Document Title	Requirements on LIN
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	42

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

Document Change History

Date	Release	Changed by	Description
2024-11-27	R24-11	AUTOSAR Release Management	• Support IEEE 1722 Tunneling • Editorial changes
2023-11-23	R23-11	AUTOSAR Release Management	• No content changes
2022-11-24	R22-11	AUTOSAR Release Management	• No content changes
2021-11-25	R21-11	AUTOSAR Release Management	• No content changes
2020-11-30	R20-11	AUTOSAR Release Management	• No content changes
2019-11-28	R19-11	AUTOSAR Release Management	• No content changes • Changed Document Status from Final to published
2018-10-31	4.4.0	AUTOSAR Release Management	• LIN Slave Support (CONC_631) • Added [SRS_Lin_01593] to make TP timer to connection specific • Replaced references to LIN 2.1 by ISO 17987:2016 (with no functional modification) • Minor corrections / clarifications / editorial changes; For details please refer to the ChangeDocumentation

▽

△

2017-12-08	4.3.1	AUTOSAR Release Management	• Changed [SRS_Lin_01514] to solve inconsistency on channel state upon initialization • Corrected [SRS_Lin_01564] to fit to current priority handling of LIN Schedule Table
2016-11-30	4.3.0	AUTOSAR Release Management	• Added requirement tracing section
2014-10-31	4.2.1	AUTOSAR Release Management	• Changed [SRS_Lin_01564] Schedule Table change request buffering
2013-10-31	4.1.2	AUTOSAR Release Management	• Added [SRS_Lin_01592] Transmission of functional requests by LinTp • Changed [SRS_Lin_01534] LinTp support for half-duplex physical connections
2013-03-15	4.1.1	AUTOSAR Administration	• TPS_STDT_0078 formatting • Traceability of BSWAndRTE_Features
2011-12-22	4.0.3	AUTOSAR Administration	• Delete [BSW01527] • Change [SRS_Lin_01588] – Add requirement of wake pin
2009-12-03	4.0.1	AUTOSAR Administration	• Conformation the use of the terms cluster, network, bus and channel • Add LIN 2.1 support • Additional requirements for the LIN Transceiver Driver • Legal disclaimer revised
2008-08-13	3.1.1	AUTOSAR Administration	• Legal disclaimer revised
2007-12-21	3.0.1	AUTOSAR Administration	• Document meta information extended • Small layout adaptations made
2007-01-24	2.1.15	AUTOSAR Administration	• “Advice for users” revised • “Revision Information” added
2006-11-28	2.1	AUTOSAR Administration	• Legal disclaimer revised • Extend description of rationale of [SRS_Lin_01555]

▽

△

2006-05-16	2.0	AUTOSAR Administration	• Initial Release

Disclaimer

Contents

1 Scope of Document

This document specifies the requirements for the following Basic Software Modules (module names in brackets):

- LIN Driver (Lin)
- LIN Transceiver Driver (LinTrcv)
- LIN Interface (LinIf)
- LIN Transport Layer (LinTp)

The intention is to reference as much as possible to the ISO 17987 specifications [1]. The behavior covers LIN master nodes and LIN slave nodes. It is the goal to support LIN nodes already existing on the market (i.e. that conforms to the respective specification).

The reader of this document should know the LIN specifications.

Note: The ISO 17987 specifications cover ISO 14229-7 [2] and the behavior of previous versions of LIN specifications: LIN 2.2, LIN 2.1, LIN 2.0 and LIN 1.3 by LIN Consortium. See Annex B of ISO 17987-3 [3] for compatibility information.

2 Conventions to be used

2.1 Document Conventions

The representation of requirements in AUTOSAR documents follows the table specified in [TPS_STDT_00078], see Standardization Template, chapter Support for Traceability ([4]).

The verbal forms for the expression of obligation specified in [TPS_STDT_00053] shall be used to indicate requirements, see Standardization Template, chapter Support for Traceability ([4]).

2.2 Requirements Guidelines

Each module specific chapter contains a short functional description of the Basic Software Module. Requirements of the same kind within each chapter are grouped under the following headlines (where applicable):

The "Requirements on LIN" is divided into five sections.

- LIN General requirements
- LIN Interface
- LIN Driver
- LIN Transceiver Driver
- LIN Transport Layer (TP)

The subchapters are only applied, if they are needed. See following structure.

- Non-functional requirements
- Functional requirements
- Configuration
- Initialization
- Normal Operation
- Fault Operation
- Shutdown Operations

3 Acronyms and abbreviations

The glossary below includes acronyms and abbreviations relevant to the Requirements on LIN that are not included in the AUTOSAR Glossary [5].

Acronym:	Description:
LIN-PDU	LIN Protocol Data Unit is the LIN header and the LIN response, i.e. Break, synch, PID, Data (1-8) and checksum In the ISO 17987 specifications, this is called just frame. LIN_PDU is more precise and omits confusion.
LIN-SDU	LIN Service Data Unit. The data-part of the LIN response.
Schedule Table	A Schedule Table determines the traffic on a LIN bus (one channel). One LIN bus could have more than one Schedule Table.
Schedule Table Handler	The Schedule Table Handler is placed at the LIN Interface of a LIN master node. It will initiate LIN-PDU's and confirm/ indicate LIN-PDU's. It will be called by upper layers.
Schedule Table Manager	Keeps track of all available schedule and processes the active schedule table in a LIN master node.
LIN Driver	Module name Lin. Describes the Software Driver.
LIN Interface	Module name LinIf. LIN Interface, describes the ISO 17987 protocol communication stack.
Sleep-mode	In the ISO 17987 specifications the term stand-by and sleep-mode is used in similar manner. To be consequent here only sleep-mode is used.

Abbreviation:	Description:
LIN	Local Interconnect Network
FF	First Frame
CF	Consecutive Frames
SF	Single Frames
N_PDU	Network Protocol Data Unit
PDUR	Protocol Data Unit Router
N_SDU	Network Service Data Unit
N_TA	Extended Addressing Mode Connection
UART	Universal Asynchronous Receiver Transmitter. Dear children have many names, it is also known as SCI and ESCI.
MRF	Master Request Frame
SRF	Slave Response Frame

4 Requirements Specification

This chapter describes all requirements driving the work to define the Requirements of LIN.

