

| | |
|-----------------------------------|------------------------------------|
| Document Title | Specification of LIN State Manager |
| Document Owner | AUTOSAR |
| Document Responsibility | AUTOSAR |
| Document Identification No | 255 |
| Document Classification | Standard |

| | |
|-------------------------|-------|
| Document Version | 1.3.0 |
| Document Status | Final |
| Part of Release | 4.0 |
| Revision | 3 |

| Document Change History | | | |
|-------------------------|---------|------------------------|--|
| Date | Version | Changed by | Change Description |
| 02.11.2011 | 1.3.0 | AUTOSAR Administration | <ul style="list-style-type: none"> Added post-build configuration support Added completion of Production error concept in Com Stack Removed local network index |
| 13.10.2010 | 1.2.0 | AUTOSAR Administration | <ul style="list-style-type: none"> Post-build configuration variant added Module version check changed according SRS_General BSW004 TrcvModeType definition moved from LinIf to LinTrcv |
| 10.12.2009 | 1.1.0 | AUTOSAR Administration | <ul style="list-style-type: none"> Controlling of I-PDU groups has been moved to the BSW Mode Manager module Interface to the LIN Transceiver module has been introduced since LIN Transceiver driver is a new module in release 4.0 Legal disclaimer revised |
| 23.06.2008 | 1.0.1 | AUTOSAR Administration | Legal disclaimer revised |
| 03.12.2007 | 1.0.0 | AUTOSAR Administration | Initial Release |

Disclaimer

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

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

This specification may be utilized or reproduced without any modification, in any form or by any means, for informational purposes only.

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

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

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

Advice for users

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

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

Table of Contents

| | | |
|---------|--|----|
| 1 | Introduction and functional overview | 6 |
| 1.1 | Architectural overview | 6 |
| 1.2 | Functional overview..... | 7 |
| 2 | Acronyms and abbreviations | 8 |
| 3 | Related documentation..... | 9 |
| 3.1 | Input documents..... | 9 |
| 3.2 | Related standards and norms | 9 |
| 4 | Constraints and assumptions | 10 |
| 4.1 | Limitations | 10 |
| 4.2 | Applicability to car domains..... | 10 |
| 5 | Dependencies to other modules..... | 11 |
| 5.1 | Relation to Upper layers..... | 12 |
| 5.1.1 | Operating System | 12 |
| 5.1.2 | Module DET (Development Error Tracer) | 13 |
| 5.1.3 | Module DEM (Diagnostic Event Manager) | 13 |
| 5.1.4 | ComM | 13 |
| 5.1.5 | BSW Mode Manager..... | 13 |
| 5.2 | Relation to Lower layers..... | 13 |
| 5.2.1 | LinIf | 13 |
| 5.3 | File structure | 14 |
| 5.3.1 | Code file structure | 14 |
| 5.3.2 | Header File structure..... | 14 |
| 5.3.2.1 | LinSM header files..... | 14 |
| 5.3.2.2 | Included header files | 15 |
| 6 | Requirements traceability | 17 |
| 7 | Functional specification | 21 |
| 7.1 | States and transitions of the LinSM state machine..... | 21 |
| 7.1.1 | LINSM_UNINIT | 22 |
| 7.1.2 | LINSM_INIT | 22 |
| 7.1.3 | LINSM_NO_COM | 23 |
| 7.1.4 | LINSM_FULL_COM | 24 |
| 7.1.5 | Goto sleep..... | 25 |
| 7.1.6 | Changing schedule table..... | 26 |
| 7.1.7 | Wake up process | 26 |
| 7.1.8 | Timeout of requests | 27 |
| 7.2 | Handling multiple networks and drivers..... | 28 |
| 7.2.1 | Multiple networks | 28 |
| 7.3 | Error classification | 29 |
| 7.4 | Error detection..... | 30 |
| 7.5 | Error notification | 30 |
| 7.6 | Debugging..... | 30 |
| 8 | API specification..... | 32 |

| | | |
|----------|---|----|
| 8.1 | Imported types..... | 32 |
| 8.1.1 | Standard types | 32 |
| 8.2 | Type definitions | 32 |
| 8.2.1 | LinSM_ModeType | 32 |
| 8.2.2 | LinSM_ConfigType..... | 32 |
| 8.3 | LinSM API | 33 |
| 8.3.1 | LinSM_Init | 33 |
| 8.3.2 | LinSM_ScheduleRequest..... | 33 |
| 8.3.3 | LinSM_GetVersionInfo | 34 |
| 8.3.4 | LinSM_GetCurrentComMode..... | 35 |
| 8.3.5 | LinSM_RequestComMode | 36 |
| 8.4 | Scheduled Functions | 37 |
| 8.4.1 | LinSM_MainFunction | 38 |
| 8.5 | LinSM callbacks | 38 |
| 8.5.1 | LinSM_ScheduleRequestConfirmation | 38 |
| 8.5.2 | LinSM_GotoSleepConfirmation..... | 39 |
| 8.5.3 | LinSM_WakeupConfirmation..... | 40 |
| 8.6 | Mandatory Interfaces..... | 40 |
| 8.7 | Optional Interfaces | 41 |
| 8.8 | Configurable Interfaces | 41 |
| 9 | Sequence diagrams | 42 |
| 9.1 | Goto-sleep process | 43 |
| 9.2 | Internal wake up..... | 44 |
| 9.3 | Schedule switch | 45 |
| 10 | Configuration specification | 47 |
| 10.1 | How to read this chapter | 47 |
| 10.1.1 | Configuration and configuration parameters | 47 |
| 10.1.2 | Containers..... | 47 |
| 10.1.3 | Specification template for configuration parameters | 47 |
| 10.2 | Containers and configuration parameters | 48 |
| 10.2.1 | Configuration Tool..... | 48 |
| 10.2.2 | Variants..... | 48 |
| 10.2.2.1 | VARIANT-PRE-COMPILE | 49 |
| 10.2.2.2 | VARIANT-LINK-TIME | 49 |
| 10.2.2.3 | VARIANT-POST_BUILD..... | 49 |
| 10.3 | LinSM_Configuration..... | 49 |
| 10.3.1 | LinSM..... | 49 |
| 10.3.2 | LinSMConfigSet | 49 |
| 10.3.3 | LinSMChannel..... | 50 |
| 10.3.4 | LinSMGeneral | 51 |
| 10.3.5 | LinSMSchedule | 52 |
| 10.4 | Published Information..... | 53 |
| 11 | Changes between Release 3.0/3.1 and 4.0 | 54 |
| 11.1 | Deleted SWS Items | 54 |
| 11.2 | Changed SWS Items..... | 54 |
| 11.3 | Added SWS Items..... | 54 |
| 12 | Not applicable requirements..... | 56 |

1 Introduction and functional overview

This specification specifies the functionality, API and the configuration of the AUTOSAR Basic Software module LIN State Manager (LinSM). The LinSM together with the LIN Interface, LIN driver, LIN Transceiver driver forms the complete LIN protocol.

The LIN State Manager is designed to be hardware independent.

The LinSM is dependent on upper module Communication Manager [11] (ComM) and lower module LIN Interface [7] (LinIf).

It is assumed that the reader is familiar with the LIN 2.1 specification [13]. This document will not describe functionality already described in the LIN 2.1 specification [13].

Note that figures in this document are not regarded as requirements. All requirements are described in text prefixed with a requirement tag (e.g. LINSM042). Text not prefixed with a requirement shall be seen as informative text.

1.1 Architectural overview

The Layered Software Architecture [1] positions the LinSM within the BSW architecture as shown below.

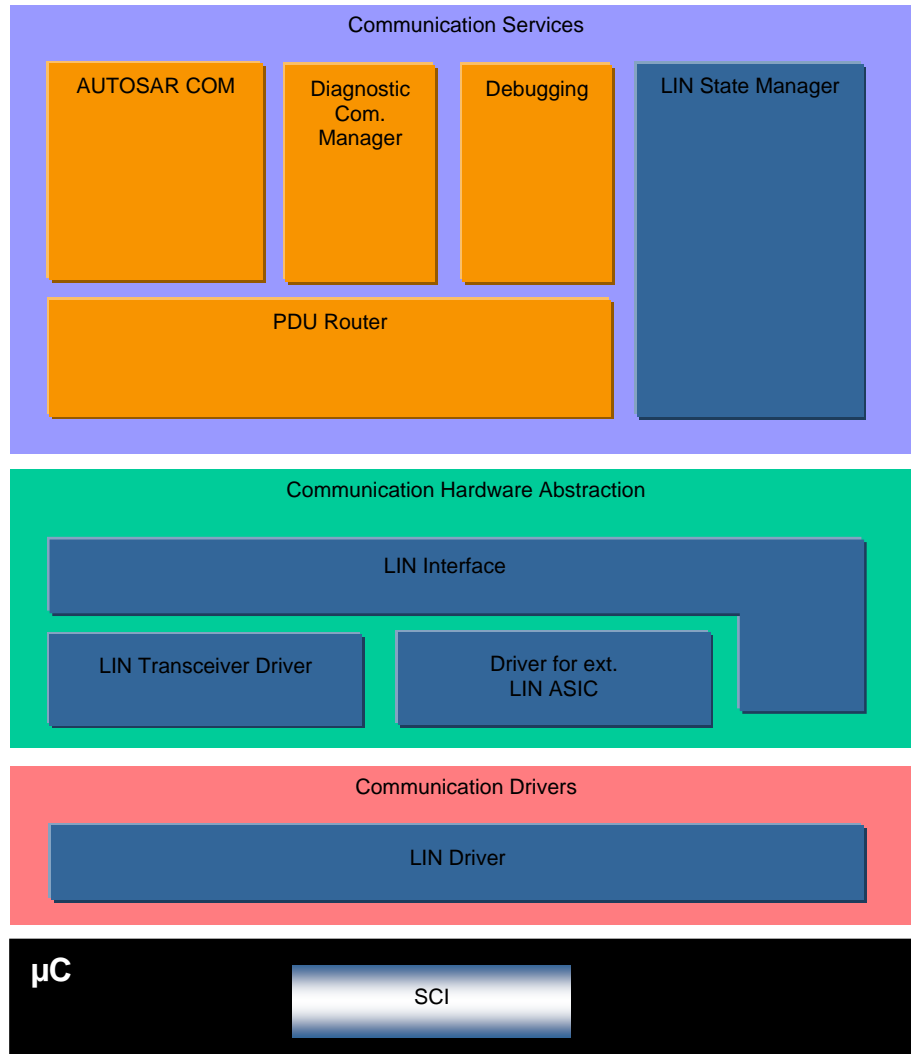


Figure 1 – AUTOSAR BSW software architecture - LIN stack scope

1.2 Functional overview

The LinSM is responsible for the control flow of the LIN Bus.

This means:

- Switching schedule tables when requested by the upper layer(s).
- Go to sleep and wake up handling, when requested by the upper layer(s)
- Notification to upper layers when new state is entered.

