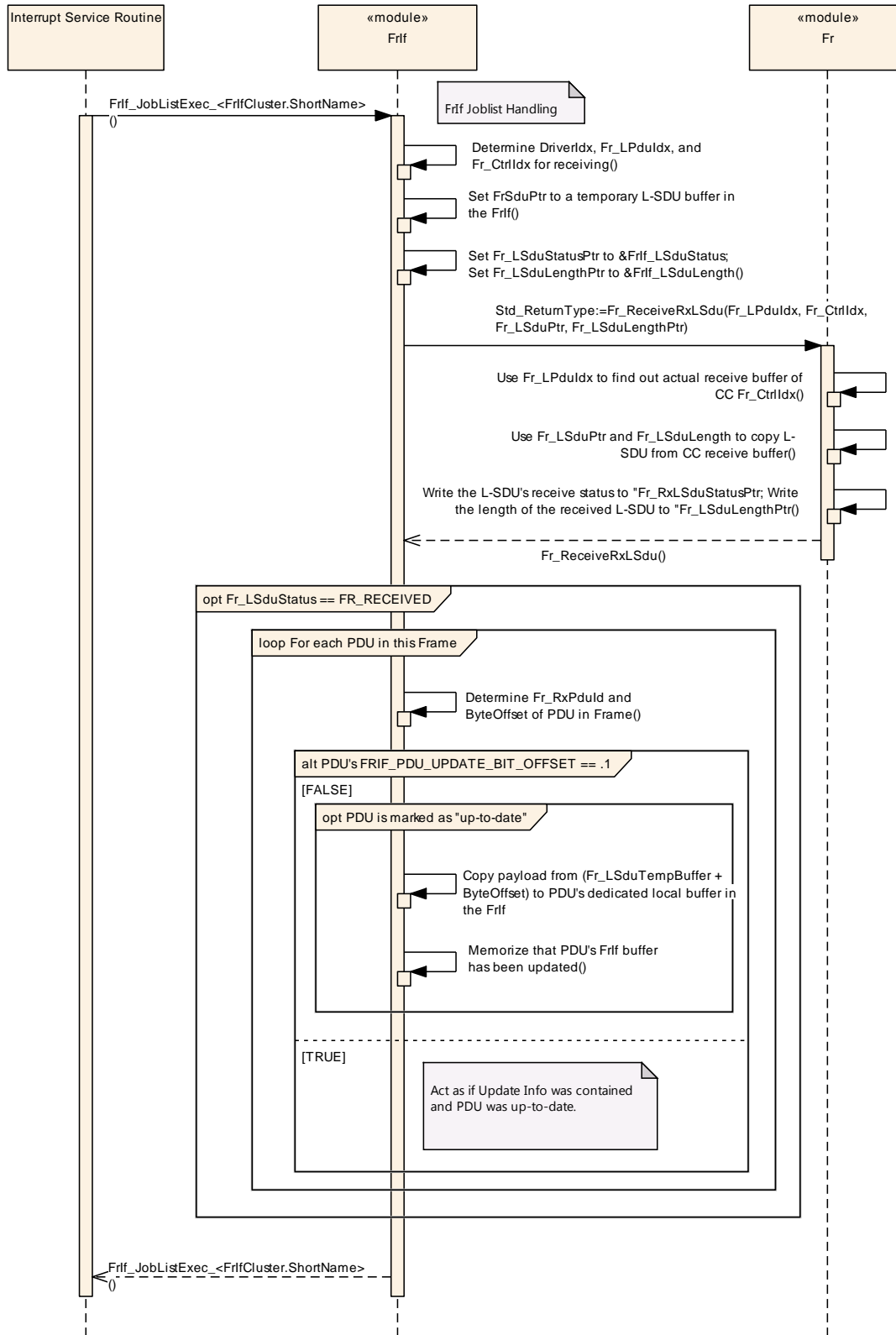


Figure 9-5: ReceiveAndStore

**9.2.3 ProvideRxIndication**

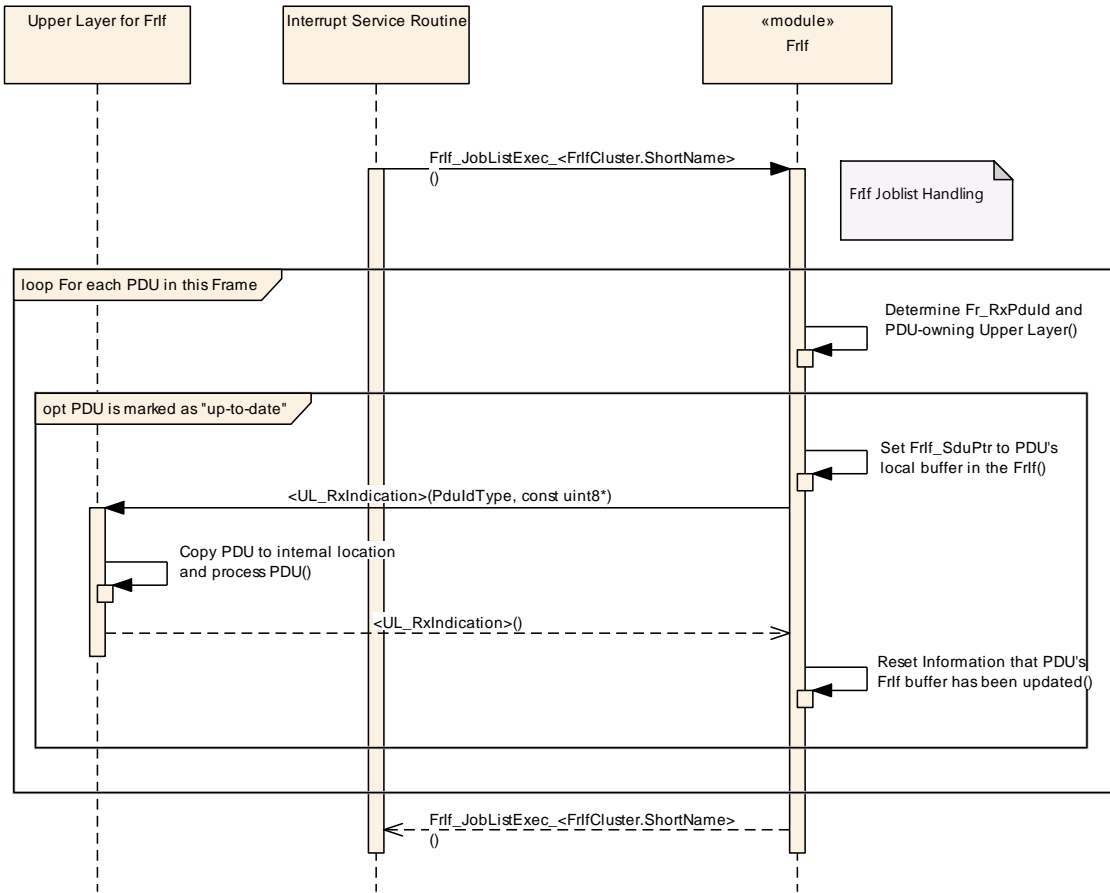


Figure 9-6: ProvideRxIndication



### 9.3 Prepare LPDU

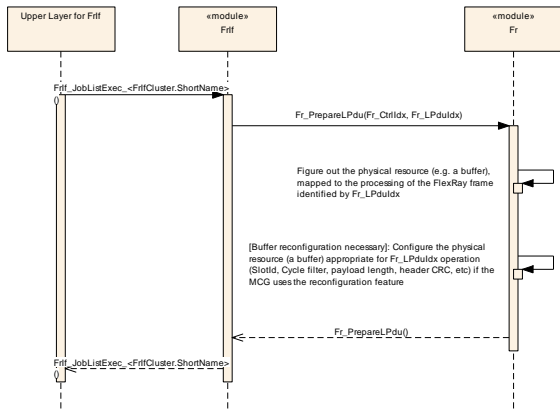


Figure 9-8: Prepare LPdu

## 10 Configuration Specification

This chapter defines configuration parameters and their clustering into containers. Chapter 10.1 gives information to help understanding the subsequent chapters. Chapter 10.2 specifies the structure (containers) and the parameters of the FlexRay Interface. Chapter 9.3 specifies published information of the FlexRay Interface.

### 10.1 How to Read this Chapter

For details refer to the chapter 10.1 “Introduction to configuration specification” in *SWS\_BSWGeneral*.

### 10.2 Containers and Configuration Parameters

The following chapters summarize all configuration parameters. The detailed meanings of the parameters are described in chapter 7 and chapter 8

The listed configuration items can be derived from a network description database, which is based on the EcuConfigurationTemplate. The configuration tool has to extract all information to configure the [Erlf](#) module.

Note:

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

Note:

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

### 10.2.1 FrIf

<b>SWS Item</b>	<b>ECUC_FrIf_06087 :</b>
<b>Module Name</b>	<i>FrIf</i>
<b>Module Description</b>	Configuration of the FrIf (FlexRay Interface) module.
<b>Post-Build Variant Support</b>	true
<b>Supported Config Variants</b>	VARIANT-LINK-TIME, VARIANT-POST-BUILD, VARIANT-PRE-COMPILE

Included Containers		
Container Name	Multiplicity	Scope / Dependency
FrIfConfig	1	This container contains the configuration parameters and sub containers of the AUTOSAR FrIf module.
FrIfGeneral	1	This container contains the general configuration parameters of the FlexRay Interface.

