
User Manual

for S32K14X LIN Driver

Document Number: UM2LINASR4.2 Rev0002R1.0.1
Rev. 1.0





Contents

Section number	Title	Page
Chapter 1		
Revision History		
Chapter 2		
Introduction		
2.1	Supported Derivatives.....	9
2.2	Overview.....	9
2.3	About this Manual.....	10
2.4	Acronyms and Definitions.....	10
2.5	Reference List.....	11
Chapter 3		
Driver		
3.1	Requirements.....	13
3.2	Driver Design Summary.....	13
3.3	Hardware Resources.....	14
3.4	Deviation from Requirements.....	15
3.5	Driver limitations.....	17
3.6	Runtime Errors.....	17
3.7	Software specification.....	18
3.7.1	Define Reference.....	18
3.7.1.1	Define LIN_BIT_ERROR.....	18
3.7.1.2	Define LIN_BREAK_DELIMITER_ERROR.....	19
3.7.1.3	Define LIN_BUFFER_OVER_RUN_ERROR.....	19
3.7.1.4	Define LIN_CH_NOT_READY_STATE.....	19
3.7.1.5	Define LIN_CH_OPERATIONAL.....	20
3.7.1.6	Define LIN_CH_READY_STATE.....	20
3.7.1.7	Define LIN_CH_RECEIVE_NOTHING_STATE.....	20
3.7.1.8	Define LIN_CH_SLEEP_STATE.....	21
3.7.1.9	Define LIN_CHECKSUM_ERROR.....	21

Section number	Title	Page
3.7.1.10	Define LIN_E_INVALID_CHANNEL.....	22
3.7.1.11	Define LIN_E_INVALID_POINTER.....	22
3.7.1.12	Define LIN_E_PARAM_POINTER.....	22
3.7.1.13	Define LIN_E_STATE_TRANSITION.....	23
3.7.1.14	Define LIN_E_UNINIT.....	23
3.7.1.15	Define LIN_FRAMING_ERROR.....	23
3.7.1.16	Define LIN_IDENTIFIER_PARITY_ERROR.....	24
3.7.1.17	Define LIN_NO_ERROR.....	24
3.7.1.18	Define LIN_NOISE_ERROR.....	25
3.7.1.19	Define LIN_RX_COMPLETE_STATE.....	25
3.7.1.20	Define LIN_SYNC_FIELD_ERROR.....	25
3.7.1.21	Define LIN_TIMEOUT_ERROR.....	26
3.7.1.22	Define LIN_TX_COMPLETE_STATE.....	26
3.7.1.23	Define LIN_TX_MASTER_RES_COMMAND.....	26
3.7.1.24	Define LIN_TX_NO_COMMAND.....	27
3.7.1.25	Define LIN_TX_SLAVE_RES_COMMAND.....	27
3.7.1.26	Define LIN_TX_SLEEP_COMMAND.....	27
3.7.1.27	Define LIN_UNINIT.....	28
3.7.2	Enum Reference.....	28
3.7.2.1	Enumeration Lin_ApiFunctionIdType.....	28
3.7.2.2	Enumeration Lin_ClockModesType.....	29
3.7.2.3	Micro Second Channel status type.....	29
3.7.3	Function Reference.....	30
3.7.3.1	Function Lin_CheckWakeup.....	30
3.7.3.2	Function Lin_GetStatus.....	31
3.7.3.3	Function Lin_GetVersionInfo.....	32
3.7.3.4	Function Lin_GoToSleep.....	33
3.7.3.5	Function Lin_GoToSleepInternal.....	33
3.7.3.6	Function Lin_Init.....	34

Section number	Title	Page
3.7.3.7	Function Lin_SendFrame.....	35
3.7.3.8	Function Lin_Wakeup.....	36
3.7.3.9	Function Lin_WakeupInternal.....	36
3.7.4	Structs Reference.....	37
3.7.4.1	Structure Lin_StaticConfig_ChannelConfigType.....	37
3.7.4.2	Structure Lin_ChannelConfigType.....	37
3.7.4.3	Structure Lin_ConfigType.....	38
3.7.4.4	Structure Lin_MscDataType.....	38
3.7.5	Types Reference.....	39
3.8	Symbolic Names Disclaimer.....	39

Chapter 4

Tresos Configuration Plug-in

4.1	Configuration elements of Lin.....	41
4.2	Form IMPLEMENTATION_CONFIG_VARIANT.....	41
4.3	Form NonAutosar.....	42
4.3.1	LinDisableDemReportErrorStatus (NonAutosar).....	42
4.3.2	LinDisableFrameTimeout (NonAutosar).....	43
4.3.3	LinEnableUserModeSupport (NonAutosar).....	43
4.4	Form LinDemEventParameterRefs.....	44
4.4.1	LIN_E_TIMEOUT (LinDemEventParameterRefs).....	44
4.5	Form LinGeneral.....	44
4.5.1	LinDevErrorDetect (LinGeneral).....	45
4.5.2	LinIndex (LinGeneral).....	45
4.5.3	LinTimeoutDuration (LinGeneral).....	45
4.5.4	LinVersionInfoApi (LinGeneral).....	46
4.6	Form CommonPublishedInformation.....	46
4.6.1	ArReleaseMajorVersion (CommonPublishedInformation).....	47
4.6.2	ArReleaseMinorVersion (CommonPublishedInformation).....	47
4.6.3	ArReleaseRevisionVersion (CommonPublishedInformation).....	48

Section number	Title	Page
4.6.4	ModuleId (CommonPublishedInformation).....	48
4.6.5	SwMajorVersion (CommonPublishedInformation).....	49
4.6.6	SwMinorVersion (CommonPublishedInformation).....	49
4.6.7	SwPatchVersion (CommonPublishedInformation).....	50
4.6.8	VendorApiInfix (CommonPublishedInformation).....	50
4.6.9	VendorId (CommonPublishedInformation).....	50
4.7	Form LinGlobalConfig.....	51
4.7.1	Form LinChannel.....	51
4.7.1.1	LinChannelId (LinChannel).....	52
4.7.1.2	LinChannelBaudRate (LinChannel).....	52
4.7.1.3	BreakLength (LinChannel).....	53
4.7.1.4	LinHwChannel (LinChannel).....	53
4.7.1.5	LinChannelWakeupSupport (LinChannel).....	54
4.7.1.6	LinClockRef (LinChannel).....	54
4.7.1.7	LinClockRef_Alternate (LinChannel).....	54
4.7.1.8	LinChannelEcuMWakeupSource (LinChannel).....	55

Chapter 1

Revision History

Table 1-1. Revision History

Revision	Date	Author	Description
1.0	13/07/2018	NXP MCAL Team	Updated version for ASR 4.2.2S32K14X1.0.1 Release



Chapter 2

Introduction

This User Manual describes NXP Semiconductors AUTOSAR Local Interconnect Network (LIN) for S32K14X .

AUTOSAR LIN driver configuration parameters and deviations from the specification are described in LIN Driver chapter of this document. AUTOSAR LIN driver requirements and APIs are described in the AUTOSAR LIN driver software specification document.

2.1 Supported Derivatives

The software described in this document is intended to be used with the following microcontroller devices of NXP Semiconductors .

Table 2-1. S32K14X Derivatives

NXP Semiconductors	s32k148_lqfp144, s32k148_lqfp176, s32k148_mapbga100, s32k146_lqfp144, s32k146_lqfp100, s32k146_lqfp64, s32k146_mapbga100, s32k144_lqfp100, s32k144_lqfp64, s32k144_mapbga100, s32k142_lqfp100, s32k142_lqfp64, s32k118_lqfp48, s32k118_lqfp64
--------------------	---

All of the above microcontroller devices are collectively named as S32K14X .

2.2 Overview

AUTOSAR (AUTomotive Open System ARchitecture) is an industry partnership working to establish standards for software interfaces and software modules for automobile electronic control systems.

