

| | |
|-----------------------------------|---|
| Document Title | Specification of LIN Transceiver Driver |
| Document Owner | AUTOSAR |
| Document Responsibility | AUTOSAR |
| Document Identification No | 257 |

| | |
|---------------------------------|------------------|
| Document Status | published |
| Part of AUTOSAR Standard | Classic Platform |
| Part of Standard Release | R23-11 |

| Document Change History | | | |
|--------------------------------|----------------|----------------------------|--|
| Date | Release | Changed by | Description |
| 2023-11-23 | R23-11 | AUTOSAR Release Management | <ul style="list-style-type: none"> • Editorial changes |
| 2022-11-24 | R22-11 | AUTOSAR Release Management | <ul style="list-style-type: none"> • Updated LinTrcv_CheckWakeup return values |
| 2021-11-25 | R21-11 | AUTOSAR Release Management | <ul style="list-style-type: none"> • Cleaned error codes |
| 2020-11-30 | R20-11 | AUTOSAR Release Management | <ul style="list-style-type: none"> • Cleaned error classification |
| 2019-11-28 | R19-11 | AUTOSAR Release Management | <ul style="list-style-type: none"> • MCALMulticoreDistribution (CONC_639) • Changed Document Status from Final to published |
| 2018-10-31 | 4.4.0 | AUTOSAR Release Management | <ul style="list-style-type: none"> • Minor corrections / clarifications / editorial changes; For details please refer to the ChangeDocumentation |
| 2017-12-08 | 4.3.1 | AUTOSAR Release Management | <ul style="list-style-type: none"> • Restricted initial state to LINTRCV_TRCV_MODE_SLEEP • Editorial changes |
| 2016-11-30 | 4.3.0 | AUTOSAR Release Management | <ul style="list-style-type: none"> • Change in GetVersionInfo API • minor corrections / editorial changes; For details please refer to the ChangeDocumentation |





| | | | |
|------------|-------|----------------------------|---|
| 2015-07-31 | 4.2.2 | AUTOSAR Release Management | <ul style="list-style-type: none"> • Development Error Tracer replaced with Default Error Tracer • Standardized the initialization function |
| 2014-10-31 | 4.2.1 | AUTOSAR Release Management | <ul style="list-style-type: none"> • Supports Time service for transceiver state change waits |
| 2014-03-31 | 4.1.3 | AUTOSAR Release Management | <ul style="list-style-type: none"> • Editorial Changes |
| 2013-10-31 | 4.1.2 | AUTOSAR Release Management | <ul style="list-style-type: none"> • Added intimation to LinIf for wakeup by transceiver • Modified header file structure and mandatory interfaces • Removed [SWS_LinTrcv_00160] • Editorial changes • Removed chapter(s) on change documentation |
| 2013-03-15 | 4.1.1 | AUTOSAR Administration | <ul style="list-style-type: none"> • Removed [SWS_LinTrcv_00141] and [SWS_LinTrcv_00118] • DET Errors LINTRCV_E_PARAM_TRCV_WAKEUP_MODE and LINTRCV_E_PARAM_TRCV_OPMODE removed from [SWS_LinTrcv_00050] • LINIF_TRCV_WU changed to LINTRCV_WUMODE for LINIF_TRCV_WU_ENABLE, LINIF_TRCV_WU_DISABLE and LINIF_TRCV_WU_CLEAR • Rework of configuration parameter LinTrcvDioAccess and changed scope of configuration parameters to "local" or "ECU" • Rework of header file structure • Removal of specification items covered by the new SWS BSW General • Formal rework |



△

| | | | |
|------------|-------|------------------------|--|
| 2011-12-22 | 4.0.3 | AUTOSAR Administration | <ul style="list-style-type: none"> • Update of wake-up validation (power-up) • Several minor corrections (typos and wordings) |
| 2010-09-30 | 3.1.5 | AUTOSAR Administration | <ul style="list-style-type: none"> • Literals changed names: • 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 |
| 2010-02-02 | 3.1.4 | AUTOSAR Administration | <ul style="list-style-type: none"> • Initial release |

Disclaimer

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

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

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

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

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

Contents

| | | |
|-------|---|----|
| 1 | Introduction and functional overview | 7 |
| 1.1 | Goal of LIN transceiver driver | 7 |
| 1.2 | Explicitly uncovered LIN transceiver functionality | 7 |
| 2 | Acronyms and Abbreviations | 8 |
| 3 | Related documentation | 9 |
| 3.1 | Input documents & related standards and norms | 9 |
| 3.2 | Related specification | 9 |
| 4 | Constraints and assumptions | 10 |
| 4.1 | Limitations | 10 |
| 4.2 | Applicability to car domains | 10 |
| 5 | Dependencies to other modules | 11 |
| 5.1 | File structure | 11 |
| 5.1.1 | Naming convention for transceiver driver implementation | 11 |
| 5.1.2 | Code file structure | 11 |
| 5.1.3 | Header file structure | 11 |
| 6 | Requirements Tracing | 12 |
| 7 | Functional specification | 14 |
| 7.1 | LIN transceiver driver operation modes | 14 |
| 7.2 | LIN transceiver hardware operation modes | 15 |
| 7.3 | LIN transceiver wakeup types | 15 |
| 7.4 | LIN transceiver wakeup modes | 16 |
| 7.5 | Error Classification | 17 |
| 7.5.1 | Development Errors | 17 |
| 7.5.2 | Runtime Errors | 17 |
| 7.5.3 | Transient Faults | 17 |
| 7.5.4 | Production Errors | 18 |
| 7.5.5 | Extended Production Errors | 18 |
| 7.6 | Preconditions for driver initialization | 18 |
| 7.7 | Instance concept | 18 |
| 7.8 | Wait states | 18 |
| 8 | API specification | 20 |
| 8.1 | Imported types | 20 |
| 8.2 | Type definitions | 20 |
| 8.2.1 | LinTrcv_ConfigType | 20 |
| 8.2.2 | LinTrcv_TrvcModeType | 21 |
| 8.2.3 | LinTrcv_TrvcWakeupModeType | 21 |
| 8.2.4 | LinTrcv_TrvcWakeupReasonType | 21 |
| 8.3 | Function definitions | 22 |

| | | |
|--------|---|----|
| 8.3.1 | LinTrcv_Init | 22 |
| 8.3.2 | LinTrcv_SetOpMode | 23 |
| 8.3.3 | LinTrcv_GetOpMode | 25 |
| 8.3.4 | LinTrcv_GetBusWuReason | 26 |
| 8.3.5 | LinTrcv_GetVersionInfo | 28 |
| 8.3.6 | LinTrcv_CheckWakeup | 28 |
| 8.3.7 | LinTrcv_SetWakeupMode | 29 |
| 8.4 | Callback notifications | 30 |
| 8.5 | Scheduled functions | 31 |
| 8.6 | Expected interfaces | 31 |
| 8.6.1 | Mandatory interfaces | 31 |
| 8.6.2 | Optional interfaces | 31 |
| 8.6.3 | Configurable interfaces | 32 |
| 9 | Sequence diagrams | 33 |
| 10 | Configuration specification | 34 |
| 10.1 | How to read this chapter | 34 |
| 10.2 | Containers and configuration parameters | 34 |
| 10.2.1 | Variants | 34 |
| 10.2.2 | LinTrcv | 34 |
| 10.2.3 | LinTrcvGeneral | 36 |
| 10.2.4 | LinTrcvChannel | 38 |
| 10.2.5 | LinTrcvAccess | 42 |
| 10.2.6 | LinTrcvDioAccess | 43 |
| 10.2.7 | LinTrcvDioChannelAccess | 44 |
| 10.2.8 | LinTrcvSpiSequence | 45 |
| 10.3 | Published Information | 46 |
| A | Not applicable requirements | 47 |

1 Introduction and functional overview

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.

[SWS_LinTrcv_00042] [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.] ([SRS_BSW_00162](#))

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

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

| Abbreviation / Acronym: | 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 |
| Det | Default 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. |

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

3 Related documentation

3.1 Input documents & related standards and norms

- [1] Glossary
AUTOSAR_FO_TR_Glossary
- [2] General Specification of Basic Software Modules
AUTOSAR_CP_SWS_BSWGeneral
- [3] General Requirements on Basic Software Modules
AUTOSAR_CP_SRS_BSWGeneral
- [4] Requirements on LIN
AUTOSAR_CP_SRS_LIN
- [5] Requirements on CAN
AUTOSAR_CP_SRS_CAN

3.2 Related specification

AUTOSAR provides a General Specification on Basic Software modules [2, SWS BSW General], which is also valid for LIN Transceiver Driver.

