Document Title	Specification of FlexRay Network Management
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	28

Document Status	published
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	R24-11

Document Change History

Date	Release	Changed by	Description
2024-11-27	R24-11	AUTOSAR Release Management	• Changed lower layer to LSduR • Fixed include file for callback functions
2023-11-23	R23-11	AUTOSAR Release Management	• Removed all HandleId configuration parameters • Minor corrections
2022-11-24	R22-11	AUTOSAR Release Management	• Changes to Partial Networking • Traceability changes
2021-11-25	R21-11	AUTOSAR Release Management	• Note added under [SWS_FrNm_00492] • Changed occurrences of FRNM_PASSIVE_MODE_ENABLED to FrNmPassiveModeEnabled. • Changes to Synchronized PNC shutdown. • Uptrace from SRS_Nm changed to RS_Nm

▽

△

2020-11-30	R20-11	AUTOSAR Release Management	• Chapter 7.12 Error detection removed • Chapter 7.13 Error notification removed. • [SWS_FrNm_00056], [SWS_FrNm_00057], [SWS_FrNm_00051] moved to Chapter 8. • Specification for synchronized PNC shutdown added. • Configurable interfaces moved from Chapter 8.6.2 to 8.6.3 • PRS_NetworkManagementProtocol added as input document.
2019-11-28	R19-11	AUTOSAR Release Management	• No Content Changes • Changed Document Status from Final to published
2018-10-31	4.4.0	AUTOSAR Release Management	• Introduced Reliable TxConfirmation • Multiple function instance updated to use shortname instead of Ids • Removed CBV configuration • Added new Nm notification call for Synchronization • Header File Cleanup • Removed obsolete elements
2017-12-08	4.3.1	AUTOSAR Release Management	• Node Detetcion Configuration per channel • Minor corrections with respect to Default Error Tracer • FrNmActiveWakeupBitEnabled dependency updated

▽

△

2016-11-30	4.3.0	AUTOSAR Release Management	• API Harmonizations • Clarification on initiatlization of FrNm • Introduction of Reliable TX Confirmation • Update in TriggerTransmit • Minor corrections
2015-07-31	4.2.2	AUTOSAR Release Management	• Clarification on FrNmPassiveModeEnabled • Clarification on FrNmNumberOfClusters • Clarity on scheduling of MainFunction • Debugging support marked as obsolete • Minor corrections
2014-10-31	4.2.1	AUTOSAR Release Management	• Correction of Partial Networking aggregation algorithm • Harmonize description of identical API's • Const usage consistent in specifications
2014-03-31	4.1.3	AUTOSAR Release Management	• Corrections for Partial Networking • Correction in Initialization sequence • Modification in State Chart • Timing dependencies between parameters in FrNm were updated with more clarifications • Changes in Header file structure
2013-10-31	4.1.2	AUTOSAR Release Management	• Revised configuration parameter related to Partial Networking • Fix file inclusion in Chapter 5 • Fix Mandatory Interfaces • Revised Passive Startup requirements • Editorial changes • Removed chapter(s) on change documentation

▽

△

2013-03-15	4.1.1	AUTOSAR Administration	• Fix potential deadlock in repeat message state • Fixes in the NM coordinator algorithm • Clarification regarding FrNmReadySleepCnt and FrNmRepetitionCycle • Improve PostBuild support • Formal changes to improve traceability
2011-12-22	4.0.3	AUTOSAR Administration	• Support of a coordinated shutdown if more than one gateway coordinator is connected to the same network • Support of CarWakeup in NM user data • Extension for Partial Network
2010-09-30	3.1.5	AUTOSAR Administration	• Added [SWS_FrNm_00066], [SWS_FrNm_00220], [SWS_FrNm_00395], [SWS_FrNm_00387], [SWS_FrNm_00388], [SWS_FrNm_00389], [SWS_FrNm_00390], [SWS_FrNm_00391], [SWS_FrNm_00392] • Update [SWS_FrNm_00235], FRNM254 • Modified [SWS_FrNm_00074], [SWS_FrNm_00021], [SWS_FrNm_00272], [SWS_FrNm_00074], [SWS_FrNm_00135], [SWS_Nm_00192], [SWS_FrNm_00154], [SWS_FrNm_00155], [SWS_FrNm_00324], [SWS_FrNm_03829], [SWS_FrNm_00394], [SWS_FrNm_00181], [SWS_FrNm_00229], [SWS_FrNm_00066], ▽

▽

△

			△ [SWS_FrNm_00359], [SWS_FrNm_00106], [SWS_FrNm_00035], [SWS_FrNm_00257]
2010-02-02	3.1.4	AUTOSAR Administration	• Improved configurable handling of bus faults to support a smooth transition to sleep or a resynchronisation of the network to ensure its coherency. • Multiple variants of network coordination support dual channel and complex FlexRay networks • Relaxed timing constraints of the FrNm main function • Legal disclaimer revised
2009-02-04	3.1.2	AUTOSAR Administration	• Layout adaptations
2008-08-13	3.1.1	AUTOSAR Administration	• Incorporation of core partner change requests for R3.0 • Legal disclaimer revised
2007-12-21	3.0.1	AUTOSAR Administration	• Updated chapter 8, 10 after changes in BSW UML model
2007-07-24	2.1.16	AUTOSAR Administration	• FlexRay NM machine has been reworked completely • Support of Hardware NM Vector of communication controller • Separation of NM vote and data (transmission of NM vote and data can be done with different update intervals) • FlexRay NM State machine is now synchronised to FlexRay communication cycle • Document meta information extended • Small layout adaptations made • Added [SWS_FRNM_00311]

▽

△

2007-01-24	2.1.15	AUTOSAR Administration	• FlexRay NM machine has been reworked completely • Support of Hardware NM Vector of communication controller • Separation of NM vote and data (transmission of NM vote and data can be done with different update intervals) • FlexRay NM State machine is now synchronised to FlexRay communication cycle • Legal disclaimer revised • Release Notes added • “Advice for users” revised • “Revision Information” added
2005-05-31	1.0	AUTOSAR Administration	• Initial Release

Contents

1 Introduction and functional overview

This specification describes the functionality, API and the configuration for the AUTOSAR Basic Software module FlexRay Network Management (FrNm).

The AUTOSAR FlexRay Network Management is a hardware independent protocol that can only be used on FlexRay (for limitations see 4.1). Its main purpose is to coordinate the transition between normal operation and bus-sleep mode of the network.

In addition to the core functionality optional features are provided e.g. to implement a service to detect all present nodes or to detect if all other nodes are ready to sleep.

2 Acronyms and Abbreviations

The glossary below includes acronyms and abbreviations relevant to the FrNm module that are not included in the [1, AUTOSAR glossary].

Abbreviation / Acronym:	Description:
CC	Communication Controller
NM	Network Management
WCET	Worst Case Execution Time
DET	Default Error Tracer. AUTOSAR Module for detection and reporting of errors during development.
SRS	Software requirements specification
SWS	Software working specification
API	Application program interface
Com	Communication module
OS	Operating system
SchM	Schedule Manager
PDU	Protocol data unit
CPU	Central processing unit
PNL	Partial Network Learning
FrIf	Abbreviation for the FlexRay Interface
FrNm	Abbreviation for the Network Management on FlexRay
NmIf	Abbreviation for the generic Network Management
ECU	Electronic control unit
ComM	Communication manager
FrSm	FlexRay State Manager
HW	Hardware
LSduR	Module that transfers L-SDUs from one module to another module. The L-SDU Router module can be utilized for internal routing purposes.

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

Term:	Definition:
Bus-Sleep Mode	Network mode where all interconnected communication controllers are in the sleep mode.
NM-Network	Instance of the FlexRay NM to handle one physical FlexRay Bus. Caution: The FlexRay Bus contains two FlexRay channels which cannot be handled independent of the other. Therefore the NM-Network covers both FlexRay bus channels. This is equivalent to one NM-Cluster
Network requested	NM network is requested if FrNm_NetworkRequest has been called and neither the network has been released nor Bus Sleep Mode has been entered afterwards
Network released	NM network is released if FrNm_NetworkRelease has been called and network has not been requested afterwards
Repeat Message Request active	A Repeat Message Request is active if FrNm_RepeatMessageRequest has been called or a NM PDU with Repeat Message Request bit set has been received in Network Mode. It is not active anymore if Repeat Message State is left or Bus Sleep Mode is entered.
NM Data Cycle	Number of FlexRay cycles necessary for all nodes to be able to send NM Data at least once.
NM Message	Packet of information exchanged for purposes of the NM algorithm.

▽

△

NM Repetition Cycle	Number of FlexRay cycles where no change of voting behavior is possible. Value has to be a multiple of the NM Voting Cycle. Used to improve the reliability of the voting.
NM Slot	Slot reserved for purposes of network management.
NM Timeout	Timeout in the NM algorithm that initiates transition into Bus-Sleep Mode.
NM User Data	Supplementary application specific data that is sent independent of the NM Vote on the bus.
NM Voting Cycle	Number of FlexRay cycles necessary for all nodes to vote at least once.
NM-Vote	Information transmitted using the FlexRay Bus indicating the vote of a ECU to keep the bus awake
NM-Vector	FlexRay NM Vector is the aggregated data available when the FlexRay CC optional NM Hardware Vector Service is used.
NM-Data	Data related to NM transmitted using the FlexRay Bus.
NM-Cluster	Obsolete, equivalent to NM-Network
CBV	Control bit vector
ClusterAwakeVote	At least one Node other than itself votes for keeping the cluster awake.
Positive NM-Vote in static segment	NM-Vote PDU reception or transmission with Voting Bit set to '1'
Positive NM-Vote in dynamic segment	NM-Vote PDU received or transmitted
Negative NM-Vote in static segment	NM-Vote PDU reception or transmission with Voting Bit set '0'
Negative NM-Vote	In dynamic segment: NM-Vote PDU is neither received nor transmitted
Top-level PNC coordinator	An ECU acts as top-level PNC coordinator for those PNCs which are actively coordinated on all assigned channels. This ECU has the PNC gateway functionality enabled. The top-level PNC coordinator triggers for those PNCs a synchronized PNC shutdown, if no other ECU in the network requests them and if the synchronized PNC shutdown is enabled.
Intermediate PNC coordinator	An ECU acts as intermediate PNC coordinator for those PNCs which are passively coordinated on at least one channel. This ECU has the PNC gateway functionality enabled. The intermediate PNC coordinator forwards a synchronized PNC shutdown to active coordinated channels for PNCs which are passively coordinated, if the synchronized PNC shutdown is enabled.
PNC leaf node	A PNC leaf node is an ECU that acts not as a PNC coordinator at all in the network. It processes PN shutdown message as usual NM messages.
PN shutdown message	A top-level PNC coordinator transmit PN shutdown messages to indicate a synchronized PNC shutdown across the PN topology. A PN shutdown message is as NM message which has PNSR bit in the control bit vector and all PNCs which are indicated for a synchronized shutdown set to '1'.

Table 2.2: Definitions used in the scope of this Document

3 Related documentation

3.1 Input documents & related standards and norms

[1] Glossary
AUTOSAR_FO_TR_Glossary

[2] General Specification of Basic Software Modules
AUTOSAR_CP_SWS_BSWGeneral

[3] Requirements on AUTOSAR Network Management
AUTOSAR_FO_RS_NetworkManagement

[4] General Requirements on Basic Software Modules
AUTOSAR_CP_RS_BSWGeneral

[5] Specification of ECU State Manager
AUTOSAR_CP_SWS_ECUStateManager

[6] Specification of FlexRay State Manager
AUTOSAR_CP_SWS_FlexRayStateManager

[7] FlexRay Communications System Protocol Specification V2.1
http://www.flexray.com/

[8] System Template
AUTOSAR_CP_TPS_SystemTemplate

[9] Specification of FlexRay Interface
AUTOSAR_CP_SWS_FlexRayInterface

[10] Specification of Operating System
AUTOSAR_CP_SWS_OS

[11] Guide to Mode Management
AUTOSAR_CP_EXP_ModeManagementGuide

[12] Specification of Network Management Interface
AUTOSAR_CP_SWS_NetworkManagementInterface

3.2 Related specification

AUTOSAR provides a General Specification on Basic Software modules [2, SWS BSW General], which is also valid for FlexRay Network Management.

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

4 Constraints and assumptions

4.1 Limitations

1. FlexRay NM can be applied to FlexRay communication systems that support bus sleep mode and that are implemented with appropriate wakeup mechanisms.
2. One instance of FlexRay NM can be applied to only one instance of FlexRay Interface within the same ECU.
3. One instance of FlexRay NM can be applied to only one FlexRay NM-Cluster in one FlexRay network. One FlexRay NM-Cluster can have only one instance of FlexRay NM.
4. FlexRay NM can be applied to both FlexRay channels of the same FlexRay Bus at the same time.

Figure 4.1 presents an AUTOSAR NM stack within an example ECU belonging to a FlexRay NM-clusters.

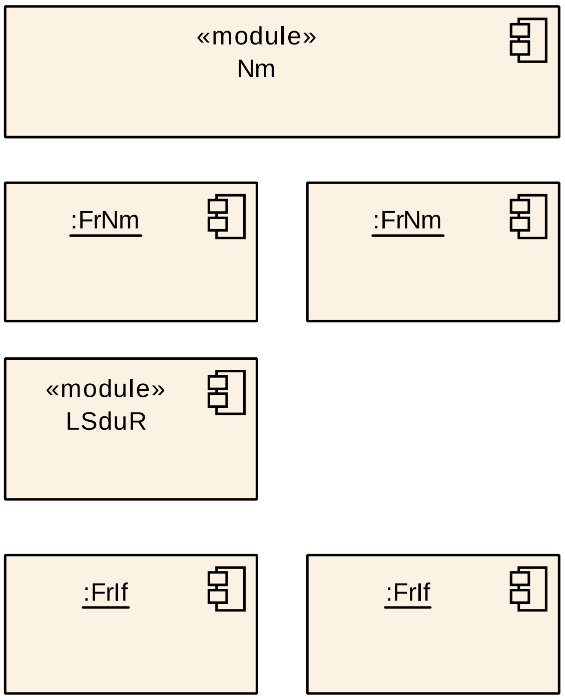

Figure 4.1: AUTOSAR NM Stack on FlexRay

4.2 Applicability to car domains

AUTOSAR NM can be applied to any car domain, wherever FlexRay technology is used, under limitations provided above.

5 Dependencies to other modules

FlexRay NM provides services to the Generic Network Management Interface (NmIf) and uses services of FlexRay Interface

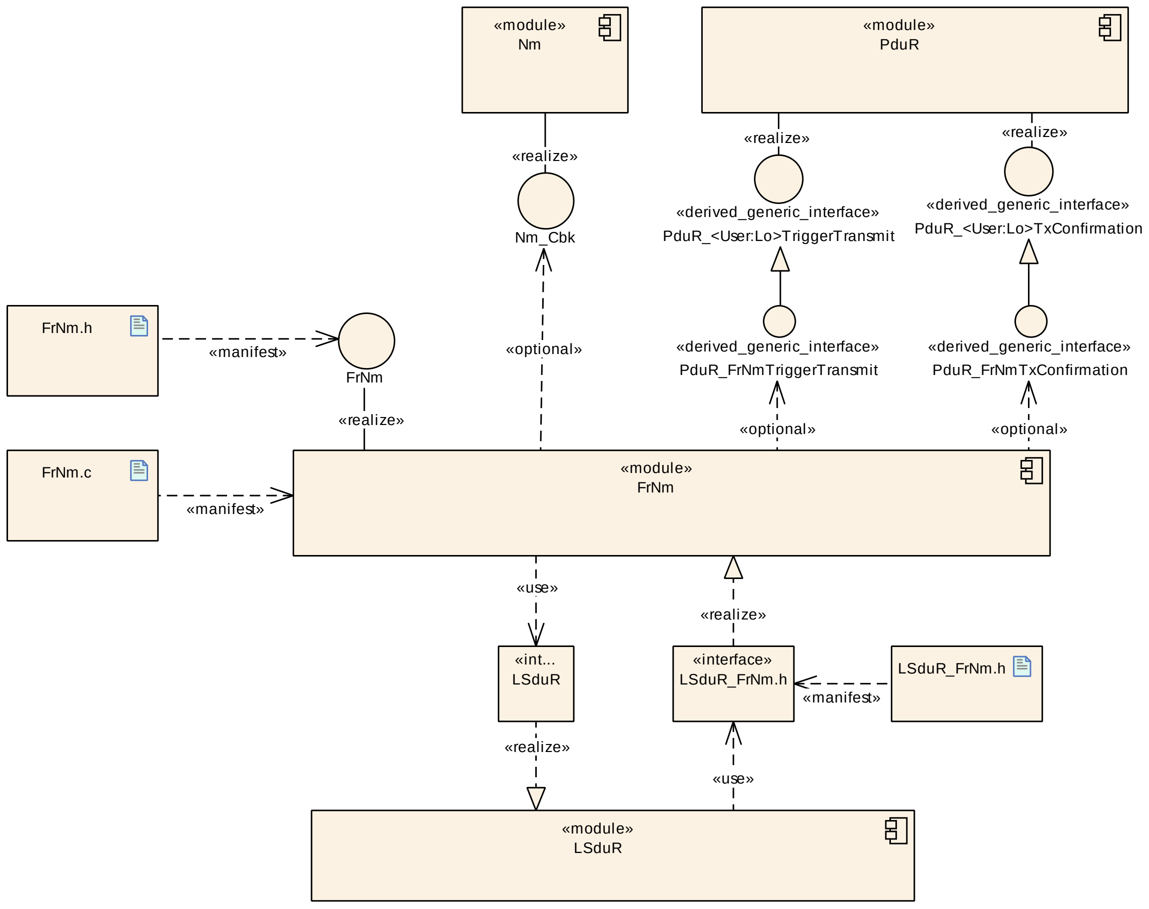

Figure 5.1: NM Overview

Note: In addition to the modules depicted in Figure 5.1, FlexRay Nm uses some additional modules (like the DET).

The FlexRay NM uses OS objects and/or related OS services according to [SWS_FrNm_00220].

[SWS_FrNm_00220]

Upstream requirements: SRS_BSW_00429

⌈

Module:	Dependencies:
Nm	FlexRay NM provides services to the Generic Network Management Interface (NmIf)
PduR	FlexRay NM uses services of the PDU Router
LSduR	FlexRay NM uses services of the L-SDU Router.
FrIf	FlexRay NM uses services of the FlexRay Interface.
Det	The FlexRay Network Management informs the Default Error Tracer on development errors
EcuM	EcuM gets wake up event information from FlexRay Network Management if supported by hardware.
RTE	The FlexRay Network Management main function may be scheduled by the by the RTE.

⌋

5.1 File structure

5.1.1 Code file structure

For details refer to the chapter 5.1.6 "Code file structure" in SWS_BSWGeneral.

6 Requirements Tracing

The following tables reference the requirements specified in [3] and [4] and links to the fulfillment of these. Please note that if column “Satisfied by” is empty for a specific requirement this means that this requirement is not fulfilled by this document.

Requirement	Description	Satisfied by
[RS_Nm_00045]	Nm shall provide services to coordinate shutdown of Nm-clusters independently of each other	[SWS_FrNm_00034]
[RS_Nm_00046]	It shall be possible to trigger the startup of all Nodes at any Point in Time	[SWS_FrNm_00168]
[RS_Nm_00048]	Nm shall put the communication controller into sleep mode if there is no bus communication	[SWS_FrNm_00101]
[RS_Nm_00050]	The Nm shall provide the current state of Nm	[SWS_FrNm_00104]
[RS_Nm_00051]	Nm shall inform application when Nm state changes occur.	[SWS_FrNm_00106]
[RS_Nm_00052]	The Nm interface shall signal to the application that all other ECUs are ready to sleep.	[SWS_FrNm_00181]
[RS_Nm_00054]	There shall be a deterministic time from the point where all nodes agree to go to bus sleep to the point where bus is switched off.	[SWS_FrNm_00101]
[RS_Nm_00137]	Nm shall perform communication system error handling for errors that have impact on the Nm behavior.	[SWS_FrNm_00035]
[RS_Nm_00144]	Nm shall support communication clusters of up to 64 ECUs	[SWS_FrNm_00179]
[RS_Nm_00149]	The timing of Nm shall be configurable.	[SWS_FrNm_00036]
[RS_Nm_00150]	Specific features of the Network Management shall be configurable	[SWS_FrNm_00077] [SWS_FrNm_00179] [SWS_FrNm_00180] [SWS_FrNm_00187]
[RS_Nm_00153]	The Network Management shall optionally provide a possibility to detect present nodes	[SWS_FrNm_00491] [SWS_FrNm_00493] [SWS_FrNm_00494]
[RS_Nm_00154]	The Network Management API shall be independent from the communication bus	[SWS_FrNm_00034]
[RS_Nm_02503]	The Nm API shall optionally give the possibility to send user data	[SWS_FrNm_00043]
[RS_Nm_02504]	The Nm API shall optionally give the possibility to get user data	[SWS_FrNm_00044]
[RS_Nm_02505]	The Nm shall optionally set the local node identifier to the Nm-message	[SWS_FrNm_00222]
[RS_Nm_02506]	The Nm API shall give the possibility to read the source node identifier of the sender	[SWS_FrNm_00047]
[RS_Nm_02508]	Every node shall have a node identifier associated with it that is unique in the Nm-cluster.	[SWS_FrNm_00037]

▽

△

Requirement	Description	Satisfied by
[RS_Nm_02509]	The Nm interface shall signal to the application that at least one ECU is not ready to sleep anymore.	[SWS_FrNm_00185]
[RS_Nm_02511]	It shall be possible to configure the Network Management of a node so that it does not contribute to the cluster shutdown decision.	[SWS_FrNm_00187]
[RS_Nm_02512]	The Nm shall give the possibility to enable or disable the network management related communication configured for an active Nm node	[SWS_FrNm_00387] [SWS_FrNm_00390]
[RS_Nm_02517]	CanNm shall support Partial Networking on CAN	[SWS_FrNm_00055] [SWS_FrNm_00404] [SWS_FrNm_00405] [SWS_FrNm_00407] [SWS_FrNm_00409] [SWS_FrNm_00452] [SWS_FrNm_00537]
[RS_Nm_02519]	The Nm Control Bit Vector shall contain a PNI (Partial Network Information) bit.	[SWS_FrNm_00565]
[RS_Nm_02527]	Nm shall implement a filter algorithm dropping all Nm messages that are not relevant for the ECU	[SWS_FrNm_00404]
[RS_Nm_02540]	The Nm Control Bit Vector shall contain a PN shutdown request bit.	[SWS_FrNm_00055]
[RS_Nm_02544]	Nm shall forward the indication of a PN shutdown message	[SWS_FrNm_00504] [SWS_FrNm_00540]
[RS_Nm_02548]	<Bus>Nm shall be able to propagate and evaluate the need for synchronized PNC shutdown in the role of a top-level PNC coordinator or intermediate PNC coordinator (optional)	[SWS_FrNm_00055] [SWS_FrNm_00504]
[RS_Nm_02550]	FrNm shall support Partial Networking on FlexRay	[SWS_FrNm_00565]
[RS_Nm_02571]	Nm shall handle requests for synchronized PNC shutdown	[SWS_FrNm_00562] [SWS_FrNm_00563] [SWS_FrNm_00564]
[RS_Nm_02572]	<Bus>Nm shall transmit requests for synchronized PNC shutdown as NM-PDU	[SWS_FrNm_00568] [SWS_FrNm_00569] [SWS_FrNm_00570] [SWS_FrNm_00572] [SWS_FrNm_91006] [SWS_FrNm_91007]
[SRS_BSW_00101]	The Basic Software Module shall be able to initialize variables and hardware in a separate initialization function	[SWS_FrNm_00028]
[SRS_BSW_00159]	All modules of the AUTOSAR Basic Software shall support a tool based configuration	[SWS_FrNm_00020]
[SRS_BSW_00331]	All Basic Software Modules shall strictly separate error and status information	[SWS_FrNm_00021]
[SRS_BSW_00337]	Classification of development errors	[SWS_FrNm_00021]
[SRS_BSW_00369]	All AUTOSAR Basic Software Modules shall not return specific development error codes via the API	[SWS_FrNm_00056] [SWS_FrNm_00057]
[SRS_BSW_00406]	API handling in uninitialized state	[SWS_FrNm_00071] [SWS_FrNm_00544] [SWS_FrNm_00545] [SWS_FrNm_00546]
[SRS_BSW_00429]	Access to OS is restricted	[SWS_FrNm_00220]

▽

△

Requirement	Description	Satisfied by
[SRS_BSW_00432]	Modules should have separate main processing functions for read/receive and write/transmit data path	[SWS_FrNm_00255] [SWS_FrNm_00449]

Table 6.1: Requirements Tracing

7 Functional specification

7.1 Coordination algorithm

The AUTOSAR FlexRay NM is based on a decentralized direct network management strategy, which means that every network node individually performs self-sufficient NM activities based only on the NM-messages that are received or transmitted within the communication system.

The AUTOSAR FlexRay NM coordination algorithm is based on periodic NM-Vote messages received by all nodes in the cluster. Reception of an NM-Vote message indicates that the sending node wants to keep the NM-cluster awake. If any node is ready to enter the Bus-Sleep Mode, it stops sending NM-messages, but as long as NM-messages from other nodes are received, it postpones transition into the Bus-Sleep Mode. Ultimately, if a designated timer elapses as a result of prolonged absence of NM-messages, then the node initiates transition into the Bus-Sleep Mode.

If any node in the NM-cluster requires bus-communication, then it can "wake-up"[1] the NM-cluster from the Bus-Sleep Mode by transmitting NM-Vote messages. For additional details concerning the wakeup procedure, please refer to the Mode Management (see [5], [6]).

FlexRay Network Management is responsible for the following functionalities:

- Periodic Update of FlexRay NM-PDU's
- Encoding and Decoding of FlexRay NM-PDU's
- Transmission Error Handling for FlexRay NM-PDU's
- Notification of the Generic Network Management Interface (NmIf) regarding changes of the FlexRay NM state machine

A special case is the possibility to configure the FrNm of a node as "passive". Such a "passive node" will listen to the NM-messages on the FlexRay Bus (to determine whether to stay awake or to go to sleep), but will not send any NM-messages itself. Thus such a node will follow the decisions the network global consensus but will not influence it. A more detailed description of the requirements of passive nodes can be found in chapter 7.8.5.

The main concept of the AUTOSAR FlexRay NM coordination algorithm can be defined by the following key-requirements:

[SWS_FrNm_00100] ⌈Every network node shall transmit periodic NM-Vote messages to indicate if the node requires bus-communication.⌋

[1]The “wake-up” of the NM-cluster shall not be confused with “wake-up” of the FlexRay cluster, which is not part of the wake-up procedure of the FlexRay NM, as the FlexRay NM requires an already started FlexRay cluster.

[SWS_FrNm_00101]

Upstream requirements: RS_Nm_00048, RS_Nm_00054

⌈If bus communication is released the FlexRay NM module shall perform the transition into the Bus-Sleep Mode at the end of the `FrNmReadySleepCnt` + 1 repetition cycle without any positive NM vote. The absolute time until transition to Bus-Sleep is therefore given with the formula (`FrNmReadySleepCnt` +1) * `FrNmRepetitionCycle` * 'Duration of one FlexRay Cycle' and it starts at the completion of the last repetition cycle containing a positive NM vote.⌋

Example: `FrNmReadySleepCnt` = 3; Repetition Cycle = 4 FlexRay Cycle; FlexRay Cycle = 5 msec: Ready Sleep Time = (3+1)* 4* 5 msec = 80 msec.

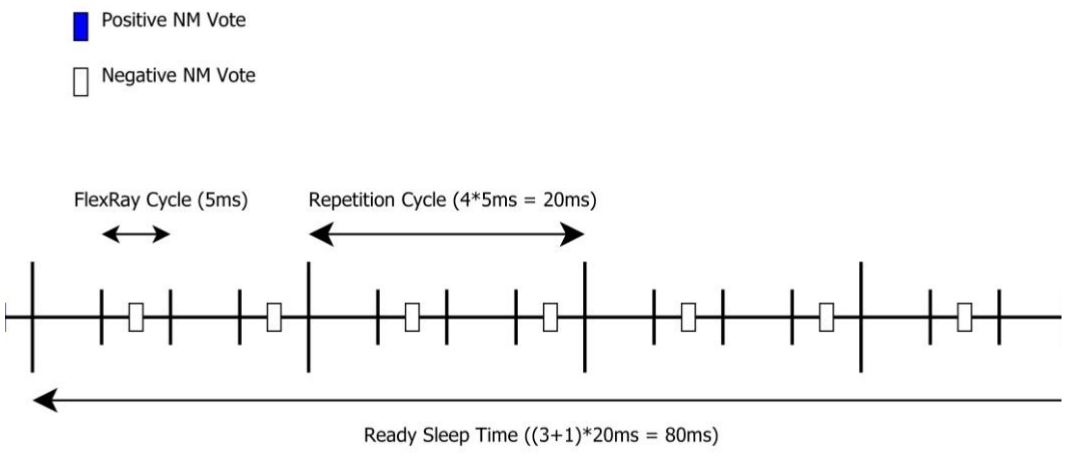
Positive NM Vote
Negative NM Vote
FlexRay Cycle (5ms)
Repetition Cycle (4*5ms = 20ms)
Ready Sleep Time ((3+1)*20ms = 80ms)

Figure 7.1: Transition To BusSleep

[SWS_FrNm_00102] ⌈The AUTOSAR FlexRay NM state machine shall contain states, transitions and triggers required by the AUTOSAR FlexRay NM coordination algorithm as seen from the point of view of one single node in the NM-cluster.⌋

[SWS_FrNm_00103] ⌈Transitions in the AUTOSAR FlexRay NM state machine shall be triggered by calls to selected interface functions or by the expiration of internal timers or counters.⌋

[SWS_FrNm_00168]

Upstream requirements: RS_Nm_00046

⌈The FrNm module shall synchronize state changes in the FlexRay NM state machine with the FlexRay periodic Schedule.⌋

Rationale: The FlexRay NM algorithm is based on the fact that all ECUs, which participate in the FlexRay NM, are synchronized to a global time (based on periodic repetition of its communication scheme, the so called Cycle - see [7]). To prevent asymmetric behavior of the ECUs (e.g., only a subset of the ECUs changes to sleep mode, while the remaining subset stays awake) the FlexRay NM aligns the state changes to a NM Repetition Cycle, which is aligned to a basic FlexRay communication cycle, to guarantee a synchronous behavior of the NM state machines on all ECUs in the NM cluster.

