Elektrobit

EB tresos® AutoCore Generic 8

CAN Stack documentation

release notes update for the CanIf module

product release 8.8.7

Elektrobit Automotive GmbH
Am Wolfsmantel 46
91058 Erlangen, Germany
Phone: +49 9131 7701 0
Fax: +49 9131 7701 6333
Email: info.automotive@elektrobit.com

Technical support

https://www.elektrobit.com/support

Legal disclaimer

Confidential information.

Table of Contents

1. Overview 4
2. CanIf module release notes 5
    2.1. Change log 5
    2.2. New features 21
    2.3. Elektrobit-specific enhancements 21
    2.4. Deviations 26
    2.5. Limitations 35
    2.6. Open-source software 37

1. Overview

This document provides you with the release notes to accompany an update to the `CanIf` module. Refer to the changelog Section 2.1, “Change log” for details of changes made for this update.

Release notes details

- EB tresos AutoCore release version: 8.8.7
- EB tresos Studio release version: 29.2.0
- AUTOSAR R4.0 Rev 3
- Build number: B577598

2. CanIf module release notes

- AUTOSAR R4.0 Rev 3
- AUTOSAR SWS document version: 5.0.0
- Module version: 6.10.31.B577598
- Supplier: Elektrobit Automotive GmbH

2.1. Change log

This chapter lists the changes between different versions.

Module version 6.10.31

2022-11-04

- Fixed known issue: Possible mirroring message loss due to missing TxConfirmations.

Module version 6.10.30

2022-10-28

- Fixed known issue: Possible mirroring message loss due to missing TxConfirmations.

Module version 6.10.29

2022-09-16

- ASCCANIF-1660 Fixed known issue: HRH index calculation can lead to out-of-bounds access.

Module version 6.10.28

2022-08-19

- ASCCANIF-1650 Fixed known issue: Inconsistent CanIf configuration might lead to EB Tresos Studio errors.

Module version 6.10.27

2022-07-22

- Tx and Rx Bus-Adapter user specific callout functions are added to support interception of Can messages.

Module version 6.10.26

2022-06-10

- Internal module improvement. This module version update does not affect module functionality.

Module version 6.10.25

2022-05-13

- Support of PB variant for configparameter CanIfTrcvDrvCfg in CanIf such that only CanTrcvs referenced by CanSM could be a wakeup source.

Module version 6.10.24

2022-04-08

- Added support for SEVs reporting to IdsM.

Module version 6.10.23

2022-02-18

- Added truncation options for a transmitting PDU.
- Changed wakeup behavior.
- Added support for queue size measurement for decoupled processing.
- ASCCANIF-1499 Fixed known issue: Duplicate CanObjectId assigns PDUs to wrong HOH
- ASCCANIF-1502 Fixed known issue: HTH offset is not considered during Tx buffering on multiple CAN drivers
- Internal module improvement. This module version update does not affect module functionality.
- Internal module improvement. This module version update does not affect module functionality.
- Added the capability of having Tx callback function with result (E_OK/E_NOT_OK) for CDD and J1939 upper layer only.

- ASCCANIF-1528 Added support for reporting Can frames to non AUTOSAR mirroring concepts.
- Internal module improvement. This module version update does not affect module functionality.
- Extend Reliable TxConfirmation for CanTSyn.
- Added support for distribution of CAN stack along network boudaries.
- Added support for distribution of CAN stack along network boudaries for decoupled processing.
- ASCCANIF-1548 Fixed known issue: CanIf does not notify the upper layer with TxConfirmation with a result E_NOT_OK
- ASCCANIF-1562 Fixed known issue: Deadlock in MainFunction occurs when adding a new element in the queue during decoupled processing
- ASCCANIF-1564 Fixed known issue: Rx indications can be lost with CanIfDecoupledProcessing turned ON
- Change Bus mirroring buffer to be a shared space among all TX PDUs
- ASCCANIF-1571 Fixed known issue: RxIndication is lost during decoupled processing when the remaining queue size equals PDU length(+1)
- Internal module improvement. This module version update does not affect module functionality.
- ASCCANIF-1580 Fixed known issue: Configuration is not valid if Tx Pdu Notify feature is enabled and there are Rx Pdu's configured with notification function enabled.
- ASCCANIF-1515 CanIf supports multicore with Bus Mirroring.
- ASCCANIF-1585 Fixed known issue: Wrong MemMap section is used when ReadTx/RxPduNotifyStatusApi and Multicore are enabled.
- Internal module improvement. This module version update does not affect module functionality.

