VECTOR

MICROSAR LIN Transceiver Driver

Technical Reference

Generic

Version 3.02.01

Authors	Edgar Tongoona, Nessim Ben Ammar, Jan Gaukel
Status	Released

Document Information

History

Author	Date	Version	Remarks
Edgar Tongoona	2013-09-27	2.00.00	Creation
Edgar Tongoona	2014-02-07	2.01.00	Added ICU notification
Edgar Tongoona	2014-10-10	3.00.00	Added Post-build Selectable and wakeup handling compliance to AR4.1.x,
Edgar Tongoona	2015-07-15	3.01.00	Adapted according to ESCAN00083261
Nessim Ben Ammar	2017-04-27	3.02.00	Removed all place holders and replaced with Generic/GENERIC Changed filenames of the driver Added Appl_Generic_Wait
Jan Gaukel	2018-05-04	3.02.01	Added explanation to the example

Reference Documents

No.	Source	Title	Version
[1]	AUTOSAR	AUTOSAR_SWS_LIN_TransceiverDriver.pdf	V1.2.0
[2]	AUTOSAR	AUTOSAR_SWS_DET.pdf	V3.2.0
[3]	AUTOSAR	AUTOSAR_TR_BSWModuleList.pdf	V1.6.0

Scope of the Document

This technical reference describes the specific use of the Generic LIN transceiver driver.

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

Illustrations

Tables

1 Component History

The component history gives an overview over the important milestones that are supported in the different versions of the component.

Component Version	New Features
2.00.00	first version of the module supporting AUTOSAR R4.0.3
3.00.00	Support of Post-Build-Selectable Support of external wakeup handling according to AUTOSAR R4.1.2

Table 1-1 Component history

2 Introduction

This document describes the functionality, API and configuration of the AUTOSAR BSW module LINTRCV as specified in [1]. The MICROSAR LIN Transceiver Driver provides an abstraction layer for the used LIN transceiver hardware. It offers a hardware independent interface to the upper layer components.

Supported AUTOSAR Release*:	4	
Supported Configuration Variants:	pre-compile	
Vendor ID:	LINTRCV_30_GENERIC_VENDOR_ID	30 decimal (= Vector-Informatik, according to HIS)
Module ID:	LINTRCV_30_GENERIC_MODULE_ID	64 decimal (according to ref. [3])

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

Caution
The Generic LinTrcv driver offers the standardized hardware independent part of the driver. The user will still have to add some code on his/her own to complete the driver.

There are several sections within the drive which the user can implement with hints on what must be implemented in those sections.