Thus, the specification SWS BSW General shall be considered as additional and required specification for LIN Transceiver Driver.

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

| <i>Module</i> | <i>Dependencies</i> |
|---------------|--|
| 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. |
| 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

[SWS_LinTrcv_00070] [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.] ([SRS_BSW_00347](#))

5.1.2 Code file structure

For details, refer to the section "Code file structure" of the [2].

5.1.3 Header file structure

[SWS_LinTrcv_00067] [The include file structure shall be as follows:

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 Dio.h (DIO APIs needed to access Transceiver pins)

LinTrcv.c shall include Icu.h (if ICU APIs needed to perform LIN transceiver hardware interrupts)

LinTrcv.c shall include Spi.h (if the LIN bus transceiver driver use drivers for Spi to control the LIN bus transceiver hardware)

LinTrcv.c shall include Tm.h (needed for wait states for changing transceiver operation modes)] ([SRS_BSW_00301](#), [SRS_BSW_00409](#))

6 Requirements Tracing

The following tables reference the requirements specified in [3], [4], [5] and links to the fulfillment of these. Please note that if column "Satisfied by" is empty for a specific requirement this means that this requirement is not fulfilled by this document.

| Requirement | Description | Satisfied by |
|-----------------|--|---|
| [SRS_BSW_00101] | The Basic Software Module shall be able to initialize variables and hardware in a separate initialization function | [SWS_LinTrcv_00001] |
| [SRS_BSW_00162] | The AUTOSAR Basic Software shall provide a hardware abstraction layer | [SWS_LinTrcv_00042] |
| [SRS_BSW_00301] | All AUTOSAR Basic Software Modules shall only import the necessary information | [SWS_LinTrcv_00067] |
| [SRS_BSW_00310] | API naming convention | [SWS_LinTrcv_00001] [SWS_LinTrcv_00002] [SWS_LinTrcv_00005] [SWS_LinTrcv_00007] [SWS_LinTrcv_00008] [SWS_LinTrcv_00012] |
| [SRS_BSW_00327] | Error values naming convention | [SWS_LinTrcv_00050] |
| [SRS_BSW_00347] | A Naming separation of different instances of BSW drivers shall be in place | [SWS_LinTrcv_00016] [SWS_LinTrcv_00070] |
| [SRS_BSW_00357] | For success/failure of an API call a standard return type shall be defined | [SWS_LinTrcv_00002] |
| [SRS_BSW_00358] | The return type of init() functions implemented by AUTOSAR Basic Software Modules shall be void | [SWS_LinTrcv_00001] |
| [SRS_BSW_00369] | All AUTOSAR Basic Software Modules shall not return specific development error codes via the API | [SWS_LinTrcv_00002] [SWS_LinTrcv_00005] [SWS_LinTrcv_00007] [SWS_LinTrcv_00008] [SWS_LinTrcv_00012] |
| [SRS_BSW_00375] | Basic Software Modules shall report wake-up reasons | [SWS_LinTrcv_00012] |
| [SRS_BSW_00377] | A Basic Software Module can return a module specific types | [SWS_LinTrcv_00005] [SWS_LinTrcv_00007] |
| [SRS_BSW_00385] | List possible error notifications | [SWS_LinTrcv_00050] |
| [SRS_BSW_00386] | The BSW shall specify the configuration and conditions for detecting an error | [SWS_LinTrcv_00050] |
| [SRS_BSW_00406] | A static status variable denoting if a BSW module is initialized shall be initialized with value 0 before any APIs of the BSW module is called | [SWS_LinTrcv_00002] [SWS_LinTrcv_00007] [SWS_LinTrcv_00008] [SWS_LinTrcv_00012] |
| [SRS_BSW_00407] | Each BSW module shall provide a function to read out the version information of a dedicated module implementation | [SWS_LinTrcv_00008] |
| [SRS_BSW_00409] | All production code error ID symbols are defined by the Dem module and shall be retrieved by the other BSW modules from Dem configuration | [SWS_LinTrcv_00067] |
| [SRS_BSW_00413] | An index-based accessing of the instances of BSW modules shall be done | [SWS_LinTrcv_00016] |





| Requirement | Description | Satisfied by |
|-----------------|---|---|
| [SRS_BSW_00414] | Init functions shall have a pointer to a configuration structure as single parameter | [SWS_LinTrcv_00001] [SWS_LinTrcv_00172] [SWS_LinTrcv_00173] |
| [SRS_Can_01096] | The bus transceiver driver shall provide an API to initialize the driver internally | [SWS_LinTrcv_00001] |
| [SRS_Can_01097] | CAN Bus Transceiver driver API shall be synchronous | [SWS_LinTrcv_00001] [SWS_LinTrcv_00002] [SWS_LinTrcv_00005] [SWS_LinTrcv_00007] [SWS_LinTrcv_00012] |
| [SRS_Can_01098] | The bus transceiver driver shall support an API to send the addressed transceiver into its Standby mode | [SWS_LinTrcv_00002] [SWS_LinTrcv_00055] |
| [SRS_Can_01099] | The bus transceiver driver shall support an API to send the addressed transceiver into its Sleep mode | [SWS_LinTrcv_00002] [SWS_LinTrcv_00055] |
| [SRS_Can_01100] | The bus transceiver driver shall support an API to send the addressed transceiver into its Normal mode | [SWS_LinTrcv_00002] [SWS_LinTrcv_00055] |
| [SRS_Can_01101] | The bus transceiver driver shall support an API to read out the current operation mode of the transceiver of a specified bus within the ECU | [SWS_LinTrcv_00005] |
| [SRS_Can_01103] | The bus transceiver driver shall support an API to read out the reason of the last wakeup of a specified bus within the ECU | [SWS_LinTrcv_00007] |
| [SRS_Lin_01514] | The LIN Interface shall inform an upper layer about wake-up events | [SWS_LinTrcv_00066] |
| [SRS_Lin_01524] | The LIN Driver shall be able to put the LIN hardware to a reduced power operation mode if needed | [SWS_LinTrcv_00002] [SWS_LinTrcv_00055] |
| [SRS_Lin_01563] | The LIN Driver shall provide a notification for wake-up events | [SWS_LinTrcv_00066] |
| [SRS_Lin_01566] | Transition to sleep-mode shall be handled | [SWS_LinTrcv_00002] [SWS_LinTrcv_00055] |
| [SRS_Lin_01580] | The LIN Transceiver Driver shall support separate configuration parameters per bus | [SWS_LinTrcv_00074] [SWS_LinTrcv_00075] |

Table 6.1: RequirementsTracing

7 Functional specification

7.1 LIN transceiver driver operation modes

[SWS_LinTrcv_00055] [The LIN transceiver driver operation modes are described in the state diagram below.] ([SRS_Lin_01566](#), [SRS_Lin_01524](#), [SRS_Can_01098](#), [SRS_Can_01099](#), [SRS_Can_01100](#))

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.