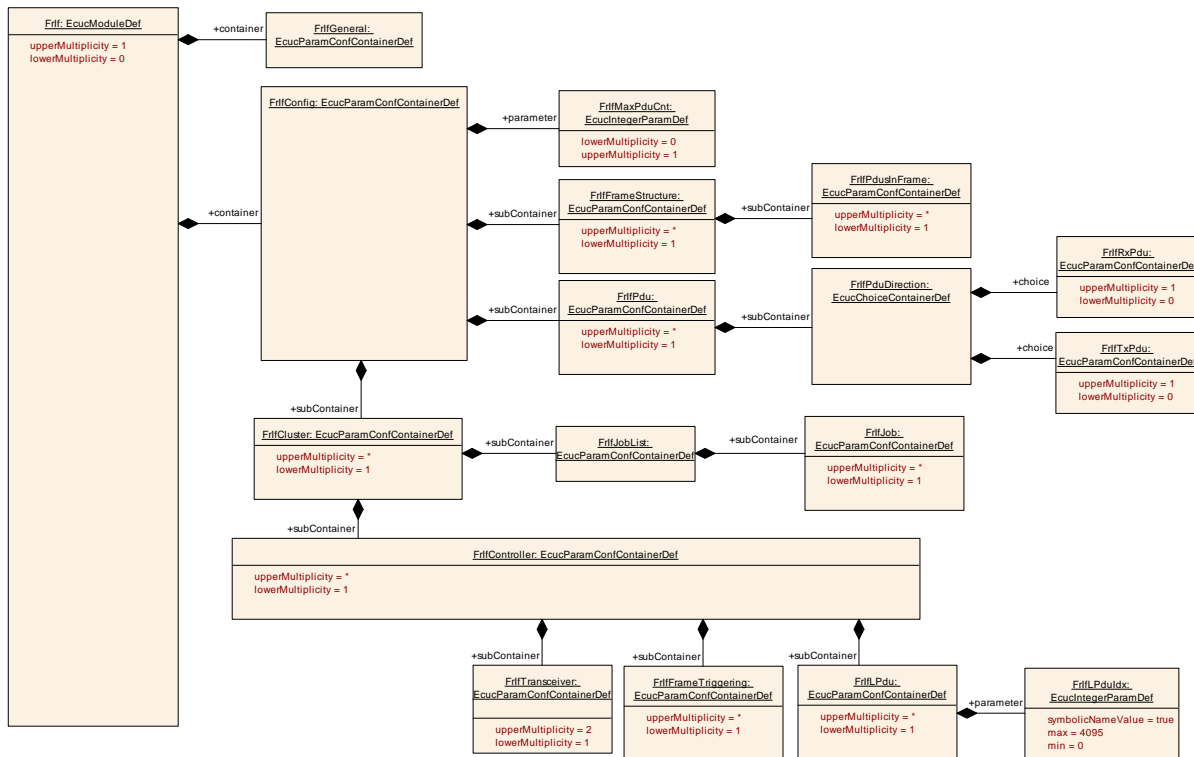


Figure 10-1: FlexRay Interface Module

## 10.2.2 FrlfGeneral

<b>SWS Item</b>	<b>ECUC_Frlf_05360 :</b>		
<b>Container Name</b>	FrlfGeneral		
<b>Description</b>	This container contains the general configuration parameters of the FlexRay Interface.		
<b>Configuration Parameters</b>			

<b>SWS Item</b>	<b>ECUC_Frlf_06112 :</b>		
<b>Name</b>	FrlfAbsTimerIdx		
<b>Parent Container</b>	FrlfGeneral		
<b>Description</b>	Maximum number of supported absolute timers.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	1 .. 15		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06108 :</b>		
<b>Name</b>	FrlfAllSlotsSupport		
<b>Parent Container</b>	FrlfGeneral		
<b>Description</b>	Configuration parameter to enable/disable Frlf support to enable/disable of switching from key-slot / single-slot mode to all slot mode.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06124 :</b>		
<b>Name</b>	FrlfBusMirroringSupport		
<b>Parent Container</b>	FrlfGeneral		
<b>Description</b>	Configuration parameter to enable/disable Frlf support to enable/disable reporting received/transmitted frames to the Bus Mirroring module.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_00002 :</b>		
<b>Name</b>	FrlfCancelTransmitSupport		
<b>Parent Container</b>	FrlfGeneral		



<b>Description</b>	Configuration parameter to enable/disable FrIf support to request the cancellation of the I-PDU transmission to FrDrv.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FrIf_06080 :</b>		
<b>Name</b>	FrIfDevErrorDetect		
<b>Parent Container</b>	FrIfGeneral		
<b>Description</b>	Switches the development error detection and notification on or off. <ul style="list-style-type: none"> <li>• true: detection and notification is enabled.</li> <li>• false: detection and notification is disabled.</li> </ul>		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FrIf_06110 :</b>		
<b>Name</b>	FrIfDisableLPduSupport		
<b>Parent Container</b>	FrIfGeneral		
<b>Description</b>	Configuration parameter to enable/disable FrIf support to disables the hardware resource of a LPdu for transmission/reception.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FrIf_06102 :</b>		
<b>Name</b>	FrIfDisableTransceiverBranchSupport		
<b>Parent Container</b>	FrIfGeneral		
<b>Description</b>	Configuration parameter to enable/disable FrIf support to disable branches of an active star.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FrIf_06103 :</b>		
<b>Name</b>	FrIfEnableTransceiverBranchSupport		
<b>Parent Container</b>	FrIfGeneral		
<b>Description</b>	Configuration parameter to enable/disable FrIf support to enable branches of an active star.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FrIf_06118 :</b>		
<b>Name</b>	FrIfFreeOpAApiName		
<b>Parent Container</b>	FrIfGeneral		
<b>Description</b>	API name that is called when FREE_OP_A is selected as communication operation. See also chapter 8.8.3 Configurable Interfaces.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucStringParamDef		
<b>Default value</b>	--		
<b>maxLength</b>	--		
<b>minLength</b>	--		
<b>regularExpression</b>	--		
<b>Post-Build Variant Multiplicity</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FrIf_06119 :</b>		
<b>Name</b>	FrIfFreeOpBApiName		
<b>Parent Container</b>	FrIfGeneral		
<b>Description</b>	API name that is called when FREE_OP_B is selected as communication operation. See also chapter 8.8.3 Configurable Interfaces.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucStringParamDef		
<b>Default value</b>	--		
<b>maxLength</b>	--		
<b>minLength</b>	--		
<b>regularExpression</b>	--		
<b>Post-Build Variant Multiplicity</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	

<b>Scope / Dependency</b>	scope: local		
<b>SWS Item</b>	<b>ECUC_Frlf_06120 :</b>		
<b>Name</b>	FrlfFreeOpsHeader		
<b>Parent Container</b>	FrlfGeneral		
<b>Description</b>	Defines header file for configurable FREE_OP_A / FREE_OP_B functions.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucStringParamDef		
<b>Default value</b>	--		
<b>maxLength</b>	--		
<b>minLength</b>	--		
<b>regularExpression</b>	--		
<b>Post-Build Variant Multiplicity</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06106 :</b>		
<b>Name</b>	FrlfGetClockCorrectionSupport		
<b>Parent Container</b>	FrlfGeneral		
<b>Description</b>	Configuration parameter to enable/disable Frlf support to enable/disable of polling the FlexRay Driver to getting CC clock correction values.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06105 :</b>		
<b>Name</b>	FrlfGetGetChannelStatusSupport		
<b>Parent Container</b>	FrlfGeneral		
<b>Description</b>	Configuration parameter to enable/disable Frlf support to enable/disable of polling the FlexRay Driver to getting error information about the FlexRay communications bus.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06114 :</b>		
<b>Name</b>	FrlfGetNmVectorSupport		
<b>Parent Container</b>	FrlfGeneral		
<b>Description</b>	Configuration parameter to enable/disable Frlf support to request the		

	FlexRay hardware NMVector.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06104 :</b>		
<b>Name</b>	FrlfGetNumOfStartupFramesSupport		
<b>Parent Container</b>	FrlfGeneral		
<b>Description</b>	Configuration parameter to enable/disable Frlf support to enable/disable of polling the FlexRay Driver for the actual number of received startup frames on the bus.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06107 :</b>		
<b>Name</b>	FrlfGetSyncFrameListSupport		
<b>Parent Container</b>	FrlfGeneral		
<b>Description</b>	Configuration parameter to enable/disable Frlf support to enable/disable of polling the FlexRay Driver to getting a list of actual received sync frames.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06101 :</b>		
<b>Name</b>	FrlfGetTransceiverErrorSupport		
<b>Parent Container</b>	FrlfGeneral		
<b>Description</b>	Configuration parameter to enable/disable Frlf support to get the FlexRay Transceiver errors by calling the FlexRay Transceiver module.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06111 :</b>		
<b>Name</b>	FrlfGetWakeupRxStatusSupport		
<b>Parent Container</b>	FrlfGeneral		

<b>Description</b>	Configuration parameter to enable/disable FrIf support to get the wakeup received information from the FlexRay controller.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FrIf_06081 :</b>		
<b>Name</b>	FrIfNumClstSupported		
<b>Parent Container</b>	FrIfGeneral		
<b>Description</b>	Maximum number of FlexRay Clusters that the FlexRay Interface supports.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	1 .. 15		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FrIf_06082 :</b>		
<b>Name</b>	FrIfNumCtrlSupported		
<b>Parent Container</b>	FrIfGeneral		
<b>Description</b>	Maximum number of FlexRay CCs that the FlexRay Interface supports		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	1 .. 15		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FrIf_06116 :</b>		
<b>Name</b>	FrIfPublicCddHeaderFile		
<b>Parent Container</b>	FrIfGeneral		
<b>Description</b>	Defines header files for callback functions which shall be included in case of CDDs. Range of characters is 1.. 32.		
<b>Multiplicity</b>	0..*		
<b>Type</b>	EcucStringParamDef		
<b>Default value</b>	--		
<b>maxLength</b>	--		
<b>minLength</b>	--		
<b>regularExpression</b>	--		
<b>Post-Build Variant Multiplicity</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	