2.1 Architecture Overview

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

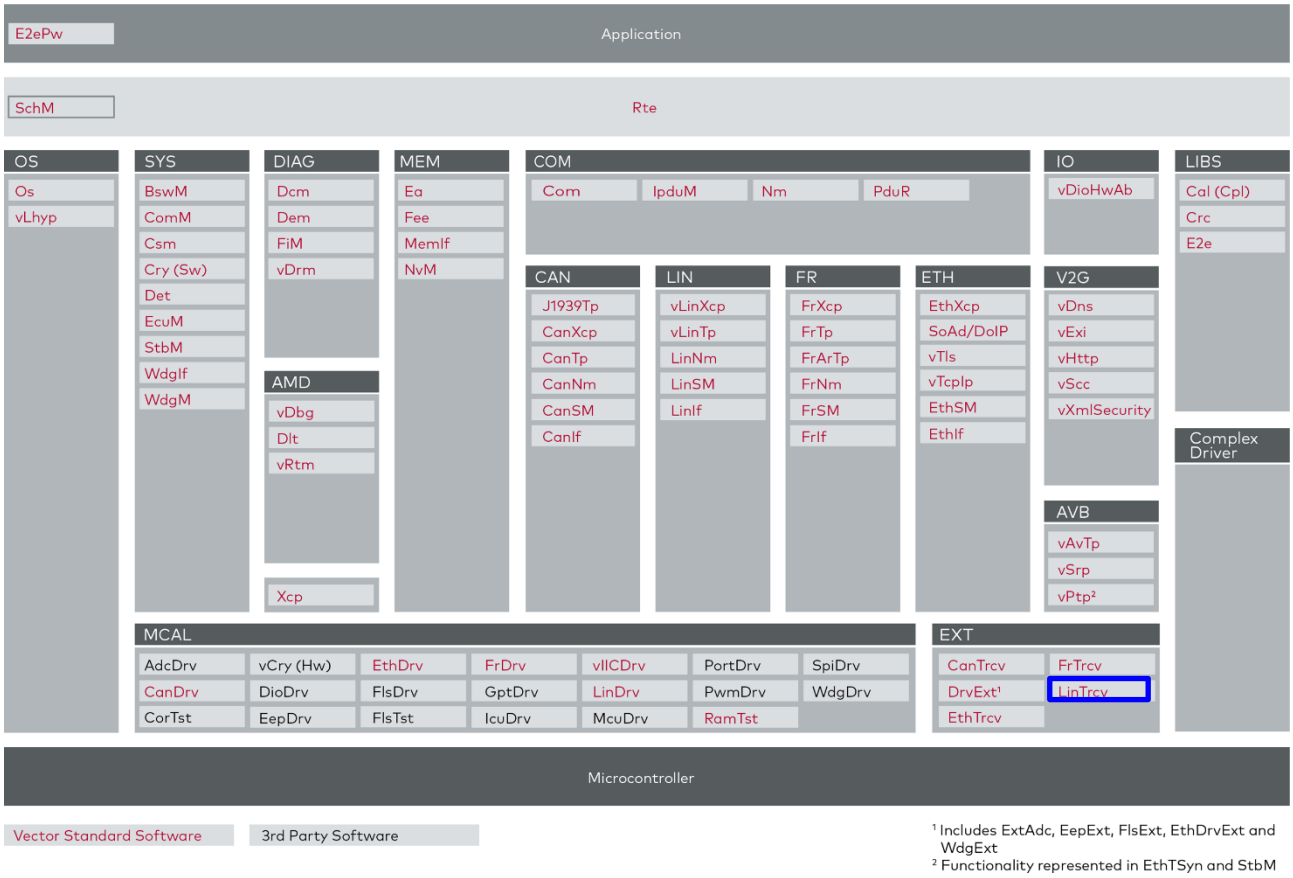

Figure 2-1 AUTOSAR 4.x Architecture Overview

3 Functional Description

3.1 Features

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

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 Transceiver initialization.
LIN Transceiver control via DIO.
Detection of wakeup.
Reading and setting operation mode of LIN Transceiver.

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
LIN Transceiver self-diagnostics.

Table 3-2 Not supported AUTOSAR standard conform features

3.1.2 Additions/ Extensions

3.1.2.1 Memory initialization

To have an independent memory initialization for this BSW module the additional function LinTrcv_30_Generic_InitMemory() was added. This must be called before normal initialization if initialized variables are not initialized during startup phase.

3.1.2.2 Initialization

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

The function LinTrcv_30_Generic_InitMemory() initializes all initialized variables of the LIN Transceiver Driver. This function has to be called after power on or reset before any other function in case initialized variables are not set after power on or reset (i.e. by the startup code).

The function LinTrcv_30_Generic_Init() initializes all LIN Transceiver Driver channels which are selected by the generation tool. Each LIN transceiver is switched into the configured channel specific init mode. Operation mode after power on or reset can be normal or sleep.

3.2 Operation modes

The operation modes are described as follows:

Operation Mode	Communication	Wakeup Detection	MCU power
Normal	Enabled	Disabled	Enabled
Standby	Disabled	Enabled	Enabled
Sleep	Disabled	Enabled	Disabled

The LIN transceiver driver supports the normal and sleep operation modes only. The standby mode is merged with the sleep mode: the hardware layout will determine whether the MCU power supply is cut or not.

Caution
Once the transceiver hardware detects a wakeup, it switches to a “Standby” mode which is different from the standby mode definition in AUTOSAR. The driver then considers the hardware to be in sleep mode with a wakeup event.

3.3 Set operation mode

The operation mode of the LIN transceiver hardware is changed by service function `LinTrcv_30_Generic_SetOpMode()`. It can be switched from normal into sleep mode and vice-versa.

Caution
Each operation mode transition requires a certain time to be complete. Depending on how fast the MCU is running, the MCU might be trying to communicate although the transceiver hardware is not ready yet. This can be avoided by configuring a non-zero wait count value: the Appl_LinTrcv_30_Generic_Wait call-out will then be invoked after each state transition allowing the user to implement the required delay to ensure the proper functioning.

3.4 Get operation mode

To get the current operation mode of a specified LIN transceiver hardware, the service function LinTrcv_30_Generic_GetOpMode() has to be called.

3.5 Wakeup event detection

Call LinTrcv_30_Generic_CheckWakeup() to check if the transceiver hardware has detected a wakeup by bus or pin event. It always stores the wakeup event and returns E_OK only if:

- The LIN Transceiver channel is in sleep mode
- A wakeup by bus or by pin was detected
- Wakeup mode is enabled for the transceiver channel

Otherwise, it will return E_NOT_OK.
If the above conditions are met, the EcuM and the LIN Interface is informed by calling EcuM_SetWakeupEvent() and LinIf_WakeupConfirmation().

