

# MICROSAR LIN Driver

## Technical Reference

VttCanoe

Version 3.0.0

Authors	Lutz Pflüger
Status	Released

## Document Information

### History

Author	Date	Version	Remarks
Lutz Pflüger	2013-10-28	1.0.0	Migrate VipCanoe from Canoeemu
Lutz Pflüger	2015-02-03	2.0.0	R12, rename VIP to VTT
Lutz Pflüger	2016-04-15	3.0.0	Migrate Core to VttCanoe Technical Reference (Vtt Standalone Technical Reference)

### Reference Documents

No.	Source	Title	Version
[1]	AUTOSAR	AUTOSAR_SWS_LINDriver.pdf	2.2.0
[2]	AUTOSAR	AUTOSAR_SWS_DevelopmentErrorTracer.pdf	3.2.0
[3]	AUTOSAR	AUTOSAR_SWS_DiagnosticEventManager.pdf	4.2.0
[4]	AUTOSAR	AUTOSAR_BasicSoftwareModules.pdf	V1.0.0

### Scope of the Document

This technical reference describes the general use of the LIN driver basis software.



#### Caution

We have configured the programs in accordance with your specifications in the questionnaire. Whereas the programs do support other configurations than the one specified in your questionnaire, Vector's release of the programs delivered to your company is expressly restricted to the configuration you have specified in the questionnaire.

## Contents

<b>1. Deviations of VTT .....</b>	<b>5</b>
<b>2. Introduction .....</b>	<b>6</b>
2.1 Architecture Overview .....	6
<b>3. Functional Description .....</b>	<b>8</b>
3.1 Features.....	8
3.1.1 Deviations .....	8
3.2 Initialization .....	9
3.3 Wake up handling.....	9
3.4 Sleep handling .....	9
3.5 Error Handling .....	9
3.5.1 Development Error Reporting.....	9
<b>4. Integration .....</b>	<b>11</b>
4.1 Scope of Delivery .....	11
4.1.1 Static Files .....	11
4.1.2 Dynamic Files .....	11
4.2 Critical Sections .....	11
<b>5. API Description .....</b>	<b>12</b>
5.1 Type Definitions.....	12
5.2 Services provided by LIN.....	14
5.2.1 Lin_InitMemory.....	14
5.2.2 Lin_Init .....	15
5.2.3 Lin_GetVersionInfo.....	15
5.2.4 Lin_SendFrame.....	15
5.2.5 Lin_GoToSleep.....	16
5.2.6 Lin_GoToSleepInternal.....	16
5.2.7 Lin_Wakeup .....	17
5.2.8 Lin_WakeupInternal .....	17
5.2.9 Lin_CheckWakeup .....	18
5.2.10 Lin_GetStatus .....	18
5.3 Services used by LIN .....	19
<b>6. Glossary and Abbreviations .....</b>	<b>20</b>
6.1 Glossary.....	20
6.2 Abbreviations .....	20
<b>7. Contact.....</b>	<b>21</b>

## Illustrations

Figure 2-1	AUTOSAR 4.1 Architecture Overview .....	6
Figure 2-2	Interfaces to adjacent modules of the LIN.....	7

## Tables

Table 3-1	Supported AUTOSAR standard conform features .....	8
Table 3-2	Not supported AUTOSAR standard conform features .....	8
Table 3-3	Service IDs .....	10
Table 3-4	Errors reported to DET .....	10
Table 4-1	Static files .....	11
Table 4-2	Generated files .....	11
Table 5-1	Type definitions.....	13
Table 5-2	Lin_PduType.....	14
Table 5-3	Lin_InitMemory .....	14
Table 5-4	Lin_Init.....	15
Table 5-5	Lin_GetVersionInfo .....	15
Table 5-6	Lin_SendFrame .....	16
Table 5-7	Lin_GoToSleep .....	16
Table 5-8	Lin_GoToSleepInternal .....	17
Table 5-9	Lin_Wakeup.....	17
Table 5-10	Lin_WakeupInternal.....	18
Table 5-11	Lin_CheckWakeup.....	18
Table 5-12	Lin_GetStatus.....	19
Table 5-13	Services used by the LIN.....	19
Table 6-1	Glossary .....	20
Table 6-2	Abbreviations.....	20

## 1. Deviations of VTT

**Caution**

Significant functional limitations regarding network management. It is possible that the network modes of Lin Driver and CANoe are not synchrony.