4.1 Functional Requirements

4.1.1 LIN General

4.1.1.1 General Requirements

[SRS_Lin_01576] The ISO 17987 specifications shall be reused as far as possible

Upstream requirements: RS_BRF_01768

⌈

Description:	The following sections in the ISO 17987 specifications shall be reused ISO 17987-3 [3]: • Clause 5.2 Frame (handling of different types of LIN-PDUs, not the specific bytes of the LIN-PDU) • Clause 5.3 Schedules tables • Clause 5.4 Task Behaviour Model (handling of LIN-PDUs and errors. Not handling specific bytes in the LIN-PDU) • Clause 5.5 Status Management • Clause 6 Node Configuration and identification ISO 17987-2 [6]: • Clause 5 Network Management • Clause 7 Transport Layer Protocol • Clause 9 Diagnostic communication requirements (partly) The diagnose classes II and III in AUTOSAR are optional and a precompiled option (see [SRS_Lin_01579]). The Diagnostic Transport Protocol is also used by the node configuration and identification in the ISO 17987 specifications, so it is mandatory in AUTOSAR as a precompiled option. The remaining chapters in the ISO 17987 specifications will not be reused as is. Refer to the corresponding LIN Driver and Interface sections for the exact details. There are optional functionality in the ISO 17987 specifications (e.g. in the configuration): ▽

▽

△

	△ • All other optional functionality is decided by the design. The following item shall be used with AUTOSAR adaptations: • Application Program Interface Specification The [SRS_Lin_01577] will take care that the LIN Interface is compatible with the ISO 17987 specifications, also [SRS_Lin_01578] for the LIN Driver and [SRS_Lin_01579] for the LIN TP. The usage of nodes with previous LIN versions is already covered with the ISO 17987 specifications. If a cluster nodes based on previous/non-ISO versions of the LIN specification, the LIN Master has to support the configuration services of the previous/non-ISO versions of the LIN specification, too.
Rationale:	Reuse of existing standards. This ensures the reusability of LIN slave ECU's inside the vehicle architecture. Each ISO 17987 LIN Master will support both models to assign LIN IDs ("2.0" and "2.1 or later"), the enhanced checksum for LIN 2.x/ISO 17987 and the classic checksum for LIN 1.x slaves and diagnostic messages (see ISO 17987-3 [3] clause 5.2.2.7 Checksum) The LIN Physical Layer Specification is not in the scope of AUTOSAR The checksum models (classic and enhanced) will be configurable for each LIN ID, except for the reserved LIN ID's (MRF and SRF)
Use Case:	–
Dependencies:	[SRS_Lin_01577], [SRS_Lin_01578], [SRS_Lin_01579]
Supporting Material:	ISO 17987 specifications [1]

⌋

[SRS_Lin_01504] The usage of AUTOSAR architecture shall be applicable for LIN master nodes

Upstream requirements: RS_BRF_01768

⌈

Description:	The AUTOSAR LIN should cover LIN master nodes.
Rationale:	Modelling and implementation of LIN master nodes according to LIN 2.1, LIN 2.2 and ISO 17987.
Use Case:	–
Dependencies:	–
Supporting Material:	–

⌋

[SRS_Lin_01598] The usage of AUTOSAR architecture shall be applicable for LIN slave nodes

Upstream requirements: RS_BRF_01768

⌈

Description:	The AUTOSAR LIN should cover LIN slave nodes.
Rationale:	Modelling and implementation of LIN slave nodes according to LIN 1.3, LIN 2.0, LIN2.1, LIN2.2 and ISO 17987.
Use Case:	–
Dependencies:	–
Supporting Material:	Concept 631 "LIN Slave Support"

⌋

[SRS_Lin_01594] LIN slave shall support the node configuration and identification services for slave nodes.

Upstream requirements: RS_BRF_01768

⌈

Description:	This requirement is only applicable to LIN slave nodes. LIN slave nodes shall support the mandatory node configuration and identification services to assign frame IDs and to retrieve the node identification information. LIN slave nodes shall also support the assign node address for diagnostics (NAD) service.
Rationale:	Implementation of the LIN standard.
Use Case:	Basic functionality and support of LIN conformance test
Dependencies:	–
Supporting Material:	ISO 17987-3 [3] specification

⌋

4.1.1.2 Initialization

[SRS_Lin_01590] The node configuration of LIN slaves shall only be done via defined schedule table(s) in master nodes.

Upstream requirements: RS_BRF_01768

⌈

Description:	The AUTOSAR LIN should cover the "normal" behavior to do the configuration (assign IDs, NADs, . . .) of LIN slaves. This shall avoid non LIN-conform configuration methods.
Rationale:	This shall be part of the verification of the system design.
Use Case:	LIN node configuration at bus start or after a node fault
Dependencies:	–
Supporting Material:	ISO 17987 specifications [1]

⌋

4.1.1.3 Normal Operation

[SRS_Lin_01522] LIN-SDU shall be copied consistently for transfer

Upstream requirements: RS_BRF_01768

⌈

Description:	The data from the upper layers needs to be copied consistently to the LIN Driver before transmission. The data from the LIN Driver shall be copied consistently to the upper layers after reception. The consistent coping includes the payload (data) and the flags.
Rationale:	Basic functionality. Guarantee 100% message LIN-SDU consistency for transmission and reception. Needed for every LIN-PDU transmission/reception on the LIN Bus.
Use Case:	–
Dependencies:	–
Supporting Material:	–

⌋

4.1.1.4 Shutdown Operation

[SRS_Lin_01560] If a wakeup occurs during transition to sleep-mode, this channel shall go back to the running mode

Upstream requirements: RS_BRF_01768, RS_BRF_01104

⌈

Description:	If a wakeup occurs during transition to sleep-mode, the affected channels shall go back to the running mode. The upper layer shall be notified. In detail: If an upper layer wake-up is received during Stop: Stop process should be completed, and network affected LIN cluster started afterwards. If a bus wake-up is received during Stop: Stop operation should be cancelled, and upper layer notified.
Rationale:	Safe wakeup and sleep handling.
Use Case:	The following use-cases shall be detected for LIN master nodes: If the master is processing the go-to-sleep command and the upper layer requests a wakeup. There is a time from the go-to-sleep command is transmitted on the bus until it is confirmed in the LIN Interface. During this time it is possible that a slave will transmit a wakeup-request The following use-cases shall be detected for LIN slave: If the slave is processing a sleep transition and a wakeup request is received. There is a time after a sleep mode frame is received or a bus idle condition is detected until confirmation in upper layer. During this time, it is possible that another node will transmit a wakeup-request.
Dependencies:	–
Supporting Material:	–

⌋

4.1.1.4.1 Fault Operation

None

4.1.2 LIN Interface

4.1.2.1 General requirements

[SRS_Lin_01577] It shall be compatible to LIN protocol specification

Upstream requirements: RS_BRF_01768

⌈

Description:	The following parts and clauses of the ISO 17987 specifications shall be reused by the LIN Interface: ISO 17987-3 [3]: • Clause 5.2 Frame (handling of different types of LIN-PDUs, not the specific bytes of the LIN-PDU) • Clause 5.3 Schedules tables • Clause 5.4 Task Behaviour Model (handling of LIN-PDUs and errors. Not handling specific bytes in the LIN-PDU) • Clause 5.5 Status Management ISO 17987-2 [6]: • Clause 5 Network Management The ISO 17987 specifications cover the behavior of previous versions of LIN specifications. If a cluster uses slave nodes based on previous/non-ISO versions of the LIN specification, the LIN Master node has to support the behavior of this previous LIN protocol version.
Rationale:	Basic LIN functionality
Use Case:	–
Dependencies:	–
Supporting Material:	ISO 17987-2 [6] ISO 17987-3 [3]

⌋

[SRS_Lin_01551] One LIN Interface shall support one or more LIN Drivers.

