Document Title	Specification of FlexRay Network Management
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	028
Document Classification	Standard

Document Status	Final
Part of AUTOSAR Release	4.2.2

Document Change History

Release	Changed by	Change Description
4.2.2	AUTOSAR Release Management	• Clarification on FrNmPassiveModeEnabled • Clarification on FrNmNumberOfClusters • Clarity on scheduling of MainFunction • Debugging support marked as obsolete • Minor corrections
4.2.1	AUTOSAR Release Management	• Correction of Partial Networking aggregation algorithm • Harmonize description of identical API's • Const usage consistent in specifications
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
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
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
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

Document Change History		
Release	**Changed by**	**Change Description**
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
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
3.1.2	AUTOSAR Administration	• Layout adaptations
3.1.1	AUTOSAR Administration	• Incorporation of core partner change requests for R3.0 • Legal disclaimer revised
3.0.1	AUTOSAR Administration	• Updated chapter 8/10 after changes in BSW UML model
2.0.16	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

Document Change History		
Release	**Changed by**	**Change Description**
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
1.0	AUTOSAR Administration	• Initial release

Disclaimer

Advice for users

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

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

Table of Contents

1 Introduction and functional overview

This document describes the concept, core functionality, optional features, interfaces and configuration issues of the AUTOSAR 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, abbreviations, and glossary

Acronym:	Description
CC	Communication Controller
NM	Network Management
WCET	Worst Case Execution Time
DET	Development Error Tracer. AUTOSAR Module for detection and reporting of errors during development.
DEM	Diagnostic Event Manager. AUTOSAR Module which is a sub-component of the diagnostic module within AUTOSAR. It is responsible for processing and storing diagnostic events (errors) and associated freeze frame data. Additionally, the DEM provides fault information to the DCM (e.g., read all stored DTCs from the error memory).
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

Abbreviation:	Description
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

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

3 Related documenation

3.1 Input documents

[1] List of Basic Software Modules
AUTOSAR_TR_BSWModuleList.pdf

[2] Layered Sofware Architecture,
AUTOSAR_EXP_LayeredSoftwareArchitecture.pdf

[3] General Requirements on Basic Software Modules
AUTOSAR_SRS_BSWGeneral.pdf

[4] Requirements on Network Management
AUTOSAR_SRS_NetworkManagement.pdf

[5] General Specification of Basic Software Modules
AUTOSAR_SWS_BSWGeneral.pdf

3.2 Related standards and norms

[6] FlexRay Communications System Specifications, V2.1

3.3 Related AUTOSAR documents

[7] AUTOSAR Specification of Communication Manager
AUTOSAR_SWS_COMManager.pdf

[8] Specification of Generic Network Management Interface
AUTOSAR_SWS_NetworkManagement.pdf

[9] Specification of FlexRay Interface
AUTOSAR_SWS _FlexRayInterface. pdf

[10] Specification of ECU State Manager
AUTOSAR_SWS_ECUStateManager.pdf

[11] Specification of Diagnostic Event Manager
AUTOSAR_SWS_DiagnosticEventManager.pdf

[12] Specification of Development Error Tracer
AUTOSAR_SWS_DevelopmentErrorTracer.pdf

[13] Specification of Standard Types
AUTOSAR_SWS_StandardTypes.pdf

[14] Specification of Platform Types
AUTOSAR_SWS_PlatformTypes.pdf

[15] Specification of Compiler Abstraction
AUTOSAR_SWS_CompilerAbstraction.pdf

[16] Specification of Operating System
AUTOSAR_SWS_OS.pdf

[17] Specification of FlexRay State Manager
AUTOSAR_SWS_FlexRayStateManager.pdf

AUTOSAR provides a General Specification on Basic Software modules [5] (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.

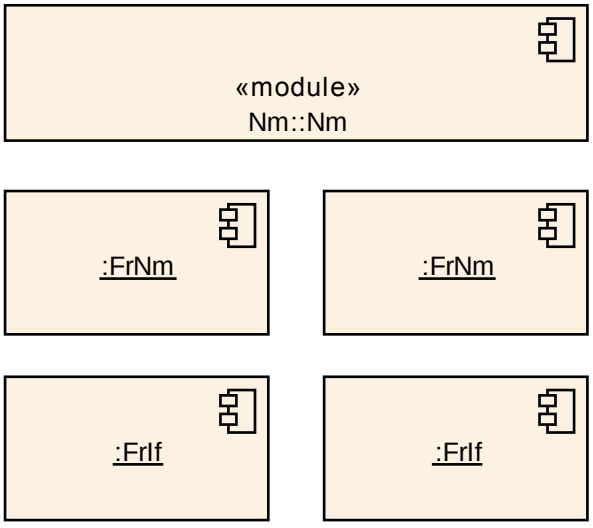


Figure 4-1 AUTOSAR NM Stack on FlexRay

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

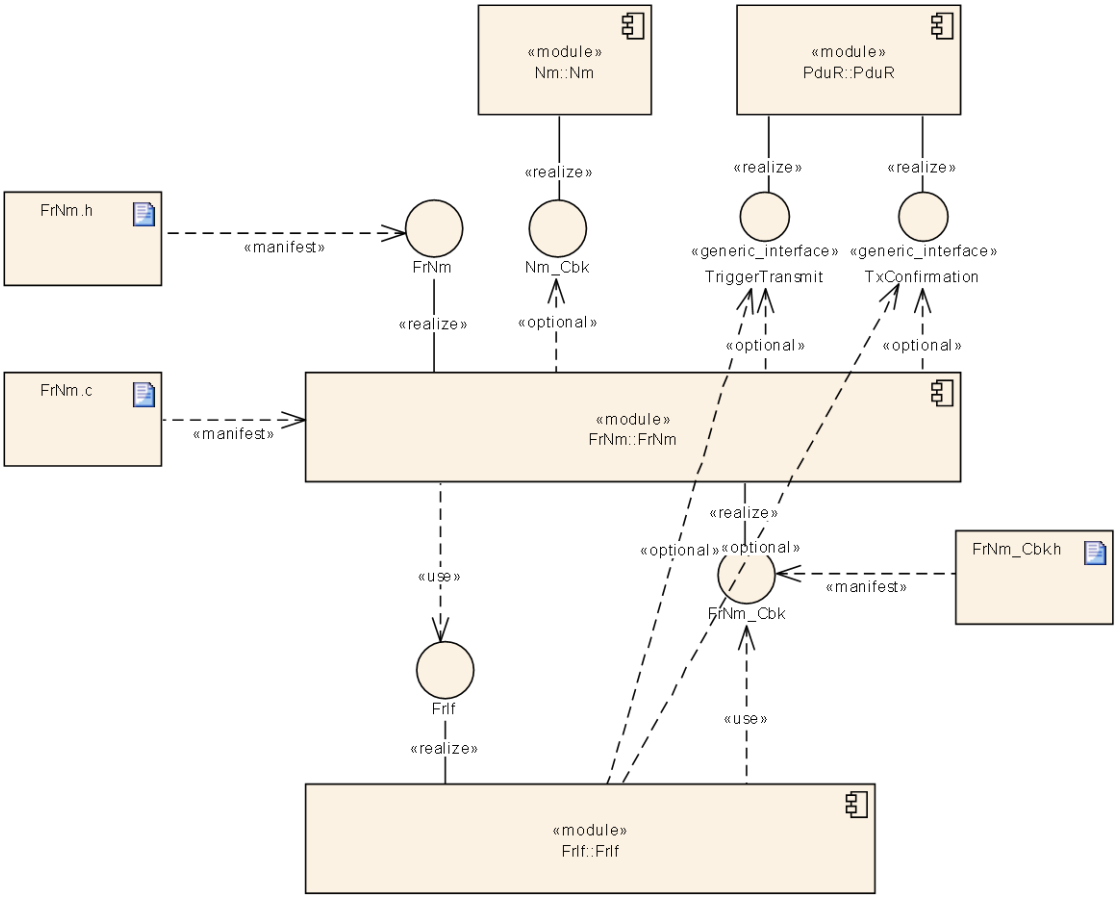


Figure 5-1 NM Overview

Note: In addition to the modules depicted in Figure 5-1 (oben), FlexRay Nm uses some additional modules (like the DET). A complete list can be found in 5.1.2.

[SWS_FrNm_00220] ⌈The FlexRay NM shall use only OS objects and/or related OS services

according to the table defined in [16]⌋ (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
FrIf	FlexRay NM uses services of the FlexRay Interface.

Det	The FlexRay Network Management informs the Development Error Tracer on development errors
Dem	Dem gets production error information from FlexRay Network Management.
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.*

5.1.2 Header file structure

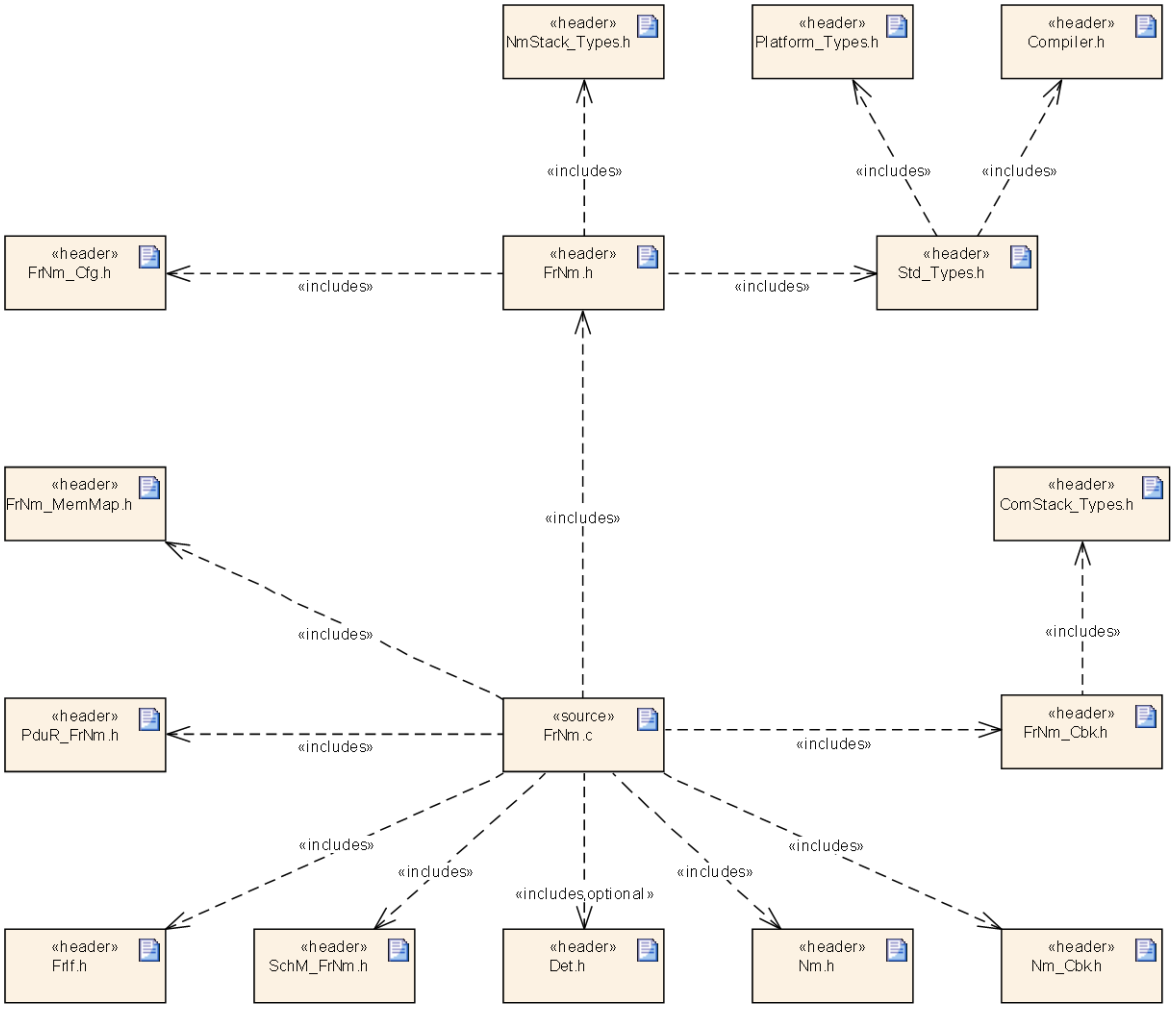


Figure 5-2 Header File Structure

[SWS_FrNm_00367] ⌈The file FrNm.h contains declaration of provided interface functions. ⌋ ( )

[SWS_FrNm_00066] ⌈The following header files shall be included within the FrNm module.

Std_Types.h (for AUTOSAR standard types),
FrIf.h (for interface of FlexRay Interface),
PduR_FrNm.h (for interface of PduR),
Nm_Cbk.h (for callbacks of NmIf),
Det.h (for interface of DET – optional, included only if Det is configured),

NmStack_Types.h (for common NMIF types)
Nm.h (for needed interface functions),
SchM_FrNm.h (for interface to Schedule Manager),
FrNm_MemMap.h (for locating software in memory using the memory map),
⌋ ( )

6 Requirements traceability

Requirement	Description	Satisfied by
-	-	SWS_FrNm_00006
-	-	SWS_FrNm_00010
-	-	SWS_FrNm_00013
-	-	SWS_FrNm_00030
-	-	SWS_FrNm_00042
-	-	SWS_FrNm_00045
-	-	SWS_FrNm_00046
-	-	SWS_FrNm_00051
-	-	SWS_FrNm_00055
-	-	SWS_FrNm_00058
-	-	SWS_FrNm_00066
-	-	SWS_FrNm_00069
-	-	SWS_FrNm_00076
-	-	SWS_FrNm_00100
-	-	SWS_FrNm_00102
-	-	SWS_FrNm_00103
-	-	SWS_FrNm_00105
-	-	SWS_FrNm_00107
-	-	SWS_FrNm_00108
-	-	SWS_FrNm_00110
-	-	SWS_FrNm_00115
-	-	SWS_FrNm_00116
-	-	SWS_FrNm_00117
-	-	SWS_FrNm_00120
-	-	SWS_FrNm_00121
-	-	SWS_FrNm_00122
-	-	SWS_FrNm_00123
-	-	SWS_FrNm_00124
-	-	SWS_FrNm_00125
-	-	SWS_FrNm_00126
-	-	SWS_FrNm_00129
-	-	SWS_FrNm_00130
-	-	SWS_FrNm_00131
-	-	SWS_FrNm_00134
-	-	SWS_FrNm_00135
-	-	SWS_FrNm_00136

-	-	SWS_FrNm_00137
-	-	SWS_FrNm_00138
-	-	SWS_FrNm_00143
-	-	SWS_FrNm_00147
-	-	SWS_FrNm_00148
-	-	SWS_FrNm_00151
-	-	SWS_FrNm_00155
-	-	SWS_FrNm_00156
-	-	SWS_FrNm_00160
-	-	SWS_FrNm_00161
-	-	SWS_FrNm_00162
-	-	SWS_FrNm_00169
-	-	SWS_FrNm_00170
-	-	SWS_FrNm_00172
-	-	SWS_FrNm_00174
-	-	SWS_FrNm_00175
-	-	SWS_FrNm_00186
-	-	SWS_FrNm_00188
-	-	SWS_FrNm_00189
-	-	SWS_FrNm_00190
-	-	SWS_FrNm_00191
-	-	SWS_FrNm_00192
-	-	SWS_FrNm_00193
-	-	SWS_FrNm_00194
-	-	SWS_FrNm_00195
-	-	SWS_FrNm_00196
-	-	SWS_FrNm_00205
-	-	SWS_FrNm_00215
-	-	SWS_FrNm_00216
-	-	SWS_FrNm_00226
-	-	SWS_FrNm_00228
-	-	SWS_FrNm_00229
-	-	SWS_FrNm_00230
-	-	SWS_FrNm_00231
-	-	SWS_FrNm_00234
-	-	SWS_FrNm_00235
-	-	SWS_FrNm_00236
-	-	SWS_FrNm_00237
-	-	SWS_FrNm_00238

-	-	SWS_FrNm_00239
-	-	SWS_FrNm_00240
-	-	SWS_FrNm_00241
-	-	SWS_FrNm_00242
-	-	SWS_FrNm_00243
-	-	SWS_FrNm_00244
-	-	SWS_FrNm_00245
-	-	SWS_FrNm_00246
-	-	SWS_FrNm_00247
-	-	SWS_FrNm_00248
-	-	SWS_FrNm_00249
-	-	SWS_FrNm_00251
-	-	SWS_FrNm_00252
-	-	SWS_FrNm_00256
-	-	SWS_FrNm_00257
-	-	SWS_FrNm_00258
-	-	SWS_FrNm_00260
-	-	SWS_FrNm_00261
-	-	SWS_FrNm_00262
-	-	SWS_FrNm_00263
-	-	SWS_FrNm_00264
-	-	SWS_FrNm_00265
-	-	SWS_FrNm_00266
-	-	SWS_FrNm_00267
-	-	SWS_FrNm_00268
-	-	SWS_FrNm_00269
-	-	SWS_FrNm_00270
-	-	SWS_FrNm_00271
-	-	SWS_FrNm_00276
-	-	SWS_FrNm_00277
-	-	SWS_FrNm_00280
-	-	SWS_FrNm_00283
-	-	SWS_FrNm_00290
-	-	SWS_FrNm_00291
-	-	SWS_FrNm_00292
-	-	SWS_FrNm_00297
-	-	SWS_FrNm_00298
-	-	SWS_FrNm_00307
-	-	SWS_FrNm_00308

-	-	SWS_FrNm_00311
-	-	SWS_FrNm_00313
-	-	SWS_FrNm_00316
-	-	SWS_FrNm_00317
-	-	SWS_FrNm_00319
-	-	SWS_FrNm_00322
-	-	SWS_FrNm_00323
-	-	SWS_FrNm_00324
-	-	SWS_FrNm_00335
-	-	SWS_FrNm_00338
-	-	SWS_FrNm_00340
-	-	SWS_FrNm_00342
-	-	SWS_FrNm_00344
-	-	SWS_FrNm_00349
-	-	SWS_FrNm_00356
-	-	SWS_FrNm_00357
-	-	SWS_FrNm_00364
-	-	SWS_FrNm_00365
-	-	SWS_FrNm_00366
-	-	SWS_FrNm_00367
-	-	SWS_FrNm_00375
-	-	SWS_FrNm_00376
-	-	SWS_FrNm_00378
-	-	SWS_FrNm_00379
-	-	SWS_FrNm_00381
-	-	SWS_FrNm_00384
-	-	SWS_FrNm_00387
-	-	SWS_FrNm_00388
-	-	SWS_FrNm_00389
-	-	SWS_FrNm_00390
-	-	SWS_FrNm_00391
-	-	SWS_FrNm_00392
-	-	SWS_FrNm_00393
-	-	SWS_FrNm_00394
-	-	SWS_FrNm_00396
-	-	SWS_FrNm_00397
-	-	SWS_FrNm_00398
-	-	SWS_FrNm_00402
-	-	SWS_FrNm_00404

