Document Title	Specification of LIN Transceiver Driver
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	257
Document Classification	Standard

Document Version	1.2.0
Document Status	Final
Part of Release	4.0
Revision	3

Document Change History

Date	Version	Changed by	Change Description
09.12.2011	1.2.0	AUTOSAR Administration	• Update of wake-up validation (power-up) • Several minor corrections (typos and wordings)
15.11.2010	1.1.0	AUTOSAR Administration	Literals changed names: 1. the imported LIN interface parameters (from LINInterface) are removed, instead 3 local parameters are introduced. • LINIF_TRCV_MODE_NORMAL -> LINTRCV_TRCV_MODE_NORMAL • LINIF_TRCV_MODE_STANDBY -> LINTRCV_TRCV_MODE_STANDBY • LINIF_TRCV_MODE_SLEEP -> LINTRCV_TRCV_MODE_SLEEP
10.12.2009	1.0.0	AUTOSAR Administration	Initial release

Disclaimer

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

Table of Contents

1 Introduction

This specification specifies functionality, API and configuration of the module LIN transceiver driver. It is responsible to handle the LIN transceiver hardware on an ECU.

A LIN bus transceiver is a hardware device. It is the interface between LIN protocol controller and physical LIN bus. On one hand the transmit data stream of a LIN protocol controller is converted into LIN physical layer compliant bus signals. On the other hand LIN bus data streams are converted into protocol controller input signals. A LIN protocol controller is typically a microcontroller implementation.

Most LIN transceivers support power supply control and wakeup via the bus. A lot of different wakeup/sleep and power supply concepts are available on the market.

In addition so called system basis chips (SBC) are available. Beside LIN transceiver functionalities these devices provide additional features, e.g. detection of electrical malfunctions (e.g. short-circuit to dominant level (GND)), power supply control, advanced watchdogs, LIN transceiver, SPI etc.

1.1 Goal of LIN transceiver driver

The target of this document is to specify interfaces and behaviour, which are applicable to most current LIN transceiver hardware implementations.

[LinTrcv042] ⌈The LIN transceiver driver abstracts the applied LIN transceiver hardware and covers hardware independent interfaces to the higher layers. It abstracts also from ECU layout by using APIs of MCAL layer to access LIN transceiver hardware.⌋(BSW162)

1.2 Explicitly uncovered LIN transceiver functionality

Some LIN bus transceivers offer additional functionality like ECU self test or error detection capability for diagnostics.

ECU self test and error detection are not defined within AUTOSAR and requiring such functionality in general would lock out most currently used transceiver hardware chips. Therefore, features like “ground shift detection”, “selective wakeup”, “slope control” and others are not supported.

2 Acronyms and abbreviations

Abbreviation	Description
API	Application Program Interface
Channel	A channel is a software exchange medium for data that are defined with the same criteria.
ComM	Communication Manager
Dem	Diagnostic Event Manager
Det	Development Error Tracer
Dio/DIO	Digital input output, one of the SPAL SW modules
EcuM	ECU State Manager
ECU	Electronic Control Unit
Frt	Free Running Timer
Gpt	General purpose Timer
ICU	Interrupt Control Unit
ISR	Interrupt Service Routine
LinTrcv	Lin Transceiver Driver
MCAL	Micro Controller Abstraction Layer
n/a	Not applicable
PDU	Protocol Data Unit
SBC	System Basis Chip; a device, which integrates e.g. LIN and/or LIN transceiver, watchdog and power control.
SPAL	Standard Peripheral Abstraction Layer
SW	Software
SPI	Serial Peripheral Interface
SPI Channel	A channel is a software exchange medium for data that are defined with the same criteria: configuration parameters, number of data elements with same size and data pointers (source & destination) or location. See specification of SPI driver for more details.
SPI Job	A job is composed of one or several channels with the same chip select. A job is considered to be atomic and therefore cannot be interrupted. A job has also an assigned priority. See specification of SPI driver for more details.
SPI Sequence	A sequence is a number of consecutive jobs to be transmitted. A sequence depends on a static configuration. See specification of SPI driver for more details.

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] Requirements on LIN
AUTOSAR_SRS_LIN.pdf

[5] Specification of ECU Configuration
AUTOSAR_TPS_ECUConfiguration.pdf

3.2 Related standards and norms

[6] Specification of LIN Driver
AUTOSAR_SWS_LINDriver.pdf

[7] Specification of LIN Interface
AUTOSAR_SWS_LINInterface.pdf

[8] Specification of ECU State Manager
AUTOSAR_SWS_ECUStateManager.pdf

[9] Specification of Standard Types
AUTOSAR_SWS_StandardTypes.pdf

[10] Specification of Communication Stack Types
AUTOSAR_SWS_CommunicationStackTypes.pdf

[11] Basic Software Module Description Template
AUTOSAR_TPS_BSWModuleDescriptionTemplate.pdf

4 Constraints and assumptions

4.1 Limitations

The used APIs of underlying drivers like DIO or SPI shall be synchronous. Implementations of underlying drivers, which do not support synchronous behavior, cannot be used together with LIN transceiver driver.

4.2 Applicability to car domains

This driver might be applicable in all car domains using LIN for communication.

5 Dependencies to other modules

Module	*Dependencies*
LinIf	All LIN transceiver drivers are arranged below LinIf.
ComM	ComM steers LIN transceiver driver communication modes via LinIf. Independent steering of each single LIN transceiver channel is possible.
Det	Det gets development error information from LIN transceiver driver.
Dem	Dem gets production error information from LIN transceiver driver.
Dio	Dio module is used to access LIN transceiver hardware connected via ports.
EcuM	EcuM gets wakeup information from LIN transceiver driver via LinIf.
Icu	Icu module might perform LIN transceiver hardware interrupts.
Spi	Spi module is used to access LIN transceiver hardware connected via Spi

5.1 File structure

5.1.1 Naming convention for transceiver driver implementation

[LinTrcv070] 「In case different LIN transceiver hardware implementations are used in one ECU the function names of the different LIN transceiver drivers must be modified such that no two functions with the same names are generated. The names may be extended with a vendor ID or a type ID.」(BSW00347)

5.1.2 Code file structure

[LinTrcv064] 「The naming convention prescribed by AUTOSAR is applied to all files of the LinTrcv module.」(BSW00300)

[LinTrcv065] 「Module consists of listed files:

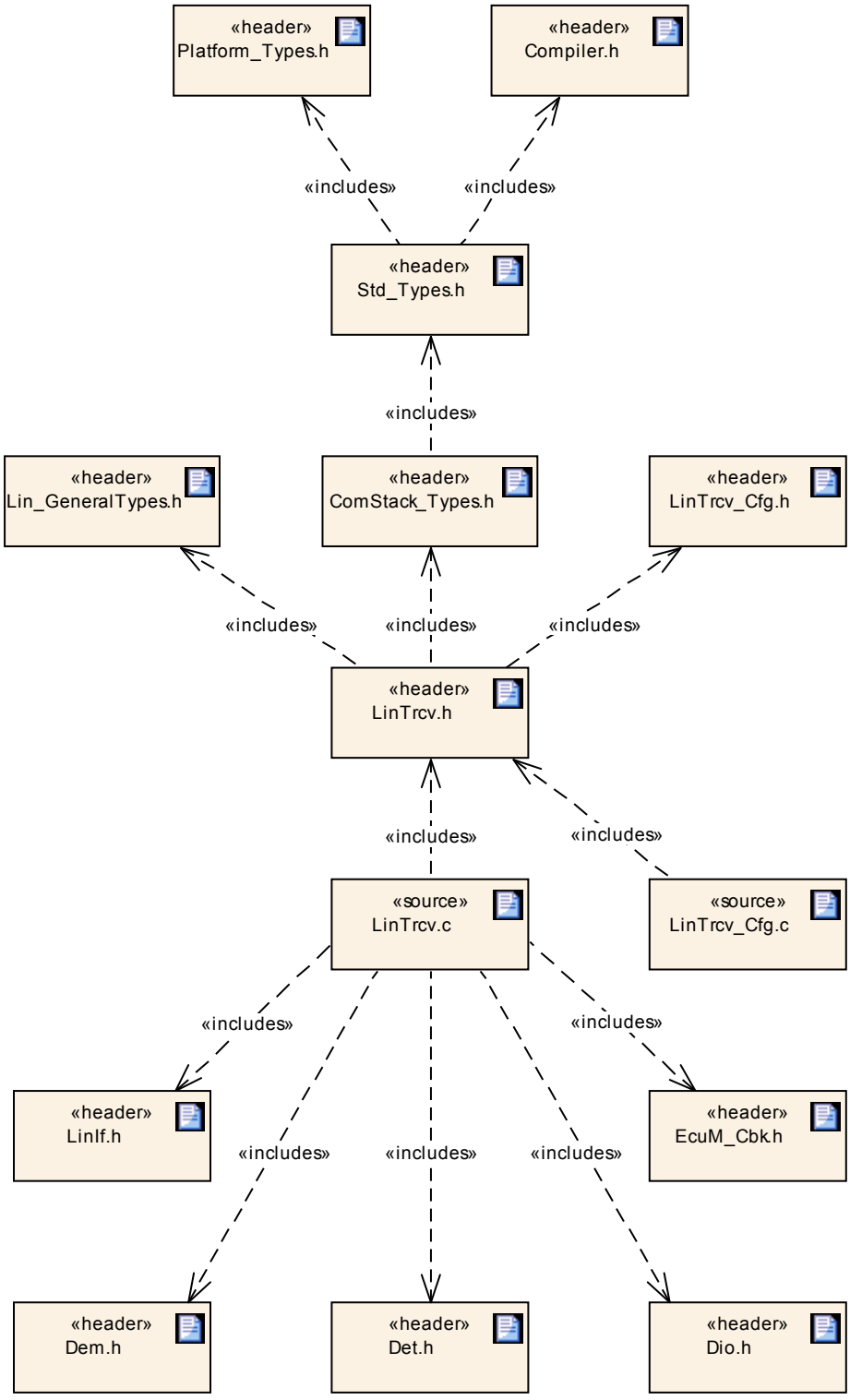
File name	*Requirements*	*Description*
LinTrcv.c	LinTrcv069	The implementation general c file. It does not contain interrupt routines.
LinTrcv.h	LinTrcv052	It contains only information relevant for other BSW modules (API). Differences in API depending on configuration are encapsulated.
LinTrcv_Cfg.h	LinTrcv083	Pre-compile time configuration parameter file. It's generated by the configuration tool.
LinTrcv_Cfg.c	LinTrcv062	Pre-compile time configuration code file. It's generated by the configuration tool.

(BSW00346, BSW158)

5.1.3 Header file structure

[LinTrcv067] ⌈The include file structure shall be as follows

- LinTrcv.c shall include LinIf.h (typedefs needed for Lin Transceiver Driver APIs)
- LinTrcv.c shall include Dem.h (needed to notify about production errors)
- LinTrcv.c shall include Det.h (needed to notify about development errors) if development error detection for the module LinTrcv is enabled.
- LinTrcv.c shall include EcuM_Cbk.h (needed to notify about wakeup of a LIN channel)
- LinTrcv.c shall include Dio.h (DIO APIs needed to access Transceiver pins)
- LinTrcv.c shall include LinTrcv.h (own function prototypes, defines …)
- LinTrcv_Cfg.c shall include LinTrcv.h (own function prototypes, defines …)
- LinTrcv.h shall include LinTrcv_Cfg.h
- LinTrcv_Cfg.h shall include ComStack_Types.h⌋(BSW00301, BSW00409)



Header include structure 5-1

[LinTrcv068] ⌈For AUTOSAR standard data types header file Std_Types.h is included.⌋(BSW00348)

[LinTrcv164] ⌈LinTrcv_TrcvWakeupModeType and LinTrcv_TrcvWakeupReasonTyp shall be defined in Lin_GeneralTypes.h, see also chapter “8.1 Imported types”.

[LinTrcv061] ⌈Name of compiler specific header file is Compiler.h. All mappings of not standardized keywords of compiler specific scope shall be placed and organized in this compiler specific type and keyword header.⌋(BSW00361)

[LinTrcv063] ⌈Name of platform specific header file is Platform_Types.h. All integer type definitions of target and compiler specific scope shall be placed and organized in this single type header.⌋(BSW00353)

6 Requirements Traceability

Requirement	Description	Satisfied by
BSW00300	The naming convention prescribed by AUTOSAR is applied to all files of the LinTrcv module.	LinTrcv064
BSW00301	The include file structure shall be as follows	LinTrcv067
BSW00304	These requirements are not applicable to this specification.	LinTrcv999
BSW00305	These requirements are not applicable to this specification.	LinTrcv999
BSW00306	These requirements are not applicable to this specification.	LinTrcv999
BSW00307	These requirements are not applicable to this specification.	LinTrcv999
BSW00308	These requirements are not applicable to this specification.	LinTrcv999
BSW00309	These requirements are not applicable to this specification.	LinTrcv999
BSW00310		LinTrcv001, LinTrcv002, LinTrcv005, LinTrcv007, LinTrcv008, LinTrcv012
BSW00312	These requirements are not applicable to this specification.	LinTrcv999
BSW00321	These requirements are not applicable to this specification.	LinTrcv999
BSW00323	If the configuration parameter LINTRCV_DEV_ERROR_DETECT is enabled, API parameter checking is act...	LinTrcv048
BSW00325	These requirements are not applicable to this specification.	LinTrcv999
BSW00326	These requirements are not applicable to this specification.	LinTrcv999
BSW00327		LinTrcv050
BSW00328	These requirements are not applicable to this specification.	LinTrcv999
BSW00329		LinTrcv001, LinTrcv002, LinTrcv005, LinTrcv007, LinTrcv008, LinTrcv012
BSW00330	These requirements are not applicable to this specification.	LinTrcv999
BSW00331	These requirements are not applicable to this specification.	LinTrcv999
BSW00333	These requirements are not applicable to this specification.	LinTrcv999
BSW00334	These requirements are not applicable to this specification.	LinTrcv999
BSW00335	These requirements are not applicable to this	LinTrcv999

	specification.	
BSW00336	These requirements are not applicable to this specification.	LinTrcv999
BSW00337	Development error values are of type uint8.	LinTrcv057
BSW00338	Detected development errors will be reported to the error hook of the Development Error Tracer (D...	LinTrcv040
BSW00339	The detection of production code errors cannot be switched off.	LinTrcv058, LinTrcv024
BSW00341	These requirements are not applicable to this specification.	LinTrcv999
BSW00342	These requirements are not applicable to this specification.	LinTrcv999
BSW00344	These requirements are not applicable to this specification.	LinTrcv999
BSW00347	In case different LIN transceiver hardware implementations are used in one ECU the function names...	LinTrcv070, LinTrcv016
BSW00348	For AUTOSAR standard data types header file Std_Types.	LinTrcv068
BSW00350	The detection of all development errors is configurable (ON/OFF) at pre-compile time.	LinTrcv023
BSW00353	Name of platform specific header file is Platform_Types.	LinTrcv063
BSW00355	These requirements are not applicable to this specification.	LinTrcv999
BSW00357		LinTrcv002
BSW00358		LinTrcv001
BSW00359	These requirements are not applicable to this specification.	LinTrcv999
BSW00360	These requirements are not applicable to this specification.	LinTrcv999
BSW00361	Name of compiler specific header file is Compiler.	LinTrcv061
BSW00369		LinTrcv002, LinTrcv005, LinTrcv007, LinTrcv008, LinTrcv012
BSW00371		LinTrcv001, LinTrcv002, LinTrcv005, LinTrcv007, LinTrcv008, LinTrcv012
BSW00375		LinTrcv012
BSW00377		LinTrcv005, LinTrcv007
BSW00378	These requirements are not applicable to this specification.	LinTrcv999
BSW00383	These requirements are not applicable to this specification.	LinTrcv999
BSW00384	These requirements are not applicable to this specification.	LinTrcv999
BSW00385		LinTrcv050
BSW00386		LinTrcv050