Note: Section 7.7.2 describes on how the implementation will fulfill [SWS_FrNm_00168].

[SWS_FrNm_00225] ⌈The FrNm module's implementer shall realize the FlexRay NM coordination algorithm processor independent, which means the FlexRay NM coordination algorithm shall not rely on any processor specific hardware support and thus shall be realizable on any processor architecture that is in the scope of AUTOSAR.⌋

7.2 Operational modes

In this chapter, the operational modes of the AUTOSAR FlexRay NM coordination algorithm are described in detail. Figure 7.2 shows the detailed UML state chart of the FlexRay NM.

[SWS_FrNm_00105] ⌈The AUTOSAR FrNm shall consist of three operational modes:

- Bus-Sleep Mode
- Synchronize Mode
- Network Mode

⌋

[SWS_FrNm_00106]

Upstream requirements: RS_Nm_00051

⌈Changes in the AUTOSAR FrNm operational modes shall be notified by the FrNm module to the Generic NM Interface module by calling callback function `Nm_StateChangeNotification`.⌋

7.2.1 Bus-Sleep Mode

The Bus Sleep Mode is the default mode after the initialization of the FrNm State Machine, where it remains unless the NM is started (either with a passive startup request or with a network request) or when the power of the CPU is switched off.

In the Bus Sleep Mode, the communication controller can be switched into the sleep mode where wakeup detection mechanisms are activated and power consumption is reduced to a minimal level. The corresponding functionality (shut down of FlexRay, power down of the CPU) will be implemented in other modules. The FlexRay NM will only indicate the readiness of Sleep Mode.

[SWS_FrNm_00134] ⌈When Bus-Sleep Mode is entered, the FlexRay NM shall notify the Generic Network Management Interface by calling `Nm_StateChangeNotification`, except when the Bus-Sleep Mode is entered by default during initialization.⌋

Note: The FlexRay NM will notify the ComM via the Generic NM Interface (macro adaptation layer). This notification is an optional interface.

[SWS_FrNm_00135] ⌈When Bus-Sleep Mode is entered, except by default at initialization, the FrNm module shall notify the upper layer by calling the call-back function `Nm_BusSleepMode`.⌋

[SWS_FrNm_00137] ⌈When the FrNm module enters the Bus-Sleep Mode, the module shall deactivate the transmission of both NM-Data and NM-Vote.⌋

[SWS_FrNm_00175] ⌈When the FrNm module successfully receives a positive Nm vote, and is currently in the Bus-Sleep Mode, it shall notify the ComM via the NmIf module by calling `Nm_NetworkStartIndication`.⌋

Rationale: [SWS_FrNm_00175] is required to avoid race conditions and state inconsistency between Network Management and Mode Management. NM-message reception handling in Bus-Sleep Mode is dependent on the current state of the ECU shutdown/startup process.

Note: The FlexRay NM will notify the ComM via the Generic NM Interface (macro adaptation layer).

[SWS_FrNm_00316] ⌈BusSleep Mode shall be left and the Synchronize Mode shall be entered if Generic NM Interface calls `FrNm_NetworkRequest`.⌋

[SWS_FrNm_00317] ⌈BusSleep Mode shall be left and the Synchronize Mode shall be entered if Generic NM Interface calls `FrNm_PassiveStartUp`.⌋

7.2.2 Synchronize Mode

In the Synchronize Mode, the FrNm state machine waits to be synchronized to the FrNm Repetition Cycle. This is necessary as the FlexRay NM is dependent on state changes being synchronized across the NM Cluster.

[SWS_FrNm_00143] ⌈The FrNm module shall leave the Synchronize Mode and enter the Network Mode at the first boundary between two NM Repetition Cycles.⌋

[SWS_FrNm_00308] ⌈When the FrNm module enters the Synchronize mode, it shall deactivate the transmission of NM-Data and deactivate the transmission of the NM-Vote.⌋

[SWS_FrNm_00340] ⌈If FlexRay NM is in Synchronize state and FlexRay NM receives the indication `FrNm_StartupError` and if the network is requested, then FlexRay NM shall remain in Synchronize state.⌋

[SWS_FrNm_00376] ⌈If FlexRay NM is in Synchronize state and FlexRay NM receives the indication `FrNm_StartupError` and if the network is released, then FlexRay NM shall perform a transition to Bus Sleep State.⌋

[SWS_FrNm_00475] ⌈When the FrNm module has entered Synchronize Mode, it shall notify the Generic NM Interface module by calling `Nm_SynchronizeMode`.⌋

7.2.3 Network Mode

[SWS_FrNm_00107] ⌈The Network Mode of the FrNm module shall consist of three internal states:

- Repeat Message State
- Normal Operation State
- Ready Sleep State

⌋

[SWS_FrNm_00115] ⌈The FrNm module shall synchronize all state changes into and within the Network Mode with the boundary between two NM Repetition Cycles.⌋

Rationale: The FlexRay NM defines a number of FlexRay cycles as NM Repetition Cycle to improve the reliability of the NM vote transmission. Within a NM Repetition Cycle, the NM is not allowed to change the NM-vote. For details see chapter 7.9.

[SWS_FrNm_00108] ⌈On entering the Network Mode state, the FlexRay Network Management shall first enter the internal sub-state Repeat message state.⌋

[SWS_FrNm_00110] ⌈When the FrNm module has entered the Network Mode, it shall notify the Generic NM Interface module by calling `Nm_NetworkMode`.⌋

[SWS_FrNm_00307] ⌈The FrNm module shall immediately leave the Network mode when the function `FrNm_Init` is called.⌋

[SWS_FrNm_00491]

Upstream requirements: RS_Nm_00153

⌈If function `FrNm_PnLearningRequest` is called on a channel where `FrNmDynamicPncToChannelMappingEnabled` is set to TRUE and FrNm is in the Network Mode the FrNm module shall set the Repeat Message Bit and the Partial Network Learning Bit in the CBV to 1 on this channel and change to or restart the Repeat Message State at the end of a NM Repetition Cycle.⌋

[SWS_FrNm_00492] ⌈If the bits Partial Network Learning and Repeat Message Requests both are received on a channel where `FrNmDynamicPncToChannelMappingEnabled` is set to TRUE and FrNm is in the Network Mode the FrNm module shall set the Partial Network Learning Bit in the CBV to 1 on this channel and change to or restart the Repeat Message State at the end of a NM Repetition Cycle.⌋

Note: Restart in [SWS_FrNm_00491] or [SWS_FrNm_00492] means that FrNm is already in Repeat Message State and then a complete re-entry of the Repeat Message State has to be performed once.

7.2.3.1 Repeat Message State

For nodes that are not configured as passive, the Repeat Message State ensures that any transition from the Bus-Sleep to the Network Mode becomes visible to the other nodes on the network. Additionally, it ensures that any node stays active for a minimum amount of time. Optionally it can be used for detection of present nodes (see explanation in chapter 7.8.3).

[SWS_FrNm_00116] ⌈When the FrNm module enters the Repeat Message State, it shall activate the transmission of NM-Data if the node is configured for NM-Data transmissions and the node shall vote to keep the cluster awake if the node is configured to allow voting.⌋

[SWS_FrNm_00117] ⌈When the FrNm module enters the Repeat Message State, it shall start the NM Repeat Message Timer (time is configurable by parameter `FrNmRepeatMessageTime`).⌋

[SWS_FrNm_00120] ⌈The FrNm module shall leave the Repeat Message State if the NM Repeat Message Timer is expired. During this transition it shall clear the Repeat Message Request Bit.⌋

Note: This transition takes place at a repetition cycle boundary as `FrNmRepeatMessageTime` is an integer multiple of one repetition cycle.

[SWS_FrNm_00121] ⌈When the FlexRay NM module leaves the Repeat Message State (see [SWS_FrNm_00120]), it shall enter the Normal Operation State if the network is requested.⌋

[SWS_FrNm_00122] ⌈When the FlexRay NM module leaves the Repeat Message State (see [SWS_FrNm_00120]) it shall enter the Ready Sleep State if the network is released.⌋

[SWS_FrNm_00384] ⌈If FlexRay NM is in Repeat Message State and global time could not be retrieved, then FlexRay NM shall perform a transition to Synchronize state.⌋

[SWS_FrNm_00493]

Upstream requirements: RS_Nm_00153

⌈If `FrNmDynamicPncToChannelMappingEnabled` is set to TRUE FrNm shall clear the Partial Network Learning Bit when leaving the Repeat Message State.⌋

7.2.3.2 Normal Operation State

The Normal Operation State ensures that any node can keep the NM-cluster awake as long as the network is requested.

Note: This State will not be reached if the node is configured as "passive node" [SWS_FrNm_00187]. For such a node, it is up to the implementation to optimize this state and remove the code corresponding to this state.

[SWS_FrNm_00123] ⌈When the FrNm module enters the Normal Operation State, it shall activate the transmission of NM-Data and the Node shall vote to keep the cluster awake (send "positive" NM-Votes).⌋

[SWS_FrNm_00124] ⌈If FrNmNodeDetectionEnabled is set to TRUE the FlexRay NM module shall leave the Normal Operation State and enter the Repeat Message State at the end of a NM Repetition Cycle when a Repeat Message Request is active. During this transition it shall set the Repeat Message Request Bit if the Repeat Message Request is active due to a call of FrNm_RepeatMessageRequest.⌋

[SWS_FrNm_00125] ⌈The FlexRay NM module shall leave the Normal Operation State and enter the Ready Sleep State if no Repeat Message Request is active and the network has been released and if a Repetition Cycle is completed.⌋

[SWS_FrNm_00342] ⌈If FlexRay NM is in Normal Operation state and FlexRay NM cannot retrieve global time, then FlexRay NM shall perform a transition to Synchronize state.⌋

7.2.3.3 Ready Sleep State

The Ready Sleep State ensures that any node in the NM-cluster waits to transition to the Bus-Sleep Mode as long as any other node keeps the NM-cluster awake.

[SWS_FrNm_00126] ⌈When the FrNm module enters the Ready Sleep State, it shall deactivate the transmission of NM-Data and shall transmit negative NM-Votes.⌋

Note: If passive mode is disabled, in some cases NM PDUs have to be transmitted in Ready Sleep State to grant a synchronized shutdown in the network, e.g. retransmission of PN shutdown messages

[SWS_FrNm_00129] ⌈The FlexRay NM module shall leave the Ready Sleep State (and the Network Mode) and enter the Bus-Sleep Mode at the end of the FrNmReadySleepCnt +1 NM Repetition Cycle without any positive NM-Votes.⌋

[SWS_FrNm_00130] ⌈If FrNmNodeDetectionEnabled is set to TRUE the FlexRay NM module shall leave the Ready Sleep State and enter the Repeat Message State at the end of a NM Repetition Cycle when a Repeat Message Request is active and Ready Sleep Time has not expired. During this transition it shall set the Repeat Message Request Bit if the Repeat Message Request is active due to a call of FrNm_RepeatMessageRequest.⌋

[SWS_FrNm_00131] ⌈The FlexRay NM module shall leave the Ready Sleep State and enter the Normal Operation State at the end of a NM Repetition Cycle if Ready Sleep Time has not expired and no Repeat Message Request is active and the network has been requested.⌋

Note: A local request to keep the network awake (i.e., Network Request) will be ignored in the Ready Sleep State in the last Repetition Cycle where sleep shall be entered due to [SWS_FrNm_00129].

[SWS_FrNm_00444] ⌈If FlexRay NM is in Ready Sleep state, the configuration parameter `FrNmCycleCounterEmulation` is set to FALSE, global time cannot be retrieved and network is requested again then FlexRay NM shall perform a transition to Synchronize state.⌋

[SWS_FrNm_00338] ⌈If FlexRay NM is in Ready Sleep state, the configuration parameter `FrNmCycleCounterEmulation` is set to FALSE, network is still not requested and on reception of `FrNm_StartupError` the FlexRay NM shall perform a transition to Bus Sleep state.⌋

[SWS_FrNm_00378] ⌈If FlexRay NM enters Ready Sleep state and the configuration parameter `FrNmCycleCounterEmulation` is set to TRUE, the FlexRay NM shall provide a mechanism that detects the repetition cycle end independently from the global time.⌋

Note: [SWS_FrNm_00378] could be fulfilled e.g. by a timer or counter which duration fits to one repetition cycle. As long as global time can be retrieved this timer or counter could be synchronized to the exact timing.

[SWS_FrNm_00379] ⌈If FlexRay NM is in Ready Sleep state, the configuration parameter `FrNmCycleCounterEmulation` is set to TRUE and the FlexRay Global Time could not be retrieved, every time the emulated repetition cycle end is reached, the FlexRay NM shall count this as one NM Repetition Cycle without any positive vote. If FlexRay NM counted `FrNmReadySleepCnt` plus one NM Repetition Cycles without any positive NM-Vote the FlexRay NM shall perform a transition to Bus Sleep state.⌋

7.3 Network states

Network states (i.e., 'requested' and 'released') are two additional "states of the AUTOSAR FlexRay NM state machine" that exist in parallel to the state machine described in chapter 7.4.

Network states distinguish between whether the software components need to communicate on the bus (the network state is then 'requested') or not (the network state is then 'released'). Note that if the network is released an ECU may still communicate because at least one other ECU still requests the network.

This network states reflect the demand of an upper layer (e.g., ComM) on keeping the bus awake (network requested) or not (network released).

7.4 UML State Chart Diagram

The figure shows an UML state diagram with respect to the API specification.

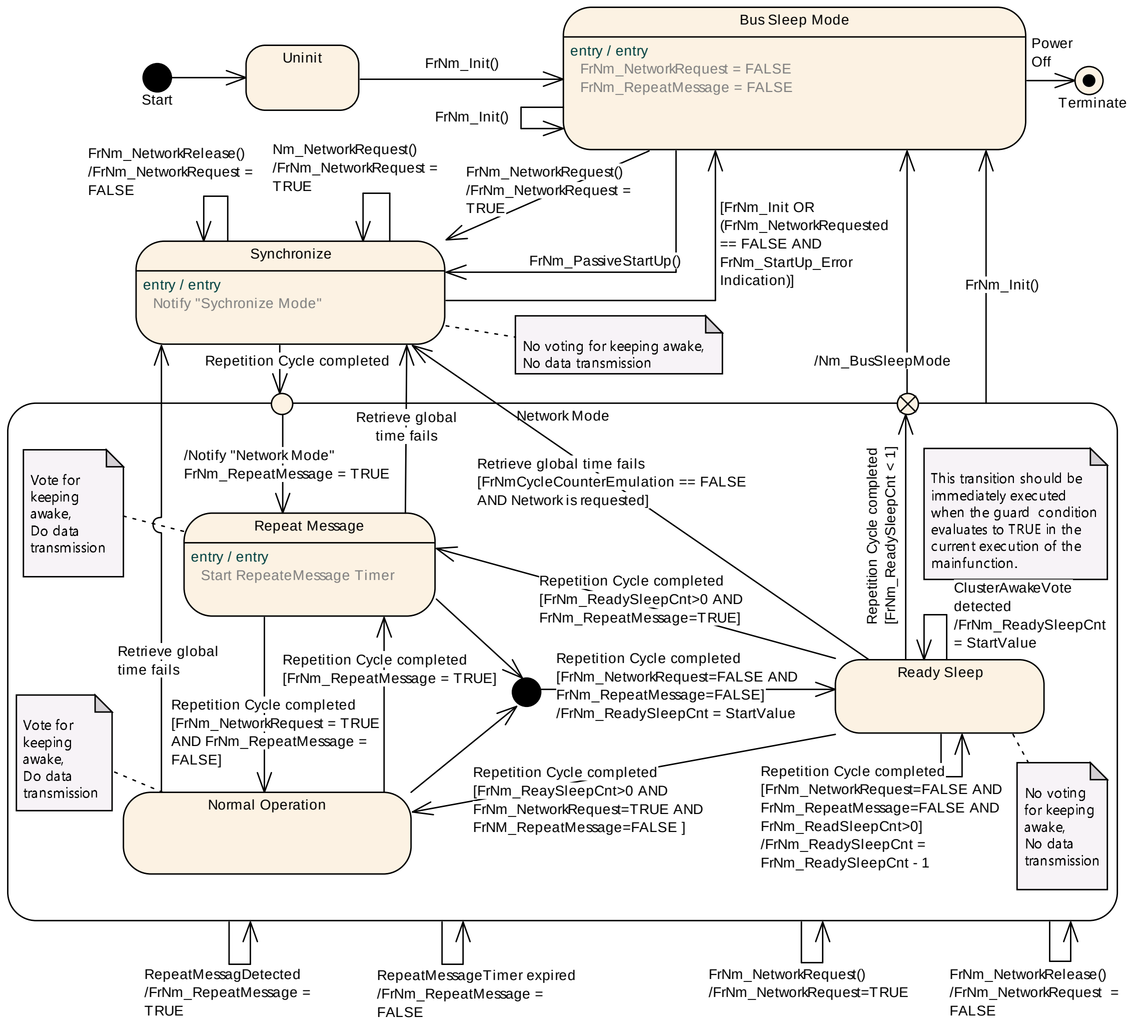
Start
Uninit
FrNm_Init()
Bus Sleep Mode
entry / entry
FrNm_NetworkRequest = FALSE
FrNm_RepeatMessage = FALSE
FrNm_Init()
Power Off
Terminate
FrNm_NetworkRelease() /FrNm_NetworkRequest = FALSE
Nm_NetworkRequest() /FrNm_NetworkRequest = TRUE
FrNm_NetworkRequest() /FrNm_NetworkRequest = TRUE
FrNm_PassiveStartUp()
[FrNm_Init OR (FrNm_NetworkRequested == FALSE AND FrNm_StartUp_Error Indication)]
FrNm_Init()
Synchronize
entry / entry
Notify "Sychronize Mode"
No voting for keeping awake, No data transmission
Repetition Cycle completed
/Nm_BusSleepMode
Retrieve global time fails
Network Mode
/Notify "Network Mode" FrNm_RepeatMessage = TRUE
Retrieve global time fails [FrNmCycleCounterEmulation == FALSE AND Network is requested]
Repetition Cycle completed [FrNm_ReadySleepCnt < 1]
This transition should be immediately executed when the guard condition evaluates to TRUE in the current execution of the mainfunction.
Vote for keeping awake, Do data transmission
Repeat Message
entry / entry
Start RepeateMessage Timer
Repetition Cycle completed [FrNm_ReadySleepCnt>0 AND FrNm_RepeatMessage=TRUE]
ClusterAwakeVote detected /FrNm_ReadySleepCnt = StartValue
Retrieve global time fails
Repetition Cycle completed [FrNm_RepeatMessage = TRUE]
Repetition Cycle completed [FrNm_NetworkRequest=FALSE AND FrNm_RepeatMessage=FALSE] /FrNm_ReadySleepCnt = StartValue
Ready Sleep
Repetition Cycle completed [FrNm_NetworkRequest = TRUE AND FrNm_RepeatMessage = FALSE]
Vote for keeping awake, Do data transmission
Normal Operation
Repetition Cycle completed [FrNm_ReaySleepCnt>0 AND FrNm_NetworkRequest=TRUE AND FrNM_RepeatMessage=FALSE]
Repetition Cycle completed [FrNm_NetworkRequest=FALSE AND FrNm_RepeatMessage=FALSE AND FrNm_ReadySleepCnt>0] /FrNm_ReadySleepCnt = FrNm_ReadySleepCnt - 1
No voting for keeping awake, No data transmission
RepeatMessagDetected /FrNm_RepeatMessage = TRUE
RepeatMessageTimer expired /FrNm_RepeatMessage = FALSE
FrNm_NetworkRequest() /FrNm_NetworkRequest=TRUE
FrNm_NetworkRelease() /FrNm_NetworkRequest = FALSE

Figure 7.2: UML State Chart

7.5 Initialization and Startup

[SWS_FrNm_00071]

Upstream requirements: SRS_BSW_00406

⌈The FrNm module shall store the initialization status in a private variable.⌋

Note: The FrNm module's environment shall initialize the FrNm module after the corresponding FlexRay Interface is initialized and before any other FlexRay NM service is called.

[SWS_FrNm_00136] ⌈FlexRay NM shall set the flag FrNm_NetworkRequest to FALSE after initialization and enter into bus sleep mode.⌋

[SWS_FrNm_00544]

Upstream requirements: SRS_BSW_00406

⌈During initialization the FrNm module shall set each byte of the user data to 0xFF.⌋

[SWS_FrNm_00545]

Upstream requirements: SRS_BSW_00406

⌈During initialization the FrNm module shall set the Control Bit Vector to 0x00.⌋

[SWS_FrNm_00546]

Upstream requirements: SRS_BSW_00406

⌈During initialization and if FrNmPnEnabled is TRUE, the FrNm module shall set each byte of the PNC bit vector to 0x00.⌋

Note: FrNm do not support re-transmission of PN shutdown messages (as it is supported e.g. on CanNm), since NM messages on FlexRay are transmitted with a short period. At initialization, the control bit vector is set to 0x00, which includes the PNSR bit (PNSR bit set to '0' == transmission of PN shutdown message deactivated). The setting of the PNSR bit is controlled with FrNm_ActivateTxPnShutdownMsg and FrNm_DeactivateTxPnShutdownMsg. No additional requirements as e.g. on CanNm are needed.

7.6 Communication

Using NM-Messages, FlexRay NM provides mechanisms for information exchange to coordinate shutdown. These messages are sent according to the schedule configured within each ECU and coordinated across the NM Cluster.

7.6.1 General requirements

The FrIf shall be configured for every node to receive all NM vote messages that are not aggregated by the FlexRay controller.

Note: FlexRay supports an automatic aggregation of Network Management data called Network Management Vector (see chapter 9.3.3.4 Network management service, page 209, of the FlexRay protocol specification [7]).

The FlexRay Controller calculates this NM-Vector by exchanging the NM-Vector in selected network management enabled frames within the static segment of the communication cycle. Every node may be configured to send one NM-Vector in one of its transmission slots.

The FlexRay communication controller will maintain an aggregated network management vector throughout each communication cycle by applying a bit-wise OR between each received Network Management Vectors (regardless of whether the frame is subscribed to a receive buffer).

This method is a powerful way to receive the NM-Vector from nodes on the network without involving the CPU, but requires at least one send-slot for every node (participating in the Network Management) in the static segment.

[SWS_FrNm_00058] ⌈The FlexRay NM decisions shall be influenced by every received NM-Vote and every NM-Vote aggregated by the FlexRay controller.⌋

[SWS_FrNm_00205] ⌈A FlexRay NM Message shall only contain NM-Vote, NM-Data or both.⌋

[SWS_FrNm_00147] ⌈The FrNm module shall be able to separately transmit NM Data and NM Vote.⌋

Rationale: The voting algorithm of FlexRay is kept independent of the transmission of the NM data as the FlexRay Protocol provides a HW support for sending and receiving NM votes (see [7]). To use this feature and to increase the update rate of NM-Votes (compared to the update rate of the NM-Data), the transmission of NM-Data and NM-Vote may be separated.

[SWS_FrNm_00160] ⌈It shall be configurable with the configuration parameter `FrNmPduScheduleVariant` which NM-message transmission format (NM-Data, NM-Vote and the combined NM-Data/Vote format) are recognized by the FrNm module.⌋

Rationale: The FrNm module must be capable of receiving and processing the NM-messages which are on the FlexRay bus ([SWS_FrNm_00058]). To optimize the resource need of the NM it must be configurable which formats are supported by the NM, in order to avoid the overhead of "unused" formats.

[SWS_FrNm_00148] ⌈Every FlexRay NM node shall be capable of sending the NM-Vote (in the either the static or the dynamic segment) of the FlexRay bus schedule if the node is not configured as "Passive".⌋

Note: The FrIf configuration is responsible for the actual FlexRay schedule configuration. This requirement is to require that the FrNm will support such a configuration.

[SWS_FrNm_00169] ⌈In every FlexRay NM node it shall be independently configurable through the configuration parameter `FrNmHwVoteEnable` to use the FlexRay NM HW support for reception of NM-Votes that are transmitted in the static segment.⌋

[SWS_FrNm_00151] ⌈Every FlexRay NM node shall be capable of sending the NM-Data in either a static slot or in a dynamic slot.⌋

Note: The FrIf and FrDv configuration is responsible for the actual FlexRay schedule configuration. This requirement requires that the FrNm support such a configuration.

Although it is possible to use multiple FlexRay slots to transmit NM-Data, only one slot should be used in order to limit the number of FlexRay buffers needed for the reception of the NM-Data from different nodes.

[SWS_FrNm_00335] ⌈When NM-Vector hardware service from the FlexRay CC is used, then `FrIf_GetNmVector` will be used to retrieve the aggregated NM-Vector.⌋

Rationale: The FlexRay NM must be able to go to Bus Sleep state after synchronization is lost.

7.6.2 FlexRay NM-PDU format

As specified in [SWS_FrNm_00147] the FlexRay NM is capable of sending NM-Vote and NM-Data independently. Therefore several corresponding PDU formats exist for the NM-Vote and for the NM-Data. To also support an associated transmission of NM-Vote and NM-Data in the static segment the NM-Data PDU contains an optional Voting Bit.

7.6.2.1 FlexRay NM-Data PDU format

The figure below shows the format of the Network Management PDU for example with 8 byte PDU length where the control bit vector is locacted at the first byte, the source node identifier is located in the second byte, user data is used and partial networt is enabled. The user data range is located between the system bytes and the PNC bit vector.

	Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Byte 7	PNC bit vector - byte 3							
Byte 6	PNC bit vector - byte 2							
Byte 5	PNC bit vector - byte 1							
Byte 4	PNC bit vector - byte 0							

▽

△

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 3	User data 1							
Byte 2	User data 0							
Byte 1	Source Node Identifier							
Byte 0	Set to "0"	Control Bit Vector						

Table 7.1: FlexRay NM-Data PDU Format

Note:

- In FrNm the control bit vector is mandatory and therefore not configurable.
- The NM-PDU length is defined by the PduLength parameter [EcuC003_Conf] in the "global" ECUC module (see Ecu Configuration specification).

[SWS_FrNm_00222]

Upstream requirements: RS_Nm_02505

⌈The support of the Source Node Identifier by the FrNm module shall be configurable at pre-compile time by the configuration parameter `FrNmSourceNodeIdentifierEnabled`.⌋

Note: Setting the `FrNmSourceNodeIdentifierEnabled` to FALSE means that in the NM PDU no space is occupied by the source node identifier. Hence one more byte is available for user data or PNC bit vector

[SWS_FrNm_00531] ⌈The remaining bytes not assigned to Nm System Bytes or PNC bit vector shall be available for User Data.⌋

Note: According to [8] ([TPS_SYST_03069], [TPS_SYST_03070], [TPS_SYST_03071], [TPS_SYST_03072]) the use and location of user data is configurable. If user data are used, the user data are placed within the PduLenght of the NM-PDU and do not overlap with the range of system bytes or PNC bit vector. If partial network functionaliy is enabled (`FrNmPnEnabled` is set to TRUE) and user data are used, the user data range is exclusively located either between the system bytes and the PNC bit vector or between the PNC bit vector and the end of the NM-PDU. The length of user data range shall be calculated according the following restrictions:

- If the user data range resides between the system bytes and the PNC bit vector, then the length of the user data range is determined by the difference of the PNC bit vector offset and the length of the system bytes.
- If the user data range resides between the PNC bit vector and the end of the NM-PDU, then the length of the user data range is determined by the difference of the NM-PDU length and the position/index of the last byte of the PNC bit vector (defined by PNC bit offset + PNC bit vector length)

If partial network functionaliy is disabled (`FrNmPnEnabled` is set to FALSE) and user data are used, the user data range is determined by the difference of NM-PDU length and the length of the system bytes.

The NM-Data PDU Control Bit Vector of the FrNm module format is described in [SWS_FrNm_00156].

[SWS_FrNm_00156] Control Bit Vector Format ⌈

	Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Byte	Reserved	Partial Network Information Bit	Partial Network Learning Bit	Active Wakeup Bit	NM Coordinator Sleep Ready	Reserved	PN Shutdown Request Bit	RptMsgRequest

⌋

Note: Bit 1 and 2 were used in R3.2 as NM Coordinator ID (Low Bit)

[SWS_FrNm_00055]

Upstream requirements: RS_Nm_02517, RS_Nm_02548, RS_Nm_02540

⌈The Control Bit Vector of the FrNm module shall consist of:

Bit 0 Repeat Message Request Bit (RptMsgRequest):

- 0: Repeat Message State not requested
- 1: Repeat Message State requested

Bit 1 PN Shutdown Request Bit (PNSR):

- 0: NM message does not contain synchronized Partial Network shutdown request
- 1: NM message does contain synchronized Partial Network shutdown request for at least one PNC

Bit 3 NM Coordinator Sleep Ready Bit:

- 0: NM cluster is not ready to sleep
- 1: NM cluster is ready to sleep (All nodes of the NM cluster are ready to sleep)

Bit 4 Active Wakeup Bit:

- 0: Node has not woken up the network
- 1: Node has woken up the network

Bit 5 Partial Network Learning Bit (PNL):

- 0: PNC learning is not requested
- 1: PNC learning is requested

Bit 6 Partial Network Information Bit (PNI):

- 0: NM message contains no Partial Network request information
- 1: NM message contains Partial Network request information

Bit 2, 7 are reserved for future extensions

- 0: Disabled / Reserved for future usage.