2 Acronyms and abbreviations

Acronyms and abbreviations used in this document. Additional abbreviations can be found in the LIN 2.1 specification [13].

| Abbreviation / Acronym: | Description: |
|--------------------------------|---|
| API | Application Program Interface |
| AUTOSAR | Automotive Open System Architecture |
| BSW | Basic Software |
| BswM | BSW Mode Manager |
| ComM | Communication Manager |
| DCM | Diagnostic Communication Manager |
| DEM | Diagnostic Event Manager |
| DET | Development Error Tracer |
| ECU | Electric Control Unit |
| ID | Identifier |
| ISR | Interrupt Service Routine |
| Jitter | Difference between longest delay and shortest delay (e.g. Worst case execution time – Best case execution time) |
| LIN | Local Interconnect Network |
| LinIf | LIN Interface |
| LinSM | LIN State Manager (the subject of this document) |
| MCAL | Microcontroller Abstraction Layer |
| PDU | Protocol Data Unit |
| RAM | Random Access memory |
| RTE | Run Time Environment |
| RX | Receive |
| SPAL | Standard Peripheral Abstraction Layer |
| SRS | Software Requirement Specification |
| SW | Software |
| SWS | Software Design Specification |
| TP | Transport Protocol |
| TX | Transmit |
| UART | Universal Asynchronous Receiver Transmitter |
| UML | Universal Modelling Language |
| URL | Uniform Resource Locator |
| WP11 | Work Package in AUTOSAR phase 2 |
| XML | Extensible Markup Language |

3 Related documentation

3.1 Input documents

- [1] List of Basic Software Modules
AUTOSAR_TR_BSWModuleList.pdf
- [2] Layered Software Architecture,
AUTOSAR_EXP_LayeredSoftwareArchitecture.pdf
- [3] General Requirements on Basic Software Modules
AUTOSAR_SRS_BSWGeneral.pdf
- [4] Specification of Standard Types
AUTOSAR_SWS_StandardTypes.pdf
- [5] Specification of Development Error Tracer
AUTOSAR_SWS_DevelopmentErrorTracer.pdf
- [6] Requirements on LIN
AUTOSAR_SRS_LIN.pdf
- [7] Specification of LIN Interface
AUTOSAR_SWS_LINInterface.pdf
- [8] Specification of Diagnostic Event Manager
AUTOSAR_SWS_DiagnosticEventManager.pdf
- [9] Specification of ECU Configuration
AUTOSAR_TPS_ECUConfiguration.pdf
- [10] Specification of LIN Driver
AUTOSAR_SWS_LINDriver.pdf
- [11] Specification of Communication Manager
AUTOSAR_SWS_COMManager.pdf
- [12] Specification of Basic Software Mode Manager
AUTOSAR_SWS_BSWModeManager.pdf

3.2 Related standards and norms

- [13] LIN Specification Package Revision 2.1, September 23, 2003
<http://www.lin-subbus.org/>

4 Constraints and assumptions

4.1 Limitations

The LinSM module can only be used as a LIN master in a LIN cluster. There is at most one instance of the LinSM in each ECU. If the underlying LIN Driver [10] supports multiple networks, the LinSM may be master on more than one cluster.

4.2 Applicability to car domains

This specification is applicable to all car domains, where LIN is used.

5 Dependencies to other modules

This section describes the relations to other modules within the basic software. It describes the services that are used from these modules. Figure 2 shows the modules that are required or optional for the realization of the LinSM module. The figure is complete but is not regarded as requirement.

To be able for the LinSM module to operate the following modules will be interfaced:

[LINSM001] [LIN Interface – LinIf] (BSW00384)

[LINSM085] [Diagnostic Event Manager – DEM] (BSW00384)

[LINSM086] [Development Error Tracer (if enabled) – DET] (BSW00384)

[LINSM105] [Communication Manager – ComM] (BSW00384)

[LINSM196] [BSW Mode Manager - BswM] (BSW00384)

Note that modules that are using the interface (except callbacks) from this module are not listed.

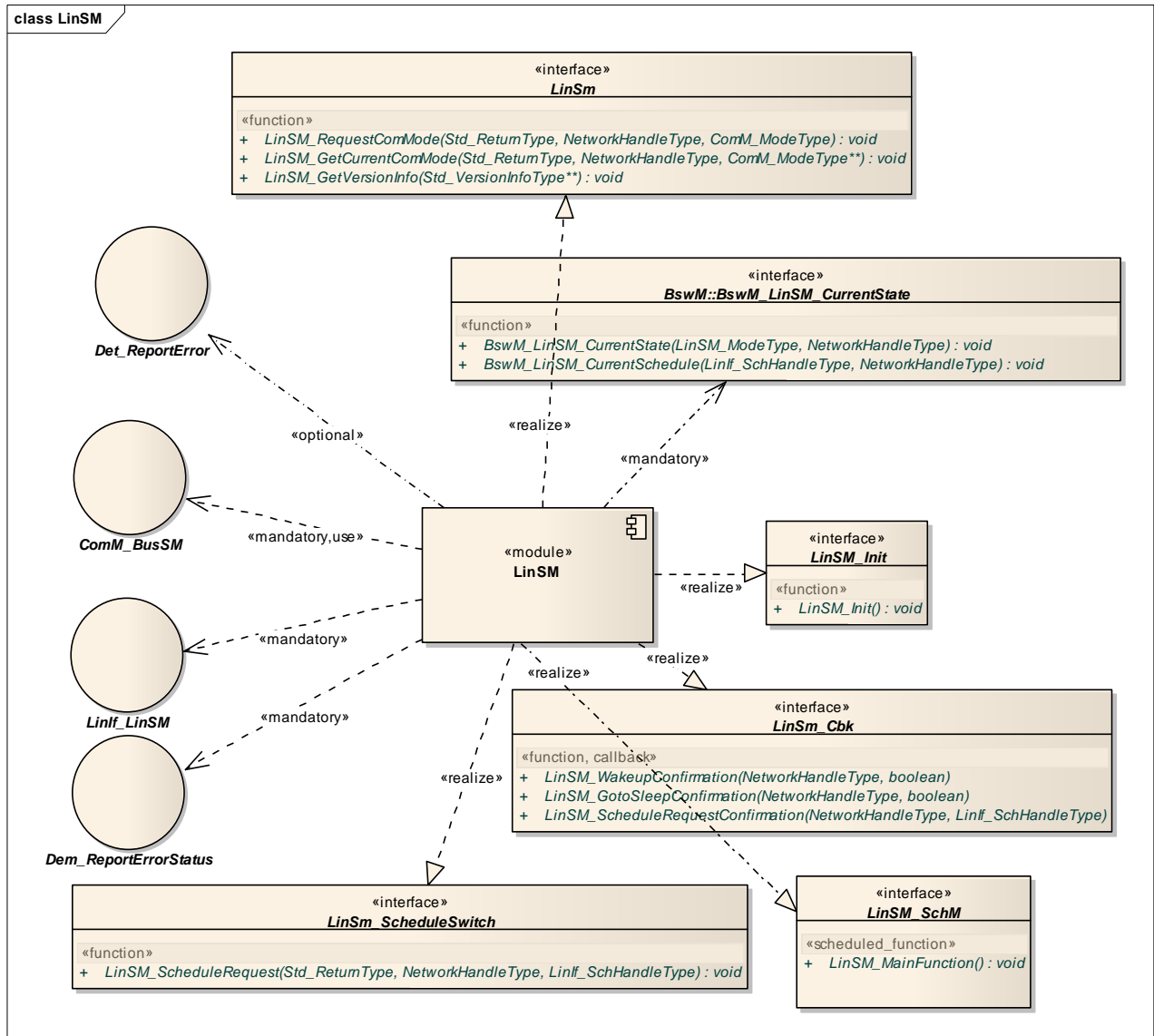


Figure 2 Dependencies to other modules

5.1 Relation to Upper layers

In principle, there is no requirement that specific modules shall call the interfaces of the LinSM module. Below, the normal users of LinSM module are listed.

5.1.1 Operating System

The LinSM module does contain access of shared data with above or below modules (using the API). The data that is shared will not need a help of the OS to protect the data for consistency (there are no array accesses, only simple type accesses). However, there may be reentrant functions that access the same data in the LinSM module. It is up to the implementer to solve these accesses.

5.1.2 Module DET (Development Error Tracer)

In development mode the Det_ReportError – function of module DET [5] will be called.

5.1.3 Module DEM (Diagnostic Event Manager)

Production errors will be reported to the Diagnostic Event Manager [8] module.

5.1.4 ComM

The Com manager module requests the communication via the LIN stack and queries the state of the LinSM module.

5.1.5 BSW Mode Manager

The LinSM module will notify the BSW Mode Manager module [12] when a state is changed. The BSW Mode Manager module will interface the LinSM module when requesting a new schedule table.

5.2 Relation to Lower layers

Below are the BSW modules that will be interfaced by LinSM module.

5.2.1 LinIf

The LinSM module assumes the following primitives to be provided by the LinIf [7] module:

- Transmission of the goto-sleep command (LinIf_GoToSleep)
- The wakeup of the Lin bus (LinIf_Wakeup)
- Request to change schedule tables (LinIf_ScheduleRequest)

It is assumed that the LinIf module will call the following callbacks:

- Confirming the Wake Up signals has been transmitted (LinSM_WakeupConfirmation)
- Confirming that the goto-sleep command has been transmitted (LinSM_GotoSleepConfirmation)
- Confirming a schedule change (LinSM_ScheduleRequestConfirmation)

[LINSM002] [The LinSM module shall not use or access the LIN driver or assume information about it any way other than what the LinIf module provides through the function calls to the LinIf module listed above.] ()

5.3 File structure

5.3.1 Code file structure

This chapter describes the c-files that implement the LinSM module Configuration. The code file structure is not defined completely within this specification. It is up to each implementer to design the missing structure details.

The pre compile and link time configuration parameters has to be kept in separate files:

[LINSM003] [A c-file LinSM_Lcfg.c shall exist and contain all link time configurable parameters] (BSW00344, BSW00380, BSW00412)

[LINSM078] [A c-file LinSM_Cfg.c shall exist and contain all pre compile parameters that are “const”.] (BSW00380, BSW00412, BSW00419)

5.3.2 Header File structure

This chapter describes the header files that will be included by the LinSM module and possible other modules.

5.3.2.1 LinSM header files

Following header files will exist in a LinSM implementation:

[LINSM005] [A LinSM implementation shall provide a header file LinSM.h that contains all data exported from the LinSM – API declarations (except callbacks), extern types, and global data.] ()

[LINSM006] [A LinSM implementation shall provide a header file LinSM_Cbk.h that contains function declarations for the callback functions in the LinSM.] (BSW00370)

[LINSM007] [A header file LinSM_Cfg.h shall exist that contains the pre compile time parameters.] (BSW00345, BSW00381)

[LINSM195] [The LinSM implementation shall include the header file SchM_LinSM.h] (BSW00435)

5.3.2.2 Included header files

Figure 3 shows the header file structure of the LinSM realization.

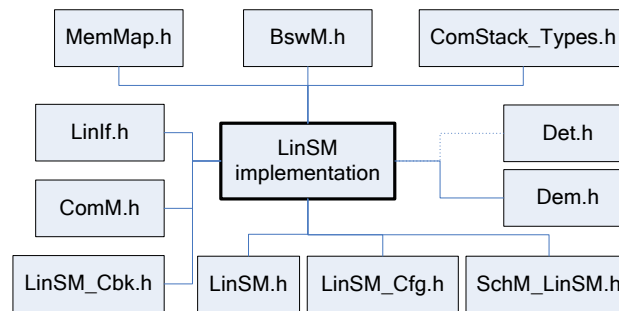


Figure 3 Header file structure

The LinSM implementation object in Figure 3 represent one or more .c files. It is not required to provide the complete implementation in one file.