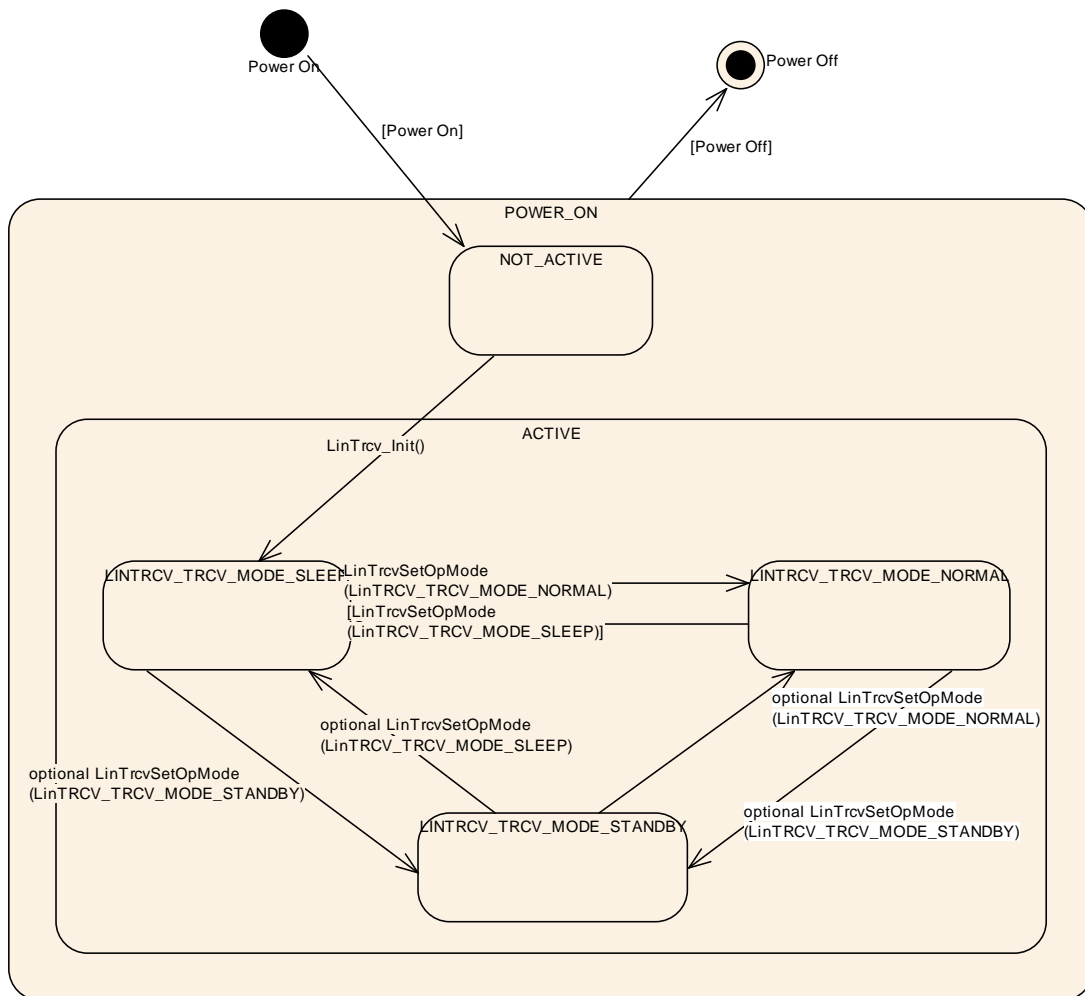


Figure 7.1: LIN Transceiver Operation Modes

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.

| 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 <code>LinTrcv_Init</code> was called. It carries LIN transceiver driver to active state. LIN transceiver driver enters state <code>LINTRCV_TRCV_MODE_SLEEP</code> . |
| <code>LINTRCV_TRCV_MODE_NORMAL</code> | Full bus communication. If LIN transceiver hardware controls MCU power supply, MCU is fully powered. The LIN transceiver driver detects no further wakeup information. |
| <code>LINTRCV_TRCV_MODE_STANDBY</code> | 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. |
| <code>LINTRCV_TRCV_MODE_SLEEP</code> | 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

[SWS_LinTrcv_00066] [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.] ([SRS_Lin_01514](#), [SRS_Lin_01563](#))

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.

[SWS_LinTrcv_00074] [Selection of wakeup mode shall be done by configuration parameter `LinTrcvWakeUpSupport`.] ([SRS_Lin_01580](#))

[SWS_LinTrcv_00075] [Support of wakeup shall be switched on and off for each LIN transceiver channel individually by configuration parameter `LinTrcvWakeupByBusUsed`.] ([SRS_Lin_01580](#))

[SWS_LinTrcv_00161] [LinTrcv driver shall use the following APIs provided by ICU driver, to enable and disable the wakeup event notification:

- `Icu_EnableNotification`
- `Icu_DisableNotification`

]()

[SWS_LinTrcv_00162] [LinTrcv driver shall enable the ICU channels when the transceiver transmits to standby mode (LINTRCV_STANDBY)]()

[SWS_LinTrcv_00163] [LinTrcv driver shall disable the ICU channels when the transceiver transmits to Normal mode (LINTRCV_NORMAL)]()

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

7.5 Error Classification

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

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

7.5.1 Development Errors

There are no development errors.

[SWS_LinTrcv_00050] Definiton of development errors in module LinTrcv [

| <i>Type of error</i> | <i>Related error code</i> | <i>Error value</i> |
|---|-------------------------------|--------------------|
| API called with wrong parameter for LIN network | LINTRCV_E_INVALID_LIN_NETWORK | 0x01 |
| API called with null pointer parameter | LINTRCV_E_PARAM_POINTER | 0x02 |
| API service used without initialization | LINTRCV_E_UNINIT | 0x11 |
| API service called in wrong transceiver operation mode | LINTRCV_E_TRCV_NOT_SLEEP | 0x21 |
| API service called in wrong transceiver operation mode | LINTRCV_E_TRCV_NOT_NORMAL | 0x22 |
| API service called with invalid mode because optional transition is not enabled | LINTRCV_E_INVALID_TRCV_OPMODE | 0x25 |

]([SRS_BSW_00327](#), [SRS_BSW_00385](#), [SRS_BSW_00386](#))

7.5.2 Runtime Errors

There are no runtime errors.

7.5.3 Transient Faults

There are no transient faults.

7.5.4 Production Errors

There are no production errors.

7.5.5 Extended Production Errors

There are no extended production errors.

7.6 Preconditions for driver initialization

[SWS_LinTrcv_00099] [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

- 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.
- 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.
- 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. 50 μ s before changing it again to reach a specific state, e.g. sleep.

7.7 Instance concept

[SWS_LinTrcv_00016] [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.] ([SRS_BSW_00347](#), [SRS_BSW_00413](#))

7.8 Wait states

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

[SWS_LinTrcv_00171] [The LIN Transceiver Driver shall use the Time service Tm_BusyWait1us16bit to realize the wait time for transceiver state changes.] ()

8 API specification

8.1 Imported types

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

[SWS_LinTrcv_91001] Definition of imported datatypes of module LinTrcv [

| <i>Module</i> | <i>Header File</i> | <i>Imported Type</i> |
|---------------|--------------------|-----------------------|
| Dio | Dio.h | Dio_ChannelGroupType |
| | Dio.h | Dio_ChannelType |
| | Dio.h | Dio_LevelType |
| | Dio.h | Dio_PortLevelType |
| | Dio.h | Dio_PortType |
| EcuM | EcuM.h | EcuM_WakeupSourceType |
| Icu | Icu.h | Icu_ChannelType |
| Spi | Spi.h | Spi_ChannelType |
| | Spi.h | Spi_DataBufferType |
| | Spi.h | Spi_NumberOfDataType |
| | Spi.h | Spi_SequenceType |
| | Spi.h | Spi_StatusType |
| Std | Std_Types.h | Std_ReturnType |
| | Std_Types.h | Std_VersionInfoType |

]()

8.2 Type definitions

8.2.1 LinTrcv_ConfigType

[SWS_LinTrcv_00172] Definition of datatype LinTrcv_ConfigType [

| | | |
|----------------------|---|---|
| Name | LinTrcv_ConfigType | |
| Kind | Structure | |
| Elements | implementation specific | |
| | Type | – |
| | Comment | – |
| Description | Configuration data structure of the LinTrcv module. | |
| Available via | LinTrcv.h | |

] ([SRS_BSW_00414](#))

8.2.2 LinTrcv_TrcvModeType

[SWS_LinTrcv_00168] Definition of datatype LinTrcv_TrcvModeType [

| | | | |
|----------------------|---|---|--------------------------|
| Name | LinTrcv_TrcvModeType | | |
| Kind | 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 | | |
| Available via | Lin_GeneralTypes.h | | |

]()

8.2.3 LinTrcv_TrcvWakeupModeType

[SWS_LinTrcv_00169] Definition of datatype LinTrcv_TrcvWakeupModeType [

| | | | |
|----------------------|--|---|--|
| Name | LinTrcv_TrcvWakeupModeType | | |
| Kind | 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. | | |
| Available via | Lin_GeneralTypes.h | | |

]()

8.2.4 LinTrcv_TrcvWakeupReasonType

[SWS_LinTrcv_00170] Definition of datatype LinTrcv_TrcvWakeupReasonType [

| | | | |
|--------------|------------------------------|---|---|
| Name | LinTrcv_TrcvWakeupReasonType | | |
| Kind | Enumeration | | |
| Range | LINTRCV_WU_ERROR | – | Due to an error wake up reason was not detected. |
| | 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. | | |
| Available via | Lin_GeneralTypes.h | | |

]()

8.3 Function definitions

8.3.1 LinTrcv_Init

[SWS_LinTrcv_00001] Definition of API function LinTrcv_Init [

| | | |
|---------------------------|--|--|
| Service Name | LinTrcv_Init | |
| Syntax | <pre>void LinTrcv_Init (const LinTrcv_ConfigType* ConfigPtr)</pre> | |
| Service ID [hex] | 0x00 | |
| Sync/Async | Synchronous | |
| Reentrancy | Non Reentrant | |
| Parameters (in) | ConfigPtr | Pointer to the selected configuration set. |
| Parameters (inout) | None | |
| Parameters (out) | None | |
| Return value | None | |
| Description | Initializes the Lin Transceiver Driver module. | |
| Available via | LinTrcv.h | |

] ([SRS_BSW_00310](#), [SRS_BSW_00358](#), [SRS_BSW_00414](#), [SRS_BSW_00101](#), [SRS_Can_01096](#), [SRS_Can_01097](#))

[SWS_LinTrcv_00173] [The Configuration pointer `ConfigPtr` shall always have a `NULL_PTR` value] ([SRS_BSW_00414](#))

The Configuration pointer `ConfigPtr` is currently not used and shall therefore be set `NULL_PTR` value.