⌋

[SWS_FrNm_00161] ⌈The FrNm module shall set the reserved bit(s) of the Control Bit Vector to 0b.⌋

7.6.2.2 FlexRay NM-Vote PDU format

The NM-Vote PDU format of the FrNm module is decribed in [SWS_FrNm_00215].

[SWS_FrNm_00215] NM-Vote PDU Format ⌈

	Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Byte 0	Vote	Set to "0"						

⌋

[SWS_FrNm_00216] ⌈The NM-Vote PDU format of the FrNm module shall contain a Voting Bit (Vote) with the following meaning:

- 0: vote against keeping awake
- 1: vote for keeping awake

⌋

Note: In case of transmitting the NM-Vote (combined or separately) in the dynamic segment the vote bit is not needed as the presence of the NM PDU at all is sufficient. Therefore within the dynamic segment it is not necessary to set the vote bit.

7.6.2.3 Combination of NM-PDUs

When the NM-Vote and NM-Data are combined within one PDU (see chapter 7.9) the content of the NM-Vote will be combined with the content of the Control Bit Vector

(CBV) Byte of the NM-Data as shown in Table 7.2 below. The following requirements specify this combination.

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
NM-Data PDU - CBV	Set to "0"	Partial Network Information Bit	Partial Network Learning Bit	Active Wakeup Bit	NM Coordinator Sleep Ready	Res	Res	RptMsgRequest
+								
NM-Vote PDU	Vote	Set to "0"						
=								
Combined CBV and Vote	Vote	Partial Network Information Bit	Partial Network Learning Bit	Active Wakeup Bit	NM Coordinator Sleep Ready	Res	Res	RptMsgRequest

Table 7.2: Combined NM-Vote and NM-Data CBV Format

[SWS_FrNm_00162] ⌈The FrNm module shall combine the NM-Vote PDU Format with the Control Bit Vector Format of the NM-Data PDU in case the FrNm module shall transmit the NM-Vote in the same PDU as the NM-Data.⌋

7.6.3 FlexRay NM-PDU transmission

For the FlexRay NM-PDU transmission both decoupled or immediate buffer access can be used. For more details see FlexRay Interface SWS [9].

7.6.4 FlexRay NM-PDU reception

The FlexRay Reception Indication is used to indicate reception of FlexRay NM-PDU receptions. For more details see FlexRay Interface SWS [9].

7.6.5 Functional requirements on FrNm API

The following requirements define the available FlexRay NM functions.

[SWS_FrNm_00037]

Upstream requirements: RS_Nm_02508

⌈The set of the Source Node Identifier shall be configurable using the configuration parameter `FrNmNodeId`.⌋

[SWS_FrNm_00494]

Upstream requirements: RS_Nm_00153

⌈If `FrNmRepeatMsgIndEnabled` is set to TRUE and the Repeat Message Request bit set to 1 is received FrNm module shall call the callout function `Nm_RepeatMessageIndication` only the first time until Repeat Message State has been left again. In case the Partial Network Learning Bit is also received with value 1 and `FrNmDynamicPncToChannelMappingEnabled` is set to TRUE the parameter pnLearningBitSet shall be set to TRUE in this function call, otherwise to FALSE.⌋

Note: When Repeat Message Bit is received NM will enter or restart Repeat Message State, but the bits will still be received as requestor will send until he leaves Repeat Message State to be fault-tolerant regarding possible loss of messages. State Change and callout are only needed once the first time the node received it.

7.7 Execution

The FlexRay NM State machine and hence the invocation of the FlexRay NM MainFunction has to be synchronized with the FlexRay communication schedule ([SWS_FrNm_00168]). FlexRay NM decisions and state changes have to be aligned to the FlexRay communication cycle. To guarantee synchronized state changes and decisions of the FrNm, the FlexRay NM MainFunction (`FrNm_MainFunction_<FrNmChannel.ShortName>`) has to be executed within a specific time window as shown in Figure 7.3. The borders of this window are defined on one side by the availability of all NM-Votes (and availability of the repeat message bit if node detection is enabled) of the actual cycle and on the other side by the last point in time where the own NM-Vote (of the next cycle) has to be sent.

As the relative time for a FlexRay cycle may vary due to the FlexRay clock rate correction, and the FlexRay NM algorithm is dependent on the synchronisation to the FlexRay network, it is not recommended to use a CPU time service. Instead this can (for example) be achieved by using an AUTOSAR OS Schedule Table which is synchronized to the global (FlexRay) time.

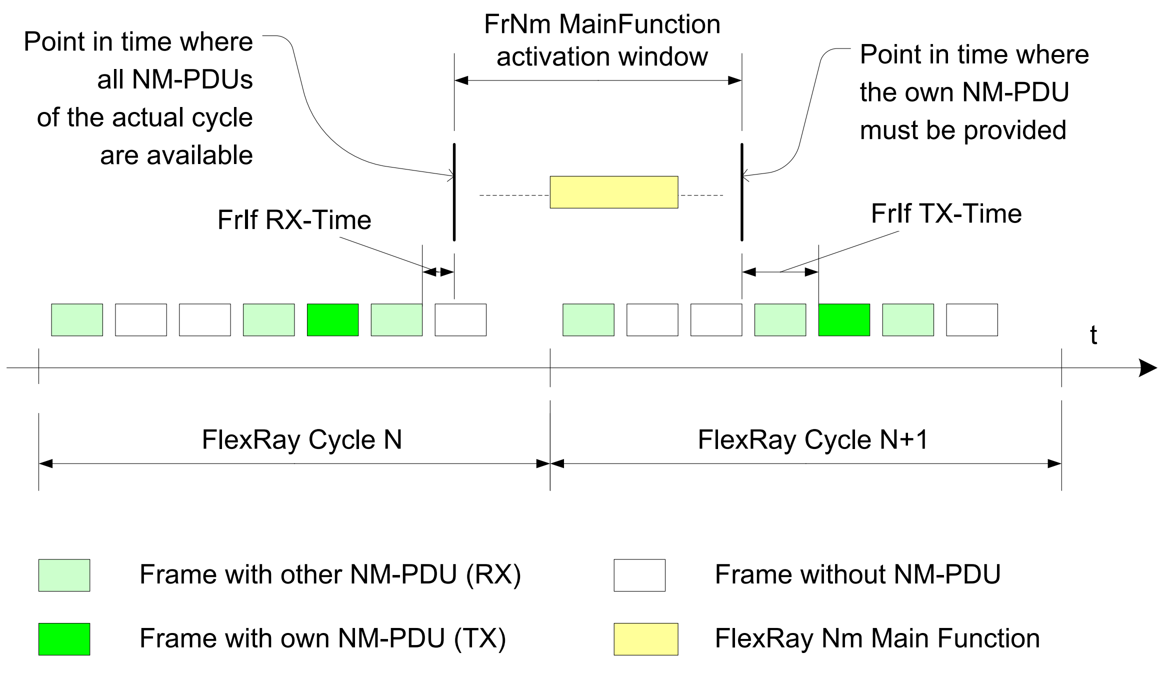
Point in time where all NM-PDUs of the actual cycle are available
FrNm MainFunction activation window
Point in time where the own NM-PDU must be provided
FrIf RX-Time
FrIf TX-Time
t
FlexRay Cycle N
FlexRay Cycle N+1
Frame with other NM-PDU (RX)
Frame without NM-PDU
Frame with own NM-PDU (TX)
FlexRay Nm Main Function

Figure 7.3: FlexRay NM Mainfunction Execution

Note: The time duration denoted as FrIf RX-Time is the time duration between the end of transmission of the FlexRay frame till the call to `FrNm_RxIndication` by the FlexRay Interface (FrIf) via the L-SDU Router (LSduR). The time duration denoted as FrIf TX-Time is the time duration between the FrNm's call to `FrIf_Transmit` via the L-SDU Router (LSduR) till the start of the transmission of the corresponding frame.

7.7.1 FlexRay NM-Main Function structure

The FlexRay NM MainFunction will hold the "automated" functionality of the FlexRay NM - as there is the periodic transmission of NM-Messages, the processing of received NM-Messages and the periodic processing of the FlexRay NM state machine (at the boundary between two NM-Repetition Cycles).

[SWS_FrNm_00010] ⌈The FrNm module shall call `LSduR_FrNmTransmit`, which results in a call of the FlexRay Interface function `FrIf_Transmit`, to transmit NM-Vote and NM-Data if the transmission of cyclic NM-messages is started.⌋

Note: The FlexRay Interface module calls `LSduR_FrIfTxConfirmation`, which in turn calls the FrNm module function `FrNm_TxConfirmation` with `E_OK`, when a NM-message is successfully transmitted and a transmit confirmation is supported and configured within the FlexRay Interface.

Note: The FlexRay Interface module calls `LSduR_FrIfRxIndication`, which in turn calls the FrNm module function `FrNm_RxIndication`, when a NM-message is received. It is up to the implementation as to how the FrNm module is to handle the data from the NM-message. It can be immediately processed (to reduce the memory consumption), or it can be stored to be available to be processed (to reduce computing time).

[SWS_FrNm_00375] ⌈The FrNm module shall retrieve the current FlexRay communication cycle via the API `FrIf_GetGlobalTime()`.⌋

7.7.2 FlexRay NM-MainFunction execution

Note: The FlexRay NM module integrator shall define a schedule to activate the FlexRay NM MainFunction synchronously to the FlexRay communication cycle.

Note: The FrNm module's environment (SchM[2]) should execute the FlexRay NM MainFunction (`FrNm_MainFunction_<FrNmChannel.ShortName>`) at least once a FlexRay communication cycle, synchronous to the FlexRay global time when FlexRay global time is available within the time window depicted in Figure 7.3. The execution of the FlexRay NM MainFunction should not cross a FlexRay cycle boundary. The FlexRay NM MainFunction should be executed periodically even when FlexRay global time is no longer available. The periodicity of FlexRay NM MainFunction is configurable in the SchM configuration which has to correspond to the value of the `FrNmMainFunctionPeriod` configuration parameter.

Rationale: This is necessary because FlexRay NM state changes may influence whether the NM-Vote PDU should be transmitted in the subsequent cycle. Since state changes are influenced by the aggregated NM vote in a given cycle and the state change subsequently influences the NM-Vote, the FlexRay NM MainFunction must execute in a time window bounded by the cycle end and the transmission slot for the NM-Vote PDU ([SWS_FrNm_00168]).

Note: The AUTOSAR OS [10] provides the method of Schedule Tables which can be synchronized with a Global Time. These could be used to fulfill the FlexRay NM execution requirements. The AUTOSAR FlexRay NM shall try to use the second absolute timer whenever possible as race conditions might occur between the FlexRay NM access to the first absolute timer and the FlexRay Interface Job List execution function access to the first absolute timer.

[SWS_FrNm_00356] ⌈The state machine guards, transitions, conditions, and actions should be evaluated at most once in each execution of the `FrNm_MainFunction_<FrNmChannel.ShortName>`.⌋

[SWS_FrNm_00311] ⌈FlexRay NM module shall provide a configuration parameter named `FrNmMainAcrossFrCycle` to select adapt the behavior of the FlexRay NM MainFunction to the point in time the FlexRay NM MainFunction is executed.⌋

[2]part of the RTE module

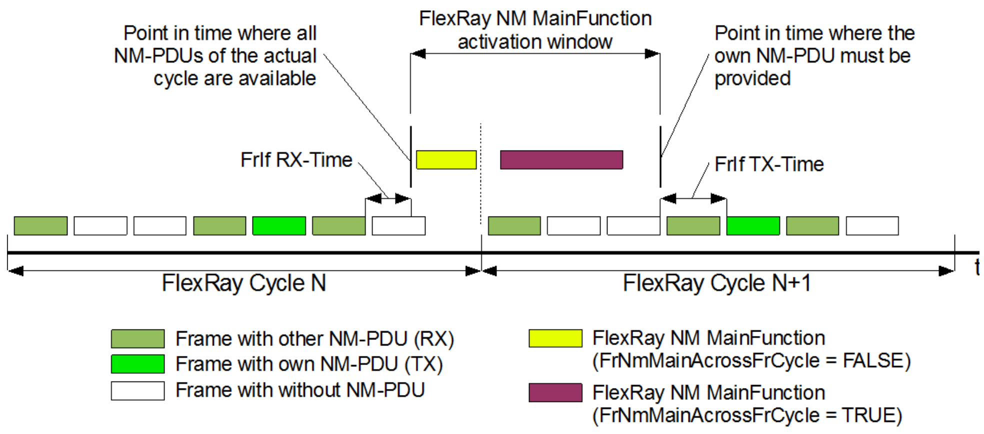
Point in time where all NM-PDUs of the actual cycle are available
FlexRay NM MainFunction activation window
Point in time where the own NM-PDU must be provided
FrIf RX-Time
FrIf TX-Time
FlexRay Cycle N
FlexRay Cycle N+1
t
Frame with other NM-PDU (RX)
Frame with own NM-PDU (TX)
Frame with without NM-PDU
FlexRay NM MainFunction (FrNmMainAcrossFrCycle = FALSE)
FlexRay NM MainFunction (FrNmMainAcrossFrCycle = TRUE)

Figure 7.4: Example NM PDU containing PNC bit vector

Note: If the FlexRay NM Vector is only available at the end of a FlexRay cycle then evaluation takes place in the following FlexRay cycle. - This depends on the configuration and implementation of the FlexRay communication controller hardware.

[SWS_FrNm_00467] ⌈The evaluation of the condition RepetitionCycleCompleted and thus the generation of the RepetitionCycleCompleted event shall be subject to the following rules:

1. If `FrNmMainAcrossFrCycle` is set to FALSE, then a RepetitionCycleCompleted event shall be generated in the last FlexRay communication cycle of the repetition cycle after we receive all the votes.
2. If `FrNmMainAcrossFrCycle` is set to TRUE, then a RepetitionCycleCompleted event shall be generated if a repetition cycle boundary has been crossed since the previous call of the FlexRay NM MainFunction execution before any vote is transmitted.

⌋

Example for case 1: If the votes are scheduled in cycles 1 and 2, and the repetition cycle period is 4 FlexRay communication cycles, then the RepetitionCycleCompleted event can be generated (once only) any time during cycle 3, 7, 11, 15, etc..

Example for case 2: If the votes are scheduled in cycles 1 and 2, and the repetition cycle period is 4 FlexRay communication cycles, then the RepetitionCycleCompleted can be generated (once only) any time during cycle 0, 4, 8, 12, 16, etc.

Note: CycleNumber stands for the FlexRay communication cycle number whose value is anywhere between 0 and 63 as integers. RepetitionCycleLength is the number of communication cycles within one NM Repetition Cycle. FlexRay "CycleEnd" Event is the event generated at the boundary of two consecutive FlexRay communication cycles.

7.8 Additional Features

7.8.1 Cluster size

[SWS_FrNm_00179]

Upstream requirements: RS_Nm_00150, RS_Nm_00144

⌈The AUTOSAR FlexRay NM algorithm shall support up to 64 nodes per NM-Cluster.⌋

Note: The AUTOSAR FlexRay NM algorithm can support an arbitrary number of nodes per NM-cluster (even more than the maximum of 64 nodes per FlexRay cluster). This upper limit is only a matter of configuration, since the upper limit is not fixed and depends on the trade off between response time, fault-tolerance and resulting bus load configured for the AUTOSAR FlexRay NM coordination algorithm.

7.8.2 Detection of Remote Sleep Indication (optional)

The "Remote Sleep Indication" signals a situation where a node detects that all other nodes are ready to sleep, but the node where the indication occurs is still keeping the bus awake.

[SWS_FrNm_00180]

Upstream requirements: RS_Nm_00150

⌈The detection of remote sleep indication by the FrNm module shall be configurable at pre-compile time by the configuration parameter `FrNmRemoteSleepIndicationEnabled`.⌋

Note: If a node is configured as Passive [SWS_FrNm_00187], then the remote sleep indication shall not be used because the node cannot vote so it is incapable of being the only node keeping the NM Cluster awake. Consequently the remote sleep indication simply cannot occur.

[SWS_FrNm_00181]

Upstream requirements: RS_Nm_00052

⌈If no NM-messages with an indication to keep the bus awake are received in the Normal Operation State for a configurable amount of time determined by the `FrNmRemoteSleepIndTime`, then the NM shall notify the Generic NM Interface module that all other nodes in the cluster are ready to sleep (the 'Remote Sleep Indication') by calling `Nm_RemoteSleepIndication`.⌋

[SWS_FrNm_00186] ⌈The FrNm module shall reject a check of Remote Sleep Indication (`FrNm_CheckRemoteSleepIndication`) when not in Network Mode. The function FrNm_CheckRemoteSleepIndication shall immediately return the value E_NOT_OK when not in Network Mode and shall not execute any functionality.⌋

[SWS_FrNm_00229] ⌈If a Remote Sleep Indication has been previously detected by the FrNm module and if an NM-message with an indication to keep the bus awake is received in the Normal Operation State or Ready Sleep State, then the NM shall notify the Generic NM Interface module that some nodes in the cluster are not ready to sleep anymore (Remote Sleep Cancellation) by calling Nm_RemoteSleepCancellation. The Remote Sleep cancellation needs to be provided as soon as a vote is received to keep the network awake.⌋

Note: Specifically, this should not be delayed until the end of the repetition cycle boundary.

[SWS_FrNm_00230] ⌈If Remote Sleep Indication has been previously detected and the FrNm enters the Repeat Message State from the Normal Operation State or from the Ready Sleep State, then the NM shall notify the Generic NM Interface module that some nodes in the cluster are not ready to sleep anymore (the 'Remote Sleep Cancellation') by calling Nm_RemoteSleepCancellation.⌋

Note: In case of FrNmRemoteSleepIndTime = FrNmRepetitionCycle it is possible that NM calls RemoteSleepIndication() and RemoteSleepCancellation() within one Repetition Cycle. This does not affect any external behavior since NM does not change NM Mode. Therefore, the FrNmRemoteSleepIndTime should be greater than one FrNmRepetitionCycleto ensure that all possible votes are regarded.

[SWS_FrNm_00322] ⌈In order to support the NM Coordination algorithm in the NmIf module, an indication Nm_SynchronizationPoint provided to the NmIf module only at the repetition cycle boundaries, when the FrNm module is in Network Mode, and when the configuration parameter FrNmSynchronizationPointEnabled is set to TRUE.⌋

[SWS_FrNm_00323] ⌈FrNmSynchronizationPointEnabled is allowed to be set to true only when FrNmRemoteSleepIndicationEnabled is set to TRUE.⌋

7.8.3 Detection of Nodes (optional)

Nodes that have the Node Detection Feature enabled will send Identification Data (NM-Data PDU) on the bus (see chapter 7.6.2) if in the Repeat Message State (see chapter 7.2.3.1) or in the Normal Operation State. These data can be received by other nodes using the FrNm_GetNodeIdentifier function.

A node can identify (detect) nodes on the FlexRay Bus by repeatedly calling the previously mentioned function.

To ensure that all nodes (that are configured to do so) will send their Identification Data, the `FrNm_RepeatMessageRequest` can be used to bring all nodes to the Repeat Message State.

7.8.4 User data (optional)

[SWS_FrNm_00077]

Upstream requirements: RS_Nm_00150

⌈The support for user data in the FrNm module shall be configurable at pre-compile time by the configuration parameter `FrNmUserDataEnabled`.⌋

[SWS_FrNm_00447] ⌈When `FrNm_SetUserData` is called the FrNm module shall set the Network Management user data for the Network Management PDUs transmited next on the bus.⌋

[SWS_FrNm_00448] ⌈When `FrNm_GetUserData` is called FrNm module shall return the Network Management user data of the most recently received Network Management PDU.⌋

Alternatively to the usage of the FrNm APIs to set and get user data, FrNm may use the COM to retrieve its user data.

[SWS_FrNm_00446] ⌈If `FrNmComUserDataSupport` is enabled the API `FrNm_SetUserData` shall not be available.⌋

[SWS_FrNm_00364] ⌈If `FrNmComUserDataSupport` is set to TRUE and NM-PDU is not configured for triggered transmission in FrIf (`FrIfImmediate` = TRUE) the FrNm shall collect the NM User Data from the referenced NM I-PDU by calling "PduR_FrNmTriggerTransmit" and combine the user data with the further NM bytes each time before it requests the transmission of the corresponding NM message.⌋

[SWS_FrNm_00450] ⌈When PduR_FrNmTriggerTransmit() returns `E_NOT_OK`, the FrNm shall use the last transmitted value for NmUserData.⌋

Note: The transmission of outdated NM data can be avoided by not stopping the IPdu in COM used for NmUserData transmission.

[SWS_FrNm_00365] ⌈If FrNmComUserDataSupport is set to TRUE, the FrNm shall call PduR_FrNmTxConfirmation within the message transmission confirmation function FrNm_TxConfirmation called by the LSduR and with the result passed by LSduR.⌋

[SWS_FrNm_00366] ⌈If FrNmComUserDataSupport is set to TRUE the FrNm implementation shall provide an API FrNm_Transmit. FrNm_Transmit shall be an empty function returning E_OK at any time as an additional asynchronous message transmission is not possible on FlexRay.⌋

Hint: NM user data is handled by a SW-C like a "normal signal". User data consistency is guaranteed even if more than one SWC send user data. Relocation of a SW-C / VFB concept supported.

Hint: SW-C sends a signal containing user data, RTE propagates signal to COM, COM have to be configured that all user data signals of a channel are aggregated in one IPDU. Send type of that IPDU have to be configured "NONE" (-> COM never sends this signal)

Note: does not define a receive path for NM User Data.

Note: Missing abstraction of user data content and structure (length and structure of NM user data is OEM specific

7.8.5 Passive Node Configuration (optional)

Nodes that are configured as a "Passive" Mode shall participate in the cluster NM only in a passive way–unable to transmit any NM vote or NM data. Such nodes can only receive NM-Votes, but can not transmit votes that keep the cluster awake. Such a passive node can never change to the Normal Operation State (in the State Machine). It is a configuration error if the ComM calls the Nm_NetworkRequest for such a node.

[SWS_FrNm_00187]

Upstream requirements: RS_Nm_00150, RS_Nm_02511

⌈The Passive Mode Configuration shall be configurable at pre-compile time by the configuration parameter FrNmPassiveModeEnabled.⌋

[SWS_FrNm_00188] ⌈If Passive Mode Configuration is enabled [SWS_FrNm_00187], then the FrNm module shall not use the Remote Sleep Indication options [SWS_FrNm_00180].⌋

Note: Configuration parameter FrNmRemoteSleepIndicationEnabled shall be set to false.

7.8.6 NM PDU Rx Indication (optional)

Upper layers could use the optional PDU Rx Indication to detect NM activity (reception of a NM-Vote, NM-Data, or combined NM-Data and Vote PDU) on the FlexRay bus. However, since many NM PDUs could be received, especially when using the static segment for PDU transmission, it is not recommended to use this service.

[SWS_FrNm_00189] ⌈The NM PDU Reception indication shall be configurable at pre-compile time by the configuration parameter `FrNmPduRxIndicationEnabled`.⌋

[SWS_FrNm_00190] ⌈If NM PDU Reception indication is enabled [SWS_FrNm_00189], then the FrNm module shall call the function `Nm_PduRxIndication` at the successful reception of an NM-PDU.⌋

7.8.7 State change notification (optional)

[SWS_FrNm_00191] ⌈The inclusion of the optional state change notification service shall be configurable at pre-compile time by the configuration parameter `FrNmStateChangeIndicationEnabled`.⌋

[SWS_FrNm_00192] ⌈If the optional state change notification service is enabled [SWS_FrNm_00191], then the FrNm module shall notify all state changes except when entering Bus Sleep Mode during initialization to the NmIf module by calling `Nm_StateChangeNotification`.⌋

7.8.8 Dual FlexRay Channel PDU support (optional)

As described in more detail in 7.9, the FlexRay NM shall support the transmission and reception of PDU on both FlexRay channels (A and B). For the static segment this feature is supported by the FrIf, where for the dynamic segment this feature must be provided by the FlexRay NM itself, by sending (and receiving) two PDUs (one for FlexRay channel A and one for FlexRay channel B). The following requirements describe the required functionality.

[SWS_FrNm_00231] ⌈The dual FlexRay channel PDU support of the FlexRay NM shall be statically configurable at pre-compile time by the configuration parameter `FrNmDualChannelPduEnable`.⌋

[SWS_FrNm_00357] ⌈Vote changes for all nodes except one in a FlexRay cluster in certain dual-FlexRay channel topologies[3] shall be forbidden in the next-to-last repetition cycle before the Ready Sleep Counter expires, i.e., the NM votes transmitted by the Single-FlexRay channel nodes will be identical in both the last repetition cycle and the next-to-last repetition cycle if the Ready Sleep Counter were to expire. This shall be statically configurable at pre-compile time by the configuration parameter FrNmVotingNextToLastRepetitionCycleDisable.⌋

Rationale: One of the challenges with extending the current strategy to dual-FlexRay channel configurations is that votes transmitted by single-FlexRay channel nodes are not visible to single-FlexRay channel nodes on the opposite FlexRay channel. Avoidable race conditions might arise due to an inherent delay in forwarding the vote by the dual-FlexRay channel node for topologies where certain nodes are connected to only a single FlexRay channel and certain other nodes are connected to the other FlexRay channel. However, it can be readily mitigated with a slight modification where vote changes by nodes voting only on a single FlexRay channel are forbidden to change their vote in the next-to-last repetition cycle before the possible expiration of the Ready Sleep Counter.

7.8.9 Car Wakeup (optional)

[SWS_FrNm_00402] ⌈The position of the CWU bit in NM messages shall be defined by FrNmCarWakeUpBytePosition and FrNmCarWakeUpBitPosition.⌋

7.8.9.1 Rx Path

[SWS_FrNm_00410] ⌈If FrNmCwuRxEnabled is TRUE, FrNmCarWakeUpFilterEnabled is FALSE and NM PDU is received where the car wakeup bit is '1', FrNm shall call Nm_CarWakeUpIndication and perform the standard Rx indication handling.⌋

[SWS_FrNm_00411] ⌈If FrNm_GetPduData is called in the context of Nm_CarWakeUpIndication, FrNm shall return the PDU data of the PDU that causes the call of Nm_CarWakeUpIndication.⌋

Note: This is required to enable the ECU to identify detail about the sender of the car wakeup request.

[3]The challenging topology to manage effectively is the topology that has nodes attached only to Channel A, nodes attached only to Channel B, and nodes that are dual-channel.

[SWS_FrNm_00412] ⌈If FrNmCwuRxEnabled is TRUE, `FrNmCarWakeUpFilterEnabled` is TRUE and a Nm PDU is received where the car wakeup bit is ’1’ and the received Node ID is equal to FrNmCarWakeUpFilterNodeId the FrNm module shall call `Nm_CarWakeUpIndication` and perform the standard Rx Indication handling.⌋

Note: The car wakeup filter is necessary to realize sub gateways that only consider the car wakeup of the central Gateway to avoid wrong wakeups.

7.8.9.2 Tx Path

The transmission of the car wakeup bit shall be handled by the application using the NM user data mechanism provided by the FrNm module.

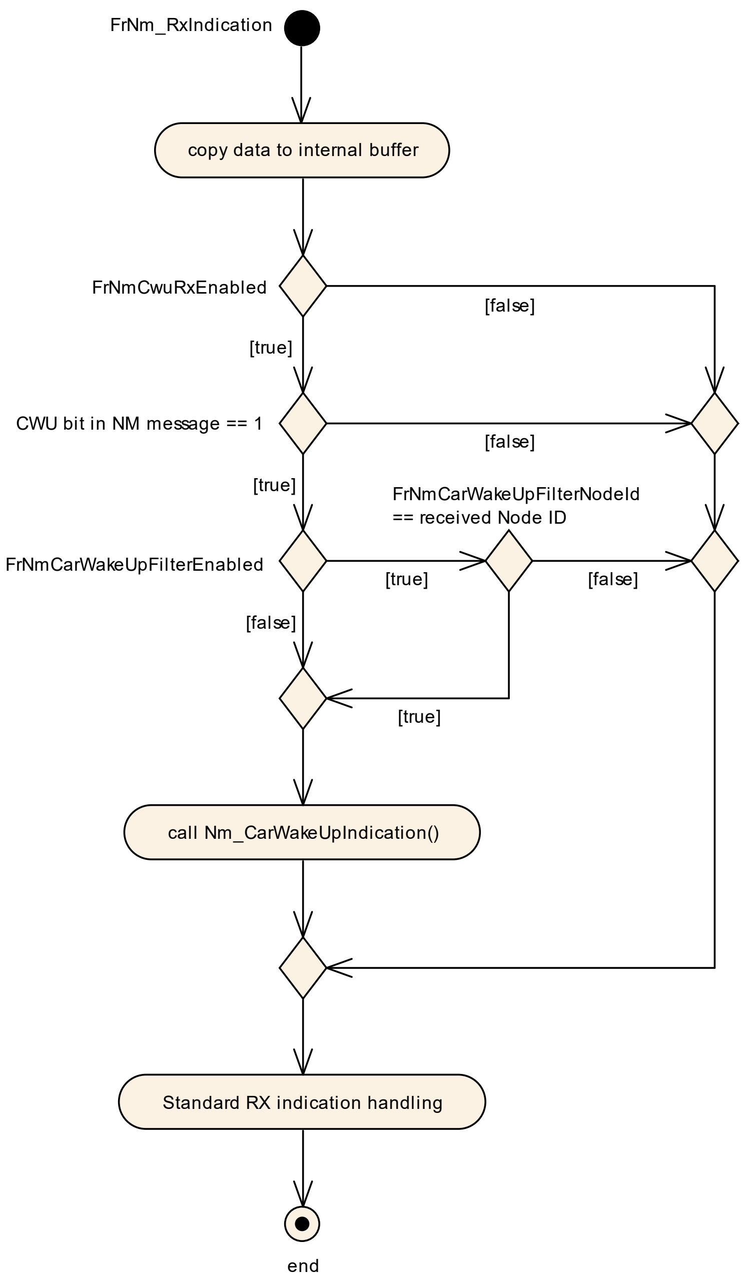
FrNm_RxIndication
copy data to internal buffer
FrNmCwuRxEnabled
[false]
[true]
CWU bit in NM message == 1
[false]
[true]
FrNmCarWakeUpFilterNodeId
== received Node ID
FrNmCarWakeUpFilterEnabled
[true]
[false]
[false]
[true]
call Nm_CarWakeUpIndication()
Standard RX indication handling
end

Figure 7.5: CarWakeUp indication handling

7.8.10 Coordinated Bus Shutdown (optional)

When having more than one coordinator connected to the same bus a special bit in the CBV, the NmCoordinatorSleepReady bit is used to indicate that the main coordinator requests to start shutdown sequence. The main functionality of the algorithm is described in the Nm module.