BSW00398	These requirements are not applicable to this specification.	LinTrcv999
BSW00399	These requirements are not applicable to this specification.	LinTrcv999
BSW004	LinTrcv module shall perform inter-module checks to avoid integration of incompatible files.	LinTrcv158
BSW00400	These requirements are not applicable to this specification.	LinTrcv999
BSW00401	These requirements are not applicable to this specification.	LinTrcv999
BSW00404	These requirements are not applicable to this specification.	LinTrcv999
BSW00405	These requirements are not applicable to this specification.	LinTrcv999
BSW00406	If development errors are enabled and the state of the LIN Transceiver is NOT_ACTIVE and a functi...	LinTrcv105, LinTrcv002, LinTrcv007, LinTrcv008, LinTrcv012
BSW00407		LinTrcv008
BSW00409	The include file structure shall be as follows	LinTrcv067
BSW00410	These requirements are not applicable to this specification.	LinTrcv999
BSW00413	For each LIN transceiver hardware type an ECU has one LIN transceiver driver instance.	LinTrcv016
BSW00414		LinTrcv001
BSW00416	These requirements are not applicable to this specification.	LinTrcv999
BSW00417	These requirements are not applicable to this specification.	LinTrcv999
BSW00420	These requirements are not applicable to this specification.	LinTrcv999
BSW00421	The detection of production code errors cannot be switched off.	LinTrcv058
BSW00422	These requirements are not applicable to this specification.	LinTrcv999
BSW00423	These requirements are not applicable to this specification.	LinTrcv999
BSW00426	These requirements are not applicable to this specification.	LinTrcv999
BSW00427	These requirements are not applicable to this specification.	LinTrcv999
BSW00429	These requirements are not applicable to this specification.	LinTrcv999
BSW00431	These requirements are not applicable to this specification.	LinTrcv999
BSW00432	These requirements are not applicable to this specification.	LinTrcv999
BSW00433	These requirements are not applicable to this	LinTrcv999

	specification.	
BSW00434	These requirements are not applicable to this specification.	LinTrcv999
BSW005	These requirements are not applicable to this specification.	LinTrcv999
BSW006	These requirements are not applicable to this specification.	LinTrcv999
BSW007	These requirements are not applicable to this specification.	LinTrcv999
BSW009	These requirements are not applicable to this specification.	LinTrcv999
BSW010	These requirements are not applicable to this specification.	LinTrcv999
BSW01096		LinTrcv001
BSW01097		LinTrcv001, LinTrcv002, LinTrcv005, LinTrcv007, LinTrcv012
BSW01098	The LIN transceiver driver operation modes are described in the state diagram below.	LinTrcv055, LinTrcv002
BSW01099	The LIN transceiver driver operation modes are described in the state diagram below.	LinTrcv055, LinTrcv002
BSW01100	The LIN transceiver driver operation modes are described in the state diagram below.	LinTrcv055, LinTrcv002
BSW01101		LinTrcv005
BSW01103		LinTrcv007
BSW01115	These requirements are not applicable to this specification.	LinTrcv999
BSW01502	These requirements are not applicable to this specification.	LinTrcv999
BSW01503	These requirements are not applicable to this specification.	LinTrcv999
BSW01504	These requirements are not applicable to this specification.	LinTrcv999
BSW01514	Wakeup notification must be supported by Lin Transceiver driver, therefore LIN transceiver driver...	LinTrcv066
BSW01515	These requirements are not applicable to this specification.	LinTrcv999
BSW01522	These requirements are not applicable to this specification.	LinTrcv999
BSW01523	These requirements are not applicable to this specification.	LinTrcv999
BSW01524	The LIN transceiver driver operation modes are described in the state diagram below.	LinTrcv055, LinTrcv002
BSW01526	Detected development errors will be reported to the error hook of the Development Error Tracer (D...	LinTrcv040, LinTrcv024
BSW01527	These requirements are not applicable to this specification.	LinTrcv999

BSW01534	These requirements are not applicable to this specification.	LinTrcv999
BSW01539	These requirements are not applicable to this specification.	LinTrcv999
BSW01540	These requirements are not applicable to this specification.	LinTrcv999
BSW01544	These requirements are not applicable to this specification.	LinTrcv999
BSW01545	These requirements are not applicable to this specification.	LinTrcv999
BSW01546	These requirements are not applicable to this specification.	LinTrcv999
BSW01547	These requirements are not applicable to this specification.	LinTrcv999
BSW01549	These requirements are not applicable to this specification.	LinTrcv999
BSW01551	These requirements are not applicable to this specification.	LinTrcv999
BSW01552	These requirements are not applicable to this specification.	LinTrcv999
BSW01553	These requirements are not applicable to this specification.	LinTrcv999
BSW01555	These requirements are not applicable to this specification.	LinTrcv999
BSW01556	These requirements are not applicable to this specification.	LinTrcv999
BSW01558	These requirements are not applicable to this specification.	LinTrcv999
BSW01560	These requirements are not applicable to this specification.	LinTrcv999
BSW01563	Wakeup notification must be supported by Lin Transceiver driver, therefore LIN transceiver driver...	LinTrcv066
BSW01564	These requirements are not applicable to this specification.	LinTrcv999
BSW01566	The LIN transceiver driver operation modes are described in the state diagram below.	LinTrcv055, LinTrcv002
BSW01568	These requirements are not applicable to this specification.	LinTrcv999
BSW01569	These requirements are not applicable to this specification.	LinTrcv999
BSW01571	These requirements are not applicable to this specification.	LinTrcv999
BSW01572	These requirements are not applicable to this specification.	LinTrcv999
BSW01574	These requirements are not applicable to this specification.	LinTrcv999
BSW01576	These requirements are not applicable to this specification.	LinTrcv999

BSW01577	These requirements are not applicable to this specification.	LinTrcv999
BSW01579	These requirements are not applicable to this specification.	LinTrcv999
BSW01580	Selection of wakeup mode shall be done by configuration parameter LinTrcvWakeUpSupportLINTRCV_WAK...	LinTrcv074, LinTrcv075
BSW101		LinTrcv001
BSW159	These requirements are not applicable to this specification.	LinTrcv999
BSW161	These requirements are not applicable to this specification.	LinTrcv999
BSW162	The LIN transceiver driver abstracts the applied LIN transceiver hardware and covers hardware ind...	LinTrcv042
BSW164	These requirements are not applicable to this specification.	LinTrcv999
BSW167	These requirements are not applicable to this specification.	LinTrcv999
BSW168	These requirements are not applicable to this specification.	LinTrcv999
BSW170	These requirements are not applicable to this specification.	LinTrcv999

Document: AUTOSAR General Requirements on Basic Software Modules

Requirement	***Satisfied by***
[BSW003] Version identification	LinTrcv021
[BSW00300] Module naming convention.	LinTrcv064
[BSW00301] Limit imported information	LinTrcv067
[BSW00302] Limit exported information.	LinTrcv
[BSW00304] AUTOSAR integer data types	not applicable (general implementation requirement)
[BSW00305] Self-defined data types naming convention	not applicable (no self defined data types)
[BSW00306] Avoid direct use of compiler and platform specific keyword	not applicable (general implementation requirement)
[BSW00307] Naming convention for global variables	not applicable (general implementation requirement)
[BSW00308] Definition of global data	not applicable (general implementation requirement)
[BSW00309] Global read only data with read only constraint	not applicable (general implementation requirement)
[BSW00310] API naming convention	LinTrcv001, LinTrcv002, LinTrcv005, LinTrcv007, LinTrcv008, LinTrcv012
[BSW00312] Shared code shall be reentrant	not applicable (general implementation requirement)
[BSW00314] Separation of interrupt frames and services routines	LinTrcv069
[BSW00318] Format of module version numbers	LinTrcv021
[BSW00321] Enumeration of module version numbers	not applicable (general implementation requirement)
[BSW00323] API parameter checking	LinTrcv048, LinTrcv107
[BSW00325] Runtime of interrupt service routines	not applicable

	(LIN transceiver driver implements no ISRs)
[BSW00326] Transition from ISRs to OS tasks	not applicable (no such transitions are performed)
[BSW00327] Error values naming convention	LinTrcv050
[BSW00328] Avoid duplication of code	not applicable (general implementation requirement)
[BSW00329] Avoidance of generic interfaces	LinTrcv001, LinTrcv002, LinTrcv005, LinTrcv007, LinTrcv008, LinTrcv012
[BSW00330] Use of macros and inline functions	not applicable (general implementation requirement)
[BSW00331] Separation of error and status values	not applicable (no such values defined)
[BSW00333] Documentation of callback function context	not applicable (general documentation requirement)
[BSW00334] Provision of XML file	not applicable (general implementation requirement)
[BSW00335] Status values naming convention	not applicable
[BSW00336] Shut down interface	not applicable (no need for such interfaces)
[BSW00337] Classification of errors	LinTrcv057
[BSW00338] Detection and reporting of development errors	LinTrcv040, LinTrcv090_Conf, LinTrcv073
[BSW00339] Reporting of production relevant error status	LinTrcv024, LinTrcv060, LinTrcv058
[BSW00341] Microcontroller compatibility documentation	not applicable (general documentation requirement)
[BSW00342] Use of source code and object code	not applicable (general implementation requirement)
[BSW00343] Specification and configuration of time	LinTrcv090_Conf
[BSW00344] Reference to link time configuration	not applicable (only pre-compile time configuration supported)
[BSW00345] Pre-compile time configuration	LinTrcv062, LinTrcv083
[BSW00346] Basic set of module files	LinTrcv065
[BSW00347] Naming separation of different instances of BSW drivers	LinTrcv016, LinTrcv070
[BSW00348] Standard type header	LinTrcv068
[BSW00350] Development error detection keyword	LinTrcv023, LinTrcv090_Conf
[BSW00353] Platform specific type header	LinTrcv063
[BSW00355] Do not redefine AUTOSAR integer data types	not applicable (general implementation requirement)
[BSW00357] Standard API return type	LinTrcv002
[BSW00358] Return type of init() functions	LinTrcv001
[BSW00359] Return type of callback functions	not applicable
[BSW00360] Parameters of callback functions	not applicable
[BSW00361] Compiler specific language extension header	LinTrcv061
[BSW00369] Do not return development error codes via API	LinTrcv001_Conf, LinTrcv002, LinTrcv005, LinTrcv007, LinTrcv008, LinTrcv012
[BSW00370] Separation of callback interfaces from API	LinTrcv071
[BSW00371] Do not pass function pointers via API	LinTrcv001, LinTrcv002, LinTrcv005, LinTrcv007, LinTrcv008, LinTrcv012
[BSW00374] Module vendor identification	LinTrcv021
[BSW00375] Notification of wakeup reason	LinTrcv012
[BSW00377] Module specific API return types	LinTrcv005, LinTrcv007
[BSW00378] AUTOSAR □ropaga type	not applicable (general implementation requirement)