**Caution**

Significant functional limitations of LIN Bus disturbing

- > Frames are queued until Bus disturbing stops
- > Disturbing Go-To-Sleep Command leads to not synchrony network management states

**Caution**

A Frame Header with ID 0 on start measurement after ca. 100ms on Bus.

**Caution**

Sending Frames within the first 100ms after start measurement are not working.

## 2. Introduction

This document describes the functionality, API and configuration of the AUTOSAR BSW module LIN as specified in [1].

<b>Supported AUTOSAR Release*:</b>	4	
<b>Vendor ID:</b>	LIN_VENDOR_ID	30 decimal (= Vector-Informatik, according to HIS)
<b>Module ID:</b>	LIN_MODULE_ID	82 decimal (according to ref. [4])

\* For the precise AUTOSAR Release 4.x please see the release specific documentation.

### 2.1 Architecture Overview

The following figure shows where the LIN is located in the AUTOSAR architecture.

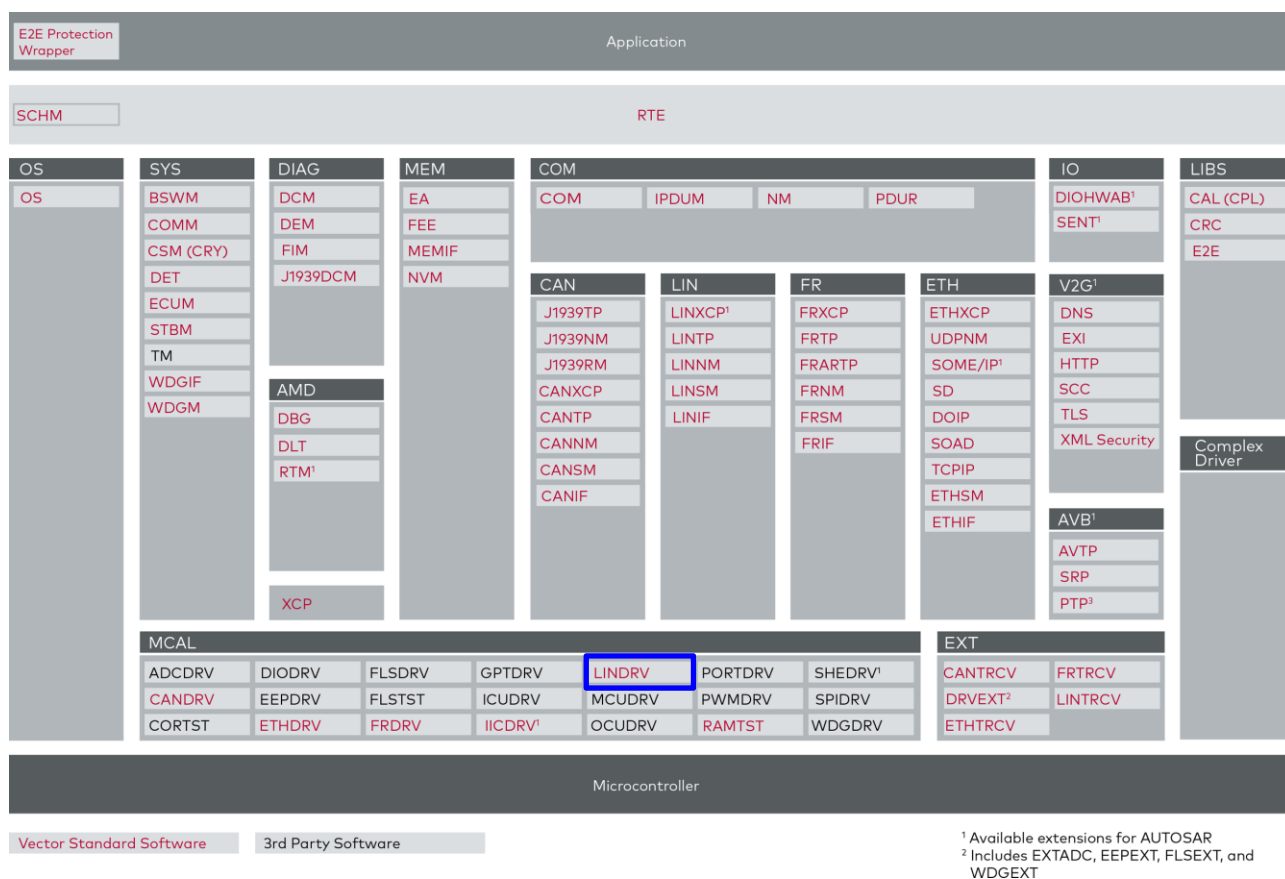


Figure 2-1 AUTOSAR 4.1 Architecture Overview

The next figure shows the interfaces to adjacent modules of the LIN. These interfaces are described in chapter 5.

