

Document Title	Specification of FlexRay Interface
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	027

Document Status	Final
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	4.3.1

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

Document Change History			
Date	Release	Changed by	Change Description
2010-09-30	3.1.5	AUTOSAR Administration	<ul style="list-style-type: none"> • API "Frlf_GetCycleLength" added • API "Frlf_ReadCCConfig" added • APIs Frlf_EnableTransceiverWakeup / Frlf_DisableTransceiverWakeup removed • Configuration parameter "FrlfByteOrder" added
2010-02-02	3.1.4	AUTOSAR Administration	<ul style="list-style-type: none"> • Added support for FlexRay 3.0 hardware (CCs and transceivers) • Added functionalities to get detailed (error) information of the communications bus • Added support for single/key-slot mode • Added "cancel transmission" support • Legal disclaimer revised
2008-08-13	3.1.1	AUTOSAR Administration	<ul style="list-style-type: none"> • Legal disclaimer revised
2008-02-01	3.0.2	AUTOSAR Administration	<ul style="list-style-type: none"> • Correction of Figure 5.1
2007-12-21	3.0.1	AUTOSAR Administration	<ul style="list-style-type: none"> • Simplification of the FlexRay Interface State Machine due to the introduction of the new AUTOSAR SWS FlexRay State Manager • Cluster-based APIs were removed due to the introduced AUTOSAR SWS FlexRay State Manager • 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 • Small layout adaptations made

Document Change History			
Date	Release	Changed by	Change Description
2007-01-24	2.1.15	AUTOSAR Administration	<ul style="list-style-type: none">• “Advice for users” revised• Legal disclaimer added• “Revision Information” added• Release Notes added
2006-05-16	2.0	AUTOSAR Administration	<ul style="list-style-type: none">• Second Release
2005-05-31	1.0	AUTOSAR Administration	<ul style="list-style-type: none">• Initial Release

Disclaimer

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

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

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

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

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

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 FlexRay Driver	17
5.7	AUTOSAR FlexRay Transceiver Driver.....	17
5.8	File Structure.....	18
5.8.1	Header File Structure	18
6	Requirements Traceability	21
6.1	Specification Items	23
7	Functional Specification.....	24
7.1	FlexRay BSW Stack.....	24
7.2	Indexing Scheme.....	24
7.2.1	Principle	24
7.2.2	Supported Indexed Resources.....	28
7.3	FlexRay Interface State Machine	28
7.3.1	FlexRay Interface Main Function.....	30
7.4	Implementation Requirements	32
7.5	Configuration description.....	33
7.6	Data Communication via FlexRay	33
7.6.1	PDU Packing, PDU update bits, and Frame Construction Plans.....	34
7.6.2	Dynamic PDU length	36
7.6.3	AlwaysTransmit.....	36
7.6.4	Realization of the Time-Driven FlexRay Schedule	37
7.6.5	Communication Operations.....	40
7.6.6	Transmission with Immediate Buffer Access.....	46
7.7	Error Classification	47
7.7.1	Development Errors	47
7.7.2	Runtime Errors	47

7.7.3	Transient Faults	47
7.7.4	Production Errors	48
7.7.5	Extended Production Errors	51
7.8	Error Detection	51
8	API Service Specification	53
8.1	Imported types.....	53
8.2	Type Definitions.....	53
8.2.1	Frlf_ConfigType	53
8.2.2	Frlf_StateType	54
8.2.3	Frlf_StateTransitionType.....	54
8.3	Function Definitions.....	55
8.3.1	Frlf_Init.....	55
8.3.2	Frlf_ControllerInit	56
8.3.3	Frlf_SetAbsoluteTimer	56
8.3.4	Frlf_EnableAbsoluteTimerIRQ	57
8.3.5	Frlf_AckAbsoluteTimerIRQ	58
8.3.6	Frlf_StartCommunication	59
8.3.7	Frlf_HaltCommunication	60
8.3.8	Frlf_AbortCommunication	61
8.3.9	Frlf_GetState.....	62
8.3.10	Frlf_SetState	63
8.3.11	Frlf_SetWakeupChannel	64
8.3.12	Frlf_SendWUP	65
8.3.13	Frlf_GetPOCStatus	66
8.3.14	Frlf_GetGlobalTime.....	67
8.3.15	Frlf_AllowColdstart.....	68
8.3.16	Frlf_GetMacroticksPerCycle	68
8.3.17	Frlf_GetMacrotickDuration	69
8.3.18	Frlf_Transmit.....	70
8.3.19	Frlf_SetTransceiverMode.....	71
8.3.20	Frlf_GetTransceiverMode	72
8.3.21	Frlf_GetTransceiverWUReason	73
8.3.22	Frlf_ClearTransceiverWakeup	75
8.3.23	Frlf_CancelAbsoluteTimer.....	76
8.3.24	Frlf_GetAbsoluteTimerIRQStatus	77
8.3.25	Frlf_DisableAbsoluteTimerIRQ	78
8.3.26	Frlf_GetCycleLength	78
8.4	Optional Function Definitions	79
8.4.1	Frlf_AllSlots.....	79
8.4.2	Frlf_GetChannelStatus	80
8.4.3	Frlf_GetClockCorrection	81
8.4.4	Frlf_GetSyncFrameList.....	82
8.4.5	Frlf_GetNumOfStartupFrames	83
8.4.6	Frlf_GetWakeupRxStatus	84
8.4.7	Frlf_CancelTransmit.....	85
8.4.8	Frlf_DisableLPdu	86
8.4.9	Frlf_GetTransceiverError	87
8.4.10	Frlf_EnableTransceiverBranch.....	88
8.4.11	Frlf_DisableTransceiverBranch.....	89

8.4.12	Frlf_ReconfigLPdu	91
8.4.13	Frlf_GetNmVector	92
8.4.14	Frlf_GetVersionInfo	93
8.4.15	Frlf_ReadCCConfig	93
8.5	Interrupt Service Routines	95
8.5.1	Frlf_JobListExec_<ClstIdx>	95
8.6	Call-back Notifications	96
8.6.1	Frlf_CheckWakeupByTransceiver	96
8.7	Scheduled Functions	97
8.7.1	Frlf_MainFunction_<ClstIdx>	97
8.8	Expected Interfaces	98
8.8.1	Mandatory Interfaces	98
8.8.2	Optional Interfaces	99
8.8.3	Configurable Interfaces	101
9	Sequence Diagrams	106
9.1	Data Transmission	106
9.1.1	TransmitWithImmediateBufferAccess	106
9.1.2	TransmitWithDecoupledBufferAccess	107
9.1.3	ProvideTxConfirmation	108
9.2	Data Reception	109
9.2.1	ReceiveAndIndicate	109
9.2.2	ReceiveAndStore	110
9.2.3	ProvideRxIndication	111
9.2.4	Cancel Transmission	112
9.3	Prepare LPDU	113
10	Configuration Specification	114
10.1	How to Read this Chapter	114
10.2	Containers and Configuration Parameters	114
10.2.1	Frlf	115
10.2.2	FrlfGeneral	116
10.2.3	FrlfCluster	123
10.2.4	FrlfController	134
10.2.5	FrlfTransceiver	136
10.2.6	FrlfLPdu	137
10.2.7	FrlfFrameTriggering	138
10.2.8	FrlfJobList	142
10.2.9	FrlfJob	143
10.2.10	FrlfCommunicationOperation	145
10.2.11	FrlfFrameStructure	146
10.2.12	FrlfPdusInFrame	147
10.2.13	FrlfPdu	148
10.2.14	FrlfTxPdu	149
10.2.15	FrlfRxPdu	152
10.2.16	FrlfPduDirection	154
10.2.17	FrlfConfig	154
10.2.18	FrlfClusterDemEventParameterRefs	155
10.2.19	FrlfFrameTriggeringDemEventParameterRefs	157
10.3	Published Information	158

11 Not applicable requirements 159

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 CC
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 executable code itself (i.e. the sequence of instructions executed during runtime), but the data used to configure <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 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.

5.7 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).

5.8 File Structure

5.8.1 Header File Structure

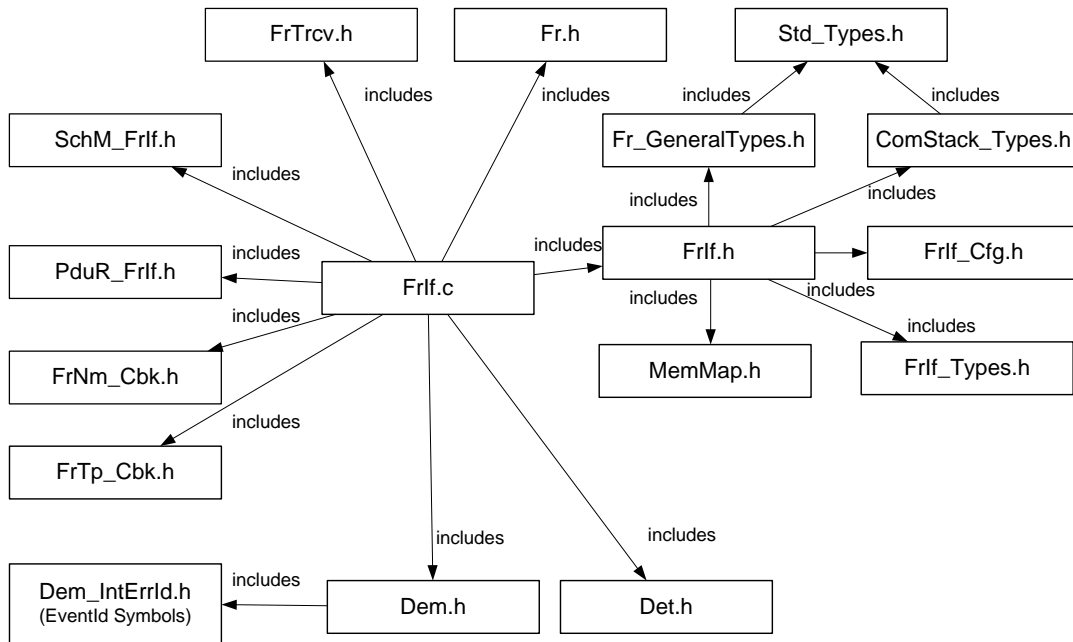


Figure 5-1: FlexRay Interface Header File Structure

The header file structure shall contain the following header files:

[SWS_FrIf_05744] [Fr.h contains the declarations of the API services of the FlexRay Driver used by the FlexRay Interface] ()

[SWS_FrIf_05745] [FrTrcv.h contains the declarations of the API services of the FlexRay Transceiver Driver used by the FlexRay Interface.] ()

[SWS_FrIf_05746] [Fr_GeneralTypes.h contains declarations shared by all AUTOSAR FlexRay [BSW](#) modules] ()

[SWS_FrIf_05747] [ComStack_Types.h contains the communication module abstracted datatypes shared by AUTOSAR communication BSW.] ()

[SWS_FrIf_05748] [PduR_FrIf.h contains the declarations of API services the PDU router offers to the FlexRay Interface] ()

[SWS_FrIf_05749] [FrNm_Cbk.h contains the declarations of API services the FrNm offers to the FlexRay Interface] ()

[SWS_FrIf_05750] [FrTp_Cbk.h contains the declarations of API services the FrTp offers to the FlexRay Interface] ()

[SWS_FrIf_05751] [Det.h contains the declarations of the API services of the Det used by the FlexRay Interface] ()

[SWS_FrIf_05752] [SchM_FrIf.h contains the declaration of the API services the SchM offers to the FlexRay Interface] ()

[SWS_FrIf_05753] [FrIf_Types.h contains the declaration of FrIf specific types.] ()

Note:

By this inclusion the APIs to report errors as well as the required Event Id symbols are included.

Note:

This specification defines the name of the Event Id symbols, which are provided by XML to the DEM configuration tool.

[SWS_FrIf_15140] [The implementation of the FrIf module shall provide the header file *FrIf.h*, which is the main module interface file.] ()

[SWS_FrIf_25140] [It shall contain all types and function prototypes required by the FrIf module's environment.] ()

[SWS_FrIf_05087] [The FrIf module source code file(s) shall include *SchM_FrIf.h* if data consistency mechanisms of the BSW scheduler are required as described in [13].] ()

[SWS_Frlf_05090] [The header file *Frlf.h* shall contain a software and specification version number.] (SRS_BSW_00004)

[SWS_Frlf_05091] [*Frlf.h* shall include *Fr_GeneralTypes.h* for the include of general FlexRay type declaration] (SRS_BSW_00456)

[SWS_Frlf_05097] [The types specified in SWS_Frlf_05091 shall be declared in *Fr_GeneralTypes.h*] (SRS_BSW_00456)

[SWS_Frlf_05098] [The header file *FrTSyn_Cbk.h* shall contain the declarations of API services the FrTSyn offers to the FlexRay Interface] (SRS_BSW_00456)

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
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_00406	A static status variable denoting if a BSW module is initialized shall be initialized with value 0 before any APIs of the BSW module is called	SWS_Frlf_05298
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	SWS_Frlf_05002

	the API	
SRS_BSW_00414	Init functions shall have a pointer to a configuration structure as single parameter	SWS_Frlf_05003
SRS_BSW_00456	- A Header file shall be defined in order to harmonize BSW Modules	SWS_Frlf_05091, SWS_Frlf_05097, SWS_Frlf_05098
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 Driver(s)	SWS_Frlf_05053
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	Frlf05368
Communication Job execution start time	SWS_Frlf_05133
Actions specified by Communication Operation	SWS_Frlf_05134
Communication Operation properties	Frlf05369
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.

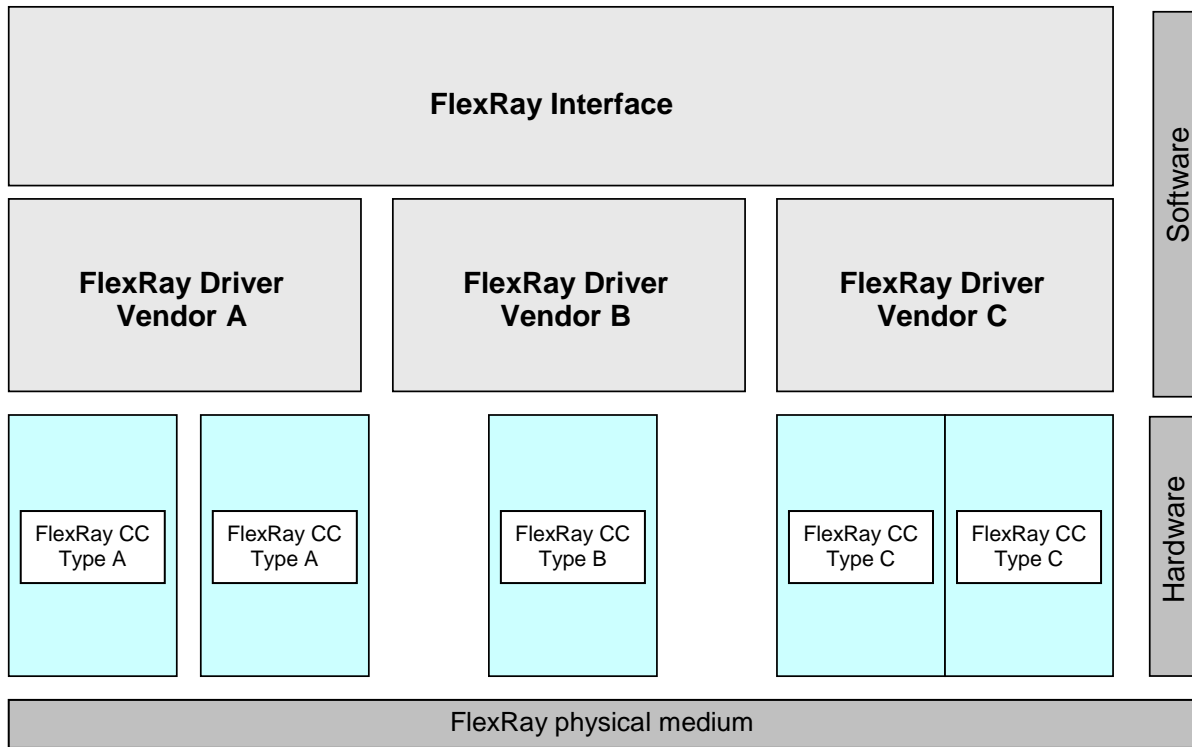


Figure 7-1: Basic Structure of the FlexRay BSW Stack

7.2 Indexing Scheme

7.2.1 Principle

Most of the [Frlf](#) module's API services used for accessing the numerous (hardware and software) resources¹ 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 entities² 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.

¹ E.g. timers, configuration data sets, etc.