[BSW00379] Module identification	LinTrcv021
[BSW00380] Separate C file for configuration parameters	LinTrcv062
[BSW00381] Separate configuration H file for pre-compile time parameters	LinTrcv083
[BSW00383] List dependencies of configuration elements	not applicable (general documentation requirement)
[BSW00384] List dependencies to other modules	not applicable (general documentation requirement)
[BSW00385] List possible error notifications	LinTrcv050
[BSW00386] Configuration for detecting an error	LinTrcv050
[BSW00387] Specify the configuration class of callbacks	LinTrcv012_Conf
[BSW00388] Introduce containers	LinTrcv090_Conf, LinTrcv091_Conf, LinTrcv092 LinTrcv093, LinTrcv094_Conf, LinTrcv095
[BSW00389] Container shall have names	LinTrcv090_Conf, LinTrcv091_Conf, LinTrcv092 LinTrcv093, LinTrcv094_Conf, LinTrcv095
[BSW00390] Parameter content unique within the module	LinTrcv090_Conf, LinTrcv091_Conf, LinTrcv092 LinTrcv093, LinTrcv094_Conf, LinTrcv095
[BSW00391] Parameters shall have unique names	LinTrcv090_Conf, LinTrcv091_Conf, LinTrcv092 LinTrcv093, LinTrcv094_Conf, LinTrcv095
[BSW00392] Parameters shall have unique types	LinTrcv090_Conf, LinTrcv091_Conf, LinTrcv092 LinTrcv093, LinTrcv094_Conf, LinTrcv095
[BSW00393] Parameters shall have a range	LinTrcv090_Conf, LinTrcv091_Conf, LinTrcv092 LinTrcv093, LinTrcv094_Conf, LinTrcv095
[BSW00394] Specify the scope of the parameters	LinTrcv090_Conf, LinTrcv091_Conf, LinTrcv092 LinTrcv093, LinTrcv094_Conf, LinTrcv095
[BSW00395] List the required parameters (per parameter)	LinTrcv091_Conf, LinTrcv092, LinTrcv093 LinTrcv094_Conf, LinTrcv095
[BSW00396] Configuration classes	LinTrcv017
[BSW00397] Pre-compile time parameters	LinTrcv062, LinTrcv083
[BSW00398] Link time parameters	not applicable (only pre-compile time configuration supported)
[BSW00399] Loadable post build time parameters	not applicable (only pre-compile time configuration supported)
[BSW004] Version check	LinTrcv158
[BSW00400] Selectable post build time parameters	not applicable (only pre-compile time configuration supported)
[BSW00401] Documentation of multiple instances of configuration parameters	not applicable (general documentation requirement)
[BSW00402] Published information	LinTrcv021
[BSW00404] Reference to post build time configuration	not applicable (only pre-compile time configuration supported)
[BSW00405] Reference to multiple configuration sets	not applicable (only pre-compile time configuration supported)
[BSW00406] Check module initialization	LinTrcv002, LinTrcv005_Conf, LinTrcv007, LinTrcv008, LinTrcv012, LinTrcv105
[BSW00407] Function to read out published parameters	LinTrcv008
[BSW00408] Configuration Parameter naming convention	LinTrcv090_Conf, LinTrcv091_Conf, LinTrcv092 LinTrcv093, LinTrcv094_Conf, LinTrcv095
[BSW00409] Header files for production code error	LinTrcv067
[BSW00410] Compiler switches shall have defined values	not applicable (general implementation requirement)
[BSW00411] Get version information keyword	LinTrcv090_Conf
[BSW00412] Separate H file for configuration parameters	LinTrcv083
[BSW00413] Accessing instances of BSW	LinTrcv016

modules	
[BSW00414] Parameters of init function	LinTrcv001
[BSW00415] User dependent include files	LinTrcv052
[BSW00416] Sequence of initialization	not applicable (this is out of LIN transceiver driver's scope)
[BSW00417] Reporting of error events by non basic software	not applicable (Requirement concerns application components only)
[BSW00419] Separate C file for pre-compile time configuration parameters	LinTrcv062
[BSW00420] Production relevant error event rate detection	not applicable (it's an Dem requirement)
[BSW00421] Reporting of production relevant error events	LinTrcv058 LinTrcv060
[BSW00422] Debouncing of production relevant error status	not applicable (it's an Dem requirement)
[BSW00423] Usage of SW C template to describe BSW modules with AUTOSAR interfaces	not applicable (general implementation requirement)
[BSW00425] Trigger condition for schedulable objects	LinTrcv090_Conf
[BSW00426] Exclusive areas in BSW modules	not applicable (LIN transceiver driver is part of ECU abstraction layer)
[BSW00427] ISR description for BSW modules	not applicable (No such areas or function in LIN transceiver driver)
[BSW00429] Restricted BSW OS functionality access	not applicable (general implementation requirement)
[BSW00431] The BSW scheduler module implements task bodies	not applicable (requirement concerns BSW scheduler module)
[BSW00432] Modules should have separate main processing functions for read/receive and write/transmit data path	not applicable (LIN transceiver driver does not □ropagate data)
[BSW00433] Calling of main processing functions	not applicable (requirement concerns BSW scheduler module)
[BSW00434] The schedule module shall provide an API for exclusive areas	not applicable (requirement concerns BSW scheduler module)
[BSW005] No hard coded horizontal interfaces within MCAL	not applicable (LIN transceiver driver is part of ECU abstraction layer)
[BSW006] Platform independency	not applicable (general implementation requirement)
[BSW007] HIS Misra C	not applicable (general implementation requirement)
[BSW009] Module user documentation	not applicable (general documentation requirement)
[BSW010] Memory resource documentation	not applicable (general documentation requirement)
[BSW101] Initialization interface	LinTrcv001
[BSW158] Separation of configuration from implementation	LinTrcv065
[BSW159] Tool-based configuration	not applicable (general tool requirement)
[BSW160] Human readable configuration data	LinTrcv090_Conf, LinTrcv091_Conf, LinTrcv092 LinTrcv093, LinTrcv094_Conf, LinTrcv095
[BSW161] Microcontroller abstraction	not applicable (LIN transceiver driver is part of ECU abstraction layer)
[BSW162] ECU layout abstraction	LinTrcv042

[BSW164] Implementation of interrupt service routines	not applicable (LIN transceiver driver implements no ISRs)
[BSW167] Static configuration checking	not applicable (general tool requirement)
[BSW168] Diagnostic Interface of SW components	not applicable (LIN transceiver driver has no such needs)
[BSW170] Data for reconfiguration of AUTOSAR SW components	not applicable (general tool requirement)
[BSW171] Configurability of optional functionality	LinTrcv012_Conf
[BSW172] Compatibility and documentation of scheduling strategy	LinTrcv001, LinTrcv090_Conf LinTrcv091_Conf, LinTrcv099

6.1 Document: AUTOSAR Requirements on LIN

Requirement	***Satisfied by***
[BSW01576] The LIN 2.1 specification shall be reused as far as possible	Not applicable (no influence of LIN version to this document)
[BSW01504] Usage of AUTOSAR architecture only in LIN master nodes mandatory	Not applicable (anyway only LIN master is in the scope of AUTOSAR)
[BSW01522] Consistent data transfer	Not applicable (out of LIN transceivers scope)
[BSW01560] Support for wakeup during transition to sleep-mode	Not applicable (because APIs are synchronous)
[BSW01577] Compatibility to LIN 2.1 protocol specification	Not applicable (no influence of LIN version to this document)
[BSW01551] Multiple LIN channel support for interface	Not applicable (LIN interface related)
[BSW01568] Hardware Independence	Not applicable (HW is abstracted by LIN driver)
[BSW01569] LIN Interface initialization	Not applicable (LIN interface related)
[BSW01570] Selection of static configuration sets	See [BSW01091]
[BSW01564] Schedule Table Manager	Not applicable (out of LIN transceivers scope)
[BSW01546] Schedule Table Handler	Not applicable (out of LIN transceivers scope)
[BSW01561] Main function	
[BSW01549] Timer service for Scheduling	Not applicable (out of LIN transceivers scope)
[BSW01571] Transmission request service	Not applicable (out of LIN transceivers scope)
[BSW01514] Wakeup notification support	LinTrcv066
[BSW01515] API to wakeup by upper layer to LIN Interface	Not applicable (out of LIN transceivers scope)
[BSW01502] RX indication and TX confirmation call-backs	Not applicable (out of LIN transceivers scope)
[BSW01558] Check successful communication	Not applicable (out of LIN transceivers scope)
[BSW01527] Notification for missing or faulty receive LIN-PDU	Not applicable (out of LIN transceivers scope)
[BSW01523] API to send the LIN to sleep-mode	Not applicable (out of LIN transceivers scope)

[BSW01577] Compatibility to LIN 2.1 protocol specification	Not applicable (no influence of LIN version to this document)
[BSW01553] Basic Software SPAL General requirements	Not applicable (general implementation requirement)
[BSW01552] Hardware abstraction LIN	Not applicable (Lin interface related)
[BSW01503] Frame based API for send and received data	Not applicable (out of LIN transceivers scope)
[BSW01555] LIN Interface shall poll the LIN Driver for transmit/receive notifications	Not applicable (Lin interface related)
[BSW01547] Support of UART and LIN optimized HW	Not applicable (LIN driver related)
[BSW01572] LIN driver initialization	Not applicable (Lin driver related)
[BSW01563] Wakeup Notification	LinTrcv066
[BSW01556] Multiple LIN channel support for driver	Not applicable (Lin driver related)
[BSW01566] Transition to sleep-mode	LinTrcv002, LinTrcv055, LinTrcv056
[BSW01524] Support of reduced power operation mode	LinTrcv002, LinTrcv055, LinTrcv056
[BSW01526] Error notification	LinTrcv024, LinTrcv040
[BSW01579] Compatibility to TP of LIN 2.1 specification	Not applicable (TP related)
[BSW01540] LIN Transport Layer Initialization	Not applicable (TP related)
[BSW01545] LIN Transport Layer Availability	Not applicable (TP related)
[BSW01534] Concurrent connection configuration	Not applicable (TP related)
[BSW01574] Multiple Transport Layer instances	Not applicable (TP related)
[BSW01539] Transport connection properties	Not applicable (TP related)
[BSW01544] Error handling	Not applicable (Lin interface related)

6.2 Document: AUTOSAR Requirements on LIN / LIN Transceiver Driver

Requirement	***Satisfied by***
[BSW01090] Configuration Data for LIN Bus Transceiver	LinTrcv090_Conf, LinTrcv091_Conf, LinTrcv092 LinTrcv093, LinTrcv094_Conf, LinTrcv095
[BSW01091] Support for more than one LIN transceiver. Only pre-compile time configuration allowed.	LinTrcv002, LinTrcv005_Conf, LinTrcv009_Conf, LinTrcv011_Conf, LinTrcv012_Conf, LinTrcv016, LinTrcv017, LinTrcv151_Conf
[BSW01096] API to initialize the LIN bus transceiver driver	LinTrcv001
[BSW01097] LIN bus transceiver driver API shall be synchronous	LinTrcv001, LinTrcv002, LinTrcv005, LinTrcv007, LinTrcv012
[BSW01098] API to request operation mode Standby	LinTrcv002, LinTrcv055
[BSW01099] API to request operation mode Sleep	LinTrcv002, LinTrcv055, LinTrcv056
[BSW01100] API to request operation mode Normal	LinTrcv002, LinTrcv055
[BSW01101] API to read out current operation mode	LinTrcv005

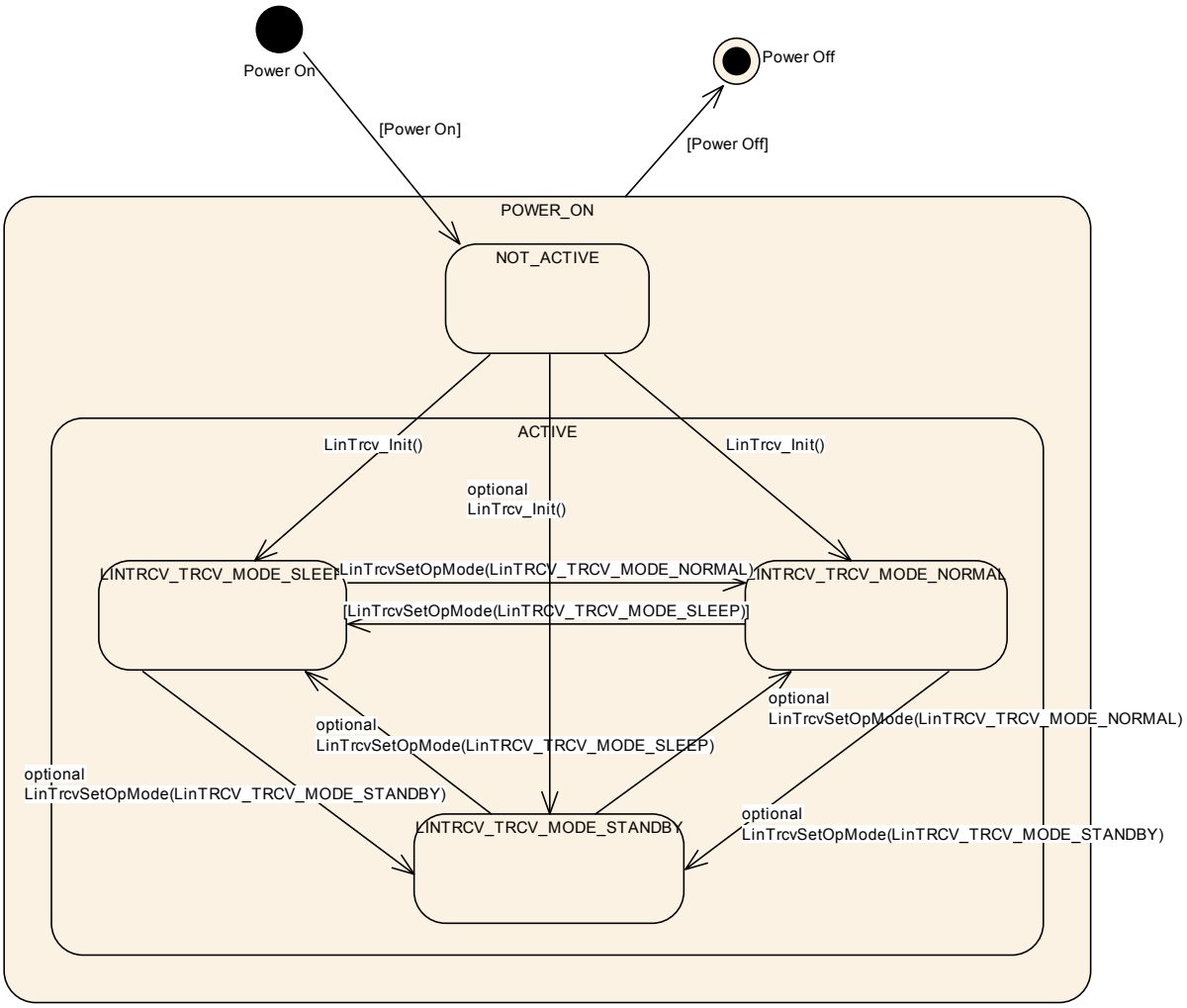
[BSW01103] API to read out wakeup reason	LinTrcv007
[BSW01115] Support API for enable/disable and clear wakeup event	not applicable
[BSW01580] Configuration Data for LIN Transceiver Driver.	LinTrcv074, LinTrcv075

7 Functional specification

7.1 LIN transceiver driver operation modes

[LinTrcv055] ⌈The LIN transceiver driver operation modes are described in the state diagram below.⌋(BSW01566, BSW01524, BSW01098, BSW01099, BSW01100)

The main idea behind this diagram is to support the majority of available LIN bus transceivers in a common model view. Depending on the LIN transceiver hardware, the model may have one or two states more than necessary for a given LIN transceiver hardware, but this will clearly decouple the ComM and EcuM from the used hardware.