AUTOSAR

- paves the way for innovative electronic systems that further improve performance, safety and environmental friendliness.
- is a strong global partnership that creates one common standard: "Cooperate on standards, compete on implementation".
- is a key enabling technology to manage the growing electrics/electronics complexity. It aims to be prepared for the upcoming technologies and to improve cost-efficiency without making any compromise with respect to quality.
- facilitates the exchange and update of software and hardware over the service life of the vehicle.

2.3 About this Manual

This Technical Reference employs the following typographical conventions:

Boldface type: Bold is used for important terms, notes and warnings.

Italic font: Italic typeface is used for code snippets in the text. Note that C language modifiers such "const" or "volatile" are sometimes omitted to improve readability of the presented code.

Notes and warnings are shown as below:

Note

This is a note.

2.4 Acronyms and Definitions

Table 2-2. Acronyms and Definitions

Term	Definition
API	Application Programming Interface
ASM	Assembler
AUTOSAR	Automotive Open System Architecture
BSMI	Basic Software Make file Interface
C/CPP	C and C++ Source Code
DEM	Diagnostic Event Manager
DET	Default Error Tracer
EcuM	ECU state Manager
GUI	Graphical User Interface
ISR	Interrupt Service Routine
LIN	Local Interconnect Network

Table continues on the next page...

Table 2-2. Acronyms and Definitions (continued)

Term	Definition
MCU	Micro Controller Unit
N/A	Not Applicable
OS	Operating System
PB Variant	Post Build Variant
PC Variant	Pre Compile Variant
VLE	Variable Length Encoding

2.5 Reference List

Table 2-3. Reference List

#	Title	Version
1	Specification of LIN Driver	AUTOSAR Release 4.2.2
2	S32K14X Reference Manual	Reference Manual, Rev. 4, 06/2017
3	S32K142 Mask Set Errata for Mask 0N33V (0N33V)	30/11/2017
4	S32K144 Mask Set Errata for Mask 0N57U (0N57U)	30/11/2017
5	S32K146 Mask Set Errata for Mask 0N73V (0N73V)	30/11/2017
6	S32K148 Mask Set Errata for Mask 0N20V (0N20V)	30/11/2017
7	S32K118 Mask Set Errata for Mask 0N97V (0N97V)	26/02/2018

Chapter 3 Driver

3.1 Requirements

Requirements for this driver are detailed in the AUTOSAR 4.2 Rev0002LIN Driver Software Specification document (See Table [Reference List](#)).

3.2 Driver Design Summary

The LIN driver is part of the microcontroller abstraction layer (MCAL), performs the hardware access and offers a hardware independent API to the upper layer.

The only upper layer, which has access to the LIN driver, is the LIN Interface.

A LIN driver can support more than one channel.

This means that the LIN driver can handle one or more LIN channels as long as they belong to the same LIN hardware unit.

The LIN Driver for S32K14X, uses the LPUART on-chip hardware module which provides special support for the LIN protocol.

It can be used to automate most tasks of a LIN master.

It is possible to transmit entire frames (or sequences of frames) and receive data from LIN slaves without any CPU intervention.

The LIN physical interface should be connected to the LPUART module pins in order to get the LIN bus voltage levels.

The S32K14X contains up to six blocks.

The LPUART has the following major features:

- Transmit and receive baud rate can operate asynchronous to the bus clock
- Programmable baud rates (13-bit modulo divider) with configurable oversampling ratio from 4x to 32x
- Receive data register full, transmit data register empty and transmission complete interrupts
- Receive overrun, framing error, and noise error detection
- Optional 13-bit break character generation

3.3 Hardware Resources

The device family includes S32K118, S32K142, S32K144, S32K146 and S32K148 derivatives.

The hardware configured by the Lin driver is LPUART.

Lin Physical Channels:

- For S32K118 derivative has 2 channels: LPUART_0, LPUART_1.
- For S32K142 derivative has 2 channels: LPUART_0, LPUART_1.
- For S32K144, S32K146 and S32K148 derivatives have 3 channels: LPUART_0, LPUART_1 and LPUART_2.

For example:

In EB tresos, LPUART_0 channel has named LPUART_0, correspondingly as below:

Lin hardware channel



Figure 3-1. LPUART_0 configuration in EB tresos

The LIN channel to microcontroller pin mapping can be done using file "S32K142_IO_Signal_Description_Input_Multiplexing.xlsx" from Reference manual.

The RM chapter stated above has a table as below:

Chip	Instances	TX FIFO (word)	RX FIFO (word)
S32K118	LPUART0	4	4
	LPUART1	4	4
S32K142	LPUART0	4	4
	LPUART1	4	4
S32K144	LPUART0	4	4
	LPUART1	4	4
	LPUART2	4	4
S32K146	LPUART0	4	4
	LPUART1	4	4
	LPUART2	4	4
S32K148	LPUART0	4	4
	LPUART1	4	4
	LPUART2	4	4

Figure 3-2. LPUART configuration in Reference manual

LPUART_0 channel can be found in the S32K142_IO_Signal_Description_Input_Multiplexing.xlsx file with naming is LPUART0. And the Pin-Muxing is:

PTB1	PCR_PT B1	0000_0000	DISABLED		Signal Path Disabled	-	GPIO	53
		0000_0001	PTB1	PTB	Port B I/O	I/O		
		0000_0010	LPUART0_TX	LPUART0	Transmit	I/O		
		0000_0011	LPSPiO_SOUT	LPSPiO	LPSPi Serial Data Output	I/O		
		0000_0100	TCLK0	FTM	FTM External Clock Input	I		
		0000_0101	CAN0_TX	CAN0	CAN Tx Channel	O		
		-	ADC0_SE5	ADC0	ADC Single Ended Input	I		
		-	ADC1_SE15	ADC1	ADC Single Ended Input	I		

PTB0	PCR_PT B0	0000_0000	DISABLED		Signal Path Disabled	-	GPIO	54
		0000_0001	PTB0	PTB	Port B I/O	I/O		
		0000_0010	LPUART0_RX	LPUART0	Receive	I		
		0000_0011	LPSPiO_PCS0	LPSPiO	Peripheral Chip Select 0	I/O		
		0000_0100	LPTMRO_ALT3	LPTMRO	Low Power Timer Input Channel	I		
		0000_0101	CAN0_RX	CAN0	CAN Rx channel	I		
		-	ADC0_SE4	ADC0	ADC Single Ended Input	I		
		-	ADC1_SE14	ADC1	ADC Single Ended Input	I		

Figure 3-3. IO Signal Description for LPUART_0 channel

- For transmit data signal TXD connected to pin PTB[1].
- For receive data signal RXD connected to pin PTB[0].

3.4 Deviation from Requirements

The driver deviates from the AUTOSAR LIN Driver software specification in some places.

Table identifies the AUTOSAR requirements that are not fully implemented, implemented differently, or out of scope for the LIN driver. Table Table 3-1 provides Status column description.

Table 3-1. Deviations Status Column Description

Term	Definition
N/A	Not available
N/T	Not testable
N/S	Out of scope
N/R	Unclear Requirement
N/I	Not implemented
N/F	Not fully implemented
I/D	Implemented with Deviations

Below table identifies the AUTOSAR requirements that are not fully implemented, implemented differently, or out of scope for the driver.