Upstream requirements: RS_BRF_01768

⌈

Description:	There shall only be one instance of the LIN Interface in each ECU. One ECU might contain more than one LIN channel. Thus the LIN Interface shall support one or more LIN Drivers.

▽

△

Rationale:	Devices, which use more than one LIN channels, exist on the market.
Use Case:	–
Dependencies:	–
Supporting Material:	–

⌋

[SRS_Lin_01568] The LIN Interface implementation and interface shall be independent from underlying LIN hardware.

Upstream requirements: RS_BRF_01768, RS_BRF_01000

⌈

Description:	The implementation may depend on the amount of available resources of the underlying hardware. The different mechanisms of hardware access are encapsulated by the LIN driver.
Rationale:	Portability and reusability.
Use Case:	If the underlying LIN device driver just handles one controller the implementation of the LIN interface may be more efficient.
Dependencies:	[SRS_Lin_01552]
Supporting Material:	–

⌋

[SRS_Lin_01599] LinIf Forwarding of L-PDUs to LSduR

Status: DRAFT
Upstream requirements: RS_Main_00280

⌈

Description:	The LinIf shall have forward L-PDUs to the LSduR. Additional Information: The LinIf must have exactly one upper layer module which perform the PDU routing with frame specific information. Therefore a L-PDU must be split in frame specific information and payload. Frame specific information must be added to meta data and payload must be assigned to the according PDU ID.
Rationale:	–
Use Case:	Time Synchronous and Non-Time Synchronous Control Frames Format to Support Tunneling of CAN and LIN
Dependencies:	–
Supporting Material:	[7, IEEE1722]

⌋

4.1.2.2 Initialization

[SRS_Lin_01569] The LIN Interface shall support initialization of each LIN channel separately

Upstream requirements: RS_BRF_01768, RS_BRF_01136, RS_BRF_01096

⌈

Description:	The LIN Interface shall support initialization of each LIN channel separately. The selection of at least one static configuration set shall be done by a parameter.
Rationale:	–
Use Case:	If there are more than one LIN channels, than there exist more than one LDF-files.
Dependencies:	–
Supporting Material:	Comparing with the ISO 17987 specifications' APIs, the LIN Interface init will do both work of l_ifc_init and l_sys_init

⌋

[SRS_Lin_01570] The LIN Interface shall support dynamic selection of configuration sets.

Upstream requirements: RS_BRF_01768, RS_BRF_01136

⌈

Description:	The LIN Interface shall support the dynamic selection of at least one static configuration set by a parameter passed via the initialization interface. The selection of the appropriate configuration set itself as well as the way to incorporate the configuration sets into the ECU (Post-Build, Pre-Compile) is not affected by this requirement.
Rationale:	Support of different configurations during runtime
Use Case:	–
Dependencies:	–
Supporting Material:	–

⌋

4.1.2.3 Normal Operation

[SRS_Lin_01564] A Schedule Table Manager shall be available for LIN master nodes.

Upstream requirements: RS_BRF_01768, RS_BRF_01592

⌈

Description:	This requirement is only applicable to LIN master nodes. The schedule table manager will keep schedule table to execute. The schedule table manager shall: • Be able to receive requests from an upper layer which schedule table to execute • Keep a list of schedule table • Execute a schedule table once or continuously One or more modules from an upper layer will create a sequence of schedule tables and request the schedule table manager to execute specific schedule tables. Priority handling is not handled inside LinStack. The schedule table manager will only coordinate the running schedule table and schedule table requests. There exist one memory space for the "continuously execution schedule table", it will be overwritten by a newer request.
Rationale:	In the ISO 17987 specifications, the application interfaces directly to the LIN API. In AUTOSAR above modules shall be able to independently request a schedule table to be executed. Therefore the schedule table manager is a necessarily extension to the schedule table handler.
Use Case:	Example system start: a) "Run"- schedule table (execution continuously, low priority 1) b) "Wakeup"- schedule table (execution one time, high priority 10) c) "Node-01-init"- schedule table (execution one time, high priority 9) d) "Node-02-init"- schedule table (execution one time, high priority 8) Sequence b) c) d) a) a) a)... Example re-init of a node after a node-reset e)"Run"- schedule table (execution continuously, low priority 1) f) "Node-02-init"- schedule table (execution one time, high priority 8) Sequence ... a) a) f) a) a) ...
Dependencies:	–
Supporting Material:	–

⌋

[SRS_Lin_01546] The LIN Interface shall contain a Schedule Table Handler for LIN master nodes.

Upstream requirements: RS_BRF_01768, RS_BRF_01592

⌈

Description:	This requirement is only applicable to LIN master nodes. The schedule table handler will handle the transmission and reception of the LIN-PDUs on the LIN bus. It will query the Schedule table manager when active schedule table has reached the point to start transmission or reception of the LIN-PDU (i.e. when a schedule entry is due). The schedule table handler shall notify the upper layer about a successful / erroneous LIN-PDU transfer / reception through callbacks. The ISO 17987 specifications define that the change of a schedule table occur at the end of a timeslot. The recommendation: A schedule table change from a "continuously execution schedule table" to a "execution one time schedule table" should occur at the next timeslot. A schedule table change from a "execution one time schedule table" to a "continuously execution schedule table" or other "execution one time schedule table" should occur at the end of the current schedule table.
Rationale:	–
Use Case:	–
Dependencies:	[SRS_Lin_01564]
Supporting Material:	ISO/TR 17987-5 [8], clause 4.3.6.1 (LIN API l_sch_tick)

⌋

[SRS_Lin_01561] The LIN Interface shall define a main function per channel

Upstream requirements: RS_BRF_01768, RS_BRF_01056

⌈

Description:	For LIN master nodes, the main function is responsible for executing the schedule table handler for the corresponding channel. For LIN slave nodes, the main function is responsible for timeout observation for the corresponding channel.
Rationale:	–
Use Case:	If an ECU is connected to three LIN buses there are three main functions that execute corresponding schedule table for each channel.
Dependencies:	[SRS_Lin_01546]
Supporting Material:	–

⌋

[SRS_Lin_01549] The LIN Interface needs to use a timer service for scheduling for LIN master nodes

Upstream requirements: RS_BRF_01768, RS_BRF_01592

⌈

Description:	This requirement is only applicable to LIN master nodes. The LIN Interface needs to use a timer service for scheduling. The LIN-PDU transmission and reception must be transported at the right time. The main function is taking care of the schedule handler, so it means that this function must be called with a given period.
Rationale:	To uphold normal communication.
Use Case:	–
Dependencies:	[SRS_Lin_01561]
Supporting Material:	This is adequate to the "time base" that is defined in the ISO 17987-2 [6], clause 12 LIN description file (LDF).