Lin Transceiver Operation Modes 7-1

Hint: There are several optional interfaces that might not be needed for current LIN transceiver hardware. E.g. the mode “LINTRCV_TRCV_MODE_STANDBY” might be only an internal state that is used for internal hardware transitions. Especially if functionality of “inhibit pin” is used to control the uC only the states “LINTRCV_TRCV_MODE_SLEEP” and “LINTRCV_TRCV_MODE_NORMAL” are of interest.

The function LinTrcv_Init() causes a state change to either LINTRCV_TRCV_MODE_SLEEP, LINTRCV_TRCV_MODE_NORMAL or LINTRCV_TRCV_MODE_STANDBY (any of these 3 states belong to the upper state ACTIVE). This depends on the configuration and is independent configurable for each channel.

State	Description
POWER_ON	MCU is fully powered.
NOT_ACTIVE	State of LIN transceiver hardware depend on ECU hardware and on Dio and Port driver configuration. LIN transceiver driver is not initialized and therefore not active.
ACTIVE	The function LinTrcv_Init() was called. It carries LIN transceiver driver to active state. Depending on configuration LIN transceiver driver enters state LINTRCV_TRCV_MODE_SLEEP, LINTRCV_TRCV_MODE_STANDBY or LINTRCV_TRCV_MODE_NORMAL.
LINTRCV_TRCV_MODE_NORMAL	Full bus communication. If LIN transceiver hardware controls MCU power supply, MCU is fully powered. The LIN transceiver driver detects no further wakeup information.
LINTRCV_TRCV_MODE_STANDBY	No communication is possible. If LIN transceiver hardware controls MCU power supply, the MCU is still powered. A wakeup by bus or by a local wakeup event is possible. Note: This is an optional state.
LINTRCV_TRCV_MODE_SLEEP	No communication is possible. If LIN transceiver hardware controls MCU power supply, the MCU is not powered. A wakeup by bus or by a local wakeup event is possible.

If a LIN transceiver driver covers more than one LIN channel, all channels are either in state NOT_ACTIVE or in state ACTIVE. In state ACTIVE each channel may be in a different sub state.

7.2 LIN transceiver hardware operation modes

The LIN transceiver hardware may support more mode transitions than the software. The dependencies and the recommended implementations behaviour are explained in this chapter.

It is up to the implementation to decide which LIN transceiver hardware state is covered by which LIN transceiver driver software state. An implementation has to guarantee that whole functionality of described LIN transceiver driver is given by the implementation.

7.3 LIN transceiver wakeup types

There are four different scenarios, which are often called wakeup:

1) MCU is not powered, parts of ECU including LIN transceiver hardware are powered. The considered LIN transceiver hardware is in mode LINTRCV_TRCV_MODE_SLEEP. A wakeup event on LIN is detected by LIN transceiver hardware. LIN transceiver hardware causes powering of MCU (e.g. via pin “inhibit”). In terms of AUTOSAR this is kept as a cold start and not as a wakeup.

2) MCU is in low power mode, parts of ECU including LIN transceiver hardware are powered. Depending on the hardware implementation the considered LIN transceiver hardware is either in mode LINTRCV_TRCV_MODE_STANDBY or LINTRCV_TRCV_MODE_SLEEP. A wakeup event on LIN is detected by LIN transceiver hardware. LIN transceiver hardware is informing MCU about wakeup. In terms of AUTOSAR this is kept as a wakeup of the LIN channel and of the MCU.

3) MCU is in full power mode, at least parts of the ECU including LIN transceiver hardware are powered. Depending on the hardware implementation the considered LIN transceiver hardware is either in mode LINTRCV_TRCV_MODE_STANDBY or LINTRCV_TRCV_MODE_SLEEP. A wakeup event on LIN is detected by LIN transceiver hardware. LIN transceiver hardware is informing MCU about wakeup or is polled cyclically for wakeup events. In terms of AUTOSAR this is kept as a wakeup of a LIN channel.

4) MCU is in full power mode, at least parts of the ECU including LIN transceiver hardware are powered. Depending on the hardware implementation the considered LIN transceiver hardware is either in mode LINTRCV_TRCV_MODE_STANDBY or LINTRCV_TRCV_MODE_SLEEP. The MCU is now setting the LIN transceiver hardware to mode LINTRCV_TRCV_MODE_NORMAL and is waking up the LIN channel. In terms of AUTOSAR this is kept as an internal wakeup of a LIN channel (through MCU).

7.4 LIN transceiver wakeup modes

[LinTrcv066] ⌈Wakeup notification must be supported by Lin Transceiver driver, therefore LIN transceiver driver covers 2 wakeup modes, internal wakeup by an upper layer or external wakeup by LIN channel.⌋(BSW01514, BSW01563)

1) Internal wakeup
   An internal wakeup is initiated by an upper layer, e.g. by calling LinTrcv_Init() or LinTrcv_SetOpMode.

2) External wakeup
   Wakeup detected by LIN transceiver driver is forwarded to the upper layer through the API LinTrcv_CheckWakeup which has to be called by the LinIf.

Hint: WakeUp through ISR is not supported by the Lin Transceiver Driver but is only possible through ICU.

[LinTrcv074] ⌈Selection of wakeup mode shall be done by configuration parameter LinTrcvWakeUpSupport. (cf. LinTrcv107_Conf)⌋(BSW01580)

[LinTrcv075] ⌈Support of wakeup shall be switched on and off for each LIN transceiver channel individually by configuration parameter LinTrcvWakeupByBusUsed. (cf. LinTrcv006_Conf)⌋(BSW01580)

[LinTrcv161]⌈ LinTrcv driver shall use the following APIs provided by ICU driver, to enable and disable the wakeup event notification:

- Icu_EnableNotification
- Icu_DisableNotification ⌋()

[LinTrcv162]⌈LinTrcv driver shall enable the ICU channels when the transceiver transmits to standby mode (LINTRCV_STANDBY) ⌋()

[LinTrcv163]⌈ LinTrcv driver shall disable the ICU channels when the transceiver transmits to Normal mode (LINTRCV_NORMAL) ⌋()

Rationale: CanTrcv driver shall avoid the loss of wakeup events.

7.5 Error classification

Values for production code event ids are assigned externally by the configuration of the Dem. They are published in the file Dem_IntErrId.h and included via Dem.h.

[LinTrcv057] ⌈Development error values are of type uint8.⌋(BSW00337)

[LinTrcv050] ⌈

Type or error	*Relevance*	*Related error code*	*Value [hex]*
API called with wrong parameter for LIN network	Development	LINTRCV_E_INVALID_LIN_NETWORK	0x01
API called with null pointer parameter	Development	LINTRCV_E_PARAM_POINTER	0x02
API service used without initialization	Development	LINTRCV_E_UNINIT	0x11
API service called in wrong transceiver operation mode	Development	LINTRCV_E_TRCV_NOT_SLEEP LINTRCV_E_TRCV_NOT_NORMAL	0x21 0x22
API service called with invalid parameter for transceiver wakeup mode	Development	LINTRCV_E_PARAM_TRCV_WAKEUP_MODE	0x23
API service called with invalid parameter for operation mode	Development	LINTRCV_E_PARAM_TRCV_OPMODE	0x24
API service called with invalid mode because optional transition is not enabled	Development	LINTRCV_E_INVALID_TRCV_OPMODE	0x25

- * Assignment is done in a header file of module Dem.⌋(BSW00327, BSW00385, BSW00386)

<TrcvIdx> represents transceiver index. The symbol is generated for each transceiver that is managed in the transceiver driver module.

Remark: Development errors are notified to DET.
Production errors are notified to DEM.

[LinTrcv152] ⌈
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.⌋()

7.6 Error detection

[LinTrcv023] ⌈The detection of all development errors is configurable (ON/OFF) at pre-compile time. The configuration parameter *LINTRCV_DEV_ERROR_DETECT* shall activate or deactivate the detection of all development errors. ⌋(BSW00350)

[LinTrcv048] ⌈If the configuration parameter *LINTRCV_DEV_ERROR_DETECT* is enabled, API parameter checking is active.⌋(BSW00323)

[LinTrcv058] ⌈The detection of production code errors cannot be switched off.⌋ (BSW00339, BSW00421)

7.7 Error notification

[LinTrcv040] ⌈Detected development errors will be reported to the error hook of the Development Error Tracer (Det) if the pre-processor configuration parameter *LINTRCV_DEV_ERROR_DETECT* is set.⌋(BSW00338, BSW01526)

[LinTrcv024]⌈ Production errors shall be reported to Diagnostic Event Manager (Dem).⌋(BSW00339, BSW01526)

[LinTrcv105] ⌈If development errors are enabled and the state of the LIN Transceiver is NOT_ACTIVE and a function is called except LinTrcv_Init or LinTrcv_GetVersionInfo the corresponding function shall raise the development error code LINTRCV_E_UNINIT.⌋(BSW00406)

[LinTrcv106] ⌈If development errors are enabled and any API that uses the parameter “LinNetwork” receives an invalid value for this parameter this function shall raise the development error code LINTRCV_E_INVALID_LIN_NETWORK.⌋()

[LinTrcv159] ⌈If development errors are enabled and any API that uses a pointer as parameter receives a null pointer as parameter shall raise the development error code LINTRCV_E_PARAM_POINTER.⌋()

7.8 Debugging

[LinTrcv101] ⌈Each variable that shall be accessible by AUTOSAR debugging, shall be defined as global variable.⌋()

[LinTrcv102] ⌈All type definitions of variables which shall be debugged, shall be accessible by the header file LinTrcv.h.⌋()

[LinTrcv103] ⌈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”.⌋()

[LinTrcv104] 「Variables available for debugging shall be described in the respective Basic Software Module Description.」()

7.9 Preconditions for driver initialization

[LinTrcv099] ⌈The LIN bus transceiver driver might use drivers for Dio or Spi to control the LIN bus transceiver hardware. Thus these drivers must be available and ready to operate before the LIN bus transceiver driver is initialized.⌋()

The LIN transceiver driver may have timing requirements for the initialization sequence and the access to the transceiver device, which must be fulfilled by these used underlying drivers.

The timing requirements might be that
1. The call of the LIN bus transceiver driver initialization has to be performed very early after power up to be able to read all necessary information out of the transceiver hardware in time for all other users within the ECU.
2. The runtime of the used underlying services is very short and synchronous to enable the driver to keep his own timing requirements limited by the used hardware device.
3. The runtime of the driver may be enlarged, as some hardware devices have the need to have the port pin level valid for e.g. 50µs before changing it again to reach a specific state, e.g. sleep.

7.10 Instance concept

[LinTrcv016] ⌈For each LIN transceiver hardware type an ECU has one LIN transceiver driver instance. One instance serves all LIN transceiver hardware of the same type.⌋(BSW00347, BSW00413)

7.11 Wait states

For changing operation modes, the LIN transceiver hardware may have to perform wait states.

[LinTrcv150_Conf] ⌈The wait states shall be realized with the configuration parameter
`LinTrcvWaitCount`.⌋()

7.12 Version checking

[LinTrcv158] ⌈LinTrcv module shall perform inter-module checks to avoid integration of incompatible files.⌋(BSW004)

The imported include files shall be checked by pre-processor 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 LinTrcv module.

If the values are not identical to the expected values, an error shall be reported.

8 API specification

8.1 Imported types

Module	*Imported Type*
Dio	Dio_ChannelType
	Dio_LevelType
	Dio_PortLevelType
	Dio_PortType
	Dio_ChannelGroupType
EcuM	EcuM_WakeupSourceType
Icu	Icu_ChannelType
Lin_GeneralTypes	LinTrcv_TrcvWakeupModeType
	LinTrcv_TrcvWakeupReasonType
Spi	Spi_ChannelType
	Spi_DataType
	Spi_NumberOfDataType
	Spi_SequenceType
	Spi_StatusType
Std_Types	Std_ReturnType
	Std_VersionInfoType

8.2 Type definitions

8.2.1 TrcvModeType

Name:	`LinTrcv_TrcvModeType`	
Type:	`Enumeration`	
Range:	`LINTRCV_TRCV_MODE_NORMAL`	Transceiver mode NORMAL
	`LINTRCV_TRCV_MODE_STANDBY`	Transceiver mode STANDBY
	`LINTRCV_TRCV_MODE_SLEEP`	Transceiver mode SLEEP
Description:	Operating modes of the LIN Transceiver Driver	

8.2.2 TrcvWakeupModeType

Name:	`LinTrcv_TrcvWakeupModeType`	
Type:	`Enumeration`	
Range:	`LINTRCV_WUMODE_ENABLE`	The notification for wakeup events is enabled on the addressed network.
	`LINTRCV_WUMODE_DISABLE`	The notification for wakeup events is disabled on the addressed network.
	`LINTRCV_WUMODE_CLEAR`	A stored wakeup event is cleared on the addressed network.
Description:	Wake up operating modes of the LIN Transceiver Driver.	

8.2.3 TrcvWakeupReasonType

Name:	LinTrcv_TrcvWakeupReasonType	
Type:	Enumeration	
Range:	LINTRCV_WU_ERROR	Due to an error wake up reason was not detected. This value may only be reported when error was reported to DEM before.
	LINTRCV_WU_NOT_SUPPORTED	The transceiver does not support any information for the wake up reason.
	LINTRCV_WU_BY_BUS	The transceiver has detected, that the network has caused the wake up of the ECU.
	LINTRCV_WU_BY_PIN	The transceiver has detected a wake-up event at one of the transceiver's pins (not at the LIN bus).
	LINTRCV_WU_INTERNALLY	The transceiver has detected, that the network has been woken up by the ECU via a request to NORMAL mode.
	LINTRCV_WU_RESET	The transceiver has detected, that the wake up is due to an ECU reset.
	LINTRCV_WU_POWER_ON	The transceiver has detected, that the wake up is due to an ECU reset after power on.
Description:	This type denotes the wake up reason detected by the LIN transceiver in detail.	