[SWS_LinTrcv_00119] [The function `LinTrcv_Init` shall set the LIN transceiver hardware to the state `LINTRCV_TRCV_MODE_SLEEP`.]()

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

[SWS_LinTrcv_00002] Definition of API function LinTrcv_SetOpMode [

| | | |
|---------------------------|---|--|
| Service Name | LinTrcv_SetOpMode | |
| Syntax | <pre>Std_ReturnType LinTrcv_SetOpMode (uint8 LinNetwork, LinTrcv_TrcvModeType OpMode)</pre> | |
| Service ID [hex] | 0x01 | |
| Sync/Async | Synchronous | |
| Reentrancy | Non Reentrant | |
| Parameters (in) | LinNetwork | LIN network to which 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 Op Mode. | |
| Available via | LinTrcv.h | |

]([SRS_BSW_00310](#), [SRS_BSW_00357](#), [SRS_BSW_00369](#), [SRS_BSW_00406](#), [SRS_Lin_01566](#), [SRS_Lin_01524](#), [SRS_Can_01097](#), [SRS_Can_01098](#), [SRS_Can_01099](#), [SRS_Can_01100](#))

[SWS_LinTrcv_00108] [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.]()

[SWS_LinTrcv_00109] [The function [LinTrcv_SetOpMode](#) shall switch the internal state of channel [LinNetwork](#) to the value of LINTRCV_TRCV_MODE_STANDBY if one of the following conditions is fulfilled:

- 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.]()

[SWS_LinTrcv_00110] [The function `LinTrcv_SetOpMode` shall switch the internal state of channel `LinNetwork` to the value of `LINTRCV_TRCV_MODE_SLEEP` if one of the following conditions is fulfilled:

- 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.]()

[SWS_LinTrcv_00147] [The function `LinTrcv_SetOpMode` shall switch the internal state of channel `LinNetwork` to the value of `LINTRCV_TRCV_MODE_NORMAL` if one of the following conditions is fulfilled:

- 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.]()

[SWS_LinTrcv_00111] [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.]()

[SWS_LinTrcv_00112] [If the requested mode is not supported by the underlying transceiver hardware, the function `LinTrcv_SetOpMode` shall return `E_NOT_OK`.]()

[SWS_LinTrcv_00113] [If there is no / incorrect communication to the transceiver, the function `LinTrcv_SetOpMode` shall return `E_NOT_OK`.]()

[SWS_LinTrcv_00114] [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 report the development error `LINTRCV_E_INVALID_TRCV_OPMODE`.]()

[SWS_LinTrcv_00148] [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 report the development error `LINTRCV_E_INVALID_TRCV_OPMODE`.]()

[SWS_LinTrcv_00115] [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 report the development error `LINTRCV_E_TRCV_NOT_NORMAL.`()

[SWS_LinTrcv_00149] [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 report the development error `LINTRCV_E_TRCV_NOT_SLEEP.`()

[SWS_LinTrcv_00116] [If development error detection for the module `LinTrcv` is enabled:

If called before the `LinTrcv` module has been initialized, the function `LinTrcv_SetOpMode` shall report the development error `LINTRCV_E_UNINIT.`()

[SWS_LinTrcv_00117] [If development error detection for the module `LinTrcv` is enabled:

If called with an invalid network number `LinNetwork`, the function `LinTrcv_SetOpMode` shall report the development error `LINTRCV_E_INVALID_LIN_NETWORK.`()

[SWS_LinTrcv_00157] [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

[SWS_LinTrcv_00005] Definition of API function `LinTrcv_GetOpMode` [

| | | |
|---------------------------|--|--|
| Service Name | LinTrcv_GetOpMode | |
| Syntax | <pre>Std_ReturnType LinTrcv_GetOpMode (uint8 LinNetwork, LinTrcv_TrcvModeType* OpMode)</pre> | |
| 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 | <p><code>E_OK</code>: will be returned if the operation mode is detected</p> <p><code>E_NOT_OK</code>: will be returned, if service request is failed due to operation mode is not detected.</p> |





| | |
|----------------------|--|
| Description | API detects the actual software state of LIN transceiver driver. |
| Available via | LinTrcv.h |

]([SRS_BSW_00310](#), [SRS_BSW_00369](#), [SRS_BSW_00377](#), [SRS_Can_01097](#), [SRS_Can_01101](#))

[SWS_LinTrcv_00121] [The function [LinTrcv_GetOpMode](#) shall return the actual state of the LIN transceiver driver in the parameter [OpMode](#).]()

[SWS_LinTrcv_00122] [If there is no / incorrect communication to the transceiver, the function [LinTrcv_GetOpMode](#) shall return [E_NOT_OK](#).]()

[SWS_LinTrcv_00123] [If development error detection for the module [LinTrcv](#) is enabled:

If called before the [LinTrcv](#) module has been initialized, the function [LinTrcv_GetOpMode](#) shall report the development error [LINTRCV_E_UNINIT](#).]()

[SWS_LinTrcv_00124] [If development error detection for the module [LinTrcv](#) is enabled:

If called with an invalid network number [LinNetwork](#), the function [LinTrcv_GetOpMode](#) shall report the development error [LINTRCV_E_INVALID_LIN_NETWORK](#).]()

[SWS_LinTrcv_00125] [If development error detection for the module [LinTrcv](#) is enabled:

If called with [OpMode](#) == NULL, the function [LinTrcv_GetOpMode](#) shall report the development error [LINTRCV_E_PARAM_POINTER](#).]()

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

8.3.4 LinTrcv_GetBusWuReason

[SWS_LinTrcv_00007] Definition of API function [LinTrcv_GetBusWuReason](#) [

| | | |
|---------------------------|---|--|
| Service Name | LinTrcv_GetBusWuReason | |
| Syntax | <pre>Std_ReturnType LinTrcv_GetBusWuReason (uint8 LinNetwork, LinTrcv_TrcvWakeupReasonType* Reason)</pre> | |
| 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 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. | |
| Available via | LinTrcv.h | |

](SRS_BSW_00310, SRS_BSW_00369, SRS_BSW_00377, SRS_BSW_00406, SRS_Can_01097, SRS_Can_01103)

[SWS_LinTrcv_00126] [The function `LinTrcv_GetBusWuReason` shall return the reason for the wake up that the LIN transceiver has detected in the parameter `Reason`.]()

[SWS_LinTrcv_00127] [If there is no / incorrect communication to the transceiver, the function `LinTrcv_GetBusWuReason` shall return `E_NOT_OK`.]()

[SWS_LinTrcv_00128] [If development error detection for the module `LinTrcv` is enabled:

If called before the `LinTrcv` module has been initialized, the function `LinTrcv_GetBusWuReason` shall report development error `LINTRCV_E_UNINIT`.]()

[SWS_LinTrcv_00129] [If development error detection for the module `LinTrcv` is enabled:

If called with an invalid network number `LinNetwork`, the function `LinTrcv_GetBusWuReason` shall report development error `LINTRCV_E_INVALID_LIN_NETWORK`.]()

[SWS_LinTrcv_00130] [If development error detection for the module `LinTrcv` is enabled:

If called with `Reason == NULL`, the function `LinTrcv_GetBusWuReason` shall report the development error `LINTRCV_E_PARAM_POINTER`.]()

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

[SWS_LinTrcv_00008] Definition of API function LinTrcv_GetVersionInfo [

| | | |
|---------------------------|---|--|
| Service Name | LinTrcv_GetVersionInfo | |
| Syntax | <pre>void LinTrcv_GetVersionInfo (Std_VersionInfoType* versioninfo)</pre> | |
| 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". | |
| Available via | LinTrcv.h | |