[SWS_FrNm_00396] ⌈If FrNm has entered Network Mode or called `Nm_CoordReadyToSleepCancellation` before it shall notify the Nm by calling `Nm_CoordReadyToSleepIndication` on the first reception of a NM message with the with the NmCoordinatorSleepReady bit (see CBV) set to 1.⌋

[SWS_FrNm_00451] ⌈If FrNm called `Nm_CoordReadyToSleepIndication` and is still in Network Mode it shall notify the NmIf by calling `Nm_CoordReadyToSleepCancellation` on the first reception of a NM Message with the NmCoordinatorSleepReady bit (See CBV) set to 0. The `Nm_CoordReadyToSleepCancellation` shall only be called in Network Mode.⌋

[SWS_FrNm_00398] ⌈The NMcoordinatorSleepReady bit in the Control Bit Vector shall be set via the API FrNm_SetSleepReadyBit.⌋

[SWS_FrNm_00397] ⌈The API `FrNm_SetSleepReadyBit()` and the feature "Coordinated Bus Shutdown" shall only be available if `FrNmCoordinatorSyncSupport` is set to TRUE.⌋

7.8.11 Extension for Partial Network (optional)

To reduce the power consumption of communication domains, it shall be possible to switch off the communication stack of ECUs during active bus communication. To control the shutdown and wakeup of these ECUs (cluster) in a standardized way, the AUTOSAR Network Management shall be used. Thus the AUTOSAR FrNm must be extended by the following Features.

An overview regarding the partial network cluster functionality can be found in document Guide to Mode Management [11].

Note: It is not possible to switch of only one ECU in a FlexRay cluster.

[SWS_FrNm_00405]

Upstream requirements: RS_Nm_02517

⌈The FrNm extensions for partial networking are enabled with the configuration switch `FrNmPnEnabled`.⌋

[SWS_FrNm_00404]

Upstream requirements: RS_Nm_02517, RS_Nm_02527

⌈FrNm shall be able to distinguish between a NM PDU which contains a PNC bit vector and a NM PDU which does not contain a PNC bit vector. This is indicated by the Partial Network Information Bit (PNI) in the CBV of the NM PDU. Meaning of the PNI bit:

- 0 == NM PDU contains no PNC bit vector

- 1 == NM PDU contains a PNC bit vector

⌋

Note: Each bit in PNC bit vector represents on particular PNC. The meaning of PNC bit is as follow:

- 0 == PNC is released
- 1 == PNC is requested

7.8.11.1 RX-Path

[SWS_FrNm_00406] ⌈If FrNmPnEnabled is FALSE, FrNm shall perform the standard FrNm RxIndication and the partial networking extensions shall be disabled.⌋

[SWS_FrNm_00407]

Upstream requirements: RS_Nm_02517

⌈If FrNmPnEnabled is TRUE and the PNI bit in the NM PDU provided by FrNm_RxIndication() has the value 0, FrNm shall perform the standard FrNm RxIndication and omitting the extensions for partial networking.⌋

[SWS_FrNm_00537]

Upstream requirements: RS_Nm_02517

⌈If FrNmPnEnabled is TRUE the PNI bit in the received NM-PDU is set to 1 and one of the following pre-conditions is valid:

- FrNmSynchronizedPncShutdownEnabled is set to FALSE
- FrNmSynchronizedPncShutdownEnabled is set to TRUE and the PNSR bit is set to 0

then the FrNm module shall extract the PNC bit vector from the received NM-PDU according to the partial network configuration (NmPncBitVectorOffset and NmPncBitVectorLength of the corresponding NM-channel) and forward the PNC bit vector by calling Nm_PncBitVectorRxIndication.⌋

Note: The PNSR bit shall be evaluated only if FrNmSynchronizedPncShutdownEnabled is set to TRUE.

Note: For FlexRay all Nm message are relevant independent of the content of a received PNC bit vector, because a single Nm Node cannot shutdown independent of the bus. Therefore the output value of "RelevantPncRequestDetectedPtr" (see function call of Nm_PncBitVectorRxIndication) has to be ignored

Example:

- Please note: control bit vector is mandatory and resides always in byte 0
- FrNmSourceNodeIdentifierEnabled = TRUE (reside in byte 1)
- NmPncBitVectorOffset = 4
- NmPncBitVectorLength = 4
- Calculated length of user data range = 2

Byte 2 and Byte 3 of the NM PDU contain user data and Byte 4 to Byte 7 of the NM PDU contain the PNC bit vector:

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
CBV	NID	User Data		PNC bit vector			
0x40	0x00	0xFF	0xFF	0x12	0x8E	0x80	0x01

For this example four NmPnFilterMaskBytes shall be defined. The values of the PN filter mask are used according to the partial network design e.g:

- NmPnFilterMaskByteIndex = 0 with NmPnFilterMaskByteValue = 0x01
- NmPnFilterMaskByteIndex = 1 with NmPnFilterMaskByteValue = 0x97
- NmPnFilterMaskByteIndex = 2 with NmPnFilterMaskByteValue = 0x00
- NmPnFilterMaskByteIndex = 3 with NmPnFilterMaskByteValue = 0x00

Note: The offset for the PNC bit vector is derived from the Nm module (NmPncBitVectorOffset). The PNC bit vector length is derived from the Nm module per NM-channel (NmPncBitVectorLength). The PN filter mask (NmPnFilterMaskByteIndex and NmPnFilterMaskByteValue) located and used in the Nm module.

[SWS_FrNm_00504]

Upstream requirements: RS_Nm_02544, RS_Nm_02548

⌈If FrNmSynchronizedPncShutdownEnabled is TRUE, the PNI bit in the received NM-PDU is 1, the PNSR bit in the received NM-PDU is 1 and the corresponding ComMChannel configured via FrNmComMNetworkHandleRef is actively coordinated (ComMPncGatewayType set to COMM_GATEWAY_TYPE_ACTIVE), then the FrNm module shall report the runtime error FRNM_E_INVALID_PN_SYNC_SHUTDOWN_REQUEST to the Default Error Tracer, ignore the PNSR bit and handle the PDU as usual NM PDU.⌋

Note: The handling should support the robustness of the PN regarding a synchronized shutdown handling when the NM of an ECU is malfunctioning.

[SWS_FrNm_00540]

Upstream requirements: RS_Nm_02544

⌈If FrNmSynchronizedPncShutdownEnabled is TRUE, the PNI bit in the received NM-PDU is set to 1 and the PNSR bit is set to 1, FrNm module shall extract the PNC bit vector from the received NM-PDU according to the partial network configuration (NmPncBitVectorOffset and NmPncBitVectorLength of the corresponding NM-channel) and forward the PNC bit vector by calling Nm_ForwardSynchronizedPnc-Shutdown.⌋

Note: PNSR Bit set to 1 is only possible if a synchronized PNC shutdown is requested. A synchronized PNC shutdown should be handled across the PN topology. Therefore, it is assumed that either all coordinators have the synchronized PNC shutdown enabled or all coordinators have the synchronized PNC shutdown disabled. A mixture of both would lead to an unsynchronized PNC shutdown, which has to be avoided.

7.8.11.2 TX-Path

[SWS_FrNm_00409]

Upstream requirements: RS_Nm_02517

⌈If FrNmPnEnabled is TRUE, FrNm shall set the value of the PNI bit to 1 in transmitted NM PDUs.⌋

Note: If partial networking is used on the ECU, this ECU always has to send Partial Network Information in its NM user data.

[SWS_FrNm_00452]

Upstream requirements: RS_Nm_02517

⌈If FrNmPnEnabled is FALSE, FrNm shall set the value of the PNI bit to 0 in transmitted NM PDUs.⌋

[SWS_FrNm_00562]

Upstream requirements: RS_Nm_02571

⌈If FrNmGlobalPnSupport is set to TRUE, the FrNm module shall store the latest PNC bit vector per NM-channel everytime the PNC bit vector has been fetched from the Nm modul via call of Nm_PncBitVectorTxIndication.⌋

[SWS_FrNm_00563]

Upstream requirements: RS_Nm_02571

⌈If FrNmGlobalPnSupport is set to TRUE, a NM-PDU has been transmitted on a NM-Channel and FrNm_TxConfirmation is called with result E_OK for this NM-PDU, then the FrNm shall forward the confirmation to Nm by calling Nm_PncBitVectorTx-Confirmation with the stored PNC bit vector (see [SWS_FrNm_00562]) for this NM-channel with result set to E_OK⌋

Note: The confirmation towards the Nm is always performed, independent of the reason for transmission of a NM-PDU (e.g. NM-PDU transmitted with the current requested PNCs or NM-PDU transmitted as PN shutdown message).

[SWS_FrNm_00564]

Upstream requirements: RS_Nm_02571

⌈If FrNmGlobalPnSupport is set to TRUE, a NM-PDU has been transmitted on a NM-Channel and FrNm_TxConfirmation is called with result E_NOT_OK or the transmission request for this NM-PDU was not accepted (LSduR_FrNmTransmit returned E_NOT_OK) for this NM-PDU, then the FrNm module shall forward the confirmation to Nm by calling Nm_PncBitVectorTxConfirmation with the stored PNC bit vector (see [SWS_FrNm_00562]) for this NM-Channel with result set to E_NOT_OK.⌋

Note: The call of Nm_PncBitVectorTxConfirmation with E_NOT_OK is used by the Nm module to perform the synchronized PNC shutdown handling if configured.

[SWS_FrNm_00565]

Upstream requirements: RS_Nm_02550, RS_Nm_02519

⌈If FrNmPnEnabled is TRUE and a NM-PDU has to be transmitted (either as NM-PDU with requested PNCs or as PN shutdown message), the FrNm module shall additionally fetch the PNC bit vector by calling Nm_PncBitVectorTxIndication and copy the PNC bit vector with respect to NmPncBitVectorOffset and NmPncBitVectorLength of the corresponding NM-channel to the NM-PDU before requesting the transmission of the NM-PDU.⌋

Note:

- The transmission of a NM-PDU has to consider user data if the usage of user data is configured. Please refer to 7.8.4 User data (optional).
- PNC bit vector is always fetched up front to a transmission request independent if immediate transmission in FrIf (FrIfImmediate) is set to TRUE or FALSE. This should ensure to re-start the PN reset timer of the affected PNC in the Nm on a transmission request.

7.8.11.3 Handling of Internal Requested Partial Network Clusters

All internal PNC requests are maintained by ComM. ComM forwards the aggregated internal PNC requests per channel as PNC bit vector to NmIf. This PNC bit vector carries the so-called "Internal Request Array". The FrNm has to retrieve the latest IRA from NmIf every time an NM-PDU is transmitted. NmIf provides the IRA information to FrNm and updates the PNC reset timer (each time a relevant PNC is transmitted, the PNC reset timer is re-started).

Note: For all configured NM-channel where `FrNmPnEnabled` is set TRUE, the FrNm will call `Nm_PncBitVectorTxIndication(<NM-channel>, <buffer to store the unfiltered PNC bit vector of aggregated internal PNC requests>)` (see [SWS_FrNm_00565], [SWS_FrNm_00569] and [SWS_FrNm_00572]) to indicate the transmission and to retrieve the current internal PNC requests as PNC bit vector with respect to the configured `NmPncBitVectorLength`. The FrNm will copy received internal PNC requests to the PNC bit vector bytes of the NM-PDU.

7.9 Schedule details

The following sections describe requirements for the scheduling of NM PDUs on the FlexRay bus - for both dynamic segment and static segment.

As mentioned in chapter 7.6, the FlexRay NM is configurable to transmit the NM-Vote and NM-Data in different PDUs. Therefore the FlexRay NM offers seven possibilities for the transmission of NM-messages. They are enumerated below and summarized in the subsequent table.

1. NM-Vote and NM Data transmitted within one PDU in static segment. The NM-Vote has to be realized as separate bit within the PDU.
2. NM-Vote and NM-Data transmitted within one PDU in dynamic segment. The presence (or non-presence) of the PDU corresponds to the NM-Vote
3. NM-Vote and NM-Data are transmitted in the static segment in separate PDUs. This alternative is not recommended - Alternative 1 should be used instead.
4. NM-Vote transmitted in static and NM-Data transmitted in dynamic segment.
5. NM-Vote is transmitted in dynamic and NM-Data is transmitted in static segment. This alternative is not recommended - Possibility 2 or 6 should be used instead.
6. NM-Vote and NM-Data are transmitted in dynamic segment in separate PDUs.
7. NM-Vote and a copy of the CBV are transmitted in the static segment (using the FlexRay NM Vector support) and NM-Data is transmitted in the dynamic segment (see chapter 7.6.2.3).

		FlexRay Segment	
		Static	Dynamic
PDU Type	NM-Vote	3 and 4	5 and 6

▽

△

	NM-Data	3 and 5	4,6 and 7
	NM-Vote and NM-Data	1	2
	Combined NM-Vote and CBV	7	-

Table 7.3: Summary of FlexRay PDU schedule alternatives

Note: If the NM-Vote is transmitted in the static segment of the FlexRay schedule (alternative 1, 3, 4 and 7), the usage of the HW NM-Vector support is possible.

Although every node can be configured independently to one of the above seven options it is beneficial to choose a "common" transmission alternative.

In addition to the above PDU transmission alternatives FlexRay offers two physical channels where data can be transmitted. Frames can be transmitted on FlexRay Channel A, FlexRay Channel B or on both FlexRay Channels. For the dynamic segment a transmission on both FlexRay channels requires two transmission- and receive-buffers as the FlexRay Protocol does not support shared transmission on FlexRay channel A and B in dynamic slots. In this special case the FlexRay NM can be configured to support a double transmit and receive (see 7.8.8).

[SWS_FrNm_00234] ⌈In the case when a combined NM-Vote and CBV is transmitted in static and dynamic segment, the FrNm module shall use the combined NM-Vote and CBV in the static part for the evaluation of the NM-Vote.⌋

7.9.1 FlexRay NM Cycle requirements

This section defines the schedule specific requirements that are required for a reliable transmission of FlexRay NM-messages.

[SWS_FrNm_00193] ⌈The FrNm module's integrator shall define the FlexRay NM Voting Cycle (`FrNmVotingCycle` configuration parameter) as the number of cycles needed to transmit–at least once–the NM-Vote of every node.⌋

Note: The value of the NM-Voting Cycle is typically determined by the number of cycles needed to transmit the votes of all nodes. For example, if only one slot in the dynamic segment is used, then the 3 nodes transmitting in the dynamic segment would require that the Voting Cycle is set to 4 - see also [SWS_FrNm_00195], [SWS_FrNm_00196] and Figure 10.9.

[SWS_FrNm_00194] ⌈The FrNm module's integrator shall define the FlexRay NM Data Cycle (`FrNmDataCycle` configuration parameter) as the number of cycles needed to transmit the NM-Data of every node at least once.⌋

Note: The value of the NM-Data Cycle is typically determined by the number of cycles needed to transmit the NM-Data of all nodes using the dynamic segment. For example, if only one slot in the dynamic segment is used and 5 nodes transmit their NM-Data in the dynamic segment, then the NM-Data Cycle is set to 8 - see also [SWS_FrNm_00195] and Figure 10.10.

[SWS_FrNm_00195] ⌈The FlexRay NM schedule specific cycle configuration parameters `FrNmVotingCycle`, `FrNmDataCycle` and `FrNmRepetitionCycle` shall have a value chosen from among the values 1, 2, 4, 8, 16, 32 or 64.⌋

Rationale: The restriction of the configuration values to the mentioned values is because FlexRay Cycle Multiplexing is used, which are only defined for these values.

[SWS_FrNm_00196] ⌈The FlexRay NM Repetition Cycle (`FrNmRepetitionCycle` configuration parameter) shall be an integer multiple (including 1) of the NM Voting Cycle (`FrNmVotingCycle`).⌋

Rationale: To improve the reliability of FlexRay NM, a number of repetitions of the NM Voting Cycle can be used. This will increase the chance that a "keep-awake" vote is not missed (e.g., due to a transmission error) – See Figure 10.9.

7.9.2 NM-Message scheduled requirements

There are no scheduling requirements for the FlexRay NM messages. Anyhow, it is highly recommended for frames containing NM information that are sent in the dynamic segment of the FlexRay Schedule to choose the first slots in the dynamic segment and to contain only NM information (NM-Vote and/or NM-Data) for the following reasons:

- Bandwidth (if no NM-messages are sent then less bandwidth is consumed)
- Flexibility (different nodes can transmit in the same slot but using different cycles)
- Predictability (it is guaranteed that the first slot is transmitted in each cycle)
- Determinism (transmission in the first slot always occurs at the same point in time in reference to the communication cycle start).

7.10 Transmission Error Handling

[SWS_FrNm_00035]

Upstream requirements: RS_Nm_00137

⌈If `FrNm_TxConfirmation` is called with result `E_NOT_OK` then FrNm shall call the function `Nm_TxTimeoutException`.⌋

Note: NM-message transmissions are expected to be sent whenever the API `LSduR_FrNmTransmit` is called.

Note: The phrase "NM cluster" must not be confused with the meaning of "FlexRay channel". The former is a logical unit, where the second is a physical bus interfaces line. FlexRay offers two channels per Communication controller, which are NOT independent, and can therefore not be seen as independent "NM clusters".

7.11 Error Classification

Section "Error Handling" of the document [2] "General Specification of Basic Software Modules" describes the error handling of the Basic Software in detail. Above all, it constitutes a classification scheme consisting of five error types which may occur in BSW modules.

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

7.11.1 Development Errors

[SWS_FrNm_00021] Definiton of development errors in module FrNm

Upstream requirements: SRS_BSW_00337, SRS_BSW_00331

⌈

Type of error	*Related error code*	*Error value*
API service used without module initialization	FRNM_E_UNINIT	0x01
API service called with invalid channel handle	FRNM_E_INVALID_CHANNEL	0x02
API service called with Invalid pointer	FRNM_E_PARAM_POINTER	0x03
API service called with invalid PDU ID as input parameter	FRNM_E_PDU_ID_INVALID	0x04
FrNm initialization has failed, e.g. selected configuration set doesn't exist	FRNM_E_INIT_FAILED	0x05

⌋

7.11.2 Runtime Errors

[SWS_FrNm_91002] Definiton of runtime errors in module FrNm ⌈

Type of error	*Related error code*	*Error value*
A NM message with PN Shutdown Request Bit was received on a channel that is actively coordinated by the ComM PNC Gateway	FRNM_E_INVALID_PN_SYNC_SHUTDOWN_REQUEST	0x20

⌋

7.11.3 Production Errors

There are no production errors.

7.11.4 Extended Production Errors

There are no extended production errors.

7.12 Version check

For details refer to the chapter 5.1.8 "Version Check" in SWS_BSWGeneral.

7.13 Send ActiveWakeupBit in CBV

To identify the ECU that awakes the Network, the FrNm shall send an information with every NM message whehter the ECU is responsible for the communication start or has been woken up by the communication. This information is pending until the ECU leaves the "Network Mode".

[SWS_FrNm_00297] ⌈If the FrNm performs a state change from Synchronize Mode to Network Mode and the previous state change from state Bus Sleep Mode to Synchronize was caused by a call of `FrNm_NetworkRequest()` (due to an active wakeup) and `FrNmActiveWakeupBitEnabled` is TRUE, the FrNm shall set the ActiveWakeupBit in the CBV.⌋

[SWS_FrNm_00298] ⌈If the FrNm module leaves the Network Mode and `FrNmActiveWakeupBitEnabled` is TRUE, the FrNm module shall clear the ActiveWakeupBit in the CBV.⌋

7.14 Security Events

The module does not report security events.

8 API specification

FlexRay NM API consists of services that are FlexRay specific and can be called as required.

[SWS_FrNm_00034]

Upstream requirements: RS_Nm_00045, RS_Nm_00154

⌈Each service other than `FrNm_Init` refers to one NM cluster only.⌋

Note: The phrase "NM cluster" must not be confused with the meaning of "FlexRay channel". The former is a logical unit, where the second is a physical bus interface line. FlexRay offers two channels per Communication controller and these FlexRay channels are NOT independent, and can therefore should not confused with an independent "NM cluster".

[SWS_FrNm_00056]

Upstream requirements: SRS_BSW_00369

⌈Development errors shall not be returned by API functions. In case of a development error the corresponding API function shall return `E_NOT_OK`, if applicable.⌋

[SWS_FrNm_00057]

Upstream requirements: SRS_BSW_00369

⌈Production errors shall not be returned by API functions. In case of a production error the corresponding API function shall return `E_NOT_OK`, when applicable.⌋

[SWS_FrNm_00051] ⌈If DET is enabled and the FlexRay NM API service is called with an invalid handle, then the corresponding function shall report `FRNM_E_INVALID_CHANNEL` error to the Default Error Tracer and it shall return `E_NOT_OK`.⌋

Note: The network handle is invalid if it is different from allowed configured values.

[SWS_FrNm_00543] ⌈When a Null pointer has been passed to a FrNm service, the called function shall not be executed and it shall return `E_NOT_OK` to the calling function if applicable. If development error detection is enabled (`FrNmDevErrorDetect` is set to TRUE) the corresponding error `FRNM_E_PARAM_POINTER` shall be reported to DET.⌋

8.1 Imported types

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

[SWS_FrNm_00235] Definition of imported datatypes of module FrNm ⌈

Module	**Header File**	**Imported Type**
Comtype	ComStack_Types.h	NetworkHandleType
	ComStack_Types.h	PduIdType
	ComStack_Types.h	PduInfoType
	ComStack_Types.h	PduLengthType
Nm	NmStack_types.h	Nm_ModeType
	NmStack_types.h	Nm_StateType
Std	Std_Types.h	Std_ReturnType
	Std_Types.h	Std_VersionInfoType

⌋

8.2 Type definitions

8.2.1 Generic NM Type Definitions

The FlexRay NM will use the type definitions as specified in Specification of Generic Network Management Interface [12] in chapter 8.2 Type definitions.

8.2.2 FlexRay NM specific Type Definitions

8.2.2.1 FrNm_ConfigType

[SWS_FrNm_00454] Definition of datatype FrNm_ConfigType ⌈

Name	FrNm_ConfigType
Kind	Structure
Description	Contains configuration parameters.
Available via	FrNm.h

⌋

For `FrNm_ConfigType` see chapter 10.4.

8.3 Function definitions

8.3.1 FrNm_Init

[SWS_FrNm_00236] Definition of API function FrNm_Init ⌈

Service Name	FrNm_Init	

Syntax	`void FrNm_Init (`
	`  const FrNm_ConfigType* nmConfigPtr`
	`)`	

Service ID [hex]	0x00	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	nmConfigPtr	Pointer to the selected configuration set.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Initializes the FlexRay NM and its internal state machine.	
Available via	FrNm.h	

⌋

[SWS_FrNm_00028]

Upstream requirements: SRS_BSW_00101

⌈The function `FrNm_Init` shall initialize the FrNm module.⌋

[SWS_FrNm_00030] ⌈The function `FrNm_Init` shall deactivate the periodic transmission of NM-Vote and NM-Data after the initialization of the FrNm module (see [SWS_FrNm_00100], [SWS_FrNm_00137]).⌋

[SWS_FrNm_00042] ⌈After the initialization of the FrNm module, the function `FrNm_Init` shall set the Reserved Bytes in the NM-Data and NM-Vote PDU to 0.⌋

[SWS_FrNm_00045] ⌈Within the initialization of the FrNm module the function `FrNm_Init` shall set the User Data bytes to FFh.⌋

Note: Please note that in case `FrNmComUserDataSupport` is enabled the initial values are for fail-safe only. They are overwritten before first transmission of NM message by values retrieved from Com.

8.3.2 FrNm_PassiveStartUp

[SWS_FrNm_00237] Definition of API function FrNm_PassiveStartUp ⌈

Service Name	FrNm_PassiveStartUp	
Syntax	`Std_ReturnType FrNm_PassiveStartUp ( NetworkHandleType NetworkHandle )`	
Service ID [hex]	0x01	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	NetworkHandle	Identification of the NM-channel
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	`E_OK`: No error `E_NOT_OK`: Start of network management has failed
Description	Initiates the Passive Startup of the FlexRay NM.	
Available via	FrNm.h	

⌋

[SWS_FrNm_00258] ⌈The function `FrNm_PassiveStartUp` shall initiate the Passive Startup of the FlexRay NM.⌋

[SWS_FrNm_00138] ⌈The function `FrNm_PassiveStartUp` shall trigger the transition from Bus-Sleep Mode to the Network Mode in Repeat Message State (via the Synchronize State).⌋

[SWS_FrNm_00260] ⌈If the current state is not Bus-Sleep Mode, the function `FrNm_PassiveStartUp` shall have no effect on the operation mode of the FrNm module and shall return `E_NOT_OK` (see [SWS_FrNm_00138]).⌋

8.3.3 FrNm_NetworkRequest

[SWS_FrNm_00238] Definition of API function FrNm_NetworkRequest ⌈

Service Name	FrNm_NetworkRequest
Syntax	`Std_ReturnType FrNm_NetworkRequest ( NetworkHandleType NetworkHandle )`
Service ID [hex]	0x02
Sync/Async	Asynchronous
Reentrancy	Non Reentrant

▽

△

Parameters (in)	NetworkHandle	Identification of the NM-channel
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	`E_OK`: No error `E_NOT_OK`: Requesting of bus communication has failed
Description	This function requests the network because the ECU needs to communicate on the bus. Network state shall be changed to 'requested'.	
Available via	FrNm.h	

⌋

[SWS_FrNm_00261] ⌈The function `FrNm_NetworkRequest` shall be only available if the configuration parameter `FrNmPassiveModeEnabled` is set to OFF.⌋

8.3.4 FrNm_NetworkRelease

[SWS_FrNm_00239] Definition of API function FrNm_NetworkRelease ⌈

Service Name	FrNm_NetworkRelease	

Syntax	`Std_ReturnType FrNm_NetworkRelease (`
	`  NetworkHandleType NetworkHandle`
	`)`	

Service ID [hex]	0x03	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	NetworkHandle	Identification of the NM-channel
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	`E_OK`: No error `E_NOT_OK`: Releasing of bus communication has failed
Description	This function releases the network because the ECU doesn't have to communicate on the bus. Network state shall be changed to 'released'.	
Available via	FrNm.h	

⌋

[SWS_FrNm_00262] ⌈The function `FrNm_NetworkRelease` shall be only available if the configuration parameter `FrNmPassiveModeEnabled` is set to OFF.⌋

8.3.5 FrNm_SetUserData

[SWS_FrNm_00240] Definition of API function FrNm_SetUserData ⌈

Service Name	FrNm_SetUserData	
Syntax	`Std_ReturnType FrNm_SetUserData ( NetworkHandleType NetworkHandle, const uint8* nmUserDataPtr )`	
Service ID [hex]	0x06	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	NetworkHandle	Identification of the NM-channel
	nmUserDataPtr	User data for the next transmitted NM message
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	`E_OK`: No error `E_NOT_OK`: Setting of user data has failed
Description	This function sets user data for NM-Data transmitted next on the bus.	
Available via	FrNm.h	

⌋

[SWS_FrNm_00043]

Upstream requirements: RS_Nm_02503

⌈If user data handling is enabled for the FrNm module, then the function `FrNm_SetUserData` shall set the user data.⌋

[SWS_FrNm_00263] ⌈The function `FrNm_SetUserData` shall be only available if the configuration parameter `FrNmUserDataEnabled` is set as ON (see [SWS_FrNm_00077]).⌋

8.3.6 FrNm_GetUserData

[SWS_FrNm_00241] Definition of API function FrNm_GetUserData ⌈

Service Name	FrNm_GetUserData
Syntax	`Std_ReturnType FrNm_GetUserData ( NetworkHandleType NetworkHandle, uint8* nmUserDataPtr )`
Service ID [hex]	0x07
Sync/Async	Synchronous

▽

△

Reentrancy	Reentrant	
Parameters (in)	NetworkHandle	Identification of the NM-channel
Parameters (inout)	None	
Parameters (out)	nmUserDataPtr	Pointer to the location where the user data from the last successfully received NM message shall be copied.
Return value	Std_ReturnType	`E_OK`: No error `E_NOT_OK`: Getting of user data has failed
Description	This function gets user data from the last successfully received NM message.	
Available via	FrNm.h	

⌋

[SWS_FrNm_00044]

Upstream requirements: RS_Nm_02504

⌈If user data handling is enabled for the FrNm module, then the function `FrNm_GetUserData` shall provide the user data from the last received NM-Data PDU.⌋

[SWS_FrNm_00264] ⌈The function `FrNm_GetUserData` shall be only available if the configuration parameter `FrNmUserDataEnabled` is set as ON (see [SWS_FrNm_00077]).⌋

8.3.7 FrNm_GetPduData

[SWS_FrNm_00242] Definition of API function FrNm_GetPduData ⌈

Service Name	FrNm_GetPduData	

Syntax	`Std_ReturnType FrNm_GetPduData (`
	`  NetworkHandleType NetworkHandle,`
	`  uint8* nmPduData`
	`)`	

Service ID [hex]	0x08	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	NetworkHandle	Identification of the NM-channel
Parameters (inout)	None	
Parameters (out)	nmPduData	Pointer where NM PDU shall be copied to.
Return value	Std_ReturnType	`E_OK`: No error `E_NOT_OK`: Getting of NM PDU Data has failed or is not configured for this network handle.
Description	Gets PDU data.	
Available via	FrNm.h	

⌋

8.3.8 FrNm_RepeatMessageRequest

[SWS_FrNm_00243] Definition of API function FrNm_RepeatMessageRequest ⌈

Service Name	FrNm_RepeatMessageRequest	
Syntax	`Std_ReturnType FrNm_RepeatMessageRequest ( NetworkHandleType NetworkHandle )`	
Service ID [hex]	0x09	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	NetworkHandle	Identification of the NM-channel
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	`E_OK`: No error `E_NOT_OK`: Setting of Repeat Message Request Bit has failed or is not configured for this network handle.
Description	This function causes a Repeat Message Request to be transmitted next on the bus.	
Available via	FrNm.h	

⌋

[SWS_FrNm_00226] ⌈If the FlexRay NM module's environment is calling the function `FrNm_RepeatMessageRequest` in "Bus Sleep Mode" or "Synchronize" no functionality shall be executed and `E_NOT_OK` shall be returned.⌋

8.3.9 FrNm_GetNodeIdentifier

[SWS_FrNm_00244] Definition of API function FrNm_GetNodeIdentifier ⌈

Service Name	FrNm_GetNodeIdentifier	
Syntax	`Std_ReturnType FrNm_GetNodeIdentifier ( NetworkHandleType NetworkHandle, uint8* nmNodeIdPtr )`	
Service ID [hex]	0x0a	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	NetworkHandle	Identification of the NM-channel
Parameters (inout)	None	
Parameters (out)	nmNodeIdPtr	Pointer to the location where the node identifier from the last successfully received NM-message shall be copied.
Return value	Std_ReturnType	`E_OK`: No error `E_NOT_OK`: Getting of the node identifier out of the last received NM-message has failed or is not configured for this network handle.