8.3 Function definitions

8.3.1 LinTrcv_Init

[LinTrcv001] ⌈

Service name:	LinTrcv_Init

Syntax:	`void LinTrcv_Init(`
	`  void`
	`)`

Service ID[hex]:	0x00
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant
Parameters (in):	None
Parameters (inout):	None
Parameters (out):	None
Return value:	None
Description:	Initializes the Lin Transceiver Driver module.

⌋(BSW00310, BSW00329, BSW00358, BSW00371, BSW00414, BSW101, BSW01096, BSW01097)

[LinTrcv119] ⌈
The function `LinTrcv_Init` shall set the LIN transceiver hardware to the state configured by the configuration parameter `LINTRCV_INIT_STATE`.
This can be LINTRCV_TRCV_MODE_NORMAL, LINTRCV_TRCV_MODE_STANDBY or LINTRCV_TRCV_MODE_SLEEP.⌋()

[LinTrcv146] ⌈
The configuration value LINTRCV_TRCV_MODE_STANDBY shall be an optional value for the configuration parameter `LINTRCV_INIT_STATE`.⌋()

[LinTrcv160] ⌈
If wake-up is supported by hardware (i.e. LinTrcvWakeUpSupport == true), during LIN channel initialization it shall be checked if there was a wake-up event on the specific LIN channel, (if supported by hardware). If a wake-up event has been detected, the wake-up shall directly be reported to the EcuM via EcuM_SetWakeupEvent call-back function.⌋()

Configuration:
Configuration parameter `LINTRCV_INIT_STATE` specifies state after call of `LinTrcv_Init`.

Caveats:

The initialization sequence after reset (e.g. power up) is a critical phase for the LIN transceiver driver. The driver will use SPAL functionality (DIO) to access the transceiver hardware. Therefore all necessary BSW drivers must be initialized and usable before.

8.3.2 LinTrcv_SetOpMode

[LinTrcv002] ⌈

Service name:	LinTrcv_SetOpMode	
Syntax:	`Std_ReturnType LinTrcv_SetOpMode( uint8 LinNetwork, LinTrcv_TrcvModeType OpMode )`	
Service ID[hex]:	0x01	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	LinNetwork	LIN network to wich API call has to be applied
	OpMode	The parameter says to which operation mode the change shall be performed.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: will be returned if the transceiver state has been changed to the requested mode. E_NOT_OK: will be returned if the transceiver state change is not accepted or has failed or the parameter is out of the allowed range.
Description:	The internal state of the LIN transceiver driver is switched to mode given in the parameter OpMode.	

⌋ (BSW00310, BSW00329, BSW00357, BSW00369, BSW00371, BSW00406, BSW01566, BSW01524, BSW01097, BSW01098, BSW01099, BSW01100)

[LinTrcv108] ⌈The function `LinTrcv_SetOpMode` shall switch the internal state of channel LinNetwork to the value of the parameter OpMode which can be LINTRCV_TRCV_MODE_NORMAL, LINTRCV_TRCV_MODE_STANDBY or LINTRCV_TRCV_MODE_SLEEP.⌋()

[LinTrcv109] ⌈
The function LinTrcv_SetOpMode shall switch the internal state of channel LinNetwork to the value of LINTRCV_TRCV_MODE_STANDBY if one of the follwing conditions is fullfilled:
a) the channel LinNetwork is in mode LINTRCV_TRCV_MODE_SLEEP and the optional transition from this mode to LINTRCV_TRCV_MODE_STANDBY is enabled.
b) the channel LinNetwork is in mode LINTRCV_TRCV_MODE_NORMAL and the optional transition from this mode to LINTRCV_TRCV_MODE_STANDBY is enabled.
⌋()

[LinTrcv110] ⌈
The function LinTrcv_SetOpMode shall switch the internal state of channel LinNetwork to the value of LINTRCV_TRCV_MODE_SLEEP if one of

the follwing conditions is fullfilled:
a) the channel LinNetwork is in mode LINTRCV_TRCV_MODE_NORMAL
b) the channel LinNetwork is in mode LINTRCV_TRCV_MODE_STANDBY and the optional transition from this mode to LINTRCV_TRCV_MODE_SLEEP is enabled.」()

[LinTrcv147] 「
The function LinTrcv_SetOpMode shall switch the internal state of channel LinNetwork to the value of LINTRCV_TRCV_MODE_NORMAL if one of the follwing conditions is fullfilled:
a) the channel LinNetwork is in mode LINTRCV_TRCV_MODE_SLEEP
b) the channel LinNetwork is in mode LINTRCV_TRCV_MODE_STANDBY and the optional transition from this mode to LINTRCV_TRCV_MODE_NORMAL is enabled.」
()

[LinTrcv111] 「This API is applicable to each transceiver with each value for parameter `LinTrcv_SetOpMode` regardless of whether the transceiver hardware supports these modes or not. This is to simplify the view of the LinIf to the assigned bus. 」()

[LinTrcv112] 「If the requested mode is not supported by the underlying transceiver hardware, the function `LinTrcv_SetOpMode` shall return E_NOT_OK.」()

[LinTrcv113] 「If there is no/incorrect communication to the transceiver, the function `LinTrcv_SetOpMode` shall return `E_NOT_OK`.」()

[LinTrcv114] 「If development error detection for the module LinTrcv is enabled:
If the function `LinTrcv_SetOpMode` is called with OpMode == LINTRCV_TRCV_MODE_STANDBY and the channel LinNetwork is in mode LINTRCV_SLEEP but the optional transition from LINTRCV_SLEEP to LINTRCV_STANDBY is not enabled, the function `LinTrcv_SetOpMode` shall raise the development error `LINTRCV_E_INVALID_TRCV_OPMODE` and return `E_NOT_OK`.」()

[LinTrcv148] 「If development error detection for the module LinTrcv is enabled:
If the function `LinTrcv_SetOpMode` is called with OpMode == LINTRCV_TRCV_MODE_STANDBY and the channel LinNetwork is in mode LINTRCV_NORMAL but the optional transition from LINTRCV_NORMAL to LINTRCV_STANDBY is not enabled, the function `LinTrcv_SetOpMode` shall raise the development error `LINTRCV_E_INVALID_TRCV_OPMODE` and return `E_NOT_OK`.」()

[LinTrcv115] 「If development error detection for the module LinTrcv is enabled:

If optional transition from LINTRCV_STANDBY to LINTRCV_SLEEP is not enabled and the function `LinTrcv_SetOpMode` is called with OpMode == LINTRCV_TRCV_MODE_SLEEP and the channel LinNetwork is not in mode LINTRCV_TRCV_MODE_NORMAL, the function `LinTrcv_SetOpMode` shall raise the development error `LINTRCV_E_TRCV_NOT_NORMAL` and return `E_NOT_OK`.」()

[LinTrcv149] 「If development error detection for the module LinTrcv is enabled: If optional transition from LINTRCV_STANDBY to LINTRCV_NORMAL is not enabled and the function `LinTrcv_SetOpMode` is called with OpMode == LINTRCV_TRCV_MODE_NORMAL and the channel LinNetwork is not in mode LINTRCV_TRCV_MODE_SLEEP, the function `LinTrcv_SetOpMode` shall raise the development error `LINTRCV_E_TRCV_NOT_SLEEP` and return `E_NOT_OK`.」()

[LinTrcv116] 「If development error detection for the module LinTrcv is enabled: If called before the LinTrcv module has been initialized, the function `LinTrcv_SetOpMode` shall raise the development error `LINTRCV_E_UNINIT` and return `E_NOT_OK`.」()

[LinTrcv117] 「If development error detection for the module LinTrcv is enabled: If called with an invalid network number LinNetwork, the function `LinTrcv_SetOpMode` shall raise the development error `LINTRCV_E_INVALID_LIN_NETWORK` and return `E_NOT_OK`.」()

[LinTrcv118] 「If development error detection for the module LinTrcv is enabled: If called with an invalid OpMode, the function `LinTrcv_SetOpMode` shall raise the development error `LINTRCV_E_PARAM_TRCV_OPMODE` and return `E_NOT_OK`.」()

Configuration:
The number of supported busses is set up in the configuration phase.

[LinTrcv157] 「A mode request of the current mode is allowed and shall not lead to an error even if DET is enabled.」()

8.3.3 LinTrcv_GetOpMode

[LinTrcv005] ⌈

Service name:	LinTrcv_GetOpMode	
Syntax:	`Std_ReturnType LinTrcv_GetOpMode( uint8 LinNetwork, LinTrcv_TrcvModeType* OpMode )`	
Service ID[hex]:	0x02	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	LinNetwork	LIN network to which API call has to be applied
Parameters (inout):	None	
Parameters (out):	OpMode	Pointer to operation mode of the bus the API is applied to.
Return value:	Std_ReturnType	E_OK: will be returned if the operation mode is detected E_NOT_OK: will be returned, if service request is failed due to development errors or the operation mode is not detected.
Description:	API detects the actual software state of LIN transceiver driver.	

⌋ (BSW00310, BSW00329, BSW00369, BSW00371, BSW00377, BSW01097, BSW01101)

[LinTrcv121] ⌈The function `LinTrcv_GetOpMode` shall return the actual state of the LIN transceiver driver in the parameter OpMode.⌋()

[LinTrcv122] ⌈If there is no/incorrect communication to the transceiver, the function `LinTrcv_GetOpMode` shall return `E_NOT_OK`.⌋()

[LinTrcv123]⌈ If development error detection for the module LinTrcv is enabled:
If called before the LinTrcv module has been initialized, the function `LinTrcv_GetOpMode` shall raise the development error `LINTRCV_E_UNINIT` and return `E_NOT_OK`.⌋()

[LinTrcv124] ⌈If development error detection for the module LinTrcv is enabled:
If called with an invalid network number LinNetwork, the function `LinTrcv_GetOpMode` shall raise the development error `LINTRCV_E_INVALID_LIN_NETWORK` and return `E_NOT_OK.`⌋()

[LinTrcv125] ⌈If development error detection for the module LinTrcv is enabled:
If called with OpMode == NULL, the function `LinTrcv_GetOpMode` shall raise the development error `LINTRCV_E_PARAM_POINTER` and return `E_NOT_OK`.⌋()

Configuration:
The number of supported busses is statically set in the configuration phase.

8.3.4 LinTrcv_GetBusWuReason

[LinTrcv007] ⌈

Service name:	LinTrcv_GetBusWuReason	

Syntax:	`Std_ReturnType LinTrcv_GetBusWuReason(`
	`  uint8 LinNetwork,`
	`  LinTrcv_TrcvWakeupReasonType* Reason`
	`)`	

Service ID[hex]:	0x03	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	LinNetwork	LIN network to which API call has to be applied
Parameters (inout):	None	
Parameters (out):	Reason	Pointer to wakeup reason of the bus the API is applied to.
Return value:	Std_ReturnType	E_OK: will be returned if the wake up reason is detected E_NOT_OK: will be returned, if service request is failed due to development errors or the wakeup reason is not detected.
Description:	This API provides the reason for the wakeup that the LIN transceiver has detected in the parameter "Reason". The ability to detect and differentiate the possible wakeup reasons depends strongly on the LIN transceiver hardware.	

⌋ (BSW00310, BSW00329, BSW00369, BSW00371, BSW00377, BSW00406, BSW01097, BSW01103)

[LinTrcv126] ⌈The function `LinTrcv_GetBusWuReason` shall return the reason for the wake up that the LIN transceiver has detected in the parameter Reason⌋()

[LinTrcv127] ⌈If there is no/incorrect communication to the transceiver, the function `LinTrcv_GetBusWuReason` shall return `E_NOT_OK`.⌋()

[LinTrcv128] ⌈If development error detection for the module LinTrcv is enabled:
If called before the LinTrcv module has been initialized, the function `LinTrcv_GetBusWuReason` shall raise development error `LINTRCV_E_UNINIT` and return `E_NOT_OK`.⌋()

[LinTrcv129] ⌈If development error detection for the module LinTrcv is enabled:
If called with an invalid network number LinNetwork, the function `LinTrcv_GetBusWuReason` shall raise development error `LINTRCV_E_INVALID_LIN_NETWORK` and return `E_NOT_OK.`⌋()

[LinTrcv130] ⌈If development error detection for the module LinTrcv is enabled:

If called with Reason == NULL, the function LinTrcv_GetBusWuReason shall raise the development error LINTRCV_E_PARAM_POINTER and return E_NOT_OK.」()

Configuration:
The number of supported busses is statically set in the configuration phase.

Caveats:
Be aware that if more than one bus is available each bus may report a different wakeup reason. E.g. if an ECU has LIN, a wakeup by LIN may occur and the incoming data may cause an internal wakeup for another LIN bus.
The LIN transceiver driver has a “per bus” view and does not vote the more important reason or sequence internally. The same may be true if e.g. one transceiver controls the power supply and the other is just powered or un-powered.

8.3.5 LinTrcv_GetVersionInfo

[LinTrcv008] 「

Service name:	LinTrcv_GetVersionInfo	
Syntax:	`void LinTrcv_GetVersionInfo( Std_VersionInfoType* versioninfo )`	
Service ID[hex]:	0x04	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	None	
Parameters (inout):	None	
Parameters (out):	versioninfo	Pointer to version information of this module.
Return value:	None	
Description:	This service provides the version information of this module through the parameter "versioninfo".	

」(BSW00310, BSW00329, BSW00369, BSW00371, BSW00406, BSW00407)

[LinTrcv131] 「The function LinTrcv_GetVersionInfo shall return the version information of this module. The version information contains all data defined in Std_VersionInfoType in “AUTOSAR_SWS_StandardTypes”.」()

[LinTrcv132] 「The function LinTrcv_GetVersionInfo shall be pre-compile time configurable On/Off by the configuration parameter LINTRCV_GET_VERSION_INFO.」()

[LinTrcv134] 「If development error detection for the module LinTrcv is enabled:
If called with VersionInfo == NULL, the function LinTrcv_GetVersionInfo shall raise development error LINTRCV_E_PARAM_POINTER and return E_NOT_OK.」()

Configuration:
This function shall be pre-compile time configurable On/Off by the configuration parameter: LINTRCV_GET_VERSION_INFO.