Table 3-2. Driver Deviations Table

Requirement	Status	Description	Notes
SWS_Lin_0020 1	N/I	For different LIN hardware units a separate LIN driver needs to be implemented. It is up to the implementer to adapt the driver to the different instances of similar LIN channels.	Rejection reason: there is only one type of hardware unit available (LPUART).
SWS_Lin_0017 7	N/I	In case several LIN driver instances (of same or different vendor) are implemented in one ECU the file names, API names, and published parameters must be modified such that no two definitions with the same name are generated. The name shall be extended according to SRS_BSW_00347 with a Vendor Id (needed to distinguish LIN drivers from different vendors) and a Vendor specific name (needed to distinguish different hardware units implemented by one Vendor).	Rejection reason: There is only one LIN driver instance.
SWS_Lin_0005 5	N/S	The Lin module shall fulfill all design and implementation guidelines as described in Specification of C Implementation Rules AUTOSAR_TR_CImplementationRules.pdf.	Requirement already covered by process.
SWS_Lin_0002 6	N/I	If the LIN hardware unit cannot queue the bytes for transmission or reception (e.g. simple UART implementation), the LIN driver shall provide a temporary communication buffer.	The LIN hardware already has a data buffer built-in.

Table continues on the next page...

Table 3-2. Driver Deviations Table (continued)

Requirement	Status	Description	Notes
SWS_Lin_00039	N/R	Values that can be configured are hardware dependent. Therefore, the rules and constraints cannot be given in the standard.	This is not a requirement.
SWS_Lin_00999	N/R	These requirements are not applicable to this specification. (SRS_BSW_00307, SRS_BSW_00312, SRS_BSW_00325, SRS_BSW_00326, SRS_BSW_00328, SRS_BSW_00329, SRS_BSW_00330, SRS_BSW_00331, SRS_BSW_00336, SRS_BSW_00339, SRS_BSW_00342, SRS_BSW_00343, SRS_BSW_00353, SRS_BSW_00357, SRS_BSW_00359, SRS_BSW_00360, SRS_BSW_00361, SRS_BSW_00373, SRS_BSW_00376, SRS_BSW_00378, SRS_BSW_00383, SRS_BSW_00395, SRS_BSW_00397, SRS_BSW_00398, SRS_BSW_00399, SRS_BSW_00400, SRS_BSW_00413, SRS_BSW_00415, SRS_BSW_00416, SRS_BSW_00417, BSW00420, SRS_BSW_00422, SRS_BSW_00423, SRS_BSW_00424, SRS_BSW_00425, SRS_BSW_00426, SRS_BSW_00427, SRS_BSW_00428, SRS_BSW_00429, BSW00431, SRS_BSW_00432, SRS_BSW_00433, BSW00434, SRS_BSW_00005, SRS_BSW_00007, SRS_BSW_00162, SRS_BSW_00168, SRS_SPAL_12056, SRS_SPAL_12267, SRS_SPAL_12163, SRS_SPAL_12463, SRS_SPAL_12075, SRS_SPAL_12078, SRS_SPAL_12092, SRS_Lin_01551, SRS_Lin_01568, SRS_Lin_01569, SRS_Lin_01570, SRS_Lin_01564, SRS_Lin_01546, SRS_Lin_01561, SRS_Lin_01549, SRS_Lin_01571, SRS_Lin_01514, SRS_Lin_01515, SRS_Lin_01502, SRS_Lin_01558, BSW01527, SRS_Lin_01523, SRS_Lin_01540, SRS_Lin_01545, SRS_Lin_01534, SRS_Lin_01574, SRS_Lin_01539, SRS_Lin_01544, SRS_Lin_01590).	This is not a requirement.

3.5 Driver limitations

None

3.6 Runtime Errors

The driver generates the following DEM errors at runtime.

Table 3-3. Runtime Errors

Function	Error Code	Condition triggering the error
Lin_GoToSleep()	LIN_E_TIMEOUT	Timeout caused by hardware error waiting for cancellation of current frame, hardware did not set TDRE and TC bits in LPUARTx_STAT register in the allocated time defined by "LinTimeoutDuration" parameter in configuration. No sleep command will be sent, and LIN driver will not enter sleep state.
Lin_GoToSleepInternal()	LIN_E_TIMEOUT	Timeout caused by hardware error waiting for cancellation of current frame, hardware did not set TDRE and TC bits in LPUARTx_STAT register in the allocated time defined by "LinTimeoutDuration" parameter in configuration. LIN driver will not enter sleep state.
Lin_SendFrame()	LIN_E_TIMEOUT	Timeout caused by hardware error waiting for cancellation of current frame, hardware did not set TDRE and TC bits in LPUARTx_STAT register in the allocated time defined by "LinTimeoutDuration" parameter in configuration. New frame will not be sent.

3.7 Software specification

The following sections contains driver software specifications.

3.7.1 Define Reference

Constants supported by the driver are as per AUTOSAR LIN Driver software specification Version 4.2 Rev0002 .

3.7.1.1 Define LIN_BIT_ERROR

Interrupt Errors conditions.

Details:

Bit error on a channel: - During response field transmission (Slave and Master modes); - During header transmission (in Master mode).

Table 3-4. Define LIN_BIT_ERROR Description

Name	LIN_BIT_ERROR
Initializer	((uint8)0x01U)

3.7.1.2 Define LIN_BREAK_DELIMITER_ERROR

Interrupt Errors conditions.

Details:

Break Delimiter too short (less than 1 bit).

Table 3-5. Define LIN_BREAK_DELIMITER_ERROR Description

Name	LIN_BREAK_DELIMITER_ERROR
Initializer	((uint8)0x04U)

3.7.1.3 Define LIN_BUFFER_OVER_RUN_ERROR

Interrupt Errors conditions.

Details:

New data byte is received on a channel and the buffer full flag is not cleared.

Table 3-6. Define LIN_BUFFER_OVER_RUN_ERROR Description

Name	LIN_BUFFER_OVER_RUN_ERROR
Initializer	((uint8)0x07U)

3.7.1.4 Define LIN_CH_NOT_READY_STATE

LIN Channel states.

Details:

The individual channel is not ready to process a frame.

Table 3-7. Define LIN_CH_NOT_READY_STATE Description

Name	LIN_CH_NOT_READY_STATE
Initializer	((uint8)0x04U)

3.7.1.5 Define LIN_CH_OPERATIONAL

LIN Channel states.

Details:

The individual channel has been initialized (using at least one statically configured data set) and is able to participate in the LIN cluster.

Table 3-8. Define LIN_CH_OPERATIONAL Description

Name	LIN_CH_OPERATIONAL
Initializer	((uint8)0x03U)

3.7.1.6 Define LIN_CH_READY_STATE

LIN Channel states.

Details:

The individual channel is ready to process a frame.

Table 3-9. Define LIN_CH_READY_STATE Description

Name	LIN_CH_READY_STATE
Initializer	((uint8)0x05U)

3.7.1.7 Define LIN_CH_RECEIVE_NOTHING_STATE

LIN Channel states.

Details:

State after the LIN frame header was correctly sent.

Table 3-10. Define LIN_CH_RECEIVE_NOTHING_STATE Description

Name	LIN_CH_RECEIVE_NOTHING_STATE
Initializer	((uint8)0x08U)

3.7.1.8 Define LIN_CH_SLEEP_STATE

LIN Channel states.

Details:

The detection of a wake-up pulse is enabled. The LIN hardware is into a low power mode if such a mode is provided by the hardware.

Table 3-11. Define LIN_CH_SLEEP_STATE Description

Name	LIN_CH_SLEEP_STATE
Initializer	((uint8)0x02U)

3.7.1.9 Define LIN_CHECKSUM_ERROR

Interrupt Errors conditions.

Details:

Checksum error on a channel.

Table 3-12. Define LIN_CHECKSUM_ERROR Description

Name	LIN_CHECKSUM_ERROR
Initializer	((uint8)0x02U)

3.7.1.10 Define LIN_E_INVALID_CHANNEL

API service used with an invalid or inactive channel parameter.

Details:

The LIN Driver module shall report the default error "LIN_E_INVALID_CHANNEL (0x02)", when API Service used with an invalid or inactive channel parameter.

Table 3-13. Define LIN_E_INVALID_CHANNEL Description

Name	LIN_E_INVALID_CHANNEL
Initializer	((uint8)0x02U)

3.7.1.11 Define LIN_E_INVALID_POINTER

API service called with invalid configuration pointer.