Module version 6.10.22

2021-01-28

Module version 6.10.21

2021-12-10

- Internal module improvement. This module version update does not affect module functionality.

Module version 6.10.20

2021-11-12

- ASCCANIF-1562 Fixed known issue: Deadlock in MainFunction occurs when adding a new element in the queue during decoupled processing
- ASCCANIF-1564 Fixed known issue: Rx indications can be lost with CanIfDecoupledProcessing turned ON
- Change Bus mirroring buffer to be a shared space among all TX PDUs
- ASCCANIF-1571 Fixed known issue: RxIndication is lost during decoupled processing when the remaining queue size equals PDU length(+1)

Module version 6.10.19

2021-10-08

- Added support for distribution of CAN stack along network boudaries for decoupled processing.
- ASCCANIF-1548 Fixed known issue: CanIf does not notify the upper layer with TxConfirmation with a result E_NOT_OK

Module version 6.10.18

2021-09-17

Module version 6.10.17

2021-10-28

- Internal module improvement. This module version update does not affect module functionality.
- Extend Reliable TxConfirmation for CanTSyn.
- Added support for distribution of CAN stack along network boudaries.
- Added support for distribution of CAN stack along network boudaries for decoupled processing.
- ASCCANIF-1548 Fixed known issue: CanIf does not notify the upper layer with TxConfirmation with a result E_NOT_OK

Module version 6.10.16

2021-07-28

- ASCCANIF-1528 Added support for reporting Can frames to non AUTOSAR mirroring concepts.

Module version 6.10.15

2021-06-25

- Added the capability of having Tx callback function with result (E_OK/E_NOT_OK) for CDD and J1939 upper layer only.

Module version 6.10.14

2021-04-30

- Internal module improvement. This module version update does not affect module functionality.
- Internal module improvement. This module version update does not affect module functionality.

Module version 6.10.13

2021-04-09

- ASCCANIF-1502 Fixed known issue: HTH offset is not considered during Tx buffering on multiple CAN drivers

Module version 6.10.12

2021-03-05

- Added support for queue size measurement for decoupled processing.
- ASCCANIF-1499 Fixed known issue: Duplicate CanObjectId assigns PDUs to wrong HOH

Module version 6.10.11

2021-02-12

- Added truncation options for a transmitting PDU.
- Changed wakeup behavior.

Module version 6.10.10

2021-01-22

- Internal module improvement. This module version update does not affect module functionality.

Module version 6.10.9

2020-12-18

- ASCCANIF-1469 Fixed known issue: CanIf requires multiple transceivers in order to generate the CanIf_-GetTrcvMode() interface needed by Mirror in multi-core context
- ASCCANIF-1474 Fixed known issue: Mirror support related out-of-bounds access
- Added support for disabling/Enabling Software Filtering feature for optimization purpose
- Added support for CanIf custom hook on succesful reception

Module version 6.10.8

2020-10-23

- Added support for Defensive Programming.

Module version 6.10.7

2020-09-25

- Internal module improvement. This module version update does not affect module functionality.

Module version 6.10.6

2020-08-28

- ASCCANIF-1449 Fixed known issue: Tx Confirmations can be lost with CanIfDecoupledProcessing turned ON

Module version 6.10.5

2020-07-31

- Tresos:Automatically calculation of settings of parameters

Module version 6.10.4

2020-06-19

- ASCCANIF-1240 Fixed known issue: Failing module initialization due to duplicate post-build configuration

Module version 6.10.3

2020-05-22

- Added support for J1939Nm and J1939Tp as standard upper layers.

Module version 6.10.2

2020-04-24

- Added support for Autosar HandleId concept for CDDs.
- ASCCANIF-1406 Fixed known issue: <UpperLayer>_RxIndication may not be called with software filtering and CanIdMask.

Module version 6.10.1

2020-03-25

- On the Tx part, frames that are larger than 64 bytes (FD)/8 bytes (nonFD), are now truncated to the maximum length and transmitted.

Module version 6.10.0

2020-02-21

- Provided support for multiple CAN drivers.
- ASCCANIF-1403 Fixed known issue: CanIf fails to compile if CanIfPublicTrcvWakeupSupport = TRUE and CanIfPublicTrcvSupport = FALSE.

- Provided ASR 4.3/4.4 MCAL Support.

Module version 6.9.24

2019-12-06

- Internal module improvement. This module version update does not affect module functionality.

Module version 6.9.23

2019-10-11

- ASCCANIF-1379 Fixed known issue: Wrong CanTrcv vendor extension is generated in CanIf.