Following external header files shall be included:

[LINSM011] [The LinSM module shall include the Dem.h file.] ()

[LINSM012] [The LinSM module shall include the LinIf.h file.] ()

[LINSM013] [The LinSM module shall include the ComM.h file] ()

[LINSM014] [The LinSM module shall include the MemMap.h file.] ()

[LINSM015] [The LinSM module shall include the Det.h file in case
LinSMDevErrorDetect is enabled] ()

[LINSM016] [The LinSM module shall include the ComStack_Types.h] ()

[LINSM201] [The LinSM module shall include the BswM.h] ()

[LINSM208] [The LinSM module shall perform a consistency check between code files and header files based on pre-process-checking the version numbers of related code files and header files.] (BSW004)

[LINSM209] [The LinSM module shall perform inter module checks to avoid integration of incompatible files. The imported header files shall be checked by preprocessing directives. The following version numbers shall be verified:

- <MODULENAME>_AR_RELEASE_MAJOR_VERSION

- <MODULENAME>_AR_RELEASE_MINOR_VERSION

where <MODULENAME> is the module short name of the other (external) modules which provide header files included by the LinSM module. If the values are not identical to the expected values, an error shall be reported.] (BSW004)

6 Requirements traceability

This chapter contains a matrix that shows the link between the SWS requirements defined for the LinSM and the input requirement documents (SRS).

| Requirement | Description | Satisfied by |
|-------------|---|-------------------------------|
| BSW00321 | These requirements are not applicable to this specification. | LINSM211 |
| BSW00323 | If the LinSMDevErrorDetect switch is enabled, API parameter checking is enabled. | LINSM055 |
| BSW00327 | | LINSM053 |
| BSW00331 | These requirements are not applicable to this specification. | LINSM211 |
| BSW00333 | These requirements are not applicable to this specification. | LINSM211 |
| BSW00334 | These requirements are not applicable to this specification. | LINSM211 |
| BSW00337 | | LINSM053 |
| BSW00338 | The detection of development errors is configurable (ON / OFF) at pre-compile time. | LINSM054 |
| BSW00339 | Production errors shall be reported to Diagnostic Event Manager (DEM) [8] module. | LINSM058 |
| BSW00341 | These requirements are not applicable to this specification. | LINSM211 |
| BSW00343 | These requirements are not applicable to this specification. | LINSM211 |
| BSW00344 | A c-file LinSM_Lcfg. | LINSM003 |
| BSW00345 | A header file LinSM_Cfg. | LINSM007 |
| BSW00350 | | LINSM053 |
| BSW00358 | | LINSM155 |
| BSW00359 | These requirements are not applicable to this specification. | LINSM211 |
| BSW00360 | These requirements are not applicable to this specification. | LINSM211 |
| BSW00369 | | LINSM113, LINSM126, LINSM122, |
| BSW00370 | A LinSM implementation shall provide a header file LinSM_Cbk. | LINSM006 |
| BSW00373 | | LINSM156 |
| BSW00375 | These requirements are not applicable to this specification. | LINSM211 |
| BSW00376 | | LINSM156 |
| BSW00380 | A c-file LinSM_Lcfg. | LINSM003, LINSM078 |
| BSW00381 | A header file LinSM_Cfg. | LINSM007 |

| | | | |
|----------|--|---|-------------------------------------|
| BSW00384 | LIN Interface - LinIf | LINSM001, LINSM086, LINSM196 | LINSM085, LINSM105, |
| BSW00385 | | LINSM053 | |
| BSW00387 | The LinSM_GotoSleepConfirmation shall be configurable. | LINSM199, LINSM198 | |
| BSW00399 | These requirements are not applicable to this specification. | LINSM211 | |
| BSW004 | The LinSM module shall perform a consistency check between code files and header files based on p... | LINSM208, LINSM209 | |
| BSW00400 | These requirements are not applicable to this specification. | LINSM211 | |
| BSW00404 | These requirements are not applicable to this specification. | LINSM211 | |
| BSW00405 | These requirements are not applicable to this specification. | LINSM211 | |
| BSW00406 | If LinSMDevErrorDetect is enabled: If the state LINSM_UNINIT is active then the error-code LINSM_... | LINSM116, LINSM128, LINSM131, LINSM134 | LINSM125, LINSM179, LINSM137, |
| BSW00407 | | LINSM117 | |
| BSW00409 | Values for production code Event Ids are assigned externally by the configuration of the Dem [8],... | LINSM051 | |
| BSW00411 | This function shall be pre compile time configurable On/Off by the configuration parameter: LinSM... | LINSM121 | |
| BSW00412 | A c-file LinSM_Lcfg. | LINSM003, LINSM078 | |
| BSW00414 | | LINSM155 | |
| BSW00415 | These requirements are not applicable to this specification. | LINSM211 | |
| BSW00416 | These requirements are not applicable to this specification. | LINSM211 | |
| BSW00417 | These requirements are not applicable to this specification. | LINSM211 | |
| BSW00419 | A c-file LinSM_Cfg. | LINSM078 | |
| BSW00422 | These requirements are not applicable to this specification. | LINSM211 | |
| BSW00425 | These requirements are not applicable to this specification. | LINSM211 | |
| BSW00432 | These requirements are not applicable to this specification. | LINSM211 | |
| BSW00433 | These requirements are not applicable to this specification. | LINSM211 | |
| BSW00435 | The LinSM implementation shall include the header file SchM_LinSM. | LINSM195 | |
| BSW00436 | These requirements are not applicable to this specification. | LINSM211 | |

| | | |
|----------|--|--|
| BSW00437 | These requirements are not applicable to this specification. | LINSM211 |
| BSW00438 | These requirements are not applicable to this specification. | LINSM211 |
| BSW00439 | These requirements are not applicable to this specification. | LINSM211 |
| BSW00440 | The LinSM_GotoSleepConfirmation shall be configurable. | LINSM199, LINSM198 |
| BSW00442 | Each variable that shall be accessible by AUTOSAR Debugging, shall be defined as global variable. | LINSM184, LINSM185, LINSM186, LINSM187, LINSM188, LINSM189 |
| BSW00443 | These requirements are not applicable to this specification. | LINSM211 |
| BSW00444 | These requirements are not applicable to this specification. | LINSM211 |
| BSW00445 | These requirements are not applicable to this specification. | LINSM211 |
| BSW00446 | These requirements are not applicable to this specification. | LINSM211 |
| BSW005 | These requirements are not applicable to this specification. | LINSM211 |
| BSW010 | These requirements are not applicable to this specification. | LINSM211 |
| BSW01560 | These requirements are not applicable to this specification. | LINSM211 |
| BSW01577 | These requirements are not applicable to this specification. | LINSM211 |
| BSW01590 | These requirements are not applicable to this specification. | LINSM211 |
| BSW101 | | LINSM155 |
| BSW161 | These requirements are not applicable to this specification. | LINSM211 |
| BSW162 | These requirements are not applicable to this specification. | LINSM211 |
| BSW167 | Detected development errors will be reported to the Det_ReportError API of the Development Error ... | LINSM057 |
| BSW168 | These requirements are not applicable to this specification. | LINSM211 |
| BSW170 | These requirements are not applicable to this specification. | LINSM211 |
| BSW171 | The detection of development errors is configurable (ON / OFF) at pre-compile time. | LINSM054 |

7 Functional specification

This chapter specifies the requirements on the module LinSM module. See the Basic Software Modules document [2] for an overview of the responsibilities of the LinSM.

The main responsibilities for the LinSM are:

- Control the communication status (no communication or full communication) of all LIN networks
- Handle schedule change requests
- Handle communication mode requests
- Notify of state changes to upper layers

The LinSM module will not directly implement functionality in the LIN specification. The LinSM module will support the behavior of the master in the LIN 2.1 specification. The master behavior provided by the LinSM module will allow the reuse of existing slaves conforming to the LIN 1.2, 1.3, 2.0 and 2.1 specifications.

[LINSM019] [The LinSM module shall be able to handle one or more LIN networks.] ()

Number of LIN networks are restricted by the LinIf specification. All networks are handled via the NetworkHandle_Type specified by the ComM module.

The identification of the LIN networks is made using reference in the configuration to the ComM network handles.

7.1 States and transitions of the LinSM state machine

The LinSM module will operate in a state-machine. Each network connected will operate in an independent sub-state-machine. The State Machine is depicted in Figure 4 shows a simplified version of the requirements below, the intention of the figure is not to be complete, rather give an overview.

[LINSM020] [The LinSM module shall have one state-machine containing the states LINSM_UNINIT and LINSM_INIT.] ()

[LINSM173] [In the LINSM_INIT there shall be a sub-state-machine for each network with the states LINSM_NO_COM and LINSM_FULL_COM.] ()

[LINSM021] [In LINSM_INIT each network may be in the sub-states LINSM_NO_COM or LINSM_FULL_COM independently.] ()

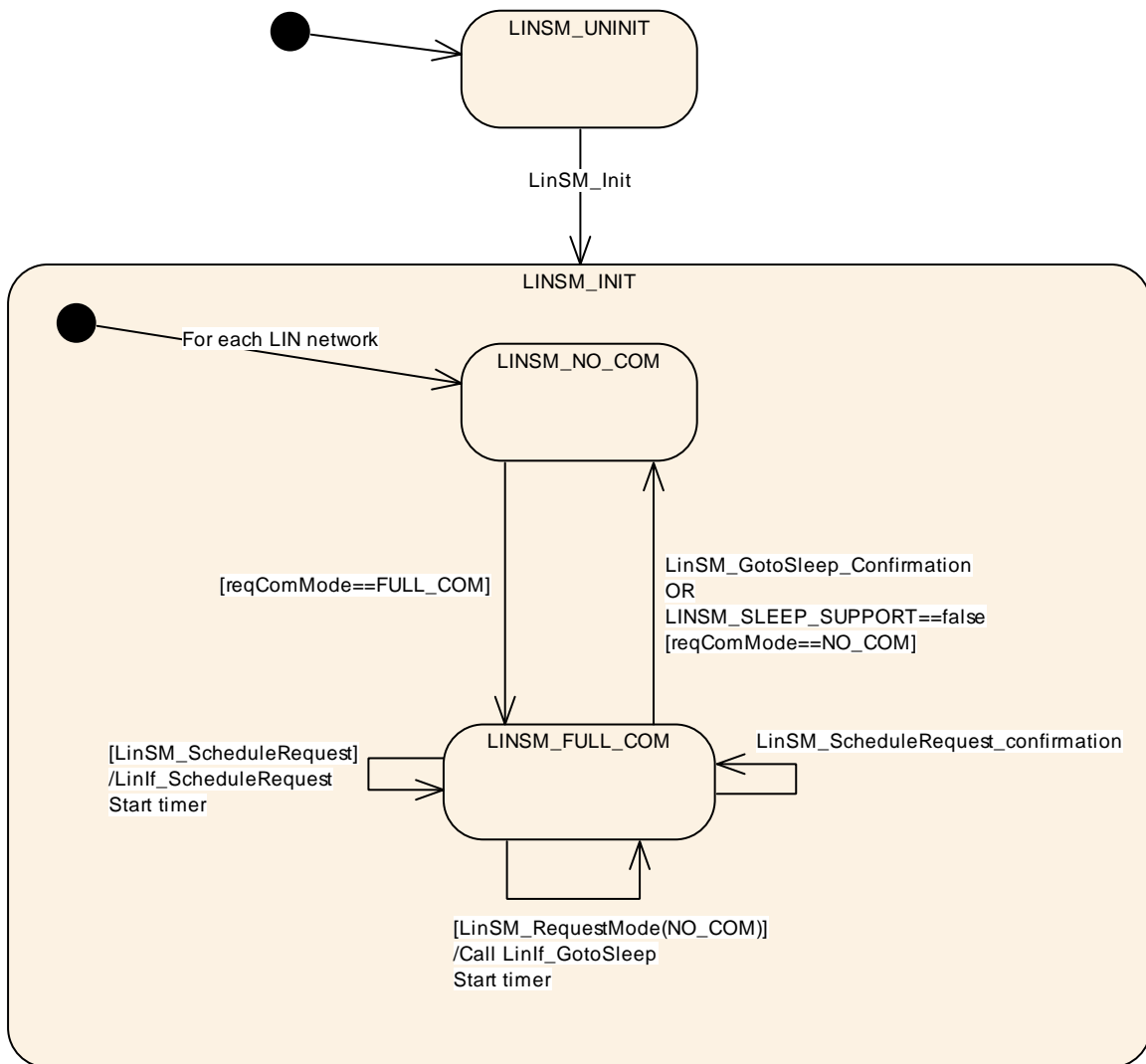


Figure 4: LinSM State Machine

7.1.1 LINSM_UNINIT

The uninit state is the first state that is active in the LinSM module.

[LINSM022] [There shall be a state called LINSM_UNINIT] ()

[LINSM161] [The state LINSM_UNINIT shall be active at start-up, before any API is called.] ()

7.1.2 LINSM_INIT

After the initialization is made the LinSM module will activate the init state. There are two sub-states in this state. One is the no communication state where no communication is made on the LIN bus, the other is full communication where

schedule tables are active and all communication is made. Each network may be in no communication or full communication independent of each other.