² FlexRay Drivers, FlexRay Communication Controllers, FlexRay Transceiver Drivers, and FlexRay Transceivers

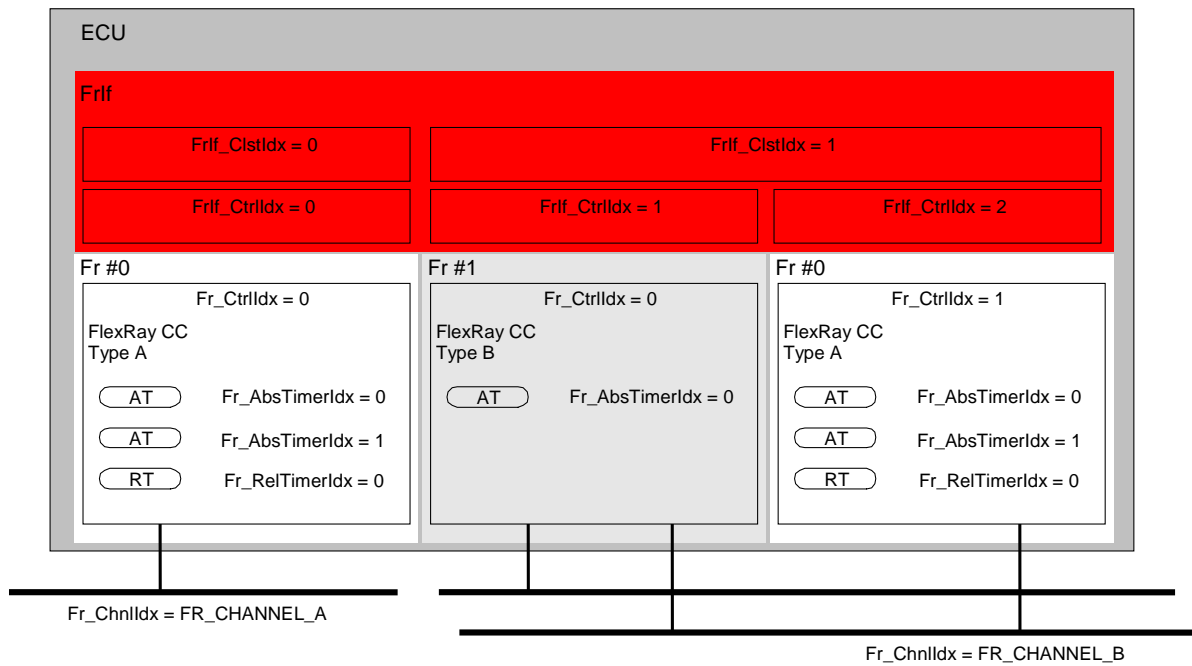


Figure 7-2: CC Indexing Scheme of the FlexRay Interface

[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)

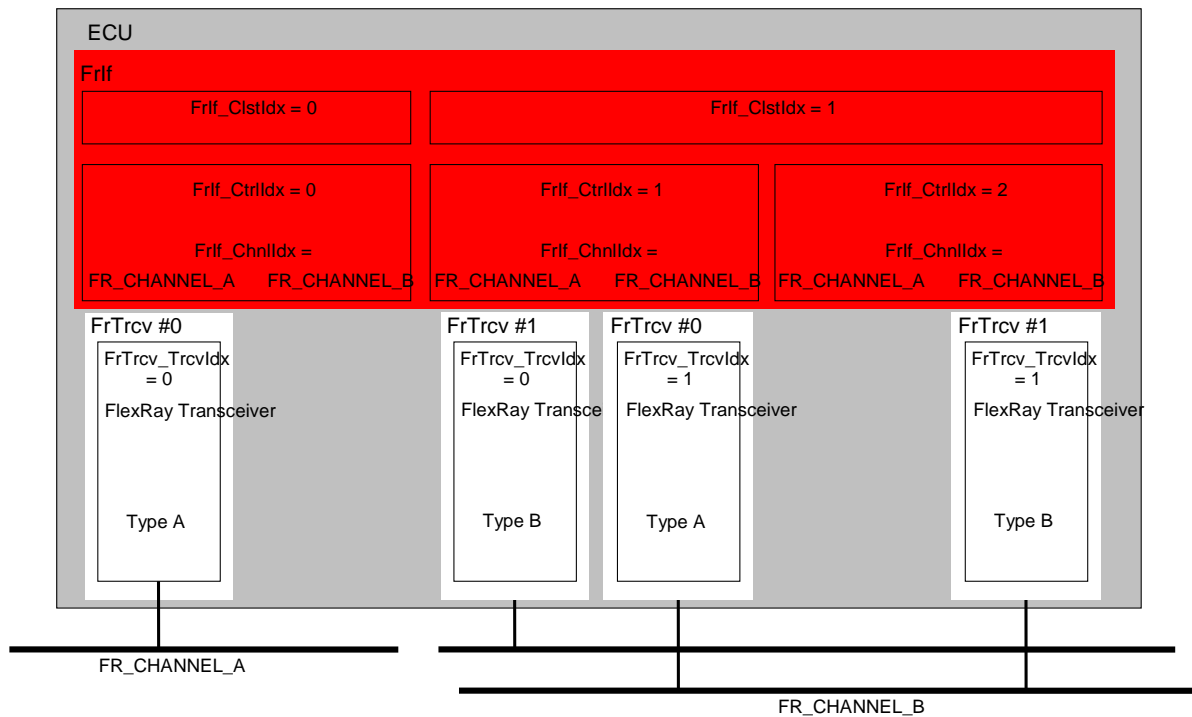


Figure 7-3: Flexray Transceiver Indexing Scheme of the FlexRay Interface

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_TrcvIdx**. (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

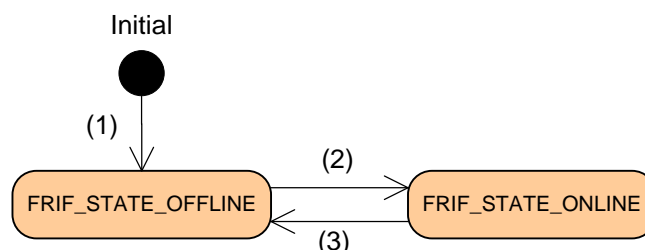


Figure 7-4: FlexRay Interface State Machine

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 `_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_0()` for Cluster # 0 (`Frlf_ClstIdx = 0`)
- `Frlf_MainFunction_1()` for Cluster # 1 (`Frlf_ClstIdx = 1`)
- `Frlf_MainFunction_2()` for Cluster # 2 (`Frlf_ClstIdx = 2`)
- `Frlf_MainFunction_3()` for Cluster # 3 (`Frlf_ClstIdx = 3`)
- ... and so on, if more than 4 FlexRay Clusters are supported.

¶ ()

[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_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 `Frlf_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 time³ 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](#)]

³ In FlexRay Global Time
37 of 159

- Macrotick Offset within the Communication Cycle [Configuration Parameter [FrlfMacrotick](#), see [Frlf06065](#)].
- 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](#)]⁴.] ()

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

⁴ 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_Frlf_05136] [The API names of the FlexRay Job List Execution Functions shall obey the following pattern:

- Frlf_JobListExec_0() for Cluster # 0 (Frlf_ClstIdx = 0)
- Frlf_JobListExec_1() for Cluster # 1 (Frlf_ClstIdx = 1)
- Frlf_JobListExec_2() for Cluster # 2 (Frlf_ClstIdx = 2)
- Frlf_JobListExec_3() for Cluster # 3 (Frlf_ClstIdx = 3)
- ... and so on, if more than 4 FlexRay Clusters are supported.] ()

[SWS_Frlf_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_Frlf_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 FrlfMaxIsrDelay, see Frlf06004], 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_Frlf_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 _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 _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 _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 `FrIfNoneMode == true` and `_TriggerTransmit` returned `E_OK` or the frame is configured to be always transmitted [Configuration Parameter `FrIfAlwaysTransmit == 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_LPduldx` is set to the configured L-PDU index [Configuration Parameter `FrIfLPduldx`, see `FrIf06058`] associated with the Communication Operation
 - c. `Fr_LSduPtr` is set to the temporary `FrIf` L-SDU assembling buffer.
 - d. `Fr_LSduLength` is set to the L-SDU length [Configuration Parameter `FrIfLSduLength`, see `FrIf06054`]
3. 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 FrIf_05287](#) are depicted in detail in the sequence chart in chapter 9.1.2.

In case the parameter '`FrIfAllowDynamicLSduLength`' 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 `_TriggerTransmit()`:

- the position of the respective PDU within the L-SDU
- the actual length of the respective PDU as passed via `_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_FrIf_05064] [If the related CC is in `FrIf_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_FrIf_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.
 2. If the transmission was performed (output parameter *Fr_TxLPduStatusPtr is set to FR_TRANSMITTED or 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_OK)> with the associated PDUId (defined by the upper layer).
 - b. If FrlfConfirm == true, decrement [TxConfCounter](#).
 3. 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.
2. 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.
3. 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 _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_Frlf_05292] ¶ If the related CC is in Frlf_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_Frlf_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_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.
- 2) Initialize ComOpLoopCounter to 0.
- 3) As long as ComOpLoopCounter < FrlfRxComOpMaxLoop 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) For each Pdu contained in the FrlfFrameStructure (see [Frlf05370](#)) of the associated frame triggering do
 -) 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
 -) Call the upper layer's function _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 FrlfPduOffset, see [Frlf06070](#)] as parameters.

d) 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 `TxPduld` 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_LPduldx` is set to the configured L-PDU index [Configuration Parameter `FrlfLPduldx`, see [Frlf06058](#)] associated with the `TxPduld`.
- 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()`.

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](#)].

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

| 0

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 DEM
error detection in NIT on channel B	FRIF_E_NIT_CH_B	Assigned by DEM
error detection in SW on channel A	FRIF_E_SW_CH_A	Assigned by DEM
error detection in SW on channel B	FRIF_E_SW_CH_B	Assigned by DEM
error detection in ACS on channel A	FRIF_E_ACS_CH_A	Assigned by DEM
error detection in ACS on channel B	FRIF_E_ACS_CH_B	Assigned by DEM

| ()

Table 7-3: Definition of Production Errors

[SWS_Frlf_05426] |

Error Name:	FRIF_E_NIT_CH_A	
Short Description:	Error detection in NIT on channel A	
Long Description:	This production error shall be issued when an error in NIT on channel A was detected	
Recommended DTC:	N/A	
Detection Criteria:	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 (SWS_Frlf_35120)
	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 (SWS_Frlf_35120)
Secondary Parameters:	N/A	
Time Required:	N/A	
Monitor Frequency	continuous	
MIL illumination:	N/A	

| ()

[SWS_Frlf_05427] |

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

	channel B was detected	
Recommended DTC:	N/A	
Detection Criteria:	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 (SWS FrIf_45120)
	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 (SWS FrIf_45120)
Secondary Parameters:	N/A	
Time Required:	N/A	
Monitor Frequency	continuous	
MIL illumination:	N/A	

] ()

[SWS_FrIf_05428]

Error Name:	FRIF_E_SW_CH_A	
Short Description:	Error detection in SW on channel A	
Long Description:	This production error shall be issued when an error in SW on channel A was detected.	
Recommended DTC:	N/A	
Detection Criteria:	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 (SWS FrIf_55120)
	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 (SWS FrIf_55120)
Secondary Parameters:	N/A	
Time Required:	N/A	
Monitor Frequency	continuous	
MIL illumination:	N/A	

] ()

[SWS_Frlf_05429]

Error Name:	FRIF_E_SW_CH_B	
Short Description:	Error detection in SW on channel B	
Long Description:	This production error shall be issued when an error in SW on channel B was detected.	
Recommended DTC:	N/A	
Detection Criteria:	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 (SWS Frlf 65120)
	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 (SWS Frlf 65120)
Secondary Parameters:	N/A	
Time Required:	N/A	
Monitor Frequency	continuous	
MIL illumination:	N/A	

] ()

[SWS_Frlf_05431]

Error Name:	FRIF_E_ACS_CH_A	
Short Description:	Error detection in ACS on channel A	
Long Description:	This production error shall be issued when an error in ACS on channel A was detected	
Recommended DTC:	N/A	
Detection Criteria:	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 (SWS Frlf 75120)
	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 (SWS FrIf_75120)
Secondary Parameters:	N/A	
Time Required:	N/A	
Monitor Frequency	continuous	
MIL illumination:	N/A	

] ()

[SWS_FrIf_05430]

Error Name:	FRIF_E_ACS_CH_B	
Short Description:	Error detection in ACS on channel B	
Long Description:	This production error shall be issued when an error in ACS on channel B was detected	
Recommended DTC:	N/A	
Detection Criteria:	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 (SWS FrIf_85120)
	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 (SWS FrIf_85120)
Secondary Parameters:	N/A	
Time Required:	N/A	
Monitor Frequency	continuous	
MIL illumination:	N/A	

] ()

7.7.5 Extended Production Errors

There are no extended production errors.

7.8 Error Detection

[SWS_Frlf_05298] [If the FrlfDevErrorDetect switch is set to ON, all [Frlf](#) module API services other than Frlf_Init() and Frlf_GetVersionInfo() shall:

- not execute their normal operation,
- report to the DET module (using FRIF_E_UNINIT),
- and return E_NOT_OK,

unless the [Frlf](#) module has been initialized with a preceding call of Frlf_Init().]

(SRS_BSW_00406)

8 API Service Specification

8.1 Imported types

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

[SWS_Frlf_05001] [

<i>Module</i>	<i>Imported Type</i>
ComStack_Types	PduldType
	PdulInfoType
Dem	Dem_EventIdType
	Dem_EventStatusType
Fr	Fr_ChannelType
	Fr_POCTestStatusType
	Fr_RxLPduStatusType
	Fr_TxLPduStatusType
FrTrcv	FrTrcv_TrvcModeType
	FrTrcv_TrvcWUReasonType
Std_Types	Std_ReturnType
	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] [

Name:	FrIf_ConfigType	
Type:	Structure	
Range:	Implementation specific	--
Description:	This type contains the implementation-specific post build time configuration structure. Only pointers of this type are allowed.	

] ()

8.2.2 FrIf_StateType

[SWS_FrIf_05755] [

Name:	FrIf_StateType	
Type:	Enumeration	
Range:	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.
Description:	Variables of this type are used to represent the FrIf_State of a FlexRay CC.	

] ()

8.2.3 FrIf_StateTransitionType

[SWS_FrIf_05303] [

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

] ()

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] [

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

] (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] [

Service name:	FrIf_ControllerInit	
Syntax:	Std_ReturnType FrIf_ControllerInit(uint8 FrIf_CtrlIdx)	
Service ID[hex]:	0x03	
Sync/Async:	Synchronous	
Reentrancy:	non reentrant for identical values of FrIf_CtrlIdx, reentrant for different values of FrIf_CtrlIdx	
Parameters (in):	FrIf_CtrlIdx	Index of the FlexRay CC to address.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	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.
Description:	Initialized a FlexRay CC.	

] (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] [

Service name:	FrIf_SetAbsoluteTimer	
Syntax:	Std_ReturnType FrIf_SetAbsoluteTimer(uint8 FrIf_CtrlIdx uint32 Time uint8 FrIf_CtrlIdx uint32 Time)	

	uint8 FrIf_CtrlIdx, uint8 FrIf_AbsTimerIdx, uint8 FrIf_Cycle, uint16 FrIf_Offset)	
Service ID[hex]:	0x19	
Sync/Async:	Synchronous	
Reentrancy:	non reentrant for the same FlexRay CC, reentrant for different FlexRay CCs	
Parameters (in):	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.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	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.
Description:	Wraps the FlexRay Driver API function Fr_SetAbsoluteTimer().	

] ()

[SWS_Frlf_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_Frlf_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_Frlf_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_Frlf_05003.] ()

8.3.4 FrIf_EnableAbsoluteTimerIRQ

[SWS_Frlf_05025] [

Service name:	FrIf_EnableAbsoluteTimerIRQ	
Syntax:	Std_ReturnType FrIf_EnableAbsoluteTimerIRQ(uint8 FrIf_CtrlIdx, uint8 FrIf_AbsTimerIdx)	
Service ID[hex]:	0x1d	
Sync/Async:	Synchronous	
Reentrancy:	non reentrant for the same FlexRay CC, reentrant for different FlexRay CCs	
Parameters (in):	FrIf_CtrlIdx	Index of the FlexRay CC to address.
	FrIf_AbsTimerIdx	Index of the absolute timer to address.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	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.
Description:	Wraps the FlexRay Driver API function Fr_EnableAbsoluteTimerIRQ().	

] ()

[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] [

Service name:	FrIf_AckAbsoluteTimerIRQ	
Syntax:	Std_ReturnType FrIf_AckAbsoluteTimerIRQ(uint8 FrIf_CtrlIdx, uint8 FrIf_AbsTimerIdx)	

Service ID[hex]:	0x21	
Sync/Async:	Synchronous	
Reentrancy:	non reentrant for the same FlexRay CC, reentrant for different FlexRay CCs	
Parameters (in):	FrIf_CtrlIdx	Index of the FlexRay CC to address.
	FrIf_AbsTimerIdx	Index of the absolute timer to address.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	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.
Description:	Wraps the FlexRay Driver API function Fr_AckAbsoluteTimerIRQ()	

] ()

[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] [

Service name:	FrIf_StartCommunication	
Syntax:	Std_ReturnType FrIf_StartCommunication(uint8 FrIf_CtrlIdx)	
Service ID[hex]:	0x04	
Sync/Async:	Asynchronous	
Reentrancy:	non reentrant for identical values of FrIf_CtrlIdx, reentrant for different values of FrIf_CtrlIdx	
Parameters (in):	FrIf_CtrlIdx	Index of the FlexRay CC to address.
Parameters (inout):	None	

Parameters (out):	None	
Return value:	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.
Description:	Wraps the FlexRay Driver API function Fr_StartCommunication().	