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

8.3.6 LinTrcv_CheckWakeup

[LinTrcv012] 「

Service name:	LinTrcv_CheckWakeup	
Syntax:	`Std_ReturnType LinTrcv_CheckWakeup( uint8 LinNetwork )`	
Service ID[hex]:	0x07	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	LinNetwork	LIN network to which API call has to be applied.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: Will be returned, if a wakeup has been detected. E_NOT_OK: Will be returned, if no wakeup has been detected
Description:	Notifies the calling function if a wakeup is detected.	

」(BSW00310, BSW00329, BSW00369, BSW00371, BSW00375, BSW00406, BSW01097)

[LinTrcv144] 「If development error detection for the module LinTrcv is enabled:
If called before the LinTrcv module has been initialized, the function `LinTrcv_CheckWakeup` shall raise the development error `LINTRCV_E_UNINIT` and return `E_NOT_OK`.」()

[LinTrcv145] 「If development error detection for the module LinTrcv is enabled:
If called with an invalid network number LinNetwork, the function `LinTrcv_CheckWakeup` shall raise the development error `LINTRCV_E_INVALID_LIN_NETWORK` and return `E_NOT_OK`.」()

[LinTrcv166]「The function `LinTrcv_CheckWakeup` shall evaluate the wakeup on the addressed LIN network. When a wake-up event on the addressed LIN network is detected (e.g. dominant bus state or negative edge at wakeup pin), the function `LinTrcv_CheckWakeup` shall notify the ECU State Manager module immediately via the `EcuM_SetWakeupEvent` callback function. 」()

[LinTrcv167]「If development error detection for the module LinTrcv is enabled: If the addressed LIN network is not in mode `LINTRCV_TRCV_MODE_SLEEP`, the function `LinTrcv_CheckWakeup` shall raise the development error LINTRCV_E_TRCV_NOT_SLEEP and return E_NOT_OK.」()

Configuration:
See configuration parameter LinTrcvWakeUpSupport.

8.3.7 LinTrcv_SetWakeupMode

[LinTrcv009]「

Service name:	LinTrcv_SetWakeupMode	
Syntax:	`Std_ReturnType LinTrcv_SetWakeupMode( uint8 LINNetwork, LinTrcv_TrcvWakeupModeType TrcvWakupMode )`	
Service ID[hex]:	0x05	
Sync/Async:	Synchronous	
Reentrancy:	non Reentrant	
Parameters (in):	LINNetwork	LIN network to which API call has to be applied
	TrcvWakupMode	Requested transceiver wakup reason.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OKwill be returned if the transceiver state has been changed to the requested mode. E_NOT_OK will be returned, if service request is failed due to development errors or the wakeup mode is not set.
Description:	This API enables, disables and clears the notification for wakeup events on the addressed network.	

」()

[LinTrcv135]「**Enabled:** If the function `LinTrcv_SetWakeupMode` is called with TrcvWakeupMode == LINIF_TRCV_WU_ENABLE and if the LinTrcv module has a stored wakeup event pending for the addressed bus, the LinTrcv module shall execute the notification within the API call or immediately after (depending on the implementation).」()

[LinTrcv136]「**Disabled:** If the function `LinTrcv_SetWakeupMode` is called with TrcvWakeupMode == LINIF_TRCV_WU_DISABLE, then the notifications for wakeup events are disabled on the addressed network. It is required by the transceiver device and the underlying communication driver to detect the wakeup events and store it internally in order to raise the event when the wakeup notification is enabled again.」()

[LinTrcv137] ⌈**Clear:** If the function `LinTrcv_SetWakeupMode` is called with TrcvWakeupMode == LINIF_TRCV_WU_CLEAR, then a stored wakeup event is cleared on the addressed network. Clearing of wakeup events have to be used when the wake up notification is disabled to clear all stored wake up events under control of the higher layer.⌋()

[LinTrcv138] ⌈If there is no/incorrect communication to the transceiver, the function `LinTrcv_SetWakeupMode` shall return `E_NOT_OK`.⌋()

[LinTrcv139] ⌈If development error detection for the module LinTrcv is enabled: If called before the LinTrcv has been initialized, the function `LinTrcv_SetWakeupMode` shall raise development error `LINTRCV_E_UNINIT` and return `E_NOT_OK`.⌋()

[LinTrcv140] ⌈If development error detection for the module LinTrcv is enabled: If called with an invalid network number LinNetwork, the function `LinTrcv_SetWakeupMode` shall raise development error `LINTRCV_E_INVALID_LIN_NETWORK` and return `E_NOT_OK`.⌋()

[LinTrcv141] ⌈If development error detection for the module LinTrcv is enabled: If called with an invalid TrcvWakeupMode, the function `LinTrcv_SetWakeupMode` shall raise the development error LINTRCV_E_PARAM_TRCV_WAKEUP_MODE and return `E_NOT_OK`.⌋()

Caveats:
The implementation can either enable or disable interrupt source for the wake up and also it may clear wake up events from the last communication cycle. If the interrupt is level triggered, a pending interrupt is automatically stored and raised after enabling the notification again. It is very important not to lose wake up events during the disabled period.

Configuration:
The number of supported busses is statically set in the configuration phase.

8.4 Scheduled functions

This chaper lists all functions provided by the LinTrcv module and called directly by the Basic Software Module Scheduler. There are no cyclical called functions provided by Lin Transceiver Driver.

8.5 Call-back notifications

There are no callback notifications provided by Lin Transceiver Driver.

8.6 Expected Interfaces

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

8.6.1 Mandatory Interfaces

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

API function	*Description*

8.6.2 Optional Interfaces

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

API function	*Description*
Det_ReportError	Service to report development errors.
Dio_ReadChannel	Returns the value of the specified DIO channel.
Dio_ReadChannelGroup	This Service reads a subset of the adjoining bits of a port.
Dio_ReadPort	Returns the level of all channels of that port.
Dio_WriteChannel	Service to set a level of a channel.
Dio_WriteChannelGroup	Service to set a subset of the adjoining bits of a port to a specified level.
Dio_WritePort	Service to set a value of the port.
EcuM_SetWakeupEvent	Sets the wakeup event.
Icu_DisableNotification	This function disables the notification of a channel.
Icu_EnableNotification	This function enables the notification on the given channel.
Spi_GetStatus	Service returns the SPI Handler/Driver software module status.
Spi_ReadIB	Service for reading synchronously one or more data from an IB SPI Handler/Driver Channel specified by parameter.
Spi_SetupEB	Service to setup the buffers and the length of data for the EB SPI Handler/Driver Channel specified.
Spi_SyncTransmit	Service to transmit data on the SPI bus
Spi_WriteIB	Service for writing one or more data to an IB SPI Handler/Driver Channel specified by parameter.

[LinTrcv165]「 LinTrcv driver shall enable/disable ICU channels only if reference is configured for the parameter `LinTrcvIcuChannelRef`.」()

8.6.3 Configurable interfaces

There are no configurable interfaces for LIN transceiver driver.

9 Sequence diagrams

For all wakeup related sequence diagrams please refer to chapter 9 of ECU State Manager.

10 Configuration specification

In general this chapter defines configuration parameters and their clustering into containers. In order to support the specification Chapter 10.1 describes fundamentals.

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

Chapter 10.3 specifies published information of the module LinTrcv.

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 [5] (this document describes the AUTOSAR configuration methodology and the AUTOSAR configuration metamodel 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 class 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: before 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 Variants

Variants describe sets of configuration parameters. E.g.
Variant 1: only pre-compile time configuration parameters;
Variant 2: mix of pre-compile- and post build time-configuration parameters.
In one variant a parameter can only be of one configuration class.

Each Variant must have a unique name which could be referenced to in later chapters. The maximum number of allowed variants is 3.

10.1.3 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.

Configuration parameters shall be clustered into a container whenever

o the configuration parameters logically belong together
(e.g. general parameters which are valid for the entire module Lin Transceiver Driver)

o the configuration parameters need to be instantiated
(e.g. parameters of the channel specification of the Lin Transceiver Driver – those parameters must be instantiated for each Lin channel)

10.2 Containers and configuration parameters

The following chapters summarize all configuration parameters. The detailed meanings of the parameters are described in preceding chapters.

10.2.1 Variants

Three configuration variants are defined for LIN Transceiver Driver.

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

- VARIANT-LINK-TIME
  The variant VARIANT-LINK-TIME shall include all configuration options of the "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.

- VARIANT-POST-BUILD
  This configuration includes all configuration options of the "VARIANT-LINK-TIME". Additionally all parameters defined below, as post build configurable shall be configurable post build for example by flashing configuration data.

[LinTrcv017] ⌈
Only pre-compile time configuration is allowed.
Thus only "VARIANT-PRE-COMPILE" is allowed.⌋(BSW00397)

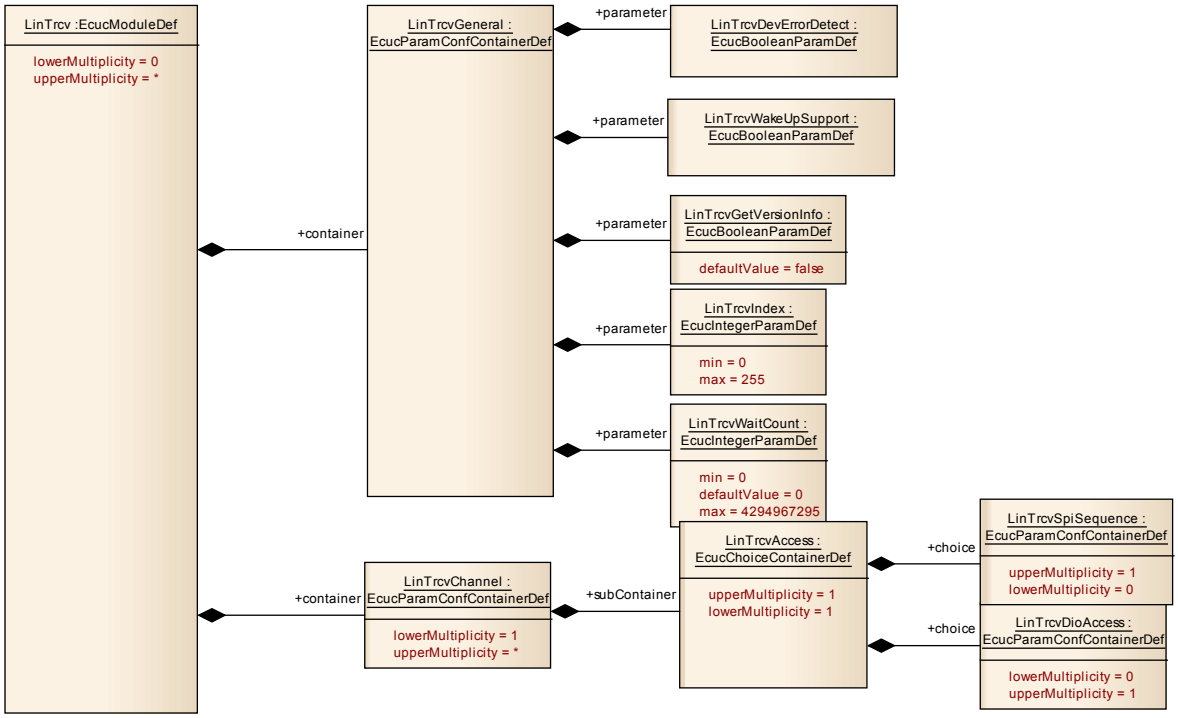
10.2.2 General configuration requirements

Configuration information is part of files LinTrcv_Cfg.h and LinTrcv_Cfg.c.

10.2.3 LinTrcv

Module Name	*LinTrcv*
Module Description	Configuration of LIN Transceiver Driver module

Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
LinTrcvChannel	1..*	Container gives LIN transceiver driver information about a single LIN transceiver channel. Any LIN transceiver driver has such LIN transceiver channels.
LinTrcvGeneral	1	Container gives LIN transceiver driver basic information.



10.2.4 LinTrcvGeneral

SWS Item	**LinTrcv090_Conf :**
Container Name	LinTrcvGeneral
Description	Container gives LIN transceiver driver basic information.
Configuration Parameters	

SWS Item	**LinTrcv001_Conf :**		
Name	LinTrcvDevErrorDetect		
Description	Switches development error detection and notification on and off. If switched on, #define LINTRCV_DEV_ERROR_DETECT ON shall be generated. If switched off, #define LINTRCV_DEV_ERROR_DETECT OFF shall be generated. Define shall be part of file LinTrcv_Cfg.h. True: Is used False: Is not used		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: module		

SWS Item	**LinTrcv003_Conf :**
Name	LinTrcvGetVersionInfo
Description	Switches version information API on and off. If switched off, function need not be present in compiled code. True: Is used False: Is not used
Multiplicity	1
Type	EcucBooleanParamDef
Default value	false

ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: module		

SWS Item	**LinTrcv153_Conf :**		
Name	LinTrcvIndex		
Description	Specifies the InstanceId of this module instance. If only one instance is present it shall have the Id 0.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: module		

SWS Item	**LinTrcv150_Conf :**		
Name	LinTrcvWaitCount		
Description	Wait count for transceiver state changes.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 4294967295		
Default value	0		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: module		

SWS Item	**LinTrcv107_Conf :**		
Name	LinTrcvWakeUpSupport		
Description	Informs whether wake up is supported or not. In case wake up is not supported by LIN transceiver hardware the setting shall be false. The wake up ability may be switched on or off for each channel of one LIN transceiver by LinTrcvWakeupSourceRef. True: Is used False: Is not used		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: module dependency: LinTrcvWakeupByBusUsed		

No Included Containers

10.2.5 LinTrcvChannel

SWS Item	**LinTrcv091_Conf :**
Container Name	LinTrcvChannel
Description	Container gives LIN transceiver driver information about a single LIN transceiver channel. Any LIN transceiver driver has such LIN transceiver

	channels.
Configuration Parameters	

SWS Item	**LinTrcv011_Conf :**		
Name	LinTrcvChannelId		
Description	Unique identifier of the LIN Transceiver Channel.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 255		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: Instance		