]([SRS_BSW_00310](#), [SRS_BSW_00369](#), [SRS_BSW_00406](#), [SRS_BSW_00407](#))

[SWS_LinTrcv_00131] [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".]()

[SWS_LinTrcv_00134] [If development error detection for the module LinTrcv is enabled:

If called with `versioninfo == NULL`, the function [LinTrcv_GetVersionInfo](#) shall report development error [LINTRCV_E_PARAM_POINTER](#).]()

8.3.6 LinTrcv_CheckWakeup

[SWS_LinTrcv_00012] Definition of callback function LinTrcv_CheckWakeup [

| | | |
|---------------------------|--|--|
| Service Name | LinTrcv_CheckWakeup | |
| Syntax | <pre>Std_ReturnType LinTrcv_CheckWakeup (uint8 LinNetwork)</pre> | |
| 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: API call has been accepted E_NOT_OK: API call has not been accepted |
| Description | Notifies the calling function if a wakeup is detected. | |
| Available via | LinTrcv.h | |

]([SRS_BSW_00310](#), [SRS_BSW_00369](#), [SRS_BSW_00375](#), [SRS_BSW_00406](#), [SRS_Can_01097](#))

[SWS_LinTrcv_00144] [If development error detection for the module LinTrcv is enabled:

If called before the LinTrcv module has been initialized, the function [LinTrcv_CheckWakeup](#) shall report the development error [LINTRCV_E_UNINIT.](#)]()

[SWS_LinTrcv_00145] [If development error detection for the module LinTrcv is enabled:

If called with an invalid network number [LinNetwork](#), the function [LinTrcv_CheckWakeup](#) shall report the development error [LINTRCV_E_INVALID_LIN_NETWORK.](#)]()

[SWS_LinTrcv_00166] [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](#) and [LinIf](#) via [LinIf_WakeupConfirmation](#) callback function.]()

[SWS_LinTrcv_00167] [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 report the development error [LINTRCV_E_TRCV_NOT_SLEEP.](#)]()

Configuration: See configuration parameter [LinTrcvWakeupSupport](#).

8.3.7 LinTrcv_SetWakeupMode

[SWS_LinTrcv_00009] Definition of API function LinTrcv_SetWakeupMode [

| | | |
|---------------------------|--|---|
| Service Name | LinTrcv_SetWakeupMode | |
| Syntax | Std_ReturnType LinTrcv_SetWakeupMode (uint8 LINNetwork, LinTrcv_TrcevWakeupModeType TrcevWakupMode) | |
| Service ID [hex] | 0x05 | |
| Sync/Async | Synchronous | |
| Reentrancy | non Reentrant | |
| Parameters (in) | LINNetwork | LIN network to which API call has to be applied |
| | TrcevWakupMode | Requested transceiver wakup reason. |
| 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 service request is failed due to wakeup mode is not set. |
| Description | This API enables, disables and clears the notification for wakeup events on the addressed network. | |
| Available via | LinTrcv.h | |

]()

[SWS_LinTrcv_00135] [Enabled: If the function [LinTrcv_SetWakeupMode](#) is called with [TrcvWakupMode](#) == LINTRCV_WUMODE_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).]()

[SWS_LinTrcv_00136] [Disabled: If the function [LinTrcv_SetWakeupMode](#) is called with [TrcvWakupMode](#) == LINTRCV_WUMODE_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.]()

[SWS_LinTrcv_00137] [Clear: If the function [LinTrcv_SetWakeupMode](#) is called with [TrcvWakupMode](#) == LINTRCV_WUMODE_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.]()

[SWS_LinTrcv_00138] [If there is no / incorrect communication to the transceiver, the function [LinTrcv_SetWakeupMode](#) shall return E_NOT_OK.]()

[SWS_LinTrcv_00139] [If development error detection for the module LinTrcv is enabled:

If called before the LinTrcv has been initialized, the function [LinTrcv_SetWakeupMode](#) shall report development error [LINTRCV_E_UNINIT](#).]()

[SWS_LinTrcv_00140] [If development error detection for the module LinTrcv is enabled:

If called with an invalid network number [LINNetwork](#), the function [LinTrcv_SetWakeupMode](#) shall report development error [LINTRCV_E_INVALID_LIN_NETWORK](#).]()

8.4 Callback notifications

There are no callback notifications provided by LIN Transceiver Driver.

8.5 Scheduled functions

This chapter 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.

8.6 Expected interfaces

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

8.6.1 Mandatory interfaces

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

[SWS_LinTrcv_91002] Definition of mandatory interfaces in module LinTrcv [

| API Function | Header File | Description |
|--------------------------|-------------|--|
| Linlf_WakeupConfirmation | Linlf.h | The LIN Driver or LIN Transceiver Driver will call this function to report the wake up source after the successful wakeup detection during CheckWakeup or after power on by bus. |

]()

8.6.2 Optional interfaces

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

[SWS_LinTrcv_91003] Definition of optional interfaces in module LinTrcv [

| API Function | Header File | Description |
|-------------------------|-------------|---|
| Det_ReportError | Det.h | Service to report development errors. |
| Dio_ReadChannel | Dio.h | Returns the value of the specified DIO channel. |
| Dio_ReadChannelGroup | Dio.h | This Service reads a subset of the adjoining bits of a port. |
| Dio_ReadPort | Dio.h | Returns the level of all channels of that port. |
| Dio_WriteChannel | Dio.h | Service to set a level of a channel. |
| Dio_WriteChannelGroup | Dio.h | Service to set a subset of the adjoining bits of a port to a specified level. |
| Dio_WritePort | Dio.h | Service to set a value of the port. |
| EcuM_SetWakeupEvent | EcuM.h | Sets the wakeup event. |
| Icu_DisableNotification | Icu.h | This function disables the notification of a channel. |





| API Function | Header File | Description |
|------------------------|--------------------|--|
| Icu_EnableNotification | Icu.h | This function enables the notification on the given channel. |
| Spi_GetStatus | Spi.h | Service returns the SPI Handler/Driver software module status. |
| Spi_ReadIB | Spi.h | Service for reading synchronously one or more data from an IB SPI Handler/Driver Channel specified by parameter. |
| Spi_SetupEB | Spi.h | Service to setup the buffers and the length of data for the EB SPI Handler/Driver Channel specified. |
| Spi_SyncTransmit | Spi.h | Service to transmit data on the SPI bus |
| Spi_WriteIB | Spi.h | Service for writing one or more data to an IB SPI Handler/Driver Channel specified by parameter. |
| Tm_BusyWait1us16bit | Tm.h | Performs busy waiting by polling with a guaranteed minimum waiting time. |

]()

[SWS_LinTrcv_00165] [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. It also specifies a template (table) you shall use for the parameter specification. We intend to leave Chapter 10.1 in the specification to guarantee comprehension.

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

Chapter 10.3 specifies published information of the module LinTrcv.

10.1 How to read this chapter

For details refer to the chapter "Introduction to configuration specification" in SWS_BSWGeneral.

[SWS_LinTrcv_00174] [The LIN Transceiver Driver module shall reject configurations with partition mappings which are not supported by the implementation.] ()