Details:

The LIN Driver module shall report the default error "LIN_E_INVALID_POINTER (0x03)", when API Service is called with invalid configuration pointer.

Table 3-14. Define LIN_E_INVALID_POINTER Description

Name	LIN_E_INVALID_POINTER
Initializer	((uint8)0x03U)

3.7.1.12 Define LIN_E_PARAM_POINTER

API service called with a NULL pointer.

Details:

The LIN Driver module shall report the default error "LIN_E_PARAM_POINTER (0x05)", when API Service is called with a NULL pointer. In case of this error, the API service shall return immediately without any further action, beside reporting this default error.

Table 3-15. Define LIN_E_PARAM_POINTER Description

Name	LIN_E_PARAM_POINTER
Initializer	((uint8)0x05U)

3.7.1.13 Define LIN_E_STATE_TRANSITION

Invalid state transition for the current state.

Details:

The LIN Driver module shall report the default error "LIN_E_STATE_TRANSITION (0x04)", when Invalid state transition occurs from the current state.

Table 3-16. Define LIN_E_STATE_TRANSITION Description

Name	LIN_E_STATE_TRANSITION
Initializer	((uint8)0x04U)

3.7.1.14 Define LIN_E_UNINIT

API service used without module initialization.

Details:

The LIN Driver module shall report the default error "LIN_E_UNINIT (0x00)", when the API Service is used without module initialization.

Table 3-17. Define LIN_E_UNINIT Description

Name	LIN_E_UNINIT
Initializer	((uint8)0x00U)

3.7.1.15 Define LIN_FRAMING_ERROR

Interrupt Errors conditions.

Details:

Invalid stop bit: - During reception of any data in the response field (Slave and Master modes); - During reception of Synch or Identifier Field (Slave mode).

Table 3-18. Define LIN_FRAMING_ERROR Description

Name	LIN_FRAMING_ERROR
Initializer	((uint8)0x06U)

3.7.1.16 Define LIN_IDENTIFIER_PARITY_ERROR

Interrupt Errors conditions.

Details:

Parity error.

Table 3-19. Define LIN_IDENTIFIER_PARITY_ERROR Description

Name	LIN_IDENTIFIER_PARITY_ERROR
Initializer	((uint8)0x05U)

3.7.1.17 Define LIN_NO_ERROR

Interrupt Errors conditions.

Details:

No error occurred on a channel.

Table 3-20. Define LIN_NO_ERROR Description

Name	LIN_NO_ERROR
Initializer	((uint8)0x00U)

3.7.1.18 Define LIN_NOISE_ERROR

Interrupt Errors conditions.

Details:

Noise detected on a received character.

Table 3-21. Define LIN_NOISE_ERROR Description

Name	LIN_NOISE_ERROR
Initializer	((uint8)0x08U)

3.7.1.19 Define LIN_RX_COMPLETE_STATE

LIN Channel states.

Details:

LIN frame was received; no errors.

Table 3-22. Define LIN_RX_COMPLETE_STATE Description

Name	LIN_RX_COMPLETE_STATE
Initializer	((uint8)0x07U)

3.7.1.20 Define LIN_SYNCH_FIELD_ERROR

Interrupt Errors conditions.

Details:

Inconsistent Synch Field.

Table 3-23. Define LIN_SYNCH_FIELD_ERROR
Description

Name	LIN_SYNCH_FIELD_ERROR
Initializer	((uint8)0x03U)

3.7.1.21 Define LIN_TIMEOUT_ERROR

Interrupt Errors conditions.

Details:

Header or Response timeout detected.

Table 3-24. Define LIN_TIMEOUT_ERROR Description

Name	LIN_TIMEOUT_ERROR
Initializer	((uint8)0x09U)

3.7.1.22 Define LIN_TX_COMPLETE_STATE

LIN Channel states.

Details:

LIN frame was sent; no errors.

Table 3-25. Define LIN_TX_COMPLETE_STATE Description

Name	LIN_TX_COMPLETE_STATE
Initializer	((uint8)0x06U)

3.7.1.23 Define LIN_TX_MASTER_RES_COMMAND

Commands IDs.

Details:

Tx frame is a master frame (response is provided by master).

**Table 3-26. Define LIN_TX_MASTER_RES_COMMAND
Description**

Name	LIN_TX_MASTER_RES_COMMAND
Initializer	((uint8)0x01U)

3.7.1.24 Define LIN_TX_NO_COMMAND

Commands IDs.

Details:

No tx master command pending.

Table 3-27. Define LIN_TX_NO_COMMAND Description

Name	LIN_TX_NO_COMMAND
Initializer	((uint8)0x04U)

3.7.1.25 Define LIN_TX_SLAVE_RES_COMMAND

Commands IDs.

Details:

Tx frame is a slave frame (response is provided by slave).

**Table 3-28. Define LIN_TX_SLAVE_RES_COMMAND
Description**

Name	LIN_TX_SLAVE_RES_COMMAND
Initializer	((uint8)0x02U)

3.7.1.26 Define LIN_TX_SLEEP_COMMAND

Commands IDs.

Details:

Tx frame is a sleep command frame.

Table 3-29. Define LIN_TX_SLEEP_COMMAND Description

Name	LIN_TX_SLEEP_COMMAND
Initializer	((uint8)0x03U)

3.7.1.27 Define LIN_UNINIT

LIN driver states.

Details:

The state LIN_UNINIT means that the Lin module has not been initialized yet and cannot be used.

Table 3-30. Define LIN_UNINIT Description

Name	LIN_UNINIT
Initializer	((uint8)0x01U)

3.7.2 Enum Reference

Enumeration of all constants supported by the driver are as per AUTOSAR LIN Driver software specification Version 4.2 Rev0002 .

3.7.2.1 Enumeration Lin_ApiFunctionIdType

API functions service IDs.

Details:

Service IDs of the AUTOSAR LIN API.

Table 3-31. Enumeration Lin_ApiFunctionIdType Values

Name	Initializer	Description
LIN_MSC_INITCHANNEL_ID	(uint8)0x0BU	Msc_InitChannel() ID.
LIN_MSC_DEINITCHANNEL_ID	(uint8)0x0CU	Msc_DeInitChannel() ID.
LIN_MSC_GETSTATUS_ID	(uint8)0x0DU	Msc_GetStatus() ID.
LIN_MSC_POLLING_ID	(uint8)0x0EU	Msc_Polling() ID.
LIN_GETSTATUS_ID	(uint8)0x08U	Lin_GetStatus() ID.
LIN_GETVERSIONINFO_ID	(uint8)0x01U	Lin_GetVersionInfo() ID.
LIN_GOTOSLEEP_ID	(uint8)0x06U	Lin_GoToSleep() ID.
LIN_GOTOSLEEPINTERNAL_ID	(uint8)0x09U	Lin_GoToSleepInternal() ID.
LIN_INIT_ID	(uint8)0x00U	Lin_Init() ID.
LIN_SENDFRAME_ID	(uint8)0x04U	Lin_SendFrame() ID.
LIN_WAKEUP_ID	(uint8)0x07U	Lin_WakeUp() ID.
LIN_CHECKWAKEUP_ID	(uint8)0x0AU	Lin_CheckWakeUp() ID.

3.7.2.2 Enumeration Lin_ClockModesType

Clock modes.

Pre:

LIN_DUAL_CLOCK_MODE must be defined and its value must be STD_ON.

Table 3-32. Enumeration Lin_ClockModesType Values

Name	Initializer	Description
LIN_NORMAL	(uint8)0x01U	LIN_NORMAL mode.
LIN_ALTERNATE	(uint8)0x02U	LIN_ALTERNATE mode.

3.7.2.3 Micro Second Channel status type.

Clock modes.