[LINSM024] [There shall be a state called LINSM_INIT] ()

[LINSM025] [The LinSM state-machine shall transit from any state or sub-state to the state LINSM_INIT when LinSM_Init is called.] ()

[LINSM152] [The LinSM state-machine shall transit from any state or sub-state to sub-state LINSM_NO_COM for all networks when LinSM_Init is called.] ()

[LINSM043] [When entering LINSM_INIT the LinSM shall be put in an init state. Init state means that global variables, etc, shall be set to default value (reset value).] ()

[LINSM160] [The sub-state LINSM_NO_COM shall be active when entering the LINSM_INIT state, for all networks when LinSM_Init is called.] ()

[LINSM0216] [The LinSM_Init function shall set the schedule type NULL_SCHEDULE for each configured channel.] ()

To make the LinSM independent from the LinIf module the LinSM module should not call the LinIf module in when the LinSM module is in the init function. The LinSM_Init has therefore additional requirement, see 8.3.1.

7.1.3 LINSM_NO_COM

The no communication state is active after initialization and when the ComM module requests no communication.

[LINSM026] [There shall be a sub-state called LINSM_NO_COM in the state LINSM_INIT.] ()

[LINSM027] [When entering LINSM_NO_COM the LinSM module shall notify (with the exception **[LINSM166]**) ComM of the state change by calling the ComM_BusSM_ModelIndication with the parameter COMM_NO_COMMUNICATION for the specific network.] ()

[LINSM193] [When entering LINSM_NO_COM the LinSM module shall notify (with the exception **[LINSM166]**) BswM of the state change by calling the BswM_LinSM_CurrentState with the parameter LINSM_NO_COM for the specific network.] ()

There is one exception to the above two requirements. The rationale is that the ComM may not be initialized when executing the LinSM_Init function.

[LINSM166] [The LinSM module shall not notify the state change to LINSM_NO_COM when the LinSM is executing the LinSM_Init function, i.e. the LinSM_Init function shall neither call ComM_BusSM_ModelIndication nor BswM_LinSM_CurrentState.] ()

[LINSM028] [When LINSM_NO_COM is active, the LinSM module shall not command the LinIf module to communicate for the selected network, i.e. bus shall be silent.

Note: Upon entering or exiting the LINSM_NO_COM state the LinSM module will not set the hardware interface or μ -controller into a new power mode. This is not in the scope of the LinSM] ()

[LINSM203] [When entering LINSM_NO_COM the transceiver shall be set to STANDBY if LinSMTransceiverPassiveMode is true and SLEEP otherwise by using the LinIf_SetTrcvMode. This requirement is applicable only when LinSMTransceiverPassiveMode is configured for the channel.] ()

[LINSM204] [The LinIf_SetTrcvMode shall not be called from the function LinSM_Init.

Note: There is no need to set the mode in the LinSM init function since the Transceiver will set the mode in its init function. The mode is selected in the Transceiver configuration.] ()

7.1.4 LINSM_FULL_COM

The LINSM_FULL_COM is the only state where communication on the LIN bus is allowed. Each network can be in LINSM_FULL_COM independent of each other. All of the following requirements are applicable for each network.

[LINSM032] [There shall be a sub-state called LINSM_FULL_COM for each network in the state LINSM_INIT.] ()

[LINSM033] [When entering LINSM_FULL_COM the ComM shall be notified of the state change by calling the ComM_BusSM_ModelIndication with the parameter COMM_FULL_COMMUNICATION for the specified network.] ()

[LINSM192] [When entering LINSM_FULL_COM the BswM shall be notified of the state change by calling the BswM_LinSM_CurrentState with the parameter LINSM_FULL_COM for the specified network.] ()

[LINSM205] [When entering LINSM_FULL_COM the transceiver shall be set to active by using the LinIf_SetTrcvMode. This requirement is applicable only when LinSMTransceiverPassiveMode is configured for the channel.] ()

7.1.5 Goto sleep

The goto sleep the LinSM will request the goto-sleep command to be sent on the LIN bus, and after that set no communication mode and notify the BswM and ComM of the new state. The callback will always be made, even if there was a problem.

[LINSM035] [The LinSM module may only call LinIf_GotoSleep API in LinIf when the state LINSM_FULL_COM is active, E_NOT_OK is returned otherwise.] ()

[LINSM0208] [If the state is LINSM_FULL_COM, the LinSMSleepSupport is true and the ComM requests COMM_NO_COMMUNICATION; the LinSM shall call LinIf_GotoSleep to transmit a goto sleep command on the requested network.] ()

[LINSM0209] [In all other cases from **[LINSM0208]** the LinIf_GotoSleep shall not be called.] ()

[LINSM036] [If the ComM module calls LinSM_RequestComMode requesting COMM_NO_COMMUNICATION the LinSM module shall directly call (and not wait for next main function call) the LinIf module function LinIf_GotoSleep on the specified network.] ()

[LINSM177] [If the LinIf_GotoSleep returns E_NOT_OK the LinSM_RequestComMode shall return E_NOT_OK.

If the LinSM module returns LinSM_RequestComMode with E_NOT_OK, the same state shall be set (so that a ComM_BusSM_ModeIndication and BswM_LinSM_CurrentState are called).] ()

[LINSM044] [After the ComM module requests COMM_NO_COMMUNICATION, the LinSM module shall return E_NOT_OK for any requests for the same network until the LinIf notifies or request timer has elapsed.] ()

[LINSM045] [After the LinIf module notifies that the goto-sleep command was sent successfully (success = true) on the bus, the sub-state LINSM_NO_COM shall be set.] ()

[LINSM046] [The sub-state LINSM_FULL_COM for the specific network shall be set, after the LinIf module notifies that the goto-sleep command was NOT sent successfully (success = false) on the bus.] ()

This may happen since the goto-sleep command may fail on the bus.

7.1.6 Changing schedule table

[LINSM079] [If the function LinSM_ScheduleRequest is called, the LinSM module shall forward (and not wait for the next main function call) the request to the LinIf module using the function call LinIf_ScheduleRequest.] ()

[LINSM167] [If LinIf_ScheduleRequest returns with E_OK the LinSM_ScheduleRequest shall return with E_OK.] ()

[LINSM168] [If LinIf_ScheduleRequest returns with E_NOT_OK the LinSM_ScheduleRequest shall return with E_NOT_OK.

The caller of LinSM_ScheduleRequest will assume that LinSM module calls the callback:] ()

[LINSM0213] [If LinIf_ScheduleRequest returns with E_NOT_OK the LinSM module shall call BswM_LinSM_CurrentSchedule with the old schedule table in the next main function call.] ()

[LINSM206] [When the LinSM module gets the confirmation of setting a schedule table from the LinIf module the BswM_LinSM_CurrentSchedule shall be called, if not timer has elapsed.] ()

[LINSM0214] [If timer has elapsed, the LinSM module shall call BswM_LinSM_CurrentSchedule with unchanged schedule table.

Be aware of that the LinIf will switch to a NULL schedule when entering sleep, then it may make a schedule switch callback.] ()

[LINSM207] [If the LinIf confirms a schedule switch without a preceding call to request new schedule table the BswM_LinSM_CurrentSchedule shall be called] ()

7.1.7 Wake up process

A LIN network will be woken up if ComM module requests a wake up through the LinSM_RequestComMode call or if a LIN slave node transmits the wakeup signal on the network. The wakeup by cluster is not handled by the LinSM module, it is handled by the EcuM module and will lead to that the ComM requests the network. In both cases the ComM will request full communication to the LinSM module for the specific network.

In case the LinIf is already awake (because of a slave node waking up the bus) the LinIf will just ignore the wakeup call.

[LINSM047] [If the state is LINSM_NO_COM, the LinSMSleepSupport is true and the ComM requests COMM_FULL_COMMUNICATION; the LinSM shall call LinIf_Wakeup directly (and not wait for next main function call) to transmit a wake up signal on the requested network.] ()

[LINSM178] [In all other cases from **[LINSM047]** the LinSM module shall not call LinIf_Wakeup.] ()

[LINSM049] [When the LinIf notifies that the WakeUp is successfully sent (success = true), the state shall be set to LINSM_FULL_COM.] ()

[LINSM0202] [In all other cases from **[LINSM049]** the state shall be set same state as previous to the request (so that a mode indication callback is made to BswM and ComM).] ()

[LINSM176] [If the LinIf_Wakeup returns E_NOT_OK the LinSM_RequestComMode shall return E_NOT_OK directly with no further action] ()

7.1.8 Timeout of requests

After calling LinIf_GotoSleep, LinIf_Wakeup or LinIf_ScheduleRequest the LinSM module is waiting for the LinIf module to confirm the transmission of the goto sleep command, the wake up on the bus or schedule is changed. There is a possibility that the confirmation is not received, and therefore the LinSM module will wait forever. The only cause for this situation is problem in the software, i.e. no bus event or similar can cause this situation.

[LINSM175] [There shall be request timers for each network. One network shall be independent of another network.] ()

[LINSM162] [The handling (countdown and expiration) of the all request timers used by the LinSM module shall be made done in the LinSM_MainFunction.] ()

[LINSM159] [All request timers shall have a time that is a divisible by the LinSM_MainFunction (i.e. LinSM_MainFunction period * m; m integer >0)] ()

[LINSM100] [Before the LinSM calls the LinIf_GotoSleep, LinIf_Wakeup or LinIf_ScheduleRequest is called, the LinSM module shall start a timer.] ()

[LINSM101] [When a timer expires, i.e. greater than the configuration parameter LinSMConfirmationTimeout, a timeout occurs.] ()

[LINSM154] [If the LinIf module calls the confirmation callback before the timeout occurs, the active timer shall stop, so that the timeout will not occur.] ()

[LINSM102] [if LinSMDevErrorDetect is enabled: When a timeout occurs, the error code LINSM_E_CONFIRMATION_TIMEOUT shall be reported to the DET module.] ()

[LINSM170] [If request timer elapses (i.e. module LinIf is not notifying within the timeout), the LinSM module shall notify ComM module with same state.] ()

[LINSM0215] [If request timer elapses (i.e. module LinIf is not notifying within the timeout), the LinSM module shall notify BswM module with same state.] ()

Making the timeout optional enhances implementation size, if the timeout is not required:

[LINSM103] [If the configuration parameter LinSMConfirmationTimeout is set to zero the timer is not used, and hence a timeout cannot occur. This means that requirements **[LINSM102]**, **[LINSM170]** and **[LINSM0215]** will not happen.] ()

[LINSM172] [If LinIf module calls the confirmation callback after the timer has elapsed, no further notification shall be made to the ComM modules, i.e. the confirmation is ignored.] ()

7.2 Handling multiple networks and drivers

Usually only one LIN driver module (supporting multiple networks) is needed in an ECU to handle all LIN networks. However, rarely, some hardware configurations the ECU contain different LIN hardware (e.g. an advanced LIN controller and a UART). In such case, more than one different LIN drivers are required. This will not affect the LinSM module since the LIN driver only interfaces the LinIf module and not the LIN driver module directly.

The LinSM will only handle networks, and is not concerned to which driver the network maps to, this will be handled by the LinIf.

7.2.1 Multiple networks

Each network has a unique network index (LinSMComMNetworkHandleRef) in the LinSM configuration.

The configuration parameter `LinSMComMNetworkHandleRef` is referencing the ComM module configuration directly. This means that no mapping between networks has to be made in the LinSM module when interfacing to the LinIf module. The network index may be used directly to the LinIf module APIs.

[LINSM164] [The LinSM module shall use the same `NetworkHandle` value, received through an API, when interfacing to the LinIf module (when LIN network is required as a parameter).] ()

7.3 Error classification

This chapter lists and classifies errors that can be detected within this software module. Each error is classified according to relevance (development / production) and related error code. For development errors, a value is defined.

[LINSM051] [Values for production code Event Ids are assigned externally by the configuration of the Dem [8], they are published in the file `Dem_InitErrId.h` and included via `Dem.h`.] (BSW00409)

The `Dem.h` includes the APIs to report errors as well as the required Event Id symbols. The table below defines the name of the Event Id symbols that are provided by XML to the DEM configuration tool. The DEM configuration tool assigns ECU dependent values to the Event Id symbols and publishes the symbols in `Dem_IntErrId.h`.

[LINSM052] [Development error values are of type `uint8`.] ()

The Table 1 shows the available error codes (to DET and DEM) for the LinSM. The list of production errors is complete. The development errors consist of a minimum number of errors detected. It is possible for the implementer to extend the development errors.