▽

△

Description	This function gets the node identifier from the last successfully received NM-message.
Available via	FrNm.h

⌋

[SWS_FrNm_00047]

Upstream requirements: RS_Nm_02506

⌈The function `FrNm_GetNodeIdentifier` shall provide the node identifier from the most recently received NM-message if `FrNmSourceNodeIdentifierEnabled` is set to TRUE.⌋

8.3.10 FrNm_GetLocalNodeIdentifier

[SWS_FrNm_00245] Definition of API function FrNm_GetLocalNodeIdentifier ⌈

Service Name	FrNm_GetLocalNodeIdentifier	
Syntax	`Std_ReturnType FrNm_GetLocalNodeIdentifier ( NetworkHandleType NetworkHandle, uint8* nmNodeIdPtr )`	
Service ID [hex]	0x0b	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	NetworkHandle	Identification of the NM-channel
Parameters (inout)	None	
Parameters (out)	nmNodeIdPtr	Pointer the location where the node identifier of the local node shall be copied.
Return value	Std_ReturnType	`E_OK`: No error `E_NOT_OK`: Getting of the node identifier of the local node has failed or is not configured for this network handle.
Description	This function gets the node identifier configured for the local node.	
Available via	FrNm.h	

⌋

[SWS_FrNm_00046] ⌈The function `FrNm_GetLocalNodeIdentifier` shall provide the node identifier configured for the local host node (`FrNmNodeId`) if `FrNmSourceNodeIdentifierEnabled` is set to TRUE.⌋

8.3.11 FrNm_RequestBusSynchronization

[SWS_FrNm_00246] Definition of API function FrNm_RequestBusSynchronization ⌈

Service Name	FrNm_RequestBusSynchronization	
Syntax	`Std_ReturnType FrNm_RequestBusSynchronization ( NetworkHandleType NetworkHandle )`	
Service ID [hex]	0xc0	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	NetworkHandle	Identification of the NM-Cluster
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	`E_OK`: No error `E_NOT_OK`: Function failed
Description	This function has no functionality - the service is provided only to be compatible to future extensions and to be compatible to the CAN-NM interface.	
Available via	FrNm.h	

⌋

[SWS_FrNm_00174] ⌈The FrNm module shall support the function `FrNm_RequestBusSynchronization` by returning `E_OK` without executing any functionality.⌋

Rationale: As the FlexRay Bus is a synchronous bus, the FrNm cannot influence the bus synchronization. This functionality is handled by the FlexRay Controller and FlexRay Driver. Since this function might be used in future extensions (e.g., Gateways) this function shall remain available.

[SWS_FrNm_00269] ⌈The function `FrNm_RequestBusSynchronization` shall be only available if the configuration parameter `FrNmBusSynchronizationEnabled` is set as ON.⌋

8.3.12 FrNm_CheckRemoteSleepIndication

[SWS_FrNm_00247] Definition of API function FrNm_CheckRemoteSleepIndication ⌈

Service Name	FrNm_CheckRemoteSleepIndication	

Syntax	`Std_ReturnType FrNm_CheckRemoteSleepIndication (`
	`  NetworkHandleType NetworkHandle,`
	`  boolean* nmRemoteSleepIndPtr`
	`)`	

Service ID [hex]	0x0d	
Sync/Async	Synchronous	
Reentrancy	Reentrant (but not for the same NM-Channel)	
Parameters (in)	NetworkHandle	Identification of the NM-channel
Parameters (inout)	None	
Parameters (out)	nmRemoteSleepIndPtr	Pointer to the location where the check result of remote sleep indication shall be copied.
Return value	Std_ReturnType	`E_OK`: No error `E_NOT_OK`: Checking of remote sleep indication bits has failed
Description	This function checks if remote sleep indication has taken place or not.	
Available via	FrNm.h	

⌋

[SWS_FrNm_00270] ⌈The FrNm module and the NmIf module shall be initialized correctly before the FrNm module's environment is calling the function `FrNm_CheckRemoteSleepIndication`.⌋

[SWS_FrNm_00185]

Upstream requirements: RS_Nm_02509

⌈The FlexRay NM function `FrNm_CheckRemoteSleepIndication` shall provide the information about current status of Remote Sleep Indication (i.e., whether a Remote Sleep Indication has already been detected or not).⌋

[SWS_FrNm_00271] ⌈The function `FrNm_CheckRemoteSleepIndication` shall be only available if the configuration parameter `FrNmRemoteSleepIndicationEnabled` is set as ON.⌋

8.3.13 FrNm_GetState

[SWS_FrNm_00248] Definition of API function FrNm_GetState ⌈

Service Name	FrNm_GetState	

Syntax	`Std_ReturnType FrNm_GetState (`
	`  NetworkHandleType NetworkHandle,`
	`  Nm_StateType* nmStatePtr,`
	`  Nm_ModeType* nmModePtr`
	`)`	

Service ID [hex]	0x0e	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	NetworkHandle	Identification of the NM-channel
Parameters (inout)	None	
Parameters (out)	nmStatePtr	Pointer to the location where the state of the network management shall be copied.
	nmModePtr	Pointer to the location where the mode of the network management shall be copied.
Return value	Std_ReturnType	`E_OK`: No error `E_NOT_OK`: Getting of NM state has failed
Description	This function returns the state and the mode of the network management.	
Available via	FrNm.h	

⌋

[SWS_FrNm_00104]

Upstream requirements: RS_Nm_00050

⌈The function `FrNm_GetState` shall provide consistent information about the current state and the current mode of the NM state machine.⌋

Note: Consistency between the provided values and the current values of the state and mode should be ensured.

8.3.14 FrNm_GetVersionInfo

[SWS_FrNm_00249] Definition of API function FrNm_GetVersionInfo ⌈

Service Name	FrNm_GetVersionInfo

Syntax	`void FrNm_GetVersionInfo (`
	`  Std_VersionInfoType* NmVerInfoPtr`
	`)`

Service ID [hex]	0x0f
Sync/Async	Synchronous
Reentrancy	Reentrant

▽

△

Parameters (in)	None	
Parameters (inout)	None	
Parameters (out)	NmVerInfoPtr	Pointer to the location where the version information of this module shall be copied.
Return value	None	
Description	Returns the version information.	
Available via	FrNm.h	

⌋

8.3.15 FrNm_StartupError

[SWS_FrNm_00393] Definition of API function FrNm_StartupError ⌈

Service Name	FrNm_StartupError	

Syntax	`void FrNm_StartupError (`
	`  NetworkHandleType NetworkHandle`
	`)`	

Service ID [hex]	0x10	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	NetworkHandle	Identification of the NM-channel
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	This function is called by the FrSM when synchronization of the FlexRay cluster could not be achieved.	
Available via	FrNm.h	

⌋

8.3.16 FrNm_Transmit

[SWS_FrNm_00359] Definition of API function FrNm_Transmit ⌈

Service Name	FrNm_Transmit

Syntax	`Std_ReturnType FrNm_Transmit (`
	`  PduIdType TxPduId,`
	`  const PduInfoType* PduInfoPtr`
	`)`

Service ID [hex]	0x49
Sync/Async	Synchronous

▽

△

Reentrancy	Reentrant for different PduIds. Non reentrant for the same PduId.	
Parameters (in)	TxPduId	Identifier of the PDU to be transmitted
	PduInfoPtr	Length of and pointer to the PDU data and pointer to MetaData.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	`E_OK`: Transmit request has been accepted. `E_NOT_OK`: Transmit request has not been accepted.
Description	Requests transmission of a PDU.	
Available via	FrNm.h	

⌋

Note: The FlexRay NM shall realize a Transmit function only if the node is configured as an active node, i.e., only if `FrNmPassiveModeEnabled` (configuration parameter) is set as OFF

8.3.17 FrNm_EnableCommunication

[SWS_FrNm_00387] Definition of API function FrNm_EnableCommunication

Upstream requirements: RS_Nm_02512

⌈

Service Name	FrNm_EnableCommunication	

Syntax	`Std_ReturnType FrNm_EnableCommunication (`
	`  NetworkHandleType nmChannelHandle`
	`)`	

Service ID [hex]	0x05	
Sync/Async	Asynchronous	
Reentrancy	Reentrant (but not for the same NM-channel)	
Parameters (in)	nmChannelHandle	Identification of the NM-channel
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	`E_OK`: No error `E_NOT_OK`: Enabling of NM PDU transmission ability has failed
Description	Enable the NM PDU transmission ability due to a ISO14229 Communication Control (28hex) service	
Available via	FrNm.h	

⌋

[SWS_FrNm_00388] ⌈The service `FrNm_EnableCommunication` shall return `E_NOT_OK`, if the current mode is not Network Mode.⌋

[SWS_FrNm_00389] ⌈If the module operates in passive mode (`FrNmPassiveModeEnabled`), then the service `FrNm_EnableCommunication` shall have no effects and directly return `E_NOT_OK`.⌋

8.3.18 FrNm_DisableCommunication

[SWS_FrNm_00390] Definition of API function FrNm_DisableCommunication

Upstream requirements: RS_Nm_02512

⌈

Service Name	FrNm_DisableCommunication	
Syntax	`Std_ReturnType FrNm_DisableCommunication ( NetworkHandleType nmChannelHandle )`	
Service ID [hex]	0x0c	
Sync/Async	Asynchronous	
Reentrancy	Reentrant (but not for the same NM-channel)	
Parameters (in)	nmChannelHandle	Identification of the NM-channel
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	`E_OK`: No error `E_NOT_OK`: Disabling of NM PDU transmission ability has failed
Description	Disable the NM PDU transmission ability due to a ISO14229 Communication Control (28hex) service	
Available via	FrNm.h	

⌋

[SWS_FrNm_00391] ⌈The service `FrNm_DisableCommunication` shall return `E_NOT_OK` if the current mode is not Network Mode.⌋

[SWS_FrNm_00392] ⌈If the module operates in passive mode (`FrNmPassiveModeEnabled`), then the service `FrNm_DisableCommunication` shall have no effect and directly return `E_NOT_OK`.⌋

8.3.19 FrNm_SetSleepReadyBit

[SWS_FrNm_91001] Definition of API function FrNm_SetSleepReadyBit ⌈

Service Name	FrNm_SetSleepReadyBit	
Syntax	`Std_ReturnType FrNm_SetSleepReadyBit ( NetworkHandleType nmChannelHandle, boolean nmSleepReadyBit )`	
Service ID [hex]	0x12	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	nmChannelHandle	Identification of the NM-channel
	nmSleepReadyBit	Value written to ReadySleep Bit in CBV
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	`E_OK`: No error `E_NOT_OK`: Writing of remote sleep indication bit has failed
Description	Set the NM Coordinator Sleep Ready bit in the Control Bit Vector	
Available via	FrNm.h	

⌋

8.3.20 FrNm_PnLearningRequest

[SWS_FrNm_91005] Definition of API function FrNm_PnLearningRequest

Status: DRAFT

⌈

Service Name	FrNm_PnLearningRequest (draft)	
Syntax	`Std_ReturnType FrNm_PnLearningRequest ( NetworkHandleType NetworkHandle )`	
Service ID [hex]	0xf1	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	NetworkHandle	Identification of the NM-channel
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	`E_OK`: No error `E_NOT_OK`: PN Learning Request has failed or is not configured for this network handle.

▽

△

Description	Set Repeat Message Request Bit and Partial Network Learning Bit for NM messages transmitted next on the bus. This will force all nodes to enter the PNC Learning Phase. This is needed for the optional Dynamic PNC-to-channel-mapping feature. **Tags:** atp.Status=draft	
Available via	FrNm.h	

⌋

[SWS_FrNm_00495] ⌈If the function `FrNm_PnLearningRequest` is called in Synchronize Mode or Bus Sleep Mode no functionality shall be executed and `E_NOT_OK` shall be returned.⌋

[SWS_FrNm_00496] ⌈The function `FrNm_PnLearningRequest` shall only be available if `FrNmDynamicPncToChannelMappingSupport` is set to TRUE.⌋

8.3.21 FrNm_ActivateTxPnShutdownMsg

[SWS_FrNm_91006] Definition of API function FrNm_ActivateTxPnShutdownMsg

Upstream requirements: RS_Nm_02572

⌈

Service Name	FrNm_ActivateTxPnShutdownMsg	

Syntax	`Std_ReturnType FrNm_ActivateTxPnShutdownMsg (`
	`  NetworkHandleType nmChannelHandle`
	`)`	

Service ID [hex]	0xf4	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different nmChannelHandle. Non reentrant for the same nmChannelHandle.	
Parameters (in)	nmChannelHandle	Identifier of the NM-Channel where the PNC shutdown process is started.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	`E_OK`:Request has been accepted. `E_NOT_OK`: Request has not been accepted.
Description	NM indicate to activate the transmission of PN shutdown messages on the given NM-Channel. This results in transmission of a NM-PDU with PNSR bit set to 1 (PN shutdown message).	
Available via	FrNm.h	

⌋

[SWS_FrNm_00568]

Upstream requirements: RS_Nm_02572

⌈If `FrNmSynchronizedPncShutdownEnabled` is set to TRUE the FrNm implementation shall provide the API `FrNm_ActivateTxPnShutdownMsg`.⌋

[SWS_FrNm_00569]

Upstream requirements: RS_Nm_02572

⌈If FrNmSynchronizedPncShutdownEnabled is set to TRUE and FrNm_ActivateTxPnShutdownMsg is called with a valid NM-Channel (nmChannelHandle), then the FrNm module shall set the PNSR bit in the CBV to 1 on the given NM-channel and return with E_OK.⌋

8.3.22 FrNm_DeactivateTxPnShutdownMsg

[SWS_FrNm_91007] Definition of API function FrNm_DeactivateTxPnShutdownMsg

Upstream requirements: RS_Nm_02572

⌈

Service Name	FrNm_DeactivateTxPnShutdownMsg	
Syntax	`Std_ReturnType FrNm_DeactivateTxPnShutdownMsg ( NetworkHandleType nmChannelHandle )`	
Service ID [hex]	0xf5	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different nmChannelHandle. Non reentrant for the same nmChannelHandle.	
Parameters (in)	nmChannelHandle	Identifier of the NM-Channel where the PNC shutdown process is stopped.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK:Request has been accepted. E_NOT_OK: Request has not been accepted.
Description	NM indicate to deactive the transmission of PN shutdown messages on the given NM-Channel. This result in transmission of a usual NM-PDUs with PNSR bit set to 0.	
Available via	FrNm.h	

⌋

[SWS_FrNm_00570]

Upstream requirements: RS_Nm_02572

⌈If FrNmSynchronizedPncShutdownEnabled is set to TRUE the FrNm implementation shall provide the API FrNm_DeactivateTxPnShutdownMsg.⌋

[SWS_FrNm_00572]

Upstream requirements: RS_Nm_02572

⌈If FrNmSynchronizedPncShutdownEnabled is set to TRUE and FrNm_DeactivateTxPnShutdownMsg is called with a valid NM-Channel (nmChannelHandle), then the FrNm module shall set PNSR bit in the CBV to 0 on the given NM-channel and return with `E_OK`.⌋

8.4 Callback notifications

This is a list of functions provided for other modules.

8.4.1 FrNm_RxIndication

[SWS_FrNm_00251] Definition of callback function FrNm_RxIndication ⌈

Service Name	FrNm_RxIndication	

Syntax	`void FrNm_RxIndication (`
	`  PduIdType RxPduId,`
	`  const PduInfoType* PduInfoPtr`
	`)`	

Service ID [hex]	0x42	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different PduIds. Non reentrant for the same PduId.	
Parameters (in)	RxPduId	ID of the received PDU.
	PduInfoPtr	Contains the length (SduLength) of the received PDU, a pointer to a buffer (SduDataPtr) containing the PDU, and the MetaData related to this PDU.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Indication of a received PDU from a lower layer communication interface module.	
Available via	FrNm.h	

⌋

[SWS_FrNm_00013] ⌈The function `FrNm_RxIndication` shall handle the data from an NM-message–this means that the function shall copy the received FlexRay NM PDU and store it locally associated with the received FlexRay NM PDU ID.⌋

[SWS_FrNm_00276] ⌈The FrIf module, the LSduR module, and the FrNm module must be initialized correctly before the FrNm module's environment calls the function `FrNm_RxIndication`.⌋

The function `FrNm_RxIndication` might be called by the FrNm module's environment in an interrupt context.

8.4.2 FrNm_TriggerTransmit

[SWS_FrNm_00252] Definition of callback function FrNm_TriggerTransmit ⌈

Service Name	FrNm_TriggerTransmit	
Syntax	`Std_ReturnType FrNm_TriggerTransmit ( PduIdType TxPduId, PduInfoType* PduInfoPtr )`	
Service ID [hex]	0x41	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different PduIds. Non reentrant for the same PduId.	
Parameters (in)	TxPduId	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 SduLengh. 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.	
Available via	FrNm.h	

⌋

Note: The PNC bit vector is not updated within the call of `FrNm_TriggerTransmit` but upfront of each NM message transmission request (see [SWS_FrNm_00565]). This ensure a common handling independent of `FrIfImmediate` setting (TRUE or FALSE).

[SWS_FrNm_00549] ⌈If `FrNm_TriggerTransmit` is called and `FrNmComUserDataSupport` is enabled, the FrNm shall collect the NM User Data from the referenced NM I-PDU by calling `PduR_FrNmTriggerTransmit` and copy the data to the user data range of the NM-PDU.⌋

[SWS_FrNm_00277] ⌈The FrIf module, the LSduR module, and the FrNm module shall be initialized correctly before the FrNm module's environment calls the function `FrNm_TriggerTransmit`.⌋

Note: The function `FrNm_TriggerTransmit` might be called by the FrIf via LSduR in an interrupt context.

8.4.3 FrNm_TxConfirmation

[SWS_FrNm_00460] Definition of callback function FrNm_TxConfirmation ⌈

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

⌋

8.5 Scheduled functions

These functions are directly called by Basic Software Scheduler. The following functions shall have no return value and no parameter. All functions shall be non reentrant.

8.5.1 FrNm_MainFunction_<FrNmChannel.ShortName>

[SWS_FrNm_00255] Definition of scheduled function FrNm_MainFunction_<FrNmChannel.ShortName>

Upstream requirements: SRS_BSW_00432

⌈

Service Name	FrNm_MainFunction_<FrNmChannel.ShortName>
Syntax	`void FrNm_MainFunction_<FrNmChannel.ShortName> ( void )`
Service ID [hex]	0xf0
Description	Main function of FlexRay NM.
Available via	SchM_FrNm.h

⌋

This cyclically executed API service of the FlexRay Network Management serves the purposes of maintenance of the FrNm State Machine (see 7.4). Please refer to chapter 7.2.

This API service of the FlexRay Network Management is called cyclically from a task body provided by the BSW Scheduler.

Since the duration of a FlexRay Cycle may be different for two different Clusters, the calling period of this API service shall be configurable independently for each Cluster at system configuration time.

[SWS_FrNm_00283] ⌈There shall be one dedicated FlexRay NM Main Function for each NM cluster. The API names are therefore:

`FrNm_MainFunction_<FrNmChannel.ShortName>()`

where <FrNmChannel.ShortName> is the Short Name of the corresponding FlexRay NM channel.⌋

[SWS_FrNm_00449]

Upstream requirements: SRS_BSW_00432

⌈It shall be possible to perform an independent processing of transmission- and reception-handling inside the main function according to [SRS_BSW_00432].⌋

[SWS_FrNm_00344] ⌈The Service ID of `FrNm_MainFunction_<FrNmChannel.ShortName>` shall be 0xf0 + NetworkHandle.⌋

8.6 Expected interfaces

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

8.6.1 Mandatory interfaces

Note: This section defines all interfaces, which are required to fulfill the core functionality of the module.

[SWS_FrNm_00256] Definition of mandatory interfaces required by module FrNm
⌈

API Function	***Header File***	***Description***
FrIf_GetGlobalTime	FrIf.h	Wraps the FlexRay Driver API function Fr_GetGlobalTime(). Important Note: FrIf_GetGlobalTime may be called within an exclusive area.
LSduR_FrNmTransmit (draft)	LsduR_FrNm.h	Requests transmission of a PDU.
Nm_BusSleepMode	Nm.h	Notification that the network management has entered Bus-Sleep Mode.
Nm_NetworkMode	Nm.h	Notification that the network management has entered Network Mode.
Nm_SynchronizeMode	Nm.h	Notification that the network management has entered Synchronize Mode.
Nm_TxTimeoutException	Nm.h	Service to indicate that an attempt to send an NM message failed.

⌋

Note: The Generic NM Interface is currently seen as thin adaptation layer (e.g. implemented as c-macros) which will be used to interface to the ComM. See [12].

8.6.2 Optional interfaces

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

[SWS_FrNm_00257] Definition of optional interfaces requested by module FrNm
⌈

API Function	***Header File***	***Description***
Det_ReportError	Det.h	Service to report development errors.
FrIf_GetNmVector	FrIf.h	Derives the FlexRay NM Vector.
Nm_CarWakeUpIndication	Nm.h	This function is called by a <Bus>Nm to indicate reception of a CWU request.
Nm_CoordReadyToSleepCancellation	Nm.h	Cancels an indication, when the NM Coordinator Sleep Ready bit in the Control Bit Vector is set back to 0.
Nm_CoordReadyToSleepIndication	Nm.h	Sets an indication, when the NM Coordinator Sleep Ready bit in the Control Bit Vector is set
Nm_ForwardSynchronizedPnc Shutdown	Nm.h	Notification that the network management has received a PN shutdown message on a particular NM-channel. This is used to grant a nearly synchronized PNC shutdown across the entire PN topology.

▽

△

API Function	Header File	Description
Nm_NetworkStartIndication	Nm.h	Notification that a NM-message has been received in the Bus-Sleep Mode, what indicates that some nodes in the network have already entered the Network Mode.
Nm_PduRxIndication	Nm.h	Notification that a NM message has been received.
Nm_PncBitVectorRxIndication	Nm.h	Indication that a bus specific network management has received a NM message on a particular NM-channel that contain a PNC bit vector. This is used to aggregate the external PNC requests. The function evaluate if a relevant PNC request (PNC bit set to '1') is available in the given PNC bit vector. If a relevant PNC request is available (PNC bit passes the PNC bit vector filter), then the RelevantPnc RequestDetectedPtr refers to a boolean with value set to TRUE. Otherwise refer to booelan with value set to FALSE. RelevantPncRequestDetectedPtr is evaluated by the callee <Bus>Nm module to qualify the further processing of the received NM-PDU.
Nm_PncBitVectorTxConfirmation	Nm.h	Function called by <Bus>Nms to confirm the state of the transmission for the given PNC bit vector on the given NM-Channel.
Nm_PncBitVectorTxIndication	Nm.h	Function called by <Bus>Nms to request the aggregated internal PNC requests for transmission within the Nm message.
Nm_RemoteSleepCancellation	Nm.h	Notification that the network management has detected that not all other nodes on the network are longer ready to enter Bus-Sleep Mode.
Nm_RemoteSleepIndication	Nm.h	Notification that the network management has detected that all other nodes on the network are ready to enter Bus-Sleep Mode.
Nm_RepeatMessageIndication	Nm.h	Service to indicate that an NM message with set Repeat Message Re- quest Bit has been received. This is needed for node detection and the Dynamic PNC-to-channel-mapping feature.
Nm_StateChangeNotification	Nm.h	Notification that the state of the lower layer <Bus>Nm has changed.
Nm_SynchronizationPoint	Nm.h	Notification to the NM Coordinator functionality that this is a suitable point in time to initiate the coordinated shutdown on.
PduR_FrNmTriggerTransmit	PduR_FrNm.h	Within this API, the upper layer module (called module) shall check whether the available data fits into the buffer size reported by PduInfoPtr->Sdu Length. 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->Sdu Length. If not, it returns E_NOT_OK without changing PduInfoPtr.
PduR_FrNmTxConfirmation	PduR_FrNm.h	The lower layer communication interface module confirms the transmission of a PDU, or the failure to transmit a PDU.

⌋

The usage of certain external APIs by FlexRay network Manangement is defined by configuration:

- `FrIf_GetNmVector` Configured by configuration parameter `FrIfGetNmVectorSupport`

- Nm_PduRxIndication Configured if FrNmPduRxIndicationEnabled is configured (see [ECUC_FrNm_00046])
- Nm_RemoteSleepCancellation Configured by FrNmRemoteSleepIndicationEnabled (see [ECUC_FrNm_00044])
- Nm_RemoteSleepIndication Configured by FrNmRemoteSleepIndicationEnabled (see [ECUC_FrNm_00044])
- Nm_StateChangeNotification Configured by FrNmStateChangeIndicationEnabled (see [ECUC_FrNm_00047])
- Nm_SynchronizationPoint Configured by FrNmSynchronizationPointEnabled (see [ECUC_FrNm_00021])

8.6.3 Configurable interfaces

In this section, all interfaces are listed where the target function could be configured. The target function is usually a callback function. The names of this kind of interfaces are not fixed because they are configurable.