] (SRS_Fr_05015)

[SWS_Frlf_05161] [If parameter Frlf_CtrlIdx of Frlf_StartCommunication has an invalid value and if development error detection is enabled (i.e. FrlfDevErrorDetect equals ON), the function Frlf_StartCommunication shall report development error code FRIF_E_INV_CTRL_IDX to the Det_ReportError service of the DET module.] ()

[SWS_Frlf_05162] [The function Frlf_StartCommunication shall wrap the FlexRay Driver API function Fr_StartCommunication() 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_StartCommunication() of the determined FlexRay Driver module with the parameters determined as described above.] ()

[SWS_Frlf_05163] [Caveats of Frlf_StartCommunication: 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] ()

8.3.7 Frlf_HaltCommunication

[SWS_Frlf_05006] [

Service name:	Frlf_HaltCommunication	
Syntax:	Std_ReturnType Frlf_HaltCommunication(uint8 Frlf_CtrlIdx)	
Service ID[hex]:	0x05	
Sync/Async:	Asynchronous	
Reentrancy:	non reentrant for identical values of Frlf_CtrlIdx, reentrant for different values of Frlf_CtrlIdx	
Parameters (in):	Frlf_CtrlIdx	Index of the FlexRay CC to address.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	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.
Description:	Wraps the FlexRay Driver API function Fr_HaltCommunication().	

] (SRS_BSW_00336, SRS_Fr_05063)

[SWS_Frlf_05164] ¶ If parameter `Frlf_CtrlIdx` of `Frlf_HaltCommunication` has an invalid value and if development error detection is enabled (i.e. `FrlfDevErrorDetect` equals ON), the function `Frlf_HaltCommunication` shall report development error code `FRIF_E_INV_CTRL_IDX` to the `Det_ReportError` service of the DET module.] ()

[SWS_Frlf_05165] ¶ The function `Frlf_HaltCommunication` shall wrap the FlexRay Driver API function `Fr_HaltCommunication()` 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_HaltCommunication()` of the determined FlexRay Driver module with the parameters determined as described above.] ()

[SWS_Frlf_05166] ¶ Caveats of `Frlf_HaltCommunication`: 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`] ()

8.3.8 Frlf_AbortCommunication

[SWS_Frlf_05007] [

Service name:	Frlf_AbortCommunication	
Syntax:	Std_ReturnType FrIf_AbortCommunication(uint8 FrIf_CtrlIdx)	
Service ID[hex]:	0x06	
Sync/Async:	Synchronous	
Reentrancy:	non reentrant for identical values of <code>Frlf_CtrlIdx</code> , reentrant for different values of <code>Frlf_CtrlIdx</code>	
Parameters (in):	<code>Frlf_CtrlIdx</code>	Index of the FlexRay CC to address.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	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.
Description:	Wraps the FlexRay Driver API function <code>Fr_AbortCommunication()</code> .	

] (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 `FrIf_AbortCommunication` shall report development error code `FRIF_E_INV_CTRL_IDX` to the `Det_ReportError` service of the DET module.] ()

[SWS_FrIf_05168] [The function `FrIf_AbortCommunication` shall wrap the FlexRay Driver API function `Fr_AbortCommunication()` 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_AbortCommunication()` of the determined FlexRay Driver module with the parameters determined as described above.] ()

[SWS_FrIf_05169] [Caveats of `FrIf_AbortCommunication`: 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`] ()

8.3.9 FrIf_GetState

[SWS_FrIf_05170] [

Service name:	<code>FrIf_GetState</code>	
Syntax:	<pre>Std_ReturnType FrIf_GetState(uint8 FrIf_ClstIdx, FrIf_StateType* FrIf_StatePtr)</pre>	
Service ID[hex]:	0x07	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	<code>FrIf_ClstIdx</code>	Index of the cluster addressed.
Parameters (inout):	None	
Parameters (out):	<code>FrIf_StatePtr</code>	Pointer to a memory location where the retrieved <code>FrIfState</code> will be stored
Return value:	<code>Std_ReturnType</code>	<p><code>E_OK</code>: Function was successfully executed. State transition request was accepted.</p> <p><code>E_NOT_OK</code>: Function execution failed due to detected errors. State transition request was not accepted.</p>
Description:	Get current <code>FrIf</code> state.	

] ()

[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] [

Service name:	<code>FrIf_SetState</code>	
Syntax:	<pre>Std_ReturnType FrIf_SetState(uint8 FrIf_ClstIdx, FrIf_StateTransitionType FrIf_StateTransition)</pre>	
Service ID[hex]:	0x08	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	<code>FrIf_ClstIdx</code>	Index of the cluster addressed.
	<code>FrIf_StateTransition</code>	Requested <code>FrIf</code> state transition.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	<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.
	Description: Requests <code>FrIf</code> state machine transition.	

] ()

[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_FrIf_05176] [Caveats of `FrIf_SetState`: 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`] ()

8.3.11 FrIf_SetWakeupChannel

[SWS_FrIf_05010] [

Service name:	FrIf_SetWakeupChannel	
Syntax:	Std_ReturnType FrIf_SetWakeupChannel (uint8 FrIf_CtrlIdx, Fr_ChannelType FrIf_ChnlIdx)	
Service ID[hex]:	0x09	
Sync/Async:	Synchronous	
Reentrancy:	non reentrant for identical values of FrIf_CtrlIdx, reentrant for different values of FrIf_CtrlIdx	
Parameters (in):	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.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	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.
Description:	Wraps the FlexRay Driver API function Fr_SetWakeupChannel(). The enum value "FR_CHANNEL_AB" shall not be used.	

] ()

[SWS_FrIf_05500] [If parameter FrIf_CtrlIdx of FrIf_SetWakeupChannel has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf_SetWakeupChannel shall report development error code FRIF_E_INV_CTRL_IDX to the Det_ReportError service of the DET module.] ()

[SWS_FrIf_05177] [If parameter FrIf_ChnlIdx of FrIf_SetWakeupChannel has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf_SetWakeupChannel shall report development error code FRIF_E_INV_CHNL_IDX to the Det_ReportError service of the DET module.] ()

[SWS_FrIf_05178] [The function FrIf_SetWakeupChannel shall wrap the FlexRay Driver API function Fr_SetWakeupChannel() 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_ChnlIdx to FrIf_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] [

Service name:	Frlf_SendWUP	
Syntax:	Std_ReturnType Frlf_SendWUP(uint8 Frlf_CtrlIdx)	
Service ID[hex]:	0x0a	
Sync/Async:	Asynchronous	
Reentrancy:	non reentrant for identical values of Frlf_CtrlIdx, reentrant for different values of Frlf_CtrlIdx	
Parameters (in):	Frlf_CtrlIdx	Index of the FlexRay CC to address.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	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.
Description:	Wraps the FlexRay Driver API function Fr_SendWUP().	

] (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_Frlf_05182] [Caveats of Frlf_SendWUP: 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.] ()

8.3.13 FrIf_GetPOCStatus

[SWS_FrIf_05014] |

Service name:	FrIf_GetPOCStatus	
Syntax:	Std_ReturnType FrIf_GetPOCStatus(uint8 FrIf_CtrlIdx, Fr_POCStatusType* FrIf_POCStatusPtr)	
Service ID[hex]:	0x0d	
Sync/Async:	Synchronous	
Reentrancy:	non reentrant for identical values of FrIf_CtrlIdx, reentrant for different values of FrIf_CtrlIdx	
Parameters (in):	FrIf_CtrlIdx	Index of the FlexRay CC to address.
Parameters (inout):	None	
Parameters (out):	FrIf_POCStatusPtr	Pointer to a memory location where output value will be stored.
Return value:	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.
Description:	Wraps the FlexRay Driver API function Fr_GetPOCStatus().	

| (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_FrIf_05193] | Caveats of FrIf_GetPOCStatus: 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. | ()

8.3.14 FrIf_GetGlobalTime

[SWS_FrIf_05015] |

Service name:	FrIf_GetGlobalTime	
Syntax:	<pre>Std_ReturnType FrIf_GetGlobalTime (uint8 FrIf_CtrlIdx, uint8* FrIf_CyclePtr, uint16* FrIf_MacroTickPtr)</pre>	
Service ID[hex]:	0x0e	
Sync/Async:	Synchronous	
Reentrancy:	non reentrant for identical values of FrIf_CtrlIdx, reentrant for different values of FrIf_CtrlIdx	
Parameters (in):	FrIf_CtrlIdx	Index of the FlexRay CC to address.
Parameters (inout):	None	
Parameters (out):	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.
Return value:	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>
Description:	Wraps the FlexRay Driver API function Fr_GetGlobalTime().	

| ()

[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] [

Service name:	FrIf_AllowColdstart	
Syntax:	Std_ReturnType FrIf_AllowColdstart (uint8 FrIf_CtrlIdx)	
Service ID[hex]:	0x10	
Sync/Async:	Asynchronous	
Reentrancy:	non reentrant for identical values of FrIf_CtrlIdx, reentrant for different values of FrIf_CtrlIdx	
Parameters (in):	FrIf_CtrlIdx	Index of the FlexRay CC to address.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	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.
Description:	Wraps the FlexRay Driver API function Fr_AllowColdstart().	

] ()

[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] [

Service name:	FrIf_GetMacroTicksPerCycle	
Syntax:	uint16 FrIf_GetMacroTicksPerCycle (uint8 FrIf_CtrlIdx)	
Service ID[hex]:	0x11	

Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	FrIf_CtrlIdx	Index of the FlexRay CC to address.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	uint16	Number of Macroticks per Cycle
Description:	Retrieves the amount of Macroticks per Cycle	

] ()

[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] [

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

] ()

[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 FrIf_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 `FrIf_CtrlIdx` out of the static configuration. `] ()`

[SWS_FrIf_05754] [Caveats of `FrIf_GetMacroTICKDuration`: 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`] `()`

8.3.18 `FrIf_Transmit`

[SWS_FrIf_05033] [

Service name:	<code>FrIf_Transmit</code>	
Syntax:	<pre>Std_ReturnType FrIf_Transmit(PduIdType TxPduId, const PduInfoType* PduInfoPtr)</pre>	
Service ID[hex]:	0x49	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant for different PduIds. Non reentrant for the same PduId.	
Parameters (in):	<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.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	<code>Std_ReturnType</code>	<code>E_OK</code> : Transmit request has been accepted. <code>E_NOT_OK</code> : Transmit request has not been accepted.
	Description: Requests transmission of a PDU.	

`] ()`

[SWS_FrIf_05318]

`FrIf_Transmit()` shall return `E_NOT_OK` in case the `FrIf`'s state is `FRIF_STATE_OFFLINE`.

[SWS_FrIf_05205] [If parameter `TxPduId` of `FrIf_Transmit` has an invalid value and if development error detection is enabled (i.e. `FrIfDevErrorDetect` equals `ON`), the function `FrIf_Transmit` shall report development error code `FRIF_E_INV_TXPDUID` to the `Det_ReportError` service of the `DET` module. `] ()`

[SWS_FrIf_05207] [If the parameter `FrIfAllowedDynamicSduLength` is set to `false` and/or if the parameter `FrIfImmediate` is set to `true` for the passed `TxPduId`, the passed `SduDataPtr` in parameter `PduInfoPtr` of `FrIf_Transmit` shall be checked for `NULL_PTR` in case development error detection is enabled (i.e. `FrIfDevErrorDetect` 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 TrigTxCounter5). 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] [

Service name:	FrIf_SetTransceiverMode	
Syntax:	Std_ReturnType FrIf_SetTransceiverMode (uint8 FrIf_CtrlIdx, Fr_ChannelType FrIf_ChnlIdx, FrTrcv_TrcevModeType FrIf_TrcevMode)	
Service ID[hex]:	0x13	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	FrIf_CtrlIdx	Index of the FlexRay CC to address.
	FrIf_ChnlIdx	Index of the FlexRay Channel to address in scope of the FlexRay

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

		controller FrIf_CtrlIdx.
	FrIf_TrvcvMode	Transceiver mode to be set.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	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.
Description:	Wraps the FlexRay Transceiver Driver API function FrTrcv_SetTransceiverMode(). The enum value "FR_CHANNEL_AB" shall not be used.	

] (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_TrvcvIdx).
2. Setting parameters
 - FrTrcv_TrvcvMode to FrIf_TrvcvMode
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] |

Service name:	FrIf_GetTransceiverMode
Syntax:	Std_ReturnType FrIf_GetTransceiverMode (uint8 FrIf_CtrlIdx, Fr_ChannelType FrIf_ChnlIdx, FrTrcv_TrvcvModeType* FrIf_TrvcvModePtr)

Service ID[hex]:	0x14	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	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.
Parameters (inout):	None	
Parameters (out):	FrIf_TrcvModePtr	Pointer to a memory location where output value will be stored.
Return value:	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.
Description:	Wraps the FlexRay Transceiver Driver API function FrTrcv_GetTransceiverMode(). The enum value "FR_CHANNEL_AB" shall not be used.	

] (SRS_Fr_05157)

[SWS_FrIf_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_FrIf_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_FrIf_05216] [The function FrIf_GetTransceiverMode shall wrap the FlexRay Transceiver Driver API function FrTrcv_GetTransceiverMode() 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_TrcvModePtr to FrIf_TrcvModePtr
3. Calling FrTrcv_GetTransceiverMode() of the determined FlexRay Driver module with the parameters determined as described above.] ()

[SWS_FrIf_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_FrIf_05003.] ()

8.3.21 FrIf_GetTransceiverWUReason

[SWS_FrIf_05036] [

Service name:	FrIf_GetTransceiverWUReason
----------------------	-----------------------------

Syntax:	<pre>Std_ReturnType FrIf_GetTransceiverWUReason(uint8 FrIf_CtrlIdx, Fr_ChannelType FrIf_ChnlIdx, FrTrcv_TrcvWUReasonType* FrIf_TrcvWUReasonPtr)</pre>	
Service ID[hex]:	0x15	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	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.
Parameters (inout):	None	
Parameters (out):	FrIf_TrcvWUReasonPtr	Pointer to a memory location where output value will be stored.
Return value:	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.
Description:	Wraps the FlexRay Transceiver Driver API function FrTrcv_GetTransceiverWUReason(). The enum value "FR_CHANNEL_AB" shall not be used.	

└ (SRS_BSW_00375, SRS_Fr_05158)

[SWS_Frlf_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_Frlf_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_Frlf_05220] └ The function FrIf_GetTransceiverWUReason shall wrap the FlexRay Transceiver Driver API function FrTrcv_GetTransceiverWUReason() 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_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 FrIf_GetTransceiverWUReason: 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. └ ()

8.3.22 FrIf_ClearTransceiverWakeup

[SWS_FrIf_05039] [

Service name:	FrIf_ClearTransceiverWakeup	
Syntax:	Std_ReturnType FrIf_ClearTransceiverWakeup(uint8 FrIf_CtrlIdx, Fr_ChannelType FrIf_ChnlIdx)	
Service ID[hex]:	0x18	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	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.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	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.
Description:	Wraps the FlexRay Transceiver Driver API function FrTrcv_ClearTransceiverWakeup(). The enum value "FR_CHANNEL_AB" shall not be used.	