[LINSM053] [

| Type or error | Relevance | Related error code | Value [hex] |
|--|-------------|------------------------------|-------------|
| API called without initialization of LinSM | Development | LINSM_E_UNINIT | 0x00 |
| Referenced network does not exist (identification is out of range) | Development | LINSM_E_NONEXISTENT_NETWORK | 0x20 |
| API service called with wrong parameter | Development | LINSM_E_PARAMETER | 0x30 |
| API service called with invalid pointer | Development | LINSM_E_PARAMETER_POINTER | 0x40 |
| Timeout of the callbacks from LinIf | Development | LINSM_E_CONFIRMATION_TIMEOUT | 0x50 |

Table 1 - Error codes for DET and DEM

] (BSW00337, BSW00385, BSW00327, BSW00350)

[LINSM200] [Additional errors that are detected because of specific implementation and/or specific hardware properties shall be added in the module's implementation documentation. The classification and enumeration shall be compatible to the errors listed above] ()

7.4 Error detection

[LINSM054] [The detection of development errors is configurable (*ON / OFF*) at pre-compile time. The switch *LinSMDevErrorDetect* (see chapter 10) shall activate or deactivate the detection of all development errors.] (BSW171, BSW00338)

[LINSM055] [If the *LinSMDevErrorDetect* switch is enabled, API parameter checking is enabled. The detailed description of the detected errors can be found in chapter 7.3 and chapter 8.] (BSW00323)

[LINSM056] [It shall not be possible to switch off detection and reporting of production code errors.] ()

7.5 Error notification

[LINSM057] [Detected development errors will be reported to the *Det_ReportError* API of the Development Error Tracer (DET) [5] module if the pre-processor switch *LinSMDevErrorDetect* is set (see paragraph 10).] (BSW167)

[LINSM058] [Production errors shall be reported to Diagnostic Event Manager (DEM) [8] module.] (BSW00339)

7.6 Debugging

To allow standardized debugging of the LinSM module, following requirements applies:

[LINSM184] [Each variable that shall be accessible by AUTOSAR Debugging, shall be defined as global variable.] (BSW00442)

[LINSM185] [All type definitions of variables which shall be debugged, shall be accessible by the header file *LinSM.h*.] (BSW00442)

[LINSM186] [The declaration of variables in the header file shall be such, that it is

possible to calculate the size of the variables by C-"sizeof".] (BSW00442)

[LINSM187] [Variables available for debugging shall be described in the respective Basic Software Module Description] (BSW00442)

[LINSM188] [The state of the state-machine (LINSM_UNINIT and LINSM_INIT) shall be accessible for debugging.] (BSW00442)

[LINSM189] [The state of the sub-state-machine (LINSM_NO_COM and LINSM_FULL_COM) shall be accessible for debugging.] (BSW00442)

8 API specification

8.1 Imported types

8.1.1 Standard types

In this chapter all types included from the following files are listed. The standard AUTOSAR types are defined in the AUTOSAR Specification of Standard Types document [4].

Following types are used by the LinSM module:

[LINSM0219] [

| Module | Imported Type |
|----------------|----------------------|
| ComM | ComM_ModeType |
| ComStack_Types | NetworkHandleType |
| LinIf | LinIf_SchHandleType |
| LinTrcv | LinTrcv_TrvcModeType |
| Std_Types | Std_ReturnType |
| | Std_VersionInfoType |

] ()

8.2 Type definitions

Following types are defined by the LinSM module:

8.2.1 LinSM_ModeType

[LINSM0220] [

| | | | |
|---------------------|--|---|--------------------|
| Name: | LinSM_ModeType | | |
| Type: | uint8 | | |
| Range: | LINSM_FULL_COM | 1 | Full communication |
| | LINSM_NO_COM | 2 | No communication |
| Description: | Type used to report the current mode to the BswM | | |

] ()

8.2.2 LinSM_ConfigType

[LINSM0221] [

| | | | |
|---------------|-------------------------|----|--|
| Name: | LinSM_ConfigType | | |
| Type: | Structure | | |
| Range: | implementation specific | -- | |

| | |
|---------------------|--|
| Description: | Data structure type for the post-build configuration parameters. |
|---------------------|--|

↓()

8.3 LinSM API

This is a list of API calls provided for upper layer modules.

8.3.1 LinSM_Init

[LINSM155] [

| | |
|----------------------------|---|
| Service name: | LinSM_Init |
| Syntax: | void const LinSM_ConfigType* LinSM_Init(ConfigPtr) |
| Service ID[hex]: | 0x01 |
| Sync/Async: | Synchronous |
| Reentrancy: | Non reentrant |
| Parameters (in): | ConfigPtr Pointer to the LinSM post-build configuration data. |
| Parameters (inout): | None |
| Parameters (out): | None |
| Return value: | None |
| Description: | This function initializes the LinSM. |

| (BSW101, BSW00358, BSW00414)

[LINSM151] [No other LinSM API or other module's (e.g. Linlf) API shall be called from the LinSM_Init function. Other modules may not be initialized.] ()

[LINSM0217] [In case of a post-build variant and if LinSMDevErrorDetect is enabled: If the ConfigPtr pointer parameter is invalid (e.g. NULL), the error code LINSM_E_PARAMETER_POINTER shall be reported to the DET module.] ()

[LINSM0218] [In case of a pre-compile or link-time variant and if LinSMDevErrorDetect is enabled: If the ConfigPtr pointer parameter is not NULL, the error-code LINSM_E_PARAMETER_POINTER shall be reported to the DET module.] ()

8.3.2 LinSM ScheduleRequest

[LINSM113] [

| | |
|----------------------|--|
| Service name: | LinSM_ScheduleRequest |
| Syntax: | <pre>Std_ReturnType LinSM_ScheduleRequest(NetworkHandleType network, LinIf_SchHandleType schedule)</pre> |

| | | |
|----------------------------|---|--|
| Service ID[hex]: | 0x10 | |
| Sync/Async: | Asynchronous | |
| Reentrancy: | Reentrant | |
| Parameters (in): | network | Identification of the LIN channel |
| | schedule | Pointer to the new Schedule table |
| Parameters (inout): | None | |
| Parameters (out): | None | |
| Return value: | Std_ReturnType | E_OK - Schedule table request has been accepted. E_NOT_OK - Schedule table switch request has not been accepted due to one of the following reasons: * LinSM has not been initialized referenced channel does not exist (identification is out of range) * Referenced schedule table does not exist (identification is out of range) * Sub-state is not LINSM_FULL_COM |
| Description: | The upper layer requests a schedule table to be changed on one LIN network. | |

] (BSW00369)

[LINSM114] [If LinSMDevErrorDetect is enabled: If the network parameter has an invalid value then the error-code LINSM_E_NONEXISTENT_NETWORK shall be reported to the DET module and E_NOT_OK shall be returned.] ()

[LINSM115] [If LinSMDevErrorDetect is enabled: If the schedule parameter has an invalid value, then the error-code LINSM_E_PARAMETER shall be reported to the DET module and E_NOT_OK shall be returned.] ()

[LINSM116] [If LinSMDevErrorDetect is enabled: If the state LINSM_UNINIT is active then the error-code LINSM_E_UNINIT shall be reported to the DET module and E_NOT_OK shall be returned.] (BSW00406)

[LINSM163] [If the function LinSM_ScheduleRequest is called and another request is in process on the same network, the LinSM_ScheduleRequest shall return directly with E_NOT_OK.] ()

[LINSM0211] [If the function LinSM_ScheduleRequest is called and the state is not LINSM_FULL_COM, the LinSM_ScheduleRequest shall return directly with E_NOT_OK.] ()

8.3.3 LinSM_GetVersionInfo

[LINSM117] [

| | | |
|-------------------------|----------------------|--|
| Service name: | LinSM_GetVersionInfo | |
| Syntax: | void | LinSM_GetVersionInfo(Std_VersionInfoType* versioninfo) |
| Service ID[hex]: | 0x02 | |
| Sync/Async: | Synchronous | |

| | |
|----------------------------|---|
| Reentrancy: | Reentrant |
| Parameters (in): | None |
| Parameters (inout): | None |
| Parameters (out): | versioninfo Pointer to where to store the version information of this module. |
| Return value: | None |
| Description: | -- |

└ (BSW00407)

[LINSM118] [The LinSM_GetVersionInfo function shall return the version information of this module. The version information includes:

- Module Id [2]
- Vendor Id
- Vendor specific version numbers (BSW00407 [3])

└ ()

[LINSM165] [It shall be possible to call this function at any time (e.g. before the init function is called).

Hint: If source code for caller and callee of this function is available, the function LinSM_GetVersionInfo should be realized as a macro, defined in the modules header file. └ ()

[LINSM119] [If LinSMDevErrorDetect is enabled: If the versioninfo pointer parameter is invalid (e.g. NULL), the error-code LINSM_E_PARAMETER_POINTER shall be reported to the DET module and E_NOT_OK shall be returned. └ ()

[LINSM121] [This function shall be pre compile time configurable On/Off by the configuration parameter: LinSMVersionInfoApi

Note that this API may be called even though the LinSM module is not initialized. └ (BSW00411)

8.3.4 LinSM_GetCurrentComMode

[LINSM122] [

| | | | |
|---------------------|---|--|--|
| Service name: | LinSM_GetCurrentComMode | | |
| Syntax: | Std_ReturnType <div>NetworkHandleType ComM_ModeType*</div> <div>LinSM_GetCurrentComMode(network, mode)</div> | | |
| Service ID[hex]: | 0x11 | | |
| Sync/Async: | Synchronous | | |
| Reentrancy: | Reentrant | | |
| Parameters (in): | network | Identification of the LIN channel | |
| Parameters (inout): | None | | |
| Parameters (out): | mode | Returns the active mode, see ComM_ModeType for descriptions of the modes | |

| | | | |
|----------------------|---|---|----|
| Return value: | Std_ReturnType | E_OK - E_NOT_OK - Not possible to perform the request, e.g. not initialized. | Ok |
| Description: | Function to query the current communication mode. | | |

] (BSW00369)

[LINSM123] [If LinSMDevErrorDetect is enabled: If the network parameter has an invalid value, then the error-code LINSM_E_NONEXISTENT_NETWORK shall be reported to the DET module and E_NOT_OK shall be returned.] ()

[LINSM124] [If LinSMDevErrorDetect is enabled: If the mode pointer parameter is invalid (e.g. NULL), then the error-code LINSM_E_PARAMETER_POINTER shall be reported to the DET module and E_NOT_OK shall be returned.] ()

[LINSM125] [If LinSMDevErrorDetect is enabled: If the state LINSM_UNINIT is active, then the error-code LINSM_E_UNINIT shall be reported to the DET module and E_NOT_OK shall be returned] (BSW00406)

[LINSM180] [If active state is LINSM_NO_COM the state COMM_NO_COMMUNICATION shall be returned.] ()

[LINSM181] [If active state is LINSM_FULL_COM the state COMM_FULL_COMMUNICATION shall be returned.] ()

[LINSM182] [If active state is LINSM_UNINIT the state COMM_NO_COMMUNICATION shall be returned. This is also captured above when the DET is enabled. This is to be defensive.] ()

Note that COMM_SILENT_COMMUNICATION is not used by the LinSM module.

8.3.5 LinSM_RequestComMode

[LINSM126] [

| | | | |
|----------------------|----------------------|--|--|
| Service name: | LinSM_RequestComMode | | |
| Syntax: | Std_ReturnType | | |

| | | |
|---------------------|--|---|
| | | E_NOT_OK - Not possible to perform the request, e.g. not initialized. |
| Description: | Requesting of a communication mode. | |
| | The mode switch will not be made instant. The LinSM will notify the caller when mode transition is made. | |

] (BSW00369)

[LINSM127] [If LinSMDevErrorDetect is enabled: If the network parameter has an invalid value, then the error-code LINSM_E_NONEXISTENT_NETWORK shall be reported to the DET module and E_NOT_OK shall be returned.] ()

[LINSM191] [If LinSMDevErrorDetect is enabled: If the mode parameter has an invalid value, then the error-code LINSM_E_PARAMETER shall be reported to the DET module and E_NOT_OK shall be returned.] ()

[LINSM128] [If LinSMDevErrorDetect is enabled: If the state LINSM_UNINIT is active, then the error-code LINSM_E_UNINIT shall be reported to the DET module and E_NOT_OK shall be returned.] (BSW00406)

[LINSM174] [If the function LinSM_RequestComMode is called and another request is in process on the same network, the LinSM_RequestComMode shall return directly with E_NOT_OK.] ()

[LINSM183] [If COMM_SILENT_COMMUNICATION is requested the function shall return E_NOT_OK directly without action] ()

[LINSM0210] [If the requested mode is the same as the current active mode, the function shall return E_NOT_OK directly without action.] ()

[LINSM0223] [LinSM_RequestComMode shall store the requested mode, if the return value is E_OK.