8.6.3.1 NetworkStartIndication

[SWS_FrNm_00470] Definition of configurable interface <Nm_NetworkStartIndication> ⌈

Service Name	<Nm_NetworkStartIndication>	

Syntax	`void <Nm_NetworkStartIndication> (`
	`  NetworkHandleType nmNetworkHandle`
	`)`	

Sync/Async	Synchronous	
Reentrancy	Reentrant for different FrNm Clusters. Non reentrant for the same FrNm Cluster	
Parameters (in)	nmNetworkHandle	Identification of the NM-Cluster
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	This API service of an upper layer Nm is called by the FlexRay Network Management to indicate to this upper layer BSW module that a NM-message was successfully received and the FrNm is in the Bus-Sleep Mode.	
Available via	<none>	

⌋

8.6.3.2 PduRxIndication

[SWS_FrNm_00471] Definition of configurable interface <Nm_PduRxIndication> ⌈

Service Name	<Nm_PduRxIndication>	

Syntax	`void <Nm_PduRxIndication> (`
	`  NetworkHandleType nmNetworkHandle`
	`)`	

Sync/Async	Synchronous	
Reentrancy	Reentrant for different FrNm Clusters. Non reentrant for the same FrNm Cluster	
Parameters (in)	nmNetworkHandle	Identification of the NM-Cluster
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	This API service of an upper NM is called by the FlexRay Network Management to indicate to this upper layer BSW module Nm that a NM-message was successfully received (see FRNM190) This function is only available on the definition of _PDU_RX_INDICATION_ENABLED	
Available via	<none>	

⌋

8.6.3.3 RemoteSleepCancellation

[SWS_FrNm_00472] Definition of configurable interface <UL_RemoteSleepCancellation> ⌈

Service Name	<UL_RemoteSleepCancellation>	

Syntax	`void <UL_RemoteSleepCancellation> (`
	`  NetworkHandleType nmNetworkHandle`
	`)`	

Sync/Async	Synchronous	
Reentrancy	Reentrant for different FrNm Clusters. Non reentrant for the same FrNm Cluster	
Parameters (in)	nmNetworkHandle	Identification of the NM-Cluster
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	This API service of an upper layer BSW module <UL> (e.g. ComM) is called by the FlexRay Network Management to indicate to this upper layer BSW module that the "Remote Sleep" has been cancelled as an NM-message with an indication to keep the bus awake has been received and the Remote Sleep indication has been previously detected .	
Available via	<none>	

⌋

8.6.3.4 RemoteSleepIndication

[SWS_FrNm_00473] Definition of configurable interface <UL_RemoteSleepIndication> ⌈

Service Name	<UL_RemoteSleepIndication>	

Syntax	`void <UL_RemoteSleepIndication> (`
	`  NetworkHandleType nmNetworkHandle`
	`)`	

Sync/Async	Synchronous	
Reentrancy	Reentrant for different FrNm Clusters. Non reentrant for the same FrNm Cluster	
Parameters (in)	nmNetworkHandle	Identification of the NM-Cluster
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	This API service of an upper layer BSW module <UL> (e.g. ComM) is called by the FlexRay Network Management to indicate to this upper layer BSW module that all other nodes in the cluster are ready to sleep (the 'Remote Sleep Indication'), as no NM-messages with an indication to keep the bus awake are received in the Normal Operation State for a configurable amount of time determined by the FRNM_REMOTE_SLEEP_IND_TIME (configuration parameter).	
Available via	<none>	

⌋

8.6.3.5 TransmissionTimeoutException

[SWS_FrNm_00474] Definition of configurable interface <UL_TransmissionTimeoutException> ⌈

Service Name	<UL_TransmissionTimeoutException>	

Syntax	`void <UL_TransmissionTimeoutException> (`
	`  NetworkHandleType nmNetworkHandle`
	`)`	

Sync/Async	Synchronous	
Reentrancy	Reentrant for different FrNm Clusters. Non reentrant for the same FrNm Cluster	
Parameters (in)	nmNetworkHandle	Identification of the NM-Cluster
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	This API service of an upper layer BSW module <UL> is called by the FlexRay Network Management when FrIf calls FrNm_TxConfirmation via LSduR with result E_NOT_OK to indicate to this upper layer BSW module that NM-message could not be transmitted successfully.	
Available via	<none>	

⌋

8.7 Service Interfaces

No service interfaces provided.

9 Sequence diagrams

9.1 Use Case 01 - Initialization

Sequence in Figure 9.1 FrNm Init Sequence shows how to initialize the FlexRay Network Management.

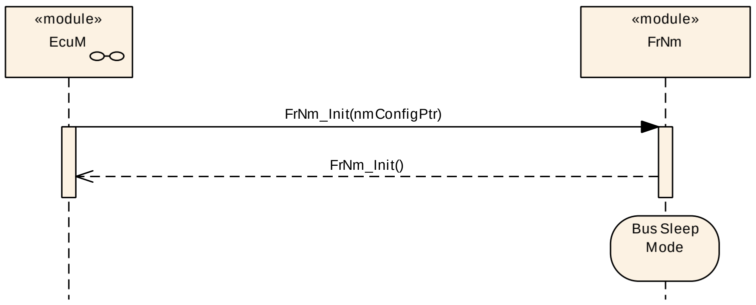

Figure 9.1: FrNm Init Sequence

9.2 Use Case 02 - Passive Startup

Sequence in Figure 9.2 shows the normal passive startup.

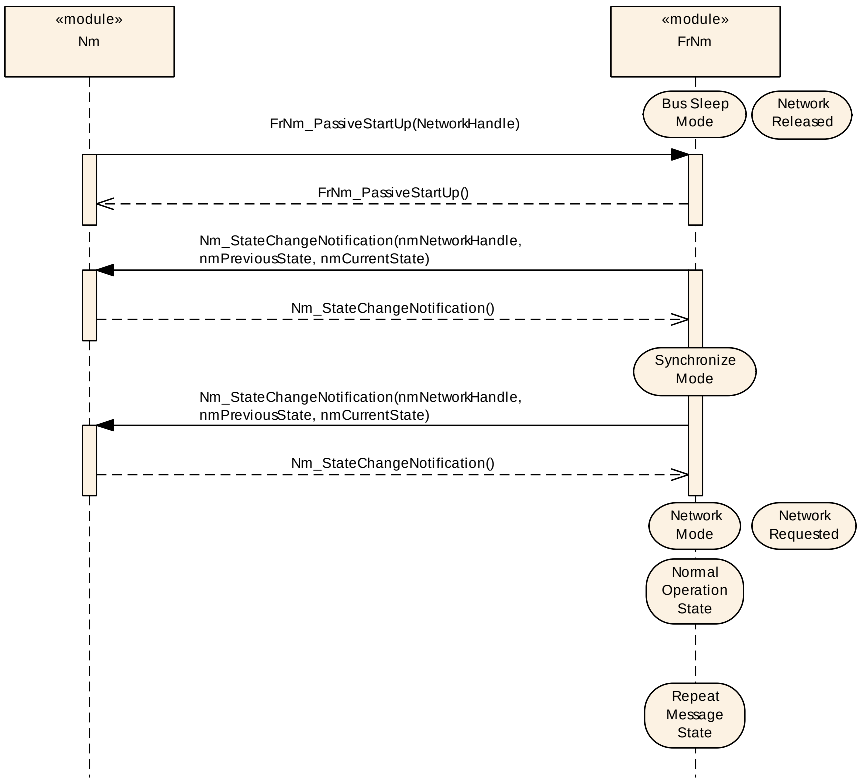
«module»
Nm
«module»
FrNm
Bus Sleep Mode
Network Released
FrNm_PassiveStartUp(NetworkHandle)
FrNm_PassiveStartUp()
Nm_StateChangeNotification(nmNetworkHandle, nmPreviousState, nmCurrentState)
Nm_StateChangeNotification()
Synchronize Mode
Nm_StateChangeNotification(nmNetworkHandle, nmPreviousState, nmCurrentState)
Nm_StateChangeNotification()
Network Mode
Network Requested
Normal Operation State
Repeat Message State

Figure 9.2: FrNm passive startup sequence

9.3 Use Case 03 - Passive Startup with a Network Request

Sequence in Figure 9.3 shows a passive startup where a network is requested before Network Mode has been reached.

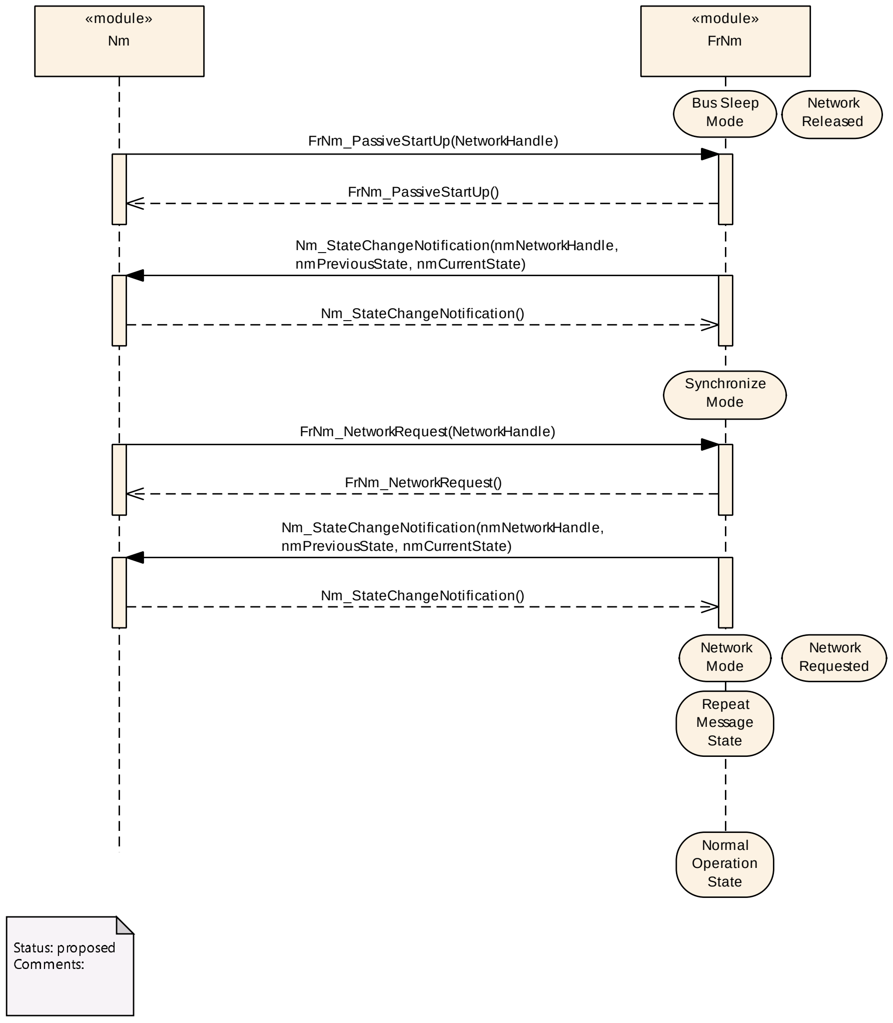
«module»
Nm
«module»
FrNm
Bus Sleep Mode
Network Released
FrNm_PassiveStartUp(NetworkHandle)
FrNm_PassiveStartUp()
Nm_StateChangeNotification(nmNetworkHandle, nmPreviousState, nmCurrentState)
Nm_StateChangeNotification()
Synchronize Mode
FrNm_NetworkRequest(NetworkHandle)
FrNm_NetworkRequest()
Nm_StateChangeNotification(nmNetworkHandle, nmPreviousState, nmCurrentState)
Nm_StateChangeNotification()
Network Mode
Network Requested
Repeat Message State
Normal Operation State
Status: proposed
Comments:

Figure 9.3: FrNm passive startup with a "active" network request sequence

9.4 Use Case 04 - Normal Operation

Sequence in Figure 9.4 shows how to request and release the network

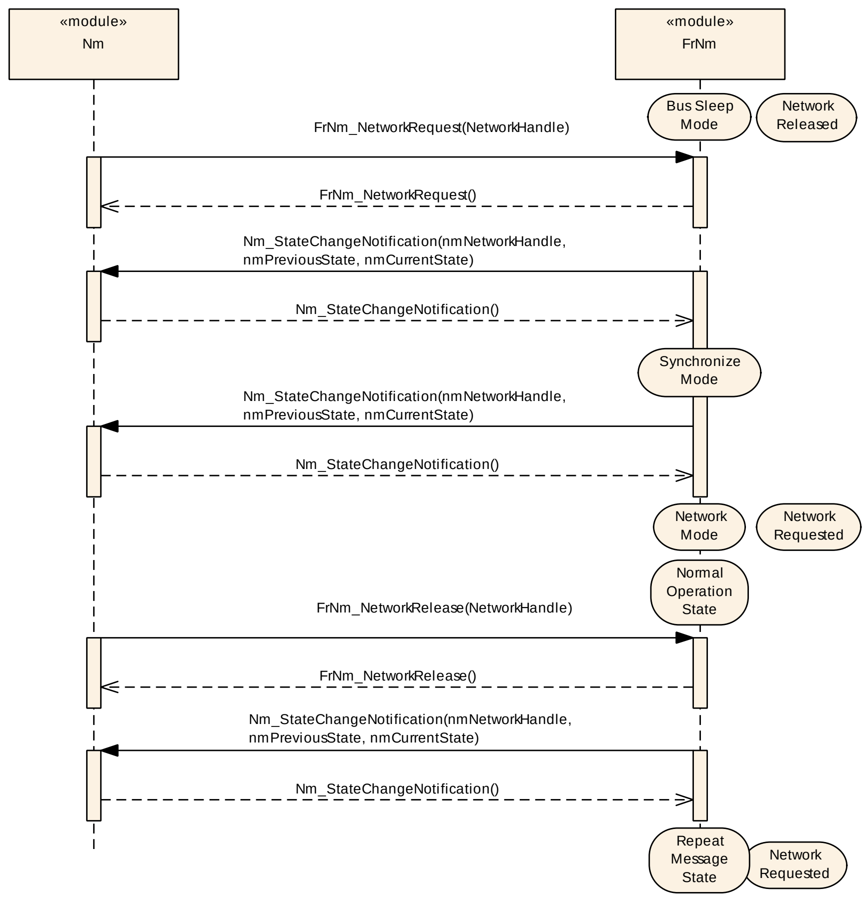
«module»
Nm
«module»
FrNm
Bus Sleep Mode
Network Released
FrNm_NetworkRequest(NetworkHandle)
FrNm_NetworkRequest()
Nm_StateChangeNotification(nmNetworkHandle, nmPreviousState, nmCurrentState)
Nm_StateChangeNotification()
Synchronize Mode
Nm_StateChangeNotification(nmNetworkHandle, nmPreviousState, nmCurrentState)
Nm_StateChangeNotification()
Network Mode
Network Requested
Normal Operation State
FrNm_NetworkRelease(NetworkHandle)
FrNm_NetworkRelease()
Nm_StateChangeNotification(nmNetworkHandle, nmPreviousState, nmCurrentState)
Nm_StateChangeNotification()
Repeat Message State
Network Requested

Figure 9.4: FrNm normal operation sequence

9.5 Use Case 05 - Shutdown

Sequence in Figure 9.5 shows a normal shutdown sequence.

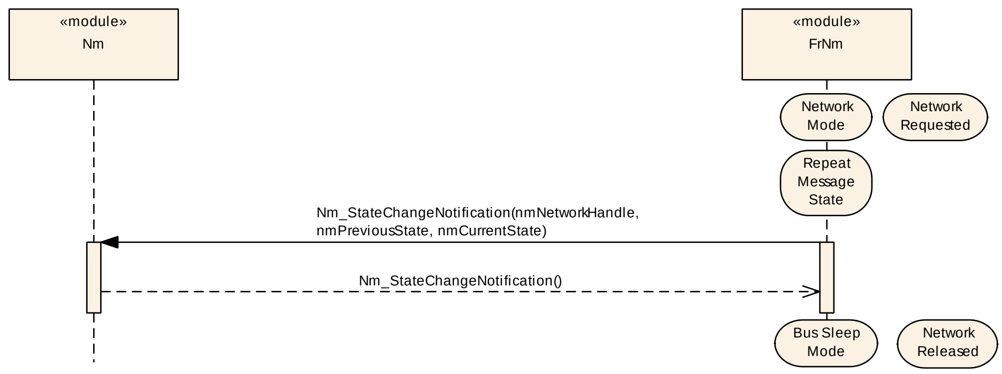
«module»
Nm
«module»
FrNm
Network Mode
Network Requested
Repeat Message State
Nm_StateChangeNotification(nmNetworkHandle, nmPreviousState, nmCurrentState)
Nm_StateChangeNotification()
Bus Sleep Mode
Network Released

Figure 9.5: FrNm Shutdown

10 Configuration specification

The following chapter contains tables of all configuration parameters and switches used to determine the functional units of the AUTOSAR Generic Network Management. The default values of configuration parameters are denoted as bold.

[SWS_FrNm_00020]

Upstream requirements: SRS_BSW_00159

⌈Both static and runtime configuration parameters shall be located outside the source code of the module.⌋

In general, this chapter defines configuration parameters and their clustering into containers. In order to support the specification Chapter 10.1 describes fundamentals. It also specifies a template (table) you shall use for the parameter specification. We intend to leave Chapter 10.1 in the specification to guarantee comprehension.

Chapter 10.2 specifies the structure (containers) and the parameters of the module FrNm.

Chapter 10.5 specifies published information of the module FrNm.

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 describe Chapter 7 and Chapter 8.

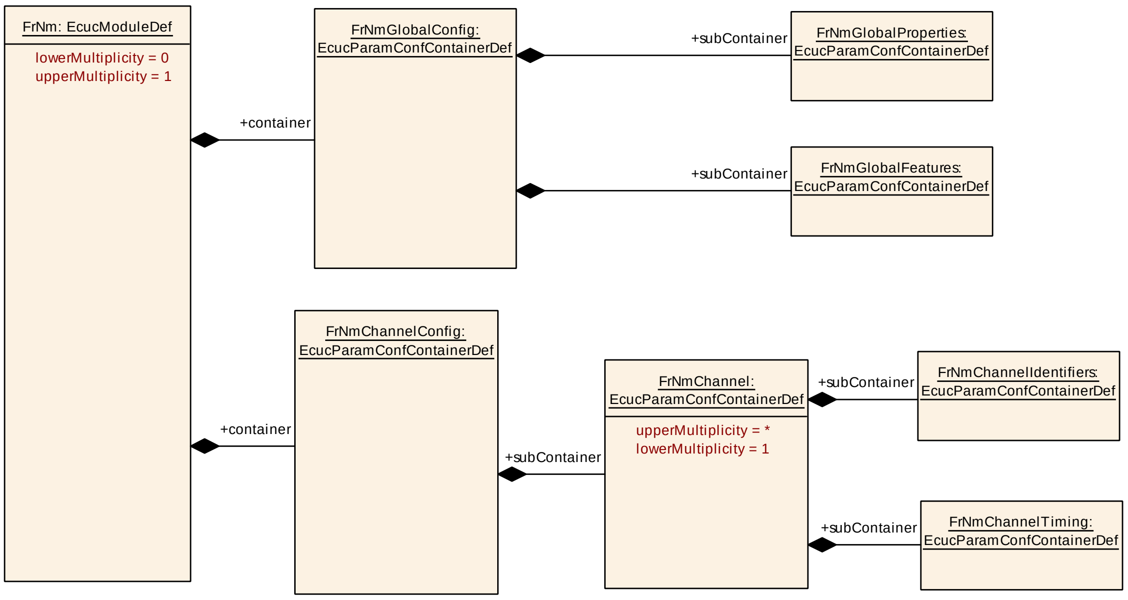
FrNm: EcucModuleDef
lowerMultiplicity = 0
upperMultiplicity = 1
+container
FrNmGlobalConfig: EcucParamConfContainerDef
+subContainer
FrNmGlobalProperties: EcucParamConfContainerDef
+subContainer
FrNmGlobalFeatures: EcucParamConfContainerDef
+container
FrNmChannelConfig: EcucParamConfContainerDef
+subContainer
FrNmChannel: EcucParamConfContainerDef
upperMultiplicity = *
lowerMultiplicity = 1
+subContainer
FrNmChannelIdentifiers: EcucParamConfContainerDef
+subContainer
FrNmChannelTiming: EcucParamConfContainerDef

Figure 10.1: Configuration Parameters

10.2.1 FrNm

[ECUC_FrNm_00083] Definition of EcucModuleDef FrNm ⌈

Module Name	FrNm
Description	The Flexray Nm 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**
FrNmChannelConfig	1	This container contains the configuration parameters for all Flex Ray NM channels.
FrNmGlobalConfig	1	This container contains all global configuration parameters for the FrNm module.

⌋

10.3 Global configurable parameters

10.3.1 FrNmGlobalConfig

[ECUC_FrNm_00001] Definition of EcucParamConfContainerDef FrNmGlobal Config ⌈

Container Name	FrNmGlobalConfig
Parent Container	FrNm
Description	This container contains all global configuration parameters for the FrNm module.
Configuration Parameters	

No Included Parameters

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
FrNmGlobalFeatures	1	This container contains module features related to the FlexRay NM functionality.
FrNmGlobalProperties	1	This container contains module properties related to the FlexRay NM functionality.

⌋

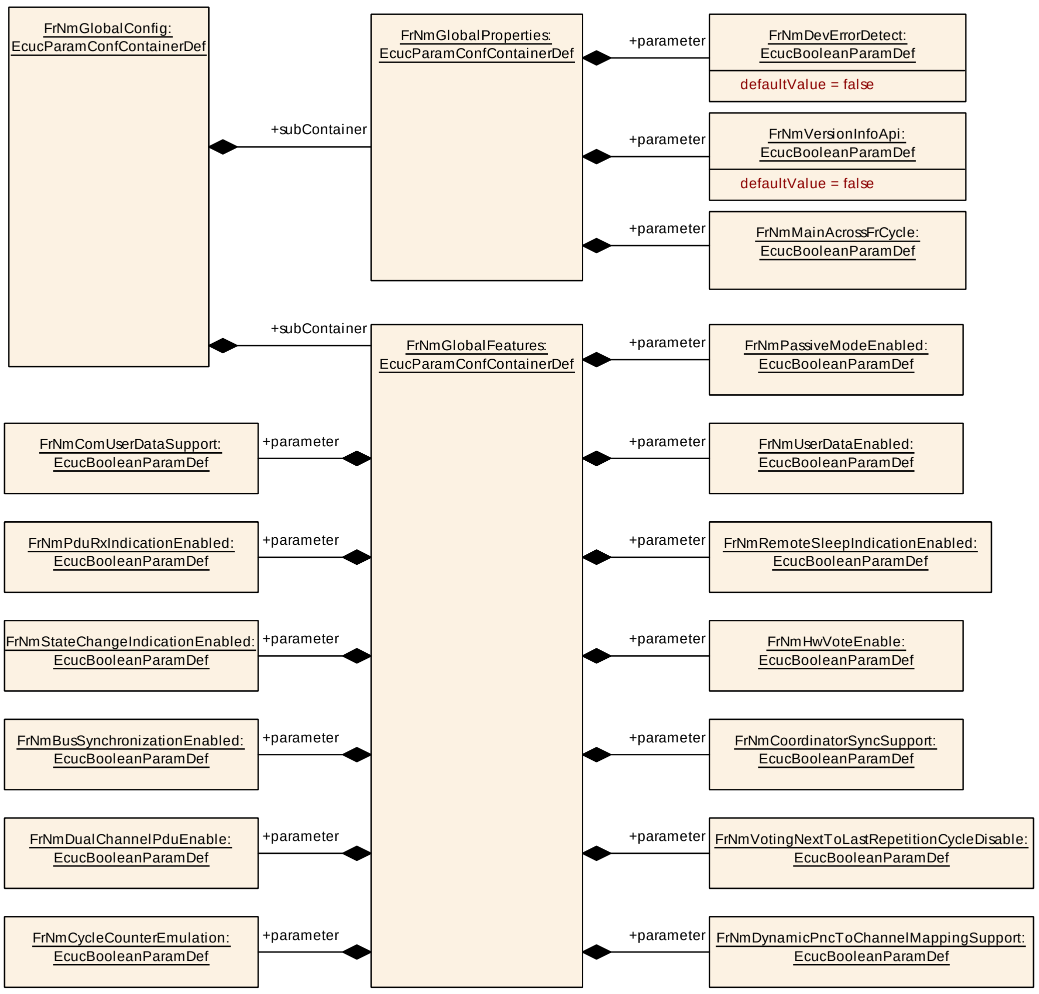
FrNmGlobalConfig: EcucParamConfContainerDef
+subContainer
FrNmGlobalProperties: EcucParamConfContainerDef
+parameter
FrNmDevErrorDetect: EcucBooleanParamDef
defaultValue = false
+parameter
FrNmVersionInfoApi: EcucBooleanParamDef
defaultValue = false
+parameter
FrNmMainAcrossFrCycle: EcucBooleanParamDef
+subContainer
FrNmGlobalFeatures: EcucParamConfContainerDef
+parameter
FrNmPassiveModeEnabled: EcucBooleanParamDef
FrNmComUserDataSupport: EcucBooleanParamDef
FrNmUserDataEnabled: EcucBooleanParamDef
FrNmPduRxIndicationEnabled: EcucBooleanParamDef
FrNmRemoteSleepIndicationEnabled: EcucBooleanParamDef
FrNmStateChangeIndicationEnabled: EcucBooleanParamDef
FrNmHwVoteEnable: EcucBooleanParamDef
FrNmBusSynchronizationEnabled: EcucBooleanParamDef
FrNmCoordinatorSyncSupport: EcucBooleanParamDef
FrNmDualChannelPduEnable: EcucBooleanParamDef
FrNmVotingNextToLastRepetitionCycleDisable: EcucBooleanParamDef
FrNmCycleCounterEmulation: EcucBooleanParamDef
FrNmDynamicPncToChannelMappingSupport: EcucBooleanParamDef

Figure 10.2: Global Configuration Parameters

10.3.2 FrNmGlobalFeatures

[ECUC_FrNm_00004] Definition of EcucParamConfContainerDef FrNmGlobalFeatures ⌈

Container Name	FrNmGlobalFeatures
Parent Container	FrNmGlobalConfig
Description	This container contains module features related to the FlexRay NM functionality.
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
FrNmBusSynchronizationEnabled	1	[ECUC_FrNm_00048]
FrNmComUserDataSupport	1	[ECUC_FrNm_00054]
FrNmCoordinatorSyncSupport	1	[ECUC_FrNm_00081]
FrNmCycleCounterEmulation	1	[ECUC_FrNm_00060]
FrNmDualChannelPduEnable	1	[ECUC_FrNm_00049]
FrNmDynamicPncToChannelMappingSupport	1	[ECUC_FrNm_00090]
FrNmHwVoteEnable	1	[ECUC_FrNm_00050]
FrNmPassiveModeEnabled	1	[ECUC_FrNm_00043]
FrNmPduRxIndicationEnabled	1	[ECUC_FrNm_00046]
FrNmRemoteSleepIndicationEnabled	1	[ECUC_FrNm_00044]
FrNmStateChangeIndicationEnabled	1	[ECUC_FrNm_00047]
FrNmUserDataEnabled	1	[ECUC_FrNm_00039]
FrNmVotingNextToLastRepetitionCycleDisable	1	[ECUC_FrNm_00073]

No Included Containers

⌋

[ECUC_FrNm_00048] Definition of EcucBooleanParamDef FrNmBusSynchronizationEnabled ⌈

Parameter Name	FrNmBusSynchronizationEnabled		
Parent Container	FrNmGlobalFeatures		
Description	Pre-processor switch for enabling the bus synchronization.		
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		

⌋

[ECUC_FrNm_00054] Definition of EcucBooleanParamDef FrNmComUserDataSupport ⌈

Parameter Name	FrNmComUserDataSupport		
Parent Container	FrNmGlobalFeatures		
Description	Preprocessor switch for enabling the Tx path of Com User Data. Use case: Setting of NMUserData via SWC.		
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 dependency: FrNmComUserDataSupport shall be set to FALSE, if all bytes of the NM PDU are used for NM System Bytes and for the PNC bit vector and no space is left for user data.		

⌋

[ECUC_FrNm_00081] Definition of EcucBooleanParamDef FrNmCoordinatorSyncSupport ⌈

Parameter Name	FrNmCoordinatorSyncSupport		
Parent Container	FrNmGlobalFeatures		
Description	Enables/disables the coordinator synchronization support.		
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 dependency: FrNmCoordinatorSyncSupport has to be set to FALSE if FrNmPassiveModeEnabled is set to TRUE.		

⌋

[ECUC_FrNm_00060] Definition of EcucBooleanParamDef FrNmCycleCounterEmulation ⌈

Parameter Name	FrNmCycleCounterEmulation
Parent Container	FrNmGlobalFeatures
Description	Pre-processor switch for enabling the cycle counter emulation.
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		

⌋

[ECUC_FrNm_00049] Definition of EcucBooleanParamDef FrNmDualChannelPdu Enable ⌈

Parameter Name	FrNmDualChannelPduEnable		
Parent Container	FrNmGlobalFeatures		
Description	Pre-processor switch for enabling the support of dual channel transmission and reception of NM messages.		
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		

⌋

[ECUC_FrNm_00090] Definition of EcucBooleanParamDef FrNmDynamicPncTo ChannelMappingSupport ⌈

Parameter Name	FrNmDynamicPncToChannelMappingSupport		
Parent Container	FrNmGlobalFeatures		
Description	Precompile time switch to enable the dynamic PNC-to-channel-mapping handling. False: Dynamic PNC-to-channel-mapping is disabled True: Dynamic PNC-to-channel-mapping is enabled		
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: ECU dependency: FrNmDynamicPncToChannelMappingSupport == TRUE only allowed if Fr NmPnEnabled == true for at least one FrNm Channel and FrNmPassiveModeEnabled == FALSE		

⌋

[ECUC_FrNm_00050] Definition of EcucBooleanParamDef FrNmHwVoteEnable ⌈

Parameter Name	FrNmHwVoteEnable		
Parent Container	FrNmGlobalFeatures		
Description	Pre-processor switch for enabling the processing of FlexRay Hardware aggregated NM-Votes. This switch enables/disables the optional API FrIf_GetNmVector.		
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		

⌋

[ECUC_FrNm_00043] Definition of EcucBooleanParamDef FrNmPassiveModeEnabled ⌈

Parameter Name	FrNmPassiveModeEnabled		
Parent Container	FrNmGlobalFeatures		
Description	Pre-processor switch for enabling Passive Node Configuration support.		
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		

⌋

[ECUC_FrNm_00046] Definition of EcucBooleanParamDef FrNmPduRxIndicationEnabled ⌈

Parameter Name	FrNmPduRxIndicationEnabled		
Parent Container	FrNmGlobalFeatures		
Description	Pre-processor switch for enabling PDU reception indication.		
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

⌋

[ECUC_FrNm_00044] Definition of EcucBooleanParamDef FrNmRemoteSleepIndicationEnabled ⌈

Parameter Name	FrNmRemoteSleepIndicationEnabled		
Parent Container	FrNmGlobalFeatures		
Description	Pre-processor switch for enabling remote sleep indication. calculationFormula = If (FrNmPassiveModeEnabled == True) then Equal(False) else Equal(False or True)		
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		

⌋

[ECUC_FrNm_00047] Definition of EcucBooleanParamDef FrNmStateChangeIndicationEnabled ⌈

Parameter Name	FrNmStateChangeIndicationEnabled		
Parent Container	FrNmGlobalFeatures		
Description	Pre-processor switch for enabling state change indication.		
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		

⌋

[ECUC_FrNm_00039] Definition of EcucBooleanParamDef FrNmUserDataEnabled ⌈

Parameter Name	FrNmUserDataEnabled		
Parent Container	FrNmGlobalFeatures		
Description	Pre-processor switch for enabling user data support.		
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 dependency: FrNmUserDataEnabled shall be set to FALSE, if all bytes of the NM PDU are used for NM System Bytes and for the PNC bit vector and no space is left for user data.		