Details:

Micro Second Channel status type.

Note

LIN_USE_MSC must be defined. MSC channel frame operation status, as returned by the API service Msc_GetStatus().

Table 3-33. Enumeration Lin_MscStatusType Values

Name	Initializer	Description
LIN_MSC_IDLE	(uint8)0x00U	Receiver is disabled and no reception is running.
LIN_MSC_READY	(uint8)0x01U	Receiver is enabled and no reception is running.
LIN_MSC_RUN	(uint8)0x02U	Receiver is enabled and reception is running.
LIN_MSC_WAKEUP	(uint8)0x03U	Receiver is in wakeup mode.
LIN_MSC_OVERRUN_ERROR	(uint8)0x04U	Erroneous reception due to an OR error.
LIN_MSC_FRAMING_ERROR	(uint8)0x05U	Erroneous reception due to a FE error.
LIN_MSC_PARITY_ERROR	(uint8)0x06U	Erroneous reception due to a PE error.
LIN_MSC_NOISE_ERROR	(uint8)0x07U	Erroneous reception due to an NF error.

3.7.3 Function Reference

Functions of all functions supported by the driver are as per AUTOSAR LIN Driver software specification Version 4.2 Rev0002 .

3.7.3.1 Function Lin_CheckWakeup

Validates for upper layers the wake up of LIN channel.

Details:

This function identifies if the addressed LIN channel has been woken up by the LIN bus transceiver. This API is used when the LIN channel wake up functionality is disabled (wake up interrupt is disabled). It checks the wake up flag from the addressed LIN channel which must be in sleep mode and have the wake up functionality disabled.

Note

Autosar Service ID: 0x0A.Synchronous, non reentrant function.

Violates: include statements in a file should only be preceded by other preprocessor directives or comments.

Violates: Precautions shall be taken in order to prevent the contents of a header file being included twice.

Prototype: `Std_ReturnType Lin_CheckWakeup(uint8 Channel);`

Table 3-34. Lin_CheckWakeup Return Values

Name	Description
E_NOT_OK	If the LIN Channel is not valid or LIN driver is not initialized or the addressed LIN Channel is not in sleep state.
E_OK	Otherwise.

3.7.3.2 Function Lin_GetStatus

Gets the status of the LIN driver.

Details:

This function returns the state of the current transmission, reception or operation status. If the reception of a Slave response was successful then this service provides a pointer to the buffer where the data is stored.

Return: Lin_StatusType.

Note

Autosar Service ID: 0x08.Synchronous, non reentrant function.

Prototype: `Lin_StatusType Lin_GetStatus(uint8 Channel, uint8 **Lin_SduPtr);`

Table 3-35. Lin_GetStatus Arguments

Type	Name	Direction	Description
uint8	Channel	input	LIN channel to be checked.
uint8 **	Lin_SduPtr	output	Lin_SduPtr pointer to pointer to a shadow buffer or memory mapped LIN Hardware receive buffer where the current SDU is stored.

Table 3-36. Lin_GetStatus Return Values

Name	Description
LIN_NOT_OK	Development or production error rised none of the below conditions.
LIN_TX_OK	Successful transmission.
LIN_TX_BUSY	Ongoing transmission of header or response.

Table continues on the next page...

Table 3-36. Lin_GetStatus Return Values (continued)

Name	Description
LIN_TX_HEADER_ERROR	Error occurred during header transmission.
LIN_TX_ERROR	Error occurred during response transmission.
LIN_RX_OK	Reception of correct response.
LIN_RX_BUSY	Ongoing reception where at least one byte has been received.
LIN_RX_ERROR	Error occurred during reception.
LIN_RX_NO_RESPONSE	No data byte has been received yet.
LIN_OPERATIONAL	Channel is ready for next header. transmission and no data are available.
LIN_CH_SLEEP	Channel is in sleep mode.

3.7.3.3 Function Lin_GetVersionInfo

Returns the version information of this module.

Details:

The version information includes:

- Two bytes for the Vendor ID
- Two bytes for the Module ID
- One byte for the Instance ID
- Three bytes version number. The numbering shall be vendor specific: it consists of:
 - The major, the minor and the patch version number of the module;
 - The AUTOSAR specification version number shall not be included. The AUTOSAR specification version number is checked during compile time and therefore not required in this API.

Return: void.

Pre: Preconditions as text description. Optional tag.

Note

Autosar Service ID: 0x01. Synchronous, non reentrant function.

Prototype: void Lin_GetVersionInfo(Std_VersionInfoType *versioninfo);

Table 3-37. Lin_GetVersionInfo Arguments

Type	Name	Direction	Description
Std_VersionInfoType *	versioninfo	input, output	Pointer for storing the version information of this module.

3.7.3.4 Function Lin_GoToSleep

The service instructs the driver to transmit a go-to-sleep-command on the addressed LIN channel.

Details:

This function stops any ongoing transmission and initiates the transmission of the sleep command (master command frame with ID = 0x3C and data = (0x00, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF)). State transition in LIN_CH_SLEEP_STATE shall be done after the completion of the sleep command transmission regardless of the success (therefore the ISR is responsible to put the channel in LIN_CH_SLEEP_STATE).

Return: Std_ReturnType.

Note

Autosar Service ID: 0x06.Synchronous, non reentrant function.

Prototype: Std_ReturnType Lin_GoToSleep(uint8 Channel);

Table 3-38. Lin_GoToSleep Arguments

Type	Name	Direction	Description
uint8	Channel	input	LIN channel to be checked.

Table 3-39. Lin_GoToSleep Return Values

Name	Description
E_NOT_OK	If the LIN Channel is not valid or LIN driver is not initialized or LIN Channel is in sleep state or a timeout occurs.
E_OK	Otherwise.

3.7.3.5 Function Lin_GoToSleepInternal

Put a Lin channel in the internal sleep state.

Details:

Stops any ongoing transmission, sets the channel state to LIN_CH_SLEEP and put the LIN hardware unit to a reduced power operation mode.

Return: Std_ReturnType.

Note

Autosar Service ID: 0x09.Synchronous, non reentrant function.

Prototype: Std_ReturnType Lin_GoToSleepInternal(uint8 Channel);

Table 3-40. Lin_GoToSleepInternal Arguments

Type	Name	Direction	Description
uint8	Channel	input	LIN channel to be addressed.

Table 3-41. Lin_GoToSleepInternal Return Values

Name	Description
E_NOT_OK	If the LIN Channel is not valid or LIN driver is not initialized or LIN Channel is in sleep state or a timeout occurs.
E_OK	Otherwise.

3.7.3.6 Function Lin_Init

Initializes the LIN module.

Details:

This function performs software initialization of LIN driver:

- Clears the shadow buffer of all available Lin channels
- Set LIN channel state machine of all available Lin channels to LIN_CH_OPERATIONAL
- Set frame operation state machine of all available LIN channels to LIN_CH_READY_STATE
- Set driver state machine to LIN_INIT.

Violates: include statements in a file should only be preceded by other preprocessor directives or comments.

Violates: Precautions shall be taken in order to prevent the contents of a header file being included twice.

Return: void.

Note

Autosar Service ID: 0x00. Synchronous, non reentrant function.
 Lin_Init always require a valid pointer

Prototype: void Lin_Init(const Lin_ConfigType *Config);

Table 3-42. Lin_Init Arguments

Type	Name	Direction	Description
const Lin_ConfigType *	Config	input	Pointer to LIN driver configuration set.

3.7.3.7 Function Lin_SendFrame

Sends a LIN frame.

Details:

Sends a LIN header and a LIN response, if necessary. The direction of the frame response (master response, slave response, slave-to-slave communication) is provided by the PduInfoPtr.

Return: Std_ReturnType.

Note

Autosar Service ID: 0x04. Synchronous, non reentrant function.