-	-	SWS_FrNm_00405
-	-	SWS_FrNm_00406
-	-	SWS_FrNm_00407
-	-	SWS_FrNm_00408
-	-	SWS_FrNm_00409
-	-	SWS_FrNm_00410
-	-	SWS_FrNm_00411
-	-	SWS_FrNm_00412
-	-	SWS_FrNm_00416
-	-	SWS_FrNm_00418
-	-	SWS_FrNm_00420
-	-	SWS_FrNm_00421
-	-	SWS_FrNm_00423
-	-	SWS_FrNm_00424
-	-	SWS_FrNm_00425
-	-	SWS_FrNm_00426
-	-	SWS_FrNm_00428
-	-	SWS_FrNm_00429
-	-	SWS_FrNm_00432
-	-	SWS_FrNm_00434
-	-	SWS_FrNm_00435
-	-	SWS_FrNm_00436
-	-	SWS_FrNm_00437
-	-	SWS_FrNm_00438
-	-	SWS_FrNm_00447
-	-	SWS_FrNm_00452
-	-	SWS_FrNm_00453
-	-	SWS_FrNm_00454
-	-	SWS_FrNm_00457
-	-	SWS_FrNm_00460
-	-	SWS_FrNm_00461
-	-	SWS_FrNm_00467
-	-	SWS_FrNm_00470
-	-	SWS_FrNm_00471
-	-	SWS_FrNm_00472
-	-	SWS_FrNm_00473
-	-	SWS_FrNm_00474
BSW00431	-	SWS_FrNm_00401
BSW00434	-	SWS_FrNm_00401

BSW136	-	SWS_FrNm_00401
BSW139	-	SWS_FrNm_00401
BSW140	-	SWS_FrNm_00401
SRS_BSW_00005	Modules of the ÂµC Abstraction Layer (MCAL) may not have hard coded horizontal interfaces	SWS_FrNm_00401
SRS_BSW_00009	All Basic SW Modules shall be documented according to a common standard.	SWS_FrNm_00401
SRS_BSW_00010	The memory consumption of all Basic SW Modules shall be documented for a defined configuration for all supported platforms.	SWS_FrNm_00401
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_00161	The AUTOSAR Basic Software shall provide a microcontroller abstraction layer which provides a standardized interface to higher software layers	SWS_FrNm_00401
SRS_BSW_00162	The AUTOSAR Basic Software shall provide a hardware abstraction layer	SWS_FrNm_00401
SRS_BSW_00164	The Implementation of interrupt service routines shall be done by the Operating System, complex drivers or modules	SWS_FrNm_00401
SRS_BSW_00168	SW components shall be tested by a function defined in a common API in the Basis-SW	SWS_FrNm_00401
SRS_BSW_00172	The scheduling strategy that is built inside the Basic Software Modules shall be compatible with the strategy used in the system	SWS_FrNm_00401
SRS_BSW_00314	All internal driver modules shall separate the interrupt frame definition from the service routine	SWS_FrNm_00401
SRS_BSW_00325	The runtime of interrupt service routines and functions that are running in interrupt context shall be kept short	SWS_FrNm_00401
SRS_BSW_00326	-	SWS_FrNm_00401
SRS_BSW_00331	All Basic Software Modules shall strictly separate error and status information	SWS_FrNm_00021
SRS_BSW_00333	For each callback function it shall be specified if it is called from interrupt context or not	SWS_FrNm_00401
SRS_BSW_00334	All Basic Software Modules shall provide an XML file that contains the meta data	SWS_FrNm_00401
SRS_BSW_00336	Basic SW module shall be able to shutdown	SWS_FrNm_00401
SRS_BSW_00337	Classification of development errors	SWS_FrNm_00021
SRS_BSW_00341	Module documentation shall contains all needed informations	SWS_FrNm_00401
SRS_BSW_00347	A Naming seperation of different instances of	SWS_FrNm_00401

	BSW drivers shall be in place	
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_00375	Basic Software Modules shall report wake-up reasons	SWS_FrNm_00401
SRS_BSW_00401	Documentation of multiple instances of configuration parameters shall be available	SWS_FrNm_00401
SRS_BSW_00406	A static status variable denoting if a BSW module is initialized shall be initialized with value 0 before any APIs of the BSW module is called	SWS_FrNm_00050, SWS_FrNm_00071, SWS_FrNm_00073
SRS_BSW_00407	Each BSW module shall provide a function to read out the version information of a dedicated module implementation	SWS_FrNm_00050
SRS_BSW_00416	The sequence of modules to be initialized shall be configurable	SWS_FrNm_00050
SRS_BSW_00417	Software which is not part of the SW-C shall report error events only after the DEM is fully operational.	SWS_FrNm_00401
SRS_BSW_00423	BSW modules with AUTOSAR interfaces shall be describable with the means of the SW-C Template	SWS_FrNm_00401
SRS_BSW_00426	BSW Modules shall ensure data consistency of data which is shared between BSW modules	SWS_FrNm_00401
SRS_BSW_00427	ISR functions shall be defined and documented in the BSW module description template	SWS_FrNm_00401
SRS_BSW_00429	BSW modules shall be only allowed to use OS objects and/or related OS services	SWS_FrNm_00220
SRS_BSW_00432	Modules should have separate main processing functions for read/receive and write/transmit data path	SWS_FrNm_00255, SWS_FrNm_00449
SRS_BSW_00433	Main processing functions are only allowed to be called from task bodies provided by the BSW Scheduler	SWS_FrNm_00401
SRS_Nm_00044	The NM shall be applicable to different types of communication systems which are in the scope of Autosar and support a bus sleep mode.	SWS_FrNm_00401
SRS_Nm_00045	NM has to provide services to coordinate shutdown of NM-clusters independently of each other	SWS_FrNm_00034
SRS_Nm_00046	It shall be possible to trigger the startup of all Nodes at any Point in Time.	SWS_FrNm_00168
SRS_Nm_00048	NM shall put the communication controller into sleep mode if there is no bus communication	SWS_FrNm_00101
SRS_Nm_00050	The NM shall provide the current state of NM	SWS_FrNm_00104

SRS_Nm_00051	NM shall inform application when NM state changes occur.	SWS_FrNm_00106
SRS_Nm_00052	The NM interface shall signal to the application that all other ECUs are ready to sleep.	SWS_FrNm_00181
SRS_Nm_00053	NM on a node which is or become bus unavailable shall have a deterministic Behavior	SWS_FrNm_00035
SRS_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
SRS_Nm_00137	NM shall perform communication system error handling for errors that have impact on the NM behavior.	SWS_FrNm_00035
SRS_Nm_00144	NM shall support communication clusters of up to 64 ECUs	SWS_FrNm_00179
SRS_Nm_00145	On a properly configured node, NM shall tolerate a loss of a predefined number of NM messages	SWS_FrNm_00401
SRS_Nm_00146	The NM shall tolerate a time jitter of NM messages in one or more ECUs	SWS_FrNm_00401
SRS_Nm_00147	The NM algorithm shall be processor independent.	SWS_FrNm_00225
SRS_Nm_00148	The specification and implementation shall be split-up into a communication system independent and communication system dependent parts	SWS_FrNm_00401
SRS_Nm_00149	The timing of NM shall be configurable.	SWS_FrNm_00036
SRS_Nm_00150	Specific functions of the Network Management shall be statically configurable at pre-compile time	SWS_FrNm_00077, SWS_FrNm_00179, SWS_FrNm_00180, SWS_FrNm_00187
SRS_Nm_00154	The Network Management API shall be independent from the communication bus	SWS_FrNm_00034
SRS_Nm_02503	The NM API shall optionally give the possibility to send user data	SWS_FrNm_00043
SRS_Nm_02504	The NM API shall optionally give the possibility to get user data	SWS_FrNm_00044
SRS_Nm_02505	The NM shall optionally set the local node identifier to the NM-message	SWS_FrNm_00222
SRS_Nm_02506	The NM API shall give the possibility to read the source node identifier of the sender	SWS_FrNm_00047
SRS_Nm_02508	Every node shall have associated with it a node identifier that is unique in the NM-cluster	SWS_FrNm_00037
SRS_Nm_02509	The NM interface shall signal to the application that at least one other ECUs is not ready to sleep anymore.	SWS_FrNm_00185
SRS_Nm_02510	For CAN NM it shall be optionally possible to immediately transmit the confirmation	SWS_FrNm_00401

SRS_Nm_02511	It shall be possible to configure the Network Management of a node in Cluster Shutdown	SWS_FrNm_00187
SRS_Nm_02524	Nm shall communicate EIRA and ERA requests to the upper layers using virtual PDUs	SWS_FrNm_00359

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 [10], [17]).

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 Seite 47.

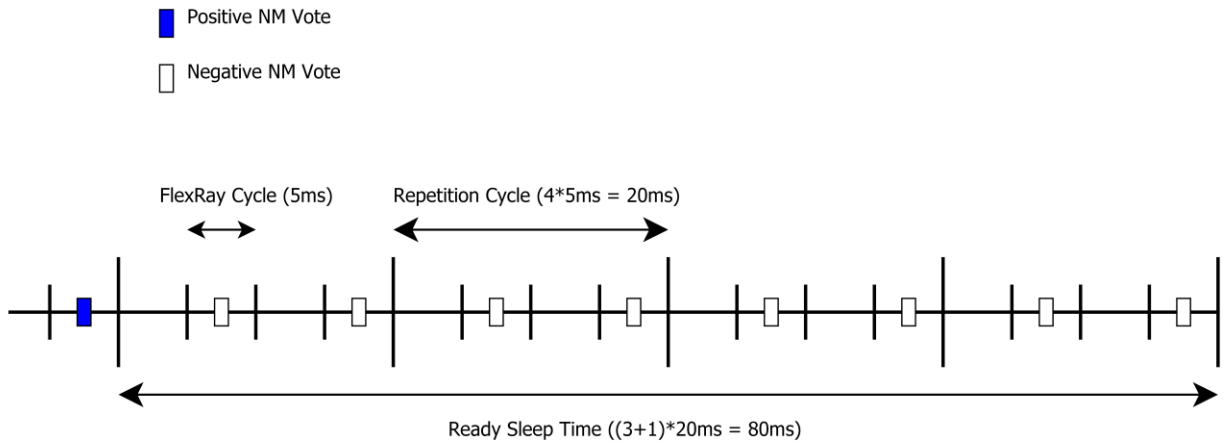
The main concept of the AUTOSAR FlexRay NM coordination algorithm can be defined by the following key-requirements:

[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] ⌈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. ⌋(SRS_Nm_00048, SRS_Nm_00054)

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



[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] ⌈The FrNm module shall synchronize state changes in the FlexRay NM state machine with the FlexRay periodic Schedule. ⌋(SRS_Nm_00046)

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 [6]). 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. ⌋(SRS_Nm_00147)

7.2 Operational modes

In this chapter, the operational modes of the AUTOSAR FlexRay NM coordination algorithm are described in detail. Figure 7-1 (in chapter 7.4 on page 34) 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] ⌈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`**. ⌋(SRS_Nm_00051)

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. ⌋ ( )

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 Seite 56.

[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. ⌋ ( )

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 'FrNmRepeastMessageTime'**). ⌋ ( )

[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. (see SWS_FrNm_00114). ⌋ ( )

[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. ⌋ ( )

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] ⌈The FlexRay NM module shall leave the Normal Operation State and enter the Repeat Message State if a Repeat Message Request is active and if a Repetition Cycle is completed. 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. ⌋ ( )

[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] ⌈The FlexRay NM module shall leave the Ready Sleep State and enter the Repeat Message State at the end of a NM Repetition Cycle if Ready Sleep Time has not expired and 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_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_129.

[SWS_FrNm_00444] If FlexRay NM is in Ready Sleep state, the configuration parameter **FRNM_CYCLE_COUNTER_EMULATION** 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 **FRNM_CYCLE_COUNTER_EMULATION** 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 **FRNM_CYCLE_COUNTER_EMULATION** 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 **FRNM_CYCLE_COUNTER_EMULATION** 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.

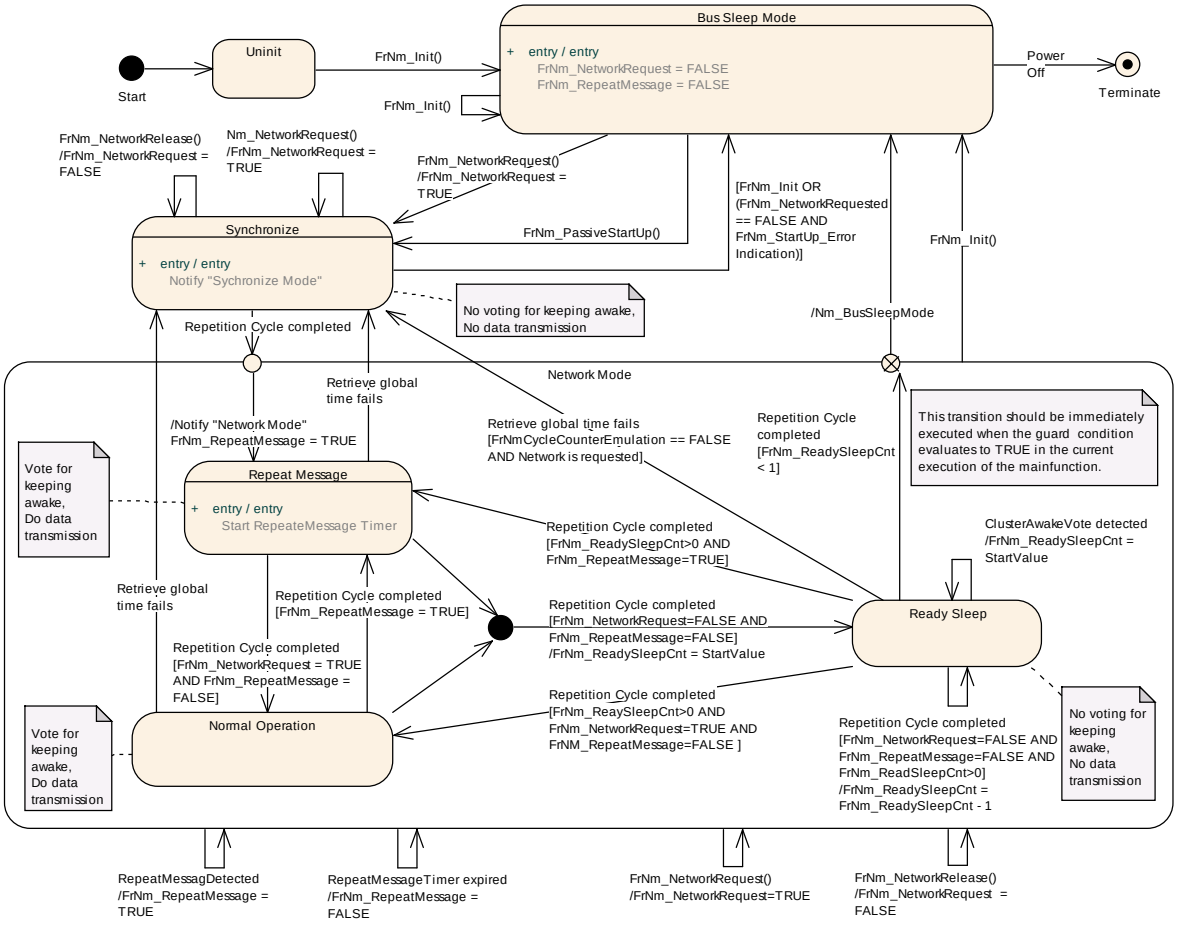


Figure 7-1 UML State Chart

7.5 Initialization and Startup

[SWS_FrNm_00071] ⌈The FrNm module shall store the initialization status in a private variable. ⌋(SRS_BSW_00406)

"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.⌋( )

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 [6]).
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 [6]). 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 **`FRNM_PDU_SCHEDULE_VARIANT`** 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

[SWS_FrNm_00006] ⌈FlexRay NM-Data PDU format shall be defined as follows: The figure below shows an example for 8 bytes PDU length

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 7	User data 5							
Byte 6	User data 4							
Byte 5	User data 3							
Byte 4	User data 2							
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

⌋( )

[SWS_FrNm_00076] ⌈The support of the Control Bit Vector by the FrNm module shall be configurable at pre-compile time by the configuration parameter **FrNmControlBitVectorEnabled** (see chapter 10). ⌋( )

[SWS_FrNm_00222] ⌈The support of the Source Node Identifier by the FrNm module shall be configurable at pre-compile time by the configuration parameter **FRNM_SOURCE_NODE_IDENTIFIER_ENABLED** (see chapter 10). ⌋(SRS_Nm_02505)

[SWS_FrNm_00381] ⌈If the parameter **FRNM_SOURCE_NODE_IDENTIFIER_ENABLED** (see chapter 10) is set to FALSE, this byte shall be used for user data. ⌋( )

Note: The FrNmPduLength is defined by the PduLength parameter [EcuC003_Conf] in the "global" ECUC module (see Ecu Configuration specification).

[SWS_FrNm_00313] ⌈The ability of the FrNm module to send NM-Data PDU shall be configurable at pre-compile time by the configuration parameter **FRNM_USER_DATA_ENABLED** (see chapter 10). ⌋( )

[SWS_FrNm_00155] ⌈The difference between applied standardized bytes and the NM-Data PDU length in the FrNm module shall be user data. ⌋( )

[SWS_FrNm_00324] ⌈If **FrNmControlBitVectorEnabled** is set to FALSE, Control Bit Vector of the NM-Data PDU shall not be used for other Data. It remains reserved. ⌋( )

[SWS_FrNm_00457] ⌈If **FrNmControlBitVectorEnabled** is set to FALSE, then Control Bit Vector of the NM-Data PDU shall be set to 0x00. ⌋( )

[SWS_FrNm_00156] ⌈The NM-Data PDU Control Bit Vector of the FrNm module format shall be defined as follows:

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte	Set to “0”	Partial Network Information Bit	Res	Active Wakeup Bit	NM Coordinator Sleep Ready	Res	Res	RptMsg Request

Table 7-2 Control Bit Vector Format」( )

[SWS_FrNm_00055] ⌈The Control Bit Vector of the FrNm module shall contain a Repeat Message Request Bit (RptMsgRequest) with the following meaning:
0: Repeat Message State not requested
1: Repeat Message State requested
The Control Bit Vector of the FlexRay NM module shall contain an Active Wakeup Bit with the following meaning.
0: Node has not woken up the network
1: Node has woken up the network
The Control Bit Vector of the FlexRay NM module shall contain a Partial Network Information Bit (PNI)
0: NM message contains no Partial Network request information
1: NM message contains Partial Network request information
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) 」( )

Hint: In AutoSAR Release 3.2 Bit 1 and 2 of the CBV are used for the NM-Coordinatior ID.

[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

[SWS_FrNm_00215] ⌈The NM-Vote PDU format of the FrNm module shall be defined as follows:

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0	Vote	Set to “0”						

Table 7-3 NM-Vote PDU Format

」( )

[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 Seite 56) 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 7-4 unterhalb. 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	Res	Active Wakeup Bit	NM Coordinator Sleep Ready	Res	Res	RptMsg Request
+								
NM-Vote PDU	Vote	Set to “0”						
Combined CBV and Vote	Vote	Partial Network Information Bit	Res	Active Wakeup Bit	NM Coordinator Sleep Ready	Res	Res	RptMsg Request

Table 7-4 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] ⌈The set of the Source Node Identifier shall be configurable using the configuration parameter **`FrNmNodeId`** (see chapter 10).⌋(SRS_Nm_02508)

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_< FrNmChannelIdRef >) has to be executed within a specific time window as shown in Figure 7-2. 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.