] (SRS_Fr_05161)

[SWS_FrIf_05230] [If parameter FrIf_CtrlIdx of FrIf_ClearTransceiverWakeup has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf_ClearTransceiverWakeup shall report development error code FRIF_E_INV_CTRL_IDX to the Det_ReportError service of the DET module.] ()

[SWS_FrIf_05231] [If parameter FrIf_ChnlIdx of FrIf_ClearTransceiverWakeup has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf_ClearTransceiverWakeup shall report development error code FRIF_E_INV_CHNL_IDX to the Det_ReportError service of the DET module.] ()

[SWS_FrIf_05232] [The function FrIf_ClearTransceiverWakeup shall wrap the FlexRay Transceiver Driver API function FrTrcv_ClearTransceiverWakeup() 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_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] |

Service name:	FrIf_CancelAbsoluteTimer	
Syntax:	Std_ReturnType FrIf_CancelAbsoluteTimer(uint8 FrIf_CtrlIdx, uint8 FrIf_AbsTimerIdx)	
Service ID[hex]:	0x1b	
Sync/Async:	Synchronous	
Reentrancy:	non reentrant for the same FlexRay CC, reentrant for different FlexRay CCs	
Parameters (in):	FrIf_CtrlIdx	Index of the FlexRay CC to address.
	FrIf_AbsTimerIdx	Index of the absolute timer to address.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	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.
Description:	Wraps the FlexRay Driver API function Fr_CancelAbsoluteTimer() .	

| ()

[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_FrIf_05242] | Caveats of FrIf_CancelAbsoluteTimer: 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. | ()

8.3.24 FrIf_GetAbsoluteTimerIRQStatus

[SWS_FrIf_05027] |

Service name:	FrIf_GetAbsoluteTimerIRQStatus	
Syntax:	<pre>Std_ReturnType FrIf_GetAbsoluteTimerIRQStatus (uint8 FrIf_CtrlIdx, uint8 FrIf_AbsTimerIdx, boolean* FrIf_IRQStatusPtr)</pre>	
Service ID[hex]:	0x1f	
Sync/Async:	Synchronous	
Reentrancy:	non reentrant for the same FlexRay CC, reentrant for different FlexRay CCs	
Parameters (in):	FrIf_CtrlIdx	Index of the FlexRay CC to address.
	FrIf_AbsTimerIdx	Index of the absolute timer to address.
Parameters (inout):	None	
Parameters (out):	FrIf_IRQStatusPtr	Pointer to a memory location where output value will be stored.
Return value:	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.
Description:	Wraps the FlexRay Driver API function Fr_GetAbsoluteTimerIRQStatus()	

| ()

[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] [

Service name:	FrIf_DisableAbsoluteTimerIRQ	
Syntax:	Std_ReturnType FrIf_DisableAbsoluteTimerIRQ(uint8 FrIf_CtrlIdx, uint8 FrIf_AbsTimerIdx)	
Service ID[hex]:	0x23	
Sync/Async:	Synchronous	
Reentrancy:	non reentrant for the same FlexRay CC, reentrant for different FlexRay CCs	
Parameters (in):	FrIf_CtrlIdx	Index of the FlexRay CC to address.
	FrIf_AbsTimerIdx	Index of the absolute timer to address.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	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.
Description:	Wraps the FlexRay Driver API function Fr_DisableAbsoluteTimerIRQ().	

] ()

[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] [

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

Parameters (out):	None	
Return value:	uint32	Time in unit of nanoseconds
Description:	This API returns the configured time of the configuration parameter "GdCycle" in nanoseconds for the FlexRay controller with index FrIf_CtrlIdx.	

] ()

[SWS_FrIf_05237] ¶ If parameter FrIf_CtrlIdx of FrIf_GetCycleLength has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function FrIf_GetCycleLength shall report development error code FRIF_E_INV_CTRL_IDX to the Det_ReportError service of the DET module.] ()

[SWS_FrIf_05238] ¶ Caveats of FrIf_GetCycleLength: 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 Optional Function Definitions

8.4.1 FrIf_AllSlots

[SWS_FrIf_05020] [

Service name:	FrIf_AllSlots	
Syntax:	Std_ReturnType FrIf_AllSlots(uint8 FrIf_CtrlIdx)	
Service ID[hex]:	0x33	
Sync/Async:	Synchronous	
Reentrancy:	non reentrant	
Parameters (in):	FrIf_CtrlIdx	Index of the FlexRay CC to address.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	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.
Description:	Wraps the FlexRay Driver API function Fr_AllSlots	

] ()

[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] [

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

] ()

[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] [

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

] ()

[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] [

Service name:	FrIf_GetSyncFrameList	
Syntax:	<pre>Std_ReturnType FrIf_GetSyncFrameList(uint8 FrIf_CtrlIdx, uint8 FrIf_ListSize, uint16* FrIf_ChannelAEvenListPtr, uint16* FrIf_ChannelBEvenListPtr, uint16* FrIf_ChannelAOddListPtr, uint16* FrIf_ChannelBOddListPtr)</pre>	
Service ID[hex]:	0x2a	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant for the same device	
Parameters (in):	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.
Parameters (inout):	None	
Parameters (out):	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.
Return value:	Std_ReturnType	E_OK: API call finished successfully. E_NOT_OK: API call aborted due to errors.
Description:	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.	

] ()

[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_FrIf_05716] ¶If development error detection for the `FrIf` module is enabled: the function `FrIf_GetSyncFrameList` shall check the parameter `FrIf_CtrlIdx` for being valid. If `FrIf_CtrlIdx` is invalid, the function `FrIf_GetSyncFrameList` shall raise the development error `FRIF_E_INV_CTRL_IDX` and return `E_NOT_OK`. ¶ ()

8.4.5 `FrIf_GetNumOfStartupFrames`

[SWS_FrIf_05073] ¶

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

¶ ()

[SWS_FrIf_05416] ¶The function `FrIf_GetNumOfStartupFrames` shall be pre compile time configurable ON/OFF by the configuration parameter `FrIfGetNumOfStartupFramesSupport` (derived from configuration parameter `FrIfGetNumOfStartupFramesSupport`, see ECUC_FrIf_06104) ¶ ()

[SWS_FrIf_05721] ¶If development error detection for the `FrIf` module is enabled: if the function `FrIf_GetNumOfStartupFrames` is called before the `FrIf` was initialized

successfully, the function `FrIf_GetNumOfStartupFrames` shall raise the development error `FRIF_E_UNINIT` and return `E_NOT_OK`. `] ()`

[SWS_FrIf_05722] `] If development error detection for the FrIf module is enabled: the function FrIf_GetNumOfStartupFrames shall check the parameter FrIf_CtrlIdx for being valid. If FrIf_CtrlIdx is invalid, the function FrIf_GetNumOfStartupFrames shall raise the development error FRIF_E_INV_CTRL_IDX and return E_NOT_OK.] ()`

8.4.6 `FrIf_GetWakeupRxStatus`

[SWS_FrIf_05102] `[`

Service name:	<code>FrIf_GetWakeupRxStatus</code>	
Syntax:	<pre>Std_ReturnType FrIf_GetWakeupRxStatus (uint8 FrIf_CtrlIdx, uint8* FrIf_WakeupRxStatusPtr)</pre>	
Service ID[hex]:	0x2b	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant for the same device	
Parameters (in):	<code>FrIf_CtrlIdx</code>	Index of FlexRay CC within the context of the FlexRay Driver.
Parameters (inout):	None	
Parameters (out):	<code>FrIf_WakeupRxStatusPtr</code>	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
Return value:	<code>Std_ReturnType</code>	<code>E_OK</code> : API call finished successfully. <code>E_NOT_OK</code> : API call aborted due to errors.
Description:	Wraps the FlexRay Driver API function <code>Fr_GetWakeupRxStatus</code> and gets the wakeup received information from the FlexRay controller.	

`] ()`

[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] [

Service name:	FrIf_CancelTransmit	
Syntax:	Std_ReturnType FrIf_CancelTransmit(PduIdType TxPduId)	
Service ID[hex]:	0x4a	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant for different PduIds. Non reentrant for the same PduId.	
Parameters (in):	TxPduId	Identification of the PDU to be cancelled.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: Cancellation was executed successfully by the destination module. E_NOT_OK: Cancellation was rejected by the destination module.
Description:	Requests cancellation of an ongoing transmission of a PDU in a lower layer communication module.	

] ()

[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 TxPduId for being valid. If TxPduId 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] [

Service name:	FrIf_DisableLPdu	
Syntax:	Std_ReturnType FrIf_DisableLPdu(uint8 FrIf_CtrlIdx, uint16 FrIf_LPduIdx)	
Service ID[hex]:	0x28	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant for the same device	
Parameters (in):	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
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: API call finished successfully. E_NOT_OK: API call aborted due to errors.
Description:	Wraps the FlexRay Driver Function Fr_DisableLPdu. It disables the hardware resource of an LPdu for transmission/reception.	

] ()

[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] |

Service name:	FrIf_GetTransceiverError	
Syntax:	<pre>Std_ReturnType FrIf_GetTransceiverError(uint8 FrIf_CtrlIdx, Fr_ChannelType FrIf_ChnlIdx, uint8 FrIf_BranchIdx, uint32* FrIf_BusErrorState)</pre>	
Service ID[hex]:	0x35	
Sync/Async:	Synchronous	
Reentrancy:	Function is non reentrant for the same channel of the same controller.	
Parameters (in):	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.
Parameters (inout):	None	
Parameters (out):	FrIf_BusErrorState	Address where the transceiver error state is stored.
Return value:	Std_ReturnType	E_OK: API call finished successfully. E_NOT_OK: API call aborted due to errors
Description:	Wraps the FlexRay Transceiver Driver API function FrTrcv_GetTransceiverError. The enum value "FR_CHANNEL_AB" shall not be used.	

| ()

[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 [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 parameters
 - `FrTrcv_BranchIdx` to `Frlf_BranchIdx`
 - `FrTrcv_BusErrorState` to `Frlf_BusErrorState`
3. Calling `FrTrcv_GetTransceiverError` of the determined FlexRay Transceiver module with the parameters determined as described above. `] ()`

8.4.10 `Frlf_EnableTransceiverBranch`

[SWS_Frlf_05085] `] ()`

Service name:	<code>Frlf_EnableTransceiverBranch</code>	
Syntax:	<pre>Std_ReturnType Frlf_EnableTransceiverBranch(uint8 Frlf_CtrlIdx, Fr_ChannelType Frlf_ChnlIdx, uint8 Frlf_BranchIdx)</pre>	
Service ID[hex]:	0x36	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	<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>	Index of the FlexRay Channel to address in scope of the FlexRay controller <code>Frlf_CtrlIdx</code> .
Parameters (inout):	None	
Parameters (out):	None	
Return value:	<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> .
Description:	Wraps the FlexRay Transceiver Driver API function <code>FrTrcv_EnableTransceiverBranch</code> . The enum value "FR_CHANNEL_AB" shall not be used.	

`] ()`

[SWS_Frlf_05420] `] ()` The function `Frlf_EnableTransceiverBranch` shall be pre compile time configurable ON/OFF by the configuration parameter `FrlfEnableTransceiverBranchSupport` (derived from configuration parameter `FrlfEnableTransceiverBranchSupport`, see [ECUC_Frlf_06103](#)) `] ()`

[SWS_Frlf_05302] `] ()` If parameter `Frlf_CtrlIdx` of `Frlf_EnableTransceiverBranch` has an invalid value and if development error detection is enabled (i.e. `FrlfDevErrorDetect` equals ON), the function `Frlf_EnableTransceiverBranch` shall report development error code `FRIF_E_INV_CTRL_IDX` to the `Det_ReportError` service of the DET module. `] ()`

[SWS_Frlf_05304] [If parameter `Frlf_ChnlIdx` of `Frlf_EnableTransceiverBranch` has an invalid value and if development error detection is enabled (i.e. `FrlfDevErrorDetect` equals ON), the function `Frlf_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] [

Service name:	<code>Frlf_DisableTransceiverBranch</code>	
Syntax:	<pre>Std_ReturnType Frlf_DisableTransceiverBranch (uint8 Frlf_CtrlIdx, Fr_ChannelType Frlf_ChnlIdx, uint8 Frlf_BranchIdx)</pre>	
Service ID[hex]:	0x37	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	<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>	Index of the FlexRay Channel to address in scope of the FlexRay controller <code>Frlf_CtrlIdx</code> .
Parameters (inout):	None	
Parameters (out):	None	

Return value:	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.
Description:	Wraps the FlexRay Transceiver Driver API function FrTrcv_DisableTransceiverBranch. The enum value "FR_CHANNEL_AB" shall not be used.	

] ()

[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 FrIf_ReconfigLPdu

[SWS_FrIf_05048] [

Service name:	FrIf_ReconfigLPdu	
Syntax:	<pre>Std_ReturnType FrIf_ReconfigLPdu(uint8 FrIf_CtrlIdx, uint16 FrIf_LPduIdx, uint16 FrIf_FrameId, Fr_ChannelType FrIf_ChnlIdx, uint8 FrIf_CycleRepetition, uint8 FrIf_CycleOffset, uint8 FrIf_PayloadLength, uint16 FrIf_HeaderCRC)</pre>	
Service ID[hex]:	0	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	FrIf_CtrlIdx	Index of FlexRay CC within the context of the FlexRay Driver.
	FrIf_LPduIdx	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_LPduIdx shall be configured to.
	FrIf_HeaderCRC	Header CRC the FrIf_LPdu shall be configured to.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: API call finished successfully. E_NOT_OK: API call aborted due to errors.
Description:	Calls the FlexRay Driver's API Fr_ReconfigLPdu. The enum value "FR_CHANNEL_AB" shall not be used.	

] ()

[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] [

Service name:	FrIf_GetNmVector	
Syntax:	<pre>Std_ReturnType FrIf_GetNmVector(uint8 FrIf_CtrlIdx, uint8* FrIf_NmVectorPtr)</pre>	
Service ID[hex]:	0x0f	
Sync/Async:	Synchronous	
Reentrancy:	non reentrant for identical values of <code>FrIf_CtrlIdx</code> , reentrant for different values of <code>FrIf_CtrlIdx</code>	
Parameters (in):	<code>FrIf_CtrlIdx</code>	Index of the FlexRay CC to address.
Parameters (inout):	None	
Parameters (out):	<code>FrIf_NmVectorPtr</code>	Pointer to a memory location where output value will be stored.
Return value:	<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>
Description:	Derives the FlexRay NM Vector.	

] ()

[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_Frlf_05198] [The function Frlf_GetNmVector wraps the FlexRay Driver API Fr_GetNmVector function.] ()

[SWS_Frlf_05199] [Caveats of Frlf_GetNmVector: 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] ()

8.4.14 Frlf_GetVersionInfo

[SWS_Frlf_05002] [

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

] (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] [

Service name:	Frlf_ReadCCConfig	
Syntax:	<pre>Std_ReturnType Frlf_ReadCCConfig(uint8 Frlf_CtrlIdx, uint8 Frlf_ConfigParamIdx, uint32* Frlf_ConfigParamValuePtr)</pre>	
Service ID[hex]:	0x3b	

Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant for the same FlexRay CC, reentrant for different FlexRay CCs	
Parameters (in):	FrIf_CtrlIdx	Index of the FlexRay CC to address.
	FrIf_ConfigParamIdx	Index of the configuration parameter to read.
Parameters (inout):	None	
Parameters (out):	FrIf_ConfigParamValuePtr	Pointer to a memory location where output value will be stored.
Return value:	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.
Description:	Wraps the FlexRay Driver API function Fr_ReadCCConfig().	

] ()

[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.] ()

8.5 Interrupt Service Routines

8.5.1 FrIf_JobListExec_<ClstIdx>

[SWS_FrIf_05040] [

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

] ()

Note:

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

[SWS_FrIf_05270] [The function FrIf_JobListExec_<ClstIdx> shall exist once per FlexRay Cluster of a FlexRay Interface module.] ()

[SWS_FrIf_05271] [The function name of each instance of FrIf_JobListExec_<ClstIdx> shall contain the index of the respective FlexRay Cluster (ClstIdx).

For each FlexRay Cluster (identified by index ClstIdx), the respective API service FrIf_JobListExec_<ClstIdx> 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<ClstIdx> can run in a regular OS task.

[SWS_FrIf_05272] [Caveats of FrIf_JobListExec_<ClstIdx>: 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_Frlf_05041] |

Service name:	Frlf_CheckWakeupByTransceiver	
Syntax:	<pre>void FrIf_CheckWakeupByTransceiver(uint8 FrIf_CtrlIdx, Fr_ChannelType FrIf_ChnlIdx)</pre>	
Service ID[hex]:	0x39	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	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.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	Wraps the FlexRay Transceiver Driver API function FrTrcv_CheckWakeupByTransceiver(). The enum value "FR_CHANNEL_AB" shall not be used.	

| ()

[SWS_Frlf_05274] | If parameter FrIf_CtrlIdx of Frlf_CheckWakeupByTransceiver has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function Frlf_CheckWakeupByTransceiver shall report development error code FRIF_E_INV_CTRL_IDX to the Det_ReportError service of the DET module. | ()

[SWS_Frlf_05275] | If parameter FrIf_ChnlIdx of Frlf_CheckWakeupByTransceiver has an invalid value and if development error detection is enabled (i.e. FrIfDevErrorDetect equals ON), the function Frlf_CheckWakeupByTransceiver shall report development error code FRIF_E_INV_CHNL_IDX to the Det_ReportError service of the DET module. | ()

[SWS_Frlf_05276] | The function Frlf_CheckWakeupByTransceiver shall wrap the FlexRay Transceiver Driver API function FrTrcv_CheckWakeupByTransceiver() 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_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_<ClstIdx>

[SWS_Frlf_05042] [

Service name:	Frlf_MainFunction_<ClstIdx>
Syntax:	void Frlf_MainFunction_<ClstIdx> (void)
Service ID[hex]:	0x27
Description:	This function will be called cyclically by a task body provided by the BSW Scheduler.

] ()

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_<ClstIdx>() if the CC does not support asynchronous buffer access.
- Monitoring the proper (in time) execution of the Frlf_JobListExec_<ClstIdx>() and resynchronize the Joblist if necessary.

Please refer to chapter 7.3 for a detailed description.