The next activation of the LinSM_MainFunction will then process this request when processing the state machine.

Note, that the state machine definition in section 7.1 refers to this stored request as reqComMode.] ()

8.4 Scheduled Functions

This chapter lists the functions that are called with a fixed period.

8.4.1 LinSM_MainFunction

This function is directly called by the Basic Software Scheduler module. The following function has no return value, no parameter and is non-reentrant.

There is no dependency to other main functions. This main function may be executed without considering other main functions. But scheduling the different main functions intelligent will minimize execution time and jitter.

[LINSM156] [

| | |
|-------------------------|--|
| Service name: | LinSM_MainFunction |
| Syntax: | void LinSM_MainFunction(void) |
| Service ID[hex]: | 0x30 |
| Timing: | FIXED_CYCLIC |
| Description: | Periodic function that runs the timers of different request timeouts |

Design hint: The function LinSM_MainFunction may be interrupted by other functions. It should be assured that the timers operated by this function are protected so that they behave correctly (e.g. by using critical sections if necessary).] (BSW00373, BSW00376)

[LINSM157] [The LinSM_MainFunction shall handle the timers that are attached to the functions LinIf_GotoSleep, LinIf_Wakeup or LinIf_ScheduleRequest (see paragraph 7.1.8)] ()

[LINSM179] [If LinSMDevErrorDetect is enabled: The function LinSM_MainFunction shall raise the error LINSM_E_UNINIT when the state LINSM_UNINIT is active.] (BSW00406)

8.5 LinSM callbacks

The function prototypes of the callback functions will be provided in the file LinSM_Cbk.h

8.5.1 LinSM_ScheduleRequestConfirmation

[LINSM129] [

| | |
|-------------------------|---|
| Service name: | LinSM_ScheduleRequestConfirmation |
| Syntax: | void LinSM_ScheduleRequestConfirmation(NetworkHandleType network, LinIf_SchHandleType schedule) |
| Service ID[hex]: | 0x20 |

| | | |
|----------------------------|---|--|
| Sync/Async: | Synchronous | |
| Reentrancy: | Reentrant | |
| Parameters (in): | network | Identification of the LIN channel |
| | schedule | Pointer to the new active Schedule table |
| Parameters (inout): | None | |
| Parameters (out): | None | |
| Return value: | None | |
| Description: | The LinIf module will call this callback when the new requested schedule table is active. | |

] ()

[LINSM130] [If LinSMDevErrorDetect is enabled: If the network parameter has an invalid value, then the error-code LINSM_E_NONEXISTENT_NETWORK shall be reported to the DET module.] ()

[LINSM131] [If LinSMDevErrorDetect is enabled: If the state LINSM_UNINIT is active, then the error-code LINSM_E_UNINIT shall be reported to DET module and E_NOT_OK shall be returned.] (BSW00406)

8.5.2 LinSM_GotoSleepConfirmation

[LINSM135] [

| | | |
|----------------------------|--|---|
| Service name: | LinSM_GotoSleepConfirmation | |
| Syntax: | <pre>void LinSM_GotoSleepConfirmation(NetworkHandleType network, boolean success)</pre> | |
| Service ID[hex]: | 0x22 | |
| Sync/Async: | Synchronous | |
| Reentrancy: | Reentrant | |
| Parameters (in): | network | Identification of the LIN channel |
| | success | True if goto sleep was successfully sent, false otherwise |
| Parameters (inout): | None | |
| Parameters (out): | None | |
| Return value: | None | |
| Description: | The LinIf will call this callback when the go to sleep command is sent successfully or not sent successfully on the network. | |

] ()

[LINSM136] [If LinSMDevErrorDetect is enabled: If the network parameter has an invalid value, then the error-code LINSM_E_NONEXISTENT_NETWORK shall be reported to the DET module.] ()

[LINSM137] [If LinSMDevErrorDetect is enabled: If the state LINSM_UNINIT is active, then the error-code LINSM_E_UNINIT shall be reported to the DET module.] (BSW00406)

[LINSM199] [The LinSM_GotoSleepConfirmation shall be configurable.]
(BSW00387, BSW00440)

8.5.3 LinSM_WakeupConfirmation

[LINSM132] [

| | | | |
|----------------------------|---|---|--|
| Service name: | LinSM_WakeupConfirmation | | |
| Syntax: | void <div>LinSM_WakeupConfirmation(NetworkHandleType network, boolean success)</div> | | |
| Service ID[hex]: | 0x21 | | |
| Sync/Async: | Synchronous | | |
| Reentrancy: | Reentrant | | |
| Parameters (in): | network | Identification of the LIN channel | |
| | success | True if wakeup was successfully sent, false otherwise | |
| Parameters (inout): | None | | |
| Parameters (out): | None | | |
| Return value: | None | | |
| Description: | The LinIf will call this callback when the wake up signal command is sent not successfully/successfully on the network. | | |

] ()

[LINSM133] [If LinSMDevErrorDetect is enabled: If the network parameter has an invalid value, then the error-code LINSM_E_NONEXISTENT_NETWORK shall be reported to the DET module.] ()

[LINSM134] [If LinSMDevErrorDetect is enabled: If the state LINSM_UNINIT is active, then the error-code LINSM_E_UNINIT shall be reported to the DET module.]
(BSW00406)

[LINSM198] [The LinSM_WakeupConfirmation shall be configurable.] (BSW00387, BSW00440)

8.6 Mandatory Interfaces

This chapter defines all interfaces that are required to fulfill the core functionality.

| API function | Description |
|----------------------------|---|
| BswM_LinSM_CurrentSchedule | Function called by LinSM to indicate the currently active schedule table for a specific LIN channel. |
| BswM_LinSM_CurrentState | Function called by LinSM to indicate its current state. |
| ComM_BusSM_ModeIndication | Indication of the actual bus mode by the corresponding Bus State Manager. ComM shall propagate the indicated state to the users with means of the RTE and BswM. |
| LinIf_ScheduleRequest | Requests a schedule table to be executed. |

8.7 Optional Interfaces

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

[LINSM138] [

| <i>API function</i> | <i>Description</i> |
|---------------------|---|
| Det_ReportError | Service to report development errors. |
| LinIf_GotoSleep | Initiates a transition into the Sleep Mode on the selected channel. |
| LinIf_SetTrcvMode | Set the given LIN transceiver to the given mode. |
| LinIf_Wakeup | Initiates the wake up process. |

] ()

8.8 Configurable Interfaces

No configurable interfaces.

9 Sequence diagrams

This chapter will show use-cases for LIN communication and API usage. As the communication is in real-time it is not easy to show the real-time behavior in the UML dynamic diagrams. It is advisable to read the corresponding descriptive text to each UML diagram.

To show the behavior of the modules in the different use-cases, there are local function calls made to show what is done and when to get information. It is not mandatory to use these local functions; they are here just to make the use-cases more understandable.

Note that all parameters and return types are left out to make the diagrams easier to read and understand. If needed for clarification the parameter value or return value are shown.

9.1 Goto-sleep process

This use-case in shows the transition into the LINSM_GOTOSLEEP state when the goto-sleep command has been sent successfully on the bus.

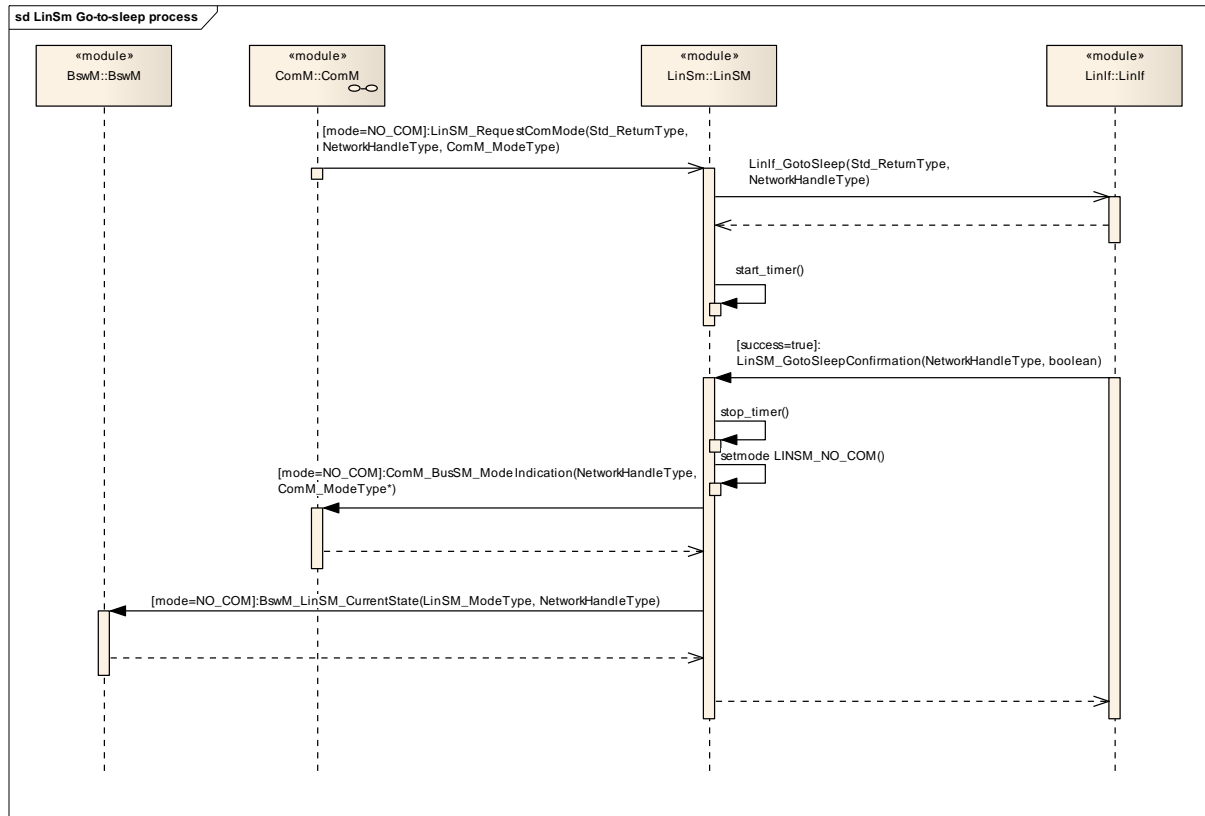


Figure 5 - Goto-sleep-command process

9.2 Internal wake up

The Figure 6 shows the internal wakeup. A wakeup is requested from module above the LinSM.