Prototype: Std_ReturnType Lin_SendFrame(uint8 Channel, Lin_PduType *PduInfoPtr);

Table 3-43. Lin_SendFrame Arguments

Type	Name	Direction	Description
uint8	Channel	input	LIN channel to be addressed.
Lin_PduType *	PduInfoPtr	input	Pointer to PDU containing the PID, Checksum model, Response type, DI and SDU data pointer.

Table 3-44. Lin_SendFrame Return Values

Name	Description
E_NOT_OK	If the LIN Channel is not valid or LIN driver is not initialized or PduInfoPtr is NULL or a timeout occurs or LIN Channel is in sleep state.
E_OK	Otherwise.

3.7.3.8 Function Lin_Wakeup

Generates a wake up pulse.

Details:

This function shall sent a wake up signal to the LIN bus and put the LIN channel in LIN_CH_OPERATIONAL state.

Return: Std_ReturnType.

Note

Autosar Service ID: 0x07.Synchronous, non reentrant function.

Prototype: Std_ReturnType Lin_Wakeup(uint8 Channel);

Table 3-45. Lin_Wakeup Arguments

Type	Name	Direction	Description
uint8	Channel	input	LIN channel to be addressed.

Table 3-46. Lin_Wakeup Return Values

Name	Description
E_NOT_OK	If the LIN driver is not in sleep state or LIN Channel is not valid or LIN driver is not initialized.
E_OK	Otherwise.

3.7.3.9 Function Lin_WakeupInternal

Generates a wake up pulse.

Details:

This function shall put the LIN channel in LIN_CH_OPERATIONAL state.

Return: Std_ReturnType.

Note

Autosar Service ID: 0x0B.Synchronous, non reentrant function.

Prototype: Std_ReturnType Lin_WakeupInternal(uint8 Channel);

Table 3-47. Lin_WakeupInternal Arguments

Type	Name	Direction	Description
uint8	Channel	input	LIN channel to be addressed.

Table 3-48. Lin_WakeupInternal Return Values

Name	Description
E_NOT_OK	If the LIN driver is not in sleep state or LIN Channel is not valid or LIN driver is not initialized.
E_OK	Otherwise.

3.7.4 Structs Reference

Data structures supported by the driver are as per AUTOSAR LIN Driver software specification Version 4.2 Rev0002 .

3.7.4.1 Structure Lin_StaticConfig_ChannelConfigType

The structure contains configuration parameters that are not variant aware.

Declaration

```
typedef struct
{
    uint8 u8LinChannelID;
    uint8 u8LinHwChannel;
    uint8 u8LinChannelWakeupSupport;
    EcuM_WakeupSourceType LinChannelEcuMWakeupSource;
} Lin_StaticConfig_ChannelConfigType;
```

Table 3-49. Structure Lin_StaticConfig_ChannelConfigType member description

Member	Description
u8LinChannelID	LIN Channel ID.
u8LinHwChannel	LIN Hardware Channel.
u8LinChannelWakeupSupport	Is wake-up supported by the LIN channel ?
LinChannelEcuMWakeupSource	[SWS_Lin_00098] This parameter contains a reference to the Wakeup Source for this controller as defined in the ECU State Manager.

3.7.4.2 Structure Lin_ChannelConfigType

The structure contains the configuration of each channel.

Declaration

```
typedef struct
{
    const Lin_StaticConfig_ChannelConfigType * pChannelConfigPC;
    uint32 u32Baudrate;
    uint32 u32Lin_BaudRate_RegValue;
    uint32 u32Lin_BaudRate_RegValue_Alternate;
    uint8 u8LinChannelBreakLength;
} Lin_ChannelConfigType;
```

Table 3-50. Structure Lin_ChannelConfigType member description

Member	Description
pChannelConfigPC	Pointer to the configuration struct that contains all the parameters that are not variant aware.
u32Baudrate	LIN Baudrate value.
u32Lin_BaudRate_RegValue	LIN baudrate register's value.
u32Lin_BaudRate_RegValue_Alternate	LIN baudrate register's value with alternate clock. This member exists only when LIN_DUAL_CLOCK_MODE is defined and its value is STD_ON
u8LinChannelBreakLength	These bits indicate the Break length in Master mode.

3.7.4.3 Structure Lin_ConfigType

The structure contains the configuration of all used channels.

Declaration

```
typedef struct
{
    const Lin_ChannelConfigType * const pLin_Channel[LIN_HW_MAX_MODULES];
} Lin_ConfigType;
```

Table 3-51. Structure Lin_ConfigType member description

Member	Description
pLin_Channel[LIN_HW_MAX_MODULES]	Constant pointer of the constant external data structure containing the overall initialization data for all the LIN Channels.

3.7.4.4 Structure Lin_MscDataType

Micro Second Channel data type.

Declaration

```
typedef struct
{
    uint8 u8Data;
    uint8 u8Address;
    uint8 u8Errors;
    Lin_MscStatusType State;
} Lin_MscDataType;
```

Table 3-52. Structure Lin_MscDataType member description

Member	Description
u8Data	Payload of the upstream MSC frame.
u8Address	Address of the receive data buffer.
u8Errors	Upstream MSC frame error type.
State	Upstream MSC channel state.

3.7.5 Types Reference

Types supported by the driver are as per AUTOSAR LIN Driver software specification Version 4.2 Rev0002 .

3.8 Symbolic Names Disclaimer

All containers having the symbolic name tag set as true in the Autosar schema will generate defines like:

```
#define <Container_Short_Name> <Container_ID>
```

For this reason it is forbidden to duplicate the name of such containers across the MCAL configuration, or to use names that may trigger other compile issues (e.g. match existing #ifdefs arguments).

Chapter 4

Tresos Configuration Plug-in

This chapter describes the Tresos configuration plug-in for the LIN Driver. The most of the parameters are described below.

4.1 Configuration elements of Lin

Included forms :

- IMPLEMENTATION_CONFIG_VARIANT
- NonAutosar
- LinGeneral
- CommonPublishedInformation
- LinGlobalConfig

Table 4-1. Revision table

Revision	Date
4.1.0	2010-12-03

4.2 Form IMPLEMENTATION_CONFIG_VARIANT

VARIANT-PRE-COMPILE: Only parameters with Pre-compile time configuration are allowed in this variant.

VARIANT-POST-BUILD: Parameters with Pre-compile time, Link time and Post-build time are allowed in this variant.

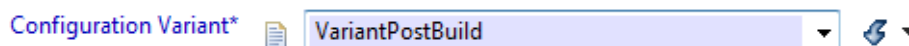


Figure 4-1. Tresos Plugin snapshot for IMPLEMENTATION_CONFIG_VARIANT form.

Table 4-2. Attribute IMPLEMENTATION_CONFIG_VARIANT detailed description

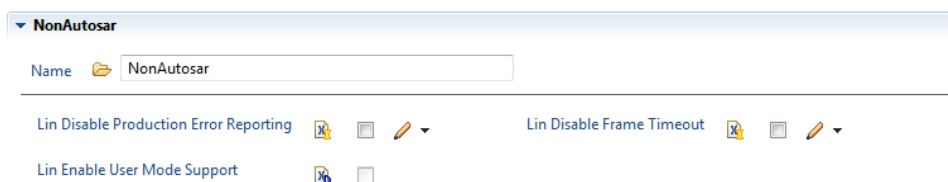
Property	Value
Label	Configuration Variant
Type	ENUMERATION
Symbolic Name	false
Default	VariantPostBuild
Range	VariantPostBuild VariantPreCompile

4.3 Form NonAutosar

NonAutosar

Autosar Requirements:

This container contains the global configuration parameters of the Non-Autosar Lin driver. This container is a MultipleConfigurationContainer, i.e. this container and its sub-containers exist once per configuration set.

**Figure 4-2. Tresos Plugin snapshot for NonAutosar form.**

4.3.1 LinDisableDemReportErrorStatus (NonAutosar)

LinDisableDemReportErrorStatus

Switches the Diagnostic Error Reporting and Notification OFF.