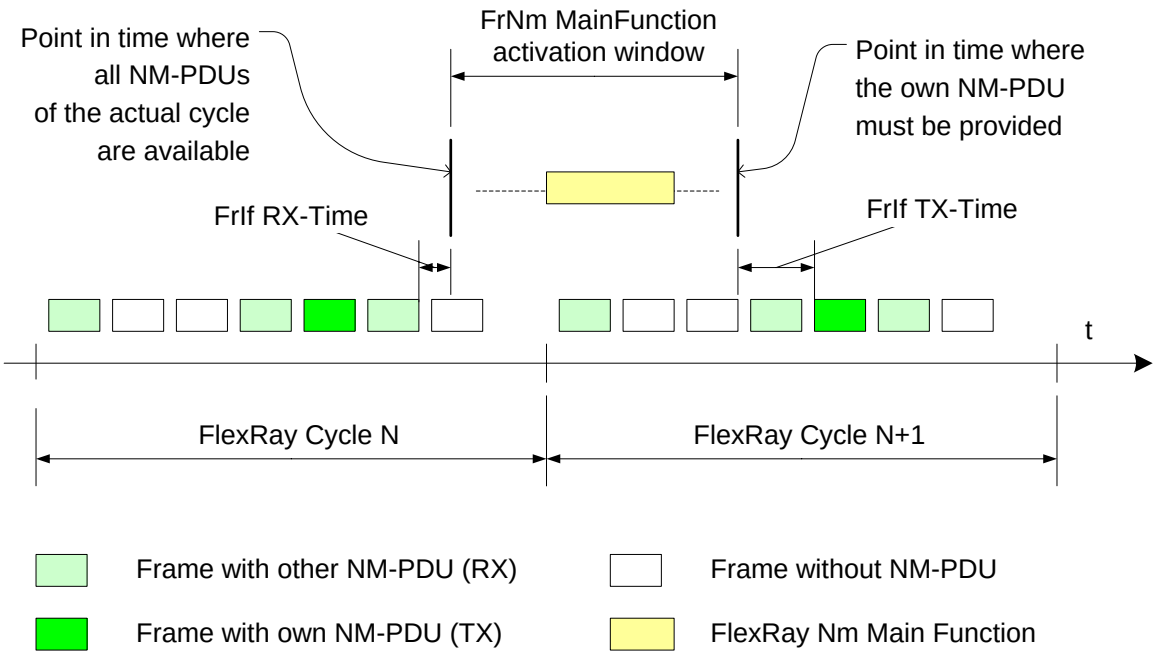


Figure 7-2 FlexRay NM Mainfunction Execution

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). The time duration denoted as FrIf TX-Time is the time duration between the FrNm's call to **`FrIf_Transmit`** 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 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 shall call the FrNm module function **`FrNm_TxConfirmation`** when a NM-message is successfully transmitted and a transmit confirmation is supported and configured within the FlexRay Interface.

Note: The FlexRay Interface module shall call 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_< FrNmChannelIdRef >) 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 7-2. 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

[2] part of the RTE module

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 [16] 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_< FrNmChannelIdRef >.⌋ ( )

[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 (see Figure 7-3). ⌋ ( )

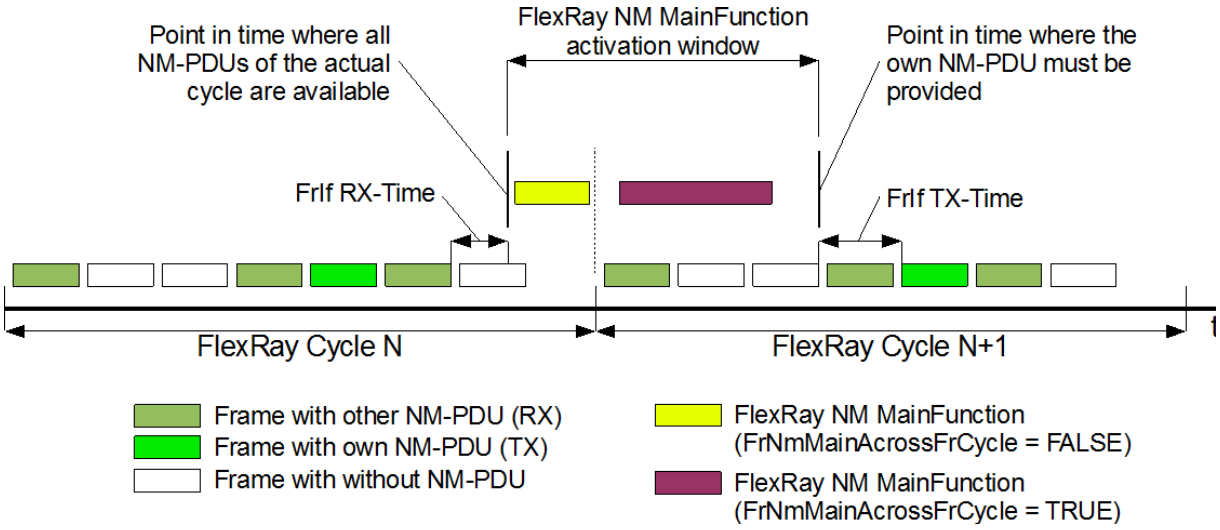


Figure 7-3 FlexRay NM MainFunction Execution – Setting of FrNmMainAcrossFrCycle

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 (see Figure 7-1) shall be subject to the following rules:

a) 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.

b) 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 a): 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 b): 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] ⌈The AUTOSAR FlexRay NM algorithm shall support up to 64 nodes per NM-Cluster.⌋(SRS_Nm_00150, SRS_Nm_00144)

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] ⌈The detection of remote sleep indication by the FrNm module shall be configurable at pre-compile time by the configuration parameter **FrNmRemoteSleepIndicationEnabled** (see chapter 10).⌋(SRS_Nm_00150)

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] ⌈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.` ⌋(SRS_Nm_00052)

[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_SyncrhonizationPoint`** provided to the NmIf module only at the repetition cycle boundaries, when the FrNm module is in Network Mode, and when the configuration parameter

`FRNM_SYNCHRONIZATIONPOINT_ENABLED` is set to TRUE. ⌋( )

[SWS_FrNm_00323] ⌈`FRNM_SYNCHRONIZATIONPOINT_ENABLED` is allowed to be set to true only when `FRNM_REMOTE_SLEEP_INDICATION_ENABLED` is set to TRUE. ⌋ ( )

7.8.3 Detection of Nodes (optional)

Nodes that have the Node Detection Feature enabled (SWS_FrNm_00170) 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.

[SWS_FrNm_00170] ⌈The support of node detection for the FrNm module shall be configurable at pre-compile time by the configuration parameter `FrNmNodeDetectionEnabled` (see chapter 10). `FrNmSourceNodeIdentifierEnabled` should also be set as ON to use this feature.. ⌋ ( )

7.8.4 User data (optional)

[SWS_FrNm_00077] ⌈The support for user data in the FrNm module shall be configurable at pre-compile time by the configuration parameter `FrNmUserDataEnabled` (see chapter 10). ⌋ (SRS_Nm_00150)

[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 `FrNmComUserDataEnabled` is set to 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_00394] ⌈If `FrNmComUserDataEnabled` is set to TRUE FrNm shall call `PduR_FrNmTriggerTransmit` within the function `FrNm_TriggerTransmit` called by the FrIf.⌋( )

[SWS_FrNm_00365] ⌈If `FrNmComUserDataEnabled` is set to TRUE the FrNm shall call "PduR_FrNmTxConfirmation" within the message transmission confirmation function "FrNm_TxConfirmation" called by the FrIf.⌋( )

[SWS_FrNm_00366] ⌈If `FrNmComUserDataEnabled` is set to TRUE the FrNm implementation shall provide an API FrNm_Transmit (see SWS_FrNm_00401). 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] ⌈The Passive Mode Configuration shall be configurable at pre-compile time by the configuration parameter **FRNM_PASSIVE_MODE_ENABLED** (see chapter 10). ⌋(SRS_Nm_00150, SRS_Nm_02511)

[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** (see chapter 10). ⌋( )

[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** (see chapter 10). ⌋( )

[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** (see chapter 10). ⌋ ( )

[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. ⌋ ( )

[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_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.

[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.

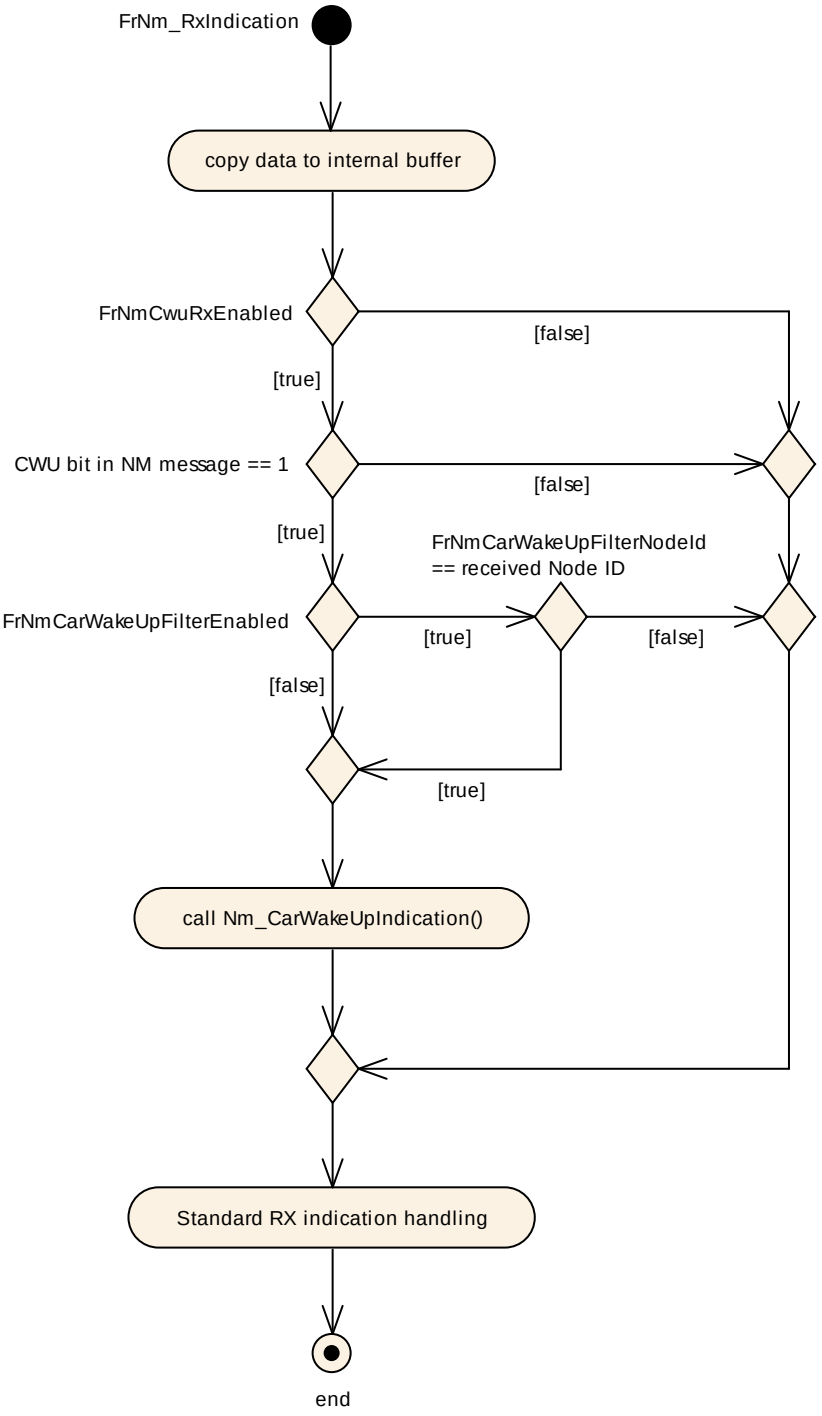


Figure 7-4 CarWakeUp indication handling

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.

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

[SWS_FrNm_00405] ⌈The FrNm extensions for partial networking are enabled with the configuration switch `FrNmPnEnabled`. ⌋( )

[SWS_FrNm_00404] ⌈FrNm shall be able to distinguish between an NM PDU without PN request information and an NM PDU with PN request information contained in the NM user data. 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 PN request information
1 -> NM PDU contains PN request information⌋( )

[SWS_FrNm_00453]⌈The range of the PN request information in the NM user data is defined by the parameters `FrNmPnInfoOffset` and `FrNmPnInfoLength`.
Each bit in the PN range shall have the following meaning:
Bit value Meaning:
0 -> The PN is not requested
1 -> The PN is requested

Note: `FrNmPnInfoOffset` + `FrNmPnInfoLength` shall not exceed the Length of the NM PDU.⌋( )

[SWS_FrNm_00418] ⌈By means of the configuration parameter `FrNmPnFilterMaskByte` the FrNm is able to detect which PN request information is relevant for the ECU and which not. Each bit of `FrNmPnFilterMaskByte` has the following meaning:
0 -> The PN request information is irrelevant for the ECU.
1 -> The PN request information is relevant for the ECU.⌋( )

[SWS_FrNm_00416] ⌈The PN filter mask `FrNmPnFilterMaskByte` shall be a byte array with the length of `FrNmPnInfoLength`.⌋( )

[SWS_FrNm_00420] ⌈If at least one bit within the PN request information Range of the received NM-PDU (defined by the parameters `FrNmPnInfoOffset` and `FrNmPnInfoLength)` matches with the corresponding bit in the PN filter mask, the PN request information is relevant for the ECU.⌋( )

7.8.11.1 RX-Path

[SWS_FrNm_00406]⌈ If FrNmPnEnabled is FALSE, FrNm shall perform the standard FrNm RxIndication.⌋ ()

[SWS_FrNm_00407] ⌈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.⌋( )

[SWS_FrNm_00408] ⌈If `FrNmPnEnabled` is `TRUE` and the PNI bit in the NM PDU provided by `FrNm_RxIndication()` has the value `1`, FrNm module shall process the Partial Networking Information of the NM-PDU as described in chapter 7.8.11.3 Aggregation of internal and external requested PNs and 7.8.11.4 Aggregation of external requested PNs⌋( )

7.8.11.2 TX-Path

[SWS_FrNm_00409] ⌈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] ⌈If `FrNmPnEnabled` is `FALSE`, FrNm shall set the value of the PNI bit to `0` in transmitted NM PDUs. ⌋ ( )

7.8.11.3 Aggregation of internal and external requested PNs

This feature is used by every ECU that has to switch on/off Pdu-Groups because of the activity of partial networks. (e.g. to prevent false timeouts). The Pdu-Groups shall be switched on if the corresponding PN is requested internally or externally. The Pdu-Groups shall be switched off not until all internal and external requests for the corresponding PN are released. The FrNm only provides the information if the PN is internally/externally requested or not. The logic for switching the Pdu-Groups shall be implemented by an upper layer.

The aggregated state of the internal/external requested PNs is called External Internal Requests Aggregated (EIRA).

[SWS_FrNm_00421] ⌈If `FrNmPnEiraCalcEnabled` is `TRUE` FrNm shall provide the possibility to store external and internal requested PNs combined over all relevant channels (all FrNm channels where `FrNmPnEnabled` is `TRUE`). At initialization the values of all PNs shall be set to 0 (not requested)⌋ ( )

[SWS_FrNm_00423] ⌈If
* `FrNmPnEiraCalcEnabled` is `TRUE`
* a NM-PDU is received
* PNs are requested within this message (bits set to 1)
* and the requested PNs are set to 1 within the [configured PN filter mask]

then FrNm shall store the request information (value 1) for these PNs. ⌋ ( )

[SWS_FrNm_00424] ⌈If :
* `FrNmPnEiraCalcEnabled` is `TRUE`
* NM-PDU is being requested to send by FrNm
* PNs are requested within this message (bits set to 1)
* and the requested PNs are set to 1 within the [configured PN filter mask]

then FrNm shall store the request information (value 1) for these PNs. ⌋ ( )

[SWS_FrNm_00425] ⌈If `FrNmPnEiraCalcEnabled` is `TRUE` FrNm shall provide a possibility to monitor for each PN if this PN is still externally or internally requested on at least one of the relevant channels. ⌋ ( )

Note: This means, only one timer is required to handle one PN on multiple connected physical channels. For example: only 8 EIRA reset timers are required to handle the requests of a Gateway with 6 physical channels and 8 partial networks.

This is possible because the external request is mirrored back to the requesting bus and provided to all other (required) physical channels. Thus it is not required to detect the physical channel that is the source of the request bit.

[SWS_FrNm_00426] ⌈If `FrNmPnEiraCalcEnabled` is `TRUE` and a PN is requested by message reception or sending (see SWS_FrNm_00423 and SWS_FrNm_00424) the monitoring for this PN shall be restarted with respect to the `FrNmPnResetTime`.⌋( )

Note: `FrNmPnResetTime` should be configured to a value greater than FlexRay NM cycle time.
Rational: If `FrNmPnResetTime` is configured to a value smaller than FlexRay NM cycle time and only one ECU requests the PN, the request state toggles in the EIRA because request state is rested before the requesting ECU is able to send the next NM message.

[SWS_FrNm_00428] ⌈If `FrNmPnEiraCalcEnabled` is `TRUE` and a PN is not requested again within `FrNmPnResetTime` the corresponding stored value for this PN shall be set to not requested (value 0).⌋( )

[SWS_FrNm_00429] ⌈If `FrNmPnEiraCalcEnabled` is `TRUE` and the stored value for a PN is set to requested or back to not requested (see SWS_FrNm_00423, SWS_FrNm_00424 and SWS_FrNm_00428) FrNm shall inform the upper layers by calling `PduR_FrNmRxIndication()` for the configured EIRA PDU (i.e. changed EIRA information shall be passed to COM).. ⌋( )

7.8.11.4 Aggregation of external requested PNs

Note: This feature is used by the Gateways to collect **only the external** PN requests. The external PN Requests are mirrored back to the requesting bus and provided to other (required) physical channels in Case of Central-Gateway. In Case of a Sub-Gateway the requests bit mustn't be mirrored back to the requesting physical channel to avoid static waking between Central-Gateway and Sub-Gateways. This logic shall be implemented by an upper layer (e.g.:ComM). The FrNm only provides the information if the PN is externally requested or not. The COM is used for data transmission to the upper layer.

The aggregated state of the external requested PNs is called "External Requests Aggregated" (ERA).

[SWS_FrNm_00432] ⌈If `FrNmPnEraCalcEnabled` is `TRUE` FrNm shall provide the possibility to store external requested PNs on each relevant channel. At initialization the values of all PNs shall be set to 0 (not requested). ⌋( )

[SWS_FrNm_00434] ⌈If:
* `FrNmPnEraCalcEnabled` is `TRUE`
* a NM-PDU is received

* PNs are requested within this message (bits set to 1)
* and the requested PNs are set to 1 within the [configured PN filter mask]

then FrNm shall store the request information (value 1) for these PNs. ⌋ ( )

[SWS_FrNm_00435] ⌈If `FrNmPnEraCalcEnabled` is `TRUE` FrNm shall provide a possibility to monitor on each relevant channel and for each PN if this PN is still externally requested. ⌋ ( )

Note: This means, a separate timer is required to handle one PN on multiple connected physical channels. For example: 48 ERA reset timers are required to handle the requests of a Gateway with 6 physical channels and 8 partial networks. It is not possible to combine the reset timer like EIRA timers, because the external request mustn't be mirrored back to the requesting bus by a Sub-Gateway. Thus it is required to detect the physical channel that is the source of the request bit.

[SWS_FrNm_00436] ⌈If `FrNmPnEraCalcEnabled` is `TRUE` and a PN is requested by message reception (see SWS_FrNm_00434) the monitoring for this PN shall be restarted with respect to the `FrNmPnResetTime`. ⌋ ( )

Note: `FrNmPnResetTime` should be configured to a value greater than FlexRay NM cycle time.
Rational: If `FrNmPnResetTime` is configurd to a value smaller than FlexRay NM cycle time and only one ECU requests the PN, the request state toggles in the ERA because request state is rested before the requesting ECU is able to send the next NM message.

