

MCAL User Manual for Lin_17_AscLin

32-bit TriCore™ AURIX™ TC3xx microcontroller

About this document

Scope and purpose

This User Manual is intended to enable users to integrate the Microcontroller Abstraction Layer (MCAL) software for the TriCore™ AURIX™ family of 32-bit microcontrollers.

This document describes responsibilities of integrator in-charge of integrating MCAL software with the basic software (BSW) stack. This document also provides detailed information on safety, configuration and functions along with examples of usage of significant features.

Note: Detailed information about package installation, safety and other generic information that are common across all modules are provided in MCAL User Manual General.

Intended audience

This document is intended for anyone using the Lin_17_AscLin module of the TC3xx MCAL software.

Document conventions

Table 1 Conventions

Convention	Explanation
Bold	Emphasizes heading levels, column headings, table and figure captions, screen names, windows, dialog boxes, menus, sub-menus
<i>Italics</i>	Denotes variable(s) and reference(s)
Courier	Denotes APIs, functions, interrupt handlers, events, data types, error handlers, file/folder names, directories, command line inputs, code snippets
New	
>	Indicates that a cascading sub-menu opens when you select a menu item
[cover parentID=<alpha numeric value>]	Used for traceability completeness. Reader should ignore these.

Reference documents

This User Manual should be read in conjunction with the following documents:

- AURIX™ TC3xx MCAL User Manual General
- Specification of LIN Driver, AUTOSAR_SWS_LIN_Driver, AUTOSAR Release 4.2.2
- Specification of LIN Driver, AUTOSAR_SWS_LIN_Driver, AUTOSAR Release 4.4.0

Table of contents

Table of contents

	About this document	1
	Table of contents	2
1	Lin_17_AscLin driver	5
1.1	User information	5
1.1.1	Description	5
1.1.2	Hardware-software mapping	5
1.1.2.1	ASCLIN: primary hardware peripheral	6
1.1.2.2	PORT : dependent hardware peripheral	7
1.1.2.3	SCU: dependent hardware peripheral	8
1.1.2.4	SRC: dependent hardware peripheral	8
1.1.3	File structure	8
1.1.3.1	C file structure	8
1.1.3.2	Code generator plugin files	10
1.1.4	Integration hints	11
1.1.4.1	Integration with AUTOSAR stack	11
1.1.4.2	Multicore and Resource Manager	14
1.1.4.3	MCU support	15
1.1.4.4	Port support	16
1.1.4.5	DMA support	18
1.1.4.6	Interrupt connections	18
1.1.4.7	Example usage	21
1.1.5	Key architectural considerations	24
1.1.5.1	ASCLIN hardware: used for LIN feature	24
1.1.5.2	Modes of Operation - TxFIFO and RxFIFO Modes	24
1.1.5.3	Addition of LinMasterInterruptEnable configuration parameter	25
1.2	Assumptions of Use (AoU)	26
1.3	Reference information	27
1.3.1	Configuration interfaces	27
1.3.1.1	Container: CommonPublishedInformation	27
1.3.1.1.1	ArMajorVersion	27
1.3.1.1.2	ArMinorVersion	28
1.3.1.1.3	ArPatchVersion	28
1.3.1.1.4	ModuleId	29
1.3.1.1.5	Release	29
1.3.1.1.6	SwMajorVersion	30
1.3.1.1.7	SwMinorVersion	30
1.3.1.1.8	SwPatchVersion	31
1.3.1.1.9	VendorApilInfix	31
1.3.1.1.10	VendorId	32

Table of contents

1.3.1.2	Container: Lin	32
1.3.1.3	Container: LinChannel	32
1.3.1.3.1	LinAutoCalcBaudParams	32
1.3.1.3.2	LinChanAssignedHw	33
1.3.1.3.3	LinChannelBaudDenominator	33
1.3.1.3.4	LinChannelBaudNumerator	34
1.3.1.3.5	LinChannelBaudPreScalar	34
1.3.1.3.6	LinChannelBaudRate	35
1.3.1.3.7	LinChannelEcuMWakeupSource	35
1.3.1.3.8	LinChannelEcucPartitionRef	36
1.3.1.3.9	LinChannelId	37
1.3.1.3.10	LinChannelWakeupSupport	37
1.3.1.3.11	LinClockRef	38
1.3.1.3.12	LinInterByteSpace	38
1.3.1.3.13	LinNodeType	39
1.3.1.3.14	LinRxAlternateInputSignal	39
1.3.1.4	Container: LinDemEventParameterRefs	40
1.3.1.4.1	LIN_E_TIMEOUT	40
1.3.1.5	Container: LinGeneral	40
1.3.1.5.1	LinCsrClksel	40
1.3.1.5.2	LinDevErrorDetect	41
1.3.1.5.3	LinEcucPartitionRef	42
1.3.1.5.4	LinHwMcuTrigSleepEnable	42
1.3.1.5.5	LinIndex	43
1.3.1.5.6	LinInitApiMode	43
1.3.1.5.7	LinMasterInterruptEnable	44
1.3.1.5.8	LinMultiCoreErrorDetect	44
1.3.1.5.9	LinSysClockRef	45
1.3.1.5.10	LinTimeoutDuration	45
1.3.1.5.11	LinVersionInfoApi	46
1.3.1.6	Container: LinGlobalConfig	46
1.3.2	Functions - Type definitions	47
1.3.2.1	Lin_SlaveErrorType	47
1.3.2.2	Lin_17_AscLin_ConfigType	47
1.3.2.3	Lin_FramePidType	48
1.3.2.4	Lin_FrameCsModelType	48
1.3.2.5	Lin_FrameResponseType	48
1.3.2.6	Lin_FrameDlType	49
1.3.2.7	Lin_PduType	49
1.3.2.8	Lin_StatusType	50
1.3.3	Functions - APIs	51
1.3.3.1	Lin_17_AscLin_CheckWakeup	51

Table of contents

1.3.3.2	Lin_17_AscLin_GetStatus	52
1.3.3.3	Lin_17_AscLin_GoToSleep	54
1.3.3.4	Lin_17_AscLin_GoToSleepInternal	55
1.3.3.5	Lin_17_AscLin_Init	56
1.3.3.6	Lin_17_AscLin_SendFrame	57
1.3.3.7	Lin_17_AscLin_Wakeup	58
1.3.3.8	Lin_17_AscLin_WakeupInternal	59
1.3.3.9	Lin_17_AscLin_GetVersionInfo	60
1.3.4	Notifications and Callbacks	60
1.3.5	Scheduled functions	61
1.3.6	Interrupt service routines	61
1.3.6.1	Lin_17_AscLin_IsrError	61
1.3.6.2	Lin_17_AscLin_IsrReceive	62
1.3.6.3	Lin_17_AscLin_IsrTransmit	63
1.3.7	Callout	63
1.3.8	Errors Handling	63
1.3.9	Deviations and limitations	65
1.3.9.1	Deviations	65
1.3.9.1.1	Software specification deviations	65
1.3.9.1.2	AMDC Violations	66
1.3.9.1.3	VSMD Violations	66
1.3.9.2	Limitations	69
	Revision history	70
	Disclaimer	71

1 Lin_17_AscLin driver**1 Lin_17_AscLin driver****1.1 User information****1.1.1 Description**

The LIN driver adheres to LIN protocol 2.1 (ISO-17987) and conforms to AUTOSAR 4.2.2 and AUTOSAR 4.4.0 versions, both the versions support Master mode however the Slave mode is supported only in AUTOSAR 4.4.0 version. The ASCLIN module provides hardware support for the LIN protocol. The LIN driver provides UI options to configure the driver parameters described in the AUTOSAR LIN specification (Version AS 4.2.2 and AS 4.4.0) and additional parameters to configure the various functional blocks of ASCLIN. The LIN driver supports LIN channels in master and slave mode. The LIN driver is implemented as Post-Build variant as specified by AUTOSAR.

1.1.2 Hardware-software mapping

This section describes the system view of the LIN driver and peripherals administered by it.

1 Lin_17_AscLin driver

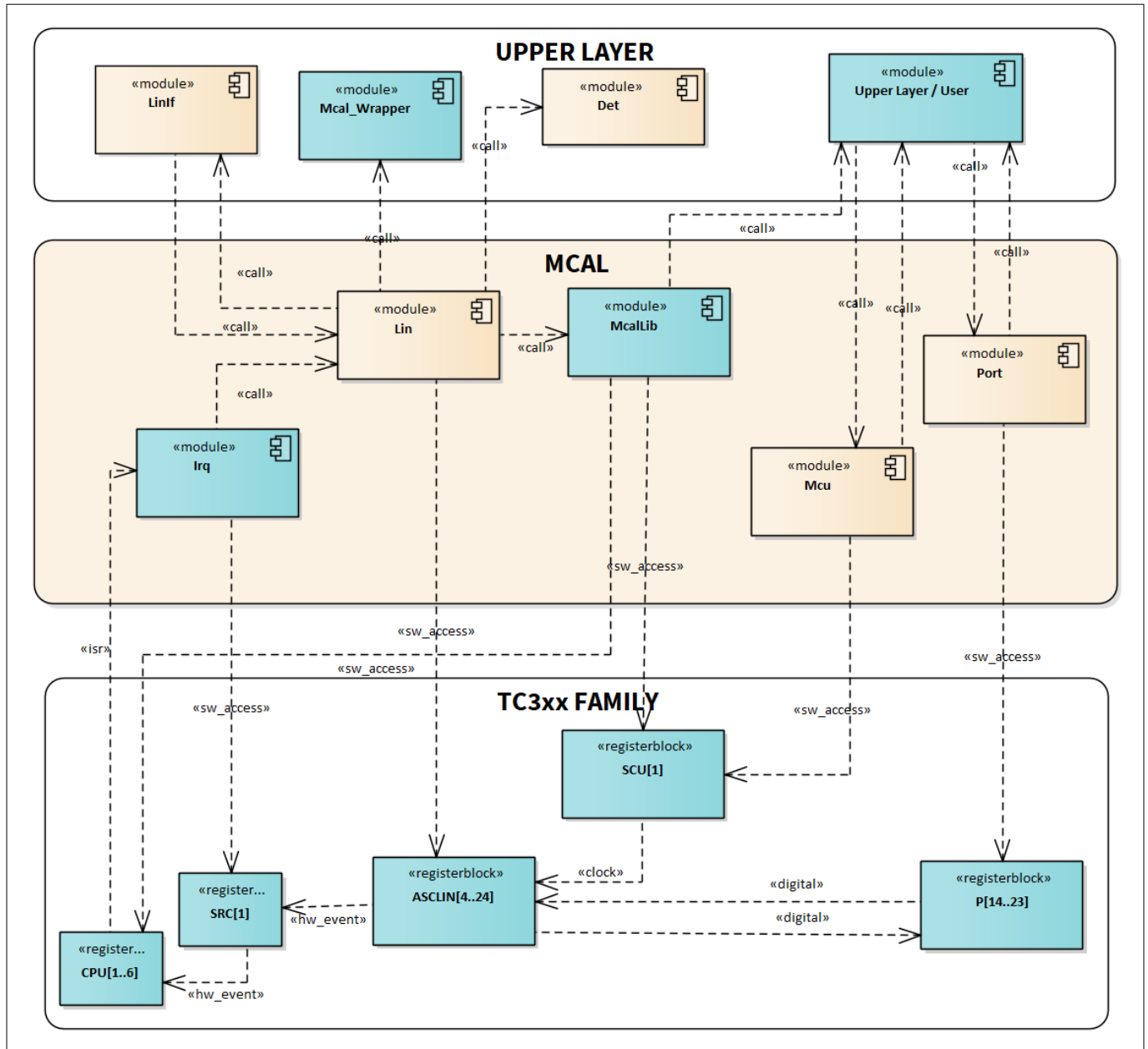


Figure 1 Mapping of hardware-software interfaces

1.1.2.1 ASCLIN: primary hardware peripheral

Hardware functional features

The LIN driver uses the ASCLIN for LIN communication in the AURIX2G MCAL implementation.

The key hardware functional features used by the driver are:

- LIN support features
- LSB first
- One stop bit
- 1 K baud to 20 K baud rate signal generation
- Tx and Rx hardware FIFO buffers

1 Lin_17_AscLin driver

- Sync break field generation
- LIN watchdogs (header timeout, response timeout) are used
- Collision detection feature
- Wake detection and generation
- Interrupts and error handlers flags

The unsupported features of the ASCLIN are:

- Autobaud detection based on sync field measurement
- Break detection
- Struck at zero/one monitoring
- Bus idle time monitoring

Users of the hardware

The LIN and UART drivers utilize the ASCLIN IP. The allocation of ASCLIN channels to LIN/UART driver is done by the MCU driver. Both LIN and UART drivers utilize only the channels allocated to them.

Hardware diagnostic features

The SMU alarms configured for the ASCLIN are not monitored by the LIN driver.

Hardware events

The LIN driver uses the following hardware events from the ASCLIN:

- Master response: Transmission of header (TH - Transmit header end flag) and response (TR - Transmit response end flag), Error in transmission of header and response, Collision error (CE - Collision detection error flag), parity error (LP - LIN parity error flag), header timeout error (HT - Header timeout flag) and detection break pulse -> Slave mode only (BD - Break detection)
- Slave response: Reception (Receive response end flag) and transmission of the response (Transmit response end flag), Error in reception (when only ID received but no response), Checksum error (CE - Collision detection error flag), Framing error (FE - Framing error flag), FIFO underflow/overflow (RFU - Receive FIFO error flag, RFO - Receive FIFO overflow flag), header timeout error (HT - Header timeout error flag), response timeout error (RT - Response timeout end flag) and detection break pulse -> Slave mode only (BD - Break detection)
- Slave to Slave: Error in transmission of Header (TH - Transmit header end flag) Collision error, (CE - Collision detection error flag), parity error (LP - LIN parity error flag), header timeout error (HT - Header timeout flag) and detection break pulse -> Slave mode only (BD - Break detection)

1.1.2.2 PORT : dependent hardware peripheral

Hardware functional features

The ATX and ARX signals are routed to the ASCLIN through the digital port pad. These are configured and enabled through the PORT driver.

Users of the hardware

The port pads are configured by the PORT driver.

Hardware diagnostic features

Not applicable.

1 Lin_17_AscLin driver**Hardware events**

Hardware events from port pads are not used by the LIN driver.

1.1.2.3 SCU: dependent hardware peripheral**Hardware functional features**

The LIN driver depends on the SCU IP for the clock, ENDINIT and reset functionalities. The driver requires the fSPB and fASCLIN clock signals for functioning.

Users of the hardware

The SCU IP supplies clock for all the peripherals and the MCU driver is responsible for configuring the clock tree. To avoid conflicts due to simultaneous writes, update to all the ENDINIT protected registers is performed using the MCALLIB APIs.

Hardware diagnostic features

The SMU alarms configured for the SCU IP are not monitored by the LIN driver.

Hardware events

Hardware events from the SCU are not used by the LIN driver.

1.1.2.4 SRC: dependent hardware peripheral**Hardware functional features**

The LIN driver depends on the interrupt router for raising an interrupt to the CPU based on the transmit/receive/error events, which indicates successful frame transmission and reception respectively.

Users of the hardware

The interrupt router is configured either by the IRQ driver or the user software.

Hardware diagnostic features

The SMU alarms configured for the interrupt router are not monitored by the LIN driver.

Hardware events

The interrupt events raised by the interrupt router are serviced by the CPU. The LIN driver provides interrupt handlers as software interfaces, which must be invoked from the ISR.

1.1.3 File structure**1.1.3.1 C file structure**

This section provides details of the C files of the LIN driver.