Table 4-3. Attribute LinDisableDemReportErrorStatus (NonAutosar) detailed description

Property	Value
Label	Lin Disable Production Error Reporting
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

4.3.2 LinDisableFrameTimeout (NonAutosar)

LinDisableFrameTimeout

When LinDisableFrameTimeout is ON, LIN driver will accept the frame that is longer than Maximal Frame Length.

Note

Due to hardware limitation, LIN driver on S32K14X will not support LIN check frame timeout. Consequently, this feature does not apply to S32K14X.

Table 4-4. Attribute LinDisableFrameTimeout (NonAutosar) detailed description

Property	Value
Label	Lin Disable Frame Timeout
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

4.3.3 LinEnableUserModeSupport (NonAutosar)

LinEnableUserModeSupport

When LinEnableUserModeSupport is ON, the Lin module will adapt to run from User Mode.

Note

Lin module does not include registers protection. So, it is accessible to all registered in any public mode.

Table 4-5. Attribute LinEnableUserModeSupport (NonAutosar) detailed description

Property	Value
Label	Lin Enable User Mode Support
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

4.4 Form LinDemEventParameterRefs

Is included by form : [Form LinGlobalConfig](#)

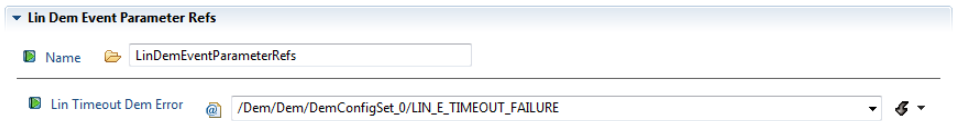


Figure 4-3. Tresos Plugin snapshot for LinDemEventParameterRefs form.

4.4.1 LIN_E_TIMEOUT (LinDemEventParameterRefs)

Table 4-6. Attribute LIN_E_TIMEOUT (LinDemEventParameterRefs) detailed description

Property	Value
Label	Lin Timeout Dem Error
Type	SYMBOLIC-NAME-REFERENCE
Origin	AUTOSAR_ECUC
Enable	true

4.5 Form LinGeneral

LinGeneral

Autosar Requirements: ECUC_Lin_00183

This container contains the parameters related to each LIN Driver Unit. This container is a MultipleConfigurationContainer, i.e. this container and its sub-containers exist once per configuration set.

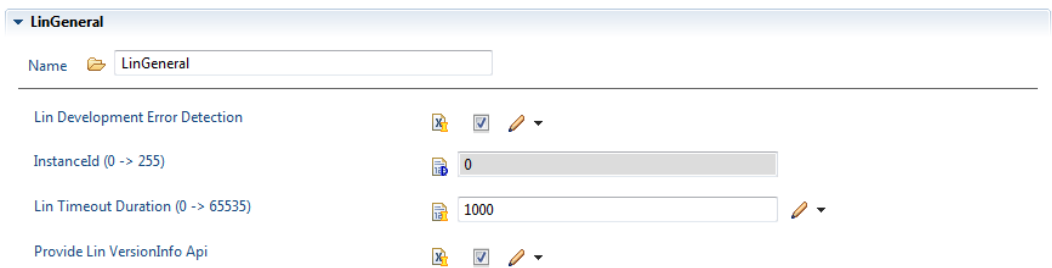


Figure 4-4. Tresos Plugin snapshot for LinGeneral form.

4.5.1 LinDevErrorDetect (LinGeneral)

LinDevErrorDetect

Autosar Requirements: ECUC_Lin_00066

Switches the Default Error Detection and Notification ON or OFF.

Table 4-7. Attribute LinDevErrorDetect (LinGeneral) detailed description

Property	Value
Label	Lin Default Error Detection
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	true

4.5.2 LinIndex (LinGeneral)

LinIndex

Autosar Requirements: ECUC_Lin_00179

Specifies the InstanceId of this module instance. If only one instance is present it shall have the Id 0.

Note, this parameter is not used in the current implementation.

Table 4-8. Attribute LinIndex (LinGeneral) detailed description

Property	Value
Label	InstanceId
Type	INTEGER
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	0
Invalid	Range <=255 >=0

4.5.3 LinTimeoutDuration (LinGeneral)

LinTimeoutDuration

Autosar Requirements: ECUC_Lin_00093

Specifies the maximum number of loops for blocking function until a timeout is raised in short term wait loops

Table 4-9. Attribute LinTimeoutDuration (LinGeneral) detailed description

Property	Value
Label	Lin Timeout Duration
Type	INTEGER
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	1000
Invalid	Range <div><=65535</div> <div>>=0</div>

4.5.4 LinVersionInfoApi (LinGeneral)

LinVersionInfoApi

Autosar Requirements: ECUC_Lin_00067

Switches the Lin_GetVersionInfo function ON or OFF.

Table 4-10. Attribute LinVersionInfoApi (LinGeneral) detailed description

Property	Value
Label	Provide Lin VersionInfo Api
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	true

4.6 Form CommonPublishedInformation

Common container, aggregated by all modules. It contains published information about vendor and versions.

Name	Value
AUTOSAR Major Version	4
AUTOSAR Minor Version	2
AUTOSAR Release Revision Version	2
Numeric Module ID	82
Software Major Version	1
Software Minor Version	0
Software Patch Version	1
Vendor Api Infix	
Vendor ID	43

Figure 4-5. Tressos Plugin snapshot for CommonPublishedInformation form.

4.6.1 ArReleaseMajorVersion (CommonPublishedInformation)

Major version number of AUTOSAR specification on which the appropriate implementation is based on.

Table 4-11. Attribute ArReleaseMajorVersion (CommonPublishedInformation) detailed description

Property	Value
Label	AUTOSAR Major Version
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false
Default	4
Invalid	Range <div>>=4</div> <div><=4</div>

4.6.2 ArReleaseMinorVersion (CommonPublishedInformation)

Minor version number of AUTOSAR specification on which the appropriate implementation is based on.

Table 4-12. Attribute ArReleaseMinorVersion (CommonPublishedInformation) detailed description

Property	Value
Label	AUTOSAR Minor Version
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false
Default	2
Invalid	Range >=2 <=2

4.6.3 ArReleaseRevisionVersion (CommonPublishedInformation)

Revision version number of AUTOSAR specification on which the appropriate implementation is based on.

Table 4-13. Attribute ArReleaseRevisionVersion (CommonPublishedInformation) detailed description

Property	Value
Label	AUTOSAR Release Revision Version
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false
Default	2
Invalid	Range >=2 <=2

4.6.4 ModuleId (CommonPublishedInformation)

Module ID of this module from Module List.

Table 4-14. Attribute ModuleId (CommonPublishedInformation) detailed description

Property	Value
Label	Module Id
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false

Table continues on the next page...

Table 4-14. Attribute ModuleId (CommonPublishedInformation) detailed description (continued)

Property	Value
Default	82
Invalid	Range >=82 <=82

4.6.5 SwMajorVersion (CommonPublishedInformation)

Major version number of the vendor specific implementation of the module. The numbering is vendor specific.

Table 4-15. Attribute SwMajorVersion (CommonPublishedInformation) detailed description

Property	Value
Label	Software Major Version
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false
Default	1
Invalid	Range >=1 <=1

4.6.6 SwMinorVersion (CommonPublishedInformation)

Minor version number of the vendor specific implementation of the module. The numbering is vendor specific.

Table 4-16. Attribute SwMinorVersion (CommonPublishedInformation) detailed description

Property	Value
Label	Software Minor Version
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false
Default	0
Invalid	Range >=0 <=0

4.6.7 SwPatchVersion (CommonPublishedInformation)

Patch level version number of the vendor specific implementation of the module. The numbering is vendor specific.