⌋

[SRS_Lin_01571] Transmission request service shall be provided

Upstream requirements: RS_BRF_01768, RS_BRF_01592, RS_BRF_01544

⌈

Description:	The LIN Interface shall provide a transmission request service. For LIN master nodes, this service allows an upper layer to request the LIN interface for a Sporadic LIN-PDU transmission. The LIN Interface transmits the Sporadic LIN-PDU according to the schedule-table rules. For LIN slave nodes, this service allows an upper layer to request the LIN interface for an unconditional LIN response transmission that is allocated to an event-triggered frame. The LIN Interface transmits the LIN response after the LIN header of the event-triggered frame (to which the LIN-PDU is allocated to) was received.
Rationale:	To enable the Sporadic and Event-triggered LIN-PDU behavior in AUTOSAR
Use Case:	–
Dependencies:	–
Supporting Material:	ISO 17987-3 [3], clause 5.2.4.3 Event-triggered frame and clause 5.2.4.4 Sporadic frame

⌋

[SRS_Lin_01514] The LIN Interface shall inform an upper layer about wake-up events

Upstream requirements: RS_BRF_01768, RS_BRF_01104, RS_BRF_01064

⌈

Description:	The LIN Interface shall inform an upper layer if a wake-up request was notified by the underlying LIN Driver.
Rationale:	Basic functionality
Use Case:	Wakeup of ECU by LIN. Inform upper layer (ECU State Manager) about the wakeup reason
Dependencies:	–
Supporting Material:	ECU state manager

⌋

[SRS_Lin_01515] The LIN Interface shall provide an API to wake-up a LIN channel cluster

Upstream requirements: RS_BRF_01768, RS_BRF_01104

⌈

Description:	The LIN Interface shall provide an API to wake-up a LIN channel cluster. The LIN Interface shall support that each LIN channel cluster can be woken up separately.
Rationale:	Wake-up of LIN by upper layer.
Use Case:	–
Dependencies:	–
Supporting Material:	ISO/TR 17987-5 [8], clause 4.3.7.4 (LIN API l_ifc_wake_up) and ISO 17987-3 [3], clause 5.3 Wake up

⌋

[SRS_Lin_01502] The LIN Interface shall support an API for RX/TX notifications.

Upstream requirements: RS_BRF_01768, RS_BRF_01064, RS_BRF_01544

⌈

Description:	The PDU router provides APIs for RX notification and TX confirmation. The LIN Interface shall use this API.
Rationale:	This allows a clear interface to the upper layers (PDU router).
Use Case:	LIN node implementation with e.g. gateway to other bus systems.
Dependencies:	–

▽

△

Supporting Material:	–

⌋

[SRS_Lin_01558] The LIN Interface shall check for successful data transfer for LIN master nodes

Upstream requirements: RS_BRF_01768

⌈

Description:	This requirement is only applicable to LIN master nodes. The LIN Interface shall query the LIN driver if the last message is successful transmitted or received on the LIN bus. This check shall be done by the schedule table handler. When the successful communication*) is detected the appropriate layer above shall be notified. The schedule table handler may also check if the LIN-PDU violates the maximum frame length. It is however not recommended since the overhead is too big and that all nodes in a LIN channel cluster shall conform to the ISO 17987-6 [9] (Data Link Layer (DLL), Network Management and Node Configuration (NMNC) test specification). *) ISO 17987-3 [3], clause 5.5 Status Management, "Successful_transfer" shall be set when a frame has been successfully transferred by the node, i.e. a frame has either been received or transmitted.
Rationale:	–
Use Case:	The normal implementation would be to make the check in the main function called periodically after the LIN-PDU has been sent.
Dependencies:	–
Supporting Material:	–

⌋

[SRS_Lin_01595] The LIN Interface shall support the setting and clearing of the response error signal for LIN slave nodes.

Upstream requirements: RS_BRF_01768

⌈

Description:	This requirement is only applicable to LIN slave nodes. For LIN slave nodes, the LIN interface shall set and clear the response error signal according to the conditions described by the LIN standard.
Rationale:	Implementation of the LIN standard.

▽

△

Use Case:	Basic functionality and support of LIN conformance test. The LIN master node shall be informed about communication errors detected by the slave node.
Dependencies:	–
Supporting Material:	ISO 17987-3 [3] specification

⌋

4.1.2.4 Shutdown Operation

[SRS_Lin_01523] There shall be an API call to set the LIN bus to sleep-mode.

Upstream requirements: RS_BRF_01768, RS_BRF_01104

⌈

Description:	The LIN Interface shall provide an API to set the sleep state on each LIN bus independently of each other. LIN Master nodes send a go-to-sleep command when entering sleep state.
Rationale:	Basic functionality
Use Case:	–
Dependencies:	–
Supporting Material:	ISO 17987 specifications [1]

⌋

[SRS_Lin_01596] The LIN Interface shall provide bus idle condition observation for slave nodes.

Upstream requirements: RS_BRF_01768, RS_BRF_01104

⌈

Description:	This requirement is only applicable to LIN slave nodes. For LIN slave nodes, the LIN interface shall observe each LIN bus for communication. If no communication occurs for a configurable time, the LIN interface shall inform the upper layer about the sleep state of the LIN bus.
Rationale:	Implementation of the LIN standard.

▽

△

Use Case:	Basic functionality and support of LIN conformance test. If communication with the master node is not possible anymore, e.g. due to a defect master node or disturbed or broken bus line, the slave node shall transit to sleep state.
Dependencies:	–
Supporting Material:	ISO 17987-2 [6] specification

⌋

4.1.2.5 Fault Operation

none

4.1.3 LIN Driver

4.1.3.1 General requirements

[SRS_Lin_01578] It shall be compatible to LIN Datalinklayer

Upstream requirements: RS_BRF_01768

⌈

Description:	The frame processor has to be emulated by LIN Driver if not already supported by hardware e.g. LIN Controller The Task Behavior Model (clause 5.4 in the ISO 17987-3 [3]) is part of the LIN Driver. The ISO 17987 specifications cover the behavior of previous/non-ISO versions of LIN specifications.
Rationale:	Basic LIN functionality
Use Case:	A device driver using an UART will implement the complete Task Behavior Model. If a LIN Hardware (e.g. LIN controller) is used, parts of the Task Behavior Model runs in hardware
Dependencies:	–
Supporting Material:	ISO 17987-3 [3]

⌋

[SRS_Lin_01553] The LIN Driver shall fulfill the general SPAL requirements for Basic Software Modules.

Upstream requirements: RS_BRF_01000

⌈

Description:	The LIN Driver shall fulfill the general SPAL requirements for Software Modules as specified in AUTOSAR RS SPAL General [10]
Rationale:	Re-use of requirements valid for all low level Drivers
Use Case:	LIN Driver is in the same layer as the SPAL Drivers (e.g.: SPI, ADC). Therefore the general SPAL requirements shall be fulfilled by the LIN Driver also, if applicable.
Dependencies:	–
Supporting Material:	AUTOSAR General Requirements on SPAL [10]

⌋

[SRS_Lin_01552] The LIN Driver shall offer a Hardware independent interface.

Upstream requirements: RS_BRF_01000

⌈

Description:	The Interface from LIN Interface to LIN Driver shall be independent from underlying hardware.
Rationale:	Portability
Use Case:	Same LIN Interface can be used for different μCs.
Dependencies:	–
Supporting Material:	–

⌋

[SRS_Lin_01503] An API shall exist that enables the LIN driver to directly copy up to 8 byte directly from/to the frame buffers.