1 Lin_17_AscLin driver

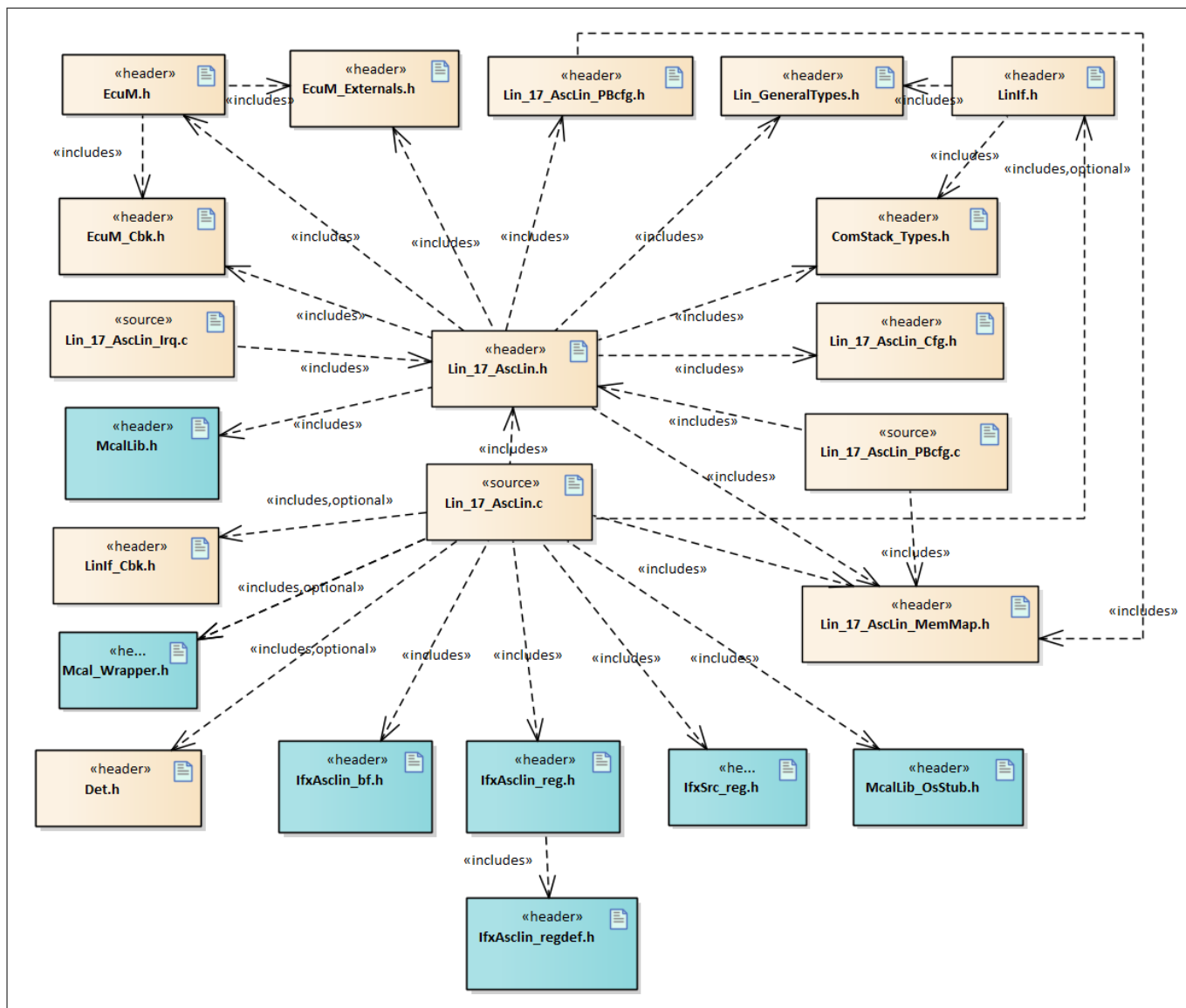


Figure 2 Lin_File_Structure-1.png

Table 2 C file structure

File name	Description
ComStack_Types.h	Type Definition for Com stack
Det.h	Provides the exported interfaces of Development Error Tracer
EcuM.h	Header file exporting the declarations of the EcuM
EcuM_Cbk.h	Header file containing declarations of the EcuM callbacks. <i>Note: This file is available only for AUTOSAR version 4.2.2</i>
EcuM_Externals.h	Header file containing declarations of the EcuM callbacks. <i>Note: This file is available only for AUTOSAR version 4.4.0</i>
IfxAsclin_bf.h	SFR header file for ASCLIN
IfxAsclin_reg.h	SFR header file for ASCLIN
IfxAsclin_regdef.h	SFR header file for ASCLIN

(table continues...)

1 Lin_17_AscLin driver
Table 2 (continued) C file structure

File name	Description
IfxSrc_reg.h	SFR header file for Interrupt Controller
LinIf.h	Header file containing the exported interfaces of LinIf.
LinIf_Cbk.h	The header file contains the function declarations for the callback functions in the LIN Interface. <i>Note: This file is available only for AUTOSAR version 4.2.2</i>
Lin_17_AscLin.c	File (Static) containing implementation of the APIs
Lin_17_AscLin.h	Header file (Static) defining prototypes of data structures, the APIs and Interrupt handlers
Lin_17_AscLin_Cfg.h	Header file (Generated) containing constants and pre-processor macros as #defines
Lin_17_AscLin_Irq.c	Interrupt handler file for LIN
Lin_17_AscLin_MemMap.h	File (Static) containing the memory section definitions used by the LIN driver
Lin_17_AscLin_PBcfg.c	File (Generated) containing definition of the configuration data structures
Lin_17_AscLin_PBcfg.h	File (Generated) containing declaration of the post-build configuration data structures
Lin_GeneralTypes.h	The header file includes general LIN type declarations
McalLib.h	Static header file defining prototypes of data structure and APIs exported by the MCALLIB.
McalLib_OsStub.h	McalLib_OsStub.h provides macros to support user mode of Tricore. This shall be included by other drivers to call OS APIs.
Mcal_Wrapper.h	Provides the exported interfaces for Production Error and Runtime Development Errors. Implemented by default to include functions of Dem.h and Det.h files. This file can be modified by the user but function prototype is not user modifiable.

1.1.3.2 Code generator plugin files

This section provides details of the code generator plugin files of the LIN driver.

1 Lin_17_AscLin driver

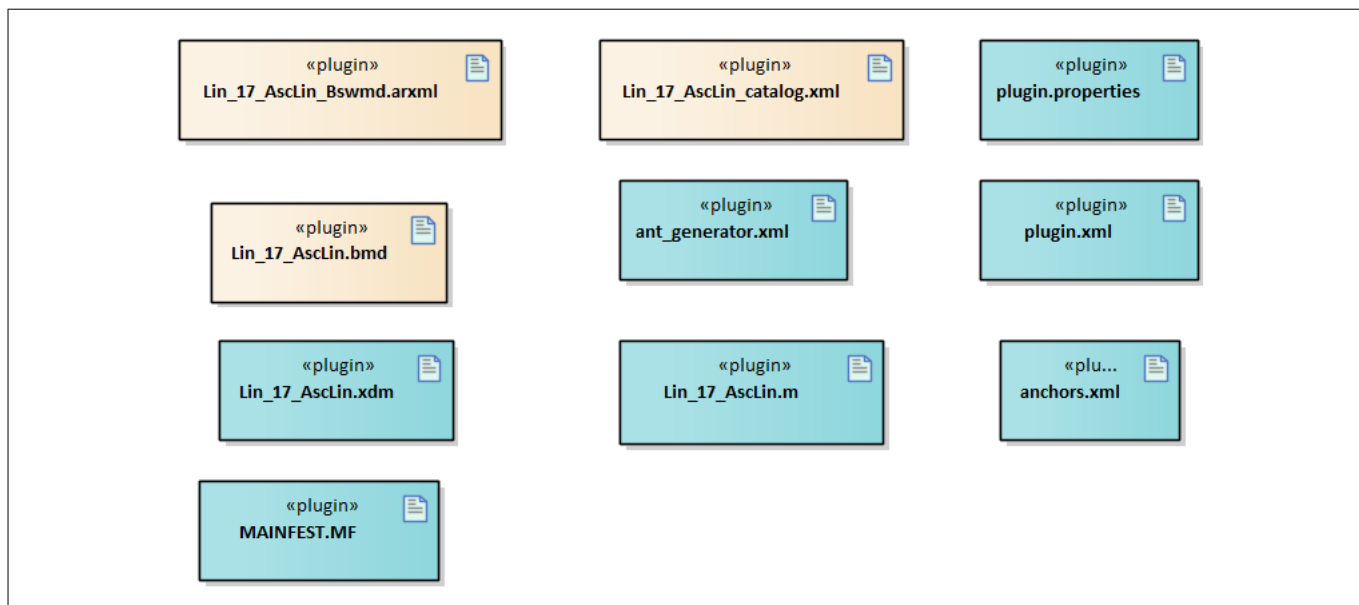


Figure 3 Lin_Code_Generator_Plugin_Files-1.png

Table 3 Code generator plugin files

File name	Description
Lin_17_AscLin.bmd	AUTOSAR format XML data model schema file
Lin_17_AscLin.m	Code template macro file for the LIN driver
Lin_17_AscLin.xdm	Tresos format XML data model schema file
Lin_17_AscLin_Bswmd.arxml	AUTOSAR format module description file
Lin_17_AscLin_catalog.xml	AUTOSAR format catalog file
MAINFEST.MF	Tresos plugin support file containing the meta data for the LIN driver
anchors.xml	Tresos anchors support file for the LIN driver
ant_generator.xml	Tresos support file to generate and rename multiple post-build configurations when using variation point
plugin.properties	Tresos plugin support file for the LIN driver
plugin.xml	Tresos plugin support file for the LIN driver

1.1.4 Integration hints

This section lists the key points that an integrator or user of the LIN driver must consider.

1.1.4.1 Integration with AUTOSAR stack

This section lists the modules, which are not part of the MCAL, but are required to integrate the LIN driver.

- **EcuM**

The ECU Manager module is a part of the AUTOSAR stack that manages common aspects of ECU. Specifically, in the context of the MCAL, the EcuM is used for initialization and de-initialization of the software drivers. To start data communication on LIN bus, the EcuM module has to initialize other related AUTOSAR BSW modules such as LIN Interface (LinIf). The LIN module notifies about the wakeup event to

1 Lin_17_AscLin driver

the EcuM module. The EcuM module provided in the MCAL package is a stub code and needs to be replaced with a complete EcuM module during the integration phase.

- **LIN Interface (LinIf)**

The LIN Interface module is a part of the AUTOSAR stack that provides upper layer a hardware independent interface to the LIN communication system.

The LIN driver uses the APIs of LinIf to provide notifications as listed.

`LinIf_WakeupConfirmation()`: Notification for wakeup confirmation and indicate successful detection of wakeup signal.

`LinIf_TxConfirmation()`: Notification for a successfully transmission of a LIN frame.

`LinIf_RxIndication()`: Notification for a successful reception of LIN response.

`LinIf_LinErrorIndication()`: Notification for any error in transmission or reception.

The LIN Interface module provided in the MCAL package is a stub code and needs to be replaced with a complete LIN Interface module during the integration phase.

- **Memory mapping**

Memory mapping is a concept from AUTOSAR that allows re location of text, variables, constants and configuration data to user-specific memory regions. To achieve this, all the re-locatable elements of the driver are encapsulated in different memory-section macros. These macros are defined in the `Lin_17_AscLin_MemMap.h` file.

The `Lin_17_AscLin_MemMap.h` file is provided in the MCAL package as a stub code. The integrator must place appropriate compiler pragmas within the memory-section macros. The pragmas ensure that the elements

1 Lin_17_AscLin driver

are re-located to the correct memory region. A sample implementation listing the memory-section macros is shown as follows.

```

/***** GLOBAL RAM DATA -- NON CLEARED LMU *****/
#if defined LIN_17_ASCLIN_START_SEC_VAR_CLEARED_QM_GLOBAL_UNSPECIFIED
/*****User pragmas here for Non-cached LMU*****/
#undef LIN_17_ASCLIN_START_SEC_VAR_CLEARED_QM_GLOBAL_UNSPECIFIED
#undef MEMMAP_ERROR
#elif defined LIN_17_ASCLIN_STOP_SEC_VAR_CLEARED_QM_GLOBAL_UNSPECIFIED
/*****User pragmas here for Non-cached LMU*****/
#undef LIN_17_ASCLIN_STOP_SEC_VAR_CLEARED_QM_GLOBAL_UNSPECIFIED
#undef MEMMAP_ERROR

/***** CORE[x] CONFIG DATA --PF[x] *****/
#elif defined LIN_17_ASCLIN_START_SEC_CONFIG_DATA_QM_CORE0_UNSPECIFIED
/*****User pragmas here for PF[x] *****/
#undef LIN_17_ASCLIN_START_SEC_CONFIG_DATA_QM_CORE0_UNSPECIFIED
#undef MEMMAP_ERROR

#elif defined LIN_17_ASCLIN_STOP_SEC_CONFIG_DATA_QM_CORE0_UNSPECIFIED
/*****User pragmas here for PF[x] *****/
#undef LIN_17_ASCLIN_STOP_SEC_CONFIG_DATA_QM_CORE0_UNSPECIFIED
#undef MEMMAP_ERROR

/***** CODE -- PF[x] *****/
#elif defined LIN_17_ASCLIN_START_SEC_CODE_QM_GLOBAL
/*****User pragmas here for PF[x] *****/
#undef LIN_17_ASCLIN_START_SEC_CODE_QM_GLOBAL
#undef MEMMAP_ERROR

#elif defined LIN_17_ASCLIN_STOP_SEC_CODE_QM_GLOBAL
/*****User pragmas here for PF[x] *****/
#undef LIN_17_ASCLIN_STOP_SEC_CODE_QM_GLOBAL
#undef MEMMAP_ERROR
#endif

#if defined MEMMAP_ERROR
#error "Lin_17_AscLin_MemMap.h, wrong pragma command"
#endif

```

- **DET**

The DET module is a part of the AUTOSAR stack that handles all the development and runtime errors reported by the BSW modules. The LIN driver reports all the development errors to the DET module through the `Det_ReportError()` API. The user of the LIN driver must process all the errors reported to the DET module through the `Det_ReportError()` API.

The files `Det.h` and `Det.c` are provided in the MCAL package as a stub code and needs to be replaced with a complete DET module during the integration phase.

- **Mcal_Wrapper**

This Driver performs reporting of the Production and Runtime errors. The Handling of the reported errors shall be done by the user. The `Mcal_Wrapper_Det_ReportRuntimeError()` API, `Mcal_Wrapper_Dem_SetEventStatus()` API and `Mcal_Wrapper_Dem_ReportErrorStatus()` API are provided in the

1 Lin_17_AscLin driver

Mcal_Wrapper.c and Mcal_Wrapper.h files as a stub code, and can be updated by the integrator to handle the reported errors. The files Mcal_Wrapper.c and Mcal_Wrapper.h are user modifiable but function prototype is not user modifiable and by default the Mcal Wrapper function shall call AUTOSAR DEM and DET Modules.

The user of the LIN driver shall process all the production errors (fail/pass) reported to the Mcal_Wrapper module. The interface used for reporting production error in AUTOSAR version 4.2.2 is Mcal_Wrapper_Dem_ReportErrorStatus() and for AUTOSAR version 4.4.0 is Mcal_Wrapper_Dem_SetEventStatus() API. The Mcal_Wrapper.c and Mcal_Wrapper.h files are provided in the MCAL package as a stub code and can be replaced with a user specific production error handling module/s during the integration phase.

- **SchM**

The SchM is not required for the integration of LIN driver.

- **Safety error**

The LIN driver does not report any safety errors.

- **Notifications and callbacks**

The LIN driver does not implement any notifications. However, the LIN driver reports wakeup confirmation for master and slave channels and transmit confirmation, receive indication and error indication for slave channels only through callback functions of the LinIf module.

- **Operating system (OS)**

The OS or the application must ensure correct type of service and interrupt priority is configured in the SR register. Enabling and disabling of interrupts must also be managed by the OS or application.

The OS files provided by MCAL package are only an example code and must be updated by the integrator with the actual OS files for the desired function.

1.1.4.2 Multicore and Resource Manager

The LIN driver supports execution of its APIs simultaneously from all CPU cores. The user has to allocate resources of ASCLIN to CPU cores at pre-compile time using the Resource Manager module. The following are the key points to be considered with respect to multicore in the driver:

- LIN channels of the LIN driver can be allocated to CPU cores at pre-compile time.
- LIN channels that are not allocated to a CPU core shall be by default allocated to the master core.
- DETs are raised in case APIs are invoked with mismatch of CPU core and channel IDs.
- Interrupts raised by a hardware unit must be serviced by the CPU core to which the hardware unit has been allocated to.
- Locating of constants, variables and configuration data to correct memory space should be done by the user. Memory sections are marked GLOBAL(common to all cores) and CORE[x](specific to a CPU core). The following should be considered by the user to ensure better performance of the driver:

Code section:

The executable code of LIN driver is placed under single MemMap section. It can be relocated to any PFlash region.

Data section:

The RAM variable memory sections marked as specific to a core should be re-located to the DSPR/DLMU of the same core. The sections marked as global should be relocated to the non-cached LMU region.

Configuration data and constants:

The configuration data sections marked as specific to a core should be re-located to the PFLASH of the same core. The sections marked as global should be relocated to the PFlash of the master core.

Note: Re-locating of code, data or constants to a distant memory region would impact execution timings.

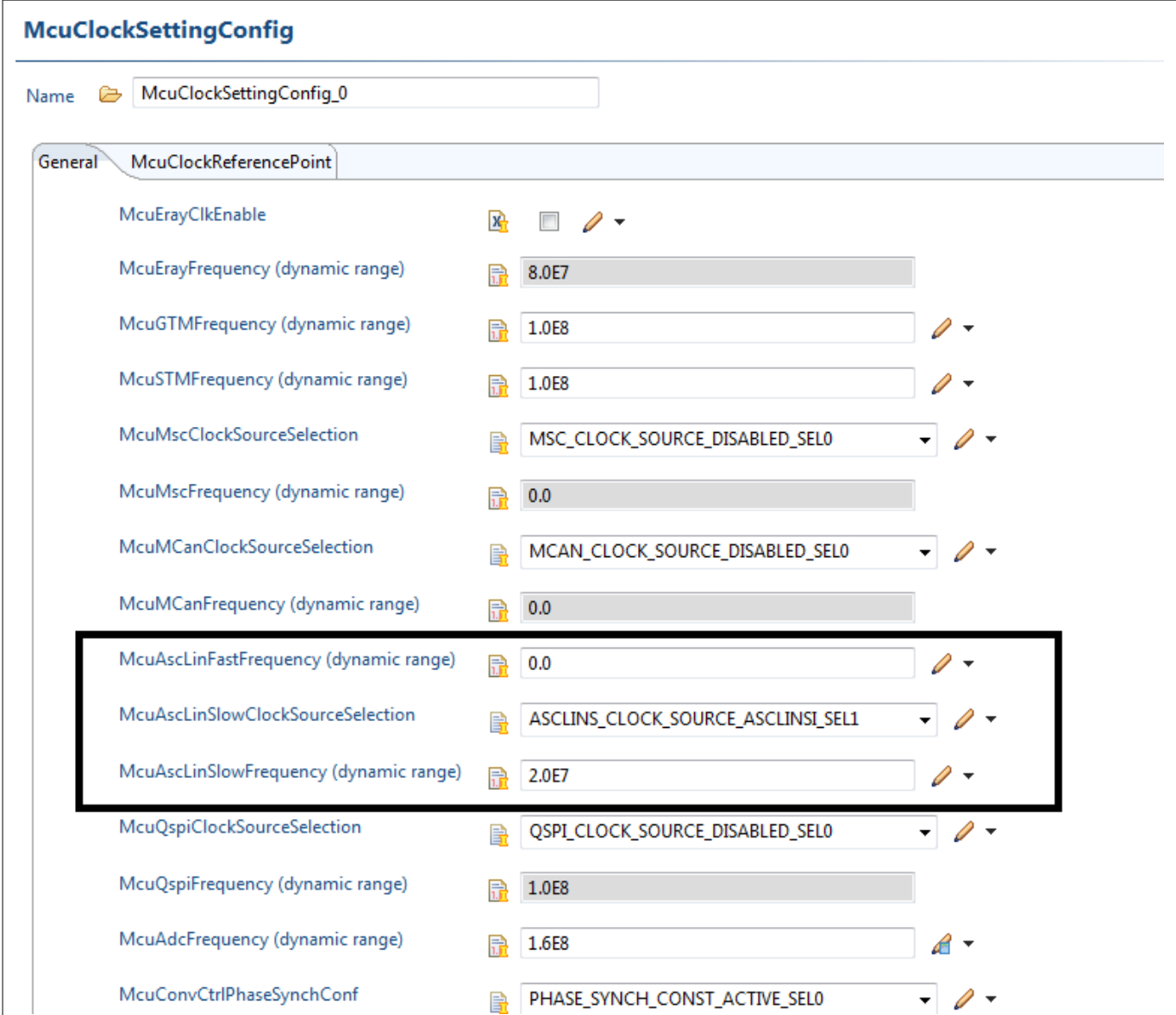
1 Lin_17_AscLin driver

Note: If the driver operates from a single(master) core, all the sections may be relocated to the PFlash/DSPR/DLMU of the same CPU core.