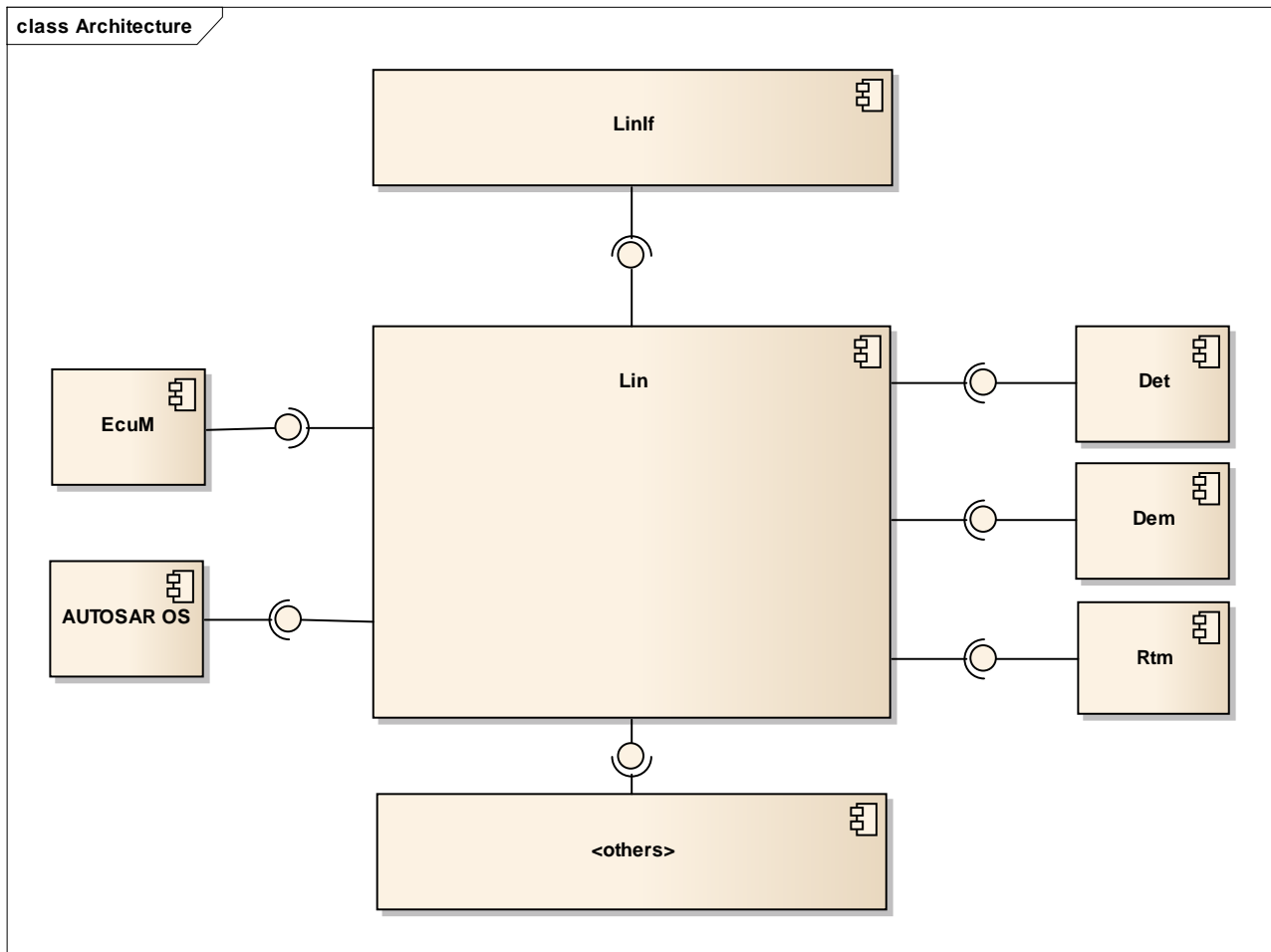


Figure 2-2 Interfaces to adjacent modules of the LIN

### 3. Functional Description

#### 3.1 Features

The features listed in the following tables cover the complete functionality specified for the LIN.