Module version 6.9.22

2019-09-06

- ASCCANIF-1372 Fixed known issue: <UpperLayer>_RxIndication() is not called with the newly translated Rx CanID.
- ASCCANIF-1374 Fixed known issue: Newly translated Rx CanID is not provided to Mirror.

Module version 6.9.21

2019-08-09

- ASCCANIF-1356 Fixed known issue: Incorrect loop variable type used in function CanIf_ResetMirrBuff.

Module version 6.9.20

2019-06-14

- Internal module improvement. This module version update does not affect module functionality.

Module version 6.9.19

2019-05-17

- Added metadata support.

Module version 6.9.18

2019-04-18

- ASCCANIF-1356 Fixed known issue: Incompatibility between CanIf and 4.2.2 CanTrcv.

Module version 6.9.17

2019-03-22

- Internal module improvement. This module version update does not affect module functionality.

Module version 6.9.16

2019-02-15

- Internal module improvement. This module version update does not affect module functionality.
- Added support for for Bus Mirroring

Module version 6.9.15

2019-01-24

- Internal module improvement. This module version update does not affect module functionality.

Module version 6.9.14

2018-12-13

- ASCCANIF-1332 Fixed known issue: Incorrect configuration of PDU queues if PduQueueSize>255.
- ASCCANIF-1333 Fixed known issue: Interruption of CanIf_MainFunctionRx_Generic can lead to inconsistent queue status.

Module version 6.9.13

2018-11-23

- ASCCANIF-1329 Fixed known issue: Unconditional dependency on optional CanTrcv interfaces.

Module version 6.9.12

2018-10-26

- Internal module improvement. This module version update does not affect module functionality.

Module version 6.9.11

2018-08-24

- Internal module improvement. This module version update does not affect module functionality.

Module version 6.9.10

2018-06-22

- Internal module improvement. This module version update does not affect module functionality.

Module version 6.9.9

2018-05-25

- Support for decoupled processing of reception and transmission confirmation events.

Module version 6.9.8

2018-04-20

- Add support for uint32 PduLengthType.

Module version 6.9.7

2018-02-16

- ASCCANIF-1286 Fixed known issue: Possible prolongation of Tx confirmation ISR when Tx buffering is enabled.

Module version 6.9.6

2018-01-19

- Added support of CanTSyn as upper layer.

Module version 6.9.5

2017-12-15

- ASCCANIF-1267 Fixed known issue: Compilation fails if DLC check is disabled for CAN 4.2 support.

Module version 6.9.4

2017-09-22

- Internal module improvement. This module version update does not affect module functionality.

Module version 6.9.3

2017-07-28

- Post-build selectable support.

Module version 6.9.2

2017-05-05

- Internal module improvement. This module version update does not affect module functionality.

Module version 6.9.1

2017-03-31

- Internal module improvement. This module version update does not affect module functionality.

Module version 6.9.0

2017-03-10

- Added support for AUTOSAR 4.2.2 CAN drivers.

Module version 6.8.0

2017-03-03

- CAN and CANFD frames on the same CanId shall be received in two seperate RxPdus
- Move integration requirements to separate reqm file.

Module version 6.7.0

2017-02-03

- Do not enable partial networking TX filter after BusOff.
- Internal module improvement. This module version update does not affect module functionality.

Module version 6.6.2

2017-01-05

- Internal module improvement. This module version update does not affect module functionality.

Module version 6.6.1

2016-11-04

- Internal module improvement. This module version update does not affect module functionality.

Module version 6.6.0

2016-09-23

- Disable partial networking TX filter in case of ongoing transmission.

Module version 6.5.6

2016-05-25

- ASCCANIF-1169 Fixed known issue: CanIf may discard L-Pdu if CanIf_TxConfirmation() preempts CanIf_-Transmit()
- Added handle Id wizard support for configuration parameter CanIfCtrlId.

Module version 6.5.5

2016-02-05

- Added support for Debug & Trace with custom header file configurable via parameter `BaseDbgHeader-File`

Module version 6.5.4

2015-11-06

- Added a check to avoid multiple references to the same Can controller.