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

<b>SWS Item</b>	<b>ECUC_Frlf_06117 :</b>		
<b>Name</b>	FrlfReadCCConfigApi		
<b>Parent Container</b>	FrlfGeneral		
<b>Description</b>	Configuration parameter to enable/disable the optional Frlf_ReadCCConfig API.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06109 :</b>		
<b>Name</b>	FrlfReconfigLPduSupport		
<b>Parent Container</b>	FrlfGeneral		
<b>Description</b>	Configuration parameter to enable/disable Frlf support to enable/disable the reconfiguration of a given LPdu according to the parameters (FrameId, Channel, CycleRepetition, CycleOffset, PayloadLength, HeaderCRC) at runtime.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06123 :</b>		
<b>Name</b>	FrlfTxConflictNotificationHeaderName		
<b>Parent Container</b>	FrlfGeneral		
<b>Description</b>	Configuration of the header file name that defines the UL_TxConflictNotification.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucStringParamDef		
<b>Default value</b>	--		
<b>maxLength</b>	--		
<b>minLength</b>	--		
<b>regularExpression</b>	--		
<b>Post-Build Variant Multiplicity</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

	dependency: FrlfTxConflictNotificationName
--	--

<b>SWS Item</b>	<b>ECUC_Frlf_06122 :</b>		
<b>Name</b>	FrlfTxConflictNotificationName		
<b>Parent Container</b>	FrlfGeneral		
<b>Description</b>	Configuration of the API name that is called in case a TxConflict has been detected.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucStringParamDef		
<b>Default value</b>	--		
<b>maxLength</b>	--		
<b>minLength</b>	--		
<b>regularExpression</b>	--		
<b>Post-Build Variant Multiplicity</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local dependency: FrlfTxConflictNotificationHeaderName		

<b>SWS Item</b>	<b>ECUC_Frlf_00001 :</b>		
<b>Name</b>	FrlfUnusedBitValue		
<b>Parent Container</b>	FrlfGeneral		
<b>Description</b>	Set unused bits of transmitted Pdus to a defined value.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 1		
<b>Default value</b>	--		
<b>Post-Build Variant Multiplicity</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06083 :</b>		
<b>Name</b>	FrlfVersionInfoApi		
<b>Parent Container</b>	FrlfGeneral		
<b>Description</b>	Enables/disables the existence of the Frlf_GetVersionInfo() API service true: Frlf_GetVersionInfo() API service exists false: Frlf_GetVersionInfo() API service does not exist		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	

	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

**No Included Containers**

### 10.2.3 FrlfCluster

<b>SWS Item</b>	<b>ECUC_Frlf_05366 :</b>		
<b>Container Name</b>	FrlfCluster		
<b>Description</b>	This container specifies a Frlf Cluster and all related data which is required to enable communication of the Cluster. A Cluster may consist of more than one Controller.		
<b>Post-Build Variant Multiplicity</b>	false		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE, VARIANT-LINK-TIME, VARIANT-POST-BUILD
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Configuration Parameters</b>			

<b>SWS Item</b>	<b>ECUC_Frlf_06002 :</b>		
<b>Name</b>	FrlfClstIdx		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	This parameter provides a zero-based consecutive index of the FlexRay Clusters. Upper layer BSW modules and the Frlf itself use this index to identify a FlexRay Cluster.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
<b>Range</b>	0 .. 63		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_00003 :</b>		
<b>Name</b>	FrlfDetectNITError		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Indicates whether NIT error status of each cluster shall be detected or not.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06006 :</b>		
<b>Name</b>	FrlfGChannels		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	The channels that are used by the cluster.		



	Implementation Type: Fr_ChannelType		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucEnumerationParamDef		
<b>Range</b>	FR_CHANNEL_A	Cluster uses channel A	
	FR_CHANNEL_AB	Cluster uses channel A and B	
	FR_CHANNEL_B	Cluster uses channel B	
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06008 :</b>		
<b>Name</b>	FrlfGColdStartAttempts		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Maximum number of times a node in the cluster is permitted to attempt to start the cluster by initiating schedule synchronization		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	2 .. 31		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06086 :</b>		
<b>Name</b>	FrlfGCycleCountMax		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Maximum cycle counter value in a given cluster. Remark: Set to 63 for FlexRay Protocol 2.1 Rev. A compliance.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	7 .. 63		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06020 :</b>		
<b>Name</b>	FrlfGdActionPointOffset		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Number of macroticks the action point is offset from the beginning of a static slot.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	1 .. 63		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		

<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: ECU		

<b>SWS Item</b>	<b>ECUC_FrIf_06021 :</b>		
<b>Name</b>	FrlfGdBit		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Nominal bit time in seconds		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucEnumerationParamDef		
<b>Range</b>	T100NS	--	
	T200NS	--	
	T400NS	--	
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FrIf_06024 :</b>		
<b>Name</b>	FrlfGdCasRxLowMax		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Upper limit of the CAS acceptance windows [gdBit] Remark: Range 67 to 99 for FlexRay Protocol 2.1 Rev. A compliance		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	28 .. 254		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FrIf_06025 :</b>		
<b>Name</b>	FrlfGdCycle		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Length of the cycle, expressed in [s] Remark: Lower limit 0.000024 for FlexRay Protocol 3.0 compliance.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucFloatParamDef		
<b>Range</b>	[2.4E-5 .. 0.016]		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FrIf_06026 :</b>		
<b>Name</b>	FrlfGdDynamicSlotIdlePhase		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Duration of the idle phase within a dynamic slot [Minislots].		

<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 2		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_00012 :</b>		
<b>Name</b>	FrlfGdIgnoreAfterTx		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Duration for which the bitstrobing is paused after transmission [gdBit]. Remark: Set to 0 for FlexRay Protocol 2.1 Rev. A compliance.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 15		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06027 :</b>		
<b>Name</b>	FrlfGdMacrotick		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Duration of the cluster wide nominal macrotick, expressed in s		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucFloatParamDef		
<b>Range</b>	[1E-6 .. 6E-6]		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06033 :</b>		
<b>Name</b>	FrlfGdMinislot		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Duration of a minislot [Macroticks]		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	2 .. 63		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: ECU		

<b>SWS Item</b>	<b>ECUC_Frlf_06032 :</b>		
<b>Name</b>	FrlfGdMiniSlotActionPointOffset		

<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Number of MacroTicks the Minislot action point is offset from the beginning of a Minislot [MacroTicks].		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	1 .. 31		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06034 :</b>		
<b>Name</b>	FrlfGdNit		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Duration of the Network Idle Time [MacroTicks] Remark: Upper limit 805 for FlexRay Protocol 2.1 Rev. A compliance.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	2 .. 15978		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06035 :</b>		
<b>Name</b>	FrlfGdSampleClockPeriod		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Sample clock period		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucEnumerationParamDef		
<b>Range</b>	T12_5NS	--	
	T25NS	--	
	T50NS	--	
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06036 :</b>		
<b>Name</b>	FrlfGdStaticSlot		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Duration of a static slot [MacroTicks]. Remark: Range 4-661 for FlexRay Protocol 2.1 Rev. A compliance.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	3 .. 664		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE

	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FrIf_06037 :</b>		
<b>Name</b>	FrlfGdSymbolWindow		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Duration of the symbol window [Macroticks]. Remark: Range 0-142 for FlexRay Protocol 2.1 Rev. A compliance.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcuIntegerParamDef		
<b>Range</b>	0 .. 162		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FrIf_00011 :</b>		
<b>Name</b>	FrlfGdSymbolWindowActionPointOffset		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Number of macroticks the action point offset is from the beginning of the symbol window [Macroticks]. Remark: Set to GdActionPointOffset for FlexRay Protocol 2.1 Rev. A compliance.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcuIntegerParamDef		
<b>Range</b>	1 .. 63		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FrIf_06038 :</b>		
<b>Name</b>	FrlfGdTSSTransmitter		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Number of bits in the Transmission Start Sequence [gdBits]. Remark: Lower limit 3 for FlexRay Protocol 2.1 Rev. A compliance.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcuIntegerParamDef		
<b>Range</b>	1 .. 15		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FrIf_06039 :</b>		
<b>Name</b>	FrlfGdWakeupRxIdle		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Number of bits used by the node to test the duration of the 'idle' or HIGH phase of a received wakeup [gdBit]. Remarks: This parameter maps to		

	FlexRay Protocol 2.1 Rev. A parameter gdWakeupSymbolRxIdle. Lower limit 14 for FlexRay Protocol 2.1 Rev. A compliance.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	8 .. 59		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06040 :</b>		
<b>Name</b>	FrlfGdWakeupRxLow		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Number of bits used by the node to test the duration of the LOW phase of a received wakeup [gdBit]. Remarks: This parameter maps to FlexRay Protocol 2.1 Rev. A parameter gdWakeupSymbolRxLow. Lower limit 11 for FlexRay Protocol 2.1 Rev. A compliance.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	8 .. 59		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06041 :</b>		
<b>Name</b>	FrlfGdWakeupRxWindow		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	The size of the window used to detect wakeups [gdBit]. Remarks: This parameter maps to FlexRay Protocol 2.1 Rev. A parameter gdWakeupSymbolRxWindow. Upper limit 301 for FlexRay Protocol 2.1 Rev. A compliance.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	76 .. 485		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06043 :</b>		
<b>Name</b>	FrlfGdWakeupTxActive		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Number of bits used by the node to transmit the LOW phase of awakeup symbol and the HIGH and LOW phases of a WUDOP [gdBit]. Remarks: This parameter maps to FlexRay Protocol 2.1 Rev. A parameter gdWakeupSymbolTxLow.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		