Pre condition: The function Frlf_MainFunction_<ClstIdx> 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_<ClstIdx>` shall exist once per FlexRay Cluster of a FlexRay Interface module.] ()

[SWS_Frlf_05279] [The function name of each instance of `Frlf_MainFunction_<ClstIdx>` shall contain the index of the respective FlexRay Cluster (`ClstIdx`).] ()

[SWS_Frlf_05280] [Caveats of `Frlf_MainFunction_<ClstIdx>`: The FlexRay Interface has to be initialized with a call of `Frlf_Init()` before this API service may be called, see `SWS_Frlf_05003`.] ()

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>Description</i>
<code>Det_ReportRuntimeError</code>	Service to report runtime errors. If a callout has been configured then this callout shall be called.
<code>Fr_AbortCommunication</code>	Invokes the CC CHI command 'FREEZE'.
<code>Fr_AckAbsoluteTimerIRQ</code>	Resets the interrupt condition of an absolute timer.
<code>Fr_AllowColdstart</code>	Invokes the CC CHI command 'ALLOW_COLDSTART'.
<code>Fr_CancelAbsoluteTimer</code>	Stops an absolute timer.
<code>Fr_CheckTxLPduStatus</code>	Checks the transmit status of the LSdu.
<code>Fr_ControllerInit</code>	Initializes a FlexRay CC.
<code>Fr_DisableAbsoluteTimerIRQ</code>	Disables the interrupt line of an absolute timer.
<code>Fr_EnableAbsoluteTimerIRQ</code>	Enables the interrupt line of an absolute timer.
<code>Fr_GetAbsoluteTimerIRQStatus</code>	Gets IRQ status of an absolute timer.
<code>Fr_GetGlobalTime</code>	Gets the current global FlexRay time.
<code>Fr_GetPOCStatus</code>	Gets the POC status.
<code>Fr_HaltCommunication</code>	Invokes the CC CHI command 'DEFERRED_HALT'.
<code>Fr_ReceiveRxLPdu</code>	Receives data from the FlexRay network.
<code>Fr_SendWUP</code>	Invokes the CC CHI command 'WAKEUP'.
<code>Fr_SetAbsoluteTimer</code>	Sets the absolute FlexRay timer.
<code>Fr_SetWakeupChannel</code>	Sets a wakeup channel.
<code>Fr_StartCommunication</code>	Starts communication.
<code>Fr_TransmitTxLPdu</code>	Transmits data on the FlexRay network.
<code>FrTrcv_CheckWakeupByTransceiver</code>	--
<code>FrTrcv_ClearTransceiverWakeup</code>	This function clears a pending wake up event.
<code>FrTrcv_GetTransceiverMode</code>	This function returns the actual state of the transceiver.
<code>FrTrcv_GetTransceiverWUReason</code>	This function returns the wakeup reason.
<code>FrTrcv_SetTransceiverMode</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	Description
Dem_SetEventStatus	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	Service to report development errors.
Fr_AllSlots	Invokes the CC CHI command 'ALL_SLOTS'.
Fr_CancelTxLPdu	Cancels the already pending transmission of a LPdu contained in a controllers physical transmit resource (e.g. message buffer).
Fr_DisableLPdu	Disables the hardware resource of a LPdu for transmission/reception.
Fr_GetChannelStatus	Gets the channel status information.
Fr_GetClockCorrection	Gets the current clock correction values. See variables vInterimRateCorrection and vInterimOffsetCorrection of [12] for details.
Fr_GetNmVector	Gets the network management vector of the last communication cycle.
Fr_GetNumOfStartupFrames	Gets the current number of startup frames seen on the cluster. See variable vStartupPairs of [12] for details.
Fr_GetSyncFrameList	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	Gets the wakeup received information from the FlexRay controller.
Fr_PrepareLPdu	Prepares a LPdu.
Fr_ReadCCConfig	Reads a FlexRay protocol configuration parameter for a particular FlexRay controller out of the module's configuration.
Fr_ReconfigLPdu	Reconfigures a given LPdu according to the parameters (FrameId, Channel, CycleRepetition, CycleOffset, PayloadLength, HeaderCRC) at runtime.
FrArTp_RxIndication	Indication of a received PDU from a lower layer communication interface module.
FrArTp_TriggerTransmit	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	The lower layer communication interface module confirms the transmission of a PDU, or the failure to transmit a PDU.
FrNm_RxIndication	Indication of a received PDU from a lower layer communication interface module.
FrNm_TriggerTransmit	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	The lower layer communication interface module confirms the transmission of a PDU, or the failure to transmit a PDU.
FrTp_RxIndication	Indication of a received PDU from a lower layer communication interface module.
FrTp_TriggerTransmit	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	The lower layer communication interface module confirms the transmission of a PDU, or the failure to transmit a PDU.
FrTrcv_DisableTransceiverBranch	This function disables the specified branch on the addressed (active star) transceiver.
FrTrcv_EnableTransceiverBranch	This function enables the specified branch on the addressed (active star) transceiver.
FrTrcv_GetTransceiverError	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.
PduR_FrlfRxIndication	Indication of a received PDU from a lower layer communication interface module.
PduR_FrlfTriggerTransmit	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	The lower layer communication interface module confirms the transmission of a PDU, or the failure to transmit a PDU.
Xcp_FrlfRxIndication	Indication of a received PDU from a lower layer communication interface module.
Xcp_FrlfTriggerTransmit	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	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] |

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

| ()

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] |

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

| ()

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] |

Service name:	<User_TriggerTransmit>	
Syntax:	Std_ReturnType <User_TriggerTransmit>(PduIdType TxPduId, PduInfoType* PduInfoPtr)	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant for different Pdulds. Non reentrant for the same Pduld.	
Parameters (in):	TxPduld	ID of the SDU that is requested to be transmitted.
Parameters (inout):	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.
Parameters (out):	None	
Return value:	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.
Description:	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.	

| ()

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] |

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

| ()

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] [

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

] ()

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] [

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

] ()