SWS Item	**LinTrcv004_Conf :**		
Name	LinTrcvChannelUsed		
Description	Shall the related LIN transceiver channel be used? True: Is used False Is not used		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	true		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: Instance		

SWS Item	**LinTrcv005_Conf :**		
Name	LinTrcvInitState		
Description	State of LIN transceiver after call to LinTrcv_Init.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	LINTRCV_TRCV_MODE_NORMAL		Normal operation mode
	LINTRCV_TRCV_MODE_SLEEP		Sleep operation mode
	LINTRCV_TRCV_MODE_STANDBY		Standby operation mode
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: Instance		

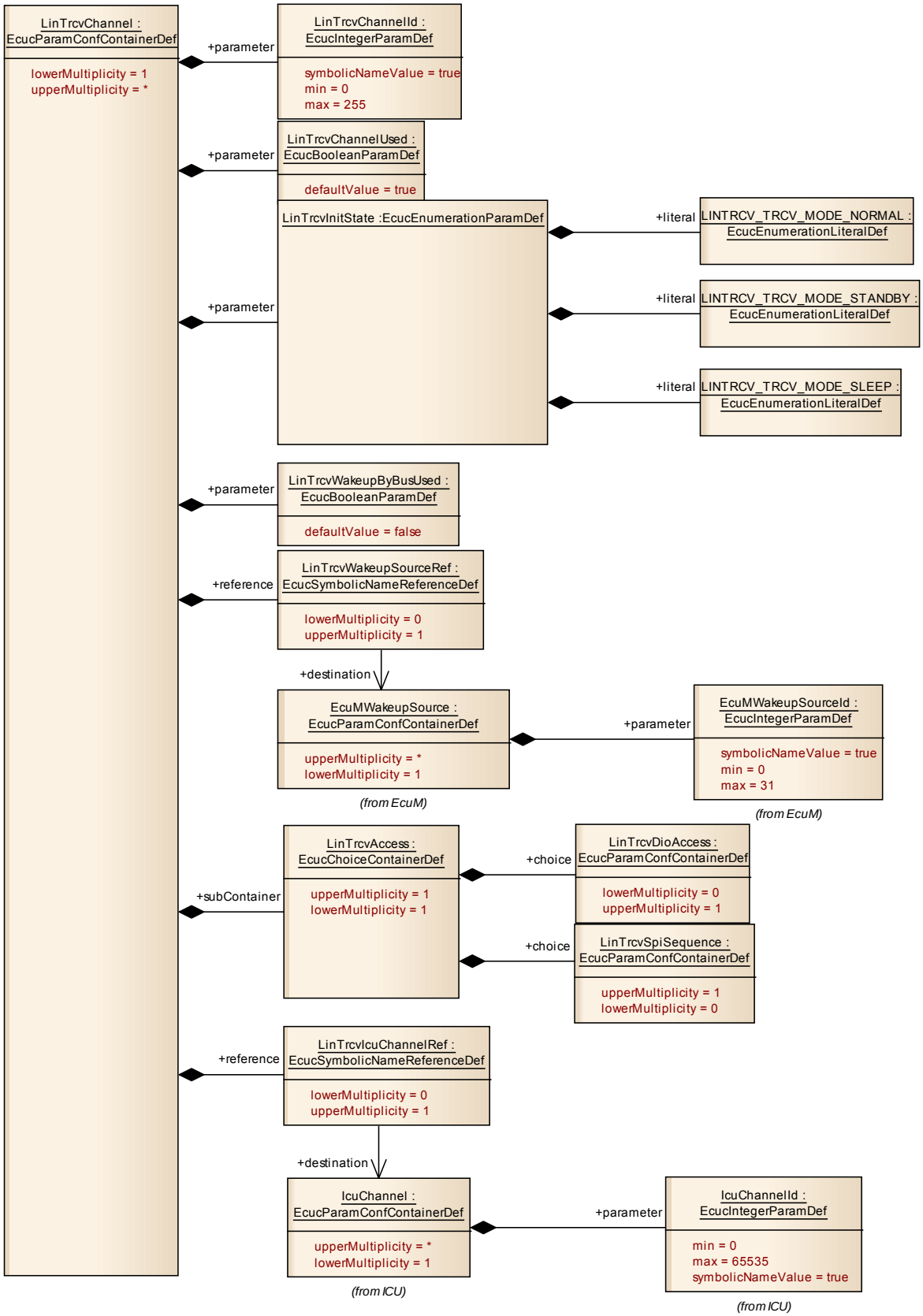
SWS Item	**LinTrcv006_Conf :**		
Name	LinTrcvWakeupByBusUsed		
Description	Is wake up by bus supported? If LIN transceiver hardware does not support wake up by bus value is always FALSE. If LIN transceiver hardware supports wake up by bus value is TRUE or FALSE depending whether it is used or not. TRUE = Is used. FALSE = Is not used.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: Instance dependency: LinTrcvWakeUpSupport		

SWS Item	**LINTrcv157_Conf :**

Name	LinTrcvIcuChannelRef		
Description	Reference to the IcuChannel to enable/disable the interrupts for wakeups.		
Multiplicity	0..1		
Type	Reference to [IcuChannel]		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: ECU		

SWS Item	**LinTrcv012_Conf :**		
Name	LinTrcvWakeupSourceRef		
Description	Reference to a wakeup source in the EcuM configuration. This reference is only needed if LinTrcvWakeupByBusUsed is true. Implementation Type: reference to EcuM_WakeupSourceType.		
Multiplicity	0..1		
Type	Reference to [EcuMWakeupSource]		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: ECU dependency: LinTrcvWakeupByBusUsed		

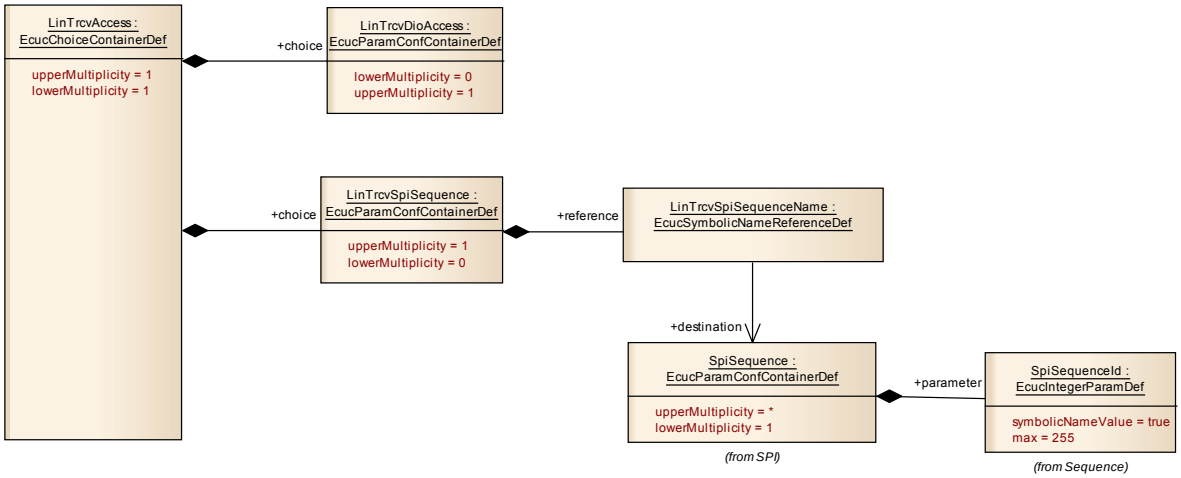
Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
LinTrcvAccess	1	Container gives LIN transceiver driver access about a single LIN transceiver channel.


LinTrcvChannel : EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = *
+parameter
LinTrcvChannelId : EcucIntegerParamDef
symbolicNameValue = true
min = 0
max = 255
+parameter
LinTrcvChannelUsed : EcucBooleanParamDef
defaultValue = true
+parameter
LinTrcvInitState :EcucEnumerationParamDef
+literal
LINTRCV_TRCV_MODE_NORMAL : EcucEnumerationLiteralDef
+literal
LINTRCV_TRCV_MODE_STANDBY : EcucEnumerationLiteralDef
+literal
LINTRCV_TRCV_MODE_SLEEP : EcucEnumerationLiteralDef
+parameter
LinTrcvWakeupByBusUsed : EcucBooleanParamDef
defaultValue = false
+reference
LinTrcvWakeupSourceRef : EcucSymbolicNameReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
+destination
EcuMWakeupSource : EcucParamConfContainerDef
upperMultiplicity = *
lowerMultiplicity = 1
(from EcuM)
+parameter
EcuMWakeupSourceId : EcucIntegerParamDef
symbolicNameValue = true
min = 0
max = 31
(from EcuM)
+subContainer
LinTrcvAccess : EcucChoiceContainerDef
upperMultiplicity = 1
lowerMultiplicity = 1
+choice
LinTrcvDioAccess : EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+choice
LinTrcvSpiSequence : EcucParamConfContainerDef
upperMultiplicity = 1
lowerMultiplicity = 0
+reference
LinTrcvIcuChannelRef : EcucSymbolicNameReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
+destination
IcuChannel : EcucParamConfContainerDef
upperMultiplicity = *
lowerMultiplicity = 1
(from ICU)
+parameter
IcuChannelId : EcucIntegerParamDef
min = 0
max = 65535
symbolicNameValue = true
(from ICU)

10.2.6 LinTrcvAccess

SWS Item	**LinTrcv154_Conf :**
Choice container Name	LinTrcvAccess
Description	Container gives LIN transceiver driver access about a single LIN transceiver channel.

Container Choices		
Container Name	***Multiplicity***	***Scope / Dependency***
LinTrcvDioAccess	0..1	Container gives LIN transceiver driver information about accessing ports and port pins. In addition relation between LIN transceiver hardware pin names and Dio port access information is given. If a LIN transceiver hardware has no Dio interface, there is no instance of this container.
LinTrcvSpiSequence	0..1	Container gives LIN transceiver driver information about one SPI sequence. One SPI sequence used by LIN transceiver driver is in exclusive use for it. No other driver is allowed to access this sequence. LIN transceiver driver may use one sequence to access n LIN transceiver hardwares chips of the same type or n sequences are used to access one single LIN transceiver hardware chip. If a LIN transceiver hardware has no SPI interface, there is no instance of this container.



10.2.7 LinTrcvDioAccess

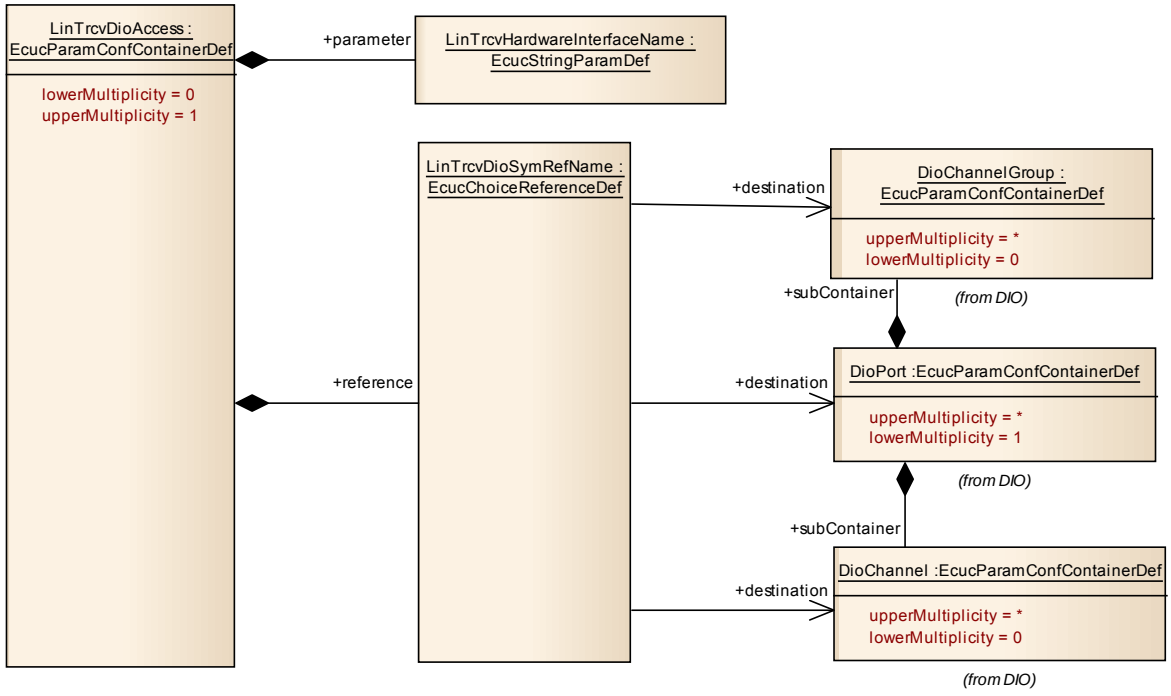
SWS Item	**LinTrcv094_Conf :**
Container Name	LinTrcvDioAccess
Description	Container gives LIN transceiver driver information about accessing ports and port pins. In addition relation between LIN transceiver hardware pin names and Dio port access information is given. If a LIN transceiver hardware has no Dio interface, there is no instance of this container.
Configuration Parameters	

SWS Item	**LinTrcv009_Conf :**
Name	LinTrcvHardwareInterfaceName
Description	LIN transceiver hardware interface name. It is typically the name of a pin. From a Dio point of view it is either a port, a single channel or a channel group. Depending on this fact either

	LINTRCV_DIO_PORT_SYMBOLIC_NAME or LINTRCV_DIO_CHANNEL_SYMBOLIC_NAME or LINTRCV_DIO_CHANNEL_GROUP_SYMBOLIC_NAME shall reference a Dio configuration. The LIN transceiver driver implementation description shall list up this name for the appropriate LIN transceiver hardware.		
Multiplicity	1		
Type	EcucStringParamDef		
Default value	--		
maxLength	--		
minLength	--		
regularExpression	--		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: Instance		