The service function LinTrcv_30_Generic_ GetBusWuReason() returns the reason which caused the wakeup.

The service function LinTrcv_30_Generic_SetWakeupMode() sets the wakeup mode requested by the LIN Interface.

3.6 Error Handling

3.6.1 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 LINTRCV_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 LINTRCV ID is 64.
The reported service IDs identify the services which are described in 5.2. The following table presents the service IDs and the related services:

Service ID	Service
0x00	LinTrcv_30_Generic_Init
0x01	LinTrcv_30_Generic_SetOpMode
0x02	LinTrcv_30_Generic_GetOpMode
0x03	LinTrcv_30_Generic_GetBusWuReason
0x04	LinTrcv_30_Generic_GetVersionInfo
0x05	LinTrcv_30_Generic_SetWakeupMode
0x07	LinTrcv_30_Generic_CheckWakeUp

Table 3-3 Service IDs

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

Error Code		Description
0x01	LINTRCV_30_GENERIC_E_INVALID_LIN_NETWORK	API called with wrong parameter for LIN network
0x02	LINTRCV_30_GENERIC_E_PARAM_POINTER	API called with null pointer parameter
0x11	LINTRCV_30_GENERIC_E_UNINIT	API service used without initialization
0x21	LINTRCV_30_GENERIC_E_TRCV_NOT_SLEEP	API service called in wrong transceiver operation mode
0x22	LINTRCV_30_GENERIC_E_TRCV_NOT_NORMAL	API service called in wrong transceiver operation mode
0x23	LINTRCV_30_GENERIC_E_PARAM_TRCV_WAKEUP_MODE	API service called with invalid parameter for transceiver wakeup mode
0x24	LINTRCV_30_GENERIC_E_PARAM_TRCV_OPMODE	API service called with invalid parameter for operation mode
0x25	LINTRCV_30_GENERIC_E_INVALID_TRCV_OPMODE	API service called with invalid mode because optional transition is not enabled

Table 3-4 Errors reported to DET

3.6.1.1 Parameter Checking

AUTOSAR requires that API functions check the validity of their parameters. The checks in

Table 3-6 are internal parameter checks of the API functions. These checks are for development error reporting and can be en-/disabled by means of en-/disabling the DET reporting via the parameter LINTRCV_DEV_ERROR_DETECT.
The following table shows which parameter checks are performed on which services:

Service \ Check	LinTrcvIndex	OpMode	Reason	TrcvWakeupMode
LinTrcv_30_Generic_SetOpMode()	■	■		
LinTrcv_30_Generic_GetOpMode()	■	■		
LinTrcv_30_Generic_GetBusWuReason()	■		■	
LinTrcv_30_Generic_SetWakeupMode()	■			■
LinTrcv_30_Generic_CheckWakeup()	■			

Table 3-5 Development Error Reporting: Assignment of checks to services

4 Integration

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

4.1 Scope of Delivery

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

4.1.1 Static Files

File Name	Source Code Delivery	Object Code Delivery	Description
LinTrcv_30_Generic.h	■		Header file which has to be included by higher layers.
LinTrcv_30_Generic.c	■		Implementation

Table 4-1 Static files

4.1.2 Dynamic Files

The dynamic files are generated by the configuration tool - DaVinci Configurator 5.

File Name	Description
LinTrcv_30_Generic_Cfg.h	Header file contains define definitions and external data declarations. It is included by LinTrcv_30_Generic_.h.
LinTrcv_30_Generic_Cfg.c	Contains the pre-compile configuration data.

Table 4-2 Generated files

4.2 Include Structure

Following the include structure of the MICROSAR LIN Transceiver Driver is given. The including of MemMap.h is not shown.