<b>Range</b>	15 .. 60		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FrIf_06042 :</b>		
<b>Name</b>	FrlfGdWakeupTxIdle		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Number of bits used by the node to transmit the 'idle' part of a wakeup symbol [gdBit].  Remarks: This parameter maps to FlexRay Protocol 2.1 Rev. A parameter gdWakeupSymbolTxIdle.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcuIntegerParamDef		
<b>Range</b>	45 .. 180		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FrIf_06009 :</b>		
<b>Name</b>	FrlfGListenNoise		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	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 node parameter pdListenTimeout.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcuIntegerParamDef		
<b>Range</b>	2 .. 16		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FrIf_06010 :</b>		
<b>Name</b>	FrlfGMacroPerCycle		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Number of macroticks in a communication cycle. Note: Lower limit 10 for FlexRay Protocol 2.1 Rev. A compliance		
<b>Multiplicity</b>	1		
<b>Type</b>	EcuIntegerParamDef		
<b>Range</b>	8 .. 16000		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06011 :</b>		
<b>Name</b>	FrlfGMaxWithoutClockCorrectFatal		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	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].		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	1 .. 15		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06012 :</b>		
<b>Name</b>	FrlfGMaxWithoutClockCorrectPassive		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	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]		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	1 .. 15		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06013 :</b>		
<b>Name</b>	FrlfGNetworkManagementVectorLength		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Length of the Network Management vector in a cluster [bytes]		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 12		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06014 :</b>		
<b>Name</b>	FrlfGNumberOfMinislots		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Number of minislots in the dynamic segment Remark: Upper limit 7986 for FlexRay Protocol 2.1 Rev. A compliance		
<b>Multiplicity</b>	1		



<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 7988		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06015 :</b>		
<b>Name</b>	FrlfGNumberOfStaticSlots		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Number of static slots in the static segment		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	2 .. 1023		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06018 :</b>		
<b>Name</b>	FrlfGPayloadLengthStatic		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Payload length of a static frame [16 bit words]		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 127		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06019 :</b>		
<b>Name</b>	FrlfGSyncFrameIDCountMax		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Maximum number of distinct syncframe identifiers present in a given cluster. This parameter maps to FlexRay Protocol 2.1 Rev. A parameter gSyncNodeMax.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	2 .. 15		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06003 :</b>		
<b>Name</b>	FrlfMainFunctionPeriod		
<b>Parent Container</b>	FrlfCluster		

<b>Description</b>	The execution cycle of the Frlf_MainFunction_<FrlfCluster.ShortName>() in seconds. The Frlf does not require this information but the BSW scheduler, which invokes the cluster main functions, needs it in order to plan its tasks.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucFloatParamDef		
<b>Range</b>	]0 .. INF[		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_00004 :</b>		
<b>Name</b>	FrlfSafetyMargin		
<b>Parent Container</b>	FrlfCluster		
<b>Description</b>	Additional timespan in macroticks which takes jitter into account to be able to set the JobListPointer to the next possible job which can be executed in case the FlexRay Job List Execution Function has been resynchronized.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 1024000		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>Included Containers</b>		
<b>Container Name</b>	<b>Multiplicity</b>	<b>Scope / Dependency</b>
FrlfClusterDemEventParameterReferences	0..1	Container for the references to DemEventParameter elements which shall be invoked using the API Dem_SetEventStatus in case the corresponding error occurs. The EventId is taken from the referenced DemEventParameter's DemEventId symbolic value. The standardized errors are provided in this container and can be extended by vendor-specific error references.
FrlfController	1..*	This container contains the configuration of FlexRay CC.
FrlfJobList	1	This container specifies a list of all FlexRay Jobs of the Cluster to be performed by Frlf_JobListExec_<FrlfCluster.ShortName>().

## 10.2.4 FrlfController

<b>SWS Item</b>	<b>ECUC_Frlf_05363 :</b>
<b>Container Name</b>	FrlfController
<b>Description</b>	This container contains the configuration of FlexRay CC.
<b>Post-Build Variant Multiplicity</b>	false

<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE, VARIANT-LINK-TIME, VARIANT-POST-BUILD
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Configuration Parameters</b>			

<b>SWS Item</b>	<b>ECUC_FrIf_06045 :</b>		
<b>Name</b>	FrlfCtrlIdx		
<b>Parent Container</b>	FrlfController		
<b>Description</b>	This parameter provides a zero-based consecutive index of the FlexRay Communication Controllers. Upper layer BSW modules and the Frlf itself use this index to identify a FlexRay CC.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
<b>Range</b>	0 .. 31		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: ECU		

<b>SWS Item</b>	<b>ECUC_FrIf_06044 :</b>		
<b>Name</b>	FrlfFrCtrlRef		
<b>Parent Container</b>	FrlfController		
<b>Description</b>	Reference to a Controller, which is handled by a specific Driver. This reference is unique for the ECU.		
<b>Multiplicity</b>	1		
<b>Type</b>	Symbolic name reference to [ FrController ]		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: ECU		

<b>Included Containers</b>		
<b>Container Name</b>	<b>Multiplicity</b>	<b>Scope / Dependency</b>
FrlfFrameTriggering	1..*	A Frame triggering contains the communication parameters of the FlexRay Frame as well as a reference to the Frame Construction Plan.
FrlfLPdu	1..*	Reference to a L-PDU index
FrlfTransceiver	1..2	Up to two FlexRay Transceivers may connect a Controller to a Cluster. This container realizes a Controller-Transceiver assignment.

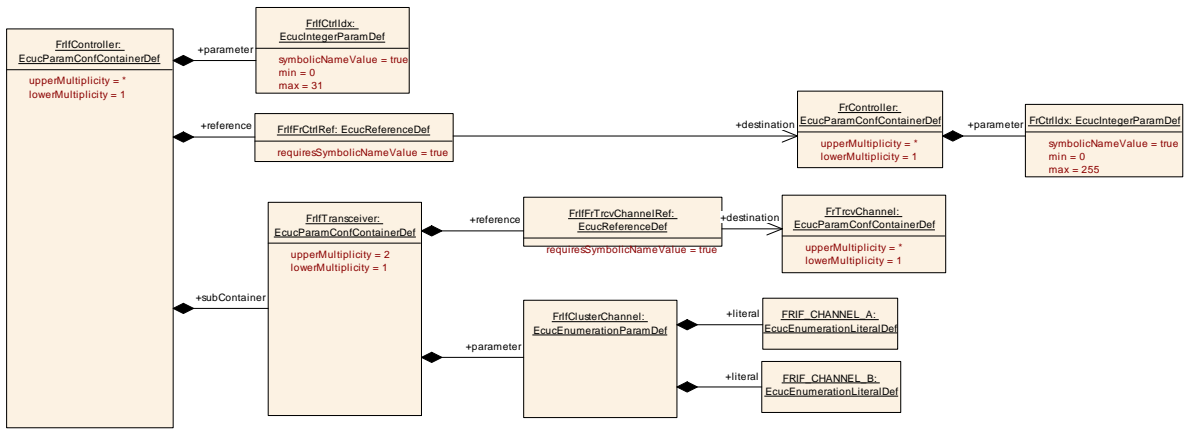


Figure 10-2: FlexRay Interface Controller (hardware reference)

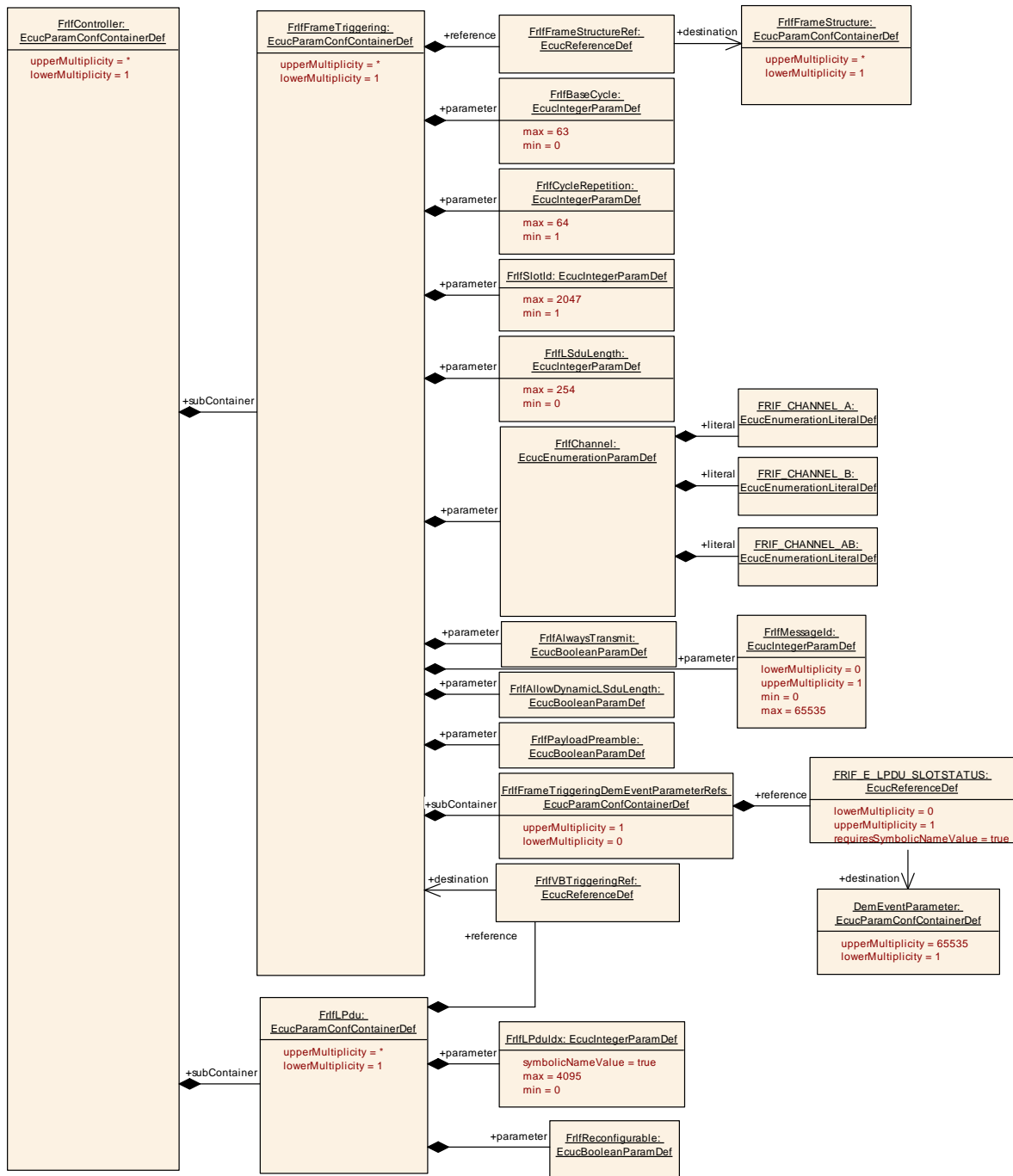


Figure 10-3: FlexRay Interface Controller (data reference)