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

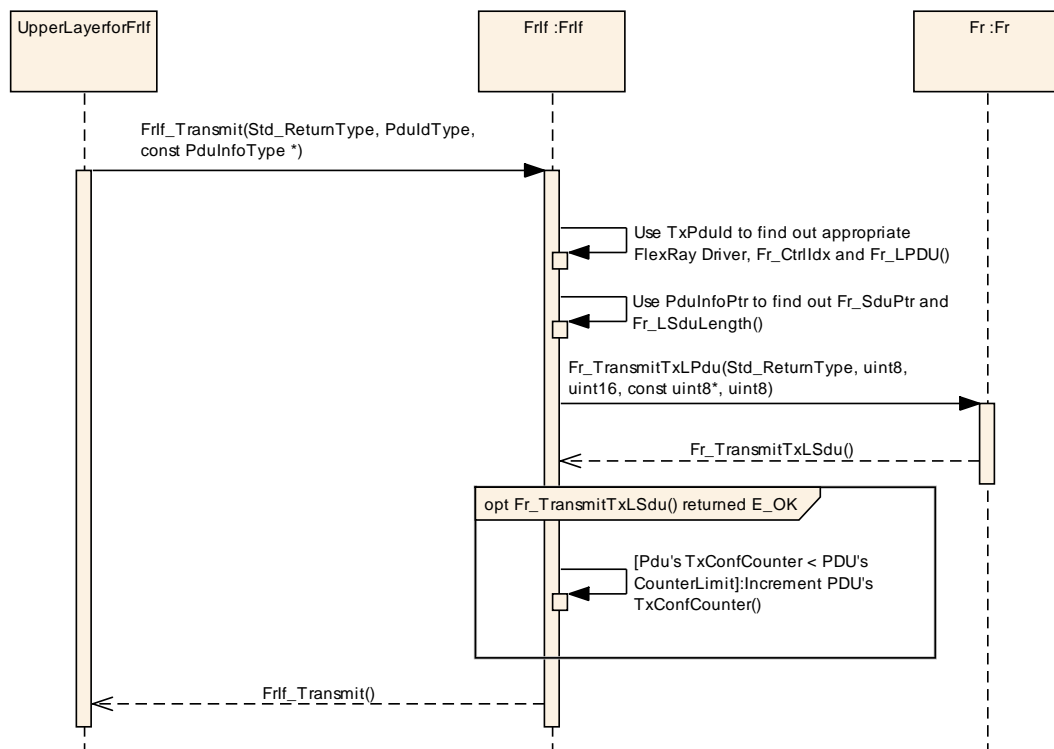


Figure 9-1: TransmitWithImmediateBufferAccess

9.1.2 TransmitWithDecoupledBufferAccess

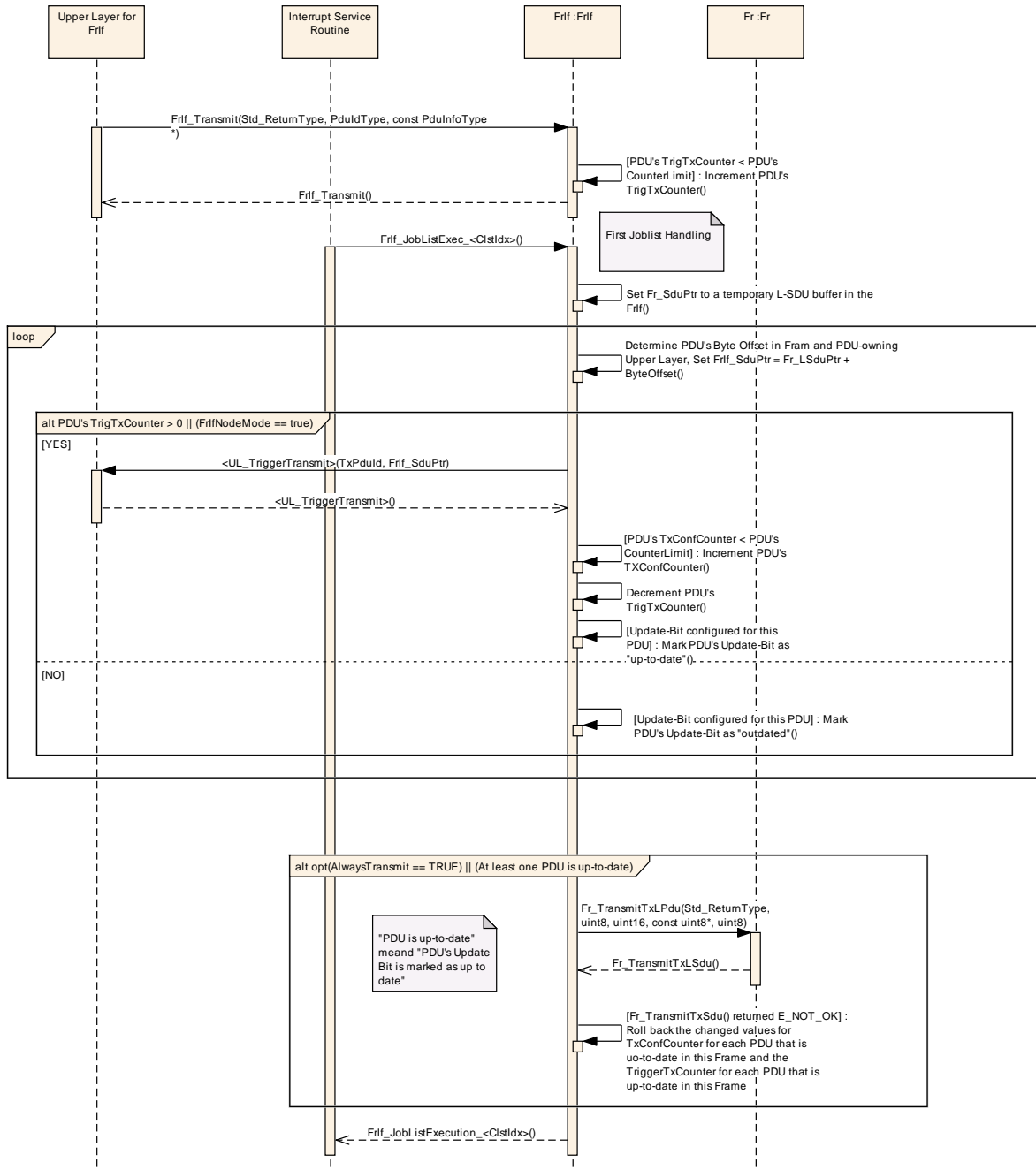


Figure 9-2: TransmitWithDecoupledBufferAccess

9.1.3 ProvideTxConfirmation

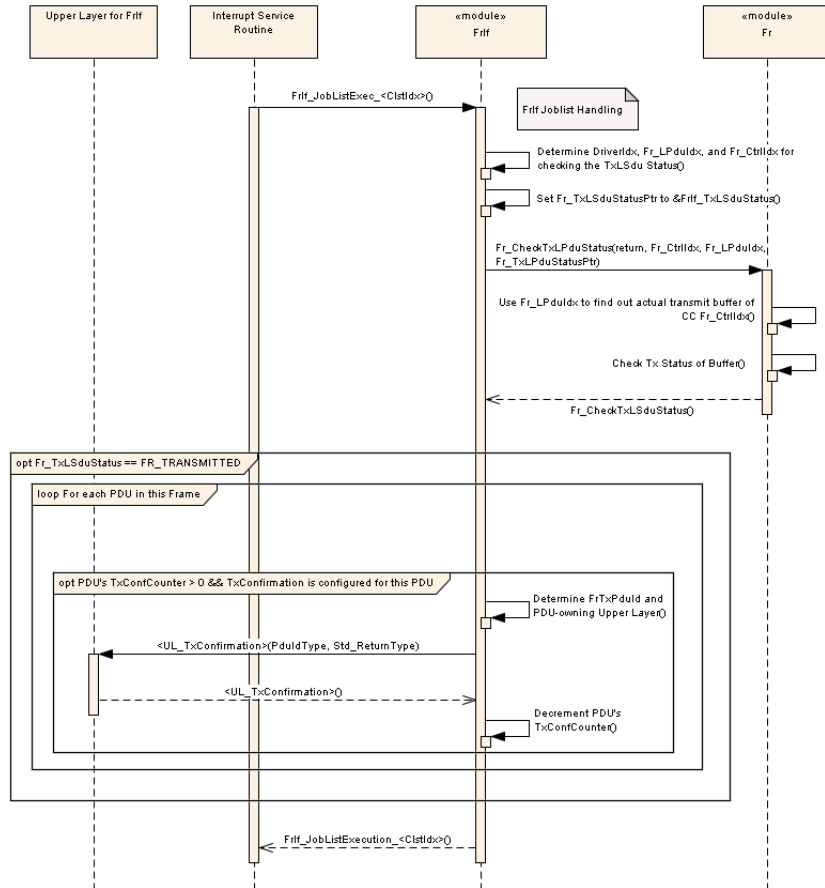


Figure 9-3: ProvideTxConfirmation

9.2 Data Reception

9.2.1 ReceiveAndIndicate

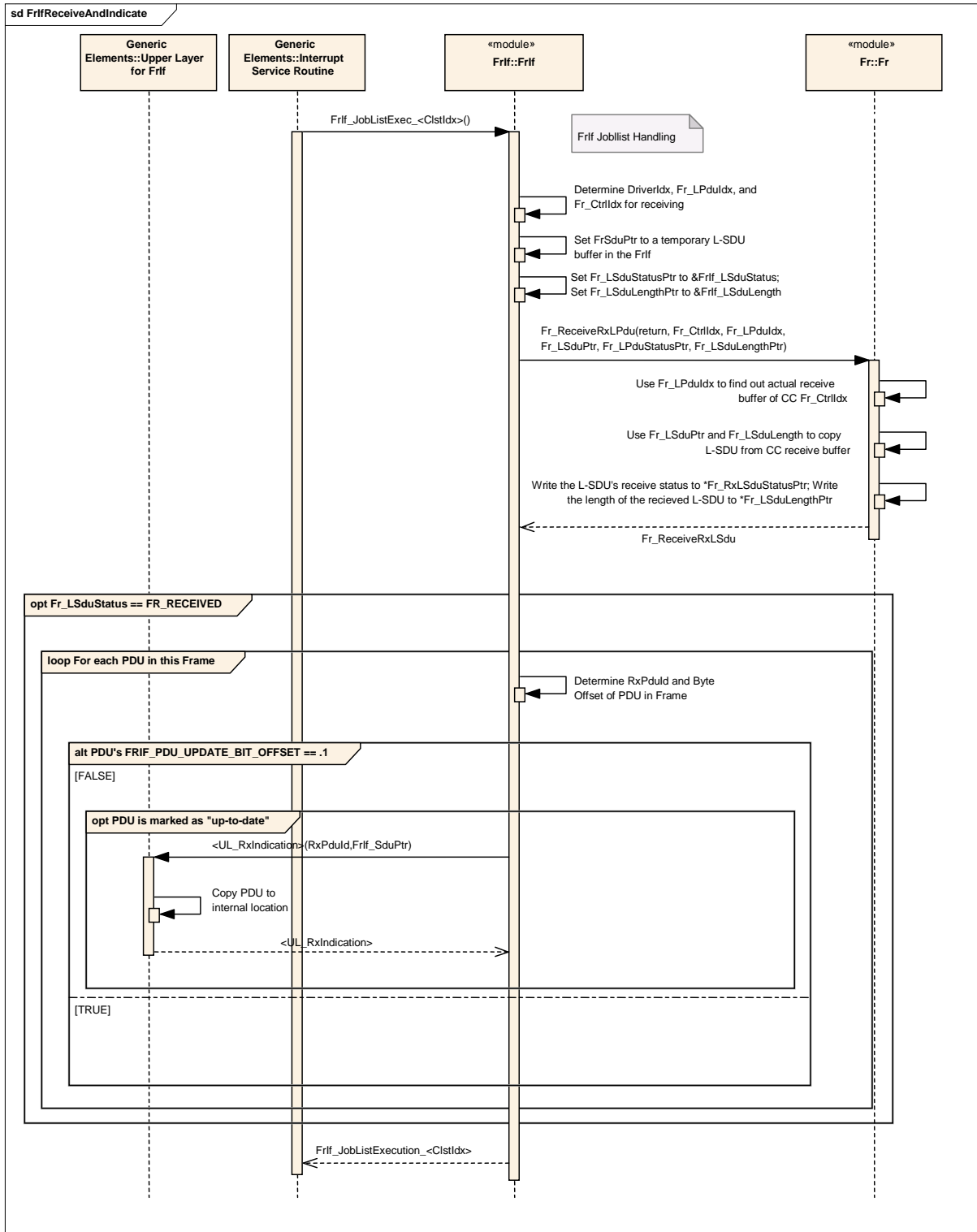


Figure 9-4: ReceiveAndIndicate

9.2.2 ReceiveAndStore

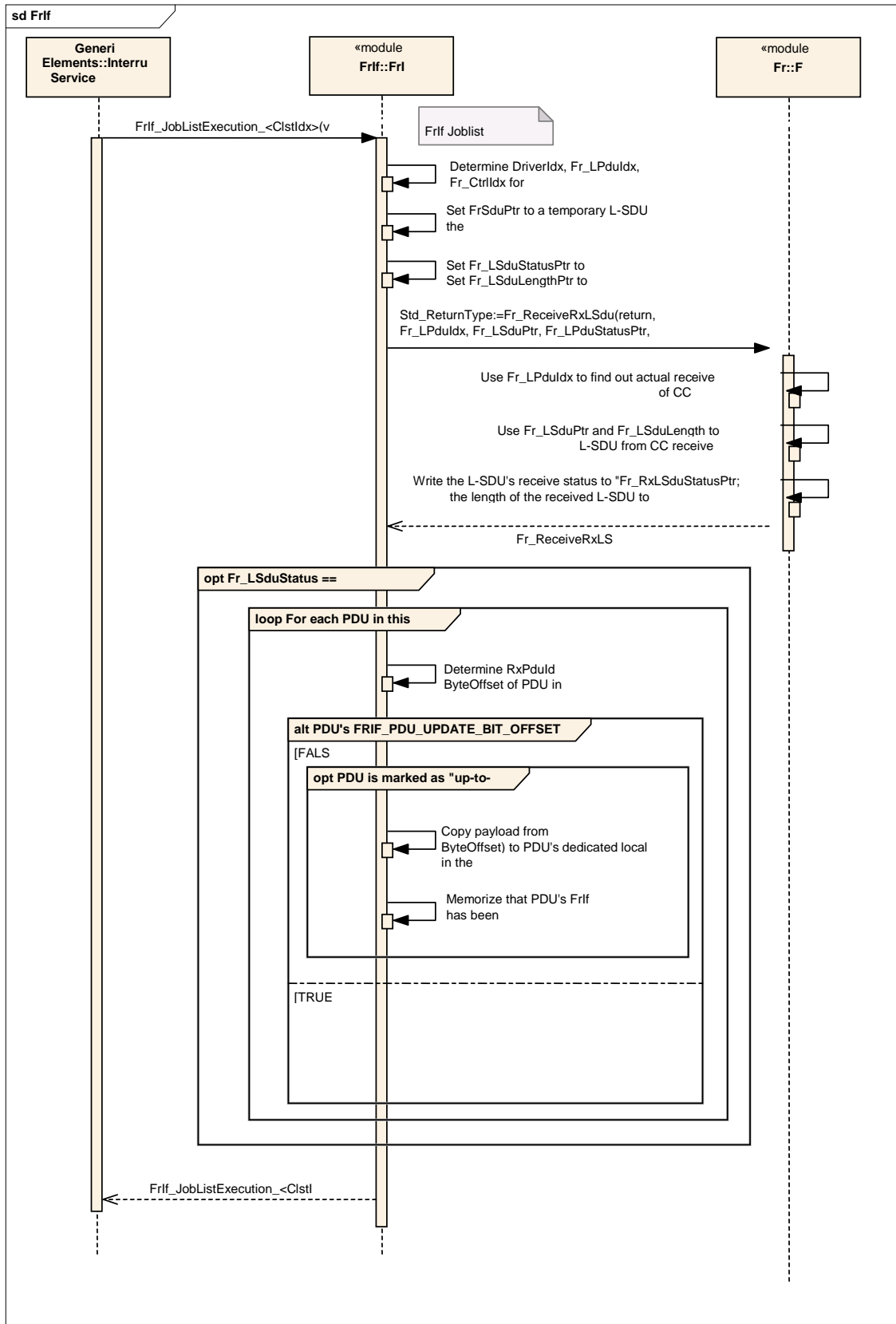


Figure 9-5: ReceiveAndStore

9.2.3 ProvideRxIndication

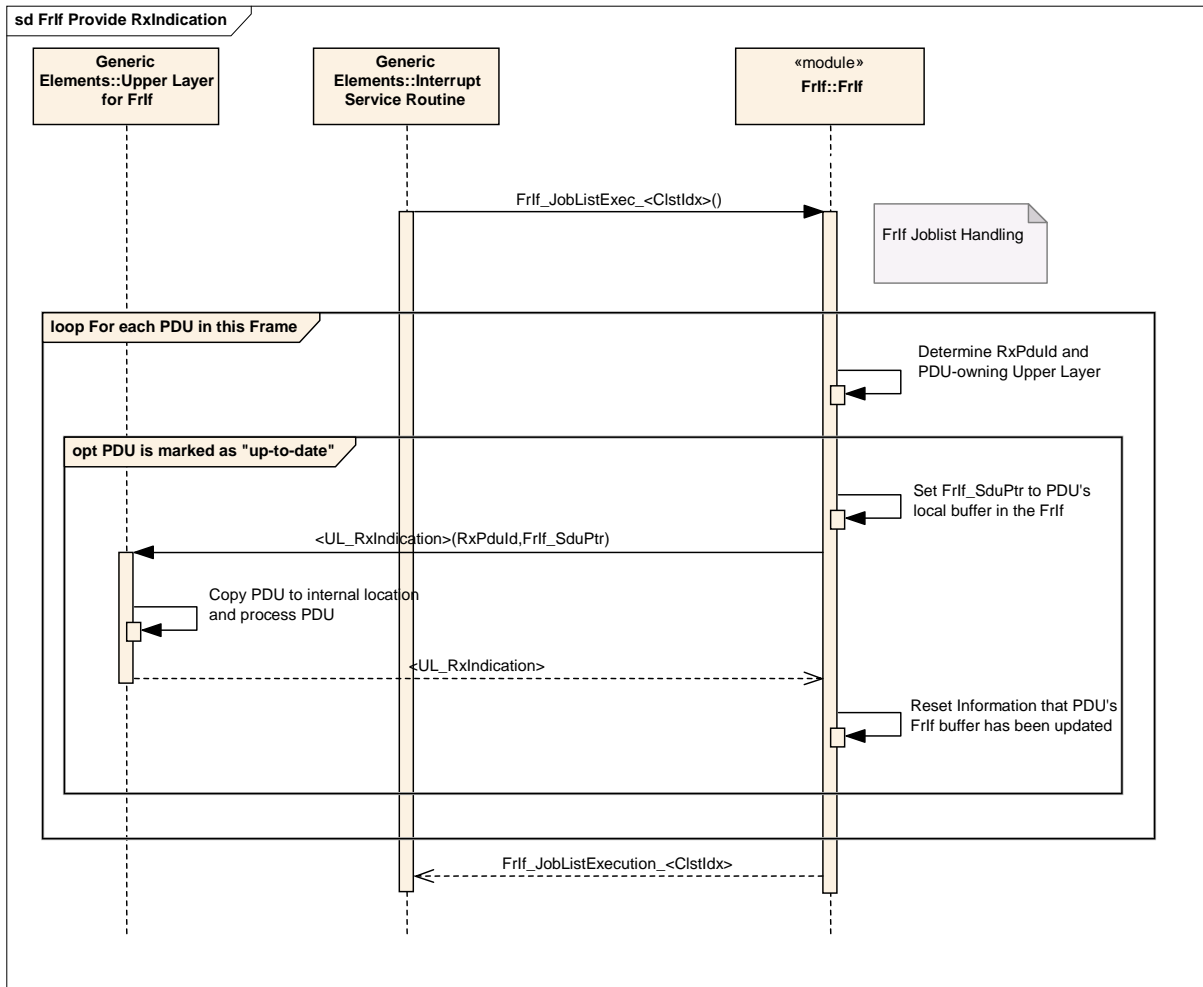


Figure 9-6: ProvideRxIndication

9.2.4 Cancel Transmission

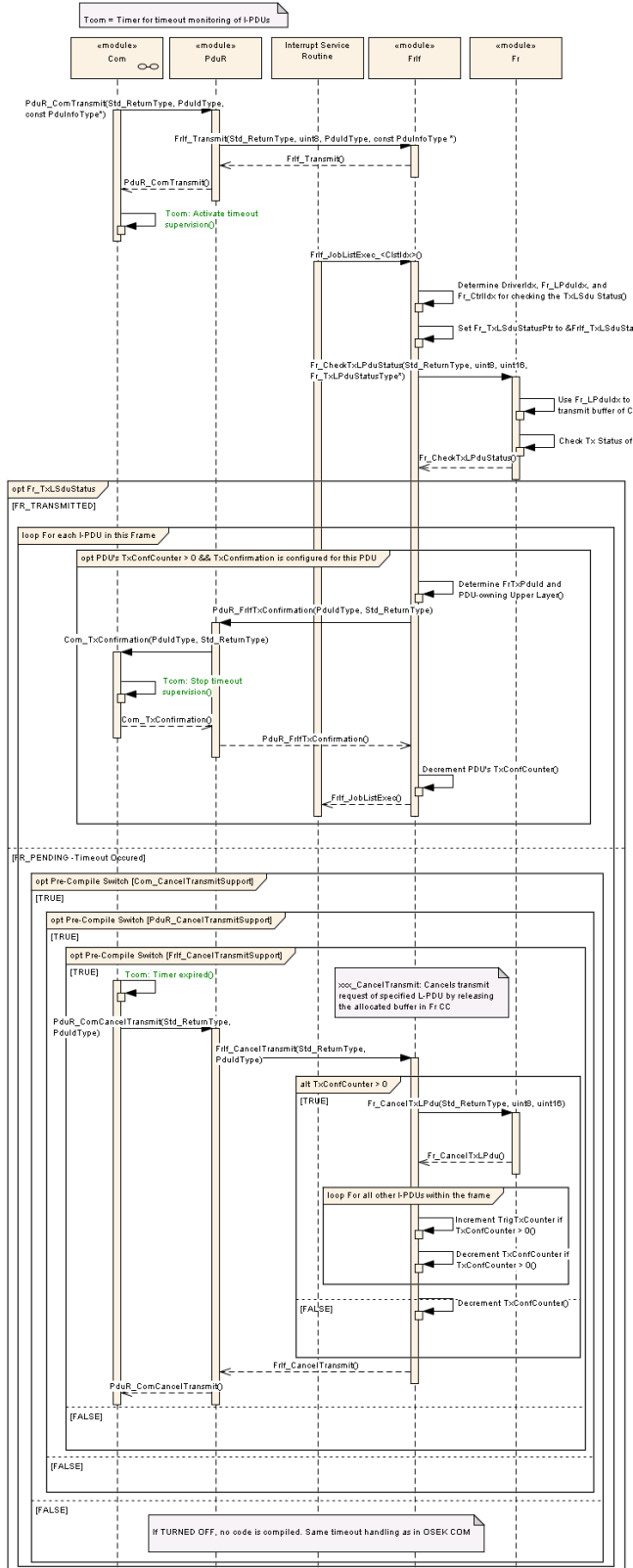


Figure 9-7: Cancel Transmission

9.3 Prepare LPDU

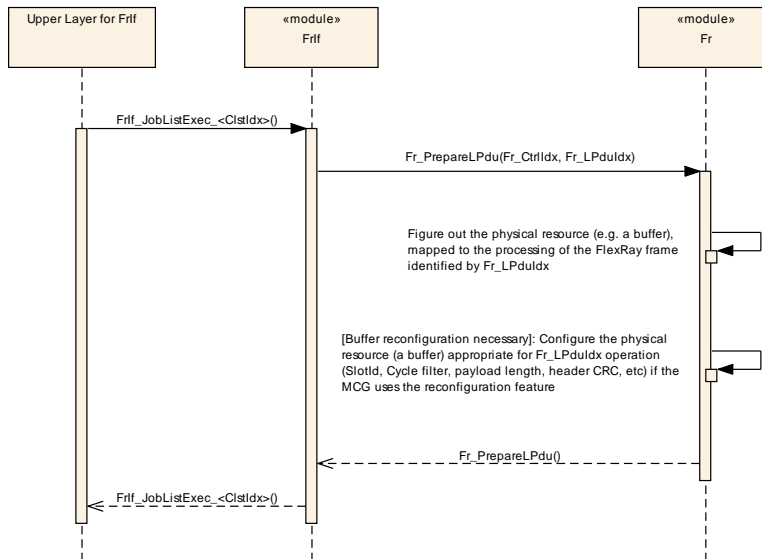


Figure 9-8: Prepare LPdu

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

SWS Item	ECUC_FrIf_06087 :
Module Name	<i>FrIf</i>
Module Description	Configuration of the FrIf (FlexRay Interface) module.
Post-Build Variant Support	true
Supported Config Variants	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.

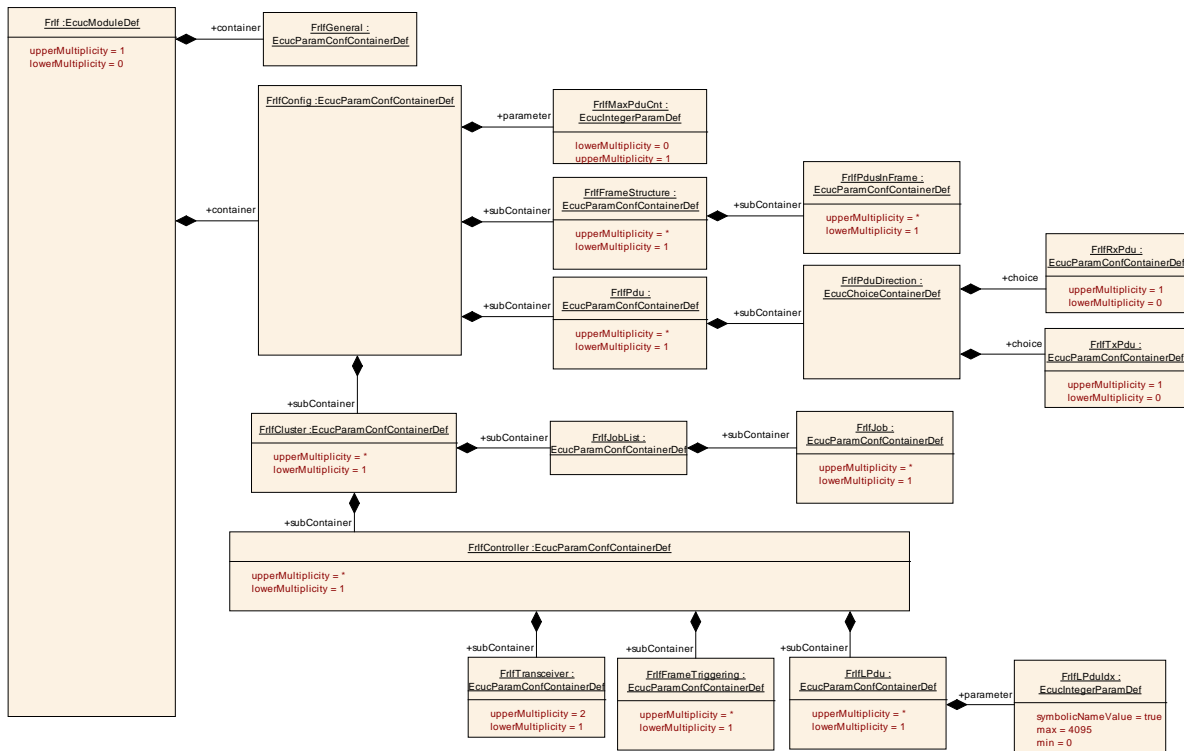


Figure 10-1: FlexRay Interface Module

10.2.2 FrlfGeneral

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

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

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

SWS Item	ECUC_Frlf_00002 :		
Name	FrlfCancelTransmitSupport		
Parent Container	FrlfGeneral		
Description	Configuration parameter to enable/disable Frlf support to request the cancellation of the I-PDU transmission to FrDrv.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06080 :		
Name	FrlfDevErrorDetect		
Parent Container	FrlfGeneral		