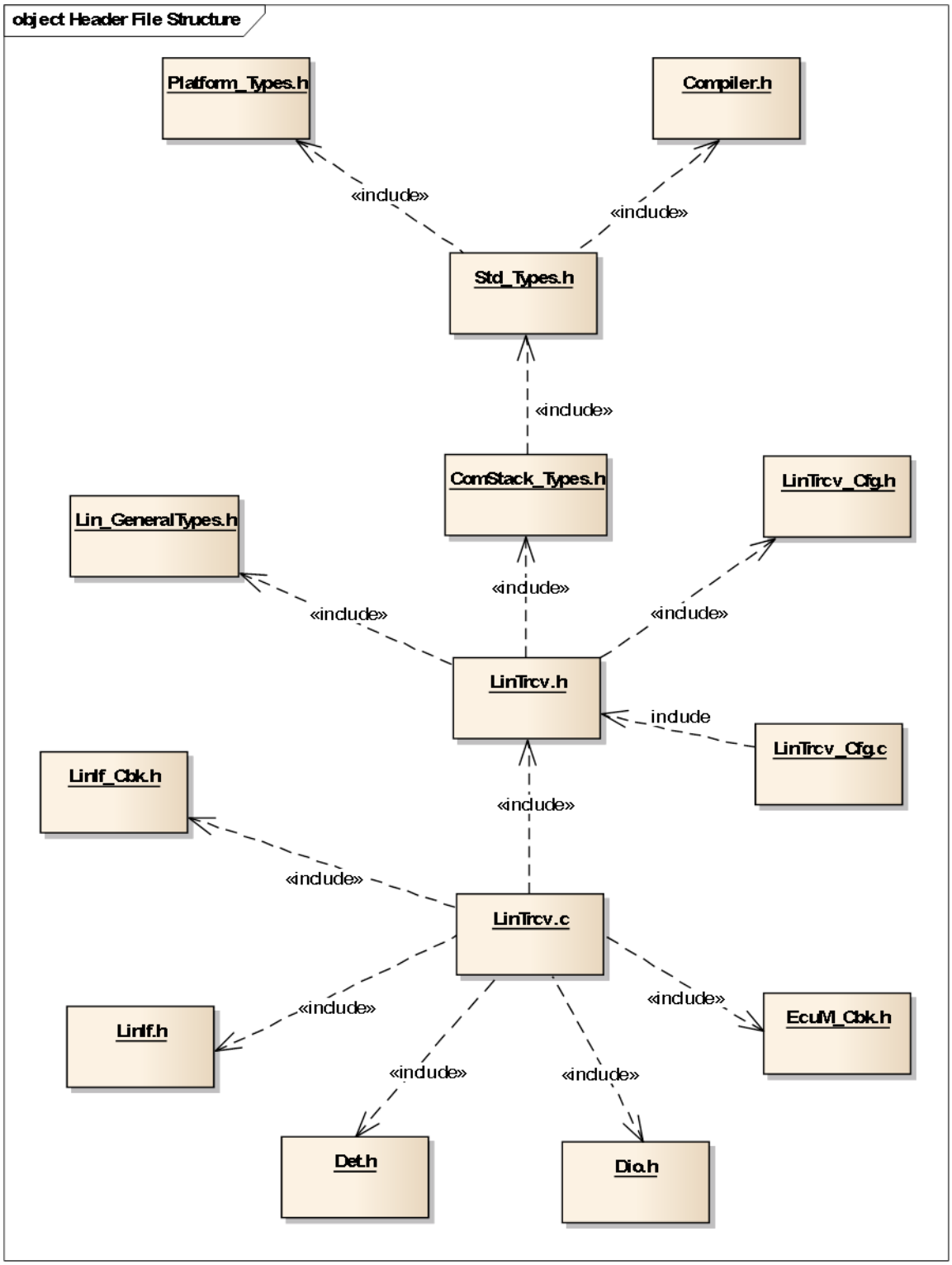

Figure 4-1 Include structure

4.3 Compiler Abstraction and Memory Mapping

The objects (e.g. variables, functions, constants) are declared by compiler independent definitions – the compiler abstraction definitions. Each compiler abstraction definition is assigned to a memory section.

The following table contains the memory section names and the compiler abstraction definitions of the LINTRCV and illustrates their assignment among each other.

Memory Mapping Sections \ Compiler Abstraction Definitions	LINTRCV_30_GENERIC_VAR	LINTRCV_30_GENERIC_VAR_ZERO_INIT	LINTRCV_30_GENERIC_APPL_VAR	LINTRCV_30_GENERIC_CONST	LINTRCV_30_GENERIC_CODE	LINTRCV_30_GENERIC_APPL_CODE
LINTRCV_30_GENERIC_START_SEC_VAR_NOINIT_UNSPECIFIED LINTRCV_30_GENERIC_STOP_SEC_VAR_NOINIT_UNSPECIFIED	■					
LINTRCV_30_GENERIC_START_SEC_VAR_ZERO_INIT_8BIT LINTRCV_30_GENERIC_STOP_SEC_VAR_ZERO_INIT_8BIT		■				
LINTRCV_30_GENERIC_START_SEC_CONST_8BIT LINTRCV_30_GENERIC_STOP_SEC_CONST_8BIT				■		
LINTRCV_30_GENERIC_START_SEC_CONST_32BIT LINTRCV_30_GENERIC_STOP_SEC_CONST_32BIT				■		
LINTRCV_30_GENERIC_START_SEC_CONST_UNSPECIFIED LINTRCV_30_GENERIC_STOP_SEC_CONST_UNSPECIFIED				■		
LINTRCV_30_GENERIC_START_SEC_CODE LINTRCV_30_GENERIC_STOP_SEC_CODE					■	

Table 4-3 Compiler abstraction and memory mapping

4.4 Critical Sections

The MICROSAR LIN Transceiver Driver has no code sections which are executed on interrupt level. But if the functions of the LIN Transceiver Driver can be interrupted by each

other due to different call context in a preemptive operating system, protection of certain areas must be granted.