### 10.2.5 FrIfTransceiver

<b>SWS Item</b>	<b>ECUC_FrIf_05391 :</b>
<b>Container Name</b>	FrIfTransceiver
<b>Description</b>	Up to two FlexRay Transceivers may connect a Controller to a Cluster. This container realizes a Controller-Transceiver assignment.
<b>Configuration Parameters</b>	

<b>SWS Item</b>	<b>ECUC_Frlf_06062 :</b>		
<b>Name</b>	FrlfClusterChannel		
<b>Parent Container</b>	FrlfTransceiver		
<b>Description</b>	This parameter identifies to which one of the two Channels (A, B, A and B) of the Cluster the Transceiver is connected. FrlfClusterChannel shall map to Fr_ChannelType: FRIF_CHANNEL_A == FR_CHANNEL_A FRIF_CHANNEL_B == FR_CHANNEL_B FR_CHANNEL_AB shall not be used.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucEnumerationParamDef		
<b>Range</b>	FRIF_CHANNEL_A	Channel A	
	FRIF_CHANNEL_B	Channel B	
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06061 :</b>		
<b>Name</b>	FrlfFrTrcvChannelRef		
<b>Parent Container</b>	FrlfTransceiver		
<b>Description</b>	Reference to a Transceiver Driver Channel. This reference is unique for the ECU.		
<b>Multiplicity</b>	1		
<b>Type</b>	Symbolic name reference to [ FrTrcvChannel ]		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: ECU		

**No Included Containers**

### 10.2.6 FrlfLPdu

<b>SWS Item</b>	<b>ECUC_Frlf_05364 :</b>		
<b>Container Name</b>	FrlfLPdu		
<b>Description</b>	Reference to a L-PDU index		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Configuration Parameters</b>			

<b>SWS Item</b>	<b>ECUC_Frlf_06058 :</b>		
<b>Name</b>	FrlfLPduldx		
<b>Parent Container</b>	FrlfLPdu		
<b>Description</b>	This parameter identifies the L-PDU in the interaction between FlexRay Interface and FlexRay Driver.		
<b>Multiplicity</b>	1		

<b>Type</b>	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
<b>Range</b>	0 .. 4095		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_00008 :</b>		
<b>Name</b>	FrlfReconfigurable		
<b>Parent Container</b>	FrlfLPdu		
<b>Description</b>	This parameter specifies that this LPdu is reconfigurable using Frlf_ReconfigLPdu. This means that this LPdu can be assigned to a different FrameTriggering at runtime. However, this reconfiguration is limited by hardware constraints. The direction of the LPdu cannot be reconfigured.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06057 :</b>		
<b>Name</b>	FrlfVBTriggeringRef		
<b>Parent Container</b>	FrlfLPdu		
<b>Description</b>	Reference to the assigned Frame triggering.		
<b>Multiplicity</b>	1		
<b>Type</b>	Reference to [ FrlfFrameTriggering ]		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

**No Included Containers**

### 10.2.7 FrlfFrameTriggering

<b>SWS Item</b>	<b>ECUC_Frlf_06090 :</b>		
<b>Container Name</b>	FrlfFrameTriggering		
<b>Description</b>	A Frame triggering contains the communication parameters of the FlexRay Frame as well as a reference to the Frame Construction Plan.		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Configuration Parameters</b>			

<b>SWS Item</b>	<b>ECUC_FrIf_06049 :</b>		
<b>Name</b>	FrIfAllowDynamicLSduLength		
<b>Parent Container</b>	FrIfFrameTriggering		
<b>Description</b>	Allows L-PDU length reduction ('FrIfLSduLength' defines max. length) and indicates that the related CC buffer has to be reconfigured for the actual length and Header-CRC before transmission of the L-PDU.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FrIf_00013 :</b>		
<b>Name</b>	FrIfAlwaysTransmit		
<b>Parent Container</b>	FrIfFrameTriggering		
<b>Description</b>	Defines whether the driver's API function Fr_TransmitTxLPdu() shall always be called for this L-PDU.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FrIf_06051 :</b>		
<b>Name</b>	FrIfBaseCycle		
<b>Parent Container</b>	FrIfFrameTriggering		
<b>Description</b>	This parameter contains the FlexRay Base Cycle used to transmit this FlexRay Frame.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 63		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FrIf_06052 :</b>		
<b>Name</b>	FrIfChannel		
<b>Parent Container</b>	FrIfFrameTriggering		
<b>Description</b>	This parameter contains the FlexRay Channel used to transmit this FlexRay Frame.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucEnumerationParamDef		
<b>Range</b>	FRIF_CHANNEL_A		Channel A
	FRIF_CHANNEL_AB		Channel A and B
	FRIF_CHANNEL_B		Channel B
<b>Post-Build Variant Value</b>	true		
<b>Value</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE



<b>Configuration Class</b>	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FrIf_06053 :</b>		
<b>Name</b>	FrlfCycleRepetition		
<b>Parent Container</b>	FrlfFrameTriggering		
<b>Description</b>	This parameter contains the FlexRay Cycle Repetition used to transmit this FlexRay Frame. Possible values for FlexRay Protocol version 2.1: 1,2,4,8,16,32,64 Possible values for FlexRay Protocol version 3.0: 1,2,4,5,8,10,16,20,32,40,50,64		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	1 .. 64		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FrIf_06054 :</b>		
<b>Name</b>	FrlfLsduLength		
<b>Parent Container</b>	FrlfFrameTriggering		
<b>Description</b>	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].		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 254		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local dependency: The parameter depends on the low level parameters of the FlexRay CC.		

<b>SWS Item</b>	<b>ECUC_FrIf_00010 :</b>		
<b>Name</b>	FrlfMessageld		
<b>Parent Container</b>	FrlfFrameTriggering		
<b>Description</b>	The first two bytes of the payload segment of the FlexRay frame format for frames transmitted in the dynamic segment can be used as receiver filterable data called the message ID.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 65535		
<b>Default value</b>	--		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE

<b>Class</b>	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06055 :</b>		
<b>Name</b>	FrlfPayloadPreamble		
<b>Parent Container</b>	FrlfFrameTriggering		
<b>Description</b>	Switching the Payload Preamble bit.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06056 :</b>		
<b>Name</b>	FrlfSlotId		
<b>Parent Container</b>	FrlfFrameTriggering		
<b>Description</b>	This parameter contains the FlexRay Slot ID used to transmit this FlexRay Frame.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	1 .. 2047		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06048 :</b>		
<b>Name</b>	FrlfFrameStructureRef		
<b>Parent Container</b>	FrlfFrameTriggering		
<b>Description</b>	Reference to the Construction Plan of the FlexRay Frame.		
<b>Multiplicity</b>	1		
<b>Type</b>	Reference to [ FrlfFrameStructure ]		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>Included Containers</b>		
<b>Container Name</b>	<b>Multiplicity</b>	<b>Scope / Dependency</b>
FrlfFrameTriggeringDemEventParameterRefs	0..1	Container for the references to DemEventParameter elements which shall be invoked using the API Dem_SetEventStatus in case the corresponding error occurs. The EventId is taken from the referenced DemEventParameter's DemEventId symbolic value. The standardized errors are provided in

		this container and can be extended by vendor-specific error references.
--	--	---

### 10.2.8 FrlfJobList

<b>SWS Item</b>	<b>ECUC_Frlf_05367 :</b>
<b>Container Name</b>	FrlfJobList
<b>Description</b>	This container specifies a list of all FlexRay Jobs of the Cluster to be performed by Frlf_JobListExec_<FrlfCluster.ShortName>().
<b>Configuration Parameters</b>	

<b>SWS Item</b>	<b>ECUC_Frlf_06063 :</b>		
<b>Name</b>	FrlfAbsTimerRef		
<b>Parent Container</b>	FrlfJobList		
<b>Description</b>	Reference to the absolute timer to be used to trigger the interrupt whose ISR contains the Frlf_JobListExec_<FrlfCluster.ShortName>() function.		
<b>Multiplicity</b>	1		
<b>Type</b>	Symbolic name reference to [ FrAbsoluteTimer ]		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>Included Containers</b>		
<b>Container Name</b>	<b>Multiplicity</b>	<b>Scope / Dependency</b>
FrlfJob	1..*	A job may contain more than one operation that are executed at a specific point in time.