[SWS_FrNm_00437] ⌈If `FrNmPnEraCalcEnabled` is `TRUE` and a PN is not requested again within `FrNmPnResetTime` the corresponding stored value for this PN shall be set to not requested (value 0).⌋ ( )

[SWS_FrNm_00438] ⌈If `FrNmPnEraCalcEnabled` is `TRUE` and the stored value for a PN changes to requested or back to not requested (see SWS_FrNm_00434 and SWS_FrNm_00437) FrNm shall inform the upper layers by calling PduR_FrNmRxIndication() for the configured ERA PDU (i.e. changed ERA information shall be passed to COM). ⌋ ( )

[SWS_FrNm_00461]⌈ If `FrNmPnEiraCalcEnabled` is `TRUE` and `FrNmPnEraCalcEnabled` is `TRUE` , the PN status information has to be stored separately for both, the EIRA and ERA information (compare SWS_FrNm_00421 and SWS_FrNm_00432).⌋()

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 7-5 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 (option 7 in Table 7-5), 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 10-3 (Seite 119).

[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 10-4 (Seite 120).

[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 10-3 (Seite 119).

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 Debugging concept

[SWS_FrNm_00349] {OBSOLETE} ⌈The FrNm states BusSleep, Synchronize, Repeat Message, Normal Operation, and Ready Sleep shall be available for debugging.⌋ ( )

7.11 Transmission Error Handling

[SWS_FrNm_00035] ⌈When periodic NM-message transmissions are expected but those NM-messages are not transmitted within a time interval given by **`FrNmMsgTimeoutTime`** (configuration parameter) and FrNmMsgTimeout is greater 0, the FlexRay NM shall notify the Generic NM Interface module that a transmission failure has occurred; this notification occurs by calling **`Nm_TxTimeoutException`**.⌋ (SRS_Nm_00053, SRS_Nm_00137)

Note: NM-message transmissions are expected to be sent whenever the API FrIf_Transmit() 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”.

7.12 Error classification

7.12.1 Development Errors

[SWS_FrNm_00021] ⌈The following errors shall be detectable by the FlexRay NM depending on its build version (development/production mode).

Type or error	*Relevance*	*Related error code*	*Value*
API service used without module	Development	FRNM_E_UINIT	01h

initialization			
API service called with invalid channel handle	Development	FRNM_E_INVALID_CHANNEL	02h
API service called with Invalid pointer	Development	FRNM_E_PARAM_POINTER	03h
API service called with invalid PDU ID as input parameter	Development	FRNM_E_PDU_ID_INVALID	04h
FrNm initialization has failed, e.g. selected configuration set doesn't exist	Development	FRNM_E_INIT_FAILED	05h

⌋(SRS_BSW_00337, SRS_BSW_00331)

7.12.2 Runtime Errors

< There are no runtime errors.>

7.12.3 Transient Faults

< There are no transient faults.>

7.12.4 Production Errors

< There are no production errors.>

7.12.5 Extended Production Errors

< There are no extended production errors.>

7.13 Error detection

[SWS_FrNm_00056] ⌈Default errors shall not be returned by API functions. In case of a default error the corresponding API function shall return `E_NOT_OK`, if applicable. ⌋(SRS_BSW_00369)

[SWS_FrNm_00057] ⌈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. ⌋(SRS_BSW_00369)

[SWS_FrNm_00050] ⌈If DET is enabled (ECUC_FrNm_0036) and APIs other than FrNm_Init or FrNm_GetVersionInfo or FrNm_MainFunction (see SWS_BSW_00037) are called when FrNm has not been initialized, the called function shall return immediately with E_NOT_OK(if the function has standard return type). Additionally, the FrNm module shall report the error code FRNM_E_UNINIT to the Default Error Tracer. ⌋ (SRS_BSW_00407, SRS_BSW_00416, SRS_BSW_00406)

[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. ⌋ ( )

7.14 Error notification

For details refer to the chapter 7.4 “Error notification” in *SWS_BSWGeneral.*

Note: The NM-cluster handle is invalid if it is different from the allowed configured values.

Note: Currently no production errors are defined.

7.15 Version check

For details refer to the chapter 5.1.8 “Version Check” in *SWS_BSWGeneral.*

7.16 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.⌋ ()

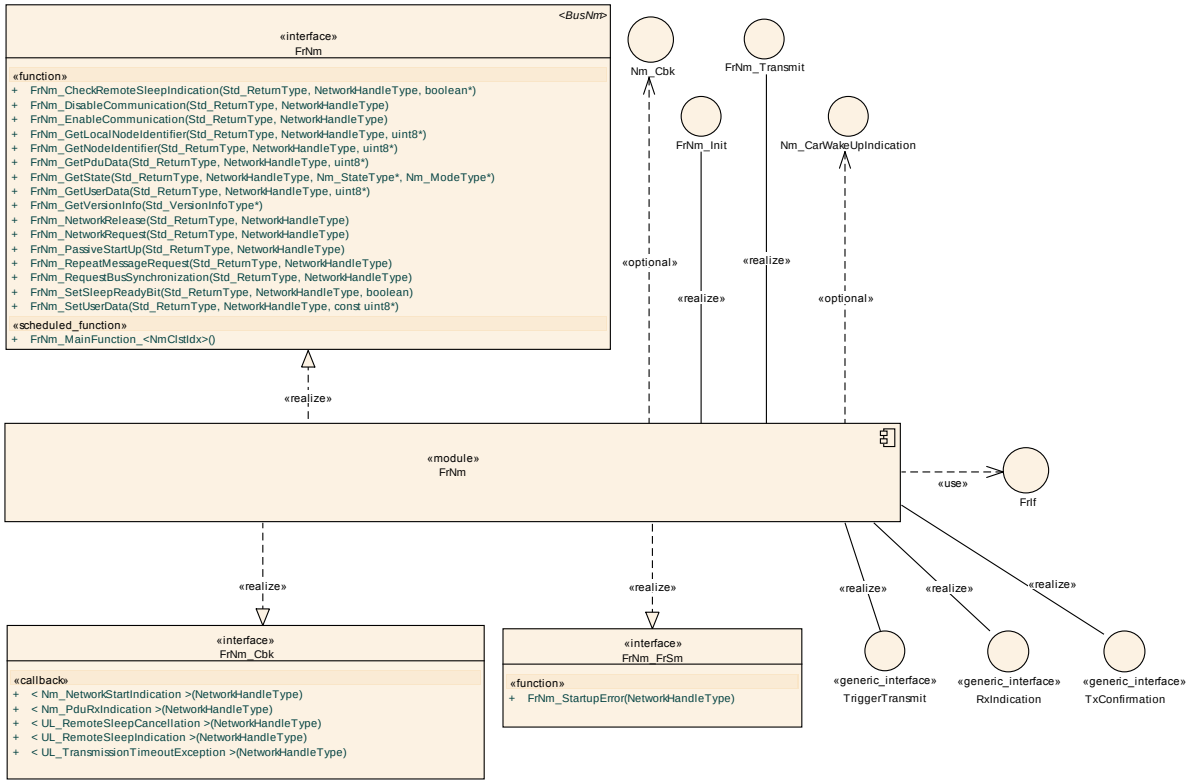
8 API specification

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

[SWS_FrNm_00034] ⌈Each service other than **`FrNm_Init`** refers to one NM cluster only.⌋ (SRS_Nm_00045, SRS_Nm_00154)

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”.

The following figure gives an overview of the available API services and interfaces:



API Specification (Overview)

8.1 Imported types

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

[SWS_FrNm_00235] ⌈

Module	*Imported Type*
ComStack_Types	NetworkHandleType

	PduIdType
	PduInfoType
Nm	Nm_ModeType
	Nm_StateType
Std_Types	Std_ReturnType
	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 ([8]) in chapter 8.2 Type definitions.

8.2.2 FlexRay NM specific Type Definitions

8.2.2.1 FrNm_ConfigType

[SWS_FrNm_00454]⌈

Name:	`FrNm_ConfigType`
Type:	`Structure`
Range:	Implementation specific.
Description:	Contains configuration parameters.

For **`FrNm_ConfigType`** see chapter 10.5 Seite 102.⌋ ()

8.3 Function definitions: FrNm Services provided to Generic NM Interface Module

8.3.1 FrNm_Init

[SWS_FrNm_00236] ⌈

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.

⌋ ( )

[SWS_FrNm_00028] ⌈The function **`FrNm_Init`** shall initialize the FrNm module. ⌋ (SRS_BSW_00101)

[SWS_FrNm_00073] ⌈After a successful initialization of the FrNm module, the function **`FrNm_Init`** shall set the initialization status to **`NM_INIT`**, otherwise to **`NM_UNINIT`**. ⌋ (SRS_BSW_00406)

[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] ⌈

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.	

⌋ ( )

[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] ⌈

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'.	

⌋ ( )

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

8.3.4 FrNm_NetworkRelease

[SWS_FrNm_00239] ⌈

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'.	

⌋( )

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

8.3.5 FrNm_SetUserData

[SWS_FrNm_00240] ⌈

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.	

⌋( )

[SWS_FrNm_00043] ⌈If user data handling is enabled for the FrNm module, then the function **`FrNm_SetUserData`** shall set the user data. ⌋(SRS_Nm_02503)

[SWS_FrNm_00319] ⌈If FrNm_SetUserData is called and the FrNm module is in UnInit state, then E_NOT_OK shall be returned. ⌋( )

[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] ⌈

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.	

⌋( )

[SWS_FrNm_00044] ⌈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. ⌋(SRS_Nm_02504)

[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] ⌈

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
Description:	Gets PDU data.	

⌋( )

[SWS_FrNm_00265] ⌈The function **`FrNm_GetPduData`** shall get the whole NM PDU data out of the most recently received NM message. ⌋( )

[SWS_FrNm_00266] ⌈The function **FrNm_GetPduData** shall only be available if the configuration parameter **FrNmControlBitVectorEnabled** or **FrNmSourceNodeIdentifierEnabled** or **FrNmUserDataEnabled** is set to ON. ⌋ ( )

8.3.8 FrNm_RepeatMessageRequest

[SWS_FrNm_00243] ⌈

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: Repeat Message Request has failed
Description:	This function causes a Repeat Message Request to be transmitted next on the bus.	

⌋ ( )

[SWS_FrNm_00172] ⌈If the node detection feature is enabled, then the function **FrNm_RepeatMessageRequest** shall request node detection on the FlexRay Bus NM nodes. ⌋ ( )

[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. ⌋ ( )

[SWS_FrNm_00228] ⌈The function FrNm_RepeatMessageRequest shall be only available if the configuration parameter FrNmNodeDetectionEnabled is set to ON. (see SWS_FrNm_00170) ⌋ ( )

8.3.9 FrNm_GetNodeIdentifier

[SWS_FrNm_00244] ⌈

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
Description:	This function gets the node identifier from the last successfully received NM-message.	

⌋( )

[SWS_FrNm_00047] ⌈If the node detection feature is enabled, then the function **`FrNm_GetNodeIdentifier`** shall provide the node identifier from the most recently received NM-message.⌋(SRS_Nm_02506)

[SWS_FrNm_00267] ⌈The function FrNm_GetNodeIdentifier shall be only available if the configuration parameter FrNmSourceNodeIdentifierEnabled is set as ON (see SWS_FrNm_00170).⌋( )

8.3.10 FrNm_GetLocalNodeIdentifier

[SWS_FrNm_00245] ⌈

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
Description:	This function gets the node identifier configured for the local node.	

⌋( )

[SWS_FrNm_00046] ⌈If node detection is enabled, then the function **`FrNm_GetLocalNodeIdentifier`** shall provide the node identifier configured for the local host node (**`FrNmNodeId`**).⌋( )

[SWS_FrNm_00268] ⌈The function FrNm_GetLocalNodeIdentifier shall be only available if the configuration parameter FrNmSourceNodeIdentifierEnabled is set as ON (refer to SWS_FrNm_00170).⌋( )

8.3.11 FrNm_RequestBusSynchronization

[SWS_FrNm_00246] ⌈

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.	

⌋ ( )

[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] ⌈

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.	

⌋ ( )

[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] ⌈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). ⌋ (SRS_Nm_02509)

[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] ⌈

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.	

⌋ ( )

[SWS_FrNm_00104] ⌈The function **`FrNm_GetState`** shall provide consistent information about the current state and the current mode of the NM state machine. ⌋ (SRS_Nm_00050)

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] ⌈

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.	

⌋ ( )

8.3.15 FrNm_StartupError

[SWS_FrNm_00393] ⌈

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.	

⌋ ( )

8.3.16 FrNm_Transmit

[SWS_FrNm_00359] ⌈

Service name:	FrNm_Transmit

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

Service ID[hex]:	0x11

Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	FrNmTxPduId	L-PDU handle of FlexRay L-PDU to be transmitted. This handle specifies the corresponding FlexRay L-PDU ID and implicitly the FlexRay Driver instance as well as the corresponding FlexRay controller device.
	PduInfoPtr	Pointer to a structure with FlexRay L-PDU related data: DLC and pointer to FlexRay L-SDU buffer.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: Always
Description:	This function is used by the PduR to trigger a spontaneous transmission of an NM message with the provided NM User Data. On FlexRay it is not possible to send spontaneous NM messages. So this function is only a dummy to avoid linker errors of the PduR and standardize the NM handling between FrNm and CanNm.	

⌋(SRS_Nm_02524)

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

8.3.17 FrNm_EnableCommunication

[SWS_FrNm_00387] ⌈

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	

⌋( )

[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 (FRNM_PASSIVE_MODE_ENABLED), then the service

FrNm_EnableCommunication shall have no effects and directly return E_NOT_OK.」
( )

8.3.18 FrNm_DisableCommunication

[SWS_FrNm_00390] ⌈

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	

」( )

[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 (FRNM_PASSIVE_MODE_ENABLED), then the service FrNm_DisableCommunication shall have no effect and directly return E_NOT_OK.」( )

8.3.19 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	

⌋ ( )

8.4 Call-back notifications: NM callbacks provided to lower layers

8.4.1 FrNm_RxIndication

[SWS_FrNm_00251] ⌈

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 I-PDU.
	PduInfoPtr	Contains the length (SduLength) of the received I-PDU and a pointer to a buffer (SduDataPtr) containing the I-PDU.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	Indication of a received I-PDU from a lower layer communication interface module.	

⌋ ( )

[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 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] ⌈

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.	

⌋( )

[SWS_FrNm_00280] ⌈The function FrNm_TriggerTransmit shall copy the triggered FlexRay NM PDU with respect to the triggered FlexRay NM PDU ID. ⌋( )

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

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

8.4.3 FrNm_TxConfirmation

[SWS_FrNm_00460]⌈

Service name:	FrNm_TxConfirmation	

Syntax:	`void FrNm_TxConfirmation(`
	`  PduIdType TxPduId`
	`)`	

Service ID[hex]:	0x40	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant for different PduIds. Non reentrant for the same PduId.	
Parameters (in):	TxPduId	ID of the I-PDU that has been transmitted.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	The lower layer communication interface module confirms the transmission of an I-PDU.	

⌋ ()

8.5 Scheduled functions: FrNm Services provided to BSW Scheduler

8.5.1 FrNm_MainFunction_< FrNmChannelIdRef >

[SWS_FrNm_00255] ⌈

Service name:	FrNm_MainFunction_<NmClstIdx>

Syntax:	`void FrNm_MainFunction_<NmClstIdx>(`
	`  void`
	`)`

Service ID[hex]:	0xf0
Description:	Main function of FlexRay NM.

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.

⌋ (SRS_BSW_00432)

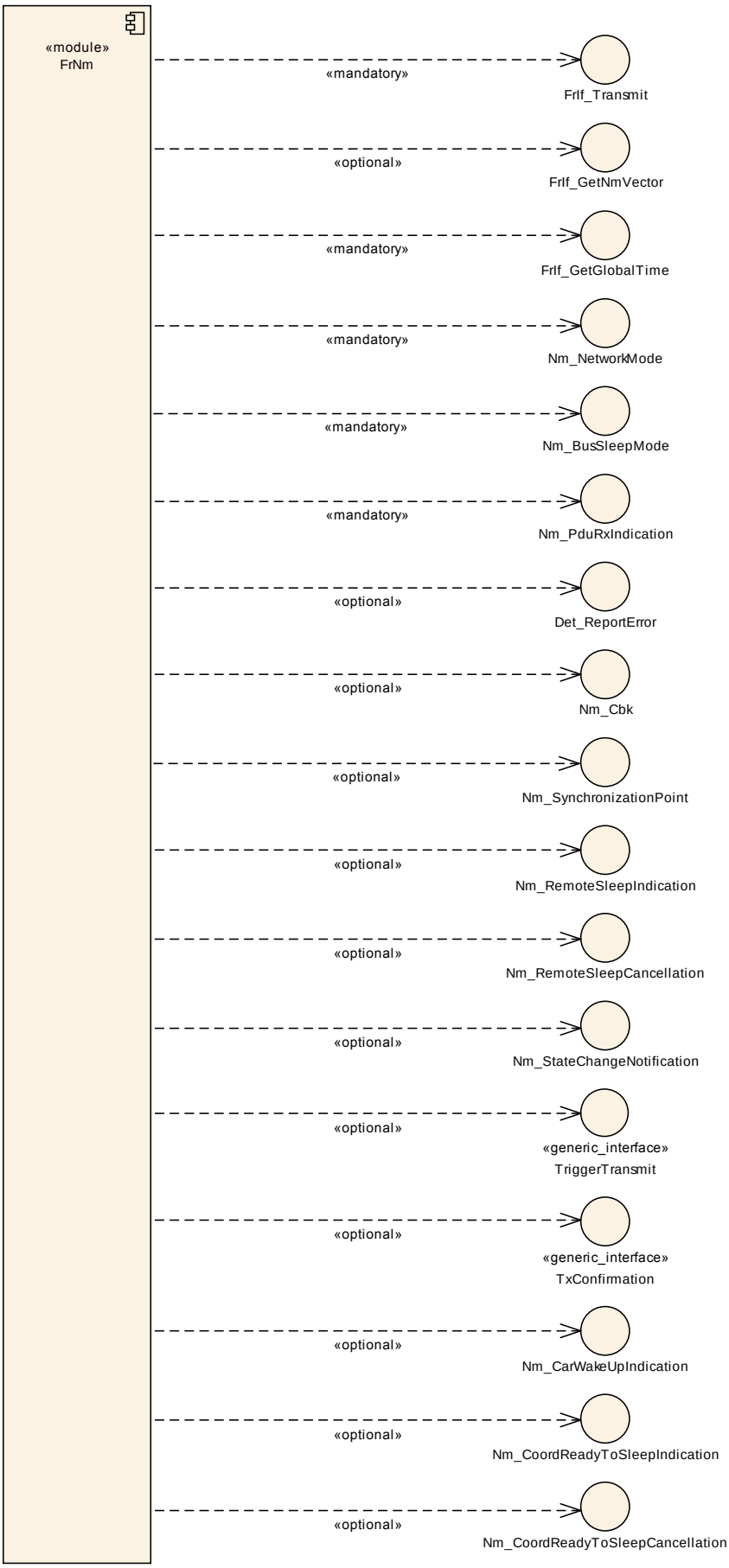
[SWS_FrNm_00283] ⌈There shall be one dedicated FlexRay NM Main Function for each NM cluster. The API names are therefore:
FrNm_MainFunction_0() for FlexRay NM cluster associated with FlexRay NM cluster 0, FrNm_MainFunction_1() for FlexRay NM cluster associated with FlexRay NM cluster 1, etc. ⌋ ( )

[SWS_FrNm_00449] ⌈ It shall be possible to perform an independent processing of transmission- and reception-handling inside the main function according to SRS_BSW_00432 [4]⌋ (SRS_BSW_00432).