The AUTOSAR standard functionality is specified in [1], the corresponding features are listed in the tables

- > Table 3-1 Supported AUTOSAR standard conform features
- > Table 3-2 Not supported AUTOSAR standard conform features

The following features specified in [1] are supported:

Supported AUTOSAR Standard Conform Features
LIN Master node
Reuse of LIN 2.1 specification
Service to initialize itself
Service to send a frame on channel
Service to provide the frame status
Service to access the version information
Provide a channel operational state
Provide a channel sleep state
Post-build loadable
Post-build selectable
Multi channel usage

Table 3-1 Supported AUTOSAR standard conform features

##### 3.1.1 Deviations

The following features specified in [1] are not supported:

Not Supported AUTOSAR Standard Conform Features
Multiple LIN Driver
Service to set the channel to sleep and optionally transmit a go-to-sleep-command
Service to set the channel to wake and optionally generate a wake up pulse
Service to evaluate a wake up event of channel
Callout to report a wake up event of channel

Table 3-2 Not supported AUTOSAR standard conform features



## 3.2 Initialization

After power on the LIN hardware has to be initialized. Therefore the LIN Driver provides two service functions.

- > `Lin_Init()` has to be called to initialize the LIN driver at power on.
- > `Lin_InitMemory()` is an additional service function to reinitialize the memory to bring the driver back to a pre-power-on state (not initialized). Afterwards `Lin_Init()` have to be called again. It is recommended to use this function before calling `Lin_Init()` to secure that no startup-code specific pre-initialized variables affect the driver startup behavior.

## 3.3 Wake up handling

Wakeup frame handling is only applicable in state sleep. A wakeup frame can be transmitted by calling the function `Lin_Wakeup()`. When calling `Lin_WakeupInternal()` no wakeup frame is transmitted. If a wakeup frame is received the EcuM is informed by calling the function `EcuM_CheckWakeupEvent()`.

## 3.4 Sleep handling

Sleep mode frame handling is only applicable in state wake. A sleep mode frame can be transmitted by calling the function `Lin_GoToSleep()`. When calling `Lin_GoToSleepInternal()` the LIN Driver enters sleep mode without transmitting a sleep mode frame.

## 3.5 Error Handling

### 3.5.1 Development Error Reporting

By default, development errors are reported to the DET using the service `Det_ReportError()` as specified in [2], if development error reporting is enabled (i.e. pre-compile parameter `LIN_DEV_ERROR_DETECT==STD_ON`).

If another module is used for development error reporting, the function prototype for reporting the error can be configured by the integrator, but must have the same signature as the service `Det_ReportError()`.

The reported LIN ID is 82.

The reported service IDs identify the services which are described in 0. The following table presents the service IDs and the related services:

Service ID	Service
0x00	<code>Lin_Init</code>
0x01	<code>Lin_GetVersionInfo</code>
0x04	<code>Lin_SendFrame</code>
0x06	<code>Lin_GoToSleep</code>
0x07	<code>Lin_Wakeup</code>

Service ID	Service
0x08	Lin_GetStatus
0x09	Lin_GoToSleepInternal
0x0A	Lin_CheckWakeup
0x0B	Lin_WakeupInternal
0x90	Lin_Interrupt

Table 3-3 Service IDs

The errors reported to DET are described in the following table:

Error Code	Description
0x00 LIN_E_UNINIT	API service used without module initialization.
0x02 LIN_E_INVALID_CHANNEL	API service used with an invalid or inactive channel parameter.
0x03 LIN_E_INVALID_POINTER	API service called with invalid configuration pointer
0x04 LIN_E_STATE_TRANSITION	Invalid state transition for the current state
0x05 LIN_E_PARAM_POINTER	API service called with a NULL pointer
0x06 LIN_E_PARAM_VALUE	API service called with invalid parameter value

Table 3-4 Errors reported to DET

## 4. Integration

This chapter gives necessary information for the integration of the MICROSAR LIN into an application environment of an ECU.