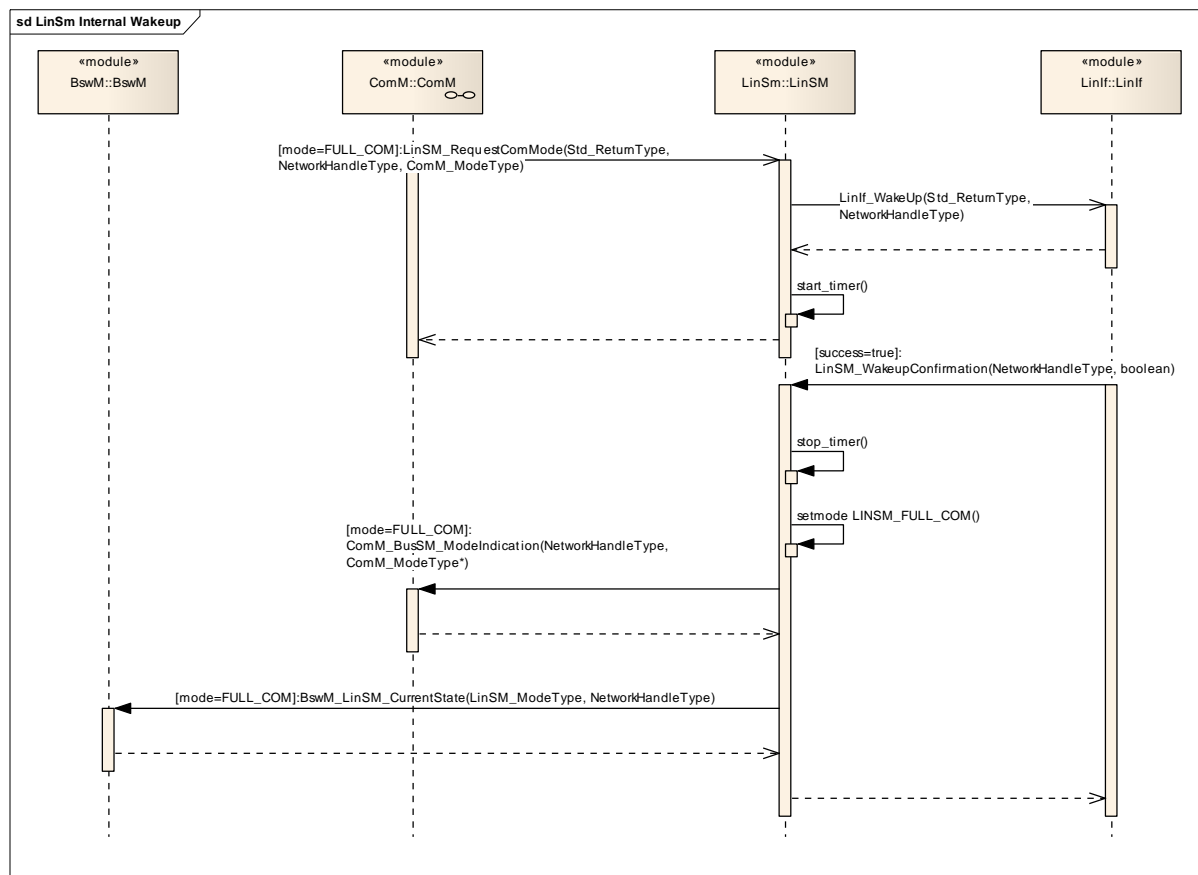


Figure 6 - Internal wake up

9.3 Schedule switch

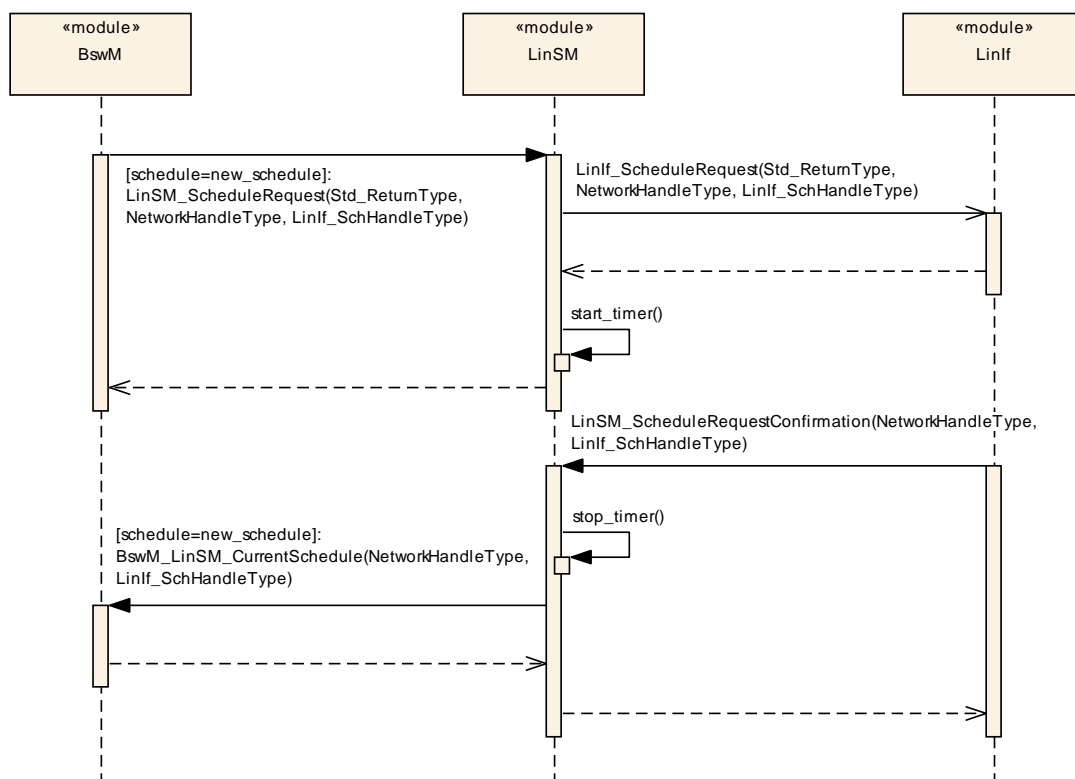


Figure 7 shows the use-cases of switching the schedule table when the LinSM accepts the request.

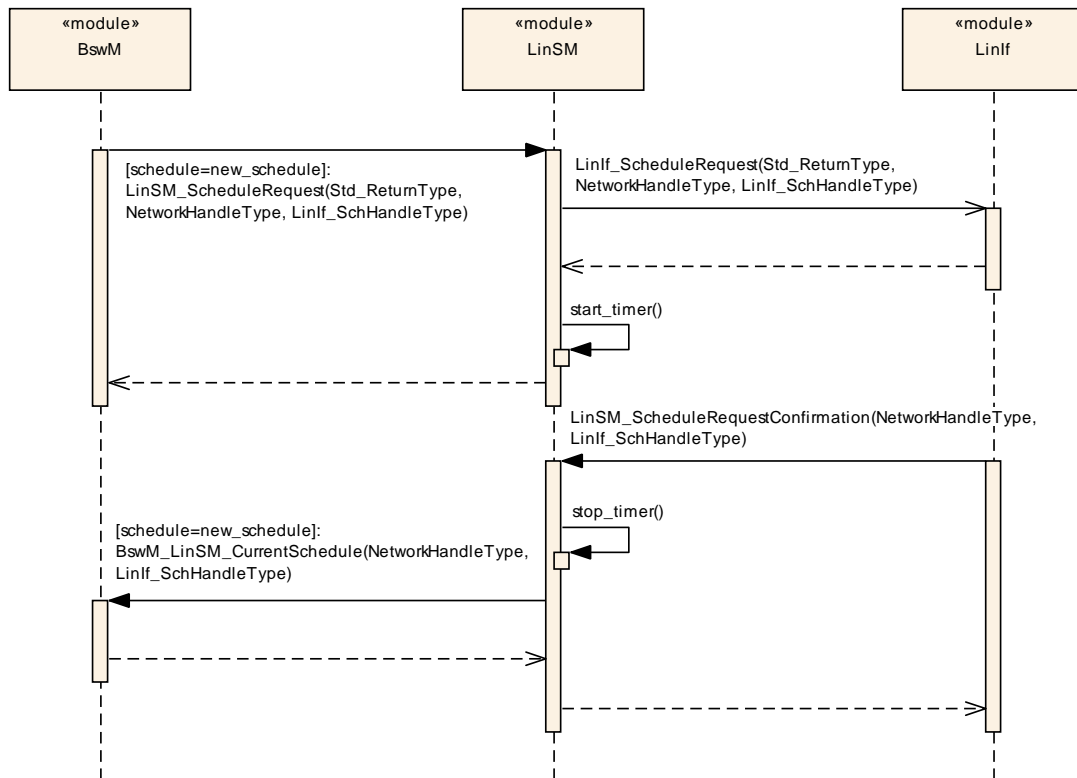


Figure 7: Schedule Table switch

10 Configuration specification

In general, this chapter defines configuration parameters and their clustering into containers.

The chapter 10.3 specifies the structure (containers) and the parameters of the LinSM module.

10.1 How to read this chapter

In addition to this section, it is highly recommended to read the documents:

AUTOSAR Layered Software Architecture [2]

AUTOSAR ECU Configuration Specification [9] - This document describes the AUTOSAR configuration methodology and the AUTOSAR configuration meta-model in detail.

The following is only a short survey of the topic and it will not replace the ECU Configuration Specification document.

10.1.1 Configuration and configuration parameters

Configuration parameters define the variability of the generic part(s) of an implementation of a module. This means that only generic or configurable module implementation can be adapted to the environment (software/hardware) in use during system and/or ECU configuration.

The configuration of parameters can be achieved at different times during the software process: pre compile time, before link time or after build time. In the following, the term “configuration class” (of a parameter) shall be used in order to refer to a specific configuration point in time.

10.1.2 Containers

Containers structure the set of configuration parameters. This means:

All configuration parameters are kept in containers.

(sub-) containers can reference (sub-) containers. It is possible to assign a multiplicity to these references. The multiplicity then defines the possible number of instances of the contained parameters.

10.1.3 Specification template for configuration parameters

The following tables consist of three sections:

- the general section
- the configuration parameter section
- the section of included/referenced containers
- Pre-compile time - specifies whether the configuration parameter shall be of configuration class *Pre-compile time* or not

| Label | Description |
|-------|---|
| x | The configuration parameter shall be of configuration class <i>Pre-compile time</i> . |
| -- | The configuration parameter shall never be of configuration class <i>Pre-compile time</i> . |

Link time - specifies whether the configuration parameter shall be of configuration class *Link time* or not

| Label | Description |
|-------|--|
| x | The configuration parameter shall be of configuration class <i>Link time</i> . |
| -- | The configuration parameter shall never be of configuration class <i>Link time</i> . |

Post Build - specifies whether the configuration parameter shall be of configuration class *Post Build* or not

| Label | Description |
|-------|--|
| x | The configuration parameter shall be of configuration class <i>Post Build</i> and no specific implementation is required. |
| L | <i>Loadable</i> - the configuration parameter shall be of configuration class <i>Post Build</i> and only one configuration parameter set resides in the ECU. |
| M | <i>Multiple</i> - the configuration parameter shall be of configuration class <i>Post Build</i> and is selected out of a set of multiple parameters by passing a dedicated pointer to the init function of the module. |
| -- | The configuration parameter shall never be of configuration class <i>Post Build</i> . |

10.2 Containers and configuration parameters

The following chapters summarize all configuration parameters. The detailed meanings of the parameters describe Chapters 7 and Chapter 8.

Example: The LinSM_Configuration is placed in a specific Flash sector. This flash sector may be reflashed after the ECU is placed in the vehicle.

10.2.1 Configuration Tool

A configuration tool will create a configuration structure that is understood by the LinSM.

[LINSM073] [The LinSM module shall not make any consistency check of the configuration in run-time in production software. It may, however, be done if the Development Error Detection is enabled.] (BSW167)

10.2.2 Variants

Following configuration variants are defined for the LinSM module.

10.2.2.1 VARIANT-PRE-COMPILE

In the the variant VARIANT-PRE-COMPILE all parameters below that are marked as Pre-compile configurable shall be configurable in a pre-compile manner, for example as #defines.

The module is most likely delivered as source code.

10.2.2.2 VARIANT-LINK-TIME

The variant VARIANT-LINK-TIME shall include all configuration options of the variant VARIANT-PRE-COMPILE. Additionally, all parameters that are marked as link-time configurable shall be configurable at link time. For example by linking a special configured parameter object file.

The module is most likely delivered as object code.

10.2.2.3 VARIANT-POST_BUILD

The variant VARIANT-POST-BUILD shall include all configuration options of the variant VARIANT-LINK-TIME. Additionally, all parameters that are marked as post-build configurable shall be configurable post build for example by flashing configuration data.

The module is most likely delivered as object code.

10.3 LinSM_Configuration

The paragraph defines the LinSM configuration.