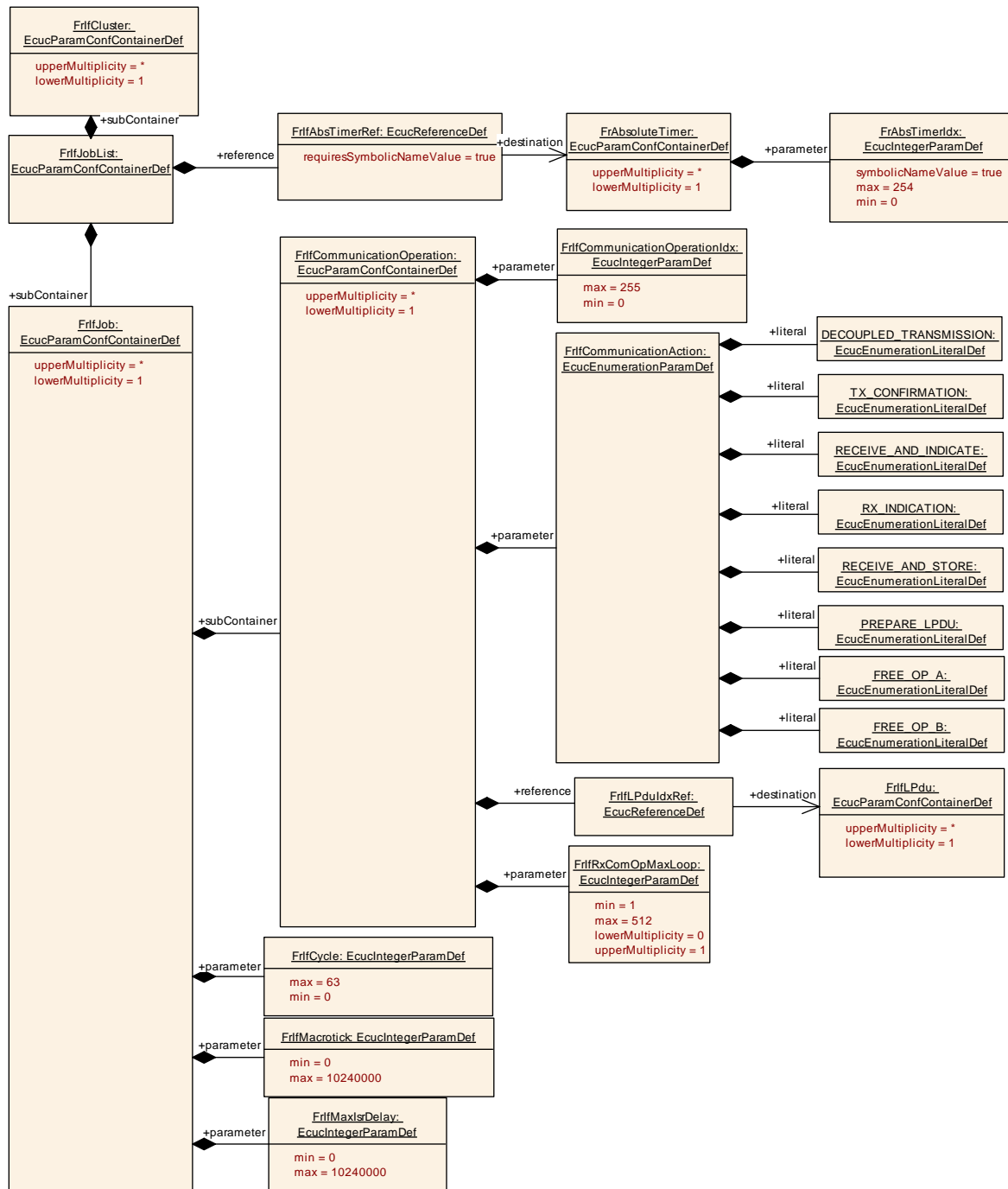


Figure 10-4: FlexRay Interface JobList

### 10.2.9 FrIfJob

<b>SWS Item</b>	<b>ECUC FrIf_05368 :</b>		
<b>Container Name</b>	FrIfJob		
<b>Description</b>	A job may contain more than one operation that are executed at a specific point in time.		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Multiplicity Configuration</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE

<b>Class</b>	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Configuration Parameters</b>			

<b>SWS Item</b>	<b>ECUC_FrIf_06064 :</b>		
<b>Name</b>	FrIfCycle		
<b>Parent Container</b>	FrIfJob		
<b>Description</b>	The FlexRay Cycle in which the communication operation will execute this job		
<b>Multiplicity</b>	1		
<b>Type</b>	EcuIntegerParamDef		
<b>Range</b>	0 .. 63		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FrIf_06065 :</b>		
<b>Name</b>	FrIfMacroTick		
<b>Parent Container</b>	FrIfJob		
<b>Description</b>	MacroTick offset in the Cycle [MacroTick]		
<b>Multiplicity</b>	1		
<b>Type</b>	EcuIntegerParamDef		
<b>Range</b>	0 .. 10240000		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FrIf_06004 :</b>		
<b>Name</b>	FrIfMaxIsrDelay		
<b>Parent Container</b>	FrIfJob		
<b>Description</b>	The maximum delay in macroticks the FrIf_JobListExec_<FrIfCluster.ShortName>() function is processed after the absolute timer interrupt was triggered.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcuIntegerParamDef		
<b>Range</b>	0 .. 10240000		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>Included Containers</b>		
<b>Container Name</b>	<b>Multiplicity</b>	<b>Scope / Dependency</b>
FrIfCommunicationOperation	1..*	A separate operation which is part of a FlexRay Job and defines what type of action is executed.

## 10.2.10 FrlfCommunicationOperation

<b>SWS Item</b>	<b>ECUC_Frlf_05369 :</b>		
<b>Container Name</b>	FrlfCommunicationOperation		
<b>Description</b>	A separate operation which is part of a FlexRay Job and defines what type of action is executed.		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Configuration Parameters</b>			

<b>SWS Item</b>	<b>ECUC_Frlf_06067 :</b>		
<b>Name</b>	FrlfCommunicationAction		
<b>Parent Container</b>	FrlfCommunicationOperation		
<b>Description</b>	The action to be performed in the FlexRay Operation		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucEnumerationParamDef		
<b>Range</b>	DECOUPLED_TRANSMISSION		Decoupled transmission
	FREE_OP_A		User defined communication operation.
	FREE_OP_B		User defined communication operation.
	PREPARE_LPDU		Prepare message buffer of CC
	RECEIVE_AND_INDICATE		Immediate reception
	RECEIVE_AND_STORE		Decoupled reception
	RX_INDICATION		Reception indication
	TX_CONFIRMATION		Transmission confirmation with optional TxConflict check
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local dependency: FrlfCommunicationAction can be configured as PREPARE_LPDU only if FrPrepareLPduSupport (ECUC_Fr_00453) is configured as TRUE.		

<b>SWS Item</b>	<b>ECUC_Frlf_06068 :</b>		
<b>Name</b>	FrlfCommunicationOperationIdx		
<b>Parent Container</b>	FrlfCommunicationOperation		
<b>Description</b>	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.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 255		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_00007 :</b>		
-----------------	--------------------------	--	--

<b>Name</b>	FrlfRxComOpMaxLoop		
<b>Parent Container</b>	FrlfCommunicationOperation		
<b>Description</b>	Defines the maximum number of loops for the receive RECEIVE_AND_INDICATE (Use case: emptying a FIFO). Please note that the parameter is mandatory if FrlfCommunicationAction parameter is set to RECEIVE_AND_INDICATE. For all other operations this parameter can be ignored.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	1 .. 512		
<b>Default value</b>	--		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06066 :</b>		
<b>Name</b>	FrlfLPduldxRef		
<b>Parent Container</b>	FrlfCommunicationOperation		
<b>Description</b>	Reference to a L-PDU index		
<b>Multiplicity</b>	1		
<b>Type</b>	Reference to [ FrlfLPdu ]		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

**No Included Containers**

### 10.2.11 FrlfFrameStructure

<b>SWS Item</b>	<b>ECUC_Frlf_05370 :</b>		
<b>Container Name</b>	FrlfFrameStructure		
<b>Description</b>	The Frame structure specifies a Construction Plan how a Frame is assembled with PDUs and their respective Update-Bits.		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Configuration Parameters</b>			

<b>SWS Item</b>	<b>ECUC_Frlf_06113 :</b>		
<b>Name</b>	FrlfByteOrder		
<b>Parent Container</b>	FrlfFrameStructure		

<b>Description</b>	This parameter defines the ByteOrder of all Pdus that are mapped into the Frame. The absolute position of a Pdu in the Frame is determined by the definition of the ByteOrder parameter: If BIG_ENDIAN is specified, the FrlfPduOffset indicates the position of the most significant bit in the Frame. If LITTLE_ENDIAN is specified, the FrlfPduOffset indicates the position of the least significant bit in the Frame.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucEnumerationParamDef		
<b>Range</b>	BIG_ENDIAN	--	
	LITTLE_ENDIAN	--	
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>Included Containers</b>		
<b>Container Name</b>	<b>Multiplicity</b>	<b>Scope / Dependency</b>
FrlfPduInFrame	1..*	This container holds all the information about a PDU in a FlexRay Frame.

### 10.2.12 FrlfPduInFrame

<b>SWS Item</b>	<b>ECUC_Frlf_05371 :</b>		
<b>Container Name</b>	FrlfPduInFrame		
<b>Description</b>	This container holds all the information about a PDU in a FlexRay Frame.		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Configuration Parameters</b>			

<b>SWS Item</b>	<b>ECUC_Frlf_06070 :</b>		
<b>Name</b>	FrlfPduOffset		
<b>Parent Container</b>	FrlfPduInFrame		
<b>Description</b>	The value specifies the offset of the PDU within the Frame [bytes].		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 253		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local 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.
--	--------------------------

<b>SWS Item</b>	<b>ECUC_Frlf_06071 :</b>		
<b>Name</b>	FrlfPduUpdateBitOffset		
<b>Parent Container</b>	FrlfPduInFrame		
<b>Description</b>	This value specifies where the PDU's Update-Bit is stored in the Frame (bit location of PDU's Update-Bit in the FlexRay Frame).		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 2031		
<b>Default value</b>	--		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local 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 be sent in static segment, this parameter depends on GPayloadLengthStatic.		