Upstream requirements: RS_BRF_01768

⌈

Description:	AUTOSAR COM creates the frames to be sent via CAN, LIN and other busses. The frames are transferred "as a block" to the lower layer. The CAN/LIN layers have therefore a need for an efficient read/write access of whole frame buffers (1 to 8 bytes).
Rationale:	Same behavior for AUTOSAR COM independently if reception/transmission is CAN or LIN based.
Use Case:	–

▽

△

Dependencies:	–
Supporting Material:	–

⌋

[SRS_Lin_01555] The LIN driver shall have an interface to retrieve transmit / receive notifications.

Upstream requirements: RS_BRF_01768, RS_BRF_01544

⌈

Description:	For LIN master nodes, the LIN Interface shall be able to poll the LIN Driver for transmit/receive notifications. For LIN slave nodes, the transmit/receive notifications shall be indicated to LIN interface on interrupt level.
Rationale:	–
Use Case:	Basic functionality
Dependencies:	–
Supporting Material:	–

⌋

[SRS_Lin_01547] The LIN Driver shall support standard UART and LIN optimized HW

Upstream requirements: RS_BRF_01768

⌈

Description:	The LIN Driver is responsible to handle the frame according to the hardware. It should be possible to support the complete the range of hardware from implementation using an UART to a complex LIN hardware controller. Using SW UART's is out of the scope.
Rationale:	Implement a common driver interface. The LIN Driver will process the complete frame by it self.
Use Case:	–
Dependencies:	–
Supporting Material:	–

⌋

[SRS_Lin_01600] Lin Driver API for Fetching L-PDU Message State

Status: DRAFT
Upstream requirements: RS_Main_00280

⌈

Description:	The Lin Driver shall provide an API to fetch the message state of an L-PDU.
Rationale:	L-PDUs encapsulated with IEEE1722 protocol require the message state which is added to the ACF header.
Use Case:	Time Synchronous and Non-Time Synchronous Control Frames Format to Support Tunneling of CAN and LIN
Dependencies:	–
Supporting Material:	[7, IEEE1722]

⌋

4.1.3.2 Initialization

[SRS_Lin_01572] The LIN Driver shall support the initialization of each LIN channel separately

Upstream requirements: RS_BRF_01056, RS_BRF_01136

⌈

Description:	The LIN Driver shall support the initialization of each LIN channel separately. The selection of at least one static configuration set shall be done by a parameter.
Rationale:	Hardware specific initialization of the UART, LIN controller. Initiation of variables used in the LIN driver.
Use Case:	If there are e.g. 2 LIN channels than there are also 2 different configuration files.
Dependencies:	–
Supporting Material:	–

⌋

[SRS_Lin_01573] The LIN Driver shall support dynamic selection of configuration sets.

Upstream requirements: RS_BRF_01152

⌈

Description:	The LIN Driver shall support the dynamic selection of at least one static configuration set by a parameter passed via the initialization interface. The selection of the appropriate configuration set itself as well as the way to incorporate the configuration sets into the ECU (Post-Build, Pre-Compile) is not affected by this requirement.
Rationale:	Support of different configurations during runtime
Use Case:	–
Dependencies:	–
Supporting Material:	–

⌋

4.1.3.3 Normal Operation

[SRS_Lin_01563] The LIN Driver shall provide a notification for wake-up events

Upstream requirements: RS_BRF_01064, RS_BRF_01104

⌈

Description:	The LIN Driver shall notify the LIN Interface in case of a wake-up interrupt. The corresponding callback function itself is implemented inside the LIN Interface. This functionality shall only be implemented, if the LIN Hardware unit has a wake-up interrupt capability. The wake-up interrupt shall only be enabled when the channel is in sleep-mode mode. Otherwise a SynchBreak will be considered as a wake-up request.
Rationale:	Inform upper layer about the occurrence of a wake-up event
Use Case:	–
Dependencies:	[SRS_Lin_01514]
Supporting Material:	–

⌋

[SRS_Lin_01556] One LIN driver shall be able to handle more than one LIN channel

Upstream requirements: RS_BRF_01768

⌈

Description:	One LIN driver shall able to handle more than one LIN channel if the underlying hardware is equipped with more than one identical LIN controllers.
Rationale:	Portability
Use Case:	If an ECU uses two channels and it contains two identical UART hardware modules there is only one LIN driver interfacing both UARTs
Dependencies:	–
Supporting Material:	–

⌋

4.1.3.4 Shutdown Operations

[SRS_Lin_01566] Transition to sleep-mode shall be handled

Upstream requirements: RS_BRF_01768, RS_BRF_01104

⌈

Description:	After the LIN Driver is requested to be set to the sleep-mode by the appropriate function call it has to do as follows: • The LIN Driver shall activate sleep-mode as soon as possible after bus is idle. • Master nodes: After successful transmission of the go-to-sleep-command the wakeup monitoring shall be activated. • Slave nodes: After successful reception of the go-to-sleep-command or detection of the bus idle condition, the wakeup monitoring shall be activated. • After wakeup monitoring is active the sleep mode shall be set and can be read out by LINif afterwards. Each LIN channel shall be handled independently.
Rationale:	Basic functionality
Use Case:	–
Dependencies:	[SRS_Lin_01524]
Supporting Material:	–

⌋

[SRS_Lin_01524] The LIN Driver shall be able to put the LIN hardware to a reduced power operation mode if needed

Upstream requirements: RS_BRF_01768, RS_BRF_01104

⌈

Description:	When going to sleep-mode mode, the LIN Driver shall put the corresponding LIN hardware to a reduced power operation mode if supported by hardware. This command shall be possible to be activated for each channel independently. This requirement does not conflict to [SRS_Lin_01566]. This requirement [SRS_Lin_01524] enables the Power Mode in the LIN hardware the other requirement [SRS_Lin_01566] sets the LIN driver in sleep-mode mode.
Rationale:	Power saving
Use Case:	–
Dependencies:	[SRS_Lin_01566]
Supporting Material:	–

⌋

4.1.3.5 Fault Operation

[SRS_Lin_01526] The LIN Driver shall provide a status for error events on the bus.