### 4.1 Scope of Delivery

The delivery of the LIN contains the files which are described in the chapters 4.1.1 and 4.1.2:

#### 4.1.1 Static Files

File Name	Description
Lin.h	Header containing the interface of the LIN Driver.
Lin.c	C code containing the functionality of the LIN Driver.
Lin_GeneralTypes.h	Header containing commonly used type definitions of the LIN cluster.
Lin_Types.h	Header containing internal type definitions of the LIN Driver.

Table 4-1 Static files

#### 4.1.2 Dynamic Files

The dynamic files are generated by the configuration tool.

File Name	Description
Lin_Cfg.h	Generated header adapting the LIN Driver to project requirements.
Lin_DrvGeneralTypes.h	Generated commonly used type definitions of the LIN cluster.
Lin_Lcfg.c	Generated C code containing tables with link time variables.
Lin_PBcfg.c	Generated C code containing tables with post build variables.

Table 4-2 Generated files

## 4.2 Critical Sections

The MICROSAR LIN Driver is either running in interrupt context or is called from LinIf. The LinIf already prevents from interruption by means of exclusive areas. Thus no exclusive area handling is done within the LIN Driver.

## 5. API Description

For an interfaces overview please see Figure 2-2.

### 5.1 Type Definitions

The types defined by the LIN are described in this chapter.

Type Name	C-Type	Description	Value Range
Lin_u8PtrType	uint8	Pointer to a uint8 variable. Use for 'uint8**' definition in Lin_GetStatus	not applicable
Lin_StatusType	enum	LIN operation states for a LIN channel or frame, as returned by the API service Lin_GetStatus().	LIN_NOT_OK LIN frame operation return value. Development or production error occurred.
			LIN_TX_OK LIN frame operation return value. Successful transmission.
			LIN_TX_BUSY LIN frame operation return value. Ongoing transmission (Header or Response).
			LIN_TX_HEADER_ERROR LIN frame operation return value. Erroneous header transmission such as: <ul style="list-style-type: none"> <li>&gt; Mismatch between sent and read back data</li> <li>&gt; Identifier parity error or</li> <li>&gt; Physical bus error</li> </ul>
			LIN_TX_ERROR LIN frame operation return value. Erroneous response transmission such as: <ul style="list-style-type: none"> <li>&gt; Mismatch between sent and read back data</li> <li>&gt; Physical bus error</li> </ul>
			LIN_RX_BUSY LIN frame operation return value. Ongoing reception: at least one response byte has been received, but the checksum byte has not been received.
			LIN_RX_OK

Type Name	C-Type	Description	Value Range
			LIN frame operation return value. Reception of correct response.
			LIN_RX_ERROR LIN frame operation return value. Erroneous response reception such as: <ul style="list-style-type: none"> <li>&gt; Framing error</li> <li>&gt; Overrun error</li> <li>&gt; Checksum error or</li> <li>&gt; Short response</li> </ul>
			LIN_RX_NO_RESPONSE LIN frame operation return value. No response byte has been received so far.
			LIN_OPERATIONAL LIN channel state return value. Normal operation; the related LIN channel is ready to transmit next header. No data from previous frame available (e.g. after initialization).
			LIN_CH_SLEEP LIN channel state return value. Sleep state operation; in this state wake-up detection from slave nodes is enabled.
Lin_ConfigType	struct	Global Configuration	A structure type is present for data in each configuration class.

Table 5-1 Type definitions

## Lin\_PduType

This Type is used to provide PID, checksum model, data length and SDU pointer of a LIN frame from the LIN Interface to the LIN driver.

Struct Element Name	C-Type	Description	Value Range
Pid	uint8 (Lin_FramePidType)	Valid protected identifier.	Represents all valid protected identifier used by Lin_SendFrame().
Cs	enum (Lin_FrameCsModelType)	Specified Checksum model	LIN_ENHANCED_CS Enhanced checksum model.
			LIN_CLASSIC_CS Classic checksum model.

Struct Element Name	C-Type	Description	Value Range
Drc	enum (Lin_FrameResponseType)	Type of response part	LIN_MASTER_RESPONSE Response is generated from this (master) node.
			LIN_SLAVE_RESPONSE Response is generated from a remote slave node.
			LIN_SLAVE_TO_SLAVE Response is generated from one slave to another slave.
DI	uint8 (Lin_FrameDIType)	SDU data bytes to copy	This type is used to specify the number of SDU data bytes to copy. Range: 1 - 8, data length of a LIN frame.
SduPtr	uint8*	Pointer to SDU data bytes	Valid pointer to SDU data.