1.1.4.3 MCU support

The LIN driver is dependent on the MCU driver for clock configuration and channel allocation services. The initialization of the LIN driver must be started only after completing the MCU initialization. The following must be considered while configuring the MCU driver in the EB tresos tool:

The `fASCLINF` or `fASCLINS` defines the clock frequency for the ASCLIN kernel. To configure clock frequency for `fASCLINF` or `fASCLINS` refer to the `McuAscLinFastFrequency` and `McuAscLinSlowFrequency` parameters from the MCU driver configuration as follows:



McuClockSettingConfig

Name  McuClockSettingConfig_0

General **McuClockReferencePoint**




























McuErayClkEnable	  
McuErayFrequency (dynamic range)	 8.0E7
McuGTMFrequency (dynamic range)	 1.0E8 
McuSTMFrequency (dynamic range)	 1.0E8 
McuMscClockSourceSelection	 MSC_CLOCK_SOURCE_DISABLED_SELO 
McuMscFrequency (dynamic range)	 0.0
McuMCanClockSourceSelection	 MCAN_CLOCK_SOURCE_DISABLED_SELO 
McuMCanFrequency (dynamic range)	 0.0
McuAscLinFastFrequency (dynamic range)	 0.0 
McuAscLinSlowClockSourceSelection	 ASCLINS_CLOCK_SOURCE_ASCLINSI_SEL1 
McuAscLinSlowFrequency (dynamic range)	 2.0E7 
McuQspiClockSourceSelection	 QSPI_CLOCK_SOURCE_DISABLED_SELO 
McuQspiFrequency (dynamic range)	 1.0E8
McuAdcFrequency (dynamic range)	 1.6E8 
McuConvCtrlPhaseSynchConf	 PHASE_SYNCH_CONST_ACTIVE_SELO 

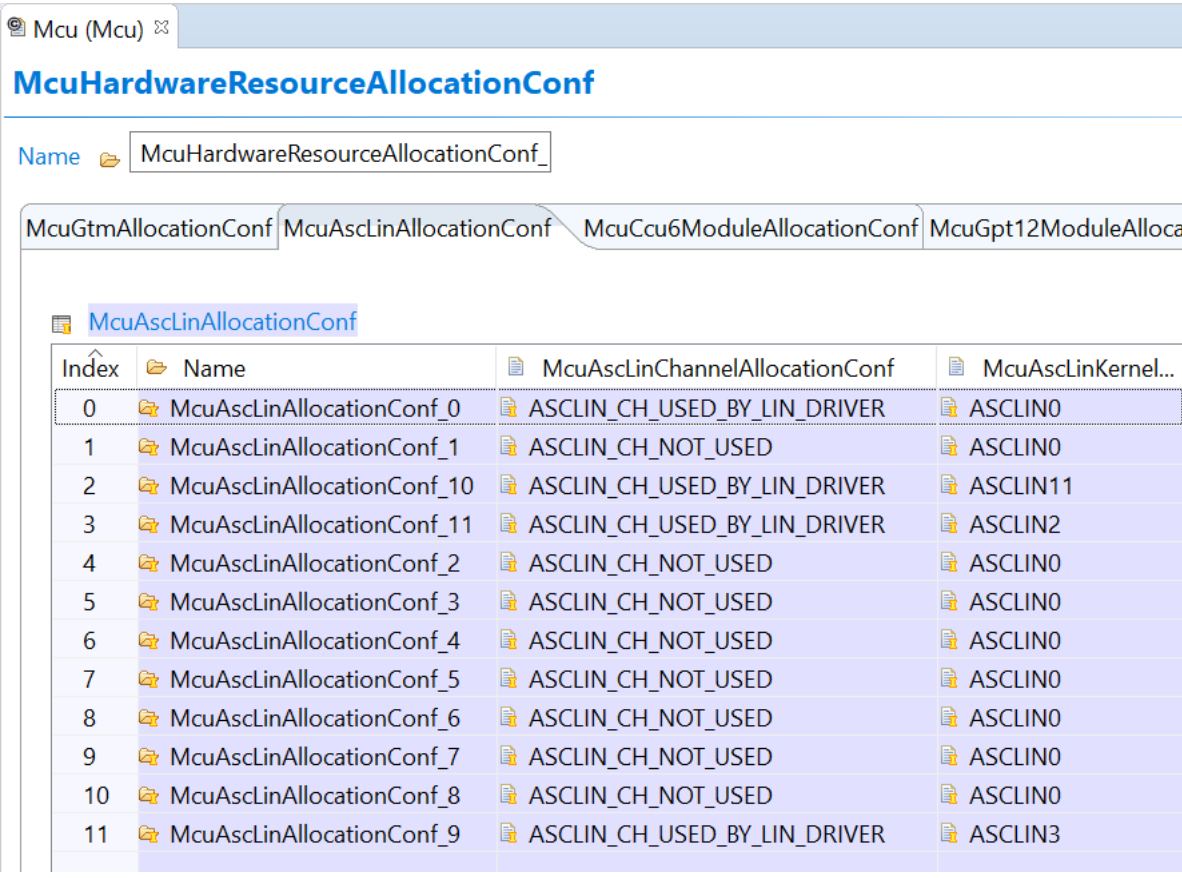
Figure 4 LIN fast or slow clock frequency configuration

Note: LinCsrClkSel configuration parameter in the LIN driver configuration selects the frequency to be used.

The ASCLIN hardware IP is shared between the LIN and the UART drivers. So the LIN driver also depends on MCU driver for allocation of the ASCLIN kernels to the driver. The resource allocation for the ASCLIN kernels configured from the MCU driver using `McuHardwareResourceAllocationConf` container.

1 Lin_17_AscLin driver

Following is the example of allocation of ASCLINs to the the LIN driver.



Index	Name	McuAscLinChannelAllocationConf	McuAscLinKernel...
0	McuAscLinAllocationConf_0	ASCLIN_CH_USED_BY_LIN_DRIVER	ASCLIN0
1	McuAscLinAllocationConf_1	ASCLIN_CH_NOT_USED	ASCLIN0
2	McuAscLinAllocationConf_10	ASCLIN_CH_USED_BY_LIN_DRIVER	ASCLIN11
3	McuAscLinAllocationConf_11	ASCLIN_CH_USED_BY_LIN_DRIVER	ASCLIN2
4	McuAscLinAllocationConf_2	ASCLIN_CH_NOT_USED	ASCLIN0
5	McuAscLinAllocationConf_3	ASCLIN_CH_NOT_USED	ASCLIN0
6	McuAscLinAllocationConf_4	ASCLIN_CH_NOT_USED	ASCLIN0
7	McuAscLinAllocationConf_5	ASCLIN_CH_NOT_USED	ASCLIN0
8	McuAscLinAllocationConf_6	ASCLIN_CH_NOT_USED	ASCLIN0
9	McuAscLinAllocationConf_7	ASCLIN_CH_NOT_USED	ASCLIN0
10	McuAscLinAllocationConf_8	ASCLIN_CH_NOT_USED	ASCLIN0
11	McuAscLinAllocationConf_9	ASCLIN_CH_USED_BY_LIN_DRIVER	ASCLIN3

Figure 5 LIN resource allocation using MCU

The MCU initialization must be completed prior to invoking the LIN initialization. The configured ASCLINs can be added in Mcu driver by adding the ASCLINs in the parameter `McuAscLinKernelId`.

1.1.4.4 Port support

The PORT driver configures the port pins of the entire microcontroller. The user must configure port pins used by the LIN driver through the PORT configuration and initialize the port pins prior to invoking the LIN initialization. The following must be considered while configuring PORT driver in the EB tresos tool:

- The TxD and RxD pins of the different ASCLIN kernels must be configured with respective direction and configuration in the PORT module. For RxD pin the selection for suitable port and pin shall be made in LIN driver configuration using `LinRxAlternateInputSignal` configuration parameter for a ASCLIN kernel.

Refer to the following sample configuration for the PORT driver.

1 Lin_17_AscLin driver

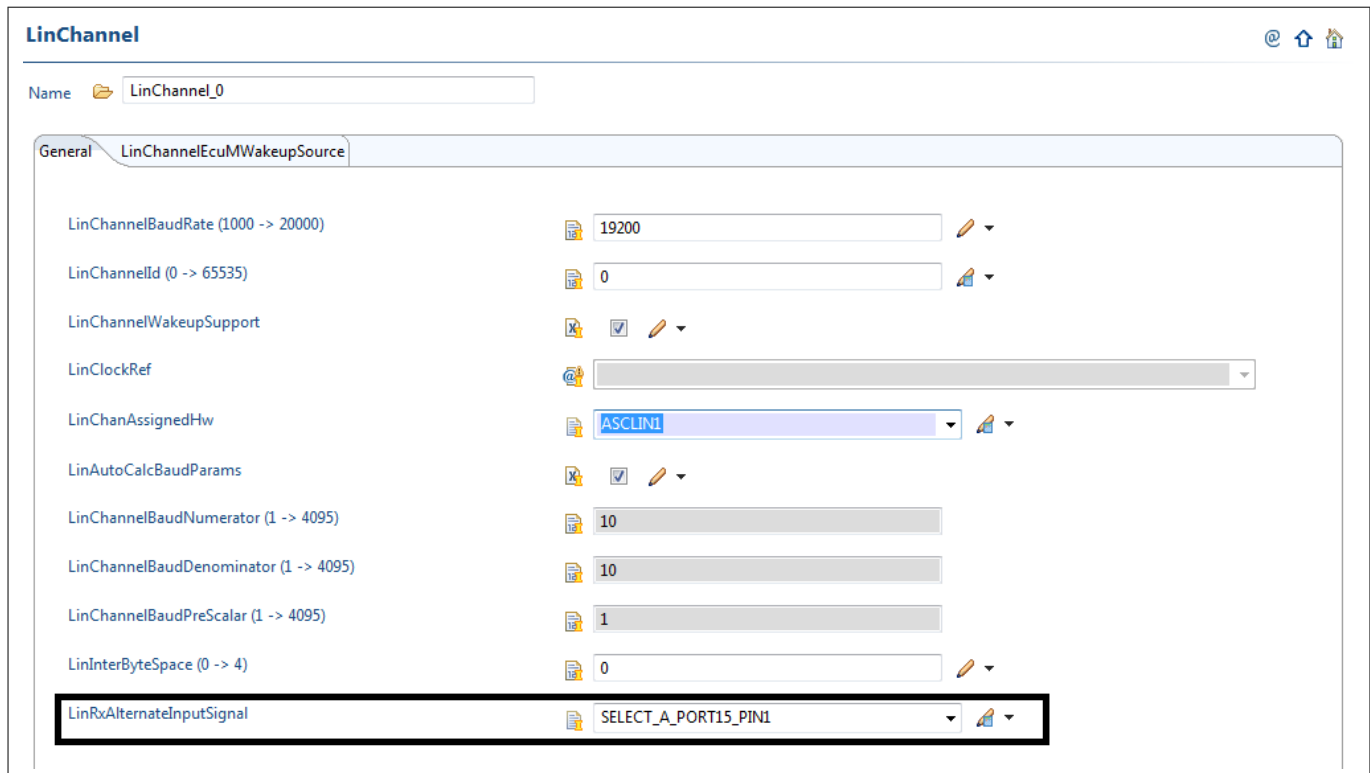


Figure 6 RX input pin selection from LIN configuration for a LIN channel

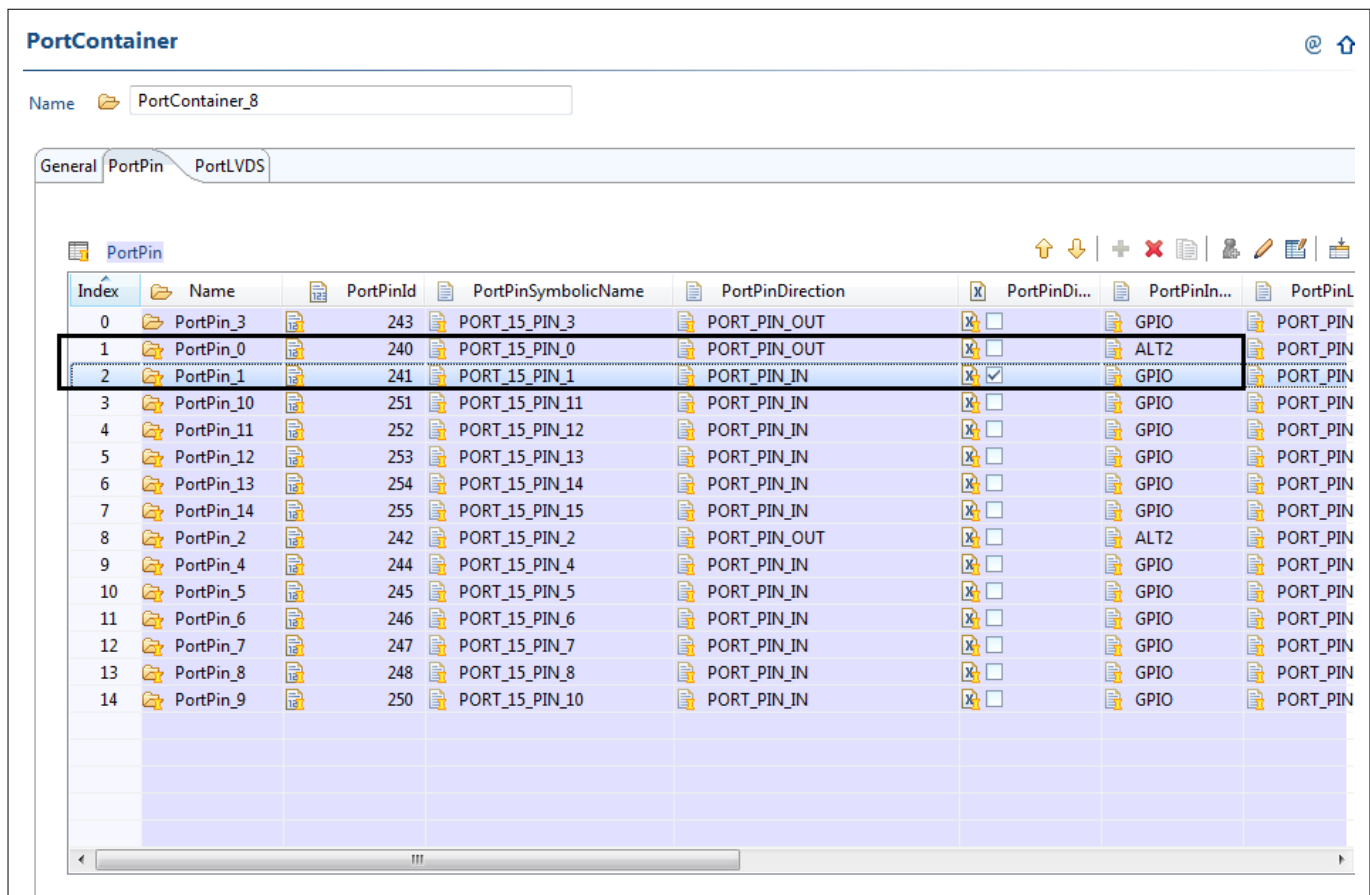


Figure 7 Port pin configuration from PORT module

1 Lin_17_AscLin driver

1.1.4.5 DMA support

The LIN driver does not use any services provided by the DMA driver.