⌋

[ECUC_FrNm_00073] Definition of EcucBooleanParamDef FrNmVotingNextToLastRepetitionCycleDisable ⌈

Parameter Name	FrNmVotingNextToLastRepetitionCycleDisable		
Parent Container	FrNmGlobalFeatures		
Description	Pre-processor switch for disabling vote changes in the last two repetition cycles before the Ready Sleep Counter expires.		
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		

⌋

10.3.3 FrNmGlobalProperties

[ECUC_FrNm_00003] Definition of EcucParamConfContainerDef FrNmGlobalProperties ⌈

Container Name	FrNmGlobalProperties
Parent Container	FrNmGlobalConfig
Description	This container contains module properties related to the FlexRay NM functionality.
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
FrNmDevErrorDetect	1	[ECUC_FrNm_00036]
FrNmMainAcrossFrCycle	1	[ECUC_FrNm_00038]
FrNmVersionInfoApi	1	[ECUC_FrNm_00037]

No Included Containers

⌋

[ECUC_FrNm_00036] Definition of EcucBooleanParamDef FrNmDevErrorDetect ⌈

Parameter Name	FrNmDevErrorDetect		
Parent Container	FrNmGlobalProperties		
Description	Switches the development error detection and notification on or off. • 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		

⌋

[ECUC_FrNm_00038] Definition of EcucBooleanParamDef FrNmMainAcrossFrCycle ⌈

Parameter Name	FrNmMainAcrossFrCycle		
Parent Container	FrNmGlobalProperties		
Description	If the FlexRay NM MainFunction is executed completely within the FlexRay communication cycle where the last NM vote of the current vote cycle is received, the FrNmMainAcrossFrCycle shall be configured to FALSE. If the FlexRay NM MainFunction is executed completely within the FlexRay communication cycle subsequent to the one where the last NM vote of the current vote cycle is received, the FrNmMainAcrossFrCycle shall be configured to TRUE.		
Multiplicity	1		
Type	EcucBooleanParamDef		
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

⌋

[ECUC_FrNm_00037] Definition of EcucBooleanParamDef FrNmVersionInfoApi ⌈

Parameter Name	FrNmVersionInfoApi		
Parent Container	FrNmGlobalProperties		
Description	Pre-processor switch for enabling version info API support.		
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		

⌋

10.4 Channel configurable parameters

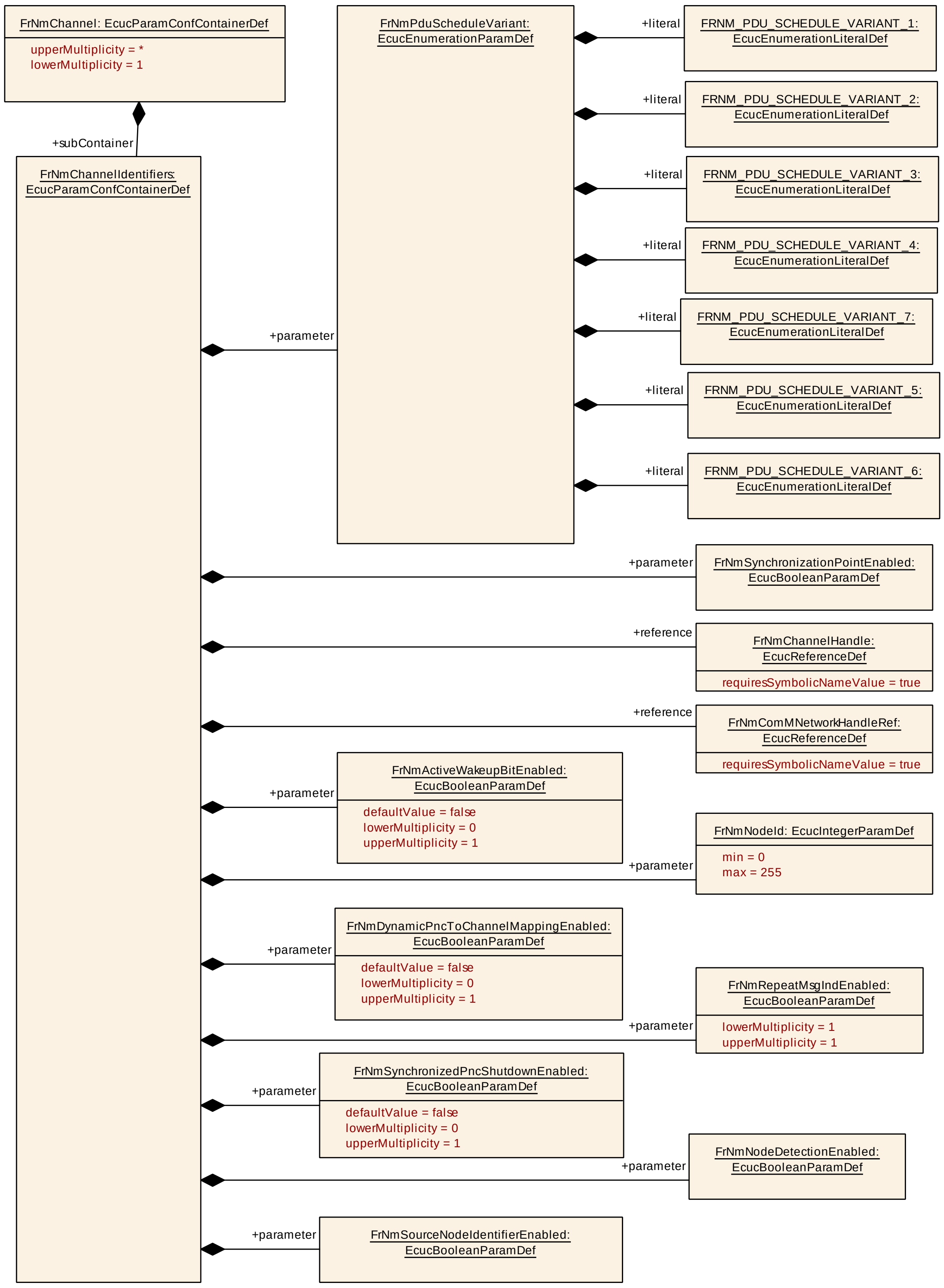
FrNmChannel: EcucParamConfContainerDef
upperMultiplicity = *
lowerMultiplicity = 1
+subContainer
FrNmChannelIdentifiers: EcucParamConfContainerDef
+parameter
FrNmPduScheduleVariant: EcucEnumerationParamDef
+literal
FRNM_PDU_SCHEDULE_VARIANT_1: EcucEnumerationLiteralDef
+literal
FRNM_PDU_SCHEDULE_VARIANT_2: EcucEnumerationLiteralDef
+literal
FRNM_PDU_SCHEDULE_VARIANT_3: EcucEnumerationLiteralDef
+literal
FRNM_PDU_SCHEDULE_VARIANT_4: EcucEnumerationLiteralDef
+literal
FRNM_PDU_SCHEDULE_VARIANT_7: EcucEnumerationLiteralDef
+literal
FRNM_PDU_SCHEDULE_VARIANT_5: EcucEnumerationLiteralDef
+literal
FRNM_PDU_SCHEDULE_VARIANT_6: EcucEnumerationLiteralDef
+parameter
FrNmSynchronizationPointEnabled: EcucBooleanParamDef
+reference
FrNmChannelHandle: EcucReferenceDef
requiresSymbolicNameValue = true
+reference
FrNmComMNetworkHandleRef: EcucReferenceDef
requiresSymbolicNameValue = true
+parameter
FrNmActiveWakeupBitEnabled: EcucBooleanParamDef
defaultValue = false
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
FrNmNodeId: EcucIntegerParamDef
min = 0
max = 255
+parameter
FrNmDynamicPncToChannelMappingEnabled: EcucBooleanParamDef
defaultValue = false
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
FrNmRepeatMsgIndEnabled: EcucBooleanParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
FrNmSynchronizedPncShutdownEnabled: EcucBooleanParamDef
defaultValue = false
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
FrNmNodeDetectionEnabled: EcucBooleanParamDef
+parameter
FrNmSourceNodeIdentifierEnabled: EcucBooleanParamDef

Figure 10.3: Channel Configurable Parameters - part 1

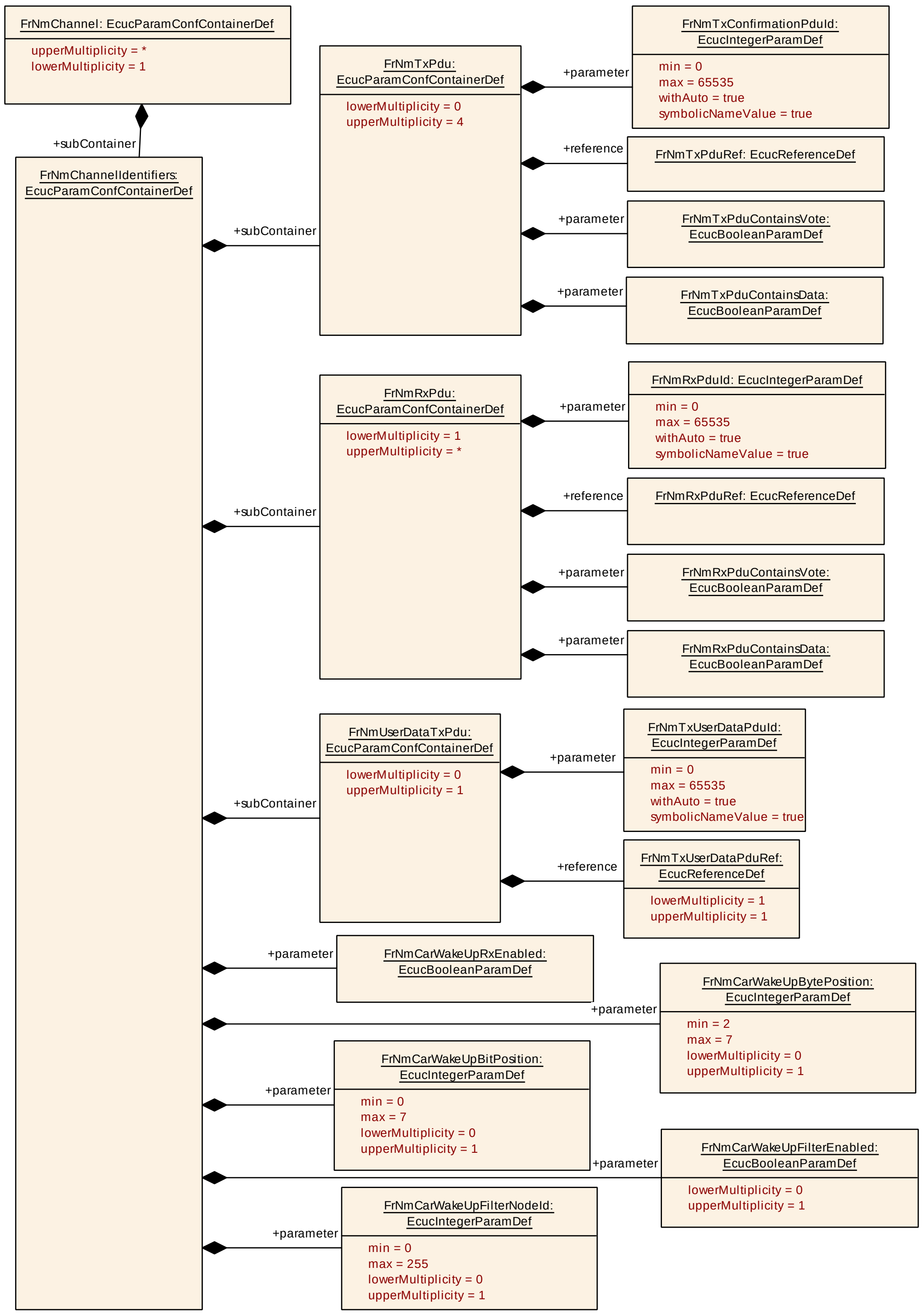
FrNmChannel: EcucParamConfContainerDef
upperMultiplicity = *
lowerMultiplicity = 1
+subContainer
FrNmChannelIdentifiers: EcucParamConfContainerDef
+subContainer
FrNmTxPdu: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 4
+parameter
FrNmTxConfirmationPduId: EcucIntegerParamDef
min = 0
max = 65535
withAuto = true
symbolicNameValue = true
+reference
FrNmTxPduRef: EcucReferenceDef
+parameter
FrNmTxPduContainsVote: EcucBooleanParamDef
+parameter
FrNmTxPduContainsData: EcucBooleanParamDef
+subContainer
FrNmRxPdu: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = *
+parameter
FrNmRxPduId: EcucIntegerParamDef
min = 0
max = 65535
withAuto = true
symbolicNameValue = true
+reference
FrNmRxPduRef: EcucReferenceDef
+parameter
FrNmRxPduContainsVote: EcucBooleanParamDef
+parameter
FrNmRxPduContainsData: EcucBooleanParamDef
+subContainer
FrNmUserDataTxPdu: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
FrNmTxUserDataPduId: EcucIntegerParamDef
min = 0
max = 65535
withAuto = true
symbolicNameValue = true
+reference
FrNmTxUserDataPduRef: EcucReferenceDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
FrNmCarWakeUpRxEnabled: EcucBooleanParamDef
+parameter
FrNmCarWakeUpBytePosition: EcucIntegerParamDef
min = 2
max = 7
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
FrNmCarWakeUpBitPosition: EcucIntegerParamDef
min = 0
max = 7
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
FrNmCarWakeUpFilterEnabled: EcucBooleanParamDef
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
FrNmCarWakeUpFilterNodeId: EcucIntegerParamDef
min = 0
max = 255
lowerMultiplicity = 0
upperMultiplicity = 1

Figure 10.4: Channel Configurable Parameters - part 2

[SWS_FrNm_00036]

Upstream requirements: RS_Nm_00149

⌈The following runtime configurable parameters shall be configurable for each channel separately.⌋

10.4.1 FrNmChannelConfig

[ECUC_FrNm_00002] Definition of EcucParamConfContainerDef FrNmChannelConfig ⌈

Container Name	FrNmChannelConfig
Parent Container	FrNm
Description	This container contains the configuration parameters for all FlexRay NM channels.
Configuration Parameters	

No Included Parameters

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
FrNmChannel	1..*	This container contains the configuration parameters for a Flex Ray NM Channel.

⌋

10.4.2 FrNmChannel

[ECUC_FrNm_00006] Definition of EcucParamConfContainerDef FrNmChannel ⌈

Container Name	FrNmChannel		
Parent Container	FrNmChannelConfig		
Description	This container contains the configuration parameters for a FlexRay NM Channel.		
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			

No Included Parameters

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
FrNmChannelIdentifiers	1	This container contains instance specific identifiers related to the respective FlexRay Channel.
FrNmChannelTiming	1	This container contains instance-specific timing related to the respective FlexRay Channel.

⌋

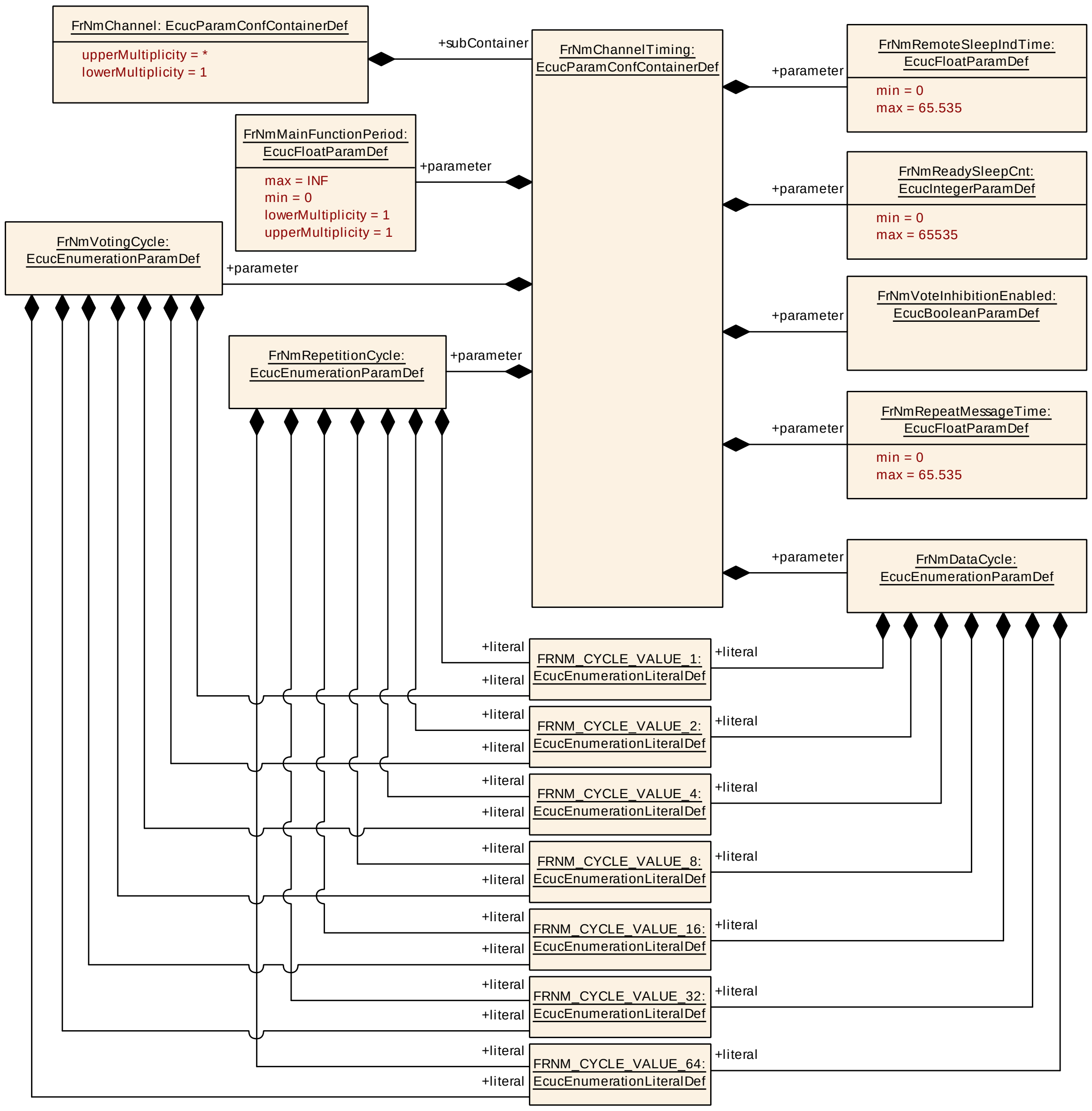
FrNmChannel: EcucParamConfContainerDef
upperMultiplicity = *
lowerMultiplicity = 1
+subContainer
FrNmChannelTiming: EcucParamConfContainerDef
FrNmMainFunctionPeriod: EcucFloatParamDef
max = INF
min = 0
lowerMultiplicity = 1
upperMultiplicity = 1
FrNmVotingCycle: EcucEnumerationParamDef
FrNmRepetitionCycle: EcucEnumerationParamDef
FrNmRemoteSleepIndTime: EcucFloatParamDef
min = 0
max = 65.535
FrNmReadySleepCnt: EcucIntegerParamDef
min = 0
max = 65535
FrNmVoteInhibitionEnabled: EcucBooleanParamDef
FrNmRepeatMessageTime: EcucFloatParamDef
min = 0
max = 65.535
FrNmDataCycle: EcucEnumerationParamDef
+parameter
+literal
FRNM_CYCLE_VALUE_1: EcucEnumerationLiteralDef
FRNM_CYCLE_VALUE_2: EcucEnumerationLiteralDef
FRNM_CYCLE_VALUE_4: EcucEnumerationLiteralDef
FRNM_CYCLE_VALUE_8: EcucEnumerationLiteralDef
FRNM_CYCLE_VALUE_16: EcucEnumerationLiteralDef
FRNM_CYCLE_VALUE_32: EcucEnumerationLiteralDef
FRNM_CYCLE_VALUE_64: EcucEnumerationLiteralDef

Figure 10.5: FrNm Channel Parameters

10.4.3 FrNmChannelTiming

[ECUC_FrNm_00008] Definition of EcucParamConfContainerDef FrNmChannelTiming ⌈

Container Name	FrNmChannelTiming
Parent Container	FrNmChannel
Description	This container contains instance-specific timing related to the respective FlexRay Channel.
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
FrNmDataCycle	1	[ECUC_FrNm_00031]
FrNmMainFunctionPeriod	1	[ECUC_FrNm_00035]
FrNmReadySleepCnt	1	[ECUC_FrNm_00051]
FrNmRemoteSleepIndTime	1	[ECUC_FrNm_00029]
FrNmRepeatMessageTime	1	[ECUC_FrNm_00030]
FrNmRepetitionCycle	1	[ECUC_FrNm_00033]
FrNmVoteInhibitionEnabled	1	[ECUC_FrNm_00053]
FrNmVotingCycle	1	[ECUC_FrNm_00032]

No Included Containers

⌋

[ECUC_FrNm_00031] Definition of EcucEnumerationParamDef FrNmDataCycle ⌈

Parameter Name	FrNmDataCycle		
Parent Container	FrNmChannelTiming		
Description	Number of FlexRay Schedule Cycles needed to transmit the NM Data of all ECUs on the FlexRay bus		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	FRNM_CYCLE_VALUE_1	–	
	FRNM_CYCLE_VALUE_16	–	
	FRNM_CYCLE_VALUE_2	–	
	FRNM_CYCLE_VALUE_32	–	
	FRNM_CYCLE_VALUE_4	–	
	FRNM_CYCLE_VALUE_64	–	
	FRNM_CYCLE_VALUE_8	–	
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		

⌋

[ECUC_FrNm_00035] Definition of EcucFloatParamDef FrNmMainFunctionPeriod ⌈

Parameter Name	FrNmMainFunctionPeriod		
Parent Container	FrNmChannelTiming		
Description	This parameter defines the processing cycle of the main function of FrNm module in seconds.		
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		

⌋

[ECUC_FrNm_00051] Definition of EcucIntegerParamDef FrNmReadySleepCnt ⌈

Parameter Name	FrNmReadySleepCnt		
Parent Container	FrNmChannelTiming		
Description	FrNm switches to bus sleep mode at the end of the FrNmReadySleepCnt+1 repetition cycle without any NM vote. E.g. on a value of "1", the NM-State Machine will leave the Ready Sleep State after two NM Repetition Cycles with no "keep awake" votes.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
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 dependency: Condition: FrNmReadySleepCnt >= 1		

⌋

[ECUC_FrNm_00029] Definition of EcucFloatParamDef FrNmRemoteSleepInd Time ⌈

Parameter Name	FrNmRemoteSleepIndTime		
Parent Container	FrNmChannelTiming		
Description	Timeout for Remote Sleep Indication. It defines the time in seconds how long it shall take to recognize that all other nodes are ready to sleep. The value "0" denotes that no Remote Sleep Indication functionality is configured.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. 65.535]		
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 dependency: Condition: FrNmRemoteSleepIndTime >= FrNmRepetitionCycle or FrNm RemoteSleepIndTime = 0		

⌋

[ECUC_FrNm_00030] Definition of EcucFloatParamDef FrNmRepeatMessage Time ⌈

Parameter Name	FrNmRepeatMessageTime		
Parent Container	FrNmChannelTiming		
Description	Timeout for Repeat Message State. Defines the time in seconds how long the NM shall stay in the Repeat Message State. The value "0" denotes that no Repeat Message State is configured, which means that Repeat Message State is transient and implies that it is left immediately after entry and consequently no startup stability is guaranteed and no node detection procedure is possible.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. 65.535]		
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: ECU dependency: Timing value shall be an integer multiple of the time for one repetition cycle.		

⌋

[ECUC_FrNm_00033] Definition of EcucEnumerationParamDef FrNmRepetitionCycle ⌈

Parameter Name	FrNmRepetitionCycle		
Parent Container	FrNmChannelTiming		
Description	Number of Flexray Schedule Cycles used to repeat the transmission of the Nm vote of all ECUs on the Flexray Bus.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	FRNM_CYCLE_VALUE_1	–	
	FRNM_CYCLE_VALUE_16	–	
	FRNM_CYCLE_VALUE_2	–	
	FRNM_CYCLE_VALUE_32	–	
	FRNM_CYCLE_VALUE_4	–	
	FRNM_CYCLE_VALUE_64	–	
	FRNM_CYCLE_VALUE_8	–	
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 dependency: Condition: FrNmRepetitionCycle = n * FrNmVotingCycle; n = [1,2,4,8,16,32,64]		

⌋

[ECUC_FrNm_00053] Definition of EcucBooleanParamDef FrNmVoteInhibitionEnabled ⌈

Parameter Name	FrNmVoteInhibitionEnabled		
Parent Container	FrNmChannelTiming		
Description	Pre-processor switch for enabling the inhibition of vote changes from the next-to-last repetition cycle to the last repetition cycle before the Ready Sleep Counter expires.		
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		

⌋

[ECUC_FrNm_00032] Definition of EcucEnumerationParamDef FrNmVotingCycle ⌈

Parameter Name	FrNmVotingCycle	
Parent Container	FrNmChannelTiming	
Description	Number of FlexRay Schedule Cycles needed to transmit the Nm vote of all ECUs on the FlexRay Bus.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	FRNM_CYCLE_VALUE_1	–
	FRNM_CYCLE_VALUE_16	–
	FRNM_CYCLE_VALUE_2	–
	FRNM_CYCLE_VALUE_32	–
	FRNM_CYCLE_VALUE_4	–
	FRNM_CYCLE_VALUE_64	–
	FRNM_CYCLE_VALUE_8	–
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		

⌋

10.4.4 FrNmChannelIdentifiers

[ECUC_FrNm_00007] Definition of EcucParamConfContainerDef FrNmChannelIdentifiers ⌈

Container Name	FrNmChannelIdentifiers
Parent Container	FrNmChannel
Description	This container contains instance specific identifiers related to the respective FlexRay Channel.
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
FrNmActiveWakeupBitEnabled	0..1	[ECUC_FrNm_00082]
FrNmCarWakeUpBitPosition	0..1	[ECUC_FrNm_00076]
FrNmCarWakeUpBytePosition	0..1	[ECUC_FrNm_00075]
FrNmCarWakeUpFilterEnabled	0..1	[ECUC_FrNm_00077]
FrNmCarWakeUpFilterNodeId	0..1	[ECUC_FrNm_00078]
FrNmCarWakeUpRxEnabled	1	[ECUC_FrNm_00074]
FrNmDynamicPncToChannelMappingEnabled	0..1	[ECUC_FrNm_00092]

▽

△

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
FrNmNodeDetectionEnabled	1	[ECUC_FrNm_00086]
FrNmNodeId	1	[ECUC_FrNm_00017]
FrNmPduScheduleVariant	1	[ECUC_FrNm_00022]
FrNmPnEnabled	1	[ECUC_FrNm_00072]
FrNmRepeatMsgIndEnabled	1	[ECUC_FrNm_00091]
FrNmSourceNodeIdentifierEnabled	1	[ECUC_FrNm_00085]
FrNmSynchronizationPointEnabled	1	[ECUC_FrNm_00021]
FrNmSynchronizedPncShutdownEnabled	0..1	[ECUC_FrNm_00094]
FrNmChannelHandle	1	[ECUC_FrNm_00013]
FrNmComMNetworkHandleRef	1	[ECUC_FrNm_00014]

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
FrNmRxPdu	1..*	This container describes the FlexRay NM RX PDU:s.
FrNmTxPdu	0..4	This container describes the FlexRay NM TX PDU:s.
FrNmUserDataTxPdu	0..1	This optional container is used to configure the UserNm PDU. This container is only available if FrNmComUserDataSupport is enabled.