Table 5-2 Lin\_PduType

## 5.2 Services provided by LIN

### 5.2.1 Lin\_InitMemory

Prototype	
void <b>Lin_InitMemory</b> (void)	
Parameter	
-	
Return code	
void	-
Functional Description	
Sets the module state to uninitialized.	
Particularities and Limitations	
Function must be called in case LIN_VAR_ZERO_INIT variables are not initialized with 0 after reset (i.e. by startup code). This service function has to be called before Lin_Init() function.	
Call Context	
Called by upper layer.	

Table 5-3 Lin\_InitMemory

## 5.2.2 Lin\_Init

Prototype	
void <b>Lin_Init</b> (const Lin_ConfigType *Config)	
Parameter	
Config	Pointer to a selected configuration structure
Return code	
void	-
Functional Description	
Initializes the LIN module channel hardware and sets the state to initialize.	
Particularities and Limitations	
This service function has to be called before any other LIN driver function. The service ID of this function is LIN_SID_INIT_ID	
Call Context	
Called by upper layer.	

Table 5-4 Lin\_Init

## 5.2.3 Lin\_GetVersionInfo

Prototype	
void <b>Lin_GetVersionInfo</b> (Std_VersionInfoType *versioninfo)	
Parameter	
versioninfo	Pointer to where to store the version information of this module.
Return code	
void	-
Functional Description	
This service returns version information as decimal, vendor ID and AUTOSAR module ID of the component.	
Particularities and Limitations	
This function shall be pre compile time configurable On/Off by the configuration parameter: LIN_VERSION_INFO_API The service ID of this function is LIN_SID_GETVERSIONINFO_ID	
Call Context	
Called by upper layer.	

Table 5-5 Lin\_GetVersionInfo

## 5.2.4 Lin\_SendFrame

Prototype	
Std_ReturnType <b>Lin_SendFrame</b> (uint8 Channel, Lin_PduType *PduInfoPtr)	
Parameter	
Channel	LIN channel to be addressed

PduInfoPtr	Pointer to PDU containing the PID, Checksum model, Response type, DI and SDU data pointer
<b>Return code</b>	
Std_ReturnType	E_OK: send command has been accepted E_NOT_OK: send command has not been accepted, development or production error occurred
<b>Functional Description</b>	
The function Lin_SendFrame generates a LIN frame on the addressed LIN channel.	
<b>Particularities and Limitations</b>	
The service ID of this function is LIN_SID_SENDFRAME_ID	
<b>Call Context</b>	
Called by upper layer.	

Table 5-6 Lin\_SendFrame

### 5.2.5 Lin\_GoToSleep

<b>Prototype</b>	
Std_ReturnType <b>Lin_GoToSleep</b> (uint8 Channel)	
<b>Parameter</b>	
Channel	LIN channel to be addressed
<b>Return code</b>	
Std_ReturnType	E_OK: Sleep command has been accepted E_NOT_OK: Sleep command has not been accepted, development or production error occurred
<b>Functional Description</b>	
Transmits a go-to-sleep command on the addressed LIN channel and sets the channel into sleep mode.	
<b>Particularities and Limitations</b>	
If supported by HW the LIN hardware unit maybe set to reduced power operation mode. The service ID of this function is LIN_SID_GOTOSLEEP_ID	
<b>Call Context</b>	
Called by upper layer.	

Table 5-7 Lin\_GoToSleep

### 5.2.6 Lin\_GoToSleepInternal

<b>Prototype</b>	
Std_ReturnType <b>Lin_GoToSleepInternal</b> (uint8 Channel)	
<b>Parameter</b>	
Channel	LIN channel to be addressed



Return code	
Std_ReturnType	E_OK: Sleep command has been accepted E_NOT_OK: Sleep command has not been accepted, development or production error occurred
Functional Description	
Sets the channel to sleep mode without sending a go-to-sleep command.	
Particularities and Limitations	
The service ID of this function is LIN_SID_GOTOSLEEPINTERNAL_ID	
Call Context	
Called by upper layer.	