1.1.4.6 Interrupt connections

The following interrupts in LIN are to be configured:

- Tx - Transmit complete interrupt

This interrupt triggers when the header transmission or the master response transmission is completed successfully. A sample invocation for Tx interrupt handler is as follows:

```
#include "Lin_17_AscLin.h"
#include "Irq.h"

/*****TX Interrupt for ASCLIN0 *****/
IFX_INTERRUPT(ASCLIN0TX_ISR, 0, IRQ_ASCLIN0_TX_PRIO)
ISR(ASCLIN0TX_ISR)
{
    /* Enable Global Interrupts */
    ENABLE();
    /* Call Lin Interrupt function*/
    Lin_17_AscLin_IsrTransmit(0U);
}
```

- Rx - Receive complete interrupt

This interrupt triggers when slave response reception is completed successfully for master / slave channels and triggers when header reception is completed successfully for slave channels. A sample invocation for Rx interrupt handler is depicted below:

```
#include "Lin_17_AscLin.h"
#include "Irq.h"

/*****RX Interrupt for ASCLIN0 *****/
IFX_INTERRUPT(ASCLIN0RX_ISR, 0, IRQ_ASCLIN0_RX_PRIO)
ISR(ASCLIN0RX_ISR)
{
    /* Enable Global Interrupts */
    ENABLE();
    /* Call Lin Interrupt function*/
    Lin_17_AscLin_IsrReceive(0U);
}
```

- Err - Error event interrupt

1 Lin_17_AscLin driver

This interrupt triggers when erroneous event occurs like collision error, header error for master / slave channels and incomplete / no response error and framing errors for slave channels.

```
#include "Lin_17_AscLin.h"
#include "Irq.h"

/*****Err Interrupt for ASCLIN0 *****/
IFX_INTERRUPT(ASCLIN0ERR_ISR, 0, IRQ_ASCLIN0_ERR_PRIO)
ISR(ASCLIN0ERR_ISR)
{
    /* Enable Global Interrupts */
    ENABLE();
    /* Call Lin Interrupt function*/
    Lin_17_AscLin_IsrError(0U);
}
```

Configuration of interrupt category and priority, shall be configured in IRQ module. Following are interrupt configuration example for ASCLIN0.

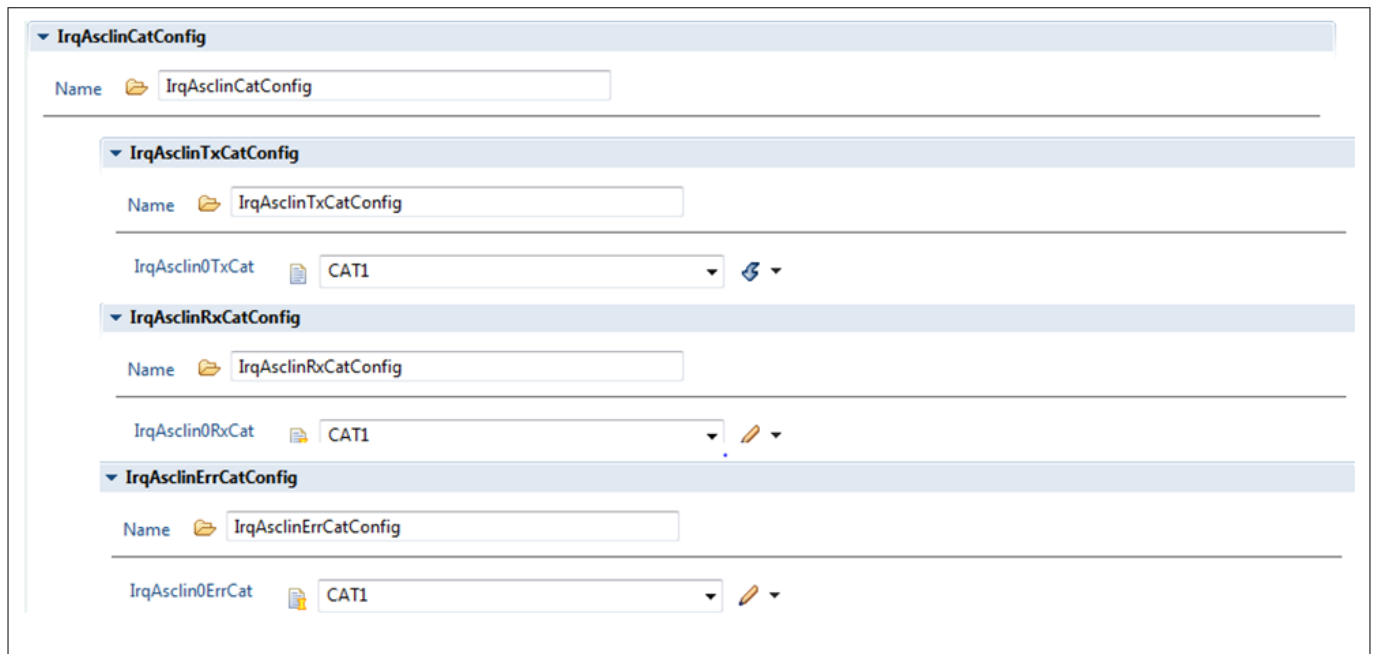
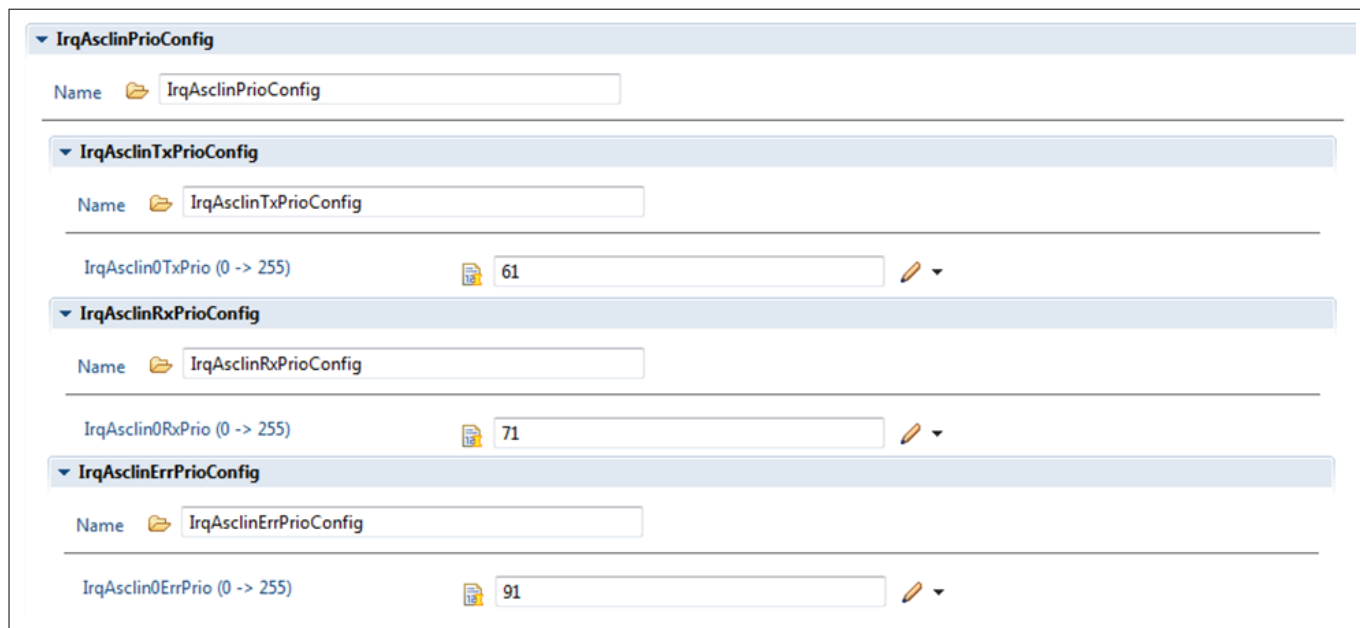


Figure 8 Interrupt category configuration for ASCLIN0

1 Lin_17_AscLin driver

The screenshot displays a configuration window for the Lin_17_AscLin driver. It is organized into four main sections, each with a dropdown arrow on the left:

- IrqAsclnPrioConfig**: Contains a 'Name' field with a folder icon and the text 'IrqAsclnPrioConfig'.
- IrqAsclnTxPrioConfig**: Contains a 'Name' field with a folder icon and the text 'IrqAsclnTxPrioConfig'. Below it, the label 'IrqAscln0TxPrio (0 -> 255)' is followed by a numeric input field containing '61' and a small edit icon.
- IrqAsclnRxPrioConfig**: Contains a 'Name' field with a folder icon and the text 'IrqAsclnRxPrioConfig'. Below it, the label 'IrqAscln0RxPrio (0 -> 255)' is followed by a numeric input field containing '71' and a small edit icon.
- IrqAsclnErrPrioConfig**: Contains a 'Name' field with a folder icon and the text 'IrqAsclnErrPrioConfig'. Below it, the label 'IrqAscln0ErrPrio (0 -> 255)' is followed by a numeric input field containing '91' and a small edit icon.

Figure 9 **Interrupt priority configuration for ASCLIN0**

1 Lin_17_AscLin driver

1.1.4.7 Example usage

Examples of LIN driver API usage are as follows:

Configuration of the driver

The LIN driver must be configured before usage and configuration files should be generated and made available during the software build process.

To configure the LIN driver, the following guidelines should be followed.

- Configuration of the system clock: Before using the LIN driver, the MCU driver should be configured and initialized so that the system clock is up and running at the required frequency. This configuration is done using the MCU driver.
- Configuration of the port pins: The TXD and RXD (for the relevant Rx pin select) pins of the LIN channels should be configured using the PORT driver.
- Configuration of LIN interrupts: For LIN drivers with interrupt mode enabled, configure the interrupt priority, type of service and interrupt type in the IRQ driver.
- Configuration of the LIN driver: Select the required API configuration and choose channel dependent parameters such as baud-rate, wakeup support, inter byte space and so on.

Initialization of LIN driver

Follow the sequence in the application code:

1. Initialize the MCU driver and clock using the `Mcu_Init()` API.
2. Initialize the PORT driver using the `Port_Init()` API.
3. Initialize the IRQ driver to enable the interrupt generation.
4. Initialize the LIN driver using the `Lin_17_AscLin_Init()` API.

Sample code for LIN driver initialization is as follows:

```
#include "Port.h"
#include "Port_PBcfg.h"
#include "Mcu.h"
#include "Mcu_PBcfg.h"
#include "McalLib.h"
#include "Irq.h"

/* Init MCU */
Mcu_Init(&Mcu_Config);
Mcu_InitClock(0U);
while(Mcu_GetPllStatus() != MCU_PLL_LOCKED);
Mcu_DistributePllClock();

/* Init Port */
Port_Init(&Port_Config);

/* Init Irq */
IrqAscLin_Init();

/* Init Driver */
Lin_17_AscLin_Init(&Lin_17_AscLin_Config);
```

Master response communication

1 Lin_17_AscLin driver

The code sequence sends a header followed by master response data.

```

/* dummy sdu data for master */
uint8 Sdu_Data[][8] =
{
    {1,2,3,4,5,6,7,8},
    {0xFF, 0x00, 0xFF, 0x00,0xFF, 0x00,0xFF, 0x00}
};

/* dummy pdu data */
Lin_PduType Lin_Pdu[] =
{
    {0x80, LIN_ENHANCED_CS, LIN_MASTER_RESPONSE, 8, Sdu_Data[0]},
    {0xC1, LIN_ENHANCED_CS, LIN_SLAVE_RESPONSE, 8, Sdu_Data[1]},
    {0xC2, LIN_ENHANCED_CS, LIN_SLAVE_TO_SLAVE, 8, Sdu_Data[1]}
};

uint8 DataRead[8];
/* dummy Shadow pointer for slave response received data */
volatile uint8 *SlaveSduPtr = DataRead;

void Lin_Master_Response(void)
{
    Std_ReturnType Ret1;
    Std_ReturnType Ret2;

    /* Master response header and frame transmission as referred in the dummy Lin_Pdu[0]*/
    Ret1 = Lin_17_AscLin_SendFrame(LinConf_LinChannel_LinChannel_0, &Lin_Pdu[0]);
    /* Ret1 is E_NOT_OK then Error in sending Frame.... */

    do
    {
        Ret2 = Lin_17_AscLin_GetStatus(LinConf_LinChannel_LinChannel_0, (uint8**)&SlaveSduPtr);

        if ((Ret2 != LIN_TX_BUSY) && (Ret2 != LIN_TX_OK)&& (Ret2 != OPERATIONAL_STATE))
        {
            /* Transmit Error.... */
            break;
        }
    }while(Ret2 != LIN_TX_OK );

    if (Ret2 == LIN_TX_OK)
    {
        /* LIN Header (PID 0x80) Transmitted sucessfully.... */
        /* Data (0x1,0x2,0x3,0x4,0x5,0x6,0x7,0x8) is transmitted. can be verify it at the slave.... */
    }
    else
    {
        /* Lin Master->Slave Response Failed. Response code: Ret2 */
    }
}

```

1 Lin_17_AscLin driver

Master to Slave response communication

Slave node as defined in LDF (LIN Description File) publishes the response and master node is the subscriber for the data frame.

The code sequence sends a header followed by slave response received by master.

```
void Lin_Master_To_Slave_Response(void)
{
    Std_ReturnType Ret1;
    Std_ReturnType Ret2;

    /* Slave response header transmission and reception of the slave response as referred in the
    dummy Lin_Pdu[1]*/
    Ret1 = Lin_17_AscLin_SendFrame(LinConf_LinChannel_LinChannel_0, &Lin_Pdu[1]);
    /* Ret1 is E_NOT_OK then Lin Slave->Master Response Failed.... */

    do
    {
        Ret2 = Lin_17_AscLin_GetStatus(LinConf_LinChannel_LinChannel_0, (uint8**)&SlaveSduPtr);

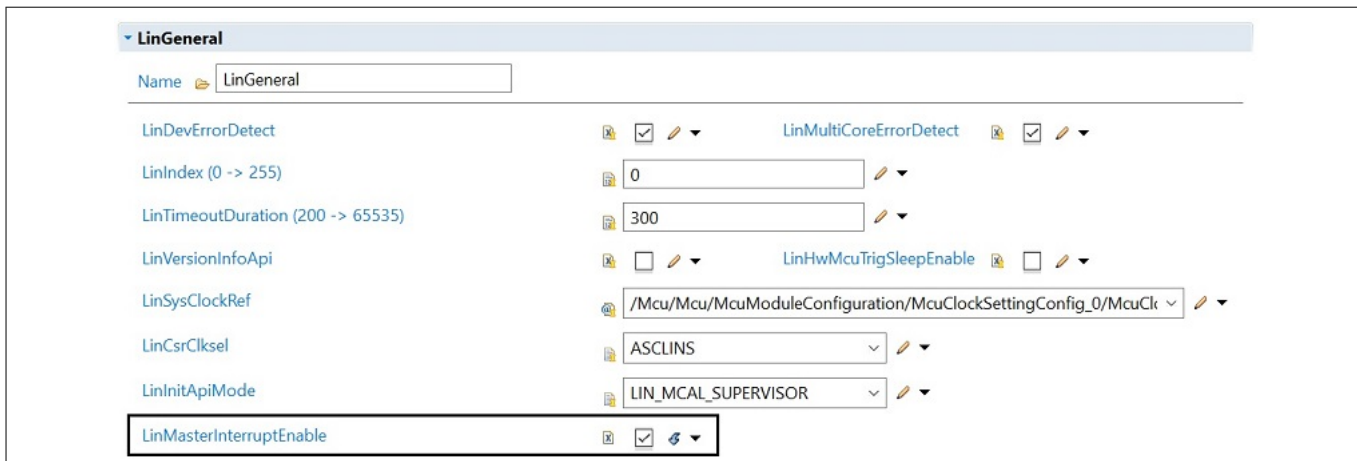
        if ((Ret2 != LIN_TX_BUSY) && (Ret2 != LIN_TX_OK) && (Ret2 != OPERATIONAL_STATE))
        {
            /* Reception Error.... */
            break;
        }
    }while(Ret2 != LIN_RX_OK );

    if (Ret2 == LIN_RX_OK)
    {
        /* LIN Header (PID 0xC1) Transmitted successfully.... */
        /* Data Received by Master.... */
        for (i=0; i < 8 ; i++)
        {
            printf("%x ",SlaveSduPtr[i]);
        }
    }
    else
    {
        /* Lin Master->Slave Response Failed. Response code: Ret2.... */
    }
}
```

LIN master channels interrupt enable

The LIN driver provides pre-compile configuration parameter `LinMasterInterruptEnable`. By disabling `LinMasterInterruptEnable` parameter, LIN master channels can be configured to work in polling mode by using the `Lin_17_AscLin_GetStatus()` API.

1 Lin_17_AscLin driver



The screenshot shows the 'LinGeneral' configuration window. The 'Name' field is set to 'LinGeneral'. The 'LinMasterInterruptEnable' parameter is highlighted at the bottom, showing a checked checkbox and a lock icon. Other visible parameters include 'LinDevErrorDetect' (checked), 'LinIndex' (0), 'LinTimeoutDuration' (300), 'LinVersionInfoApi' (unchecked), 'LinSysClockRef' (set to '/Mcu/Mcu/McuModuleConfiguration/McuClockSettingConfig_0/McuClk'), 'LinCsrClkSel' (ASCLINS), and 'LinInitApiMode' (LIN_MCAL_SUPERVISOR).

Figure 10 LinMasterInterruptEnable configuration parameter

1.1.5 Key architectural considerations

1.1.5.1 ASCLIN hardware: used for LIN feature

The LIN driver uses only the LIN features of the ASCLIN hardware. The LIN driver implements LIN protocol 2.1 and ISO-17987 version as per AUTOSAR. Since ASCLIN hardware supports multiple features like LIN, SPI, ASC, the driver uses only LIN features. The driver supports master and slave mode as per AUTOSAR specification.

Master mode has following types of communication:

Master Response: Master node is the publisher of the data frame, after header transmission response frame is transmitted.

Slave Response: One of the slave node as defined in LDF (LIN Description File) publishes the response and master node is one of the subscriber for the data frame.

Slave to Slave communication: Master node transmits a header and ignores the response in reply of the header as it is not one of the subscribers.