[SWS_FrNm_00344] ⌈The Service ID of **FrNm_MainFunction_<NmClstIdx>** shall be 0xf0 + NmClstIdx. ⌋ ( )

8.6 Expected interfaces

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


«module»
FrNm
«mandatory»
FrIf_Transmit
«optional»
FrIf_GetNmVector
«mandatory»
FrIf_GetGlobalTime
«mandatory»
Nm_NetworkMode
«mandatory»
Nm_BusSleepMode
«mandatory»
Nm_PduRxIndication
«optional»
Det_ReportError
«optional»
Nm_Cbk
«optional»
Nm_SynchronizationPoint
«optional»
Nm_RemoteSleepIndication
«optional»
Nm_RemoteSleepCancellation
«optional»
Nm_StateChangeNotification
«optional»
«generic_interface»
TriggerTransmit
«optional»
«generic_interface»
TxConfirmation
«optional»
Nm_CarWakeUpIndication
«optional»
Nm_CoordReadyToSleepIndication
«optional»
Nm_CoordReadyToSleepCancellation

8.6.1 Mandatory Interfaces

This chapter defines all interfaces which are required to fulfill the core functionality of the module.

[SWS_FrNm_00256] ⌈

API function	*Description*
FrIf_GetGlobalTime	Wraps the FlexRay Driver API function Fr_GetGlobalTime().
FrIf_Transmit	Requests the sending of a PDU.
Nm_BusSleepMode	Notification that the network management has entered Bus-Sleep Mode.
Nm_NetworkMode	Notification that the network management has entered Network Mode.
Nm_PduRxIndication	Notification that a NM message has been received.

⌋ ( )

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 [8].

8.6.2 Optional Interfaces

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

[SWS_FrNm_00257] ⌈

API function	*Description*
Det_ReportError	Service to report development errors.
FrIf_GetNmVector	Derives the FlexRay NM Vector.
Nm_CarWakeUpIndication	This function is called by a <Bus>Nm to indicate reception of a CWU request.
Nm_CoordReadyToSleepCancellation	Cancels an indication, when the NM Coordinator Sleep Ready bit in the Control Bit Vector is set back to 0.
Nm_CoordReadyToSleepIndication	Sets an indication, when the NM Coordinator Sleep Ready bit in the Control Bit Vector is set
Nm_NetworkStartIndication	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_RemoteSleepCancellation	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	Notification that the network management has detected that all other nodes on the network are ready to enter Bus-Sleep Mode.
Nm_StateChangeNotification	Notification that the state of the lower layer <BusNm> has changed.
Nm_SynchronizationPoint	Notification to the NM Coordinator functionality that this is a suitable point in time to initiate the coordinated shutdown on.
PduR_FrNmTriggerTransmit	Within this API, the upper layer module (called module) shall check whether the available data fits into the buffer size reported by PduInfoPtr->SduLength. If it fits, it shall copy its data into the buffer provided by PduInfoPtr->SduDataPtr and update the length of the actual copied data in PduInfoPtr->SduLength. If not, it returns E_NOT_OK without changing PduInfoPtr.
PduR_FrNmTxConfirmation	The lower layer communication interface module confirms the

	transmission of an I-PDU.

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

`FrIf_GetNmVector` Configured by configuration parameter FrIfGetNmVectorSupport (see FrIf06100_Conf)

`Nm_NetworkStartIndication` Configured if FrNmPduRxIndicationEnabled is configured (see ECUC_FrNm_00046)

`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)

`Nm_TxTimeoutException` Configured with the parameter FrNmMsgTimeoutTime (see ECUC_FrNm_00028). Nm_TxTimeoutException is available, if the value of FrNmMsgTimeoutTime is configured as greater than '0'(see ECUC_FrNm_00018)

⌋ ( )

8.6.2.1 NetworkStartIndication

[SWS_FrNm_00470]⌈

Service name:	< Nm_NetworkStartIndication >	
Syntax:	`void < Nm_NetworkStartIndication >( NetworkHandleType nmNetworkHandle )`	
Service ID[hex]:	0x00	
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.	

⌋ ()

8.6.2.2 PduRxIndication

[SWS_FrNm_00471]⌈

Service name:	< Nm_PduRxIndication >

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

Service ID[hex]:	0x00
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

] ()

8.6.2.3 RemoteSleepCancellation

[SWS_FrNm_00472][

Service name:	< UL_RemoteSleepCancellation >

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

Service ID[hex]:	0x00
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 .

] ()

8.6.2.4 RemoteSleepIndication

[SWS_FrNm_00473][

Service name:	< UL_RemoteSleepIndication >

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

Service ID[hex]:	0x00
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).	

] ()

8.6.2.5 TransmissionTimeoutException

[SWS_FrNm_00474][

Service name:	< UL_TransmissionTimeoutException >	
Syntax:	`void < UL_TransmissionTimeoutException >( NetworkHandleType nmNetworkHandle )`	
Service ID[hex]:	0x00	
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 to indicate to this upper layer BSW module that no NM-message could successfully be transmitted within the time interval of FRNM_MSG_TIMEOUT_TIME, and the periodic NM-message transmission is running.	

] ()

8.6.3 Configurable interfaces

In this chapter all interfaces are listed where the target function could be configured. The target function is usually a call-back function. The names of these kind of interfaces is not fixed because they are configurable.

There are no configurable interfaces.

9 Sequence diagrams

9.1 Use Case 01 – Initialization

Sequence in

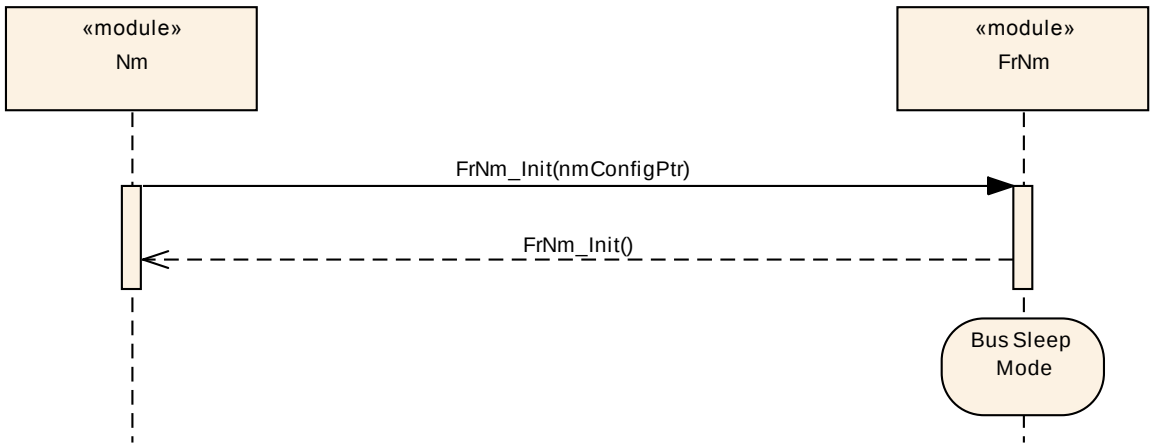


Figure 9-1 shows how to initialize the FlexRay Network Management.

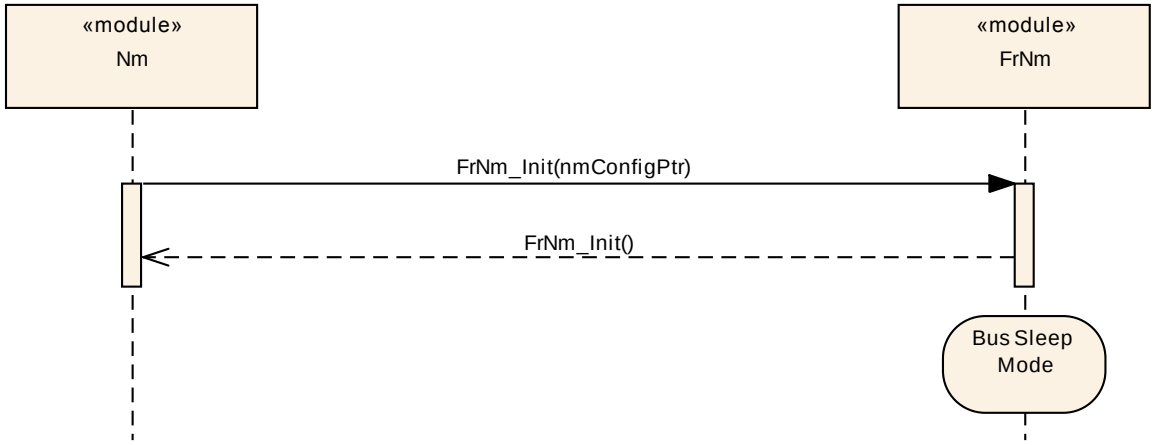


Figure 9-1 FrNm Init Sequence

9.2 Use Case 02 - Passive Startup

Sequence in

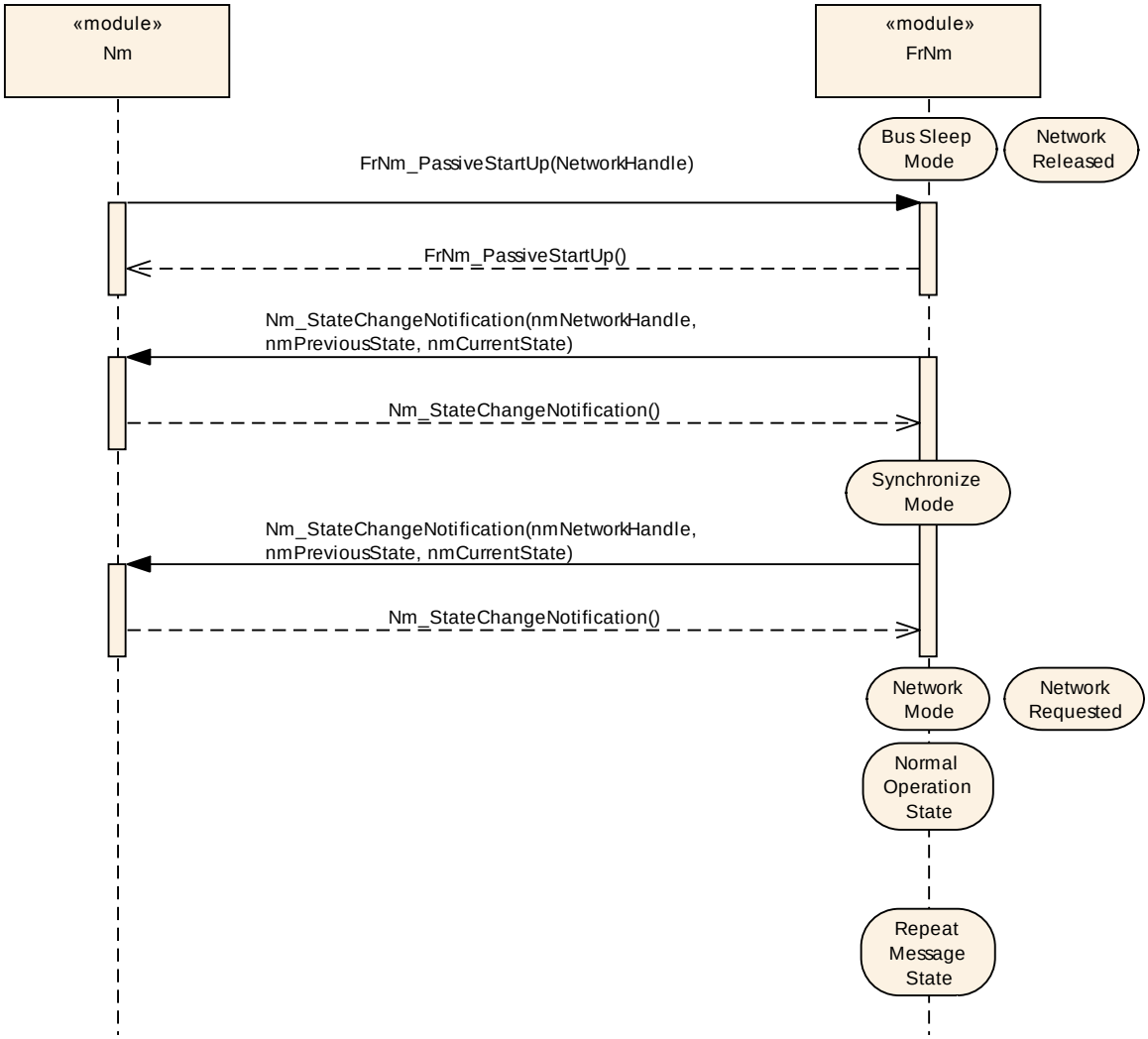


Figure 9-2 shows the normal passive startup.

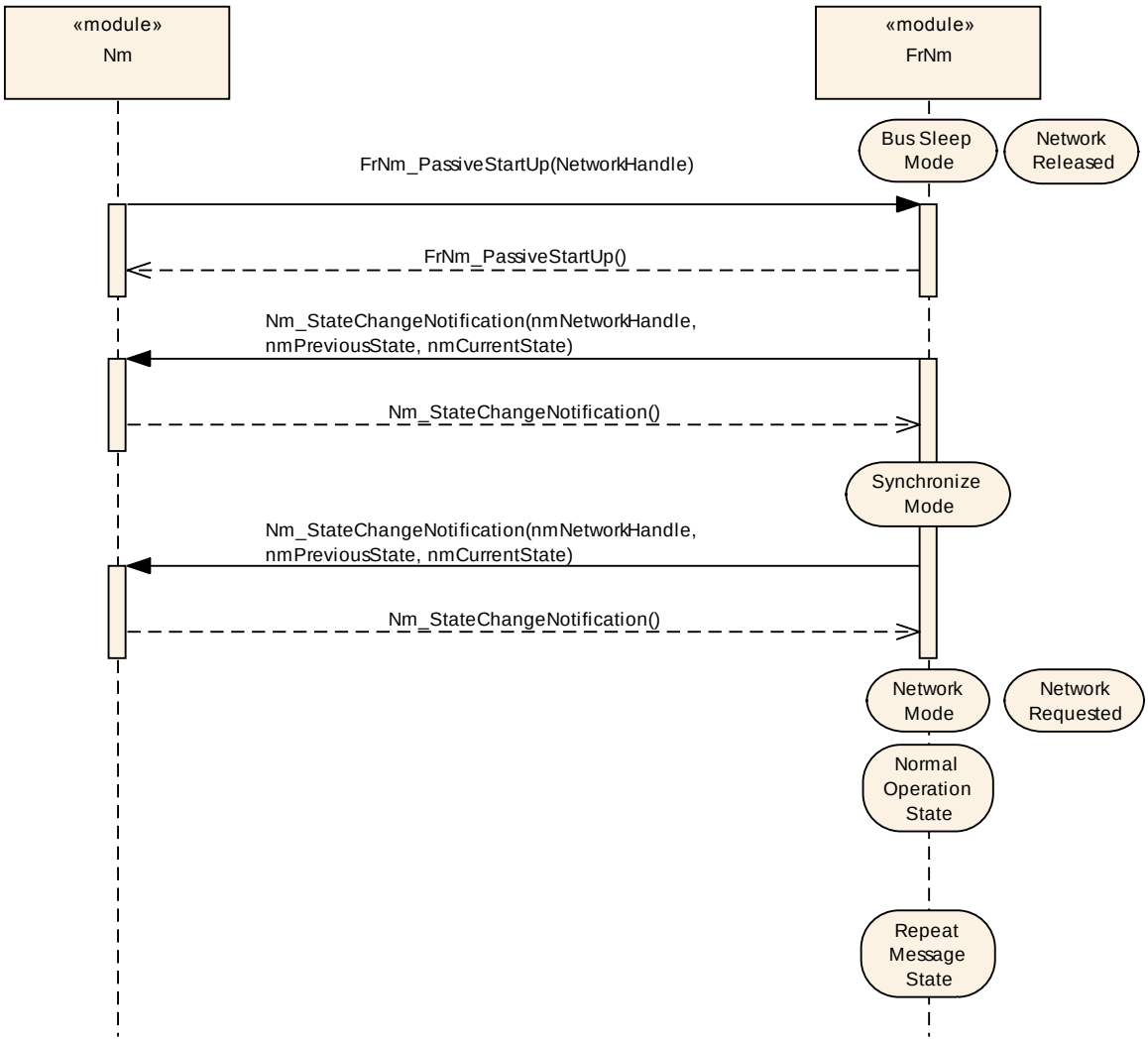


Figure 9-2 FrNm passive startup sequence

9.3 Use Case 03 – Passive Startup with a Network Request

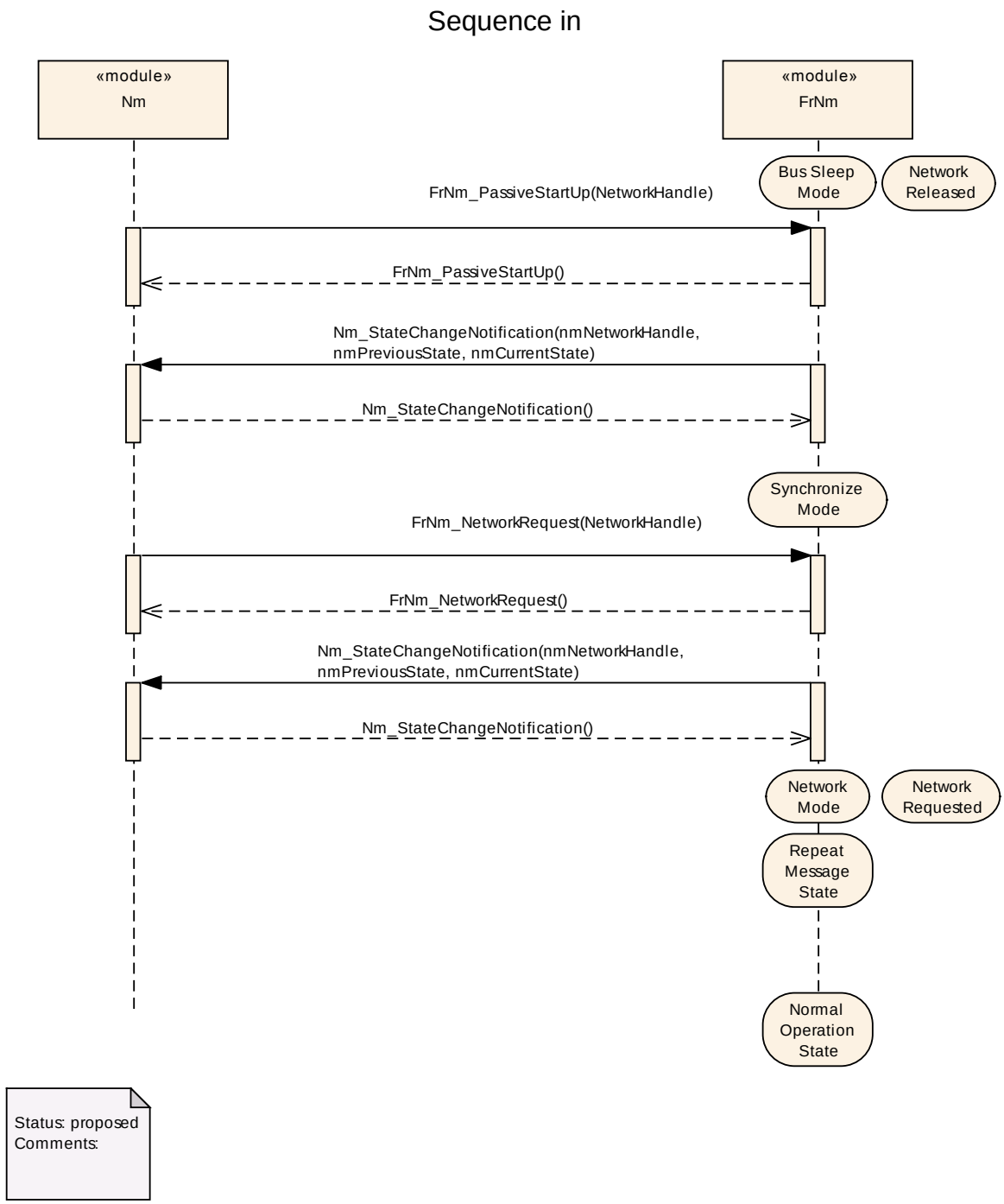


Figure 9-3 shows a passive startup where a network is requested before Network Mode has been reached.