Table 4-17. Attribute SwPatchVersion (CommonPublishedInformation) detailed description

Property	Value
Label	Software Patch Version
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false
Default	1
Invalid	Range >=1 <=1

4.6.8 VendorApiInfix (CommonPublishedInformation)

In driver modules which can be instantiated several times on a single ECU, BSW00347 requires that the name of APIs is extended by the VendorId and a vendor specific name. This parameter is used to specify the vendor specific name. In total, the implementation specific name is generated as follows:

<ModuleName>_>VendorId>_<VendorApiInfix><Api name from SWS>. E.g. assuming that the VendorId of the implementor is 123 and the implementer chose a VendorApiInfix of "v11r456" a api name Can_Write defined in the SWS will translate to Can_123_v11r456Write. This parameter is mandatory for all modules with upper multiplicity > 1. It shall not be used for modules with upper multiplicity =1.

Table 4-18. Attribute VendorApiInfix (CommonPublishedInformation) detailed description

Property	Value
Label	Vendor Api Infix
Type	STRING_LABEL
Origin	Custom
Symbolic Name	false
Default	
Enable	false

4.6.9 VendorId (CommonPublishedInformation)

Vendor ID of the dedicated implementation of this module according to the AUTOSAR vendor list.

Table 4-19. Attribute VendorId (CommonPublishedInformation) detailed description

Property	Value
Label	Vendor Id
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false
Default	43
Invalid	Range >=43 <=43

4.7 Form LinGlobalConfig

This container contains the global configuration parameter of the Lin driver. This container is a MultipleConfigurationContainer, i.e. this container and its sub-containers exit once per configuration set.

Included forms :

- [Form LinChannel](#)

4.7.1 Form LinChannel



Note


This container contains the configuration (parameters) of the LIN Controller(s).

"User should use unique names for naming the LIN channels across different LinGlobalConfig Sets."

Is included by form : [Form LinGlobalConfig](#)



LinChannel

@  



Name  LinChannel_0

General



Lin Channel ID

 0 



Lin Channel BaudRate (bps) (1000 -> 20000)

 9600 



Break Length (bits)


 BL_13 



Lin hardware channel

 LPUART_0 



LinClockRef

 nfiguration_0/McuClockSettingConfig_0/McuClockReferencePoint_0 

 LinClockRef_Alternate

 nfiguration_0/McuClockSettingConfig_0/McuClockReferencePoint_0 

Lin Channel Wake UP support*

 ☒ 

EcuM WakeUP source



 onfiguration_0/EcuMCommonConfiguration/EcuMWakeupSource_0 

Figure 4-6. Tresos Plugin snapshot for LinChannel form.

4.7.1.1 LinChannelId (LinChannel)

Identifies the LIN channel. Replaces LIN_CHANNEL_INDEX_NAME from the LIN SWS.

Table 4-20. Attribute LinChannelId (LinChannel) detailed description

Property	Value
Label	Lin Channel ID
Type	INTEGER
Origin	AUTOSAR_ECUC
Symbolic Name	true
Invalid	Range >=0 <=10

4.7.1.2 LinChannelBaudRate (LinChannel)

LinChannelBaudRate

Autosar Requirements: LIN180_Conf

Specifies the baud rate of the LIN channel in 'bps'. Valid range: 1000..20000.

Table 4-21. Attribute LinChannelBaudRate (LinChannel) detailed description

Property	Value
Label	Lin Channel BaudRate (bps)
Type	INTEGER
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	9600
Invalid	Range <=20000 >=1000

4.7.1.3 BreakLength (LinChannel)

Note

Defines the break length in bits.

This Parameter is an Implementation Specific Parameter.

Table 4-22. Attribute BreakLength (LinChannel) detailed description

Property	Value
Label	Break Length (bits)
Type	ENUMERATION
Origin	Custom
Symbolic Name	false
Default	BL_13
Range	BL_10 BL_13

4.7.1.4 LinHwChannel (LinChannel)

Note

Selects the physical LIN Channel.

This Parameter is an Implementation Specific Parameter.

Table 4-23. Attribute LinHwChannel (LinChannel) detailed description

Property	Value
Label	Lin hardware channel

Table continues on the next page...

Table 4-23. Attribute LinHwChannel (LinChannel) detailed description (continued)

Property	Value
Type	ENUMERATION
Origin	Custom
Symbolic Name	false

4.7.1.5 LinChannelWakeupSupport (LinChannel)

LinChannelWakeupSupport

Autosar Requirements: LIN182_Conf

Specifies if the LIN hardware channel supports wake up functionality.

Table 4-24. Attribute LinChannelWakeupSupport (LinChannel) detailed description

Property	Value
Label	Lin Channel Wake UP support
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	false

4.7.1.6 LinClockRef (LinChannel)

Reference to the LIN clock source configuration, which is set in the MCU driver configuration.

Table 4-25. Attribute LinClockRef (LinChannel) detailed description

Property	Value
Type	REFERENCE
Origin	AUTOSAR_ECUC

4.7.1.7 LinClockRef_Alternate (LinChannel)

Alternate reference to the LIN clock source configuration, which is set in the MCU driver configuration, used in Low Power Mode.

Table 4-26. Attribute LinClockRef_Alternate (LinChannel) detailed description

Property	Value
Type	REFERENCE
Origin	Custom

4.7.1.8 LinChannelEcuMWakeupSource (LinChannel)

Table 4-27. Attribute LinChannelEcuMWakeupSource (LinChannel) detailed description

Property	Value
Label	EcuM WakeUP source
Type	SYMBOLIC-NAME-REFERENCE
Origin	AUTOSAR_ECUC
Enable	true

How to Reach Us:**Home Page:**nxp.com**Web Support:**nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including "typicals," must be validated for each customer application by customer's technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

While NXP has implemented advanced security features, all products may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on customer's applications and products, and NXP accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP, the NXP logo, NXP SECURE CONNECTIONS FOR A SMARTER WORLD, COOLFLUX, EMBRACE, GREENCHIP, HITAG, I2C BUS, ICODE, JCOP, LIFE VIBES, MIFARE, MIFARE CLASSIC, MIFARE DESFire, MIFARE PLUS, MIFARE FLEX, MANTIS, MIFARE ULTRALIGHT, MIFARE4MOBILE, MIGLO, NTAG, ROADLINK, SMARTLX, SMARTMX, STARPLUG, TOPFET, TRENCHMOS, UCODE, Freescale, the Freescale logo, Altivec, C-5, CodeTEST, CodeWarrior, ColdFire, ColdFire+, C-Ware, the Energy Efficient Solutions logo, Kinetis, Layerscape, MagniV, mobileGT, PEG, PowerQUICC, Processor Expert, QorIQ, QorIQ Qonverge, Ready Play, SafeAssure, the SafeAssure logo, StarCore, Symphony, VortiQa, Vybrid, Airfast, BeeKit, BeeStack, CoreNet, Flexis, MXC, Platform in a Package, QUICC Engine, SMARTMOS, Tower, TurboLink, and UMEMS are trademarks of NXP B.V. All other product or service names are the property of their respective owners. Arm, AMBA, Artisan, Cortex, Jazelle, Keil, SecurCore, Thumb, TrustZone, and μ Vision are registered trademarks of Arm Limited (or its subsidiaries) in the EU and/or elsewhere. Arm7, Arm9, Arm11, big.LITTLE, CoreLink, CoreSight, DesignStart, Mali, Mbed, NEON, POP, Sensinode, Socrates, ULINK and Versatile are trademarks of Arm Limited (or its subsidiaries) in the EU and/or elsewhere. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. The Power Architecture and Power.org word marks and the Power and Power.org logos and related marks are trademarks and service marks licensed by Power.org.

© 2018 NXP B.V.

Document Number UM2LINASR4.2 Rev0002R1.0.1
Revision 1.0