1.1.5.2 Modes of Operation - Tx FIFO and Rx FIFO Modes

Tx FIFO and Rx FIFO operate in the combined mode. In LIN protocol the length of the data varies (maximum 8 bytes). Combined mode is used, in order to raise interrupt at end of each frame. This mode also defines required threshold level for each frame transfer. Three interrupt generation modes are provided by ASCLIN Tx FIFO, single move mode, batch move mode and combined move mode with 8 bit data width. For the protocol combined mode is used with frame length.

1 Lin_17_AscLin driver

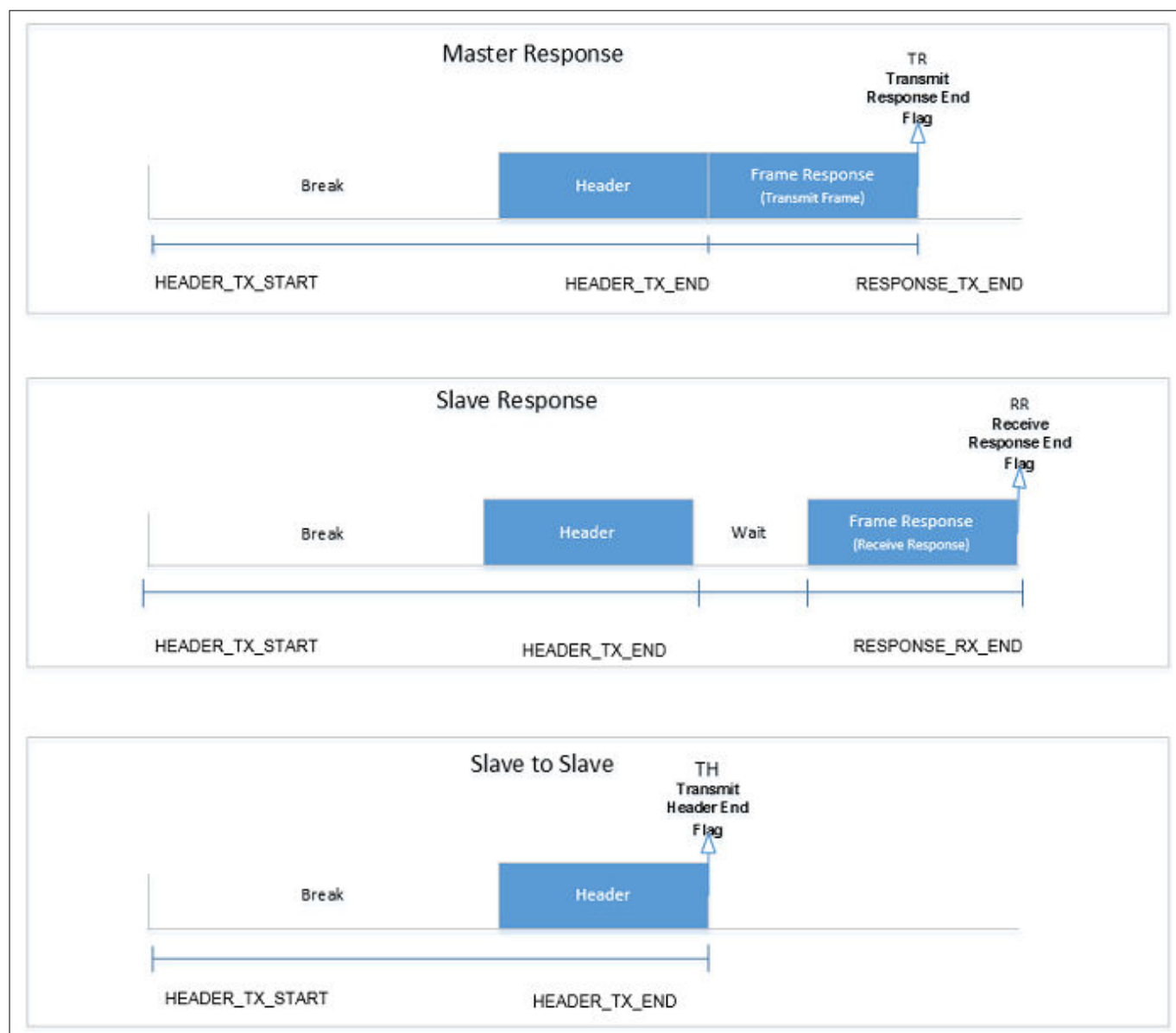


Figure 11 LIN interrupt generation for different types of communication

1.1.5.3 Addition of LinMasterInterruptEnable configuration parameter

LinMasterInterruptEnable configuration parameter is added in the LinGeneral container. By disabling LinMasterInterruptEnable parameter, LIN master channels can be configured to work in polling mode by using the Lin_17_AscLin_GetStatus() API. This parameter is only applicable to master channels.

1 Lin_17_AscLin driver

1.2 Assumptions of Use (AoU)

There are no AoU for the LIN driver.

1 Lin_17_AscLin driver

1.3 Reference information

1.3.1 Configuration interfaces

Supported configuration variant: Post-Build

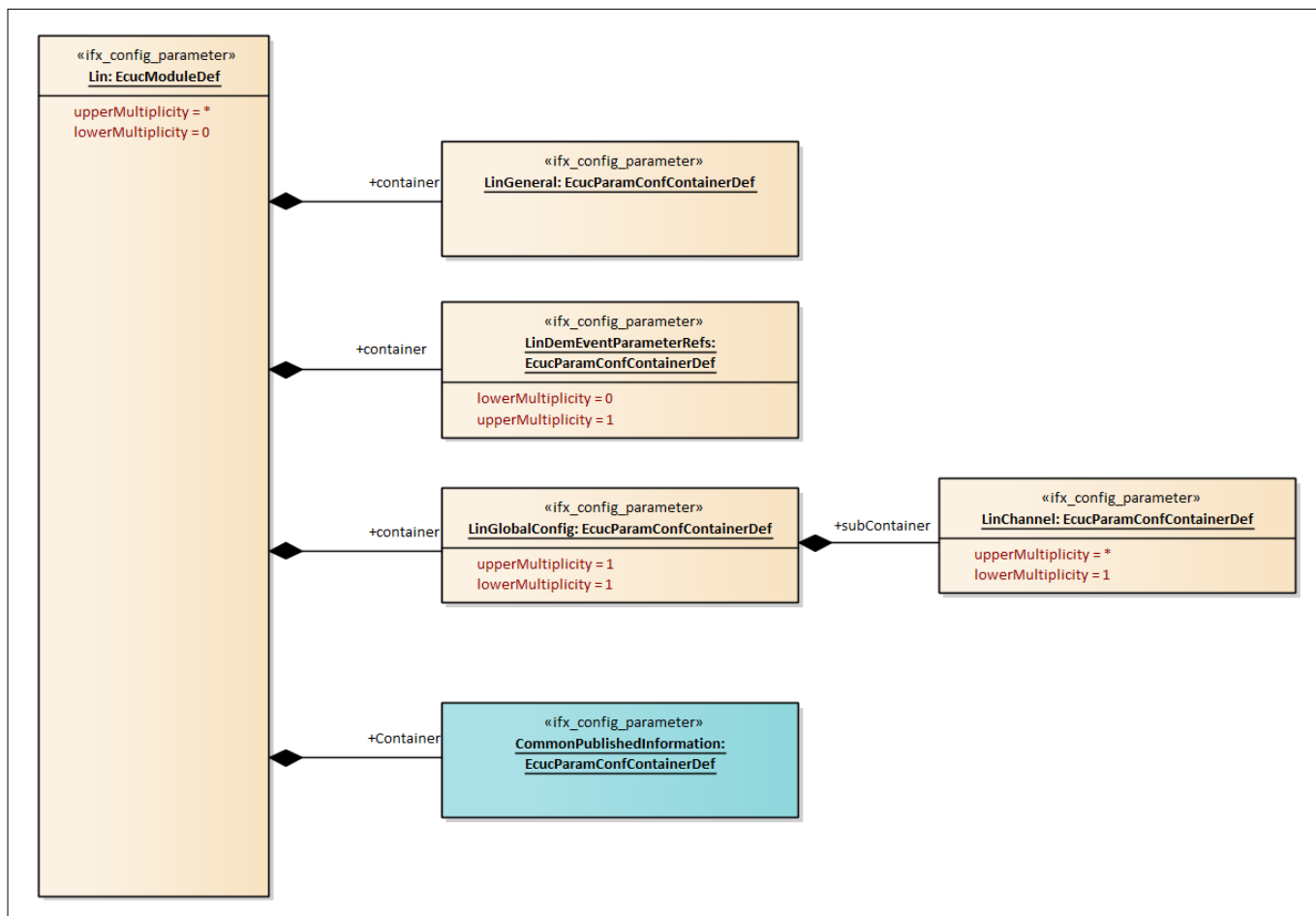


Figure 12 Container hierarchy along with their configuration parameters

1.3.1.1 Container: CommonPublishedInformation

This section describes the parameters published by the LIN driver module.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.1.1 ArMajorVersion

Table 4 Specification for ArMajorVersion

Name	ArMajorVersion		
Description	Major version number of AUTOSAR specification on which the appropriate implementation is based on.		
Multiplicity	1..1	Type	EcucIntegerParamDef

(table continues...)

1 Lin_17_AscLin driver
Table 4 (continued) Specification for ArMajorVersion

Range	0 - 255		
Default value	4		
Post-build variant value	FALSE	Post-build variant multiplicity	-
Value configuration class	Pre-Compile	Multiplicity configuration class	-
Origin	IFX	Scope	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.2 ArMinorVersion
Table 5 Specification for ArMinorVersion

Name	ArMinorVersion		
Description	Minor version number of AUTOSAR specification on which the appropriate implementation is based on.		
Multiplicity	1..1	Type	EcucIntegerParamDef
Range	0 - 255		
Default value	As per selected Autosar version		
Post-build variant value	FALSE	Post-build variant multiplicity	-
Value configuration class	Pre-Compile	Multiplicity configuration class	-
Origin	IFX	Scope	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.3 ArPatchVersion
Table 6 Specification for ArPatchVersion

Name	ArPatchVersion		
Description	Patch level version number of AUTOSAR specification on which the appropriate implementation is based on.		
Multiplicity	1..1	Type	EcucIntegerParamDef
Range	0 - 255		
Default value	As per selected Autosar version		

(table continues...)

1 Lin_17_AscLin driver
Table 6 (continued) Specification for ArPatchVersion

Post-build variant value	FALSE	Post-build variant multiplicity	-
Value configuration class	Pre-Compile	Multiplicity configuration class	-
Origin	IFX	Scope	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.4 ModuleId
Table 7 Specification for ModuleId

Name	ModuleId		
Description	Module ID of this module from the Module list.		
Multiplicity	1..1	Type	EcucIntegerParamDef
Range	0 - 65535		
Default value	82		
Post-build variant value	FALSE	Post-build variant multiplicity	-
Value configuration class	Pre-Compile	Multiplicity configuration class	-
Origin	IFX	Scope	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.5 Release
Table 8 Specification for Release

Name	Release		
Description	This parameter indicates the TC3xx device derivative used for the implementation.		
Multiplicity	1..1	Type	EcucStringParamDef
Range	String		
Default value	As per hardware derivative		
Post-build variant value	FALSE	Post-build variant multiplicity	-

(table continues...)

1 Lin_17_AscLin driver
Table 8 (continued) Specification for Release

Value configuration class	Published-Information	Multiplicity configuration class	-
Origin	IFX	Scope	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.6 SwMajorVersion
Table 9 Specification for SwMajorVersion

Name	SwMajorVersion		
Description	Major version number of the vendor specific implementation of the module. The numbering is vendor specific.		
Multiplicity	1..1	Type	EcucIntegerParamDef
Range	0 - 255		
Default value	As per Driver		
Post-build variant value	FALSE	Post-build variant multiplicity	-
Value configuration class	Pre-Compile	Multiplicity configuration class	-
Origin	IFX	Scope	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.7 SwMinorVersion
Table 10 Specification for SwMinorVersion

Name	SwMinorVersion		
Description	Minor version number of the vendor specific implementation of the module. The numbering is vendor specific.		
Multiplicity	1..1	Type	EcucIntegerParamDef
Range	0 - 255		
Default value	As per Driver		
Post-build variant value	FALSE	Post-build variant multiplicity	-
Value configuration class	Pre-Compile	Multiplicity configuration class	-

(table continues...)

1 Lin_17_AscLin driver
Table 10 (continued) Specification for SwMinorVersion

Origin	IFX	Scope	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.8 SwPatchVersion
Table 11 Specification for SwPatchVersion

Name	SwPatchVersion		
Description	Patch level version number of the vendor specific implementation of the module. The numbering is vendor specific.		
Multiplicity	1..1	Type	EcucIntegerParamDef
Range	0 - 255		
Default value	As per Driver		
Post-build variant value	FALSE	Post-build variant multiplicity	-
Value configuration class	Pre-Compile	Multiplicity configuration class	-
Origin	IFX	Scope	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.9 VendorApiInfix
Table 12 Specification for VendorApiInfix

Name	VendorApiInfix		
Description	This parameter is used to specify the vendor specific name.		
Multiplicity	1..1	Type	EcucStringParamDef
Range	String		
Default value	AscLin		
Post-build variant value	FALSE	Post-build variant multiplicity	-
Value configuration class	Pre-Compile	Multiplicity configuration class	-
Origin	IFX	Scope	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1 Lin_17_AscLin driver
1.3.1.1.10 VendorId
Table 13 Specification for VendorId

Name	VendorId		
Description	Vendor ID of the dedicated implementation of this module according to the AUTOSAR vendor list.		
Multiplicity	1..1	Type	EcucIntegerParamDef
Range	0 - 65535		
Default value	17		
Post-build variant value	FALSE	Post-build variant multiplicity	-
Value configuration class	Pre-Compile	Multiplicity configuration class	-
Origin	IFX	Scope	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.2 Container: Lin

Configuration of the LIN module.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.3 Container: LinChannel

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

Post-Build Variant Multiplicity: FALSE

Multiplicity Configuration Class: Post-Build

1.3.1.3.1 LinAutoCalcBaudParams
Table 14 Specification for LinAutoCalcBaudParams

Name	LinAutoCalcBaudParams		
Description	This parameter enables or disables the automatic baud rate parameter calculation by the configuration tool. <i>Note: The optional features are disabled by default to minimize the executable code size. Since the calculation of the baud rate is done automatically.</i>		
Multiplicity	1..1	Type	EcucBooleanParamDef
Range	TRUE FALSE		

(table continues...)

1 Lin_17_AscLin driver
Table 14 (continued) Specification for LinAutoCalcBaudParams

Default value	FALSE		
Post-build variant value	FALSE	Post-build variant multiplicity	-
Value configuration class	Pre-Compile	Multiplicity configuration class	-
Origin	IFX	Scope	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.3.2 LinChanAssignedHw
Table 15 Specification for LinChanAssignedHw

Name	LinChanAssignedHw		
Description	This parameter defines which ASCLIN module is selected by LIN channel. <i>Note: Minimum Kernel ID is selected as the default value.</i>		
Multiplicity	1..1	Type	EcucEnumerationParamDef
Range	ASCLIN0: Signifies ASCLIN hardware kernel 0. ASCLINx: This parameter signifies kernel name, where x is varies from 1 to maximum number of units as per device variant. For example ASCLIN1, ASCLIN2,, ASCLINx, where x depends on device variant.		
Default value	ASCLIN0		
Post-build variant value	FALSE	Post-build variant multiplicity	-
Value configuration class	Pre-Compile	Multiplicity configuration class	-
Origin	IFX	Scope	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.3.3 LinChannelBaudDenominator
Table 16 Specification for LinChannelBaudDenominator

Name	LinChannelBaudDenominator
Description	The parameter value is used to configure the BRG register DENOMINATOR field. The parameter is editable if LinAutoCalcBaudParams parameter is false. <i>Note: Minimum Denominator value is selected as the default value.</i>

(table continues...)

1 Lin_17_AscLin driver
Table 16 (continued) Specification for LinChannelBaudDenominator

Multiplicity	1..1	Type	EcucIntegerParamDef
Range	1 - 4095		
Default value	1		
Post-build variant value	TRUE	Post-build variant multiplicity	-
Value configuration class	Post-Build	Multiplicity configuration class	-
Origin	IFX	Scope	LOCAL
Dependency	LinAutoCalcBaudParams, LinChannelBaudNumerator		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.3.4 LinChannelBaudNumerator
Table 17 Specification for LinChannelBaudNumerator

Name	LinChannelBaudNumerator		
Description	The parameter value is used to configure the BRG register NUMERATOR field. The parameter is editable if LinAutoCalcBaudParams parameter is false. <i>Note: Minimum Numerator value is selected as the default value.</i>		
Multiplicity	1..1	Type	EcucIntegerParamDef
Range	1 - 4095		
Default value	1		
Post-build variant value	TRUE	Post-build variant multiplicity	-
Value configuration class	Post-Build	Multiplicity configuration class	-
Origin	IFX	Scope	LOCAL
Dependency	LinAutoCalcBaudParams		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.3.5 LinChannelBaudPreScalar
Table 18 Specification for LinChannelBaudPreScalar

Name	LinChannelBaudPreScalar		
Description	This parameter value is used to configure the BITCON register PRESCALAR field. The parameter is editable if LinAutoCalcBaudParams parameter is false. <i>Note: Minimum prescalar value is selected as the default value.</i>		

(table continues...)