Upstream requirements: RS_BRF_01768

⌈

Description:	The LIN driver shall provide an API that returns the errors detected in the LIN communication. When the call is made the error-flags shall be reset. Each LIN channel shall be capable to notify its errors separately to the LIN interface
Rationale:	Bus error handling
Use Case:	–
Dependencies:	–
Supporting Material:	Similar to the l_ifc_read_status function (ISO/TR 17987-5 [8], clause 4.3.7.9)

⌋

4.1.4 LIN Transceiver Driver

4.1.4.1 General Requirements

4.1.4.2 Configuration

[SRS_Lin_01580] The LIN Transceiver Driver shall support separate configuration parameters per bus

Upstream requirements: RS_BRF_01768

⌈

Description:	The bus transceiver driver shall offer configuration parameters that are needed to configure the driver for a given bus and the supported notifications Typical parameters are: - Wakeup by bus - Transceiver control via SPI or port pin - Call context of the notification functions (ISR, polling) to enable detection of necessary data consistency mechanisms during configuration time Please refer to the corresponding software specification for a more detailed view
Rationale:	Basic functionality for transceiver configuration.
Use Case:	–
Dependencies:	–
Supporting Material:	–

⌋

[SRS_Lin_01581] The LIN transceiver driver shall support the configuration for more than one channel

Upstream requirements: RS_BRF_01768

⌈

Description:	The driver shall be able to support multiple LIN busses on the ECU. It must be possible to configure the used transceiver type independently for each bus. Only Pre-Compile-Time configuration shall be possible Transceiver handling depends strongly on the used device. Therefore each transceiver may need its own implementation within the driver and only known and supported devices could be selected. A general solution for the transceiver driver for all use cases might not be possible. By default each LIN controller is attached to an own bus and needs therefore an own bus transceiver.
Rationale:	Basic functionality for transceiver configuration
Use Case:	Multi bus systems, e.g. LIN-LIN gateways
Dependencies:	–
Supporting Material:	–

⌋

4.1.4.3 Initialization

[SRS_Lin_01583] The LIN Transceiver Driver shall provide an API for initialization.

Upstream requirements: RS_BRF_01768, RS_BRF_01136

⌈

Description:	The bus transceiver driver shall provide an API to initialize the driver internally and set then all attached transceivers in their pre-selected operation modes The driver must be initialized during the power-up/reset sequence of the ECU. Depending on the used drivers to control the transceivers (e.g. DIO, SPI), they must be already available and working when the transceiver driver is initialized. The wakeup reason has to be detected and stored during the execution of the driver initialization, too
Rationale:	Set bus transceivers and driver in a pre-defined and known state
Use Case:	Basic functionality for transceiver control.

▽

△

Dependencies:	[SRS_Lin_01588] The bus transceiver driver setup information must provide the necessary configuration data to enable the generation tool to select the appropriate control mechanism (e.g. SPI, I/O ports) and to guarantee the correct allocation of the necessary communication resources and initialization sequences.
Supporting Material:	–

⌋

4.1.4.4 Normal Operation

[SRS_Lin_01582] The bus transceiver driver API shall be synchronous.

Upstream requirements: RS_BRF_01768

⌈

Description:	The bus transceiver driver API shall execute the requested action immediately and shall deliver the result state immediately to the caller. This will ease up the implementation of wakeup and sleep concepts within the AUTOSAR BSW stack
Rationale:	Better usage of transceiver functionality in the complex AUTOSAR BSW environment.
Use Case:	Atomic transition to other operation mode; easier and better abstraction for upper layers like the ECU state manager or ComManager. Improved testability compared to asynchronous handling.
Dependencies:	ECU state manager, NM. SPAL in case a transceiver is connected via SPI
Supporting Material:	–

⌋

[SRS_Lin_01584] The bus transceiver driver shall support an API to send the addressed transceiver into its Standby mode.

Upstream requirements: RS_BRF_01768, RS_BRF_01104

⌈

Description:	Many transceivers support the transition to the Sleep mode only via the transition to Standby mode. In addition, some power concepts have the need to set the transceiver to Standby only instead of Sleep mode. Not all transceivers will support such a state. If this is true for a given device, the driver shall confirm the state transition with success
Rationale:	Implementation of ECU low power modes with wakeup via bus and internal.
Use Case:	The upper service layers agreed together with other nodes to set the bus into the sleep mode. The transceiver shall be switched now to a state where the wakeup via bus is supported and the power consumption is as low as possible for the current state of the ECU.
Dependencies:	[SRS_Lin_01585]
Supporting Material:	–

⌋

[SRS_Lin_01585] The bus transceiver driver shall support an API to send the addressed transceiver into its Sleep mode.

Upstream requirements: RS_BRF_01768, RS_BRF_01104

⌈

Description:	The transition to sleep mode will be requested with this API. Not all transceivers will support such a state. If this is true for a given device, the drive shall confirm the state transition with success
Rationale:	Implementation of ECU low power modes with wakeup via bus and internal.
Use Case:	The upper service layers agreed together with other nodes to set the bus into the sleep mode. The transceiver is already in StandBy and shall be switched to Sleep with lowest power consumption. Please note that the state sleep of the transceiver is often similar to the state "unpowered" of the ECU.
Dependencies:	[SRS_Lin_01584]
Supporting Material:	–

⌋

[SRS_Lin_01586] The bus transceiver driver shall support an API to send the addressed transceiver into its Normal mode.

Upstream requirements: RS_BRF_01768, RS_BRF_01104

⌈

Description:	All transceivers support this state due to it's the "working state"
Rationale:	Communication!
Use Case:	All communication must be enable to communicate.
Dependencies:	–
Supporting Material:	–

⌋

[SRS_Lin_01587] The LIN Transceiver Driver shall support an API to read out the current operation mode.

Upstream requirements: RS_BRF_01768

⌈

Description:	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. The current operation mode of the transceiver will be necessary for upper layers (e.g. diagnostics). The API shall always return the current state seen by the transceiver driver (this may be a locally stored state, too)
Rationale:	State access to transceiver driver
Use Case:	Check for current operational mode during development and via diagnostic command.
Dependencies:	–
Supporting Material:	–

⌋

[SRS_Lin_01588] The LIN Transceiver Driver shall support an API to read out the the reason of the last wakeup.

Upstream requirements: RS_BRF_01768, RS_BRF_01104

⌈

Description:	The bus transceiver driver shall support an API to read out the reason of the last wakeup of a specified bus within the ECU. The transceiver driver shall be able to store the local view "who has requested the wakeup: bus or internally". • Bus: The bus has caused the wakeup. • Internally: The wakeup has been caused by software • Sleep: The transceiver is in operation mode sleep and no wakeup has been occurred. • Wake pin: An edge on the wake pin of the transceiver (if present) has caused the wakeup. The wakeup reason should be "sleep" when the operation mode is not Normal and no wakeup has been occurred. When a wakeup has occurred, the API shall always return the first detected wakeup reason (e.g. if a wakeup by bus occurs and than nearly at the same time an internal wakeup, the wakeup reason is "bus".). After leaving the operation mode Normal, the wakeup reason shall be set to "sleep" again
Rationale:	Detection of wakeup reason during development and via diagnostic command. May also be used by the NM or ECU state manager.
Use Case:	–
Dependencies:	–
Supporting Material:	–

⌋

[SRS_Lin_01589] The bus transceiver driver shall support an API to enable and disable the wakeup notification for each bus separately.

Upstream requirements: RS_BRF_01768, RS_BRF_01104

⌈

Description:	To enable upper layers to command the bus transceiver safe into its standby and/or sleep state, an additional API to disable and enable the wakeup notification is necessary. If the notification is disabled, driver shall not perform the notification but store the event internally until the notification is enabled again. The notification shall then be processed immediately. It shall be possible to clear a pending wakeup event. If no further wakeup event occurs, no notification shall be performed after enabling the notification again. If a further wakeup event occurs it shall be notified
Rationale:	Safe wakeup and sleep handling.
Use Case:	All busses with a wakeup by bus are affected.
Dependencies:	–
Supporting Material:	–

⌋

4.1.5 LIN Transport Layer

4.1.5.1 Configuration

[SRS_Lin_01593] The value of LIN Transport protocol timeouts shall be statically configurable for each connection.