LINTRCV_EXCLUSIVE_AREA_0

> Must lock interrupts if API functions of the LINTRCV can interrupt each other (i.e. preemptive task system, where a call of LinTrcv_30_Generic_CheckWakeup() can interrupt call of LinTrcv_30_ Generic _SetWakeupMode()).

To ensure data consistency and a correct function of the LINTRCV the code section must not be interrupted. Therefore the integrator must provide the following exclusive areas in the schedule manager.

It is recommended to use AUTOSAR OS ‘Resources’ for these exclusive areas to prevent priority inversion and dead-locks.

4.5 Dependency to DIO Driver

The MICROSAR LIN Transceiver Driver performs hardware access by calling service functions of the lower layer component DIO Driver:

> Function `Dio_WriteChannel()` is used to set the logical level of the channel pins to which the LIN transceiver hardware is connected.

> Function `Dio_ReadChannel()` is used to get the logical level of the channel pins to which the LIN transceiver hardware is connected.

> The DIO driver has to provide the pin assignment for the LIN transceiver hardware pins, which depend on the used LIN transceiver. These pins are referred by the LIN Transceiver Driver by using symbolic names or values, which must be specified in the configuration tool.

4.6 Dependency to ICU Driver

The LIN transceiver driver performs hardware access by calling service functions of the lower layer component Icu driver:

> Function Icu_DisableNotification() is used by the transceiver driver to disable the ICU notification after wakeup event notification.

> Function Icu_EnableNotification() is used by the transceiver driver to enable the ICU notification during the transition into the sleep mode.

The proposal described in this chapter enables the user to support the use-case by using an additional µC I/O port to generate the wakeup information via the ICU driver. This I/ O port is connected parallel to the LIN Rx port.

The user has to configure the ICU channel. Additionally the wakeup source for the LIN channel which shall be handled via this ICU channel have to be chosen.

Depending on the ICU configuration options provided it is possible to configure the “IcuDefaultStartEdge”. This option should be configured to ICU_FALLING_EDGE, because the wakeup event is provided by the Rx pin via a level change from recessive to dominant.

If the transceiver shall also wake up the ECU and not only the LIN channel then it is necessary to activate the “Wakeup Capability” for this ICU channel.

4.7 Example

This Generic LIN Transceiver driver has included example code. This example code is located between the following two markers.

```
/* BEGIN EXAMPLE */
/* END EXAMPLE */
```

The behavior of this example is for an Tle7259 like Transceiver.

Caution
This example is included as regular C-code. You have to define all the necessary settings or a Compiler Error might occur. The most important setting is the Dio Access.