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

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

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

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

SWS Item	ECUC_Frlf_06118 :		
Name	FrlfFreeOpAApiName		
Parent Container	FrlfGeneral		
Description	API name that is called when FREE_OP_A is selected as communication operation. See also chapter 8.8.3 Configurable Interfaces.		
Multiplicity	0..1		
Type	EcucStringParamDef		
Default value	--		
maxLength	--		
minLength	--		
regularExpression	--		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06119 :		
Name	FrlfFreeOpBApiName		
Parent Container	FrlfGeneral		
Description	API name that is called when FREE_OP_B is selected as communication operation. See also chapter 8.8.3 Configurable Interfaces.		
Multiplicity	0..1		
Type	EcucStringParamDef		
Default value	--		
maxLength	--		
minLength	--		
regularExpression	--		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06120 :		
Name	FrlfFreeOpsHeader		
Parent Container	FrlfGeneral		
Description	Defines header file for configurable FREE_OP_A / FREE_OP_B functions.		
Multiplicity	0..1		
Type	EcucStringParamDef		
Default value	--		
maxLength	--		
minLength	--		
regularExpression	--		
Post-Build Variant Multiplicity	false		

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

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

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

SWS Item	ECUC_Frlf_06114 :		
Name	FrlfGetNmVectorSupport		
Parent Container	FrlfGeneral		
Description	Configuration parameter to enable/disable Frlf support to request the FlexRay hardware NMVector.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06104 :		
Name	FrlfGetNumOfStartupFramesSupport		
Parent Container	FrlfGeneral		
Description	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.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

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

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

SWS Item	ECUC_Frlf_06111 :		
Name	FrlfGetWakeupRxStatusSupport		
Parent Container	FrlfGeneral		
Description	Configuration parameter to enable/disable Frlf support to get the wakeup received information from the FlexRay controller.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06081 :		
Name	FrlfNumClstSupported		
Parent Container	FrlfGeneral		

Description	Maximum number of FlexRay Clusters that the FlexRay Interface supports.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 15		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06082 :		
Name	FrlfNumCtrlSupported		
Parent Container	FrlfGeneral		
Description	Maximum number of FlexRay CCs that the FlexRay Interface supports		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 15		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06116 :		
Name	FrlfPublicCddHeaderFile		
Parent Container	FrlfGeneral		
Description	Defines header files for callback functions which shall be included in case of CDDs. Range of characters is 1.. 32.		
Multiplicity	0..*		
Type	EcucStringParamDef		
Default value	--		
maxLength	--		
minLength	--		
regularExpression	--		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06117 :		
Name	FrlfReadCCConfigApi		
Parent Container	FrlfGeneral		
Description	Configuration parameter to enable/disable the optional Frlf_ReadCCConfig API.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
Post-Build Variant Value	false		

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

SWS Item	ECUC_FrIf_06109 :		
Name	FrIfReconfigLPduSupport		
Parent Container	FrIfGeneral		
Description	Configuration parameter to enable/disable FrIf support to enable/disable the reconfiguration of a given LPdu according to the parameters (FrameId, Channel, CycleRepetition, CycleOffset, PayloadLength, HeaderCRC) at runtime.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	ECUC_FrIf_06123 :		
Name	FrIfTxConflictNotificationHeaderName		
Parent Container	FrIfGeneral		
Description	Configuration of the header file name that defines the UL_TxConflictNotification.		
Multiplicity	0..1		
Type	EcucStringParamDef		
Default value	--		
maxLength	--		
minLength	--		
regularExpression	--		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: FrIfTxConflictNotificationName		

SWS Item	ECUC_FrIf_06122 :		
Name	FrIfTxConflictNotificationName		
Parent Container	FrIfGeneral		
Description	Configuration of the API name that is called in case a TxConflict has been detected.		
Multiplicity	0..1		
Type	EcucStringParamDef		
Default value	--		
maxLength	--		
minLength	--		
regularExpression	--		
Post-Build Variant	false		

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

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

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

No Included Containers

10.2.3 FrlfCluster

SWS Item	ECUC_Frlf_05366 :		
Container Name	FrlfCluster		
Description	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.		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Link time	--	
	Post-build time	--	
Configuration Parameters			

SWS Item	ECUC_FrIf_06002 :		
Name	FrIfClstIdx		
Parent Container	FrIfCluster		
Description	This parameter provides a zero-based consecutive index of the FlexRay Clusters. Upper layer BSW modules and the FrIf itself use this index to identify a FlexRay Cluster.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 63		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	ECUC_FrIf_00003 :		
Name	FrIfDetectNITError		
Parent Container	FrIfCluster		
Description	Indicates whether NIT error status of each cluster shall be detected or not.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_FrIf_06006 :		
Name	FrIfGChannels		
Parent Container	FrIfCluster		
Description	The channels that are used by the cluster. Implementation Type: Fr_ChannelType		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	FR_CHANNEL_A	Cluster uses channel A	
	FR_CHANNEL_AB	Cluster uses channel A and B	
	FR_CHANNEL_B	Cluster uses channel B	
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope /	scope: local		

Dependency	
-------------------	--

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

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

SWS Item	ECUC_Frlf_06020 :		
Name	FrlfGdActionPointOffset		
Parent Container	FrlfCluster		
Description	Number of macroticks the action point is offset from the beginning of a static slot.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 63		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU		

SWS Item	ECUC_Frlf_06021 :		
Name	FrlfGdBit		
Parent Container	FrlfCluster		
Description	Nominal bit time in seconds		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	T100NS		--
	T200NS		--
	T400NS		--

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

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

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

SWS Item	ECUC_FrIf_06026 :		
Name	FrIfGdDynamicSlotIdlePhase		
Parent Container	FrIfCluster		
Description	Duration of the idle phase within a dynamic slot [Minislots].		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 2		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_FrIf_00012 :		
Name	FrIfGdIgnoreAfterTx		
Parent Container	FrIfCluster		
Description	Duration for which the bitstrobing is paused after transmission [gdBit].		

	Remark: Set to 0 for FlexRay Protocol 2.1 Rev. A compliance.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 15		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

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

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

SWS Item	ECUC_Frlf_06032 :		
Name	FrlfGdMiniSlotActionPointOffset		
Parent Container	FrlfCluster		
Description	Number of Macroticks the Minislot action point is offset from the beginning of a Minislot [Macroticks].		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 31		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06034 :		
-----------------	--------------------------	--	--

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

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

SWS Item	ECUC_Frlf_06036 :		
Name	FrlfGdStaticSlot		
Parent Container	FrlfCluster		
Description	Duration of a static slot [Macroticks]. Remark: Range 4-661 for FlexRay Protocol 2.1 Rev. A compliance.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	3 .. 664		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06037 :		
Name	FrlfGdSymbolWindow		
Parent Container	FrlfCluster		
Description	Duration of the symbol window [Macroticks]. Remark: Range 0-142 for FlexRay Protocol 2.1 Rev. A compliance.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 162		
Default value	--		
Post-Build Variant Value	true		

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

SWS Item	ECUC_Frlf_00011 :		
Name	FrlfGdSymbolWindowActionPointOffset		
Parent Container	FrlfCluster		
Description	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.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 63		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06038 :		
Name	FrlfGdTSSTransmitter		
Parent Container	FrlfCluster		
Description	Number of bits in the Transmission Start Sequence [gdBits]. Remark: Lower limit 3 for FlexRay Protocol 2.1 Rev. A compliance.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 15		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06039 :		
Name	FrlfGdWakeupRxIdle		
Parent Container	FrlfCluster		
Description	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.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	8 .. 59		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06040 :		
Name	FrlfGdWakeupRxLow		

Parent Container	FrlfCluster		
Description	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.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	8 .. 59		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06041 :		
Name	FrlfGdWakeupRxWindow		
Parent Container	FrlfCluster		
Description	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.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	76 .. 485		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06043 :		
Name	FrlfGdWakeupTxActive		
Parent Container	FrlfCluster		
Description	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.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	15 .. 60		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06042 :		
Name	FrlfGdWakeupTxIdle		
Parent Container	FrlfCluster		
Description	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 gdWakeupSymbolTxDle.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	45 .. 180		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06009 :		
Name	FrlfGListenNoise		
Parent Container	FrlfCluster		
Description	Upper limit for the start up listen timeout and wake up listen timeout in the presence of noise. It is used as a multiplier of the node parameter pdListenTimeout.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	2 .. 16		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06010 :		
Name	FrlfGMacroPerCycle		
Parent Container	FrlfCluster		
Description	Number of macroticks in a communication cycle. Note: Lower limit 10 for FlexRay Protocol 2.1 Rev. A compliance		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	8 .. 16000		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06011 :		
Name	FrlfGMaxWithoutClockCorrectFatal		
Parent Container	FrlfCluster		
Description	Threshold used for testing the vClockCorrectionFailed counter. Defines the number of consecutive even/odd Cycle pairs with missing clock correction terms that will cause the protocol to transition from the POC:normal active or POC:normal passive state into the POC:halt state. [Even/odd cycle pairs].		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 15		
Default value	--		
Post-Build Variant Value	true		

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

SWS Item	ECUC_FrIf_06012 :		
Name	FrIfGMaxWithoutClockCorrectPassive		
Parent Container	FrIfCluster		
Description	Threshold used for testing the vClockCorrectionFailed counter. Defines the number of consecutive even/odd Cycle pairs with missing clock correction terms that will cause the protocol to transition from the POC:normal active state to the POC:normal passive state. [Even/Odd cycle pairs]		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 15		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_FrIf_06013 :		
Name	FrIfGNetworkManagementVectorLength		
Parent Container	FrIfCluster		
Description	Length of the Network Management vector in a cluster [bytes]		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 12		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_FrIf_06014 :		
Name	FrIfGNumberOfMinislots		
Parent Container	FrIfCluster		
Description	Number of minislots in the dynamic segment Remark: Upper limit 7986 for FlexRay Protocol 2.1 Rev. A compliance		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 7988		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_FrIf_06015 :		
Name	FrIfGNumberOfStaticSlots		
Parent Container	FrIfCluster		
Description	Number of static slots in the static segment		
Multiplicity	1		

Type	EcucIntegerParamDef		
Range	2 .. 1023		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

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

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

SWS Item	ECUC_Frlf_06003 :		
Name	FrlfMainFunctionPeriod		
Parent Container	FrlfCluster		
Description	The execution cycle of the Frlf_MainFunction_<cluster>() 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.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range]0 .. INF[
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	ECUC_FrIf_00004 :		
Name	FrlfSafetyMargin		
Parent Container	FrlfCluster		
Description	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.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 1024000		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

Included Containers		
Container Name	Multiplicity	Scope / Dependency
FrlfClusterDemEventParameterRefs	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_<ClstIdx>().

10.2.4 FrlfController

SWS Item	ECUC_FrIf_05363 :		
Container Name	FrlfController		
Description	This container contains the configuration of FlexRay CC.		
Post-Build Variant	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Link time	--	
	Post-build time	--	
Configuration Parameters			

SWS Item	ECUC_FrIf_06045 :		
Name	FrlfCtrlIdx		
Parent Container	FrlfController		
Description	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.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 31		

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

SWS Item	ECUC_Frlf_06044 :		
Name	FrlfFrCtrlRef		
Parent Container	FrlfController		
Description	Reference to a Controller, which is handled by a specific Driver. This reference is unique for the ECU.		
Multiplicity	1		
Type	Symbolic name reference to [FrController]		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU		

Included Containers		
Container Name	Multiplicity	Scope / Dependency
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.

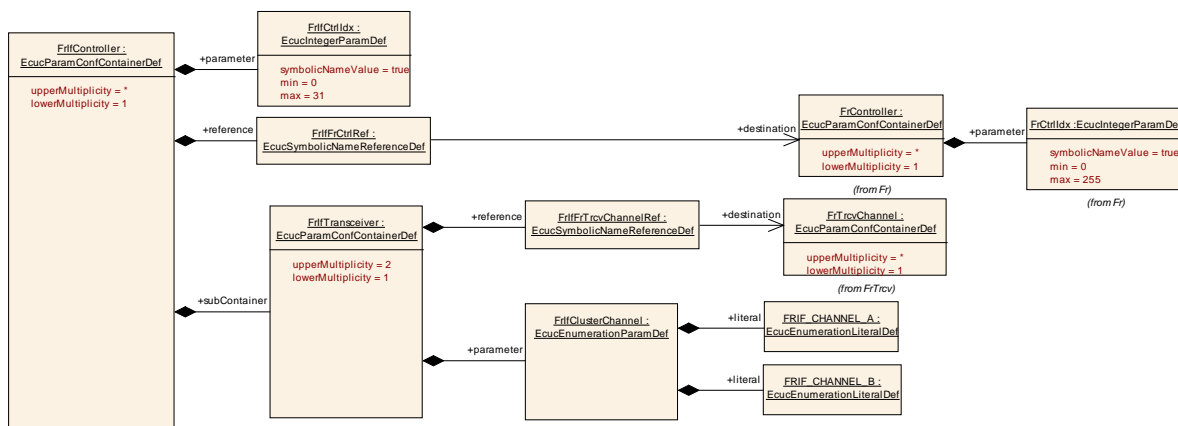


Figure 10-2: FlexRay Interface Controller (hardware reference)

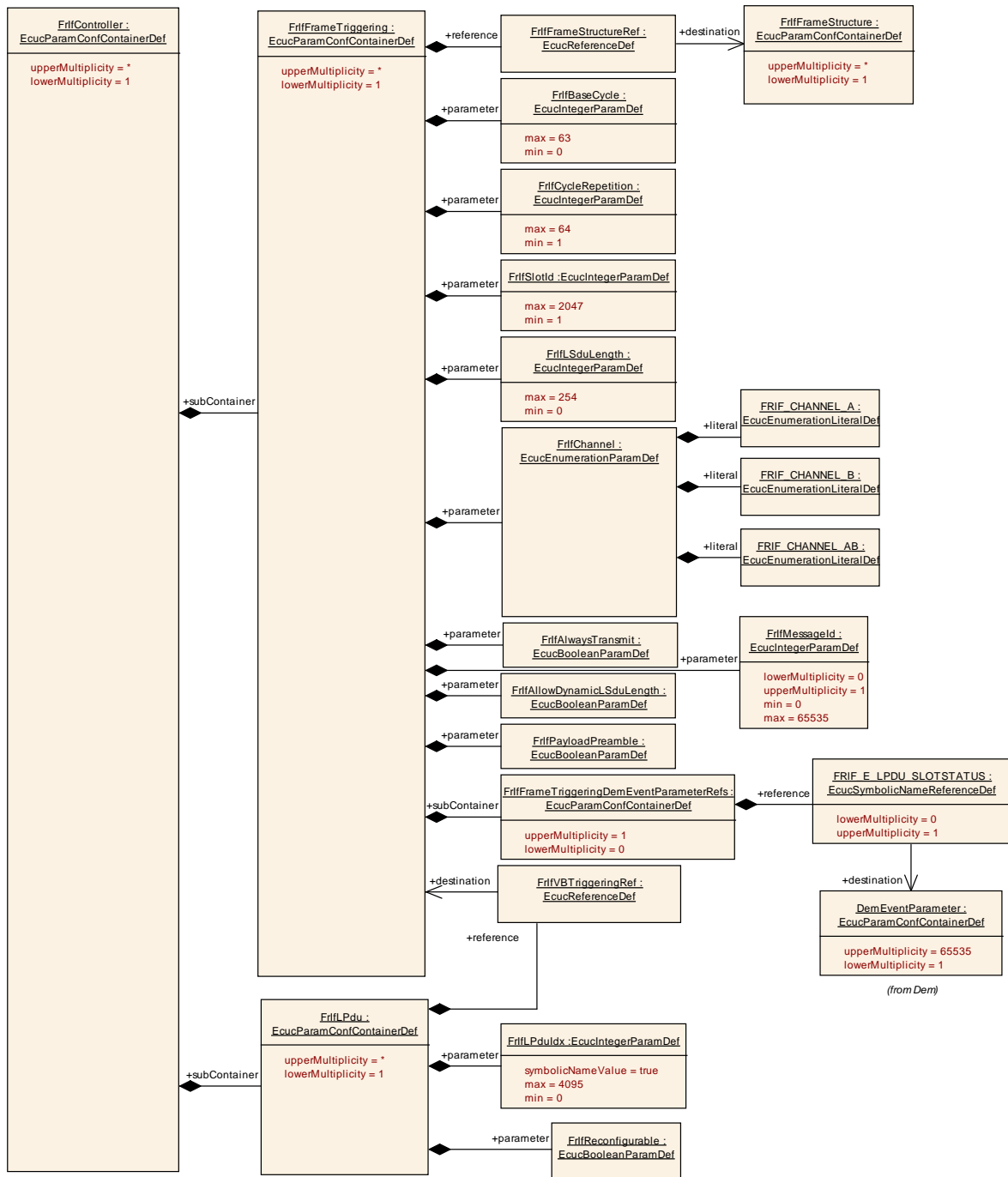


Figure 10-3: FlexRay Interface Controller (data reference)

10.2.5 FrIfTransceiver

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

SWS Item	ECUC_Frlf_06062 :		
Name	FrlfClusterChannel		
Parent Container	FrlfTransceiver		
Description	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.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	FRIF_CHANNEL_A	Channel A	
	FRIF_CHANNEL_B	Channel B	
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

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

No Included Containers

10.2.6 FrlfLPdu

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

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

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

SWS Item	ECUC_Frlf_00008 :		
Name	FrlfReconfigurable		
Parent Container	FrlfLPdu		
Description	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.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

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

No Included Containers

10.2.7 FrlfFrameTriggering

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

SWS Item	ECUC_Frlf_06049 :		
Name	FrlfAllowDynamicLSduLength		
Parent Container	FrlfFrameTriggering		
Description	Allows L-PDU length reduction ('FrlfLSduLength' 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.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_00013 :		
Name	FrlfAlwaysTransmit		
Parent Container	FrlfFrameTriggering		
Description	Defines whether the driver's API function Fr_TransmitTxLPdu() shall always be called for this L-PDU.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06051 :		
Name	FrlfBaseCycle		
Parent Container	FrlfFrameTriggering		
Description	This parameter contains the FlexRay Base Cycle used to transmit this FlexRay Frame.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 63		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06052 :		
Name	FrlfChannel		
Parent Container	FrlfFrameTriggering		
Description	This parameter contains the FlexRay Channel used to transmit this FlexRay Frame.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	FRIF_CHANNEL_A		Channel A
	FRIF_CHANNEL_AB		Channel A and B
	FRIF_CHANNEL_B		Channel B
Post-Build Variant Value	true		
Value	Pre-compile time	X	VARIANT-PRE-COMPILE

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

SWS Item	ECUC_Frlf_06053 :		
Name	FrlfCycleRepetition		
Parent Container	FrlfFrameTriggering		
Description	This parameter contains the FlexRay Cycle Repetition used to transmit this FlexRay Frame.. possible Values: 1,2,4,8,16,32,64		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 64		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06054 :		
Name	FrlfLsduLength		
Parent Container	FrlfFrameTriggering		
Description	The payload length of the Frame is given here. This parameter is required for validation if configured PDUs and update information fits into the Frame at configuration time [bytes].		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 254		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local dependency: The parameter depends on the low level parameters of the FlexRay CC.		