1 Lin_17_AscLin driver
Table 18 (continued) Specification for LinChannelBaudPreScalar

Multiplicity	1..1	Type	EcucIntegerParamDef
Range	0 - 4095		
Default value	0		
Post-build variant value	TRUE	Post-build variant multiplicity	-
Value configuration class	Post-Build	Multiplicity configuration class	-
Origin	IFX	Scope	LOCAL
Dependency	LinAutoCalcBaudParams		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.3.6 LinChannelBaudRate
Table 19 Specification for LinChannelBaudRate

Name	LinChannelBaudRate		
Description	Specifies the baud rate of the LIN channel. The parameter is editable if LinAutoCalcBaudParams parameter is true. <i>Note: Minimum baudrate value is selected as the default value.</i>		
Multiplicity	1..1	Type	EcucIntegerParamDef
Range	1000 - 20000		
Default value	1000		
Post-build variant value	TRUE	Post-build variant multiplicity	-
Value configuration class	Post-Build	Multiplicity configuration class	-
Origin	AUTOSAR_ECUC	Scope	LOCAL
Dependency	LinAutoCalcBaudParams		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.3.7 LinChannelEcuMWakeupSource
Table 20 Specification for LinChannelEcuMWakeupSource

Name	LinChannelEcuMWakeupSource
-------------	----------------------------

(table continues...)

1 Lin_17_AscLin driver
Table 20 (continued) Specification for LinChannelEcuMWakeupSource

Description	This parameter contains a reference to the Wakeup Source for this controller as defined in the ECU State Manager. <i>Note: Since the name of the dependent container is user configurable, the default value is kept as NULL.</i>		
Multiplicity	0..1	Type	EcucSymbolicNameReferenceDef
Range	Reference to Node: EcuMWakeupSource		
Default value	NULL		
Post-build variant value	FALSE	Post-build variant multiplicity	FALSE
Value configuration class	Pre-Compile	Multiplicity configuration class	Pre-Compile
Origin	AUTOSAR_ECUC	Scope	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.3.8 LinChannelEcucPartitionRef
Table 21 Specification for LinChannelEcucPartitionRef

Name	LinChannelEcucPartitionRef		
Description	Maps one single LIN channel to zero or one ECUC partitions. The ECUC partition referenced is a subset of the ECUC partitions where the LIN driver is mapped to. <i>Note: Parameter support is added only for AUTOSAR schema compliance. This parameter is not used in code generation logic, hence this parameter is made editable false.</i>		
Multiplicity	0..1	Type	EcucReferenceDef
Range	Reference to Node:		
Default value	NULL		
Post-build variant value	TRUE	Post-build variant multiplicity	TRUE
Value configuration class	Post-Build	Multiplicity configuration class	Post-Build
Origin	AUTOSAR_ECUC	Scope	ECU
Dependency	-		
Autosar Version	Applicable for Autosar version 4.4.0.		

1 Lin_17_AscLin driver
1.3.1.3.9 LinChannelId
Table 22 Specification for LinChannelId

Name	LinChannelId		
Description	Parameter defines the numeric ID of the channel, which is the logical ID for the channel. It must be in consecutive sequence starting from zero for each LIN kernel (ASCLIN HwUnit). A symbolic name is generated for each channel. <i>Note: Minimum channel ID is selected as the default value.</i>		
Multiplicity	1..1	Type	EcucIntegerParamDef
Range	0 - 65535		
Default value	0		
Post-build variant value	FALSE	Post-build variant multiplicity	-
Value configuration class	Pre-Compile	Multiplicity configuration class	-
Origin	AUTOSAR_ECUC	Scope	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.3.10 LinChannelWakeupSupport
Table 23 Specification for LinChannelWakeupSupport

Name	LinChannelWakeupSupport		
Description	Specifies if the LIN hardware channel supports wake up functionality. <i>Note: The optional features are disabled by default to minimize the executable code size.</i>		
Multiplicity	1..1	Type	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	Post-build variant multiplicity	-
Value configuration class	Pre-Compile	Multiplicity configuration class	-
Origin	AUTOSAR_ECUC	Scope	LOCAL
Dependency	LinMasterInterruptEnable		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1 Lin_17_AscLin driver
1.3.1.3.11 LinClockRef
Table 24 Specification for LinClockRef

Name	LinClockRef		
Description	Reference to the LIN clock source configuration, which is set in the MCU driver configuration. <i>Note: Configuration parameter LinClockRef is not editable since LinSysClockRef is the clock configuration common for the ASCLIN module of each channel.</i>		
Multiplicity	1..1	Type	EcucReferenceDef
Range	Reference to Node:		
Default value	NULL		
Post-build variant value	TRUE	Post-build variant multiplicity	-
Value configuration class	Post-Build	Multiplicity configuration class	-
Origin	AUTOSAR_ECUC	Scope	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.3.12 LinInterByteSpace
Table 25 Specification for LinInterByteSpace

Name	LinInterByteSpace		
Description	This parameter is used to configure LinInterByteSpace delay in bit times. In LIN mode, this is the pause inserted between transmission of bytes. LinInterByteSpace delay also applies to the pause between the header and the response (response space). Though HW supports 0 to 7 as range, in tresos range is selected between 0 to 4 since the maximum tolerance as per the LIN protocol is 40% of frame deviation is allowed. Based on this protocol definition maximum value is limited to 4. <i>Note: Minimum Inter byte space value is selected as the default value.</i>		
Multiplicity	1..1	Type	EcucIntegerParamDef
Range	0 - 4		
Default value	0		
Post-build variant value	TRUE	Post-build variant multiplicity	-
Value configuration class	Post-Build	Multiplicity configuration class	-
Origin	IFX	Scope	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1 Lin_17_AscLin driver
1.3.1.3.13 LinNodeType
Table 26 Specification for LinNodeType

Name	LinNodeType		
Description	Specifies the LIN node type of the channel		
Multiplicity	1..1	Type	EcucEnumerationParamDef
Range	MASTER: SLAVE:		
Default value	MASTER		
Post-build variant value	FALSE	Post-build variant multiplicity	-
Value configuration class	Pre-Compile	Multiplicity configuration class	-
Origin	AUTOSAR_ECUC	Scope	LOCAL
Dependency	LinMasterInterruptEnable		
Autosar Version	Applicable for Autosar version 4.4.0.		

1.3.1.3.14 LinRxAlternateInputSignal
Table 27 Specification for LinRxAlternateInputSignal

Name	LinRxAlternateInputSignal		
Description	This parameter selects the alternate input for the Rx signal for the given LIN channel. <i>Note: The first available data line for configured ASCLIN HW unit is selected as default value.</i>		
Multiplicity	1..1	Type	EcucEnumerationParamDef
Range	SELECT_A_PORT14_PIN1: Rx data pin option SELECT_x_PORTy_PINz: This parameter varies in availability as per configured LIN kernel, and device variant, where x signifies dataline , y signifies port number and z signifies pin number. For example SELECT_A_PORT14_PIN1.		
Default value	SELECT_A_PORT14_PIN1		
Post-build variant value	TRUE	Post-build variant multiplicity	-
Value configuration class	Post-Build	Multiplicity configuration class	-
Origin	IFX	Scope	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1 Lin_17_AscLin driver
1.3.1.4 Container: LinDemEventParameterRefs

Container for the references to DemEventParameter elements which shall be invoked using the Mcal_Wrapper API when the corresponding error occurs. The EventId is taken from the referenced DemEventParameter's DemEventId value. The standardized errors are provided in the container and can be extended by vendor specific error references.

Post-Build Variant Multiplicity: FALSE

Multiplicity Configuration Class: Pre-Compile

1.3.1.4.1 LIN_E_TIMEOUT
Table 28 Specification for LIN_E_TIMEOUT

Name	LIN_E_TIMEOUT		
Description	<p>Reference to the DemEventParameter which shall be issued when the error "Timeout caused by hardware error" has occurred.</p> <p>Since the name of the dependent container is user configurable, the default value is kept as NULL.</p> <p>When the reference is not configured there will be no Production Error raised.</p>		
Multiplicity	0..1	Type	EcucSymbolicNameReferenceDef
Range	Reference to Node: DemEventParameter		
Default value	NULL		
Post-build variant value	FALSE	Post-build variant multiplicity	FALSE
Value configuration class	Pre-Compile	Multiplicity configuration class	Pre-Compile
Origin	AUTOSAR_ECUC	Scope	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.5 Container: LinGeneral

This container contains the parameters related to each LIN Driver Unit.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.5.1 LinCsrClksel
Table 29 Specification for LinCsrClksel

Name	LinCsrClksel
-------------	--------------

(table continues...)

1 Lin_17_AscLin driver
Table 29 (continued) Specification for LinCsrClksel

Description	This parameter selects BaudRate logic clock for the LIN driver. <i>Note: Default value set with fast mode.</i>		
Multiplicity	1..1	Type	EcucEnumerationParamDef
Range	ASCLINF: ASCLIN fast clock mode ASCLINS: ASCLIN slow clock mode		
Default value	ASCLINF		
Post-build variant value	FALSE	Post-build variant multiplicity	-
Value configuration class	Pre-Compile	Multiplicity configuration class	-
Origin	IFX	Scope	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.5.2 LinDevErrorDetect
Table 30 Specification for LinDevErrorDetect

Name	LinDevErrorDetect		
Description	Switches the Default Error Tracer (Det) detection and notification ON or OFF. - TRUE: enabled (ON) - FALSE: disabled (OFF)		
Multiplicity	1..1	Type	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	Post-build variant multiplicity	-
Value configuration class	Pre-Compile	Multiplicity configuration class	-
Origin	AUTOSAR_ECUC	Scope	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1 Lin_17_AscLin driver
1.3.1.5.3 LinEcucPartitionRef
Table 31 Specification for LinEcucPartitionRef

Name	LinEcucPartitionRef		
Description	Maps the LIN driver to zero or multiple ECUC partitions to make the modules API available in this partition. The LIN driver will operate as an independent instance in each of the partitions. <i>Note: Parameter support is added only for AUTOSAR schema compliance. This parameter is not used in code generation logic, hence this parameter is made editable false.</i>		
Multiplicity	1..1	Type	EcucReferenceDef
Range	Reference to Node:		
Default value	NULL		
Post-build variant value	TRUE	Post-build variant multiplicity	TRUE
Value configuration class	Post-Build	Multiplicity configuration class	Post-Build
Origin	AUTOSAR_ECUC	Scope	ECU
Dependency	-		
Autosar Version	Applicable for Autosar version 4.4.0.		

1.3.1.5.4 LinHwMcuTrigSleepEnable
Table 32 Specification for LinHwMcuTrigSleepEnable

Name	LinHwMcuTrigSleepEnable		
Description	Enable or disable the ASCLIN module sleep upon request by setting corresponding CLC register EDIS bit, for all configured channels. The parameter is common for all ASCLIN channels. <i>Note: The optional features are disabled by default to minimize the executable code size.</i>		
Multiplicity	1..1	Type	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	Post-build variant multiplicity	-
Value configuration class	Pre-Compile	Multiplicity configuration class	-
Origin	IFX	Scope	LOCAL
Dependency	-		

(table continues...)

1 Lin_17_AscLin driver
Table 32 (continued) Specification for LinHwMcuTrigSleepEnable

Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.
------------------------	--

1.3.1.5.5 LinIndex
Table 33 Specification for LinIndex

Name	LinIndex		
Description	Specifies the InstanceId of this module instance. If only one instance is present it shall have the Id 0. <i>Note: Minimum channel index is selected as the default value.</i>		
Multiplicity	1..1	Type	EcucIntegerParamDef
Range	0 - 255		
Default value	0		
Post-build variant value	FALSE	Post-build variant multiplicity	-
Value configuration class	Pre-Compile	Multiplicity configuration class	-
Origin	AUTOSAR_ECUC	Scope	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.5.6 LinInitApiMode
Table 34 Specification for LinInitApiMode

Name	LinInitApiMode		
Description	This configuration parameter defines the mode in which the Init API will be used. This parameter is introduced to support the selection of the operation mode (supervisor/user1) during the init phase. Since LIN driver accesses the SFRs, it is more efficient to operate the LIN driver in supervisor mode. Hence, the default mode of operation is supervisor.		
Multiplicity	1..1	Type	EcucEnumerationParamDef
Range	LIN_MCAL_SUPERVISOR: Operating mode used is SUPERVISOR. LIN_MCAL_USER1: Operating mode used is USER1.		
Default value	LIN_MCAL_SUPERVISOR		
Post-build variant value	FALSE	Post-build variant multiplicity	-

(table continues...)

1 Lin_17_AscLin driver
Table 34 (continued) Specification for LinInitApiMode

Value configuration class	Pre-Compile	Multiplicity configuration class	-
Origin	IFX	Scope	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.5.7 LinMasterInterruptEnable
Table 35 Specification for LinMasterInterruptEnable

Name	LinMasterInterruptEnable		
Description	Specify LIN operation in interrupt disabled or enabled mode. This parameter is applicable for Master only. <i>Note: This parameter is enabled by default to support both Master and Slave channel configuration.</i>		
Multiplicity	1..1	Type	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	TRUE		
Post-build variant value	FALSE	Post-build variant multiplicity	-
Value configuration class	Pre-Compile	Multiplicity configuration class	-
Origin	IFX	Scope	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.5.8 LinMultiCoreErrorDetect
Table 36 Specification for LinMultiCoreErrorDetect

Name	LinMultiCoreErrorDetect		
Description	Switches the multi-core error detection and notification to ON or OFF. - TRUE: enabled (ON) - FALSE: disabled (OFF) <i>Note: If the LinMultiCoreErrorDetect parameter is set to TRUE with the LinDevErrorDetect parameter set to FALSE, an error is generated.</i>		

(table continues...)

1 Lin_17_AscLin driver
Table 36 (continued) Specification for LinMultiCoreErrorDetect

Multiplicity	1..1	Type	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	Post-build variant multiplicity	-
Value configuration class	Pre-Compile	Multiplicity configuration class	-
Origin	IFX	Scope	LOCAL
Dependency	LinDevErrorDetect		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.5.9 LinSysClockRef
Table 37 Specification for LinSysClockRef

Name	LinSysClockRef		
Description	This parameter refers to the system clock configured by MCU driver. This reference is used for baudrate computation. <i>Note: Since the name of the dependent container is user configurable, the default value is kept as NULL.</i>		
Multiplicity	1..1	Type	EcucReferenceDef
Range	Reference to Node: McuClockReferencePointConfig		
Default value	NULL		
Post-build variant value	FALSE	Post-build variant multiplicity	-
Value configuration class	Pre-Compile	Multiplicity configuration class	-
Origin	IFX	Scope	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.5.10 LinTimeoutDuration
Table 38 Specification for LinTimeoutDuration

Name	LinTimeoutDuration
-------------	--------------------

(table continues...)

1 Lin_17_AscLin driver
Table 38 (continued) Specification for LinTimeoutDuration

Description	Specifies the maximum waiting time in nanoseconds for hardware timeout errors. <i>Note: Minimum 200 nanoseconds is required to set/reset hardware bit. So, selecting 200 as minimum value for timeout duration. This is a deviation from both AUTOSAR 4.2.2 and AUTOSAR 4.4.0 since the range is 0-65535 as per AUTOSAR.</i>		
Multiplicity	1..1	Type	EcucIntegerParamDef
Range	200 - 65535		
Default value	300		
Post-build variant value	FALSE	Post-build variant multiplicity	-
Value configuration class	Pre-Compile	Multiplicity configuration class	-
Origin	AUTOSAR_ECUC	Scope	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.5.11 LinVersionInfoApi
Table 39 Specification for LinVersionInfoApi

Name	LinVersionInfoApi		
Description	Switches the Lin_GetVersionInfo function ON or OFF. <i>Note: The optional features are disabled by default to minimize the executable code size.</i>		
Multiplicity	1..1	Type	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	Post-build variant multiplicity	-
Value configuration class	Pre-Compile	Multiplicity configuration class	-
Origin	AUTOSAR_ECUC	Scope	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.6 Container: LinGlobalConfig

This container contains the global configuration parameter of the Lin driver.

Post-Build Variant Multiplicity: -

1 Lin_17_AscLin driver

Multiplicity Configuration Class: -

1.3.2 Functions - Type definitions

This chapter lists out all the data types of the driver.

1.3.2.1 Lin_SlaveErrorType

Table 40 Specification for Lin_SlaveErrorType

Syntax	Lin_SlaveErrorType	
Type	Enumeration	
File	Lin_GeneralTypes.h	
Range	0 - LIN_ERR_HEADER	Error in header
	1 - LIN_ERR_RESP_STOPBIT	Framing error in response
	2 - LIN_ERR_RESP_CHKSUM	Checksum error
	3 - LIN_ERR_RESP_DATABIT	Monitoring error of the transmitted data bit in response.
	4 - LIN_ERR_NO_RESP	No response.
	5 - LIN_ERR_INC_RESP	Incomplete response.
Description	This type represents the slave error types that are detected during header reception and response transmission/reception.	
Source	AUTOSAR	
Autosar Version	Applicable for Autosar version 4.4.0.	

1.3.2.2 Lin_17_AscLin_ConfigType

Table 41 Specification for Lin_17_AscLin_ConfigType

Syntax	Lin_17_AscLin_ConfigType	
Type	Structure	
File	Lin_17_AscLin.h	
Range	- []	The elements of the data structure are specific to the micro-controller.
Description	Structure holds the configuration of multiple LIN channels configured core wise used for initialization of the LIN driver.	
Source	AUTOSAR	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1 Lin_17_AscLin driver
1.3.2.3 Lin_FramePidType
Table 42 Specification for Lin_FramePidType

Syntax	Lin_FramePidType	
Type	uint8	
File	Lin_GeneralTypes.h	
Range	0..0xFE	The LIN identifier (0...0x3F) together with its two parity bits.
Description	Represents all valid protected identifier used by Lin_SendFrame().	
Source	AUTOSAR	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.4 Lin_FrameCsModelType
Table 43 Specification for Lin_FrameCsModelType

Syntax	Lin_FrameCsModelType	
Type	Enumeration	
File	Lin_GeneralTypes.h	
Range	0 - LIN_ENHANCED_CS	Enhanced checksum model
	1 - LIN_CLASSIC_CS	Classic checksum model
Description	This type is used to specify the checksum model to be used for the LIN frame.	
Source	AUTOSAR	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.5 Lin_FrameResponseType
Table 44 Specification for Lin_FrameResponseType

Syntax	Lin_FrameResponseType	
Type	Enumeration	
File	Lin_GeneralTypes.h	
Range	0 - LIN_MASTER_RESPONSE	Response is generated from this node.
	1 - LIN_SLAVE_RESPONSE	Response is generated from another node and is relevant for this node.
	2 - LIN_SLAVE_TO_SLAVE	Response is generated from another node and is irrelevant for this node.