Table 5-8 Lin\_GoToSleepInternal

### 5.2.7 Lin\_Wakeup

Prototype	
Std_ReturnType <b>Lin_Wakeup</b> (uint8 Channel)	
Parameter	
Channel	LIN channel to be addressed
Return code	
Std_ReturnType	E_OK: Wake-up request has been accepted E_NOT_OK: Wake-up request has not been accepted, development or production error occurred
Functional Description	
Sends a wakeup frame on the on the addressed LIN channel.	
Particularities and Limitations	
The service ID of this function is LIN_SID_WAKEUP_ID	
Call Context	
Called by upper layer.	

Table 5-9 Lin\_Wakeup

### 5.2.8 Lin\_WakeupInternal

Prototype	
Std_ReturnType <b>Lin_WakeupInternal</b> (uint8 Channel)	
Parameter	
Channel	LIN channel to be addressed
Return code	
Std_ReturnType	E_OK: Wake-up request has been accepted E_NOT_OK: Wake-up request has not been accepted, development or production error occurred
Functional Description	
Sets the channel state to LIN_CH_OPERATIONAL without generating a wake up pulse.	

Particularities and Limitations
The service ID of this function is LIN_SID_WAKEUPINTERNAL_ID
Call Context
Called by upper layer.

Table 5-10 Lin\_WakeupInternal

## 5.2.9 Lin\_CheckWakeup

Prototype	
Std_ReturnType <b>Lin_CheckWakeup</b> (uint8 Channel)	
Parameter	
Channel	LIN channel to be addressed
Return code	
Std_ReturnType	E_OK: No error has occurred during execution of the API E_NOT_OK: An error has occurred during execution of the API
Functional Description	
After a wake up caused by LIN bus transceiver or LIN driver the function Lin_CheckWakeup will be called by the LIN Interface module to identify the corresponding LIN channel.	
Particularities and Limitations	
The service ID of this function is LIN_SID_CHECKWAKEUP_ID	
Call Context	
Called by upper layer.	

Table 5-11 Lin\_CheckWakeup

## 5.2.10 Lin\_GetStatus

Prototype	
Lin_StatusType <b>Lin_GetStatus</b> (uint8 Channel, Lin_u8PtrType *Lin_SduPtr)	
Parameter	
Channel	LIN channel to be addressed
Lin_SduPtr	Pointer to pointer to shadow buffer or memory mapped LIN Hardware receive buffer
Return code	
Lin_StatusType	Lin_StatusType: Information about the current message state.
Functional Description	
The function Lin_GetStatus shall return the current transmission, reception or operation status of the LIN driver.	
Particularities and Limitations	
The service ID of this function is LIN_SID_GETSTATUS_ID	
Call Context	

Called by upper layer.

Table 5-12 Lin\_GetStatus

### 5.3 Services used by LIN

In the following table services provided by other components, which are used by the LIN are listed. For details about prototype and functionality refer to the documentation of the providing component.

Component	API
DET	Det_ReportError()
EcuM	EcuM_CheckWakeup() EcuM_SetWakeupEvent()
RTM	Rtm_Start() Rtm_Stop()
LinIf	LinIf_WakeupConfirmation()

Table 5-13 Services used by the LIN

## 6. Glossary and Abbreviations

### 6.1 Glossary

Term	Description
Channel	A channel defines the assignment (1:1) between a physical communication interface and a physical layer on which different modules are connected to.
Interrupt	Processor-specific event which can interrupt the execution of a current program section.
Transceiver	A transceiver adapts the physical layer to the communication interface.

Table 6-1 Glossary

### 6.2 Abbreviations

Term	Description
API	Application Program Interface
AUTOSAR	Automotive Open System Architecture
BSW	Basic Software
DEM	Diagnostic Event Manager
DET	Development Error Tracer
ECU	Electronic Control Unit
HW	Hardware
ID	Identifier (e.g. Identifier of a CAN message)
ISR	Interrupt Service Routine
LIN	Local Interconnect Network
MICROSAR	Microcontroller Open System Architecture (the Vector AUTOSAR solution)
RTM	Runtime Measurement
HIS	Hersteller Initiative Software

Table 6-2 Abbreviations

## 7. Contact

Visit our website for more information on

- > News
- > Products
- > Demo software
- > Support
- > Training data
- > Addresses

[www.vector.com](http://www.vector.com)