Upstream requirements: RS_BRF_01768, RS_BRF_01600, RS_BRF_01770

⌈

Description:	All the defined timeout of the ISO 17987-2 [6] are statically configurable for each connection. The configuration parameters shall be allowed to be of types Pre-Compile-Time, Link-Time or Post-Build-Time.
Rationale:	To adapt the timeout value to the ECU application domain.
Use Case:	The communication constraints may be totally different between a diagnostics connection and an applicative one (e.g. display data).
Dependencies:	–
Supporting Material:	ISO 17987-2 [6], clause 7.6.4. Similar to [SRS_Can_01081] for CAN TP.

⌋

4.2 Non-Functional Requirements (Qualities)

4.2.1 LIN General

None

4.2.2 LIN Interface

None

4.2.3 LIN Driver

None

4.2.4 LIN Transceiver Driver

None

4.2.5 LIN Transport Layer

4.2.5.1 General requirements

[SRS_Lin_01579] The AUTOSAR LIN Transport Layer shall be based on the Diagnostic Transport Layer for ISO 17987.

Upstream requirements: RS_BRF_01768

⌈

Description:	If no requirement is explicitly added or excluded, the implementation of the AUTOSAR LIN Transport Layer shall follow the ISO 17987 specifications (clause 7 in the ISO 17987-2 [6]). The implementation of the LIN TP is a precompiled option. LIN TP is not scalable. The ISO 17987 specifications cover ISO 14229-7 [2] and the behavior of previous versions of LIN specifications (see Annex B of ISO 17987-3 [3]).
Rationale:	Reuse of existing standards for AUTOSAR BSW.
Use Case:	The typical use-case is where a Diagnostic message is handoff from CAN to LIN through the CAN/LIN master gateway ECU.
Dependencies:	–

▽

△

Supporting Material:	ISO 17987-2 [6] ISO 17987-3 [3] ISO 14229-7 [2]

⌋

4.2.5.2 Initialization

[SRS_Lin_01540] The LIN Transport Layer shall provide an API for initialization.

Upstream requirements: RS_BRF_01768, RS_BRF_01136

⌈

Description:	The LIN Transport Layer shall support an API for initialization. This service shall initialize all global variables of the module and set all transport protocol connections in a default state. If there is an ongoing TP session it shall be immediately stopped.
Rationale:	–
Use Case:	–
Dependencies:	–
Supporting Material:	–

⌋

[SRS_Lin_01545] The LIN Transport Layer services shall not be operational before initializing the module.

Upstream requirements: RS_BRF_01768, RS_BRF_01096

⌈

Description:	Before using the transmission capabilities of the LIN Transport Layer, It shall be initialized. If it is not the case, the services have to return an error.
Rationale:	To avoid usage of the module without a complete initialization this could cause the transmission of corrupted frames.
Use Case:	–
Dependencies:	–
Supporting Material:	–

⌋

4.2.5.3 Normal Operation

[SRS_Lin_01534] The AUTOSAR LIN Transport Layer shall support half-duplex physical connections.

Upstream requirements: RS_BRF_01768

⌈

Description:	The LIN TP shall support the transmission/reception of one physical request/ response at a time. The next transmission, e.g. for a response or a request to another slave, is scheduled only after the previous transmission has been finished.
Rationale:	To enable diagnostic communication with simple TP implementations.
Use Case:	A tester-tool connected to an ECU on the CAN bus sends a physical diagnostic request through a CAN/LIN-master ECU to a LIN slave ECU. When the request is finished the LIN slave ECU transmits a TP message with the response to the diagnostic request.
Dependencies:	–
Supporting Material:	–

⌋

[SRS_Lin_01592] The AUTOSAR LIN Transport Layer shall support the transmission of functional requests at any time for master nodes.

Upstream requirements: RS_BRF_01768

⌈

Description:	This requirement is only applicable to LIN master nodes. The LIN TP shall support the transmission of a functional request at any time.
Rationale:	Implementation of the LIN standard.
Use Case:	A tester-tool connected to an ECU on the CAN bus sends a functional diagnostic request through a CAN/LIN-master ECU to a LIN slave ECU.
Dependencies:	–
Supporting Material:	–

⌋

[SRS_Lin_01574] It shall be possible to have one instance of the TP for each channel

Upstream requirements: RS_BRF_01768

⌈

Description:	It shall be possible to have one instance of the TP for each channel.
Rationale:	Since the only frames that can be used for TP on LIN are the MRF and SRF it is not possible to have more than one instance of a TP message on each LIN channel.
Use Case:	–
Dependencies:	–
Supporting Material:	–

⌋

[SRS_Lin_01539] The Transport connection properties shall be statically configured.

Upstream requirements: RS_BRF_01768

⌈

Description:	The LIN Transport connection configuration shall statically assign properties of each N-SDU: • Its unique handle (N_SDU_Handle) • Minimum length of the N_SDU • Associated N_PDU handle (N_PDU_Handle) • Physical (1 to 1 communication) addressing • Direction type: full-duplex or half-duplex communication • Addressing mode: standard
Rationale:	At runtime the LIN Transport module shall have all the needed information to manage a transport connection.
Use Case:	This information can be used at generation time to check the network LIN cluster configuration with a TP point of view.
Dependencies:	–
Supporting Material:	Similar to [SRS_Can_01074] for CAN TP

⌋

[SRS_Lin_01591] The LIN Interface shall take care of the behavior of the master node transmission handler for master nodes.

Upstream requirements: RS_BRF_01768

⌈

Description:	This requirement is only applicable to LIN master nodes. A transmission handler shall be implemented by the master node according to the clause 9.7.2 in the ISO 17987-2 [6]. This describes the following modes: • Idle state • Tx functional active • Tx physical active • Rx physical active • Interleaved functional during Tx • Interleaved functional during Rx
Rationale:	The LIN Transport layer module shall have the specified needed information to manage a transport connection.
Use Case:	This information can be used at generation time to check the network configuration with a TP point of view.
Dependencies:	–
Supporting Material:	ISO 17987-2 [6], clause 9.6 and 9.7.2

⌋

[SRS_Lin_01597] The Lin Interface shall take care of the behavior of the slavenode transmission handler for slave nodes.

Upstream requirements: RS_BRF_01768

⌈

Description:	This requirement is only applicable to LIN slave nodes. A transmission handler shall be implemented by the slave node according to the chapter 9.7.3 in the ISO 17987-2 [6] specification. This describes the following modes: • Idle state • Receive physical request • Transmit physical response • Receive functional request
Rationale:	The LIN Transport layer module shall have the specified needed information to manage a transport connection.
Use Case:	This information can be used at generation time to check the network configuration with a TP point of view.