10.3.1 LinSM

| | |
|---------------------------|--|
| Module Name | <i>LinSM</i> |
| Module Description | Configuration of the Lin State Manager module. |

| Included Containers | | |
|----------------------------|---------------------|--|
| Container Name | Multiplicity | Scope / Dependency |
| LinSMConfigSet | 1 | This container describes one of multiple configuration sets of LinSm. This is a MultipleConfigurationContainer, i.e. this container and its sub-containers exist once per configuration set. |
| LinSMGeneral | 1 | This container contains general parameters of LIN State Manager module. |

10.3.2 LinSMConfigSet

| | |
|-----------------------|--|
| SWS Item | LINSM207_Conf : |
| Container Name | LinSMConfigSet{LinSM_ConfigSet} [Multi Config Container] |

| | |
|---------------------------------|--|
| Description | This container describes one of multiple configuration sets of LinSm. This is a MultipleConfigurationContainer, i.e. this container and its sub-containers exist once per configuration set. |
| Configuration Parameters | |

| Included Containers | | |
|----------------------------|---------------------|---|
| Container Name | Multiplicity | Scope / Dependency |
| LinSMChannel | 1..* | Describes each LIN channel the LinSM is connected to. |

10.3.3 LinSMChannel

| | |
|---------------------------------|---|
| SWS Item | LINSM142_Conf : |
| Container Name | LinSMChannel{LinSM_ChannelReference} |
| Description | Describes each LIN channel the LinSM is connected to. |
| Configuration Parameters | |

| | | | |
|---------------------------|---|----|---------------------------------------|
| SWS Item | LINSM144_Conf : | | |
| Name | LinSMConfirmationTimeout {LINSM_CONFIRMATION_TIMEOUT} | | |
| Description | Timeout in seconds for the goto sleep and wakeup calls to Linlf. The timeout must be longer than a goto-sleep command on the bus (i.e. it is bit rate dependent). | | |
| Multiplicity | 1 | | |
| Type | EcucFloatParamDef | | |
| Range | 0 .. INF | | |
| Default value | -- | | |
| ConfigurationClass | Pre-compile time | X | VARIANT-PRE-COMPILE |
| | Link time | X | VARIANT-LINK-TIME, VARIANT-POST-BUILD |
| | Post-build time | -- | |
| Scope / Dependency | scope: Module | | |

| | | | |
|---------------------------|--|----|---------------------------------------|
| SWS Item | LINSM143_Conf : | | |
| Name | LinSMSleepSupport {LINSM_SLEEP_SUPPORT} | | |
| Description | Some LIN clusters does not need sleep, they will just shut off. This parameter will affect the behavior to achieve the no communication state. | | |
| Multiplicity | 1 | | |
| Type | EcucBooleanParamDef | | |
| Default value | -- | | |
| ConfigurationClass | Pre-compile time | X | VARIANT-PRE-COMPILE |
| | Link time | X | VARIANT-LINK-TIME, VARIANT-POST-BUILD |
| | Post-build time | -- | |
| Scope / Dependency | scope: Module | | |

| | | | |
|---------------------------|--|---|----------------------------------|
| SWS Item | LINSM202_Conf : | | |
| Name | LinSMTransceiverPassiveMode {LINSM_TRANSCEIVER_PASSIVE_MODE} | | |
| Description | Selects STANDBY (true) or SLEEP (false) transceiver mode when entering LINSM_NO_COM. | | |
| Multiplicity | 0..1 | | |
| Type | EcucBooleanParamDef | | |
| Default value | -- | | |
| ConfigurationClass | Pre-compile time | X | VARIANT-PRE-COMPILE |
| | Link time | X | VARIANT-LINK-TIME, VARIANT-POST- |

| | | | |
|---------------------------|------------------------|----|-------|
| | | | BUILD |
| | Post-build time | -- | |
| Scope / Dependency | scope: Module | | |

| | | | |
|---------------------------|--|----|---------------------------------------|
| SWS Item | LINSM145_Conf : | | |
| Name | LinSMComMNetworkHandleRef | | |
| Description | Unique handle to identify one certain LIN network. Reference to one of the network handles configured in the ComM. | | |
| Multiplicity | 1 | | |
| Type | Reference to [ComMChannel] | | |
| ConfigurationClass | Pre-compile time | X | VARIANT-PRE-COMPILE |
| | Link time | X | VARIANT-LINK-TIME, VARIANT-POST-BUILD |
| | Post-build time | -- | |
| Scope / Dependency | scope: Module | | |

| | | |
|----------------------------|---------------------|---|
| Included Containers | | |
| Container Name | Multiplicity | Scope / Dependency |
| LinSMSchedule | 1..* | The schedule references to a schedule that is located in the LinIf configuration. Moreover, the PDU groups are located in the COM configuration. Note that there are two references to PDU groups. The simple reason is that a PDU group is only allowed to contain one direction (TX or RX). |

10.3.4 LinSMGeneral

| | |
|---------------------------------|---|
| SWS Item | LINSM139_Conf : |
| Container Name | LinSMGeneral |
| Description | This container contains general parameters of LIN State Manager module. |
| Configuration Parameters | |

| | | | |
|---------------------------|--|----|--------------|
| SWS Item | LINSM206_Conf : | | |
| Name | LinSMDevErrorDetect {LINSM_DEV_ERROR_DETECT} | | |
| Description | Switches the Development Error Detection and Notification ON or OFF. | | |
| Multiplicity | 1 | | |
| Type | EcucBooleanParamDef | | |
| Default value | -- | | |
| ConfigurationClass | Pre-compile time | X | All Variants |
| | Link time | -- | |
| | Post-build time | -- | |
| Scope / Dependency | scope: Module | | |

| | | | |
|---------------------------|---|----|--------------|
| SWS Item | LINSM141_Conf : | | |
| Name | LinSMMainProcessingPeriod | | |
| Description | Fixed period that the MainFunction shall be called. | | |
| Multiplicity | 1 | | |
| Type | EcucFloatParamDef | | |
| Range | 0 .. INF | | |
| Default value | -- | | |
| ConfigurationClass | Pre-compile time | X | All Variants |
| | Link time | -- | |
| | Post-build time | -- | |
| Scope / Dependency | scope: Module | | |

| | | | |
|---------------------------|---|----|--------------|
| SWS Item | LINSM140_Conf : | | |
| Name | LinSMVersionInfoApi {LINSM_VERSION_INFO_API} | | |
| Description | Switches the LinSM_GetVersionInfo function ON or OFF. | | |
| Multiplicity | 1 | | |
| Type | EcucBooleanParamDef | | |
| Default value | -- | | |
| ConfigurationClass | Pre-compile time | X | All Variants |
| | Link time | -- | |
| | Post-build time | -- | |
| Scope / Dependency | scope: Module | | |

No Included Containers

10.3.5 LinSMSchedule

| | |
|---------------------------------|---|
| SWS Item | LINSM146_Conf : |
| Container Name | LinSMSchedule{LinSM_Schedule} |
| Description | The schedule references to a schedule that is located in the LinIf configuration. Moreover, the PDU groups are located in the COM configuration. Note that there are two references to PDU groups. The simple reason is that a PDU group is only allowed to contain one direction (TX or RX). |
| Configuration Parameters | |

| | | | |
|---------------------------|--|---|---------------------|
| SWS Item | LINSM001_Conf : | | |
| Name | LinSMScheduleIndex | | |
| Description | This index parameter can be used by the BswM as a SymbolicNameReference target. The LinSM just forwards the request from the BswM to LinIf. Note that the value of the LinSMScheduleIndex shall be the same as the value from the LinIf. | | |
| Multiplicity | 1 | | |
| Type | EcucIntegerParamDef (Symbolic Name generated for this parameter) | | |
| Range | 0 .. 255 | | |
| Default value | -- | | |
| ConfigurationClass | Pre-compile time | X | VARIANT-PRE-COMPILE |
| | Link time | X | VARIANT-LINK-TIME |
| | Post-build time | X | VARIANT-POST-BUILD |
| Scope / Dependency | scope: Module | | |

| | | | |
|---------------------------|--|---|---------------------|
| SWS Item | LINSM149_Conf : | | |
| Name | LinSMScheduleIndexRef {LINSM_SCHEDULE_INDEX_REF} | | |
| Description | Reference to a schedule table in the LinIf configuration | | |
| Multiplicity | 1 | | |
| Type | Reference to [LinIfScheduleTable] | | |
| ConfigurationClass | Pre-compile time | X | VARIANT-PRE-COMPILE |
| | Link time | X | VARIANT-LINK-TIME |
| | Post-build time | X | VARIANT-POST-BUILD |
| Scope / Dependency | scope: Module | | |

No Included Containers

10.4 Published Information

[LINSM210] [The standardized common published parameters as required by BSW00402 in the General Requirements on Basic Software Modules [3] shall be published within the header file of this module and need to be provided in the BSW Module Description. The according module abbreviation can be found in the List of Basic Software Modules [1].] (BSW00402, BSW00374, BSW00379, BSW00318)

Additional module-specific published parameters are listed below if applicable.

11 Changes between Release 3.0/3.1 and 4.0

11.1 Deleted SWS Items

| SWS Item | Rationale |
|-----------------|--|
| LINSM089 | PDU group switching is now made in BswM |
| LINSM080 | PDU group switching is now made in BswM |
| LINSM081 | PDU group switching is now made in BswM |
| LINSM017 | Requirement changed to information |
| LINSM018 | Requirement changed to information |
| LINSM083 | The parameter checking have requirement already for each API |
| LINSM088 | LinSM is no longer using COM |
| LINSM010 | internal version check removed |
| LINSM194 | internal version check removed |
| LINSM009 | internal version check removed |
| LINSM042 | Post-build configuration support |

11.2 Changed SWS Items

| SWS Item | Rationale |
|-----------------|--|
| LINSM078 | Post-build parameters removed |
| LINSM010 | Updated to latest SRS general requirement BSW004 |
| LINSM018 | Included all relevant LIN spec versions |
| LINSM020 | New state machine |
| LINSM043 | Requirement moved |
| LINSM036 | New state machine |
| LINSM044 | New state machine |
| LINSM045 | New state machine |
| LINSM046 | New state machine |
| LINSM039 | New state machine |
| LINSM040 | New state machine |
| LINSM041 | New state machine |
| LINSM042 | New state machine |
| LINSM151 | Requirement moved |
| LNSM155 | Post-build configuration support |

11.3 Added SWS Items

| SWS Item | Rationale |
|-----------------|------------------|
| LINSM161 | - |
| LINSM160 | - |
| LINSM163 | - |
| LINSM159 | - |
| LINSM162 | - |
| LINSM164 | - |
| LINSM165 | - |
| LINSM156 | - |
| LINSM157 | - |
| LINSM158 | - |
| LINSM190 | - |
| LINSM191 | - |
| LINSM192 | - |
| LINSM193 | - |
| LINSM194 | - |

| | |
|--|--|
| LINSM195 | - |
| LINSM196 | - |
| LINSM197 | - |
| LINSM198 | - |
| LINSM199 | - |
| LINSM200 | - |
| LINSM201 | - |
| LINSM202 | New configuration item |
| LINSM203 | - |
| LINSM204 | - |
| LINSM205 | - |
| LINSM210 | Rework of Published Information |
| LINSM208 | version check of included header files added |
| LINSM209 | version check of included header files added |
| LINSM216 | set NULL_SCHEDULE type in LinSM_Init |
| LINSM217, LINSM218, LINSM219, LINSM220, LINSM221 | Post-build configuration support |
| | |

12 Not applicable requirements

[LINSM211] 「 These requirements are not applicable to this specification.]
(BSW00404, BSW00405, BSW170, BSW00399, BSW00400, BSW00375, BSW00416, BSW00437, BSW168, BSW00425, BSW00432, BSW00433, BSW00422, BSW00417, BSW161, BSW162, BSW005, BSW00415, BSW00343, BSW00436, BSW00439, BSW00359, BSW00360, BSW00331, BSW00443, BSW00444, BSW00445, BSW00446, BSW010, BSW00333, BSW00321, BSW00341, BSW00334, BSW01590, BSW01560, BSW01577, BSW00438)