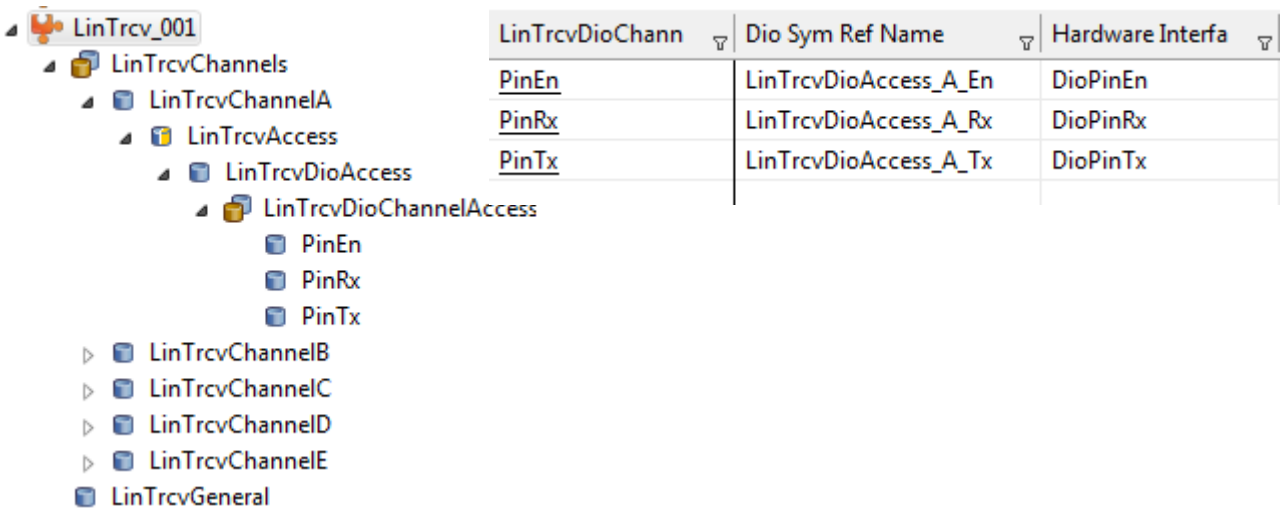

Figure 4-2 Settings example for Cfg5

5 API Description

5.1 Type Definitions

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

Type Name	C-Type	Description	Value Range
LinTrcv_TrcvModeType	enum	Operating modes of the LIN Transceiver Driver	LINTRCV_TRCV_MODE_NORMAL Transceiver mode NORMAL
			LINTRCV_TRCV_MODE_STANDBY Transceiver mode STANDBY
			LINTRCV_TRCV_MODE_SLEEP Transceiver mode SLEEP
LinTrcv_TrcvWakeupModeType	enum	Wake up operating modes of the LIN Transceiver Driver	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
LinTrcv_TrcvWakeupReasonType	enum	This type denotes the wake up reason detected by the LIN transceiver in detail	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

Type Name	C-Type	Description	Value Range
			The transceiver has detected, that the wake up is due to an ECU reset after power on

Table 5-1 Type definitions

5.2 Services provided by LINTRCV

The LINTRCV API consists of services, which are realized by function calls.

5.2.1 LinTrcv_30_Generic_InitMemory

Prototype	
`Void LinTrcv_30_Generic_InitMemory ( void )`	
Parameter	
-	-
Return code	
-	-
Functional Description	
> This function initializes all initialized variables of the LIN Transceiver Driver. This function has to be called after power on or reset before any other function only in case initialized variables are not set after power on or reset (i.e. by the startup code).	
Particularities and Limitations	
This function must be called before any other functionality of the LIN Transceiver Driver.	
Expected Caller Context	
This function must be called from task level and is not reentrant.	

Table 5-2 LinTrcv_30_**Generic**_InitMemory

5.2.2 LinTrcv_30_Generic_Init

Prototype	
`void LinTrcv_30_Generic_Init ( void )`	
Parameter	
-	-

Return code	
-	-
Functional Description	
> This function initializes all channels of the LIN Transceiver Driver which are configured in the configuration tool.	
Particularities and Limitations	
> This function must be called before any other service functionality of the LIN Transceiver Driver	
Expected Caller Context	
This function must be called from task level and is not reentrant.	

Table 5-3 LinTrcv_30_**Generic**_Init

5.2.3 LinTrcv_30_Generic_SetOpMode

Prototype	
`Std_ReturnType` **`LinTrcv_30_Generic_SetOpMode`** `(uint8 LinTrcvIndex, LinTrcv_TrcvModeType OpMode)`	
Parameter	
LinTrcvIndex	Index of the selected transceiver
OpMode	Pointer which contains the desired operation mode
Return code	
E_OK / E_NOT_OK	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 has failed or the parameter is out of the allowed range. The previous state has not been changed.
Functional Description	
Sets the LIN transceiver to the requested operation mode. These operation modes are: > LINTRCV_30_GENERIC_OP_MODE_NORMAL > LINTRCV_30_GENERIC_OP_MODE_SLEEP	
Particularities and Limitations	
> The LIN Transceiver Driver must be initialized. > Not each transition from one mode in any other is allowed.	
Expected Caller Context	
This function can be called from task or interrupt level and is not reentrant.	

Table 5-4 LinTrcv_30_**Generic**_SetOpMode

5.2.4 LinTrcv_30_Generic_GetOpMode

Prototype	
`Std_ReturnType` **`LinTrcv_30_Generic_GetOpMode`** `(uint8 LinTrcvIndex, LinTrcv_TrcvModeType *OpMode)`	
Parameter	
LinTrcvIndex	Index of the selected transceiver
OpMode	Pointer to operation mode of the bus the API is applied to
Return code	
E_OK / E_NOT_OK	E_OK: will be returned if the operation mode was detected. E_NOT_OK: will be returned if the operation mode was not detected.
Functional Description	
Returns the operation mode of the selected LIN transceiver. These operation modes are: > LINTRCV_30_GENERIC_OP_MODE_NORMAL > LINTRCV_30_GENERIC_OP_MODE_SLEEP	
Particularities and Limitations	
> The LIN Transceiver Driver must be initialized.	
Expected Caller Context	
This function can be called from task or interrupt level and is not reentrant.	