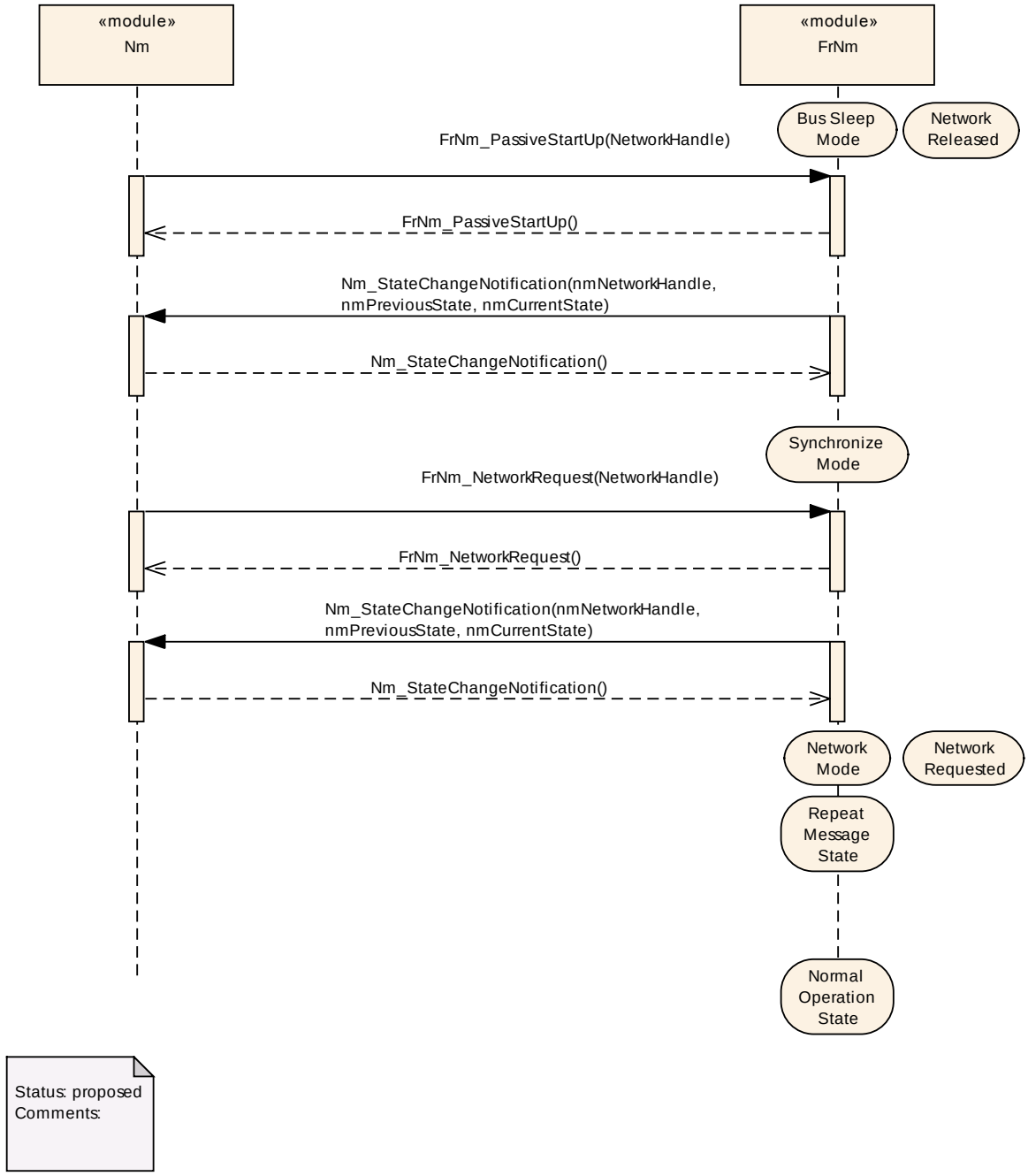


Figure 9-3 FrNm passive startup with a “active” network request sequence

9.4 Use Case 04 – Normal Operation

Sequence in

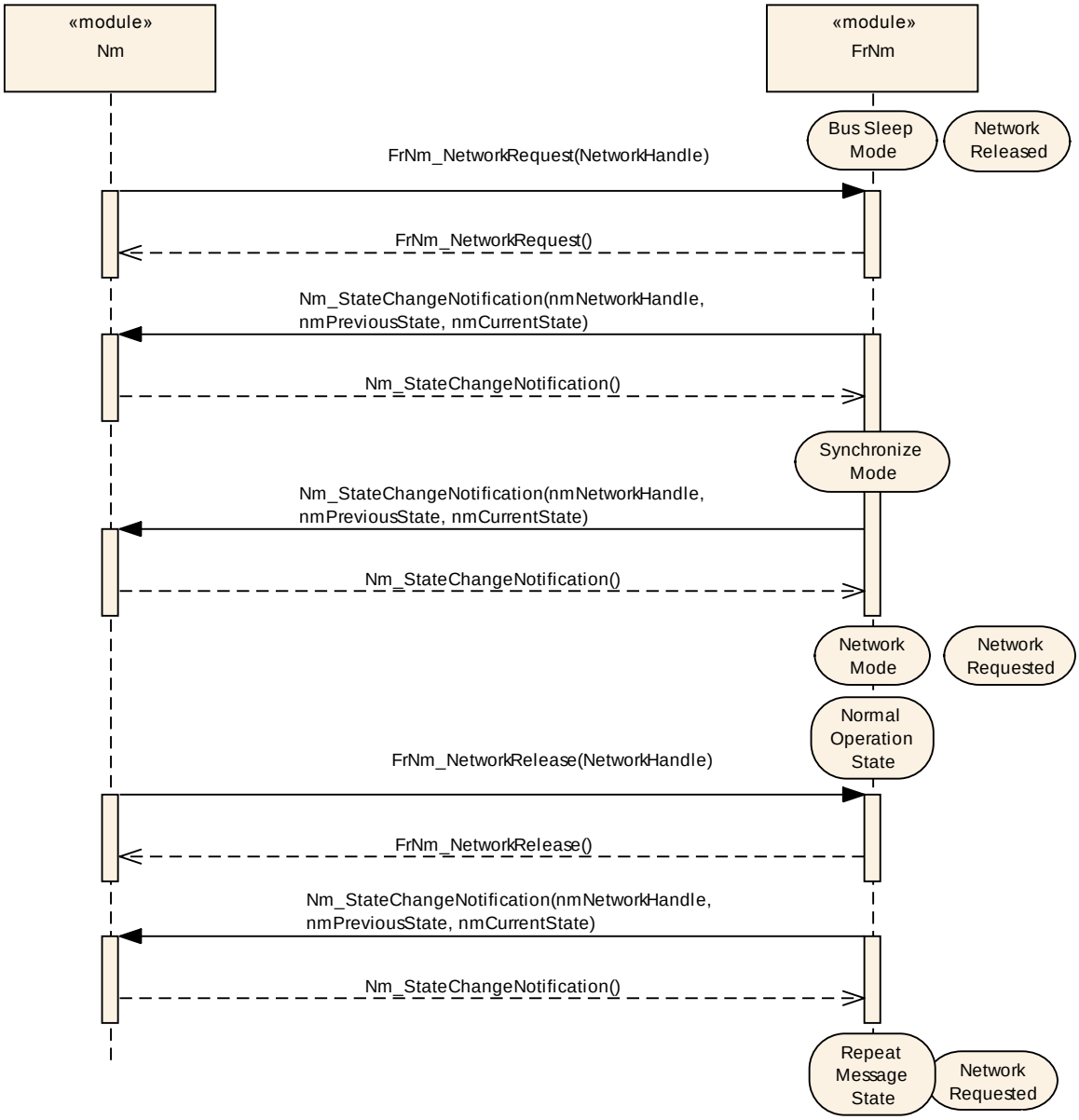


Figure 9-4 shows how to request and release the network

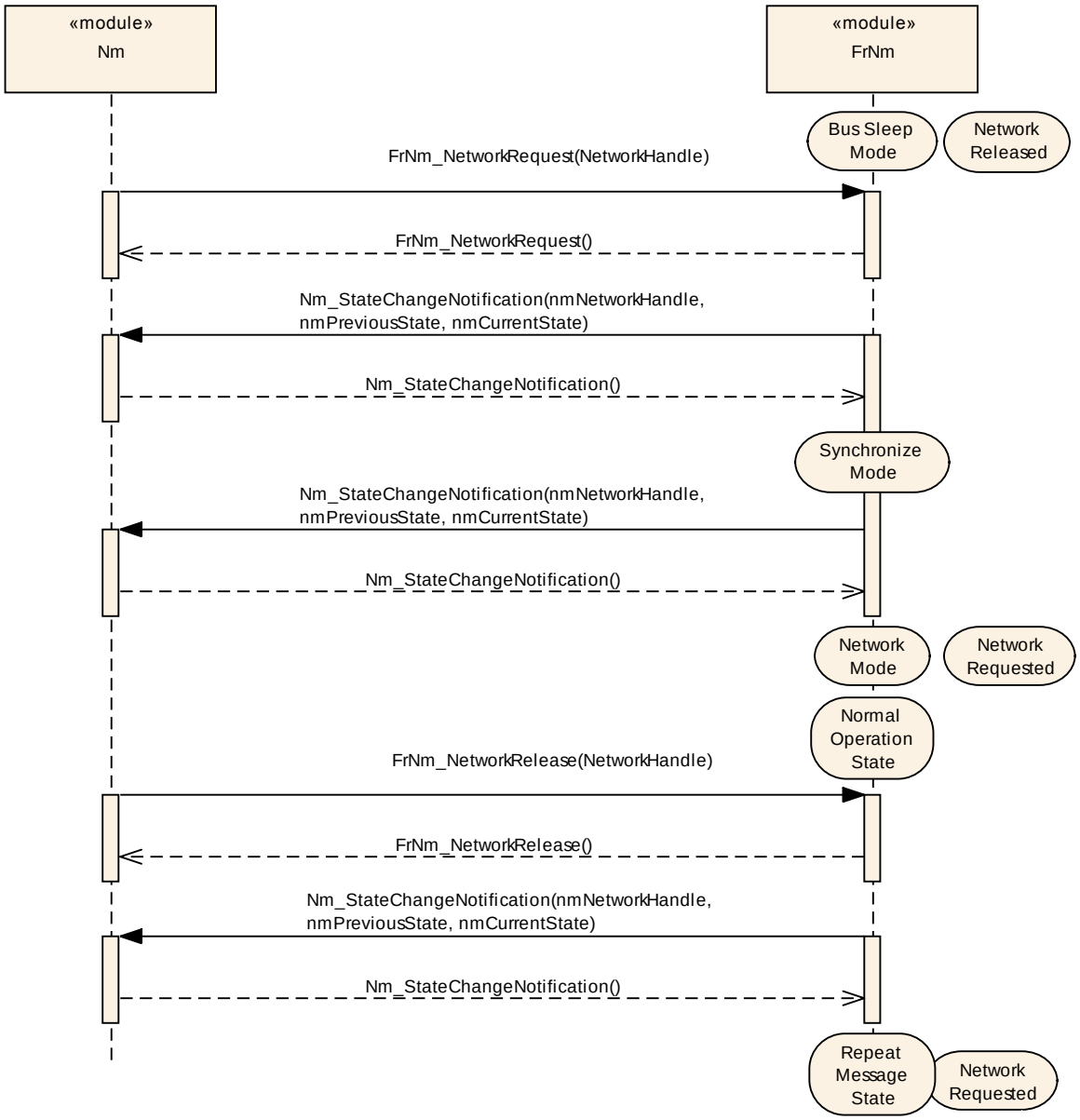


Figure 9-4 FrNm normal operation sequence

9.5 Use Case 05 – Shutdown

Sequence in Figure 9-5 shows a normal shutdown sequence.

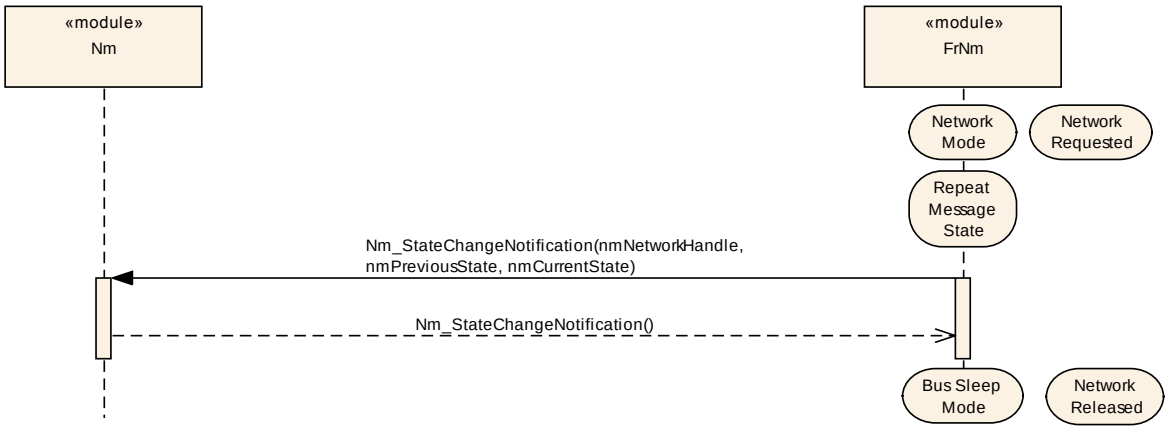


Figure 9-5 FrNm Shutdown

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] ⌈Both static and runtime configuration parameters shall be located outside the source code of the module.⌋(SRS_BSW_00159)

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) for the parameter specification. Chapter 10.2 specifies the structure (containers) and the parameters of the module FrNm. Chapter 10.3 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 Variants

10.2.1 Variant 1: VARIANT-PRE-COMPILE

[SWS_FrNm_00290] ⌈All configuration parameters are configured at pre-compile time.

Use case: Source code optimizations⌋( )

10.2.2 Variant 2: VARIANT-LINK-TIME

[SWS_FrNm_00291] ⌈All configuration parameters which support only the configuration class "Pre-Compile" shall be configured at pre-compile time. All other configuration parameters shall be configurable at link-time.

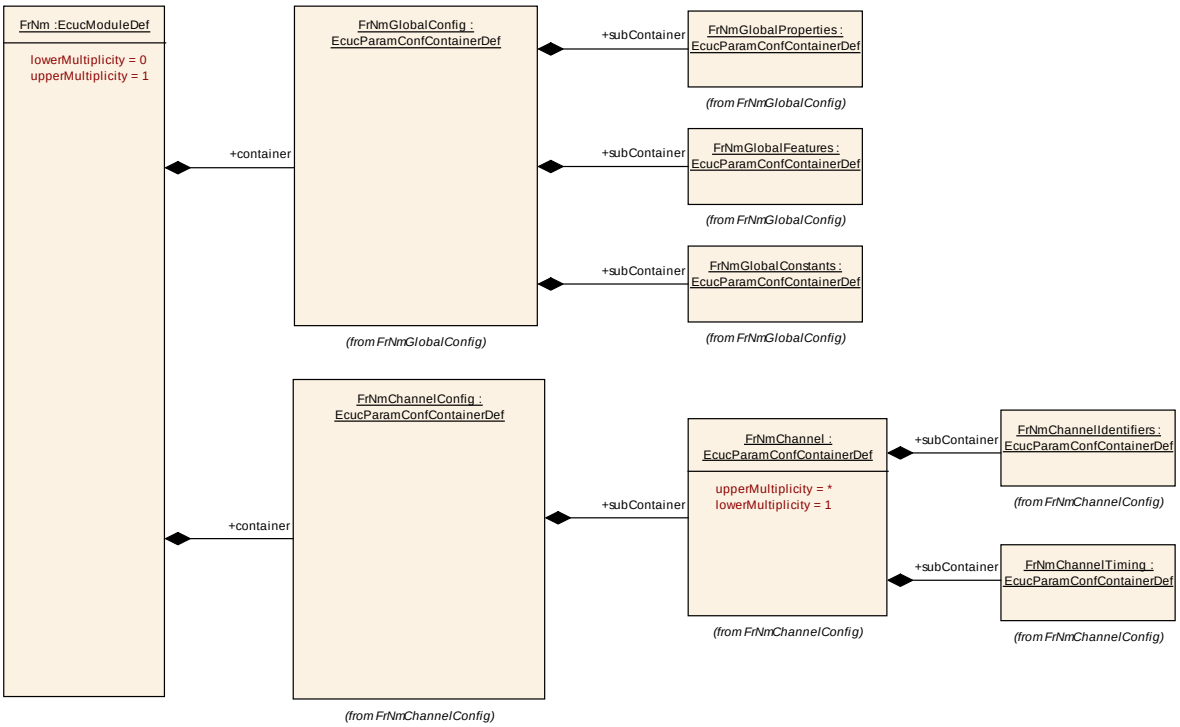
Use case: Object code libraries⌋( )

10.2.3 Variant 3: VARIANT-POST-BUILD

[SWS_FrNm_00292] ⌈ All configuration parameters that support configuration class “Post-build” shall be configurable at post-build time.

Use case: ECU configuration can be flashed (L) ⌋( )

10.3 Configurable parameters



10.3.1 FrNm

Module Name	*FrNm*
Module Description	The Flexray Nm module
Post-Build Variant Support	true

Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
FrNmChannelConfig	1	This container contains the configuration parameters for all FlexRay NM channels.
FrNmGlobalConfig	1	This container contains all global configuration parameters for the FrNm module.