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

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

10.2.2 LinTrcv

| | |
|-----------------------------------|--|
| SWS Item | [ECUC_LinTrcv_00161] |
| Module Name | LinTrcv |
| Description | Configuration of LIN Transceiver Driver module |
| Post-Build Variant Support | false |
| Supported Config Variants | VARIANT-PRE-COMPILE |

| 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. |

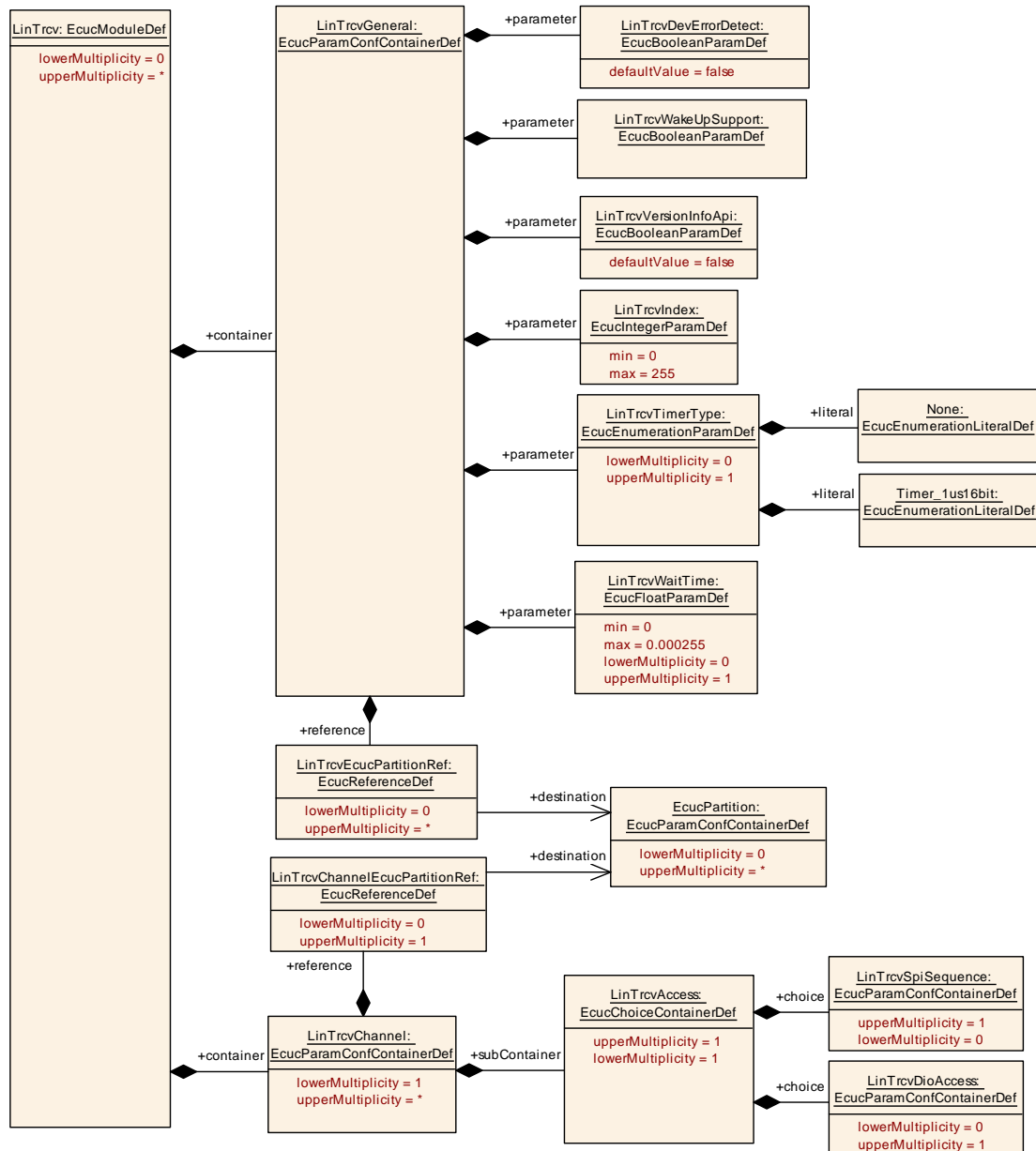


Figure 10.1: Overview about LIN Transceiver Driver configuration containers

10.2.3 LinTrcvGeneral

| | |
|---------------------------------|---|
| SWS Item | [ECUC_LinTrcv_00090] |
| Container Name | LinTrcvGeneral |
| Parent Container | LinTrcv |
| Description | Container gives LIN transceiver driver basic information. |
| Configuration Parameters | |

| | | | |
|----------------------------------|---|---|--------------|
| SWS Item | [ECUC_LinTrcv_00001] | | |
| Parameter Name | LinTrcvDevErrorDetect | | |
| Parent Container | LinTrcvGeneral | | |
| Description | Switches the development error detection and notification on or off. <ul style="list-style-type: none"> • true: detection and notification is enabled. • false: detection and notification is disabled. | | |
| Multiplicity | 1 | | |
| Type | EcucBooleanParamDef | | |
| Default value | false | | |
| Post-Build Variant Value | false | | |
| Value Configuration Class | Pre-compile time | X | All Variants |
| | Link time | – | |
| | Post-build time | – | |
| Scope / Dependency | scope: local | | |

| | | | |
|----------------------------------|---|---|--------------|
| SWS Item | [ECUC_LinTrcv_00153] | | |
| Parameter Name | LinTrcvIndex | | |
| Parent Container | LinTrcvGeneral | | |
| 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 | – | | |
| Post-Build Variant Value | false | | |
| Value Configuration Class | Pre-compile time | X | All Variants |
| | Link time | – | |
| | Post-build time | – | |
| Scope / Dependency | scope: local | | |

| | | |
|--|--|------------------|
| SWS Item | [ECUC_LinTrcv_00159] | |
| Parameter Name | LinTrcvTimerType | |
| Parent Container | LinTrcvGeneral | |
| Description | Type of the Time Service Predefined Timer. | |
| Multiplicity | 0..1 | |
| Type | EcucEnumerationParamDef | |
| Range | None | None |
| | Timer_1us16bit | 16 bit 1us timer |
| Post-Build Variant Multiplicity | false | |
| Post-Build Variant Value | false | |





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

| | | | |
|----------------------------------|--|---|--------------|
| SWS Item | [ECUC_LinTrcv_00003] | | |
| Parameter Name | LinTrcvVersionInfoApi | | |
| Parent Container | LinTrcvGeneral | | |
| 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 | | |
| Post-Build Variant Value | false | | |
| Value Configuration Class | Pre-compile time | X | All Variants |
| | Link time | – | |
| | Post-build time | – | |
| Scope / Dependency | scope: local | | |

| | | | |
|---|---|---|--------------|
| SWS Item | [ECUC_LinTrcv_00160] | | |
| Parameter Name | LinTrcvWaitTime | | |
| Parent Container | LinTrcvGeneral | | |
| Description | Wait time for transceiver state changes in seconds. | | |
| Multiplicity | 0..1 | | |
| Type | EcucFloatParamDef | | |
| Range | [0 .. 2.55E-4] | | |
| Default value | – | | |
| Post-Build Variant Multiplicity | false | | |
| Post-Build Variant Value | false | | |
| Multiplicity Configuration Class | Pre-compile time | X | All Variants |
| | Link time | – | |
| | Post-build time | – | |
| Value Configuration Class | Pre-compile time | X | All Variants |
| | Link time | – | |
| | Post-build time | – | |
| Scope / Dependency | scope: local | | |

| | | | |
|-------------------------|--------------------------------|--|--|
| SWS Item | [ECUC_LinTrcv_00107] | | |
| Parameter Name | LinTrcvWakeUpSupport | | |
| Parent Container | LinTrcvGeneral | | |





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

| | | | |
|---|--|---|--------------|
| SWS Item | [ECUC_LinTrcv_00162] | | |
| Parameter Name | LinTrcvEcucPartitionRef | | |
| Parent Container | LinTrcvGeneral | | |
| Description | Maps the Lin transceiver driver to zero or multiple ECUC partitions to make the modules API available in this partition. The Lin transceiver driver will operate as an independent instance in each of the partitions. | | |
| Multiplicity | 0..* | | |
| Type | Reference to EcucPartition | | |
| Post-Build Variant Multiplicity | true | | |
| Post-Build Variant Value | true | | |
| Multiplicity Configuration Class | Pre-compile time | X | All Variants |
| | Link time | - | |
| | Post-build time | - | |
| Value Configuration Class | Pre-compile time | X | All Variants |
| | Link time | - | |
| | Post-build time | - | |
| Scope / Dependency | scope: ECU | | |

| |
|------------------------|
| No Included Containers |
|------------------------|

[SWS_LinTrcv_00175] [The module will operate as an independent instance in each of the partitions, means the called API will only target the partition it is called in.]()