SWS Item	**LinTrcv102_Conf :**		
Name	LinTrcvDioSymRefName		
Description	Choice Reference to a DIO Port, DIO Channel or DIO Channel Group. This reference replaces the LINTRCV_DIO_PORT_SYM_NAME, LINTRCV_DIO_CHANNEL_SYM_NAME and LINTRCV_DIO_GROUP_SYM_NAME references in the Lin Trcv SWS.		
Multiplicity	1		
Type	Choice reference to [DioChannel , DioChannelGroup , DioPort]		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: instance		

No Included Containers



10.2.8 LinTrcvSpiSequence

SWS Item	**LinTrcv155_Conf :**
Container Name	LinTrcvSpiSequence{LinTransceiverSPISequences}
Description	Container gives LIN transceiver driver information about one SPI sequence. One SPI sequence used by LIN transceiver driver is in exclusive use for it. No other driver is allowed to access this sequence. LIN transceiver driver may use one sequence to access n LIN transceiver hardwares chips of the same type or n sequences are used to access one single LIN transceiver hardware chip. If a LIN transceiver hardware has no SPI interface, there is no instance of this container.
Configuration Parameters	

SWS Item	**LinTrcv156_Conf :**		
Name	LinTrcvSpiSequenceName {LINTRCV_SPI_SEQUENCE_NAME}		
Description	Reference to a Spi sequence configuration container.		
Multiplicity	1		
Type	Reference to [SpiSequence]		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: Instance dependency: SpiSequence		

No Included Containers

Published Information

[**LinTrcv161**] ⌈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].⌋()

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

11 Change history

11.1 Deleted SWS Items

SWS Item	*Rationale*
LinTrcv133	LinTrcv133 removed because it is contradicting to BSW00406
LinTrcv013	LinTrcv013 removed because cyclical function “LinTrcv_MainFunction” is not needed for the Lin Transceiver driver, the only cyclical called API is “LinTrcv_CheckWakeup” and this is called cyclically by “LinIf_CheckWakeup”.
LinTrcv142	depends on LinTrcv013 -> also removed
LinTrcv143	depends on LinTrcv013 -> also removed
LinTrcv009_Conf	API ***“LinTrcv_SetWakeupMode”*** removed.
LinTrcv135	API ***“LinTrcv_SetWakeupMode”*** removed.
LinTrcv136	API ***“LinTrcv_SetWakeupMode”*** removed.
LinTrcv137	API ***“LinTrcv_SetWakeupMode”*** removed.
LinTrcv138	API ***“LinTrcv_SetWakeupMode”*** removed.
LinTrcv139	API ***“LinTrcv_SetWakeupMode”*** removed.
LinTrcv140	API ***“LinTrcv_SetWakeupMode”*** removed.
LinTrcv141	API ***“LinTrcv_SetWakeupMode”*** removed.
LinTrcv100	Related to requirement [BSW01573] Selection of static configuration sets. This requirement is double and handled in [BSW01091].
LinTrcv007_Conf	Duplicate requirement with LinTrcv007, LinTrcv007_Conf removed because there is no requirement
LinTrcv008_Conf	Duplicate requirement with LinTrcv008, LinTrcv008_Conf removed because there is no requirement
LinTrcv009	Duplicate requirement with LinTrcv009_Conf, LinTrcv009 removed because required API is already removed

11.2 Replaced SWS Items

SWS Item	*Rationale*
BSW01501	replaced by BSW01576 due to support of LIN 2.1 Spec.
BSW01567	replaced by BSW01577 due to compatibility to LIN 2.1 Spec.
BSW01533	replaced by BSW01579 due to support of LIN 2.1 TP Spec.

11.3 Changed SWS Items

Many requirements have been changed to improve understanding without changing the technical contents.

SWS Item	*Rationale*
LinTrcv002	Duplicate requirement: The requirement - *“Selection of wakeup mode shall be done by configuration parameter LINTRCV_GENERAL_WAKE_UP_SUPPORT.”* – was replaced by **”LinTrcv107_Conf”**.
LinTrcv067	Corrected, because det.h shall only be included if development error detection is enabled.
LinTrcv109	Corrected, because transition to LINTRCV_STANDBY is only optional
LinTrcv110	Corrected, because transition to LINTRCV_SLEEP is possible either from LINTRCV_NORMAL or optional from LINTRCV_STANDBY
LinTrcv115	Corrected, because transition to LINTRCV_SLEEP is possible either from

	LINTRCV_NORMAL or optional from LINTRCV_STANDBY
LinTrcv050	Error code added for optional mode request that is not enabled
LinTrcv114	Corrected, because transition to LINTRCV_STANDBY is only optional.
LinTrcv131	Rephrased to a generic description, so changes in Std_VersionInfoType must not be adapted any more.
LinTrcv012	LinTrcv_WakeupByBus is renamed into LinTrcv_CheckWakeup to be inline with the LinIf SWS.
LinTrcv065	LinTrcv_cbk.h removed because no callback function exists.
LinTrcv067	Include structure adapted because LinTrcv_cbk.h was removed.
LinTrcv107_Conf	Configuration parameter changed because wakeup via ISR is removed from specification.
LinTrcv008_Conf	Dependency to parameter "**'LINTRCV_HARDWARE_NAME'**" removed because there is no such parameter.
LinTrcv012_Conf	Multiplicity of the parameter "**LinTrcvWakeupSourceRef**" in the container "LinTrcvChannel" is changed from "**1**" to "**0..1**", since this parameter need not be configured when the parameter "**LinTrcvWakeupByBusUsed**" is configured as false.
LinTrcv011_Conf	The range for configuration parameter LinTrcvChannelId and LINNetwork is changed from 0..18446744073709551615 to 0..255 since type used in corresponding APIs is of type uint8.
LinTrcv102_Conf	ModuleName.h replaced by LinTrcv.h
LinTrcv007_Conf	Description improved.
LinTrcv008_Conf	Description improved.
LinTrcv107_Conf LinTrcv006_Conf	"LinTrcvGeneralWakeUpSupport" renamed to "LinTrcvWakeUpSupport".
LinTrcv002_Conf	As "_Conf" requirements are only for chapter 10 this is replaced by **LinTrcv002**

11.4 Added SWS Items

Many requirements have been added because multiple requirements were described in one and these multiple requirements where separated into single requirements.

SWS Item	***Rationale***
LinTrcv152	The implementer shall have the possibility to detect additional implementation specific errors, too.
LinTrcv107_Conf	Replaces the old requirement ***"LinTrcv002_Conf"***.
LinTrcv108	Give explicit id to requirement out of API ***"LinTrcv_SetOpMode"***.
LinTrcv109	Give explicit id to requirement out of API ***"LinTrcv_SetOpMode"***.
LinTrcv110	Give explicit id to requirement out of API ***"LinTrcv_SetOpMode"***.
LinTrcv111	Give explicit id to requirement out of API ***"LinTrcv_SetOpMode"***.
LinTrcv112	Give explicit id to requirement out of API ***"LinTrcv_SetOpMode"***.
LinTrcv113	Give explicit id to requirement out of API ***"LinTrcv_SetOpMode"***.
LinTrcv114	Give explicit id to requirement out of API ***"LinTrcv_SetOpMode"***.
LinTrcv115	Give explicit id to requirement out of API ***"LinTrcv_SetOpMode"***.
LinTrcv116	Give explicit id to requirement out of API ***"LinTrcv_SetOpMode"***.
LinTrcv117	Give explicit id to requirement out of API ***"LinTrcv_SetOpMode"***.
LinTrcv118	Give explicit id to requirement out of API ***"LinTrcv_SetOpMode"***.
LinTrcv119	Give explicit id to requirement out of API ***"LinTrcv_Init".***
LinTrcv120	Give explicit id to requirement out of API ***"LinTrcv_Init".***
LinTrcv121	Give explicit id to requirement out of API ***"LinTrcv_GetOpMode"***
LinTrcv122	Give explicit id to requirement out of API ***"LinTrcv_GetOpMode"***
LinTrcv123	Give explicit id to requirement out of API ***"LinTrcv_GetOpMode"***
LinTrcv124	Give explicit id to requirement out of API ***"LinTrcv_GetOpMode"***
LinTrcv125	Give explicit id to requirement out of API ***"LinTrcv_GetOpMode"***
LinTrcv126	Give explicit id to requirement out of API ***"LinTrcv_GetBusWuReason"***

LinTrcv127	Give explicit id to requirement out of API ***"LinTrcv_GetBusWuReason"***
LinTrcv128	Give explicit id to requirement out of API ***"LinTrcv_GetBusWuReason"***
LinTrcv129	Give explicit id to requirement out of API ***"LinTrcv_GetBusWuReason"***
LinTrcv130	Give explicit id to requirement out of API ***"LinTrcv_GetBusWuReason"***
LinTrcv131	Give explicit id to requirement out of API ***"LinTrcv_GetVersionInfo"***
LinTrcv132	Give explicit id to requirement out of API ***"LinTrcv_GetVersionInfo"***
LinTrcv133	Give explicit id to requirement out of API ***"LinTrcv_GetVersionInfo"***
LinTrcv134	Give explicit id to requirement out of API ***"LinTrcv_GetVersionInfo"***
LinTrcv135	Give explicit id to requirement out of API ***"LinTrcv_SetWakeupMode"***
LinTrcv136	Give explicit id to requirement out of API ***"LinTrcv_SetWakeupMode"***
LinTrcv137	Give explicit id to requirement out of API ***"LinTrcv_SetWakeupMode"***
LinTrcv138	Give explicit id to requirement out of API ***"LinTrcv_SetWakeupMode"***
LinTrcv139	Give explicit id to requirement out of API ***"LinTrcv_SetWakeupMode"***
LinTrcv140	Give explicit id to requirement out of API ***"LinTrcv_SetWakeupMode"***
LinTrcv141	Give explicit id to requirement out of API ***"LinTrcv_SetWakeupMode"***
LinTrcv142	Give explicit id to requirement out of API ***"LinTrcv_MainFunction"***
LinTrcv143	Give explicit id to requirement out of API ***"LinTrcv_MainFunction"***
LinTrcv144	Give explicit id to requirement out of API ***"LinTrcv_WakeupByBus"***
LinTrcv145	Give explicit id to requirement out of API ***"LinTrcv_WakeupByBus"***
LinTrcv146	Classify transition from NOT_ACTIVE to LINTRCV_STANDBY as optional.
LinTrcv147	Added, because transition to LINTRCV_NORMAL is possible either from LINTRCV_SLEEP or optional from LINTRCV_STANDBY
LinTrcv148	Added, because transition to LINTRCV_STANDBY is optional possible either from LINTRCV_SLEEP or LINTRCV_NORMAL
LinTrcv149	Corrected, because transition to LINTRCV_NORMAL is possible either from LINTRCV_SLEEP or optional from LINTRCV_STANDBY
LinTrcv150_Conf	Missing parameter "***LinTrcvWaitTime***" for transceiver wait states added to container "LinTrcvGeneral".
LinTrcv151_Conf	LinTrcv011_Conf was duplicated. Now the second description for LinTrcv011_Conf was replaced by LinTrcv151_Conf.
LinTrcv001	Duplicate requirement separated into LinTrcv001_Conf and LinTrcv001
LinTrcv005	Duplicate requirement separated into LinTrcv005_Conf and LinTrcv005
LinTrcv012	Duplicate requirement separated into LinTrcv012_Conf and LinTrcv012
LinTrcv102	Duplicate requirement separated into LinTrcv102_Conf and LinTrcv102
LinTrcv106	Duplicate requirement separated into LinTrcv106 and LinTrcv152
LinTrcv107	Duplicate requirement separated into LinTrcv107_Conf and LinTrcv107
LinTrcv009	API "LinTrcv_SetWakeupMode" added again.
LinTrcv135	API "LinTrcv_SetWakeupMode" added again.
LinTrcv136	API "LinTrcv_SetWakeupMode" added again.
LinTrcv137	API "LinTrcv_SetWakeupMode" added again.
LinTrcv138	API "LinTrcv_SetWakeupMode" added again.
LinTrcv139	API "LinTrcv_SetWakeupMode" added again.
LinTrcv140	API "LinTrcv_SetWakeupMode" added again.
LinTrcv141	API "LinTrcv_SetWakeupMode" added again.
LinTrcv001_PI	Rework of Published Information
LinTrcv157	A mode request of the current mode is allowed and shall not lead to an error even if DET is enabled.
LinTrcv158	Version checking
LinTrcv159	Duplicate Reqirement ID LinTrcv107.
LinTrcv160	If a wake-up event has been detected, the wake-up shall directly be reported to the EcuM via EcuM_SetWakeupEvent call-back function.
LinTrcv161	Icu channel notification / wakeup issue 47372
LinTrcv162	Icu channel notification / wakeup issue 47372
LinTrcv163	Icu channel notification / wakeup issue 47372
LinTrcv157_Conf	Icu channel notification / wakeup issue 47372
LinTrcv165	Icu channel notification / wakeup issue 47372
LinTrcv166	50983

LinTrcv167	50983

12 Not applicable requirements

[LinTrcv999]「These requirements are not applicable to this specification.」
(BSW00304, BSW00305, BSW00306, BSW00307, BSW00308, BSW00309, BSW00312, BSW00321, BSW00325, BSW00326, BSW00328, BSW00330, BSW00331, BSW00333, BSW00334, BSW00335, BSW00336, BSW00341, BSW00342, BSW00344, BSW00355, BSW00359, BSW00360, BSW00378, BSW00383, BSW00384, BSW00398, BSW00399, BSW00400, BSW00401, BSW00404, BSW00405, BSW00410, BSW00416, BSW00417, BSW00420, BSW00422, BSW00423, BSW00426, BSW00427, BSW00429, BSW00431, BSW00432, BSW00433, BSW00434, BSW005, BSW006, BSW007, BSW009, BSW010, BSW159, BSW161, BSW164, BSW167, BSW168, BSW170, BSW01576, BSW01504, BSW01522, BSW01560, BSW01577, BSW01551, BSW01568, BSW01569, BSW01564, BSW01546, BSW01549, BSW01571, BSW01515, BSW01502, BSW01558, BSW01527, BSW01523, BSW01577, BSW01553, BSW01552, BSW01503, BSW01555, BSW01547, BSW01572, BSW01556, BSW01579, BSW01540, BSW01545, BSW01534, BSW01574, BSW01539, BSW01544, BSW01115)