(table continues...)

1 Lin_17_AscLin driver
Table 44 (continued) Specification for Lin_FrameResponseType

Description	<p>This type is used to specify whether the frame processor is required to transmit the response part of the LIN frame.</p> <p>As per Autosar version 440 the ranges are:</p> <p>0 - LIN_FRAMERESPONSE_TX</p> <p>1 - LIN_FRAMERESPONSE_RX</p> <p>2 - LIN_FRAMERESPONSE_IGNORE</p> <p>and the mapping with the ranges for master nodes of Autosar version 422 are:</p> <p>LIN_FRAMERESPONSE_TX <=> LIN_MASTER_RESPONSE</p> <p>LIN_FRAMERESPONSE_RX <=> LIN_SLAVE_RESPONSE</p> <p>LIN_FRAMERESPONSE_IGNORE <=> LIN_SLAVE_TO_SLAVE</p>
Source	AUTOSAR
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.2.6 Lin_FrameDlType
Table 45 Specification for Lin_FrameDlType

Syntax	Lin_FrameDlType	
Type	uint8	
File	Lin_GeneralTypes.h	
Range	1...8	Data length of a LIN Frame
Description	This type is used to specify the number of SDU data bytes to copy.	
Source	AUTOSAR	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.7 Lin_PduType
Table 46 Specification for Lin_PduType

Syntax	Lin_PduType	
Type	Structure	
File	Lin_GeneralTypes.h	
Range	Lin_FramePidType Pid	Describes Pid number for the current LIN frame for which header is sent.
	Lin_FrameCsModelType Cs	Describes checksum used for the LIN frame
	Lin_FrameResponseType Drc	Describes type of communication required for the LIN frame
	Lin_FrameDlType Dl	Describes data length for the current LIN frame

(table continues...)

1 Lin_17_AscLin driver
Table 46 (continued) Specification for Lin_PduType

	uint8 * SduPtr	Pointer to get the data from upper layer
Description	This type is used to provide PID, checksum model, data length and SDU pointer from the LIN interface to the LIN driver.	
Source	AUTOSAR	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.8 Lin_StatusType
Table 47 Specification for Lin_StatusType

Syntax	Lin_StatusType	
Type	Enumeration	
File	Lin_GeneralTypes.h	
Range	0 - LIN_NOT_OK	LIN frame operation return value. Development or production error occurred
	1 - LIN_TX_OK	LIN frame operation return value. Successful transmission.
	2 - LIN_TX_BUSY	LIN frame operation return value. Ongoing transmission (Header or Response).
	3 - LIN_TX_HEADER_ERROR	LIN frame operation return value. Erroneous header transmission such as: - Mismatch between sent and read back data - Identifier parity error or - Physical bus error
	4 - LIN_TX_ERROR	LIN frame operation return value. Erroneous response transmission such as: - Mismatch between sent and read back data - Physical bus error
	5 - LIN_RX_OK	LIN frame operation return value. Reception of correct response.
	6 - 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.
	7 - LIN_RX_ERROR	LIN frame operation return value. Erroneous response reception such as: - Framing error - Overrun error - Checksum error or - Short response
	8 - LIN_RX_NO_RESPONSE	LIN frame operation return value. No response byte has been received so far.

(table continues...)

1 Lin_17_AscLin driver
Table 47 (continued) Specification for Lin_StatusType

	9 - 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)
	10 - LIN_CH_SLEEP	LIN channel state return value. Sleep state operation; in this state wakeup detection from slave nodes is enabled.
Description	LIN operation states for a LIN channel or frame, as returned by the API service Lin_GetStatus().	
Source	AUTOSAR	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3 Functions - APIs

This section lists all the APIs of the LIN driver.

1.3.3.1 Lin_17_AscLin_CheckWakeup
Table 48 Specification for Lin_17_AscLin_CheckWakeup API

Syntax	Std_ReturnType Lin_17_AscLin_CheckWakeup (const uint8 Channel)	
Service ID	0x0a	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant	
Parameters (in)	Channel	LIN channel to be addressed
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Std_ReturnType	E_OK: No error has occurred during execution of the API. E_NOT_OK: - During execution of the API when development or production error is occurred. - When channel wakeup support is OFF.

(table continues...)

1 Lin_17_AscLin driver
Table 48 (continued) Specification for Lin_17_AscLin_CheckWakeup API

Description	This function checks if a wakeup has occurred on the addressed LIN channel. If wakeup (LOW) signal is detected and LinChannelWakeupSupport is ON for the LIN channel then the API calls EcuM_SetWakeupEvent and LinIf_WakeupConfirmation.
Source	AUTOSAR
Error handling	LIN_17_ASCLIN_E_INVALID_CHANNEL, LIN_17_ASCLIN_E_CORE_CHANNEL_MISMATCH, LIN_17_ASCLIN_E_UNINIT
Configuration dependencies	-
User hints	-
SFR accessed	ASCLIN_IOCRR(r) <i>Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.</i>
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.2 Lin_17_AscLin_GetStatus
Table 49 Specification for Lin_17_AscLin_GetStatus API

Syntax	<pre> Lin_StatusType Lin_17_AscLin_GetStatus (const uint8 Channel, uint8 ** const Lin_SduPtr) </pre>	
Service ID	0x08	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant	
Parameters (in)	Channel	LIN channel to be checked
Parameters (out)	Lin_SduPtr	Pointer to pointer to a shadow buffer or memory mapped LIN Hardware receive buffer where the current SDU is stored.
Parameters (in - out)	-	-

(table continues...)

1 Lin_17_AscLin driver
Table 49 (continued) Specification for Lin_17_AscLin_GetStatus API

Return	Lin_StatusType	<p>LIN_NOT_OK: Development or production error occurred.</p> <p>LIN_TX_OK: Successful transmission.</p> <p>LIN_TX_BUSY: Ongoing transmission (Header or Response).</p> <p>LIN_TX_HEADER_ERROR: Erroneous header transmission such as: - collision error, TX FIFO over flow error, or parity error.</p> <p>LIN_TX_ERROR: Erroneous response transmission such as: - collision error, TX FIFO over flow error.</p> <p>LIN_RX_OK: Reception of correct response.</p> <p>LIN_RX_BUSY: Ongoing reception: at least one response byte has been received, but the checksum byte has not been received.</p> <p>LIN_RX_ERROR: Erroneous response reception such as: - LIN frame error, frame checksum error, RX FIFO overflow or under flow error.</p> <p>LIN_RX_NO_RESPONSE: No response byte has been received so far.</p> <p>LIN_OPERATIONAL: For AS 422- Normal operation; the related LIN channel is just initialized or woken up from the LIN_CH_SLEEP and no data has been sent.</p> <p>For AS 440- Normal operation; the related LIN channel is woken up from the LIN_CH_SLEEP and no data has been sent.</p> <p>LIN_CH_SLEEP: Sleep state operation; in this state wake-up detection from slave nodes is enabled.</p>
Description	<p>Gets the status of the LIN channel. Further the API is also used to get the received data from the driver by upper layer using SDU shadow pointer.</p> <p>If Lin_SduPtr parameter is NULL pointer, DET LIN_17_ASCLIN_E_PARAM_POINTER will be triggered. Irrespective of receiving response from slave, memory needs to be allocated by upper layer and passed as Lin_SduPtr.</p> <p><i>Note: This service is only applicable if the LIN driver is configured with one or more LIN master channel.</i></p>	
Source	AUTOSAR	
Error handling	LIN_17_ASCLIN_E_CORE_CHANNEL_MISMATCH, LIN_17_ASCLIN_E_PARAM_POINTER, LIN_17_ASCLIN_E_INVALID_CHANNEL, LIN_17_ASCLIN_E_UNINIT	
Configuration dependencies	LinNodeType	
User hints	-	
SFR accessed	<p>ASCLIN_FLAGS(r), ASCLIN_FLAGSCLEAR(w), ASCLIN_FLAGSENABLE(rw), ASCLIN_RXDATA(r), ASCLIN_RXFIFOCON(rw), ASCLIN_TXFIFOCON(w)</p> <p><i>Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.</i></p>	

(table continues...)

1 Lin_17_AscLin driver
Table 49 (continued) Specification for Lin_17_AscLin_GetStatus API

Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.
------------------------	--

1.3.3.3 Lin_17_AscLin_GoToSleep
Table 50 Specification for Lin_17_AscLin_GoToSleep API

Syntax	Std_ReturnType Lin_17_AscLin_GoToSleep (const uint8 Channel)	
Service ID	0x06	
Sync/Async	Asynchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant	
Parameters (in)	Channel	LIN channel to be addressed
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Std_ReturnType	E_OK: Sleep command has been accepted. E_NOT_OK: Sleep command has not been accepted, development or production error occurred.
Description	<p>The service instructs the driver to transmit a go-to-sleep-command on the addressed LIN channel.</p> <p>If an ongoing frame transmission is still in progress then transmit and receive FIFO will be flushed and sleep command will be transmitted over LIN channel.</p> <p><i>Note: This service is only applicable if the LIN driver is configured with one or more LIN master channel.</i></p>	
Source	AUTOSAR	
Error handling	LIN_17_ASCLIN_E_INVALID_CHANNEL, LIN_17_ASCLIN_E_CORE_CHANNEL_MISMATCH, LIN_17_ASCLIN_E_UNINIT, LIN_17_ASCLIN_E_STATE_TRANSITION	
Configuration dependencies	LinNodeType	
User hints	-	
SFR accessed	ASCLIN_DATCON(w), ASCLIN_FLAGSCLEAR(w), ASCLIN_FLAGSENABLE(w), ASCLIN_FLAGSSET(w), ASCLIN_RXFIFOCON(w), ASCLIN_TXDATA(w), ASCLIN_TXFIFOCON(w) <i>Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.</i>	

(table continues...)

1 Lin_17_AscLin driver
Table 50 (continued) Specification for Lin_17_AscLin_GoToSleep API

Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.
------------------------	--

1.3.3.4 Lin_17_AscLin_GoToSleepInternal
Table 51 Specification for Lin_17_AscLin_GoToSleepInternal API

Syntax	<pre>Std_ReturnType Lin_17_AscLin_GoToSleepInternal (const uint8 Channel)</pre>	
Service ID	0x09	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant	
Parameters (in)	Channel	LIN channel to be addressed
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Std_ReturnType	E_OK: Command has been accepted. E_NOT_OK: Command has not been accepted, development or production error occurred.
Description	If an ongoing frame transmission is still in progress, transmit and receive FIFO will be flushed and channel is set to LIN_CH_SLEEP state. Hardware is configured to wakeup when falling edge is detected on the Rx pin if wakeup is enabled.	
Source	AUTOSAR	
Error handling	LIN_17_ASCLIN_E_UNINIT, LIN_17_ASCLIN_E_INVALID_CHANNEL, LIN_17_ASCLIN_E_CORE_CHANNEL_MISMATCH	
Configuration dependencies	-	
User hints	-	
SFR accessed	ASCLIN_FLAGSCLEAR(w), ASCLIN_FLAGSENABLE(w), ASCLIN_RXFIFOCON(w), ASCLIN_TXFIFOCON(w) <i>Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.</i>	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1 Lin_17_AscLin driver
1.3.3.5 Lin_17_AscLin_Init
Table 52 Specification for Lin_17_AscLin_Init API

Syntax	<pre>void Lin_17_AscLin_Init (const Lin_17_AscLin_ConfigType * const Config)</pre>	
Service ID	0x00	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant	
Parameters (in)	Config	Pointer to LIN driver configuration set.
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	<p>Initializes the LIN module.</p> <p>The state after the Init call in:</p> <p>Autosar version 422 is LIN_CH_OPERATIONAL and</p> <p>Autosar version 440 is LIN_CH_SLEEP and enables wakeup detection.</p> <p>From multicore perspective,</p> <p>only the configured cores can be initialized else DET</p> <p>LIN_17_ASCLIN_E_CORE_NOT_CONFIGURED will be triggered and</p> <p>Master and slave core is not applicable for the driver.</p>	
Source	AUTOSAR	
Error handling	LIN_17_ASCLIN_E_INVALID_POINTER, LIN_17_ASCLIN_E_CORE_NOT_CONFIGURED, LIN_17_ASCLIN_E_STATE_TRANSITION, LIN_17_ASCLIN_E_TIMEOUT	
Configuration dependencies	-	
User hints	-	
SFR accessed	ASCLIN_BITCON(w), ASCLIN_BRG(w), ASCLIN_CLC(rw), ASCLIN_CSR(rw), ASCLIN_FLAGSCLEAR(w), ASCLIN_FLAGSENABLE(w), ASCLIN_FRAMECON(w), ASCLIN_IOCRR(w), ASCLIN_KRST0(rw), ASCLIN_KRST1(w), ASCLIN_KRSTCLR(w), ASCLIN_LIN_BTIMER(w), ASCLIN_LIN_CON(w), ASCLIN_LIN_HTIMER(w), ASCLIN_RXFIFOCON(w), ASCLIN_TXFIFOCON(w) <p><i>Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.</i></p>	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1 Lin_17_AscLin driver
1.3.3.6 Lin_17_AscLin_SendFrame
Table 53 Specification for Lin_17_AscLin_SendFrame API

Syntax	<pre>Std_ReturnType Lin_17_AscLin_SendFrame (const uint8 Channel, const Lin_PduType * const PduInfoPtr)</pre>	
Service ID	0x04	
Sync/Async	Asynchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant	
Parameters (in)	Channel PduInfoPtr	LIN channel to be addressed Pointer to PDU containing the PID, checksum model, response type, DI and SDU data pointer
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Std_ReturnType	E_OK: Send command has been accepted. E_NOT_OK: Send command has not been accepted, development or production error occurred.
Description	<p>This API configures the LIN hardware channel for one of the below conditions based on the direction of frame response for particular PID.</p> <ul style="list-style-type: none"> - Transmit header only: ignore any response received - Transmit header followed by the response - Transmit header and read the response <p><i>Note: This service is only applicable if the LIN driver is configured with one or more LIN master channel.</i></p>	
Source	AUTOSAR	
Error handling	LIN_17_ASCLIN_E_CORE_CHANNEL_MISMATCH, LIN_17_ASCLIN_E_STATE_TRANSITION, LIN_17_ASCLIN_E_PARAM_POINTER, LIN_17_ASCLIN_E_INVALID_CHANNEL, LIN_17_ASCLIN_E_UNINIT	
Configuration dependencies	LinNodeType	
User hints	-	

(table continues...)

1 Lin_17_AscLin driver
Table 53 (continued) Specification for Lin_17_AscLin_SendFrame API

SFR accessed	ASCLIN_CSR(rw), ASCLIN_DATCON(w), ASCLIN_FLAGSCLEAR(w), ASCLIN_FLAGSENABLE(w), ASCLIN_FLAGSSET(w), ASCLIN_FRAMECON(w), ASCLIN_RXFIFOCON(w), ASCLIN_TXDATA(w), ASCLIN_TXFIFOCON(w) <i>Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.</i>
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.7 Lin_17_AscLin_Wakeup
Table 54 Specification for Lin_17_AscLin_Wakeup API

Syntax	Std_ReturnType Lin_17_AscLin_Wakeup (const uint8 Channel)	
Service ID	0x07	
Sync/Async	Asynchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant	
Parameters (in)	Channel	LIN channel to be addressed
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	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. - When DET is OFF and LIN driver is not in LIN_CH_SLEEP state.
Description	Generates a wake up pulse and sets the channel state to LIN_CH_OPERATIONAL. Disables the falling edge detection on Rx pin used for the LIN channel wakeup.	
Source	AUTOSAR	
Error handling	LIN_17_ASCLIN_E_STATE_TRANSITION, LIN_17_ASCLIN_E_UNINIT, LIN_17_ASCLIN_E_CORE_CHANNEL_MISMATCH, LIN_17_ASCLIN_E_INVALID_CHANNEL	
Configuration dependencies	-	
User hints	-	

(table continues...)

1 Lin_17_AscLin driver
Table 54 (continued) Specification for Lin_17_AscLin_Wakeup API

SFR accessed	ASCLIN_FLAGSCLEAR(r), ASCLIN_FLAGSENABLE(rw), ASCLIN_FLAGSSET(w), ASCLIN_RXFIFOCON(r), ASCLIN_TXDATA(w), ASCLIN_TXFIFOCON(rw) <i>Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.</i>
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.8 Lin_17_AscLin_WakeupInternal
Table 55 Specification for Lin_17_AscLin_WakeupInternal API

Syntax	Std_ReturnType Lin_17_AscLin_WakeupInternal (const uint8 Channel)	
Service ID	0x0b	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant	
Parameters (in)	Channel	LIN channel to be addressed
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Std_ReturnType	E_OK: Wakeup request has been accepted. E_NOT_OK: - Wakeup request has not been accepted, development or production error occurred. - When DET is OFF and LIN driver is not in LIN_CH_SLEEP state.
Description	Sets the channel state to LIN_CH_OPERATIONAL without generating a wake up pulse. Disables the falling edge detection on Rx pin used for the LIN channel wakeup.	
Source	AUTOSAR	
Error handling	LIN_17_ASCLIN_E_STATE_TRANSITION, LIN_17_ASCLIN_E_UNINIT, LIN_17_ASCLIN_E_INVALID_CHANNEL, LIN_17_ASCLIN_E_CORE_CHANNEL_MISMATCH	
Configuration dependencies	-	
User hints	-	

(table continues...)