<b>SWS Item</b>	<b>ECUC_Frlf_06069 :</b>		
<b>Name</b>	FrlfPduRef		
<b>Parent Container</b>	FrlfPduInFrame		
<b>Description</b>	This is the reference to the local definition of a PDU.		
<b>Multiplicity</b>	1		
<b>Type</b>	Reference to [ FrlfPdu ]		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

**No Included Containers**

### 10.2.13 FrlfPdu

<b>SWS Item</b>	<b>ECUC_Frlf_05372 :</b>		
<b>Container Name</b>	FrlfPdu		
<b>Description</b>	Contains PDU information. A PDU may be either a transmission PDU or a reception PDU.		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Configuration Parameters</b>			

<b>Included Containers</b>		
<b>Container Name</b>	<b>Multiplicity</b>	<b>Scope / Dependency</b>
FrlfPduDirection	1	A PDU is either transmit or receive

### 10.2.14 FrlfTxPdu

<b>SWS Item</b>	<b>ECUC_Frlf_05374 :</b>		
<b>Container Name</b>	FrlfTxPdu		
<b>Description</b>	This container specifies transmission PDUs.		
<b>Configuration Parameters</b>			

<b>SWS Item</b>	<b>ECUC_Frlf_06075 :</b>		
<b>Name</b>	FrlfConfirm		
<b>Parent Container</b>	FrlfTxPdu		
<b>Description</b>	Defines whether the transmission of a PDU should be checked and confirmed to the PDU owning BSW module. If "FrlfUserTxUL" is configured as FR_TSYN then this parameter has to be set to FALSE for this PDU.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local dependency: FrlfUserTxUL		

<b>SWS Item</b>	<b>ECUC_Frlf_06076 :</b>		
<b>Name</b>	FrlfCounterLimit		
<b>Parent Container</b>	FrlfTxPdu		
<b>Description</b>	This value states the maximum number of indication of ready PDU data to the Frlf (i.e. maximum number of invocations of Frlf_Transmit) without an intermediate transmission of the PDU.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	1 .. 255		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06077 :</b>		
<b>Name</b>	FrlfImmediate		
<b>Parent Container</b>	FrlfTxPdu		
<b>Description</b>	Defines whether the PDU is transmitted immediate or decoupled.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		

<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: ECU		

<b>SWS Item</b>	<b>ECUC_FrIf_06050 :</b>		
<b>Name</b>	FrIfNoneMode		
<b>Parent Container</b>	FrIfTxPdu		
<b>Description</b>	Using the "None-Mode" which means that there is no API FrIf_Transmit call of the upper layer for this PDU.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local dependency: FrIfImmediate		

<b>SWS Item</b>	<b>ECUC_FrIf_00014 :</b>		
<b>Name</b>	FrIfTxConfirmationName		
<b>Parent Container</b>	FrIfTxPdu		
<b>Description</b>	This parameter defines the name of the <User_TxConfirmation>. This parameter depends on the parameter FrIfUserTxUL. If FrIfUserTxUL equals FR_TP, FR_AR_TP, FR_NM, PDUR or XCP, the name of the <User_TxConfirmation> is fixed. If FrIfUserTxUL equals CDD, the name of the <User_TxConfirmation> is selectable.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucFunctionNameDef		
<b>Default value</b>	--		
<b>maxLength</b>	--		
<b>minLength</b>	--		
<b>regularExpression</b>	--		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: ECU		

<b>SWS Item</b>	<b>ECUC_FrIf_06078 :</b>		
<b>Name</b>	FrIfTxPduld		
<b>Parent Container</b>	FrIfTxPdu		
<b>Description</b>	The global PDU identifier, which has to be used by the upper layer BSW module. The identifier has to be zero based and consecutive.		
<b>Multiplicity</b>	1		

<b>Type</b>	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
<b>Range</b>	0 .. 65535		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: ECU		

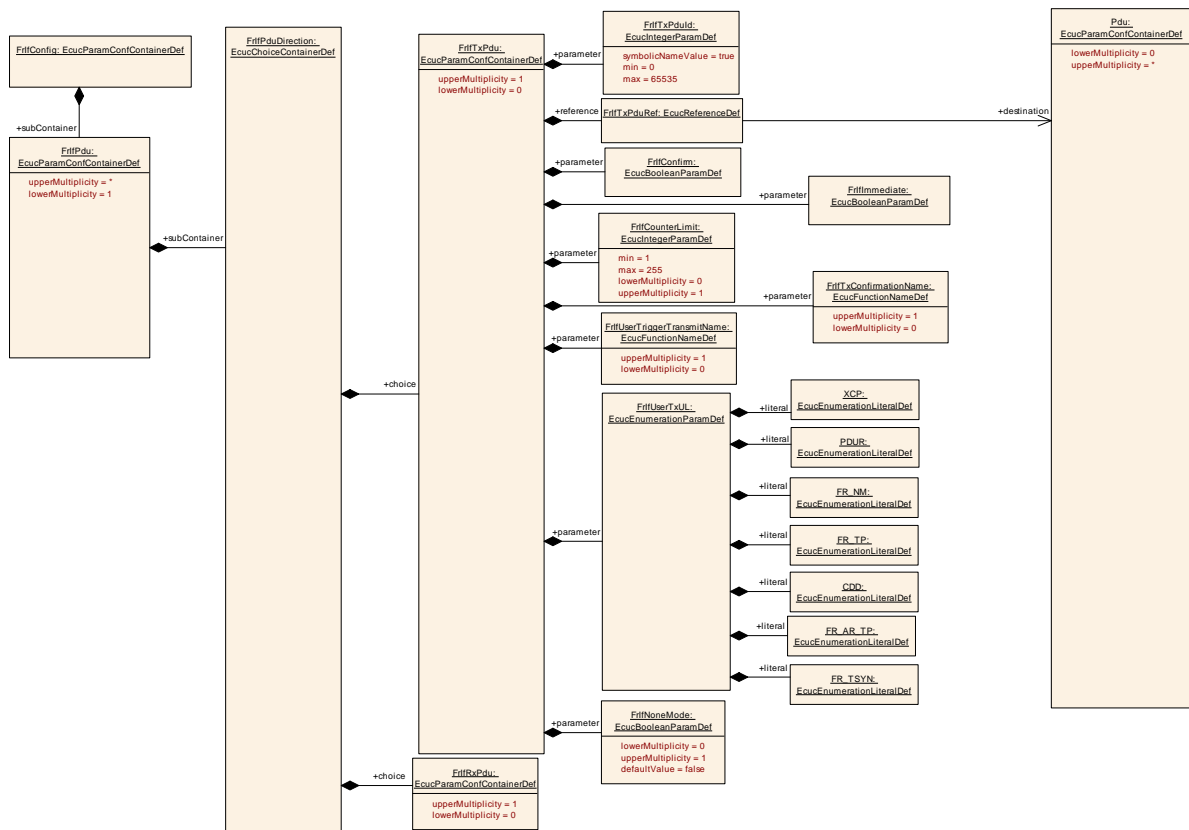
<b>SWS Item</b>	<b>ECUC_Frlf_06084 :</b>		
<b>Name</b>	FrlfUserTriggerTransmitName		
<b>Parent Container</b>	FrlfTxPdu		
<b>Description</b>	This parameter defines the name of the <User_TriggerTransmit>. This parameter depends on the parameter FrlfUserTxUL. If FrlfUserTxUL equals FR_TP, FR_AR_TP, FR_NM, PDUR, FR_TSYN or XCP the name of the <User_TriggerTransmit> is fixed. If FrlfUserTxUL equals CDD, the name of the <User_TriggerTransmit> is selectable.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucFunctionNameDef		
<b>Default value</b>	--		
<b>maxLength</b>	--		
<b>minLength</b>	--		
<b>regularExpression</b>	--		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: ECU dependency: FrlfImmediate		

<b>SWS Item</b>	<b>ECUC_Frlf_00015 :</b>		
<b>Name</b>	FrlfUserTxUL		
<b>Parent Container</b>	FrlfTxPdu		
<b>Description</b>	This parameter defines the upper layer (UL) module to which the trigger of the Pdu to be transmitted (via the <User_TriggerTransmit>) or the confirmation of the successfully transmitted Pdu has to be routed (via the <User_TxConfirmation>). Please note that handle IDs which are used in callback functions are defined by the upper layer module.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucEnumerationParamDef		
<b>Range</b>	CDD	Complex Driver	
	FR_AR_TP	FR AUTOSAR TP	
	FR_NM	FR NM	
	FR_TP	FR ISO TP	
	FR_TSYN	Global Time Synchronization over FlexRay	
	PDUR	PDU Router	
	XCP	Extended Calibration Protocol	
<b>Post-Build Variant Value</b>	true		
<b>Value</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE

<b>Configuration Class</b>	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: ECU dependency: FrIfConfirm		

<b>SWS Item</b>	<b>ECUC_FrIf_06074 :</b>		
<b>Name</b>	FrIfTxPduRef		
<b>Parent Container</b>	FrIfTxPdu		
<b>Description</b>	Reference to the external PDU definition.		
<b>Multiplicity</b>	1		
<b>Type</b>	Reference to [ Pdu ]		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: ECU		

**No Included Containers**



### 10.2.15 FrIfRxPdu

<b>SWS Item</b>	<b>ECUC_FrIf_05373 :</b>		
<b>Container Name</b>	FrIfRxPdu		
<b>Description</b>	Receive PDU		
<b>Configuration Parameters</b>			