SWS Item	ECUC_Frlf_00010 :		
Name	FrlfMessageld		
Parent Container	FrlfFrameTriggering		
Description	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.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	--		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD

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

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

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

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

Included Containers		
Container Name	Multiplicity	Scope / Dependency
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

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

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

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

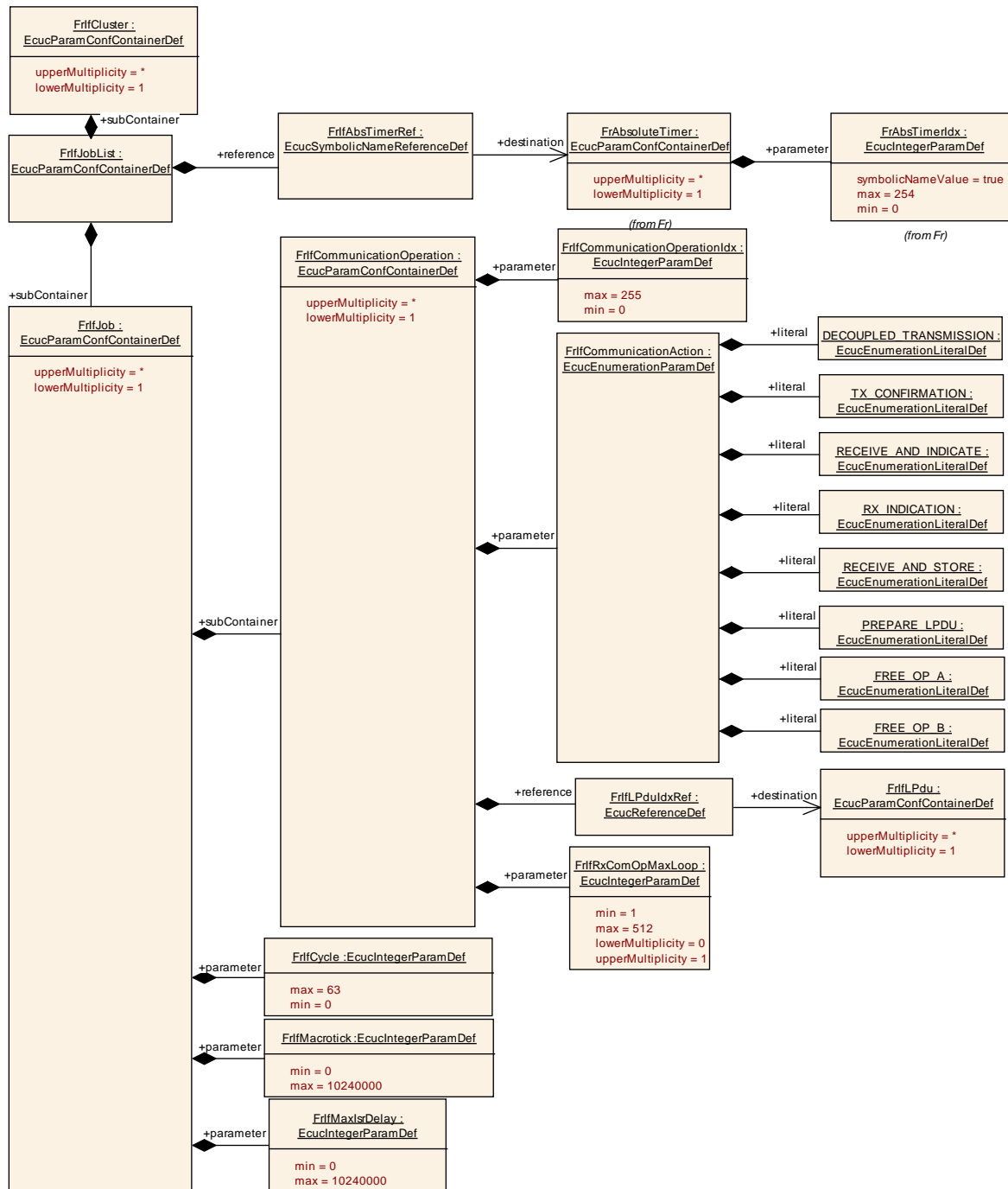


Figure 10-4: FlexRay Interface JobList

10.2.9 FrIfJob

SWS Item	ECUC_FrIf_05368 :		
Container Name	FrIfJob		
Description	A job may contain more than one operation that are executed at a specific point in time.		
Post-Build Variant Multiplicity	true		
Multiplicity Configuration	Pre-compile time	X	VARIANT-PRE-COMPILE

Class	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Configuration Parameters			

SWS Item	ECUC_Frlf_06064 :		
Name	FrlfCycle		
Parent Container	FrlfJob		
Description	The FlexRay Cycle in which the communication operation will execute this job		
Multiplicity	1		
Type	EcuIntegerParamDef		
Range	0 .. 63		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06065 :		
Name	FrlfMacroTick		
Parent Container	FrlfJob		
Description	MacroTick offset in the Cycle [MacroTick]		
Multiplicity	1		
Type	EcuIntegerParamDef		
Range	0 .. 10240000		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06004 :		
Name	FrlfMaxIsrDelay		
Parent Container	FrlfJob		
Description	The maximum delay in macroticks the Frlf_JoblistExec_<cluster>() function is processed after the absolute timer interrupt was triggered.		
Multiplicity	1		
Type	EcuIntegerParamDef		
Range	0 .. 10240000		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

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

10.2.10 FrlfCommunicationOperation

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

SWS Item	ECUC_Frlf_06067 :		
Name	FrlfCommunicationAction		
Parent Container	FrlfCommunicationOperation		
Description	The action to be performed in the FlexRay Operation		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	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
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local dependency: FrlfCommunicationAction can be configured as PREPARE_LPDU only if FrPrepareLPduSupport (ECUC_Fr_00453) is configured as TRUE.		

SWS Item	ECUC_Frlf_06068 :		
Name	FrlfCommunicationOperationIdx		
Parent Container	FrlfCommunicationOperation		
Description	For each FlexRay Communication Job, this index spans a range of zero-based consecutive values and thus defines the order of the FlexRay Communication Operation in the respective FlexRay Communication Job.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_00007 :		
-----------------	--------------------------	--	--

Name	FrlfRxComOpMaxLoop		
Parent Container	FrlfCommunicationOperation		
Description	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.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 512		
Default value	--		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

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

No Included Containers

10.2.11 FrlfFrameStructure

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

SWS Item	ECUC_Frlf_06113 :		
Name	FrlfByteOrder		
Parent Container	FrlfFrameStructure		

Description	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 FrIfPduOffset indicates the position of the most significant bit in the Frame. If LITTLE_ENDIAN is specified, the FrIfPduOffset indicates the position of the least significant bit in the Frame.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	BIG_ENDIAN	--	
	LITTLE_ENDIAN	--	
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

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

10.2.12 FrIfPduInFrame

SWS Item	ECUC_FrIf_05371 :		
Container Name	FrIfPduInFrame		
Description	This container holds all the information about a PDU in a FlexRay Frame.		
Post-Build Variant Multiplicity	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Configuration Parameters			

SWS Item	ECUC_FrIf_06070 :		
Name	FrIfPduOffset		
Parent Container	FrIfPduInFrame		
Description	The value specifies the offset of the PDU within the Frame [bytes].		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 253		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	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.
--	--------------------------

SWS Item	ECUC_Frlf_06071 :		
Name	FrlfPduUpdateBitOffset		
Parent Container	FrlfPduInFrame		
Description	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).		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 2031		
Default value	--		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	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.		

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

No Included Containers

10.2.13 FrlfPdu

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

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

10.2.14 FrlfTxPdu

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

SWS Item	ECUC_Frlf_06075 :		
Name	FrlfConfirm		
Parent Container	FrlfTxPdu		
Description	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.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local dependency: FrlfUserTxUL		

SWS Item	ECUC_Frlf_06076 :		
Name	FrlfCounterLimit		
Parent Container	FrlfTxPdu		
Description	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.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

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

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

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

SWS Item	ECUC_FrIf_00014 :		
Name	FrIfTxConfirmationName		
Parent Container	FrIfTxPdu		
Description	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.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	--		
maxLength	--		
minLength	--		
regularExpression	--		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU		

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

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

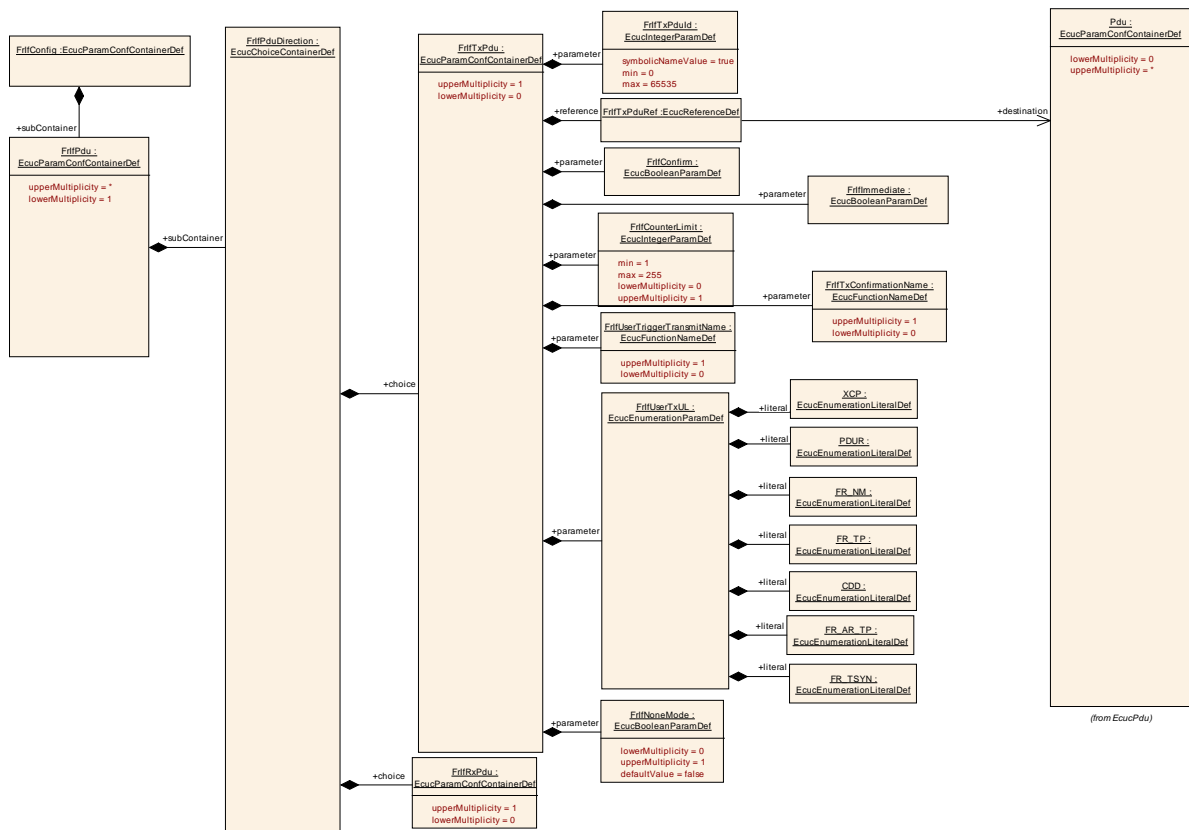
SWS Item	ECUC_Frlf_06084 :		
Name	FrlfUserTriggerTransmitName		
Parent Container	FrlfTxPdu		
Description	This parameter defines the name of the <User_TriggerTransmit>. This parameter depends on the parameter FrlfUserTxUL. If FrlfUserTxUL equals FR_TP, FR_AR_TP, FR_NM, PDUR, FR_TSYN or XCP the name of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

SWS Item	ECUC_Frlf_00015 :		
Name	FrlfUserTxUL		
Parent Container	FrlfTxPdu		
Description	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.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	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	
Post-Build Variant Value	true		
Value	Pre-compile time	X	VARIANT-PRE-COMPILE

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

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

No Included Containers



10.2.15 FrIfRxPdu

SWS Item	ECUC_FrIf_05373 :		
Container Name	FrIfRxPdu		
Description	Receive PDU		
Configuration Parameters			

SWS Item	ECUC_Frlf_00016 :		
Name	FrlfRxIndicationName		
Parent Container	FrlfRxPdu		
Description	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.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	--		
maxLength	--		
minLength	--		
regularExpression	--		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU		

SWS Item	ECUC_Frlf_00017 :		
Name	FrlfUserRxIndicationUL		
Parent Container	FrlfRxPdu		
Description	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.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	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	
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU		

SWS Item	ECUC_Frlf_06073 :		
Name	FrlfRxPduRef		
Parent Container	FrlfRxPdu		

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

No Included Containers

10.2.16 FrlfPduDirection

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

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

10.2.17 FrlfConfig

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

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

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

Included Containers		
Container Name	Multiplicity	Scope / Dependency
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

SWS Item	ECUC_Frlf_06091 :
Container Name	FrlfClusterDemEventParameterRefs
Description	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.
Configuration Parameters	

SWS Item	ECUC_Frlf_06097 :		
Name	FRIF_E_ACS_CH_A		
Parent Container	FrlfClusterDemEventParameterRefs		
Description	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.		
Multiplicity	0..1		
Type	Symbolic name reference to [DemEventParameter]		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06098 :		
Name	FRIF_E_ACS_CH_B		

Parent Container	FrlfClusterDemEventParameterRefs		
Description	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.		
Multiplicity	0..1		
Type	Symbolic name reference to [DemEventParameter]		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06093 :		
Name	FRIF_E_NIT_CH_A		
Parent Container	FrlfClusterDemEventParameterRefs		
Description	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.		
Multiplicity	0..1		
Type	Symbolic name reference to [DemEventParameter]		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06094 :		
Name	FRIF_E_NIT_CH_B		
Parent Container	FrlfClusterDemEventParameterRefs		
Description	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.		
Multiplicity	0..1		
Type	Symbolic name reference to [DemEventParameter]		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06095 :		
-----------------	--------------------------	--	--

Name	FRIF_E_SW_CH_A		
Parent Container	FrlfClusterDemEventParameterRefs		
Description	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.		
Multiplicity	0..1		
Type	Symbolic name reference to [DemEventParameter]		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_Frlf_06096 :		
Name	FRIF_E_SW_CH_B		
Parent Container	FrlfClusterDemEventParameterRefs		
Description	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.		
Multiplicity	0..1		
Type	Symbolic name reference to [DemEventParameter]		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

No Included Containers

10.2.19 FrlfFrameTriggeringDemEventParameterRefs

SWS Item	ECUC_Frlf_06099 :
Container Name	FrlfFrameTriggeringDemEventParameterRefs
Description	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.
Configuration Parameters	

SWS Item	ECUC_Frlf_00009 :
Name	FRIF_E_LPDU_SLOTSTATUS

Parent Container	FrlfFrameTriggeringDemEventParameterRefs		
Description	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.		
Multiplicity	0..1		
Type	Symbolic name reference to [DemEventParameter]		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	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)]