1 Lin_17_AscLin driver
Table 55 (continued) Specification for Lin_17_AscLin_WakeupInternal API

SFR accessed	ASCLIN_FLAGSCLEAR(r), ASCLIN_FLAGSENABLE(rw), ASCLIN_RXFIFOCON(r), ASCLIN_TXFIFOCON(r) <i>Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.</i>
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.9 Lin_17_AscLin_GetVersionInfo
Table 56 Specification for Lin_17_AscLin_GetVersionInfo API

Syntax	<pre>void Lin_17_AscLin_GetVersionInfo (Std_VersionInfoType * const versioninfo)</pre>	
Service ID	0x01	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant	
Parameters (in)	-	-
Parameters (out)	versioninfo	Pointer to where is stored the version information of this module.
Parameters (in - out)	-	-
Return	void	-
Description	Returns the version information of this module.	
Source	AUTOSAR	
Error handling	LIN_17_ASCLIN_E_PARAM_POINTER	
Configuration dependencies	LinVersionInfoApi	
User hints	-	
SFR accessed	-	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.4 Notifications and Callbacks

The LIN driver does not provide any notification or callbacks.

1 Lin_17_AscLin driver

1.3.5 Scheduled functions

The LIN driver does not provide any scheduled functions.

1.3.6 Interrupt service routines

This section lists all the interrupt handlers of the LIN driver.

1.3.6.1 Lin_17_AscLin_IsrError

Table 57 Specification for Lin_17_AscLin_IsrError API

Syntax	<pre>void Lin_17_AscLin_IsrError (const uint8 HwUnit)</pre>	
Service ID	-	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant (For different channels)	
Parameters (in)	HwUnit	Represents HW module number.
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	<p>This ISR will be called whenever there is a data transmission or reception error or a wakeup signal is detected in ASCLIN. The ISR will call EcuM_CheckWakeup, if channel wakeup support is enabled and wakeup signal is detected.</p> <p>For Slave node, if any error occurs then it will be indicated to the interface layer by callback function LinIf_LinErrorIndication with type of error as parameter.</p>	
Source	IFX	
Error handling	-	
Configuration dependencies	LinMasterInterruptEnable	
User hints	-	
SFR accessed	ASCLIN_FLAGS(r), ASCLIN_FLAGSCLEAR(rw), ASCLIN_FLAGSENABLE(rw), ASCLIN_RXFIFOCON(rw), ASCLIN_TXFIFOCON(rw) <i>Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.</i>	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1 Lin_17_AscLin driver
1.3.6.2 Lin_17_AscLin_IsrReceive
Table 58 Specification for Lin_17_AscLin_IsrReceive API

Syntax	<pre>void Lin_17_AscLin_IsrReceive (const uint8 HwUnit)</pre>	
Service ID	-	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant (For different channels)	
Parameters (in)	HwUnit	Represents HW module number.
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	<p>This ISR will be called by Master node when:</p> <ul style="list-style-type: none"> - Slave response data is completely received without any errors. <p>This ISR will be called by Slave node when:</p> <ul style="list-style-type: none"> - Header is completely received by the ASCLIN without any errors. The received LIN header is indicated to interface layer by LinIf_HeaderIndication. - Slave node is the subscriber of the response then, on the successful reception of the LIN response it should be made available to interface layer by LinIf_RxIndication. 	
Source	IFX	
Error handling	-	
Configuration dependencies	LinMasterInterruptEnable	
User hints	-	
SFR accessed	ASCLIN_DATCON(w), ASCLIN_FLAGS(r), ASCLIN_FLAGSCLEAR(rw), ASCLIN_FLAGSENABLE(rw), ASCLIN_FLAGSSET(w), ASCLIN_RXDATA(r), ASCLIN_RXFIFOCON(rw), ASCLIN_TXDATA(w), ASCLIN_TXFIFOCON(rw) <i>Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.</i>	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1 Lin_17_AscLin driver
1.3.6.3 Lin_17_AscLin_IsrTransmit
Table 59 Specification for Lin_17_AscLin_IsrTransmit API

Syntax	<pre>void Lin_17_AscLin_IsrTransmit (const uint8 HwUnit)</pre>	
Service ID	-	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant (For different channels)	
Parameters (in)	HwUnit	Represents HW module number.
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	<p>This ISR will be called by:</p> <ul style="list-style-type: none"> - Master node whenever the response data is successfully transmitted by the ASCLIN without any errors. - Slave node, the successful transmission of LIN response should be confirmed by the confirmation callback function LinIf_TxConfirmation. 	
Source	IFX	
Error handling	-	
Configuration dependencies	LinMasterInterruptEnable	
User hints	-	
SFR accessed	ASCLIN_FLAGS(r), ASCLIN_FLAGSCLEAR(rw), ASCLIN_FLAGSENABLE(rw), ASCLIN_RXFIFOCON(rw), ASCLIN_TXFIFOCON(rw) <i>Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.</i>	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.7 Callout

The driver does not support any callout functions.

1.3.8 Errors Handling

This section describes the various error types reported by the LIN driver.

1 Lin_17_AscLin driver

Error Name: Description	Source	Error ID (AS422)	Type (AS422)	Error ID (AS440)	Type (AS440)
LIN_17_ASCLIN_E_CORE_CHANNEL_MISMATCH: Error reported when LIN Kernel(HwUnit) is not allocated to the core from which the API is called	IFX	0x65	DET	0x65	DET
LIN_17_ASCLIN_E_CORE_NOT_CONFIGURED: Error reported when no channels configured for the core	IFX	0x64	DET	0x64	DET
LIN_17_ASCLIN_E_INVALID_CHANNEL: API service used with an invalid or inactive channel parameter or if the service is applicable only to Master node but called by Slave node.	AUTOSAR	0x02	DET	0x02	DET
LIN_17_ASCLIN_E_INVALID_POINTER: API service called with invalid configuration pointer.	AUTOSAR	0x03	DET	0x03	DET
LIN_17_ASCLIN_E_PARAM_POINTER: API service called with a NULL pointer.	AUTOSAR	0x05	DET	0x05	DET
LIN_17_ASCLIN_E_STATE_TRANSITION: Invalid state transition for the current state.	AUTOSAR	0x04	DET	0x04	DET

1 Lin_17_AscLin driver

Error Name: Description	Source	Error ID (AS422)	Type (AS422)	Error ID (AS440)	Type (AS440)
LIN_17_ASCLIN_E_TIMEOUT: In AUTOSAR 4.2.2, - On any error in bit set/reset within timeout duration - Mcal_Wrapper_Dem_ReportErrorStatus (LIN_17_ASCLIN_E_TIMEOUT, DEM_EVENT_STATUS_FAILED) - If bit set/reset occurred within the timeout duration - Mcal_Wrapper_Dem_ReportErrorStatus (LIN_17_ASCLIN_E_TIMEOUT, DEM_EVENT_STATUS_PASSED) In AUTOSAR 4.4.0, - On any error in bit set/reset within timeout duration - Mcal_Wrapper_Dem_SetEventStatus(LIN_17_ASCLIN_E_TIMEOUT, DEM_EVENT_STATUS_FAILED) - If bit set/reset occurred within the timeout duration - Mcal_Wrapper_Dem_SetEventStatus(LIN_17_ASCLIN_E_TIMEOUT, DEM_EVENT_STATUS_PASSED)	AUTOSAR	Assigned by DEM	Production Error	Assigned by DEM	Production Error
LIN_17_ASCLIN_E_UNINIT: API service used without module initialization.	AUTOSAR	0x00	DET	0x00	DET

1.3.9 Deviations and limitations

The section describes the deviations and limitations of the Lin_17_AscLin driver.

1.3.9.1 Deviations

This section describes the deviations of the Lin_17_AscLin driver.

1.3.9.1.1 Software specification deviations

The section describes the deviations from software specification.

Table 60 Known deviations

Reference	Deviation
(table continues...)	

1 Lin_17_AscLin driver
Table 60 (continued) Known deviations

AUTOSAR header file inclusion requirement for LIN module [SWS_Lin_00075]	As per AUTOSAR requirement, the EcuM_Cbk.h file should be included in the Lin_17_AscLin.c file. However, the LIN driver configuration structure defined in Lin_17_AscLin.h file refer to the data type EcuM_WakeupSourceType from the EcuM module. Hence to avoid compilation error, EcuM_Cbk.h is included in the Lin_17_AscLin.h file.
AUTOSAR LinClockRef configuration parameter [ECUC_Lin_00094]	The configuration parameter is not used and is not editable, instead clock reference for all LIN channels are provided commonly by the LinSysClockRef parameter.
For all requirements related to Production errors	Reporting of Production error: Dem_ReportErrorStatus is done through Mca1_Wrapper_Dem_ReportErrorStatus interface for AUTOSAR 4.2.2 and Dem_SetEventStatus is done through Mca1_Wrapper_Dem_SetEventStatus interface for AUTOSAR 4.4.0. All production error related datatypes and modified interfaces inclusion shall be done via Mca1_Wrapper.h

1.3.9.1.2 AMDC Violations

The Lin_17_AscLin driver does not have any AMDC violations.

1.3.9.1.3 VSMD Violations

<i>Violations reported by VSMD checker tool for EB03 </i>

This section describes the violations reported by the EB VSMD checker tool with respect to AUTOSAR.

Table 61 Violations reported by the VSMD checker tool for EB03

Rule ID:	EB03
VSMD Node(s):	/AURIX2G/EcucDefs/Lin/LinDemEventParameterRefs /AURIX2G/EcucDefs/Lin/LinDemEventParameterRefs/ LIN_E_TIMEOUT /AURIX2G/EcucDefs/Lin/LinGlobalConfig/LinChannel/ LinChannelEcuMWakeupSource /AURIX2G/EcucDefs/Lin/LinGlobalConfig/LinChannel/ LinChannelEcucPartitionRef
Description:	The StMD node has LOWER-MULTIPLICITY=0 and UPPER-MULTIPLICITY=1. The VSMD-node shall get the OPTIONAL-attribute instead of creating a list.
Additional Information:	

Table 62 Violations reported by the VSMD checker tool for EB09

Rule ID:	EB09
----------	------

(table continues...)

1 Lin_17_AscLin driver
Table 62 (continued) *Violations reported by the VSMD checker tool for EB09*

VSMD Node(s):	/AURIX2G/EcucDefs/Lin
Description:	EB specific rule to check consistency of parameter postBuildVariantUsed.
Additional Information:	

Table 63 *Violations reported by the VSMD checker tool for EcucSws_1014*

Rule ID:	EcucSws_1014
VSMD Node(s):	/AURIX2G/EcucDefs/Lin /AURIX2G/EcucDefs/Lin/LinGeneral /AURIX2G/EcucDefs/Lin/LinGlobalConfig/LinChannel
Description:	Additional vendor specific parameter definitions (using ParameterTypes), container definitions and references shall be added to the VSMD according to the alphabetical order.
Additional Information:	

Table 64 *Violations reported by the VSMD checker tool for EcucSws_1035*

Rule ID:	EcucSws_1035
----------	--------------

(table continues...)

1 Lin_17_AscLin driver
Table 64 (continued) *Violations reported by the VSMD checker tool for EcucSws_1035*

VSMD Node(s):	/AURIX2G/EcucDefs/Lin /AURIX2G/EcucDefs/Lin/LinDemEventParameterRefs /AURIX2G/EcucDefs/Lin/LinDemEventParameterRefs/ LIN_E_TIMEOUT /AURIX2G/EcucDefs/Lin/LinGeneral /AURIX2G/EcucDefs/Lin/LinGeneral/ LinDevErrorDetect /AURIX2G/EcucDefs/Lin/LinGeneral/ LinEcucPartitionRef /AURIX2G/EcucDefs/Lin/LinGeneral/LinIndex /AURIX2G/EcucDefs/Lin/LinGeneral/ LinTimeoutDuration /AURIX2G/EcucDefs/Lin/LinGeneral/LinVersionInfoApi /AURIX2G/EcucDefs/Lin/LinGlobalConfig /AURIX2G/EcucDefs/Lin/LinGlobalConfig/LinChannel /AURIX2G/EcucDefs/Lin/LinGlobalConfig/LinChannel/ LinChannelBaudRate /AURIX2G/EcucDefs/Lin/LinGlobalConfig/LinChannel/ LinChannelEcuMWakeupSource /AURIX2G/EcucDefs/Lin/LinGlobalConfig/LinChannel/ LinChannelEcucPartitionRef /AURIX2G/EcucDefs/Lin/LinGlobalConfig/LinChannel/ LinChannelId /AURIX2G/EcucDefs/Lin/LinGlobalConfig/LinChannel/ LinChannelWakeupSupport /AURIX2G/EcucDefs/Lin/LinGlobalConfig/LinChannel/ LinClockRef /AURIX2G/EcucDefs/Lin/LinGlobalConfig/LinChannel/ LinNodeType
Description:	For Containers, Parameters and References elements UUID must be unique (also between StMD and VSMD).
Additional Information:	

Table 65 *Violations reported by the VSMD checker tool for EcucSws_2101*

Rule ID:	EcucSws_2101
VSMD Node(s):	/AURIX2G/EcucDefs/Lin/POST_BUILD_VARIANT_USED
Description:	For each ConfigurationVariant supported by the ModuleDef, there must be one ImplementationConfigClass element. In VSMD, the ImplementationConfigClass is mandatory.
Additional Information:	

1 Lin_17_AscLin driver
Table 66 *Violations reported by the VSMD checker tool for EcucSws_6003*

Rule ID:	EcucSws_6003
VSMD Node(s):	/AURIX2G/EcucDefs/Lin
Description:	The SHORT-NAME of the AR-PACKAGEs of StMD and VSMD must be different to ensure a unique SHORT-NAME-path.
Additional Information:	

Table 67 *Violations reported by the VSMD checker tool for TpsEcuc_06051_ASR41*

Rule ID:	TpsEcuc_06051_ASR41
VSMD Node(s):	/AURIX2G/EcucDefs/Lin/POST_BUILD_VARIANT_USED
Description:	The implementationConfigClass of an EcucParameterDef or EcucAbstractReferenceDef in VSMD shall be the same or higher (where PreCompile configuration class is considered to be the lowest and PostBuild the highest) as in StMD with respect to the selected subset defined by the actually implemented supportedConfigVariant.
Additional Information:	

1.3.9.2 Limitations

The Lin_17_AscLin driver does not have any limitations.

Revision history

Revision history

Table 68 **Revision History**

Date	Version	Description
2023-07-05	5.0	Document is released
2023-07-05	4.1	<ul style="list-style-type: none"> - In Integration with AUTOSAR stack, section 1.1.4.1, DEM module section has been removed and Mcal_wrapper module section has been added. - DEM has been modified to Production error where applicable. - Updated Figure-1 Hardware-software mapping section to include Mcal_Wrapper module and removed Dem module. - Updated the C file Structure in section 1.1.3.1 to include Mcal_Wrapper.h and removed Dem.h. - In Error Handling table in Section 1.3.8, LIN_17_ASCLIN_E_TIMEOUT parameter Description is updated with Mcal_Wrapper API. - Updated the section 1.3.9.1.1: Software Specification Deviations for Autosar requirements. <p>Updated Reference from "SWS_Lin_00226: Rte_Dem_Types.h" to "For all requirements related to Production errors".</p> <p>Updated Description of "SWS_Lin_00226: Rte_Dem_Types.h" to add Mcal_Wrapper Module Information.</p> <ul style="list-style-type: none"> - ASIL Level has been updated to Safety level in Section 1.3.3.
2021-11-10	4.0	Document is released
2021-10-29	3.1	<ul style="list-style-type: none"> • Config variant attribute table information is removed and added this information in 'Configuration interfaces' section.
2021-02-24	3.0	Document is released
2021-02-23	2.1	<ul style="list-style-type: none"> • Updated SFR information in Lin_17_AscLin_SendFrame API • Added SWS_Lin_00226 in the known deviations section
2020-11-26	2.0	Document is released
2020-11-26	1.1	<ul style="list-style-type: none"> • Updated DEM section under Integration with AUTOSAR stack • Renamed configuration parameter LinInterruptEnable to LinMasterInterruptEnable • Added SFR information in the Functions - APIs and Interrupt service routines sections
2020-08-17	1.0	Document is released
2020-08-07	0.1	<ul style="list-style-type: none"> • Initial version • Lin_17_AscLin driver chapter moved from MC-ISAR_TC3xx_UM_Basic to this document • Added AMDC and VSMD violation tables • Added deviations from AUTOSAR 4.4.0

Trademarks

All referenced product or service names and trademarks are the property of their respective owners.

Edition 2023-07-05

Published by

Infineon Technologies AG
81726 Munich, Germany

© 2023 Infineon Technologies AG
All Rights Reserved.

Do you have a question about any aspect of this document?

Email: erratum@infineon.com

Document reference
IFX-ocr1484806431059

Important notice

The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics ("Beschaffenhheitsgarantie").

With respect to any examples, hints or any typical values stated herein and/or any information regarding the application of the product, Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind, including without limitation warranties of non-infringement of intellectual property rights of any third party.

In addition, any information given in this document is subject to customer's compliance with its obligations stated in this document and any applicable legal requirements, norms and standards concerning customer's products and any use of the product of Infineon Technologies in customer's applications.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer's technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

Warnings

Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies' products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.