▽

△

Dependencies:	–
Supporting Material:	ISO 17987-2 [6] specification, clause 9.7.3

⌋

4.2.5.4 Fault Operation

[SRS_Lin_01544] Errors shall be handled

Upstream requirements: RS_BRF_01768

⌈

Description:	In case of reception of unexpected N_PDU it shall respect the behavior defined in clause 7.6.4 (unexpected arrival of N_PDU) of the ISO 17987-2 [6]. For others errors, just aborts the segmentation session.
Rationale:	This is an extension to the LIN specification since it does not describe how to handle error situations occurred during transportation of a TP message.
Dependencies:	–
Supporting Material:	ISO 17987-2 [6], clause 7.6.4.

⌋

5 Requirements Tracing

The following table references the features specified in [11] and links to the fulfillments of these.

Requirement	Description	Satisfied by
[RS_BRF_01000]	AUTOSAR architecture shall organize the BSW in a hardware independent and a hardware dependent layer	[SRS_Lin_01552] [SRS_Lin_01553] [SRS_Lin_01568]
[RS_BRF_01056]	AUTOSAR BSW modules shall provide standardized interfaces	[SRS_Lin_01561] [SRS_Lin_01572]
[RS_BRF_01064]	AUTOSAR BSW shall provide callback functions in order to access upper layer modules	[SRS_Lin_01502] [SRS_Lin_01514] [SRS_Lin_01563]
[RS_BRF_01096]	AUTOSAR shall support start-up and shutdown of ECUs	[SRS_Lin_01545] [SRS_Lin_01569]
[RS_BRF_01104]	AUTOSAR shall support sleep and wake-up of ECUs and buses	[SRS_Lin_01514] [SRS_Lin_01515] [SRS_Lin_01523] [SRS_Lin_01524] [SRS_Lin_01560] [SRS_Lin_01563] [SRS_Lin_01566] [SRS_Lin_01584] [SRS_Lin_01585] [SRS_Lin_01586] [SRS_Lin_01588] [SRS_Lin_01589] [SRS_Lin_01596]
[RS_BRF_01136]	AUTOSAR shall support variants of configured BSW data resolved after system start-up	[SRS_Lin_01540] [SRS_Lin_01569] [SRS_Lin_01570] [SRS_Lin_01572] [SRS_Lin_01583]
[RS_BRF_01152]	AUTOSAR shall support limited dynamic reconfiguration	[SRS_Lin_01573]
[RS_BRF_01544]	AUTOSAR communication shall define transmission and reception of communication data	[SRS_Lin_01502] [SRS_Lin_01555] [SRS_Lin_01571]
[RS_BRF_01592]	AUTOSAR communication shall offer data transfer on user request, time based, and requested via the underlying bus	[SRS_Lin_01546] [SRS_Lin_01549] [SRS_Lin_01564] [SRS_Lin_01571]
[RS_BRF_01600]	AUTOSAR communication shall support time-out handling	[SRS_Lin_01593]
[RS_BRF_01768]	AUTOSAR communication shall support LIN	[SRS_Lin_01502] [SRS_Lin_01503] [SRS_Lin_01504] [SRS_Lin_01514] [SRS_Lin_01515] [SRS_Lin_01522] [SRS_Lin_01523] [SRS_Lin_01524] [SRS_Lin_01526] [SRS_Lin_01534] [SRS_Lin_01539] [SRS_Lin_01540] [SRS_Lin_01544] [SRS_Lin_01545] [SRS_Lin_01546] [SRS_Lin_01547] [SRS_Lin_01549] [SRS_Lin_01551] [SRS_Lin_01555] [SRS_Lin_01556] [SRS_Lin_01558] [SRS_Lin_01560] [SRS_Lin_01561] [SRS_Lin_01564] [SRS_Lin_01566] [SRS_Lin_01568] [SRS_Lin_01569] [SRS_Lin_01570] [SRS_Lin_01571] [SRS_Lin_01574] [SRS_Lin_01576] [SRS_Lin_01577] [SRS_Lin_01578] [SRS_Lin_01579] [SRS_Lin_01580] [SRS_Lin_01581] [SRS_Lin_01582] [SRS_Lin_01583] [SRS_Lin_01584] [SRS_Lin_01585] [SRS_Lin_01586] [SRS_Lin_01587] [SRS_Lin_01588] [SRS_Lin_01589] ▽

▽

△

Requirement	Description	Satisfied by
		△ [SRS_Lin_01590] [SRS_Lin_01591] [SRS_Lin_01592] [SRS_Lin_01593] [SRS_Lin_01594] [SRS_Lin_01595] [SRS_Lin_01596] [SRS_Lin_01597] [SRS_Lin_01598]
[RS_BRF_01770]	AUTOSAR communication shall support LIN transport protocol	[SRS_Lin_01593]
[RS_Main_00280]	Standardized Automotive Communication Protocols	[SRS_Lin_01599] [SRS_Lin_01600]

Table 5.1: Requirements Tracing

6 References

[1] ISO 17987:2016 (all parts), Road vehicles – Local Interconnect Network (LIN)
https://www.iso.org

[2] ISO 14229-7:2015 Road vehicles – Unified diagnostic services (UDS) – Part 7: UDS on local interconnect network (UDSonLIN)
https://www.iso.org

[3] ISO 17987-3:2016 Road vehicles – Local Interconnect Network (LIN) – Part 3: Protocol specification
https://www.iso.org

[4] Standardization Template
AUTOSAR_FO_TPS_StandardizationTemplate

[5] Glossary
AUTOSAR_FO_TR_Glossary

[6] ISO 17987-2:2016 Road vehicles – Local Interconnect Network (LIN) – Part 2: Transport protocol and network layer services
https://www.iso.org

[7] IEEE Standard 1722-2016 - IEEE Standard for a Transport Protocol for Time-Sensitive Applications in Bridged Local Area Networks

[8] ISO/TR 17987-5:2016 Road vehicles – Local Interconnect Network (LIN) – Part 5: Application programmers interface (API)
https://www.iso.org

[9] ISO 17987-6:2016 Road vehicles – Local Interconnect Network (LIN) – Part 6: Protocol conformance test specification
https://www.iso.org

[10] General Requirements on SPAL
AUTOSAR_CP_RS_SPALGeneral

[11] Requirements on AUTOSAR Features
AUTOSAR_CP_RS_Features

Note: Non-ISO LIN specifications (LIN 2.2A and LIN 2.1 by LIN Consortium) are available at <https://www.lin-cia.org/standards/>, even after closure of the LIN Consortium.

A Change history of AUTOSAR traceable items

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

A.1 Traceable item history of this document according to AUTOSAR Release R23-11

A.1.1 Added Requirements in R23-11

none

A.1.2 Changed Requirements in R23-11

none

A.1.3 Deleted Requirements in R23-11

none

A.2 Traceable item history of this document according to AUTOSAR Release R24-11

A.2.1 Added Requirements in R24-11

Number	Heading
[SRS_Lin_01599]	LinIf Forwarding of L-PDUs to LSduR
[SRS_Lin_01600]	Lin Driver API for Fetching L-PDU Message State

Table A.1: Added Requirements in R24-11

A.2.2 Changed Requirements in R24-11

none

A.2.3 Deleted Requirements in R24-11

none