⌋

[ECUC_FrNm_00082] Definition of EcucBooleanParamDef FrNmActiveWakeupBitEnabled ⌈

Parameter Name	FrNmActiveWakeupBitEnabled		
Parent Container	FrNmChannelIdentifiers		
Description	Enables/Disables the handling of the Active Wakeup Bit in the FrNm module.		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default value	false		
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		

⌋

[ECUC_FrNm_00076] Definition of EcucIntegerParamDef FrNmCarWakeUpBitPosition ⌈

Parameter Name	FrNmCarWakeUpBitPosition		
Parent Container	FrNmChannelIdentifiers		
Description	Specifies the Bit position of the CWU within the NM-Message.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 7		
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 dependency: only available if FrNmCarWakeUpRxEnabled == TRUE		

⌋

[ECUC_FrNm_00075] Definition of EcucIntegerParamDef FrNmCarWakeUpByte Position ⌈

Parameter Name	FrNmCarWakeUpBytePosition		
Parent Container	FrNmChannelIdentifiers		
Description	Specifies the Byte position of the CWU within the NM-Message.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	2 .. 7		
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 dependency: only available if FrNmCarWakeUpRxEnabled == TRUE		

⌋

[ECUC_FrNm_00077] Definition of EcucBooleanParamDef FrNmCarWakeUpFilterEnabled ⌈

Parameter Name	FrNmCarWakeUpFilterEnabled		
Parent Container	FrNmChannelIdentifiers		
Description	If CWU filtering is supported, only the CWU bit within the NM message with source node identifier FrNmCarWakeUpFilterNodeId is considered as CWU request. FALSE - CWU Filtering is not supported TRUE - CWU Filtering is supported		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
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 dependency: only available if FrNmCarWakeUpRxEnabled == TRUE		

⌋

[ECUC_FrNm_00078] Definition of EcucIntegerParamDef FrNmCarWakeUpFilterNodeId ⌈

Parameter Name	FrNmCarWakeUpFilterNodeId		
Parent Container	FrNmChannelIdentifiers		
Description	Source node identifier for CWU filtering. If CWU filtering is supported, only the CWU bit within the NM message with source node identifier FrNmCarWakeUpFilterNodeId is considered as CWU request.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
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 dependency: only available if FrNmCarWakeUpRxEnabled == TRUE		

⌋

[ECUC_FrNm_00074] Definition of EcucBooleanParamDef FrNmCarWakeUpRx Enabled ⌈

Parameter Name	FrNmCarWakeUpRxEnabled		
Parent Container	FrNmChannelIdentifiers		
Description	Enables or disables support of CarWakeUp bit evaluation in received NM messages. FALSE - CarWakeUp not supported TRUE - CarWakeUp supported		
Multiplicity	1		
Type	EcucBooleanParamDef		
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		

⌋

[ECUC_FrNm_00092] Definition of EcucBooleanParamDef FrNmDynamicPncTo ChannelMappingEnabled

Status: DRAFT

⌈

Parameter Name	FrNmDynamicPncToChannelMappingEnabled		
Parent Container	FrNmChannelIdentifiers		
Description	Channel-specific parameter to enable the dynamic PNC-to-channel-mapping feature. False: Dynamic PNC-to-channel-mapping is disabled True: Dynamic PNC-to-channel-mapping is enabled **Tags:** atp.Status=draft		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default value	false		
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 dependency: Shall only be TRUE if FrNmDynamicPncToChannelMappingSupport is TRUE		

⌋

[ECUC_FrNm_00086] Definition of EcucBooleanParamDef FrNmNodeDetectionEnabled ⌈

Parameter Name	FrNmNodeDetectionEnabled		
Parent Container	FrNmChannelIdentifiers		
Description	This parameter is used to enable or disable node detection support for a FrNm Channel.		
Multiplicity	1		
Type	EcucBooleanParamDef		
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 dependency: FrNmSourceNodeIdentifierEnabled needs to be TRUE to use this feature for the corresponding FrNm Channel. calculationFormula = If (FrNmPassiveMode Enabled == False)then Equal(NmNodeDetectionEnabled) else Equal(False)		

⌋

[ECUC_FrNm_00017] Definition of EcucIntegerParamDef FrNmNodeId ⌈

Parameter Name	FrNmNodeId		
Parent Container	FrNmChannelIdentifiers		
Description	NM node identifier configured for the respective FlexRay Channel. It is used for identifying the respective NM node in the NM-cluster. It must be unique for each NM node within one NM cluster.		
Multiplicity	1		
Type	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		

⌋

[ECUC_FrNm_00022] Definition of EcucEnumerationParamDef FrNmPduScheduleVariant ⌈

Parameter Name	FrNmPduScheduleVariant		
Parent Container	FrNmChannelIdentifiers		
Description	This parameter defines the PDU scheduling variant that should be used for this channel. Option 1 NM-Vote and NM-Data in static segment (one PDU) Option 2 NM-Vote and NM-Data in dynamic segment (one PDU) Option 3 NM-Vote and NM-Data in static segment (separate PDU) Option 4 NM-Vote in static segment and NM-Data in dynamic segment Option 5 NM-Vote in dynamic segment and NM-Data in static segment Option 6 NM-Vote and NM-Data in dynamic segment (separate PDU) Option 7 Combined NM-Vote and CBV in static segment and NM-Data in dynamic segment		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	FRNM_PDU_SCHEDULE_VARIANT_1	NM-Vote and NM-Data in static segment (one PDU)	
	FRNM_PDU_SCHEDULE_VARIANT_2	NM-Vote and NM-Data in dynamic segment (one PDU)	
	FRNM_PDU_SCHEDULE_VARIANT_3	NM-Vote and NM-Data in static segment (separate PDU)	
	FRNM_PDU_SCHEDULE_VARIANT_4	NM-Vote in static segment and NM-Data in dynamic segment	
	FRNM_PDU_SCHEDULE_VARIANT_5	NM-Vote in dynamic segment and NM-Data in static segment	
	FRNM_PDU_SCHEDULE_VARIANT_6	NM-Vote and NM-Data in dynamic segment (separate PDU)	
	FRNM_PDU_SCHEDULE_VARIANT_7	Combined NM-Vote and CBV in static segment and NM-Data in dynamic segment	
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		

⌋

[ECUC_FrNm_00072] Definition of EcucBooleanParamDef FrNmPnEnabled ⌈

Parameter Name	FrNmPnEnabled		
Parent Container	FrNmChannelIdentifiers		
Description	Enables or disables support of partial networking. false: Partial networking Range not supported true: Partial networking supported		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
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: If FrNmPnEnabled == TRUE then FrNmComUserDataSupport = TRUE

⌋

[ECUC_FrNm_00091] Definition of EcucBooleanParamDef FrNmRepeatMsgIndEnabled

Status: DRAFT

⌈

Parameter Name	FrNmRepeatMsgIndEnabled		
Parent Container	FrNmChannelIdentifiers		
Description	Enable/disable the notification that a RepeatMessageRequest bit has been received. **Tags:** atp.Status=draft		
Multiplicity	1		
Type	EcucBooleanParamDef		
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: ECU dependency: FrNmRepeatMsgIndEnabled = FALSE if FrNmPassiveModeEnabled == TRUE or (FrNmNodeDetectionEnabled == FALSE && FrNmDynamicPncToChannelMappingEnabled == FALSE). FrNmRepeatMsgIndEnabled = TRUE if FrNmDynamicPncToChannelMappingEnabled == TRUE.		

⌋

[ECUC_FrNm_00085] Definition of EcucBooleanParamDef FrNmSourceNodeIdentifierEnabled ⌈

Parameter Name	FrNmSourceNodeIdentifierEnabled		
Parent Container	FrNmChannelIdentifiers		
Description	This parameter is used to enable or disable SourceNodeIdentifier support for a FrNm Channel.		
Multiplicity	1		
Type	EcucBooleanParamDef		
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		

⌋

[ECUC_FrNm_00021] Definition of EcucBooleanParamDef FrNmSynchronizationPointEnabled ⌈

Parameter Name	FrNmSynchronizationPointEnabled		
Parent Container	FrNmChannelIdentifiers		
Description	This parameter defines if this channel shall provide the synchronization point indication to the NM Interface.		
Multiplicity	1		
Type	EcucBooleanParamDef		
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		

⌋

[ECUC_FrNm_00094] Definition of EcucBooleanParamDef FrNmSynchronizedPncShutdownEnabled ⌈

Parameter Name	FrNmSynchronizedPncShutdownEnabled		
Parent Container	FrNmChannelIdentifiers		
Description	Specifies if FrNm handle PN shutdown messages to support a synchronized PNC shutdown across a PN topology. This is only used for ECUs in the role of a top-level PNC coordinator or intermediate PNC coordinator. Thus, the PNC gateway functionality is enabled and therefore ERA calculation is used. FALSE: synchronized PNC shutdown is disabled TRUE: synchronized PNC shutdown is enabled.		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default value	false		
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
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Scope / Dependency	scope: local dependency: Only available if FrNmPnEnabled == TRUE and NmPnEraCalcEnabled == TRUE		

⌋

[ECUC_FrNm_00013] Definition of EcucReferenceDef FrNmChannelHandle ⌈

Parameter Name	FrNmChannelHandle		
Parent Container	FrNmChannelIdentifiers		
Description	Channel identifier configured for the respective instance of the NM. The FrNmChannelHandle shall be encoded in the FrNmRxPduId parameter which is passed to FrNm_RxIndication() function called by the LSduR.		
Multiplicity	1		
Type	Symbolic name reference to FrIfCluster		
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		

⌋

[ECUC_FrNm_00014] Definition of EcucReferenceDef FrNmComMNetworkHandleRef ⌈

Parameter Name	FrNmComMNetworkHandleRef		
Parent Container	FrNmChannelIdentifiers		
Description	This reference points to the unique channel defined by the ComMChannel and provides access to the unique channel index value in ComMChannelId.		
Multiplicity	1		
Type	Symbolic name reference to ComMChannel		
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: It must be unique for each NM instance within one ECU.		

⌋

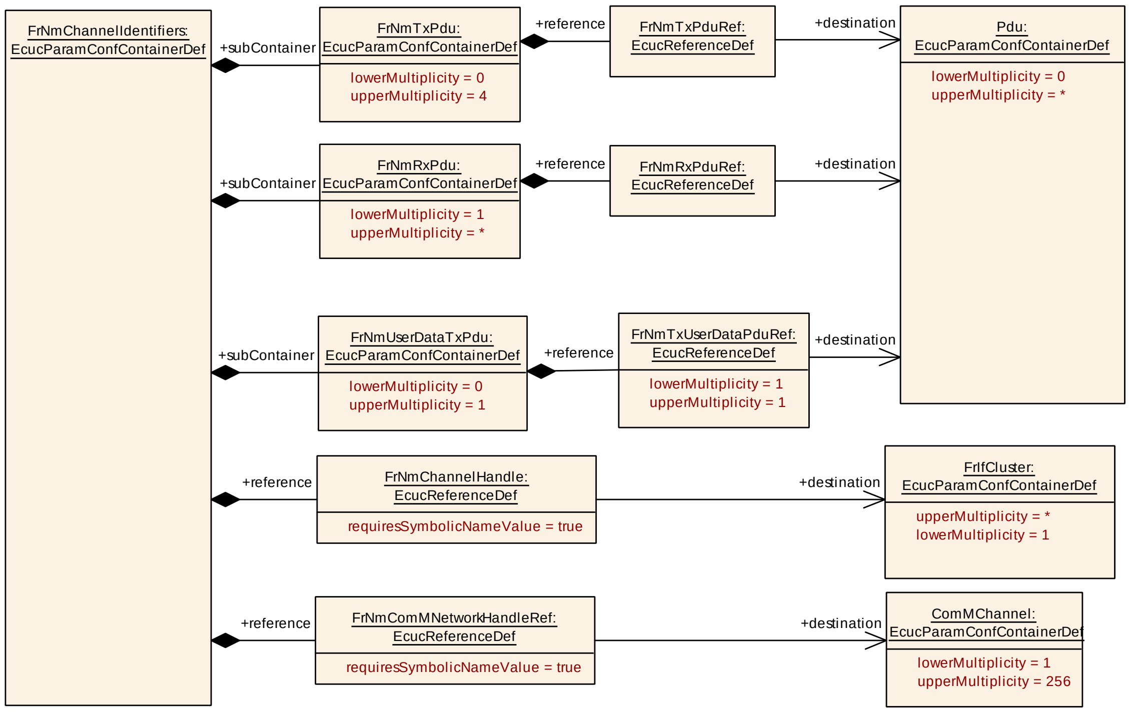
FrNmChannelIdentifiers: EcucParamConfContainerDef
+subContainer
FrNmTxPdu: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 4
+reference
FrNmTxPduRef: EcucReferenceDef
+destination
Pdu: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
+subContainer
FrNmRxPdu: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = *
+reference
FrNmRxPduRef: EcucReferenceDef
+destination
+subContainer
FrNmUserDataTxPdu: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+reference
FrNmTxUserDataPduRef: EcucReferenceDef
lowerMultiplicity = 1
upperMultiplicity = 1
+destination
+reference
FrNmChannelHandle: EcucReferenceDef
requiresSymbolicNameValue = true
+destination
FrIfCluster: EcucParamConfContainerDef
upperMultiplicity = *
lowerMultiplicity = 1
+reference
FrNmComMNetworkHandleRef: EcucReferenceDef
requiresSymbolicNameValue = true
+destination
ComMChannel: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 256

Figure 10.6: FrNm References

10.4.5 FrNmRxPdu

[ECUC_FrNm_00010] Definition of EcucParamConfContainerDef FrNmRxPdu ⌈

Container Name	FrNmRxPdu
Parent Container	FrNmChannelIdentifiers
Description	This container describes the FlexRay NM RX PDU:s.
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
FrNmRxPduContainsData	1	[ECUC_FrNm_00027]
FrNmRxPduContainsVote	1	[ECUC_FrNm_00026]
FrNmRxPduId	1	[ECUC_FrNm_00025]
FrNmRxPduRef	1	[ECUC_FrNm_00012]

No Included Containers

⌋

[ECUC_FrNm_00027] Definition of EcucBooleanParamDef FrNmRxPduContainsData ⌈

Parameter Name	FrNmRxPduContainsData		
Parent Container	FrNmRxPdu		
Description	This parameter defines if the PDU contains NM Data.		
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		

⌋

[ECUC_FrNm_00026] Definition of EcucBooleanParamDef FrNmRxPduContainsVote ⌈

Parameter Name	FrNmRxPduContainsVote		
Parent Container	FrNmRxPdu		
Description	This parameter defines if the PDU contains NM Vote information.		
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		

⌋

[ECUC_FrNm_00025] Definition of EcucIntegerParamDef FrNmRxPduId ⌈

Parameter Name	FrNmRxPduId	
Parent Container	FrNmRxPdu	
Description	PDU identifier configured for the respective FlexRay Channel. It is used for referring to the FlexRay Interface receive function. It must be consistent with the value configured in the FlexRay Interface. This ID is used for the combined reception of NM Vote and NM Data or for the reception of the NM Vote if NM Data is received in a separate PDU. ImplementationType: PduIdType	
Multiplicity	1	
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)	
Range	0 .. 65535	
Default value	–	

▽

△

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

⌋

[ECUC_FrNm_00012] Definition of EcucReferenceDef FrNmRxPduRef ⌈

Parameter Name	FrNmRxPduRef		
Parent Container	FrNmRxPdu		
Description	The reference to a PDU in the global PDU structure described in the AUTOSAR ECU Configuration Specification. This reference will be used by the LSduR module to derive the PDU Id.		
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		

⌋

10.4.6 FrNmTxPdu

[ECUC_FrNm_00009] Definition of EcucParamConfContainerDef FrNmTxPdu ⌈

Container Name	FrNmTxPdu
Parent Container	FrNmChannelIdentifiers
Description	This container describes the FlexRay NM TX PDU:s.
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
FrNmTxConfirmationPduId	1	[ECUC_FrNm_00018]
FrNmTxPduContainsData	1	[ECUC_FrNm_00024]
FrNmTxPduContainsVote	1	[ECUC_FrNm_00023]
FrNmTxPduRef	1	[ECUC_FrNm_00011]

No Included Containers

⌋

[ECUC_FrNm_00018] Definition of EcucIntegerParamDef FrNmTxConfirmationPduId ⌈

Parameter Name	FrNmTxConfirmationPduId		
Parent Container	FrNmTxPdu		
Description	Handle Id used by the Lower Layer when calling FrNm_TriggerTransmit() or FrNm_TxConfirmation().		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local withAuto = true		

⌋

[ECUC_FrNm_00024] Definition of EcucBooleanParamDef FrNmTxPduContainsData ⌈

Parameter Name	FrNmTxPduContainsData		
Parent Container	FrNmTxPdu		
Description	This parameted defines if the PDU contains NM Data.		
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		

⌋

[ECUC_FrNm_00023] Definition of EcucBooleanParamDef FrNmTxPduContainsVote ⌈

Parameter Name	FrNmTxPduContainsVote
Parent Container	FrNmTxPdu
Description	This parameted defines if the PDU contains NM Vote information.
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		

⌋

[ECUC_FrNm_00011] Definition of EcucReferenceDef FrNmTxPduRef ⌈

Parameter Name	FrNmTxPduRef		
Parent Container	FrNmTxPdu		
Description	The reference to a PDU in the global PDU structure described in the AUTOSAR ECU Configuration Specification. This reference is used to derive the PDU Id that is defined by the LSduR module.		
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		

⌋

10.4.7 FrNmUserDataTxPdu

[ECUC_FrNm_00055] Definition of EcucParamConfContainerDef FrNmUserDataTxPdu ⌈

Container Name	FrNmUserDataTxPdu
Parent Container	FrNmChannelIdentifiers
Description	This optional container is used to configure the UserNm PDU. This container is only available if FrNmComUserDataSupport is enabled.
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
FrNmTxUserDataPduId	1	[ECUC_FrNm_00056]
FrNmTxUserDataPduRef	1	[ECUC_FrNm_00057]

No Included Containers

⌋

[ECUC_FrNm_00056] Definition of EcucIntegerParamDef FrNmTxUserDataPduId ⌈

Parameter Name	FrNmTxUserDataPduId		
Parent Container	FrNmUserDataTxPdu		
Description	This parameter defines the Handle ID of the NM User Data I-PDU.		
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: local withAuto = true		

⌋

[ECUC_FrNm_00057] Definition of EcucReferenceDef FrNmTxUserDataPduRef ⌈

Parameter Name	FrNmTxUserDataPduRef		
Parent Container	FrNmUserDataTxPdu		
Description	Reference to the NM User Data I-PDU in the global PDU collection.		
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: local		

⌋

10.5 Published parameters

For details refer to the chapter 10.3 “Published Information” in [2].

10.6 Configuration constraints

[SWS_FrNm_00069] ⌈The following configuration constraints are conditionally recommended for FlexRay NM:

- NM_REPEAT_MESSAGE_TIME = 0 (conditions: (1) startup of all applications is completed as soon as the FlexRay communication is started and (2) node detection is not required in the FlexRay NM-cluster)
- `FrNmReadySleepCnt` = 0 (condition: bus communication is always shut down at the end of the NM Repetition Cycle in all nodes within the same FlexRay NM-cluster, even in presence of race conditions)

⌋

10.7 Examples

The following examples require FlexRay knowledge that is not described in the examples (e.g. the definition of a minislot). The FlexRay Communications System Specifications, V2.1 [7] contain the necessary information.

10.7.1 Example of Bus-Schedule with NM-Vote PDUs

Assume an example network of five nodes with the respective IDs of 1, 2, 3, 4 and 5 as shown in Figure 10.7.

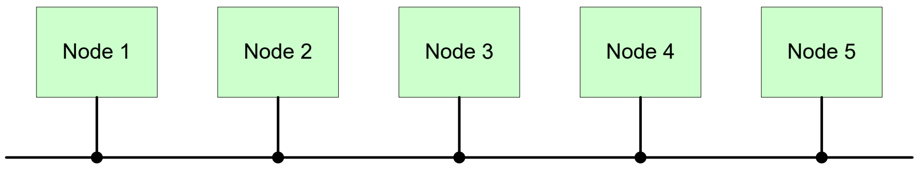

Figure 10.7: Example of five Node Network

The FlexRay Schedule allots 5 slots in the static segment and 6 (mini)slots in the dynamic segment as shown in Figure 10.8. To keep the example simple the nodes are assigned static numerically equivalent to the node numbers, e.g., Node 1 is sending in static Slot 1, Node2 is sending in static slot 2 and so on.

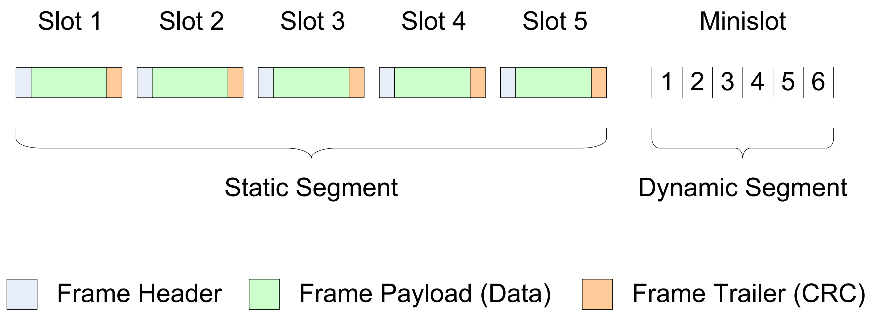

Figure 10.8: Example of Bus Schedule

Node 2 and 5 transmit their NM-Vote in the static segment, while the three remaining nodes 1, 3 and 4 transmit their NM-Vote in the dynamic segment as shown in Figure 10.9.

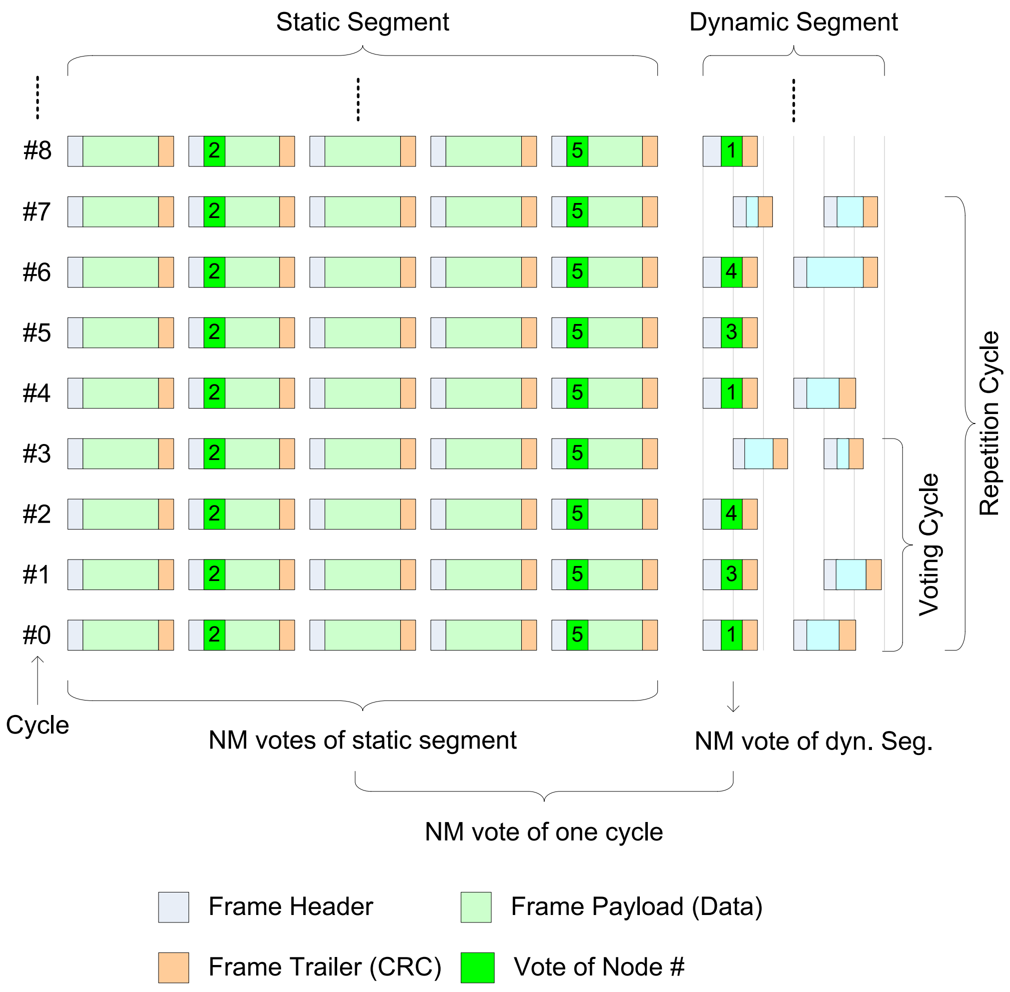

Figure 10.9: Example of NM-Vote in dynamic and static segment

As three dynamic voters exist, the Voting Cycle will be 4 and is repeated, therefore, every four cycles (cycle 0-1-2-3 and 4-5-6-7 and so on). Notice that no NM-Vote will be transmitted in last cycle of the Voting Cycle as all nodes have already voted.

In this example the Repetition Cycle is set to 2, which is twice the Voting cycle. Therefore every node will sent his vote twice.

10.7.2 Example of Bus Schedule with NM-Data PDUs

This example uses the same setup as in the previous example (10.7.1) - five nodes (Figure 10.7), 5 slot in the static segment and 6 slots in the dynamic segment (Figure 10.8).

Five Node need to send their NM-Data, which leads to a Data Cycle of "8", as only 1,2,4,8,16,32 and 64 are allowed due the restriction of the FlexRay Cycle multiplexing (see [SWS_FrNm_00195]).

Figure 10.10 shows that Node 1 will send its NM-Data in cycle 2, 10 and so on. Node 2, 3, 4 and 5 will behave in a similar way.

As the Data Cycle is "8", all nodes will send their NM-Data only every 8 cycles.

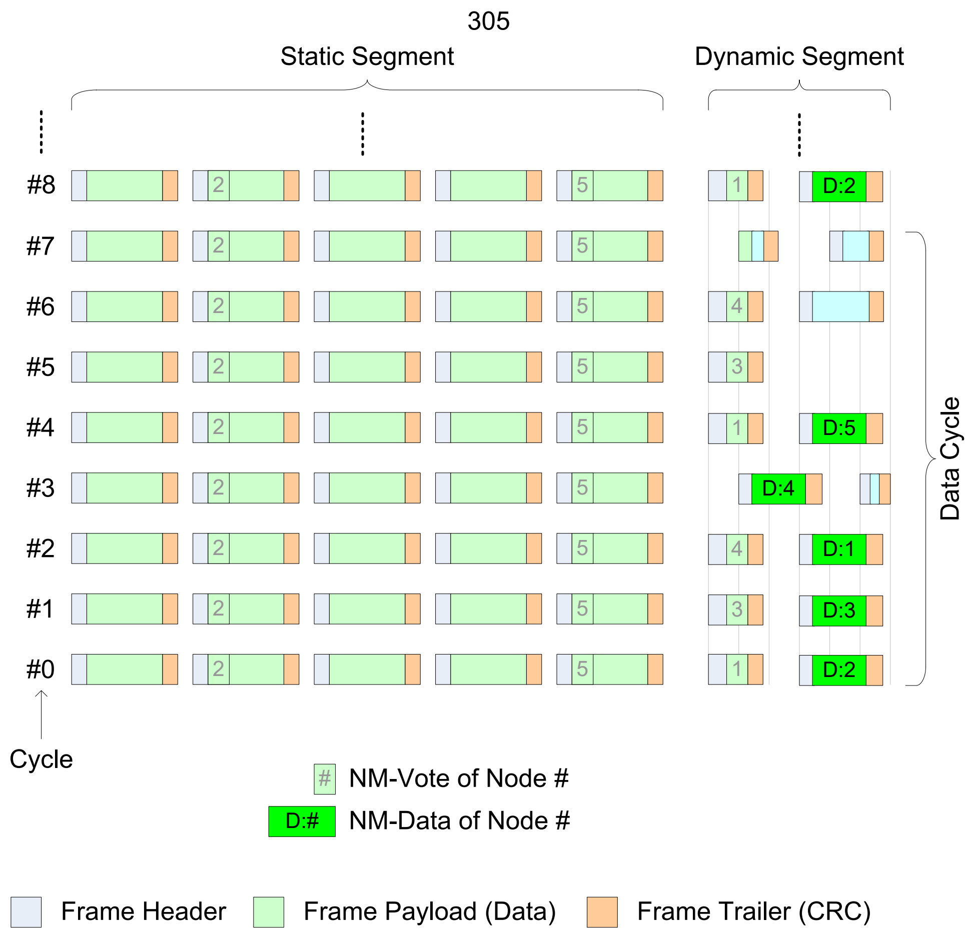

Figure 10.10: Example of Bus Schedule with NM-Data

A Not applicable requirements

[SWS_FrNm_NA_00000]

Upstream requirements: RS_Nm_00044, RS_Nm_00145, RS_Nm_00146

⌈This specification item references requirements that are not applicable, because it is no requirement against FrNm SWS or only against ECUC elements.⌋

[SWS_FrNm_NA_00001]

Upstream requirements: SRS_BSW_00375

⌈This specification item references requirements that are not applicable, because FrNm does not implement any interrupts, is not a driver or MCAL abstraction layer or has any direct access to OS.⌋

[SWS_FrNm_NA_00002]

Upstream requirements: SRS_BSW_00168, SRS_BSW_00423

⌈This specification item references requirements that are not applicable, because FrNm has no interdepencies to SW Components.⌋

[SWS_FrNm_NA_00003]

Upstream requirements: SRS_BSW_00426

⌈This specification item references requirements that are not applicable, because FrNm does not share any data with other BSW.⌋

[SWS_FrNm_NA_00004]

Upstream requirements: SRS_BSW_00427

⌈This specification item references requirements that are not applicable, because BSW module description template is not part of the FrNm SWS.⌋

[SWS_FrNm_NA_00005]

Upstream requirements: SRS_BSW_00336

⌈This specification item references requirements that are not applicable, because FrNm does not have any shutdown functionality.⌋

[SWS_FrNm_NA_00006]

Upstream requirements: SRS_BSW_00417

⌈This specification item references requirements that are not applicable, because FrNm does not report any DEM errors.⌋

[SWS_FrNm_NA_00007]

Upstream requirements: SRS_BSW_00433

⌈The following requirements are not applicable to this specification, because they are either general BSW requirements, which apply to all BSW modules and not only especially to the FrNm module or they are not applicable at all.⌋

B Change history of AUTOSAR traceable items

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

B.1 Traceable item history of this document according to AUTOSAR Release R24-11

B.1.1 Added Specification Items in R24-11

none

B.1.2 Changed Specification Items in R24-11

[SWS_FrNm_00010] [SWS_FrNm_00220] [SWS_FrNm_00251] [SWS_FrNm_00252] [SWS_FrNm_00256] [SWS_FrNm_00276] [SWS_FrNm_00277] [SWS_FrNm_00365] [SWS_FrNm_00460] [SWS_FrNm_00474] [SWS_FrNm_00564]

B.1.3 Deleted Specification Items in R24-11

none