Table 5-5 LinTrcv_30_**Generic**_GetOpMode

5.2.5 LinTrcv_30_Generic_GetBusWuReason

Prototype	
`Std_ReturnType` **`LinTrcv_30_Generic_GetBusWuReason`** `(uint8 LinTrcvIndex, LinTrcv_TrcvModeType *Reason )`	
Parameter	
LinTrcvIndex	Index of the selected transceiver.
Reason	Pointer to wake up reason of the bus the API is applied to.
Return code	
E_OK / E_NOT_OK	E_OK: will be returned if the wake up reason was detected. E_NOT_OK: will be returned if the wake up reason was not detected.

Functional Description
Returns the wakeup reason for the channel LinTrcvIndex. These wakeup reasons are: > 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.
Particularities and Limitations
> The LIN Transceiver Driver must be initialized.
Expected Caller Context
This function can be called from task or interrupt level and is not reentrant.

Table 5-6 LinTrcv_30_**Generic**_GetBusWuReason

5.2.6 LinTrcv_30_Generic_SetWakeupMode

Prototype	
`Ret_code` **`LinTrcv_30_Generic_SetWakeupMode`** `(uint8 LinTrcvIndex, LinTrcv_TrcvModeType TrcvWakeupMode)`	
Parameter	
LinTrcvIndex	Index of the selected transceiver
TrcvWakeupMode	Requested transceiver wakeup reason
Return code	
E_OK / E_NOT_OK	E_OK: Will be returned, if the wakeup state has been changed to the requested mode. E_NOT_OK: Will be returned, if the wakeup state change has failed or the parameter is out of the allowed range. The previous state has not been changed.
Functional Description	
Enables and disables the reporting from wakeup events on the channel LinTrcvIndex. To clear wake-up events on the channel LinTrcvIndex is also provided. > LINIF_TRCV_WU_ENABLE: Report wakeup events to upper layer. > LINIF_TRCV_WU_DISABLE: Do not report wakeup events to upper layer. > LINIF_TRCV_WU_CLEAR: Clear a pending wakeup event (Only when the wakeup mode is disabled)	

Particularities and Limitations
> The LIN Transceiver Driver must be initialized. > If the wakeup handling is not enabled in Cfg5 the function always return E_NOT_OK.
Expected Caller Context
This function can be called from task or interrupt level and is not reentrant.

Table 5-7 LinTrcv_30_**Generic**_SetWakeupMode

5.2.7 LinTrcv_30_Generic_CheckWakeup

Prototype	
`Std_ReturnType` **`LinTrcv_30`**`_Generic_CheckWakeup (uint8 LinTrcvIndex)`	
Parameter	
LinTrcvIndex	Index of the selected transceiver
Return code	
E_OK /E_NOT_OK	E_OK: when a wakeup is detected. E_NOT_OK: when no wakeup is detected or wakeup by bus is not supported.
Functional Description	
This function returns if a wakeup was detected on the bus. It can be used in interrupt and polling mode.	
Particularities and Limitations	
> The LIN Transceiver Driver must be initialized.	
Expected Caller Context	
This function can be called from task or interrupt level and is not reentrant.	

Table 5-8 LinTrcv_30_Generic_CheckWakeup

5.2.8 LinTrcv_30_Generic_GetVersionInfo

Prototype	
`void` **`LinTrcv_30_Generic_GetVersionInfo`** `(Std_VersionInfoType VersionInfo)`	
Parameter	
VersionInfo	Pointer to version information of this module.
Return code	
-	-
Functional Description	
Gets the version of the module and returns it in VersionInfo.	
Particularities and Limitations	
> -	

Expected Caller Context
This function can always be called.

Table 5-9 LinTrcv_30_**Generic**_GetVersionInfo

5.3 Services used by LINTRCV

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

Component	API
DET	Det_ReportError()
Dio	Dio_WriteChannel() Dio_ReadChannel()
EcuM	EcuM_SetWakeupEvent() EcuM_GeneratorCompatibilityError()
LinIf	LinIf_WakeupConfirmation()
SchM	SchM_Enter_LinTrcv_30_Generic_LINTRCV_30_GENERIC_EXCLUSIVE_AREA_0() SchM_Exit_LinTrcv_30_Generic_LINTRCV_30_GENERIC_EXCLUSIVE_AREA_0()

Table 5-10 Services used by the LINTRCV

5.4 Callback Functions

5.4.1 Appl_LinTrcv_30_Generic_Wait

Prototype	
`void` **`Appl_LinTrcv_30_Generic_Wait`** `(uint32 LinTrcvWaitCount)`	
Parameter	
LinTrcvWaitCount	Pointer to operation mode of the bus the API is applied to
Return code	
-	-
Functional Description	
> Allows the user to implement a delay after each mode transition	
Particularities and Limitations	
> The LIN Transceiver Driver must be initialized.	
Expected Caller Context	
This function is called from the LinTrcv_30_Generic_Init and LinTrcv_30_Generic_SetOpMode functions	

Table 5-11 Appl_LinTrcv_30_Generic_Wait

6 Configuration with DaVinci Configurator 5

Refer to the integrated online help and parameter descriptions of Configurator 5.