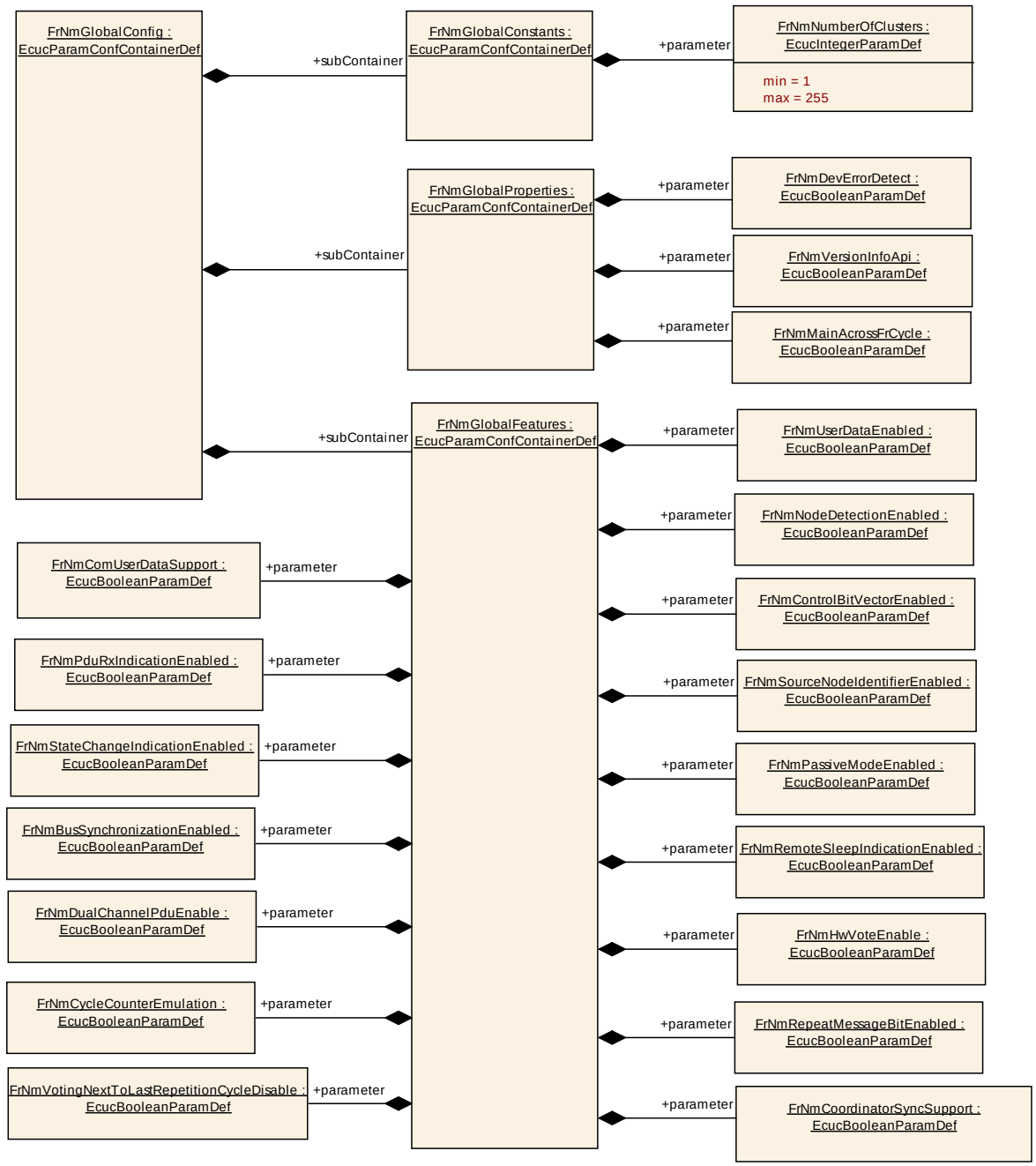
10.4 Global configurable parameters

10.4.1 FrNmGlobalConfig

SWS Item	**ECUC_FrNm_00001 :**

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

Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
FrNmGlobalConstants	1	This container contains module constants related to the FlexRay NM functionality.
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.



10.4.2 FrNmGlobalConstants

SWS Item	**ECUC_FrNm_00005 :**
Container Name	FrNmGlobalConstants
Description	This container contains module constants related to the FlexRay NM functionality.
Configuration Parameters	

SWS Item	**ECUC_FrNm_00034 : (Obsolete)**		
Name	FrNmNumberOfClusters		
Description	Number of AUTOSAR FR NM clusters allowed within one ECU. **Tags:** atp.Status=obsolete atp.StatusRevisionBegin=4.2.2		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
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		

No Included Containers

10.4.3 FrNmGlobalFeatures

SWS Item	**ECUC_FrNm_00004 :**
Container Name	FrNmGlobalFeatures
Description	This container contains module features related to the FlexRay NM functionality.
Configuration Parameters	

SWS Item	**ECUC_FrNm_00048 :**		
Name	FrNmBusSynchronizationEnabled		
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		

SWS Item	**ECUC_FrNm_00054 :**
Name	FrNmComUserDataSupport
Description	Enable/disable the 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		

SWS Item	**ECUC_FrNm_00041 :**		
Name	FrNmControlBitVectorEnabled		
Description	Pre-processor switch for enabling control bit vector support. calculationFormula = If (FrNmNodeDetectionEnabled == False) then Equal(False) else Equal(False or True)		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_FrNm_00081 :**		
Name	FrNmCoordinatorSyncSupport		
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.		

SWS Item	**ECUC_FrNm_00060 :**		
Name	FrNmCycleCounterEmulation		
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		

SWS Item	**ECUC_FrNm_00049 :**		
Name	FrNmDualChannelPduEnable		
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		

SWS Item	**ECUC_FrNm_00050 :**		
Name	FrNmHwVoteEnable		
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		

SWS Item	**ECUC_FrNm_00040 :**		
Name	FrNmNodeDetectionEnabled		
Description	Pre-processor switch for enabling node detection support. calculationFormula = If (FrNmPassiveModeEnabled == False) then Equal(NmNodeDetectionEnabled) else Equal(False)		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_FrNm_00043 :**		
Name	FrNmPassiveModeEnabled		
Description	Pre-processor switch for enabling Passive Node Configuration support. calculationFormula = Equal(NmPassiveModeEnabled)		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_FrNm_00046 :**		
Name	FrNmPduRxIndicationEnabled		
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		

SWS Item	**ECUC_FrNm_00067 :**		
Name	FrNmPnEiraCalcEnabled		
Description	Specifies if FrNm calculates the PN request information for internal an external requests. (EIRA) true: PN request are calculated false: PN request are not calculated		
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: only available if FrNmPnEnabled == true for at least one FrNm Channel		

SWS Item	**ECUC_FrNm_00068 :**		
Name	FrNmPnResetTime		
Description	Specifies the runtime of the reset timer in seconds. This reset time is valid for the reset of PN requests in the EIRA and in the ERA. The value shall be the same for every channel. Thus it is a global config parameter.		
Multiplicity	0..1		
Type	EcucFloatParamDef		
Range	0.001 .. 65.535		
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 FrNmPnEnabled == true for at least one FrNm Channel.		

SWS Item	**ECUC_FrNm_00044 :**		
Name	FrNmRemoteSleepIndicationEnabled		
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		

SWS Item	**ECUC_FrNm_00059 :**		
Name	FrNmRepeatMessageBitEnabled		
Description	Pre-processor switch for enabling the repeat message bit 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: calculationFormula = If (FrNmControlBitVectorEnabled == False) then Equal(False) else Equal(False or True)		

SWS Item	**ECUC_FrNm_00042 :**		
Name	FrNmSourceNodeIdentifierEnabled		
Description	Pre-processor switch for enabling SourceNodeIdentifier 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		

SWS Item	**ECUC_FrNm_00047 :**		
Name	FrNmStateChangeIndicationEnabled		
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		

SWS Item	**ECUC_FrNm_00039 :**		
Name	FrNmUserDataEnabled		
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		

SWS Item	**ECUC_FrNm_00073 :**		

Name	FrNmVotingNextToLastRepetitionCycleDisable		
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		

SWS Item	**ECUC_FrNm_00069 :**		
Name	FrNmPnEiraRxNSduRef		
Description	Reference to a Pdu in the COM-Stack. Only one SduRef is required for FrNm because the EIRA is the aggregation over all FlexRay Channels.		
Multiplicity	0..1		
Type	Reference to [Pdu]		
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 FrNmPnEnabled == true for at least one FrNm Channel.		

Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
FrNmPnInfo	0..1	PN information configuration

10.4.4 FrNmPnInfo

SWS Item	**ECUC_FrNm_00061 :**
Container Name	FrNmPnInfo
Description	PN information configuration
Configuration Parameters	

SWS Item	**ECUC_FrNm_00063 :**	
Name	FrNmPnInfoLength	
Description	Specifies the length of the PN request information in the NM message.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	1 .. 7	
Default value	1	
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: only available if FrNmPnEnabled == true for at least one FrNm Channel.		

SWS Item	**ECUC_FrNm_00062 :**		
Name	FrNmPnInfoOffset		
Description	Specifies the offset of the PN request information in the NM message.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 7		
Default value	1		
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: only available if FrNmPnEnabled == true for at least one FrNm Channel.		

Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
FrNmPnFilterMaskByte	0..7	Filter mask byte configuration

10.4.5 FrNmPnFilterMaskByte

SWS Item	**ECUC_FrNm_00064 :**
Container Name	FrNmPnFilterMaskByte
Description	Filter mask byte configuration
Configuration Parameters	

SWS Item	**ECUC_FrNm_00065 :**		
Name	FrNmPnFilterMaskByteIndex		
Description	Index of the filter mask byte. Specifies the position within the filter mask byte array.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 6		
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: only available if FrNmPnEnabled == true for at least one FrNm Channel. FrNmPnFilterMaskByteIndex < FrNmPnInfoLength		

SWS Item	**ECUC_FrNm_00066 :**

Name	FrNmPnFilterMaskByteValue		
Description	Parameter to configure the filter mask byte.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	0		
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: only available if FrNmPnEnabled== true for at least one FrNm Channel; FrNmPnFilterMaskByteIndex < FrNmPnInfoLength		

No Included Containers

10.4.6 FrNmGlobalProperties

SWS Item	**ECUC_FrNm_00003 :**
Container Name	FrNmGlobalProperties
Description	This container contains module properties related to the FlexRay NM functionality.
Configuration Parameters	

SWS Item	**ECUC_FrNm_00036 :**		
Name	FrNmDevErrorDetect		
Description	Switches the Default Error Tracer (Det) detection and notification ON or OFF. • true: enabled (ON). • false: disabled (OFF).		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

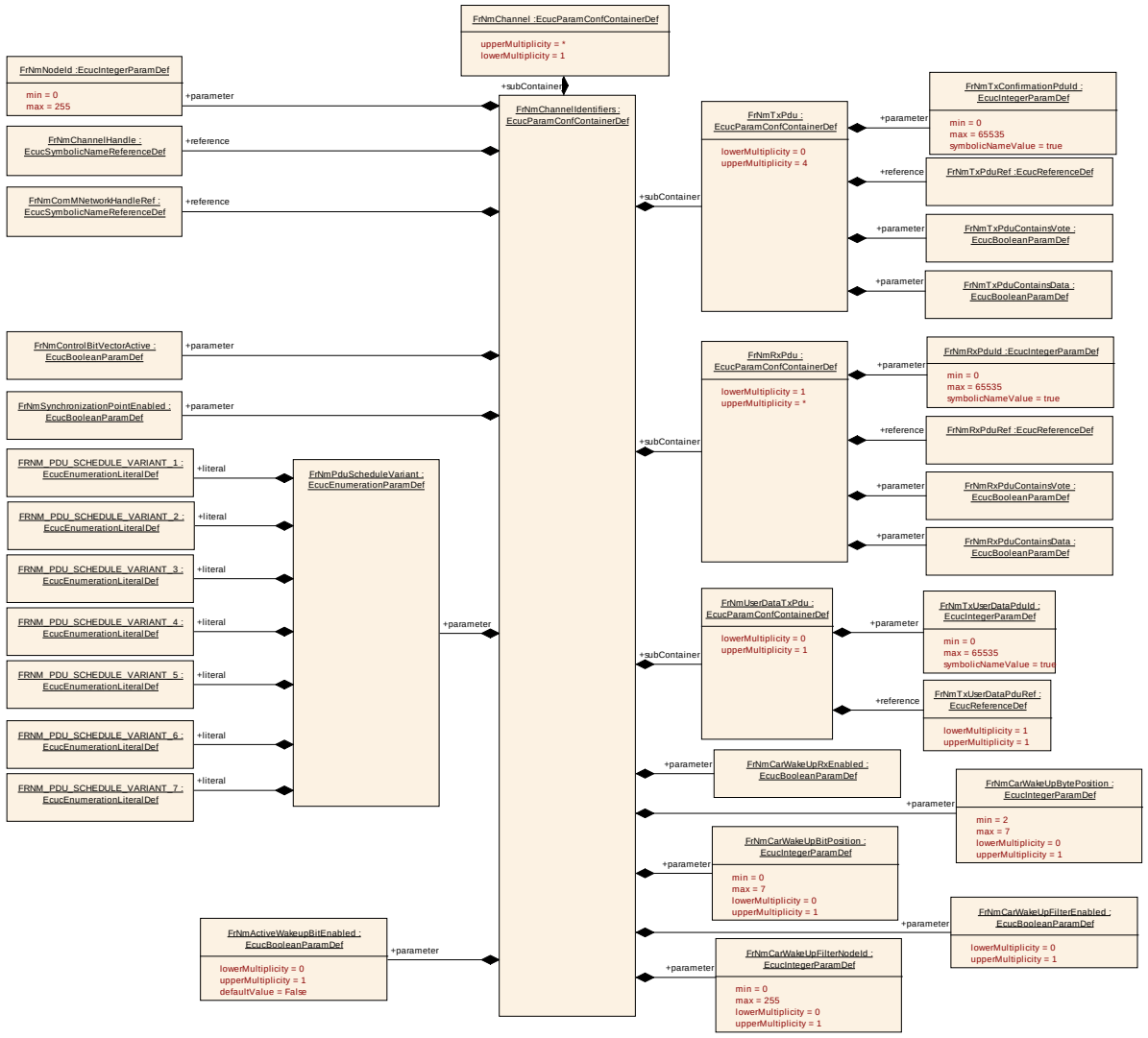
SWS Item	**ECUC_FrNm_00038 :**
Name	FrNmMainAcrossFrCycle
Description	If the FlexRay NM MainFunction is executed completely within the FlexRay communication cycle where the last NM vote of the current vote cycle is received, the FrNmMainAcrossFrCycle shall be configured to FALSE. If the FlexRay NM MainFunction 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		

SWS Item	**ECUC_FrNm_00037 :**		
Name	FrNmVersionInfoApi		
Description	Pre-processor switch for enabling version info API 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		

No Included Containers

10.5 Channel configurable parameters



[SWS_FrNm_00036] ⌈The following runtime configurable parameters shall be configurable for each channel separately.⌋ (SRS_Nm_00149)

10.5.1 FrNmChannelConfig

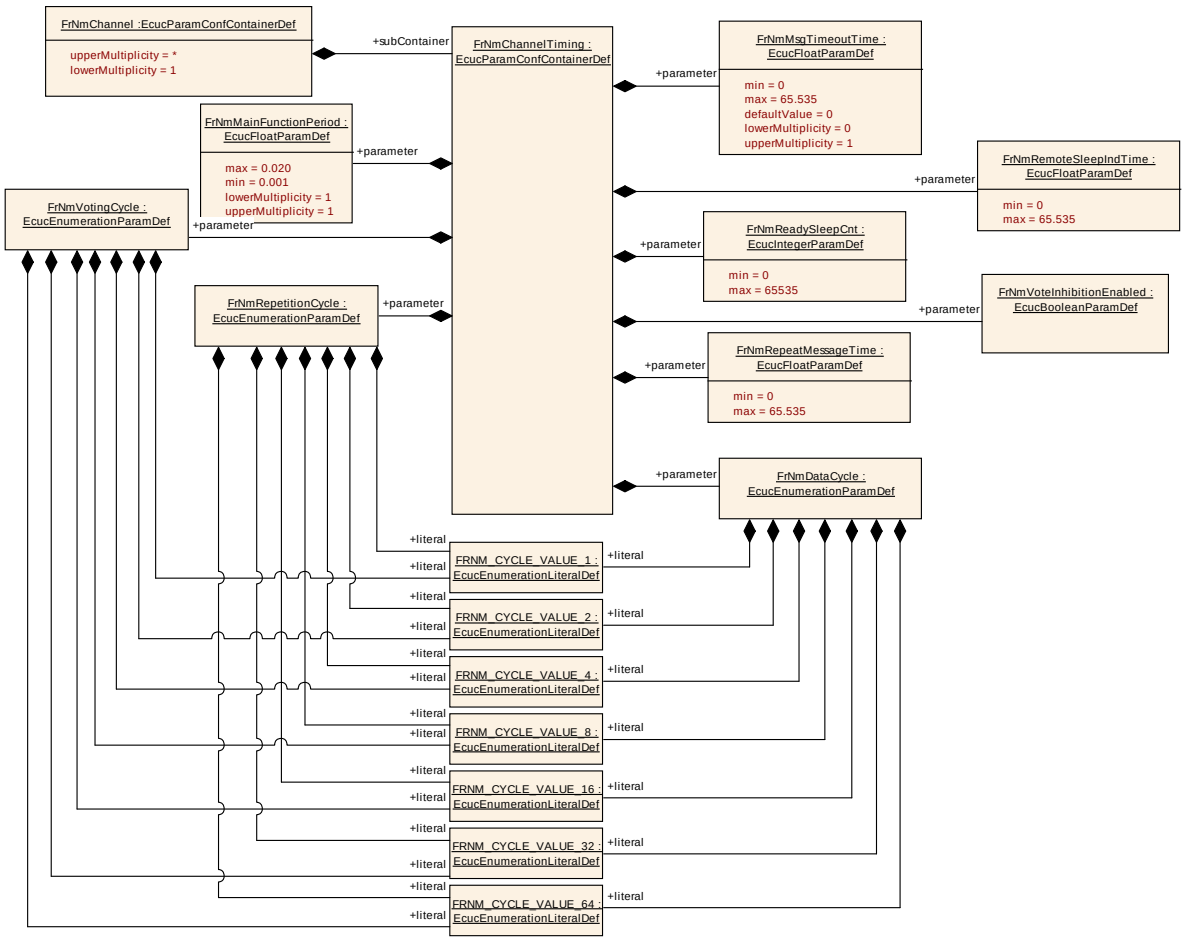
SWS Item	**ECUC_FrNm_00002 :**
Container Name	FrNmChannelConfig
Description	This container contains the configuration parameters for all FlexRay NM channels.
Configuration Parameters	

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

10.5.2 FrNmChannel

SWS Item	**ECUC_FrNm_00006 :**		
Container Name	FrNmChannel		
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			

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.



10.5.3 FrNmChannelTiming

SWS Item	**ECUC_FrNm_00008 :**

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

SWS Item	**ECUC_FrNm_00031 :**		
Name	FrNmDataCycle		
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		

SWS Item	**ECUC_FrNm_00035 :**		
Name	FrNmMainFunctionPeriod		
Description	This parameter defines the processing cycle of the main function of FrNm module in seconds.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	0.001 .. 0.02		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_FrNm_00028 :**		
Name	FrNmMsgTimeoutTime		
Description	Timeout of a NM-message. It determines in seconds how long the NM shall wait with notification of transmission failure while communication errors occur on the bus.		
Multiplicity	0..1		
Type	EcucFloatParamDef		
Range	0 .. 65.535		
Default value	0		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME

	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local dependency: Value of FrNmMsgTimeoutTime has to be greater than the duration of one FrNm voting cycle (can be calculated by multiplying FrNmVotingCycle with the duration of one FlexRay cycle). Parameter is only relevant if FrNmPassiveModeEnabled == False.		