<b>SWS Item</b>	<b>ECUC_Frlf_00016 :</b>		
<b>Name</b>	FrlfRxIndicationName		
<b>Parent Container</b>	FrlfRxPdu		
<b>Description</b>	This parameter defines the name of the <User_RxIndication>. This parameter depends on the parameter FrlfUserRxIndicationUL. If FrlfUserRxIndicationUL equals FR_TP, FR_AR_TP, FR_NM, PDUR, FR_TSYN or XCP, the name of the <User_RxIndication> is fixed. If FrlfUserRxIndicationUL equals CDD, the name of the <User_RxIndication> is selectable.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucFunctionNameDef		
<b>Default value</b>	--		
<b>maxLength</b>	--		
<b>minLength</b>	--		
<b>regularExpression</b>	--		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: ECU		

<b>SWS Item</b>	<b>ECUC_Frlf_00017 :</b>		
<b>Name</b>	FrlfUserRxIndicationUL		
<b>Parent Container</b>	FrlfRxPdu		
<b>Description</b>	This parameter defines the upper layer (UL) module to which the indication of the successfully received FrlfRxPdu has to be routed via <User_RxIndication>. This <User_RxIndication> has to be invoked when the indication of the configured FrlfRxPdu will be received by a Rx indication event from the FR Driver module. If no upper layer (UL) module is configured, no <User_RxIndication> has to be called in case of a Rx indication event of the FrlfRxPdu from the FR Driver module.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucEnumerationParamDef		
<b>Range</b>	CDD	Complex Driver	
	FR_AR_TP	FR AR TP	
	FR_NM	FR NM	
	FR_TP	FR ISO TP	
	FR_TSYN	Global Time Synchronization over FlexRay	
	PDUR	PDU Router	
	XCP	Extended Calibration Protocol	
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: ECU		

<b>SWS Item</b>	<b>ECUC_Frlf_06073 :</b>		
<b>Name</b>	FrlfRxPduRef		
<b>Parent Container</b>	FrlfRxPdu		

<b>Description</b>	Reference to the external PDU definition.		
<b>Multiplicity</b>	1		
<b>Type</b>	Reference to [ Pdu ]		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: ECU		

**No Included Containers**

### 10.2.16 FrlfPduDirection

<b>SWS Item</b>	<b>ECUC_Frlf_06072 :</b>
<b>Choice container Name</b>	FrlfPduDirection
<b>Description</b>	A PDU is either transmit or receive

<b>Container Choices</b>		
<b>Container Name</b>	<b>Multiplicity</b>	<b>Scope / Dependency</b>
FrlfRxPdu	0..1	Receive PDU
FrlfTxPdu	0..1	This container specifies transmission PDUs.

### 10.2.17 FrlfConfig

<b>SWS Item</b>	<b>ECUC_Frlf_06001 :</b>
<b>Container Name</b>	FrlfConfig
<b>Description</b>	This container contains the configuration parameters and sub containers of the AUTOSAR Frlf module.
<b>Configuration Parameters</b>	

<b>SWS Item</b>	<b>ECUC_Frlf_06121 :</b>		
<b>Name</b>	FrlfMaxPduCnt		
<b>Parent Container</b>	FrlfConfig		
<b>Description</b>	Maximum number of Pdus. This parameter is needed only in case of post-build loadable implementation using static memory allocation.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 ..		
	18446744073709551615		
<b>Default value</b>	--		
<b>Post-Build Variant Multiplicity</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD

	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>Included Containers</b>		
<b>Container Name</b>	<b>Multiplicity</b>	<b>Scope / Dependency</b>
FrlfCluster	1..*	This container specifies a Frlf Cluster and all related data which is required to enable communication of the Cluster. A Cluster may consist of more than one Controller.
FrlfFrameStructure	1..*	The Frame structure specifies a Construction Plan how a Frame is assembled with PDUs and their respective Update-Bits.
FrlfPdu	1..*	Contains PDU information. A PDU may be either a transmission PDU or a reception PDU.

### 10.2.18 FrlfClusterDemEventParameterRefs

<b>SWS Item</b>	<b>ECUC_Frlf_06091 :</b>
<b>Container Name</b>	FrlfClusterDemEventParameterRefs
<b>Description</b>	Container for the references to DemEventParameter elements which shall be invoked using the API Dem_SetEventStatus in case the corresponding error occurs. The EventId is taken from the referenced DemEventParameter's DemEventId symbolic value. The standardized errors are provided in this container and can be extended by vendor-specific error references.
<b>Configuration Parameters</b>	

<b>SWS Item</b>	<b>ECUC_Frlf_06097 :</b>		
<b>Name</b>	FRIF_E_ACS_CH_A		
<b>Parent Container</b>	FrlfClusterDemEventParameterRefs		
<b>Description</b>	Reference to the DemEventParameter which shall be issued when an error in ACS on channel A was detected. If the reference is not configured the error shall not be reported.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	Symbolic name reference to [ DemEventParameter ]		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06098 :</b>		
<b>Name</b>	FRIF_E_ACS_CH_B		



<b>Parent Container</b>	FrlfClusterDemEventParameterRefs		
<b>Description</b>	Reference to the DemEventParameter which shall be issued when an error in ACS on channel B was detected. If the reference is not configured the error shall not be reported.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	Symbolic name reference to [ DemEventParameter ]		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06093 :</b>		
<b>Name</b>	FRIF_E_NIT_CH_A		
<b>Parent Container</b>	FrlfClusterDemEventParameterRefs		
<b>Description</b>	Reference to the DemEventParameter which shall be issued when an error in NIT on channel A was detected. If the reference is not configured the error shall not be reported.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	Symbolic name reference to [ DemEventParameter ]		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06094 :</b>		
<b>Name</b>	FRIF_E_NIT_CH_B		
<b>Parent Container</b>	FrlfClusterDemEventParameterRefs		
<b>Description</b>	Reference to the DemEventParameter which shall be issued when an error in NIT on channel B was detected. If the reference is not configured the error shall not be reported.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	Symbolic name reference to [ DemEventParameter ]		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06095 :</b>		
-----------------	--------------------------	--	--

<b>Name</b>	FRIF_E_SW_CH_A		
<b>Parent Container</b>	FrlfClusterDemEventParameterRefs		
<b>Description</b>	Reference to the DemEventParameter which shall be issued when an error in SW on channel A was detected. If the reference is not configured the error shall not be reported.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	Symbolic name reference to [ DemEventParameter ]		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_Frlf_06096 :</b>		
<b>Name</b>	FRIF_E_SW_CH_B		
<b>Parent Container</b>	FrlfClusterDemEventParameterRefs		
<b>Description</b>	Reference to the DemEventParameter which shall be issued when an error in SW on channel B was detected. If the reference is not configured the error shall not be reported.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	Symbolic name reference to [ DemEventParameter ]		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

**No Included Containers**

### 10.2.19 FrlfFrameTriggeringDemEventParameterRefs

<b>SWS Item</b>	<b>ECUC_Frlf_06099 :</b>
<b>Container Name</b>	FrlfFrameTriggeringDemEventParameterRefs
<b>Description</b>	Container for the references to DemEventParameter elements which shall be invoked using the API Dem_SetEventStatus in case the corresponding error occurs. The EventId is taken from the referenced DemEventParameter's DemEventId symbolic value. The standardized errors are provided in this container and can be extended by vendor-specific error references.
<b>Configuration Parameters</b>	

<b>SWS Item</b>	<b>ECUC_Frlf_00009 :</b>
<b>Name</b>	FRIF_E_LPDU_SLOTSTATUS

<b>Parent Container</b>	FrIfFrameTriggeringDemEventParameterRefs		
<b>Description</b>	Reference to DEM event Id that is reported when FlexRay driver module detects slot errors. If this parameter is not configured, no event reporting happens.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	Symbolic name reference to [ DemEventParameter ]		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

**No Included Containers**

### 10.3 Published Information

For details refer to the chapter 10.3 “Published Information” in [SWS\\_BSWGeneral](#).

## 11 Not applicable requirements

**[SWS\_FrIf\_06118]** [These requirements are not applicable to this specification.

(SRS\_BSW\_00159, SRS\_BSW\_00167, SRS\_BSW\_00387, SRS\_BSW\_00416, SRS\_BSW\_00168, SRS\_BSW\_00423, SRS\_BSW\_00424, SRS\_BSW\_00425, SRS\_BSW\_00426, SRS\_BSW\_00427, SRS\_BSW\_00428, SRS\_BSW\_00429, BSW00431, SRS\_BSW\_00432, BSW00434, SRS\_BSW\_00417, SRS\_BSW\_00386, SRS\_BSW\_00161, SRS\_BSW\_00162, SRS\_BSW\_00005, SRS\_BSW\_00415, SRS\_BSW\_00164, SRS\_BSW\_00325, SRS\_BSW\_00326, SRS\_BSW\_00413, SRS\_BSW\_00347, SRS\_BSW\_00373, SRS\_BSW\_00335, SRS\_BSW\_00410, SRS\_BSW\_00314, SRS\_BSW\_00370, SRS\_BSW\_00328, SRS\_BSW\_00312, SRS\_BSW\_00006, SRS\_BSW\_00377, SRS\_BSW\_00306, SRS\_BSW\_00371, SRS\_BSW\_00376, SRS\_BSW\_00329, SRS\_BSW\_00330, , SRS\_BSW\_00331, SRS\_BSW\_00009, SRS\_BSW\_00172, SRS\_BSW\_00010, SRS\_BSW\_00333, SRS\_BSW\_00341, BSW05078, BSW05101, BSW05163, BSW05164, BSW05165, BSW05067, BSW05068, BSW05069, BSW05153, BSW05035, BSW05038, BSW05162, BSW05113, BSW05102, SRS\_Fr\_05009) ]