10.2.4 LinTrcvChannel

| | |
|---------------------------------|--|
| SWS Item | [ECUC_LinTrcv_00091] |
| Container Name | LinTrcvChannel |
| Parent Container | LinTrcv |
| 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 | [ECUC_LinTrcv_00011] | | |
| Parameter Name | LinTrcvChannelId | | |
| Parent Container | LinTrcvChannel | | |
| Description | Unique identifier of the LIN Transceiver Channel. | | |
| Multiplicity | 1 | | |
| Type | EcucIntegerParamDef (Symbolic Name generated for this parameter) | | |
| Range | 0 .. 255 | | |
| Default value | - | | |
| Post-Build Variant Value | false | | |
| Value Configuration Class | Pre-compile time | X | All Variants |
| | Link time | - | |
| | Post-build time | - | |
| Scope / Dependency | scope: local withAuto = true | | |

| | | | |
|----------------------------------|---|---|--------------|
| SWS Item | [ECUC_LinTrcv_00004] | | |
| Parameter Name | LinTrcvChannelUsed | | |
| Parent Container | LinTrcvChannel | | |
| Description | Shall the related LIN transceiver channel be used? True: Is used False Is not used | | |
| Multiplicity | 1 | | |
| Type | EcucBooleanParamDef | | |
| Default value | true | | |
| Post-Build Variant Value | false | | |
| Value Configuration Class | Pre-compile time | X | All Variants |
| | Link time | - | |
| | Post-build time | - | |
| Scope / Dependency | scope: local | | |

| | | | |
|----------------------------------|---|---|--------------|
| SWS Item | [ECUC_LinTrcv_00006] | | |
| Parameter Name | LinTrcvWakeupByBusUsed | | |
| Parent Container | LinTrcvChannel | | |
| 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 | | |
| Post-Build Variant Value | false | | |
| Value Configuration Class | Pre-compile time | X | All Variants |
| | Link time | - | |
| | Post-build time | - | |
| Scope / Dependency | scope: local dependency: LinTrcvWakeUpSupport | | |

| | | | |
|---|---|---|--------------|
| SWS Item | [ECUC_LinTrcv_00163] | | |
| Parameter Name | LinTrcvChannelEcucPartitionRef | | |
| Parent Container | LinTrcvChannel | | |
| Description | Maps one single Lin transceiver channel to zero or one ECUC partitions. The ECUC partition referenced is a subset of the ECUC partitions where the Lin transceiver driver is mapped to. | | |
| Multiplicity | 0..1 | | |
| Type | Reference to EcucPartition | | |
| Post-Build Variant Multiplicity | true | | |
| Post-Build Variant Value | true | | |
| Multiplicity Configuration Class | Pre-compile time | X | All Variants |
| | Link time | – | |
| | Post-build time | – | |
| Value Configuration Class | Pre-compile time | X | All Variants |
| | Link time | – | |
| | Post-build time | – | |
| Scope / Dependency | scope: ECU | | |

| | | | |
|---|---|---|--------------|
| SWS Item | [ECUC_LinTrcv_00157] | | |
| Parameter Name | LinTrcvIcuChannelRef | | |
| Parent Container | LinTrcvChannel | | |
| Description | Reference to the IcuChannel to enable/disable the interrupts for wakeups. | | |
| Multiplicity | 0..1 | | |
| Type | Symbolic name reference to IcuChannel | | |
| Post-Build Variant Multiplicity | false | | |
| Post-Build Variant Value | false | | |
| Multiplicity Configuration Class | Pre-compile time | X | All Variants |
| | Link time | – | |
| | Post-build time | – | |
| Value Configuration Class | Pre-compile time | X | All Variants |
| | Link time | – | |
| | Post-build time | – | |
| Scope / Dependency | scope: ECU | | |

| | | | |
|---|---|---|--------------|
| SWS Item | [ECUC_LinTrcv_00012] | | |
| Parameter Name | LinTrcvWakeupSourceRef | | |
| Parent Container | LinTrcvChannel | | |
| 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 | Symbolic name reference to EcuMWakeupSource | | |
| Post-Build Variant Multiplicity | false | | |
| Post-Build Variant Value | false | | |
| Multiplicity Configuration Class | Pre-compile time | X | All Variants |
| | Link time | – | |
| | Post-build time | – | |
| Value Configuration Class | Pre-compile time | X | All Variants |



△

| | | | |
|---------------------------|--|---|--|
| | Link time | - | |
| | Post-build time | - | |
| Scope / Dependency | scope: ECU dependency: LinTrcvWakeupByBusUsed | | |

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

[SWS_LinTrcv_00176] [The ECUC partitions referenced by [LinTrcvChannelEcucPartitionRef](#) shall be a subset of the ECUC partitions referenced by [LinTrcvEcucPartitionRef](#).]()

[SWS_LinTrcv_00178] [If [LinTrcvEcucPartitionRef](#) references one or more ECUC partitions, [LinTrcvChannelEcucPartitionRef](#) shall have a multiplicity of one and reference one of these ECUC partitions as well.]()

[SWS_LinTrcv_00177] [[LinTrcvChannel](#) and [LinChannel](#) of one communication channel shall all reference the same ECUC partition.]()

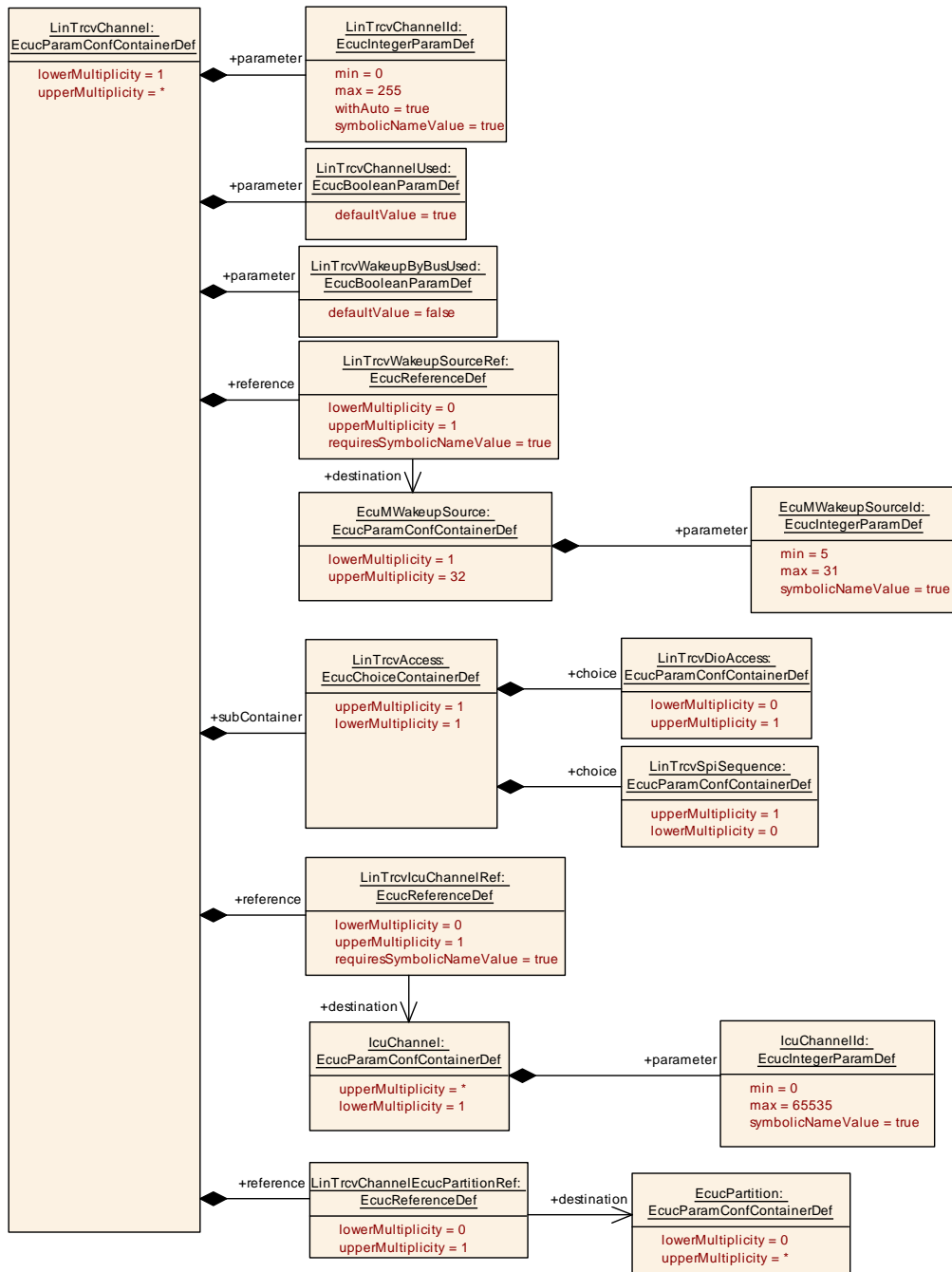


Figure 10.2: LinTrcvChannel configuration container

10.2.5 LinTrcvAccess

| | |
|-----------------------|---|
| SWS Item | [ECUC_LinTrcv_00154] |
| Choice Container Name | LinTrcvAccess |
| Parent Container | LinTrcvChannel |
| 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. |