SWS Item	**ECUC_FrNm_00051 :**		
Name	FrNmReadySleepCnt		
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		

SWS Item	**ECUC_FrNm_00029 :**		
Name	FrNmRemoteSleepIndTime		
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 FrNmRemoteSleepIndTime = 0		

SWS Item	**ECUC_FrNm_00030 :**
Name	FrNmRepeatMessageTime
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.		

SWS Item	**ECUC_FrNm_00033 :**		
Name	FrNmRepetitionCycle		
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]		

SWS Item	**ECUC_FrNm_00053 :**		
Name	FrNmVoteInhibitionEnabled		
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		

SWS Item	**ECUC_FrNm_00032 :**	
Name	FrNmVotingCycle	
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		

No Included Containers

10.5.4 FrNmChannelIdentifiers

SWS Item	**ECUC_FrNm_00007 :**
Container Name	FrNmChannelIdentifiers
Description	This container contains instance specific identifiers related to the respective FlexRay Channel.
Configuration Parameters	

SWS Item	**ECUC_FrNm_00082 :**		
Name	FrNmActiveWakeupBitEnabled		
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 dependency: This parameter is only valid if FrNmPassiveModeEnabled is False.		

SWS Item	**ECUC_FrNm_00076 :**	
Name	FrNmCarWakeUpBitPosition	
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		

SWS Item	**ECUC_FrNm_00075 :**		
Name	FrNmCarWakeUpBytePosition		
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		

SWS Item	**ECUC_FrNm_00077 :**		
Name	FrNmCarWakeUpFilterEnabled		
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		

SWS Item	**ECUC_FrNm_00078 :**		
Name	FrNmCarWakeUpFilterNodeId		
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		

SWS Item	**ECUC_FrNm_00074 :**		
Name	FrNmCarWakeUpRxEnabled		
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		

SWS Item	**ECUC_FrNm_00020 :**		
Name	FrNmControlBitVectorActive		
Description	This parameter is used to activate or deactivate the control bit vector support for a Fr Nm 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: This parameter may only be set to TRUE if FrNmControlBitVectorEnabled is set to True		

SWS Item	**ECUC_FrNm_00017 :**

Name	FrNmNodeId		
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		

SWS Item	**ECUC_FrNm_00022 :**		
Name	FrNmPduScheduleVariant		
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		

SWS Item	**ECUC_FrNm_00072 :**
Name	FrNmPnEnabled
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		

SWS Item	**ECUC_FrNm_00071 :**		
Name	FrNmPnEraCalcEnabled		
Description	Specifies if FrNm calculates the PN request information for external requests. (ERA) false: PN request are not calculated true: PN request are calculated		
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: only available if FrNmPnEnabled == true		

SWS Item	**ECUC_FrNm_00021 :**		
Name	FrNmSynchronizationPointEnabled		
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		

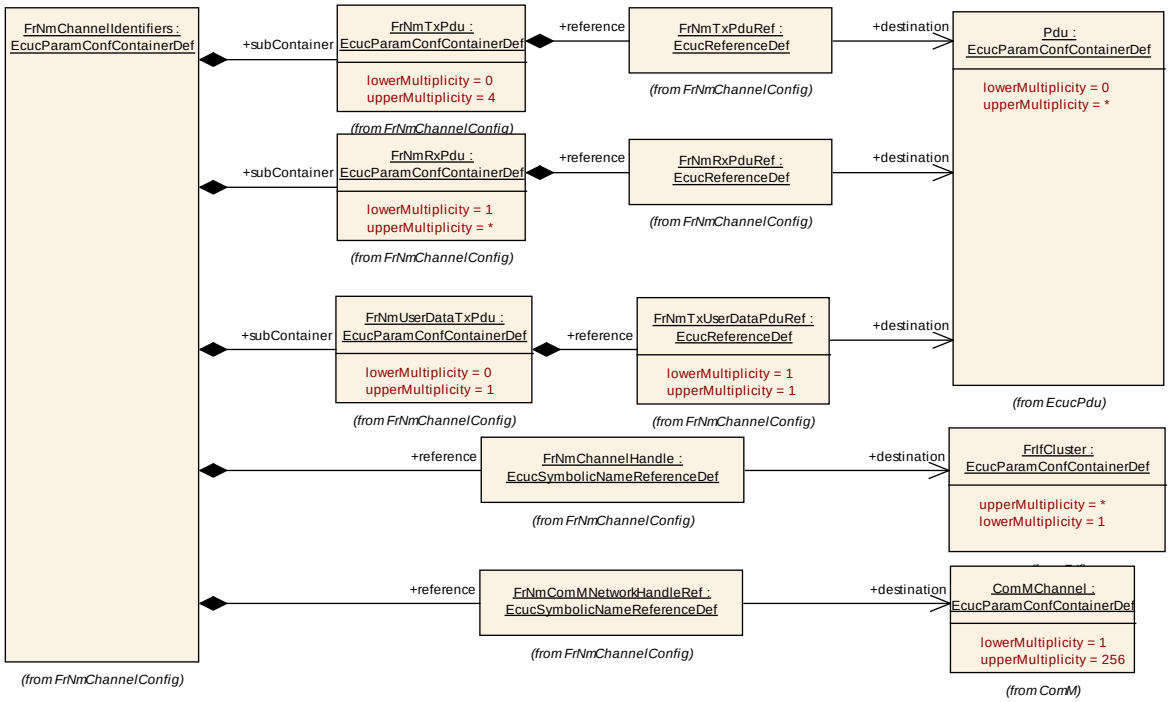
SWS Item	**ECUC_FrNm_00013 :**
Name	FrNmChannelHandle
Description	Channel identifier configured for the respective instance of the NM. The FrNmChannelHandle shall be encoded in the FrNmRxPduId parameter which is passed to FrNm_RxIndication() function called by the FrIf.
Multiplicity	1
Type	Symbolic name 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		

SWS Item	**ECUC_FrNm_00014 :**		
Name	FrNmComMNetworkHandleRef		
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.		

SWS Item	**ECUC_FrNm_00070 :**		
Name	FrNmPnEraRxNSduRef		
Description	Reference to a global Pdu. The SduRef is required for every FrNm Channel, because ERA is reported per channel.		
Multiplicity	0..1		
Type	Reference to [Pdu]		
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 FrNmPnEraCalcEnabled == true		

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.



10.5.5 FrNmRxPdu

SWS Item	**ECUC_FrNm_00010 :**
Container Name	FrNmRxPdu
Description	This container describes the FlexRay NM RX PDU:s.
Configuration Parameters	

SWS Item	**ECUC_FrNm_00027 :**		
Name	FrNmRxPduContainsData		
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		

SWS Item	**ECUC_FrNm_00026 :**		
Name	FrNmRxPduContainsVote		
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		

SWS Item	**ECUC_FrNm_00025 :**		
Name	FrNmRxPduId		
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	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_FrNm_00012 :**		
Name	FrNmRxPduRef		
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 FrIf 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		

No Included Containers

10.5.6 FrNmTxPdu

SWS Item	**ECUC_FrNm_00009 :**
Container Name	FrNmTxPdu
Description	This container describes the FlexRay NM TX PDU:s.
Configuration Parameters	

SWS Item	**ECUC_FrNm_00018 :**		
Name	FrNmTxConfirmationPduId		
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	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_FrNm_00024 :**		
Name	FrNmTxPduContainsData		
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		

SWS Item	**ECUC_FrNm_00023 :**		
Name	FrNmTxPduContainsVote		
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		

SWS Item	**ECUC_FrNm_00011 :**		
Name	FrNmTxPduRef		
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 FrIf 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		

No Included Containers

10.5.7 FrNmUserDataTxPdu

SWS Item	**ECUC_FrNm_00055 :**
Container Name	FrNmUserDataTxPdu
Description	This optional container is used to configure the UserNm PDU. This container is only available if FrNmComUserDataSupport is enabled.
Configuration Parameters	

SWS Item	**ECUC_FrNm_00056 :**
Name	FrNmTxUserDataPduId
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		

SWS Item	**ECUC_FrNm_00057 :**		
Name	FrNmTxUserDataPduRef		
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		

No Included Containers

10.6 Published parameters

For details refer to the chapter 10.3 “Published Information” in *SWS_BSWGeneral.*

10.7 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) ⌋ ( )

10.8 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 ([6]) contain the necessary information.

10.8.1 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 10-1 (unterhalb).

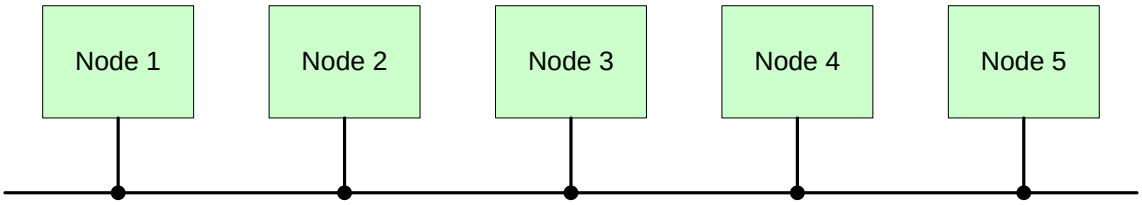


Figure 10-1 Example of five Node Network

The FlexRay Schedule allots 5 slots in the static segment and 6 (mini)slots in the dynamic segment as shown in Figure 10-2 (unterhalb). 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.

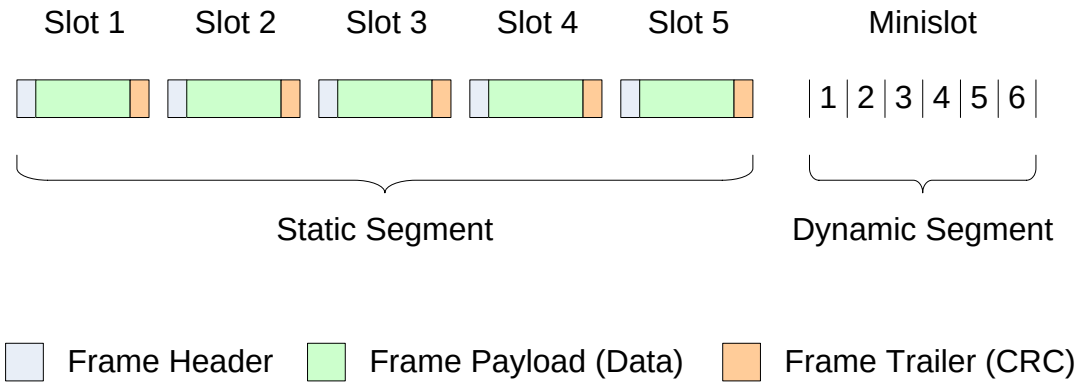


Figure 10-2 Example of Bus Schedule

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 10-3 (Seite 119).

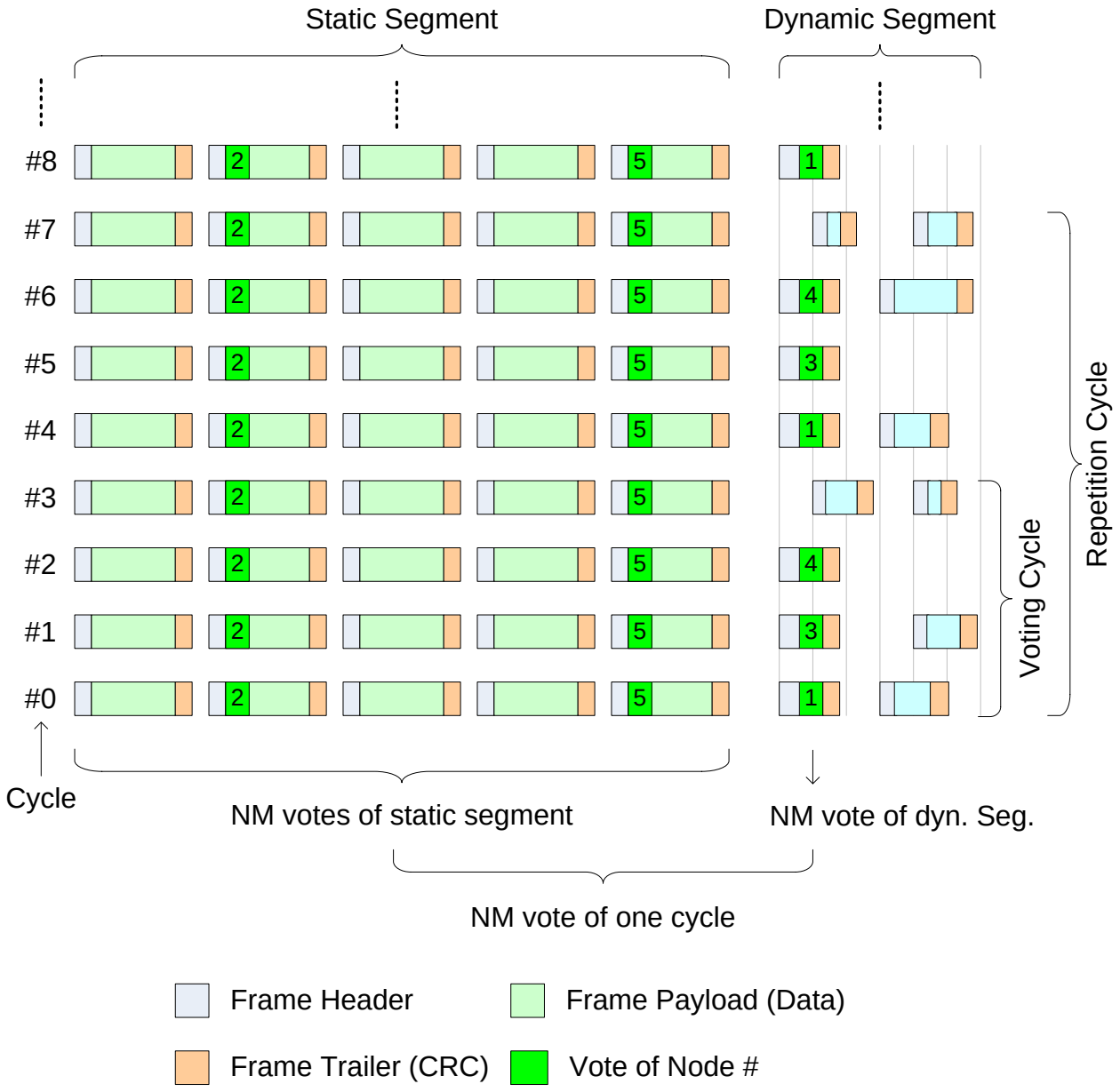


Figure 10-3 Example of NM-Vote in dynamic and static segment

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.

10.8.2 Example of Bus Schedule with NM-Data PDUs

This example uses the same setup as in the previous example (10.8.1) – five nodes (Figure 10-1 Seite 118), 5 slot in the static segment and 6 slots in the dynamic segment (Figure 10-2 Seite 118).

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 10-4 (unterhalb) 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.

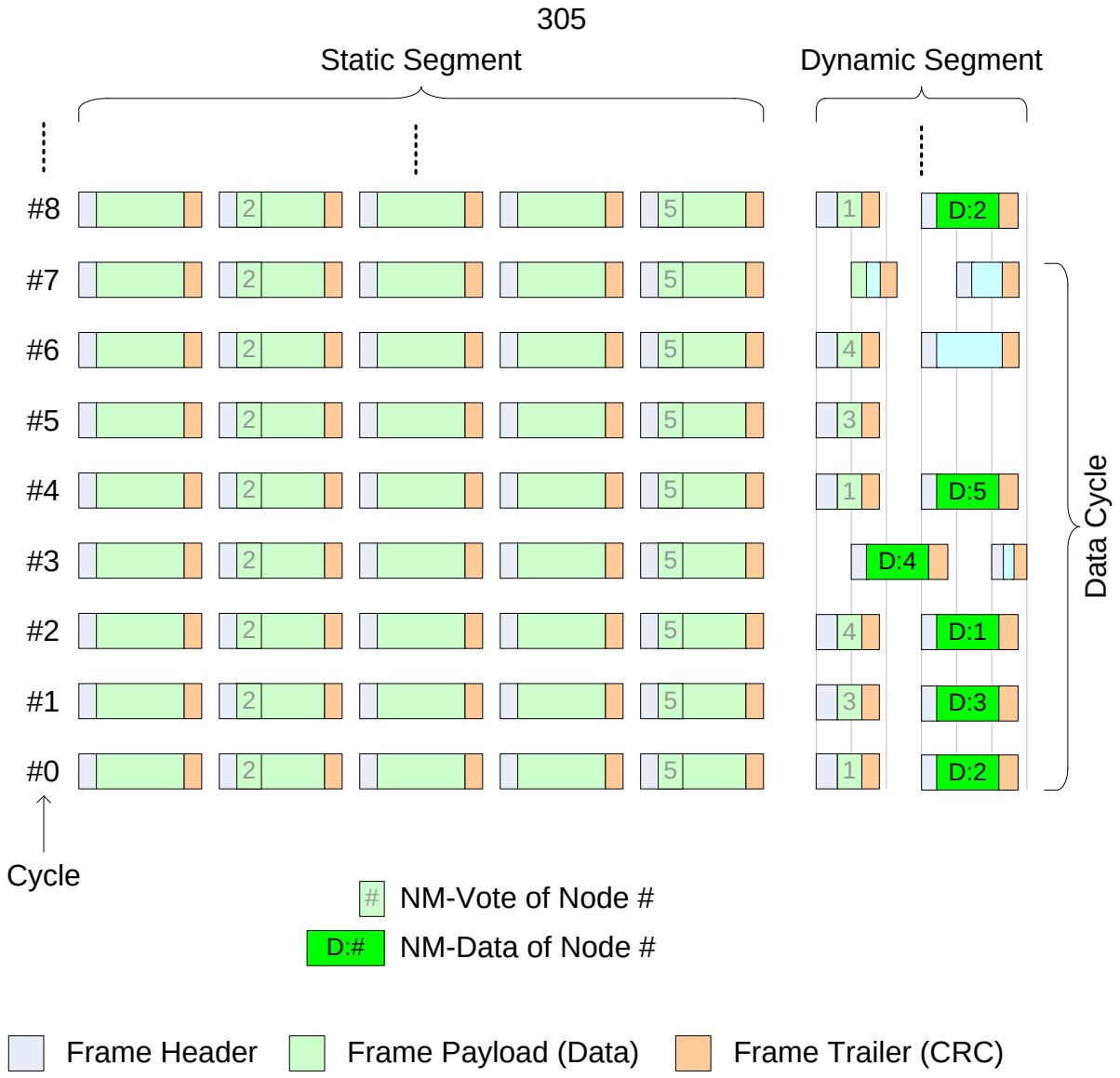


Figure 10-4 Example of Bus Schedule with NM-Data

11 Not applicable requirements

[SWS_FrNm_00401] ⌈These requirements are not applicable to this specification.⌋ (SRS_BSW_00375, SRS_BSW_00168, SRS_BSW_00423, SRS_BSW_00426, SRS_BSW_00427, BSW00431, SRS_BSW_00433, BSW00434, SRS_BSW_00336, SRS_BSW_00417, SRS_BSW_00161, SRS_BSW_00162, SRS_BSW_00005, SRS_BSW_00164, SRS_BSW_00325, SRS_BSW_00326, SRS_BSW_00347, SRS_BSW_00314, SRS_BSW_00009, SRS_BSW_00401, SRS_BSW_00172, SRS_BSW_00010, SRS_BSW_00333, SRS_BSW_00341, SRS_BSW_00334, SRS_Nm_00044, BSW136, BSW140, SRS_Nm_00145, SRS_Nm_00146, SRS_Nm_00148, BSW139, SRS_Nm_02510)