6.1 Configuration Variants

The LINTRCV supports the configuration variants

> VARIANT-PRE-COMPILE

> VARIANT-POSTBUILD-SELECTABLE

The configuration classes of the LINTRCV parameters depend on the supported configuration variants.

For their definitions please see the LinTrcv_30_Generic_bswmd.arxml file.

6.1.1 LinTrcvChannel

Container Name	LinTrcvChannel
Path	\MICROSAR\LinTrcv
Multiplicity	1…*
Description	

Table 6-1 Container LinTrcvChannel

Attribute Name	Multi-plicity	Value Type	Values Default value is typed bold	Description
LinTrcvChannelId	1…1	Integer	**0** - 255	Unique identifier of the LIN Transceiver Channel.
LinTrcvChannelUsed	1…1	Boolean	**true** false	Shall the related LIN transceiver channel be used?
LinTrcvInitState	1…1	String	LINTRCV_TRCV_MODE_NORMAL LINTRCV_TRCV_MODE_STANDBY LINTRCV_TRCV_MODE_SLEEP	State of LIN transceiver after call to LinTrcv_Init.
LinTrcvWakeupByBusUsed	1…1	Boolean	true false	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.
LinTrcvWakeupByPinUsed	1…1	Boolean	true false	Is wake up by pin supported? If LIN transceiver hardware does not support wake up by pin value is always FALSE. If

Attribute Name	Multi-plicity	Value Type	Values Default value is typed bold	Description
				LIN transceiver hardware supports wake up by pin value is TRUE or FALSE depending whether it is used or not.
LinTrcvSleepInitWorkaround	1…1	Boolean	true false	Some transceivers cannot be initialized in sleep mode after power on. When enabled and if the channel init state is set to sleep, the driver will set the transceiver to normal mode then to sleep mode. The TX pin level must remain high during the operation to avoid dominant spikes on the LIN bus.
LinTrcvIcuChannelRef	0…1	-	**-**	Reference to the IcuChannel to enable/disable the interrupts for wakeups.
LinTrcvWakeupSourceRef	0…1	-	**-**	Reference to a wakeup source in the EcuM configuration. This reference is only needed if LinTrcvWakeupByBusUsed is true. Implementation Type: reference to EcuM_WakeupSourceType.

Table 6-2 Attributes of LinTrcvChannel

Sub Container	Multiplicity
LinTrcvAccess	1…*

Table 6-3 Sub Containers of LinTrcvChannel

Sub Container	Multiplicity
LinTrcvDioAccess	0…1
LinTrcvSpiSequence	0…1

Table 6-4 Sub Containers of LinTrcvAccess

6.1.2 LinTrcvDioAccess

Container Name	LinTrcvDioAccess
Path	\MICROSAR\LinTrcv\LinTrcvChannel\LinTrcvAccess
Multiplicity	0…1
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

Table 6-5 Container LinTrcvDioAccess

Attribute Name	Multi-plicity	Value Type	Values Default value is typed bold	Description
LinTrcvHardwareInterfaceName	1…1	String	-	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.
LinTrcvDioSymRefName	1…1	-	-	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.

Table 6-6 Attributes of LinTrcvDioAccess

7 Glossary and Abbreviations

7.1 Glossary

Term	Description
CFG5	DaVinci Configurator 5, generation tool for MICROSAR components

Table 7-1 Glossary

7.2 Abbreviations

Abbreviation	Description
API	Application Programming Interface
AUTOSAR	Automotive Open System Architecture
BSW	Basis Software
DEM	Diagnostic Event Manager
DET	Development Error Tracer
EAD	Embedded Architecture Designer
ECU	Electronic Control Unit
HIS	Hersteller Initiative Software
ISR	Interrupt Service Routine
MICROSAR	Microcontroller Open System Architecture (the Vector AUTOSAR solution)
PPORT	Provide Port
RPORT	Require Port
RTE	Runtime Environment
SRS	Software Requirement Specification
SWC	Software Component
SWS	Software Specification

Table 7-2 Abbreviations

8 Contact

Visit our website for more information on

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

www.vector.com