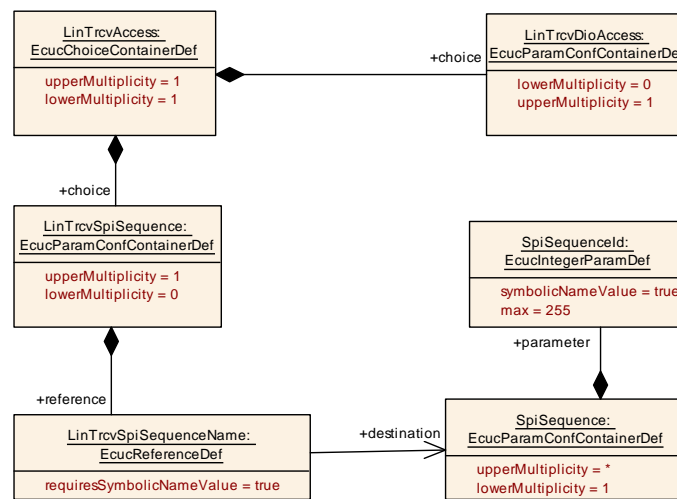


Figure 10.3: LinTrcvAccess configuration container

10.2.6 LinTrcvDioAccess

| | |
|--------------------------|---|
| SWS Item | [ECUC_LinTrcv_00094] |
| Container Name | LinTrcvDioAccess |
| Parent Container | LinTrcvAccess |
| 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 | |

| Included Containers | | |
|---|--------------|---|
| Container Name | Multiplicity | Scope / Dependency |
| LinTrcvDioChannelAccess | 1..* | Container gives DIO channel access by single Lin transceiver channel. |

10.2.7 LinTrcvDioChannelAccess

| | |
|---------------------------------|---|
| SWS Item | [ECUC_LinTrcv_00158] |
| Container Name | LinTrcvDioChannelAccess |
| Parent Container | LinTrcvDioAccess |
| Description | Container gives DIO channel access by single Lin transceiver channel. |
| Configuration Parameters | |

| | | | |
|----------------------------------|--|---|--------------|
| SWS Item | [ECUC_LinTrcv_00009] | | |
| Parameter Name | LinTrcvHardwareInterfaceName | | |
| Parent Container | LinTrcvDioChannelAccess | | |
| 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 | – | | |
| Regular Expression | – | | |
| Post-Build Variant Value | false | | |
| Value Configuration Class | Pre-compile time | X | All Variants |
| | Link time | – | |
| | Post-build time | – | |
| Scope / Dependency | scope: local | | |

| | | | |
|----------------------------------|--|---|--------------|
| SWS Item | [ECUC_LinTrcv_00102] | | |
| Parameter Name | LinTrcvDioSymRefName | | |
| Parent Container | LinTrcvDioChannelAccess | | |
| 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] | | |
| Post-Build Variant Value | false | | |
| Value Configuration Class | Pre-compile time | X | All Variants |
| | Link time | – | |
| | Post-build time | – | |
| Scope / Dependency | scope: local | | |

| |
|-------------------------------|
| No Included Containers |
|-------------------------------|

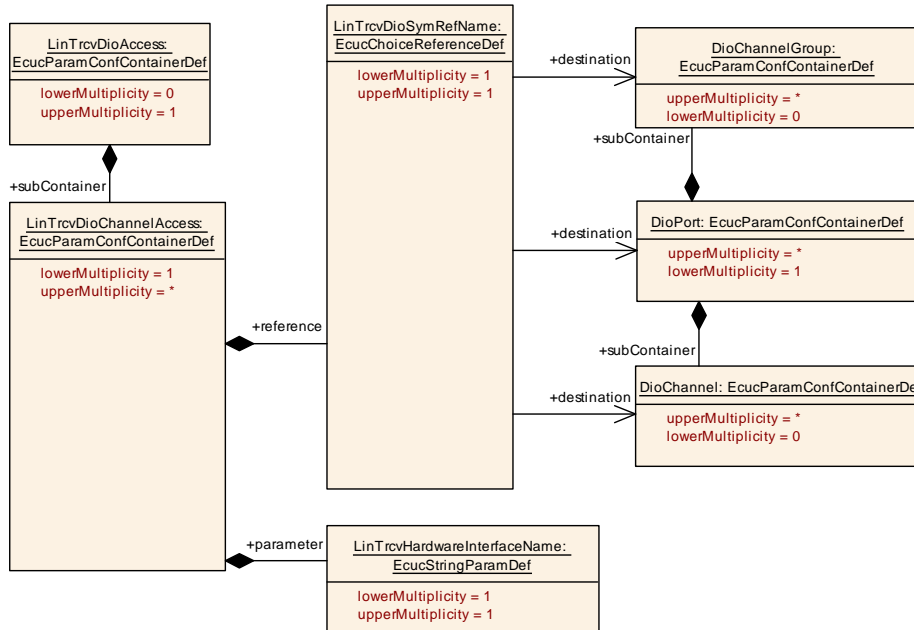


Figure 10.4: LinTrcvDioAccess configuration container

10.2.8 LinTrcvSpiSequence

| | |
|--------------------------|--|
| SWS Item | [ECUC_LinTrcv_00155] |
| Container Name | LinTrcvSpiSequence |
| Parent Container | LinTrcvAccess |
| 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 hardware 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 | [ECUC_LinTrcv_00156] | | |
| Parameter Name | LinTrcvSpiSequenceName | | |
| Parent Container | LinTrcvSpiSequence | | |
| Description | Reference to a Spi sequence configuration container. | | |
| Multiplicity | 1 | | |
| Type | Symbolic name reference to SpiSequence | | |
| Post-Build Variant Value | false | | |
| Value Configuration Class | Pre-compile time | X | All Variants |
| | Link time | – | |
| | Post-build time | – | |
| Scope / Dependency | scope: local dependency: SpiSequence | | |

| |
|------------------------|
| No Included Containers |
|------------------------|

10.3 Published Information

For details refer to the chapter "Published Information" in SWS_BSWGeneral.

A Not applicable requirements

[SWS_LinTrcv_NA_00999] [These requirements are not applicable to this specification.] (*SRS_BSW_00304, SRS_BSW_00305, SRS_BSW_00306, SRS_BSW_00307, SRS_BSW_00308, SRS_BSW_00309, SRS_BSW_00312, SRS_BSW_00321, SRS_BSW_00325, SRS_BSW_00328, SRS_BSW_00330, SRS_BSW_00331, SRS_BSW_00333, SRS_BSW_00334, SRS_BSW_00335, SRS_BSW_00336, SRS_BSW_00341, SRS_BSW_00342, SRS_BSW_00344, SRS_BSW_00359, SRS_BSW_00360, SRS_BSW_00378, SRS_BSW_00383, SRS_BSW_00384, SRS_BSW_00398, SRS_BSW_00399, SRS_BSW_00400, SRS_BSW_00401, SRS_BSW_00404, SRS_BSW_00405, SRS_BSW_00410, SRS_BSW_00416, SRS_BSW_00417, SRS_BSW_00422, SRS_BSW_00423, SRS_BSW_00426, SRS_BSW_00427, SRS_BSW_00429, SRS_BSW_00432, SRS_BSW_00433, SRS_BSW_00005, SRS_BSW_00006, SRS_BSW_00007, SRS_BSW_00009, SRS_BSW_00010, SRS_BSW_00159, SRS_BSW_00161, SRS_BSW_00164, SRS_BSW_00167, SRS_BSW_00168, SRS_BSW_00170, SRS_Lin_01576, SRS_Lin_01504, SRS_Lin_01522, SRS_Lin_01560, SRS_Lin_01577, SRS_Lin_01551, SRS_Lin_01568, SRS_Lin_01569, SRS_Lin_01564, SRS_Lin_01546, SRS_Lin_01549, SRS_Lin_01571, SRS_Lin_01515, SRS_Lin_01502, SRS_Lin_01558, SRS_Lin_01523, SRS_Lin_01553, SRS_Lin_01552, SRS_Lin_01503, SRS_Lin_01555, SRS_Lin_01547, SRS_Lin_01572, SRS_Lin_01556, SRS_Lin_01579, SRS_Lin_01540, SRS_Lin_01545, SRS_Lin_01534, SRS_Lin_01574, SRS_Lin_01539, SRS_Lin_01544, SRS_Can_01115*)