Module version 6.5.3

2015-06-19

- ASCCANIF-1139 Fixed known issue: CanIf passes the configured length value (DLC) to the upper layer modules
- Added support for 64 bytes CAN-FD frames
- ASCCANIF-1142 Fixed known issue: CanIf may include CanTrcv header file although CanTrcv support is disabled
- ASCCANIF-1145 Fixed known issue: CanIf does not disable PnTxFilter
- ASCCANIF-1147 Fixed known issue: CanIf does not include CanSM_Cbk.h

Module version 6.5.2

2015-02-20

- Internal module improvement. This module version update does not affect module functionality

Module version 6.5.1

2014-12-19

- Removed AUTOSAR 3.x compliant symbolic name value macros and updated the logic to only provide AUTOSAR 4.0.2 compliant macros if macro CANIF_PROVIDE_LEGACY_SYMBOLIC_NAMES is defined
- Added support for configurable mapping of CanIf_IsValidConfig function to dedicate memory section
- CanIf calls EcuM_ValidateWakeupEvent() only if wakeup has been successfully validated

Module version 6.5.0

2014-10-02

- ASCCANIF-1072 Fixed known issue: CanIfTxPduCfg containers cannot be added or removed at post-build time
- Introduced configuration container CanIfUpperLayerConfig to define upper layer modules at Pre-compile time
- Added proper name mangling for header files and API functions of CanTrcv
- Improved parameter description and error checks for configurations with multiple CanTrcv

Module version 6.4.4

2014-04-25

- ASCCANIF-1068 Fixed known issue: CanIf module reports an error if config time support is enabled
- ASCCANIF-1070 Fixed known issue: Build error due to missing file CanIf_PBcfg.c if code generation for CanIf is disabled and only post-build configuration is compiled

Module version 6.4.3

2013-11-15

- Removed warning from unused configuration parameter CanIfHrhCanHandleTypeRef
- Implemented support for CAN FD according to AUTOSAR R4.1 Rev 1

Module version 6.4.2

2013-10-11

- Changed MCG to introduce generation of XML code for Binary Code Generation

Module version 6.4.1

2013-06-28

- ASCCANIF-954 Fixed known issue: CanIf_SetControllerMode() does not report the DET error code CANIF_E_PARAM_CTRLMODE
- Implemented a configuration signature for consistency checks
- Updated memory sections of variables initialized during `CanIf_Init()` to NO_INIT
- Added a signature check to verify the precompile and link time configuration
- ASCCANIF-979 Fixed known issue: CanIf aborts compilation if AUTOSAR R4.0 Rev 3 compliant Can driver is used
- ASCCANIF-985 Fixed known issue: API services CanIf_ControllerBusOff() and CanIf_ControllerModeIndication() may report a DET error for a valid controller ID
- ASCCANIF-997 Fixed known issue: CanIf post-build-time configuration does not compile if used with PbcfgM
- ASCCANIF-999 Fixed known issue: CanIf calls Can_Write within a critical section causing system failure if interrupts are locked

Module version 6.4.0

2013-02-08

- ASCCANIF-934 Fixed known issue: A compiler error occurs if only a single HOH (sum of all HRHs and HTHs) is configured
- ASCCANIF-906 Fixed known issue: CanIfPublicHandleTypeEnum must be configured to UINT16 if more than 254 HOHs are used
- Updated CanIf to Can driver API according to AUTOSAR R4.0 Rev 3

Module version 6.3.0

2012-10-12

- Implementation of Handle-Id policy according to AUTOSAR R4.0 Rev 3
- The top-level structure of the software-component description in the ARXML files changed from `/AUTOSAR/CanIf`to `/AUTOSAR_CanIf`

- ASCCANIF-891 Fixed known issue: Condition for availability of configuration parameter CanIfDispatchUserConfirmPnAvailabilityName is incorrect

Module version 6.2.0

2012-06-20

- Update of config schema files according to AUTOSAR R4.0 Rev 3
- ASCCANIF-832 Fixed known issue: CanIf does not compile if wakeup validation and transmit confirmation in polling mode is disabled but partial network support is enabled
- Introduce post build configuration data structures
- Made config parameters CanIfHthCanHandleTypeRef and CanIfHrhCanHandleTypeRef optional
- ASCCANIF-856 Fixed known issue: HandleId wizard does not work for configuration parameter CanIfTxPduId

Module version 6.1.1

2012-04-20

- ASCCANIF-814 Fixed known issue: Eclipse framework crashes during execution of the handle ID generator

Module version 6.1.0

2012-03-16

- Update of Dem reporting to the AUTOSAR 4.0 CanIf specification
- Remove dependency from Can_CheckWakeup() and CanTrcv_CheckWakeup() if the functions are not used
- Update naming scheme for #defines for symbolic name values to AUTOSAR R4.0 Rev 3 naming scheme
- Add support of partial networking filters for Tx PDUs according to AUTOSAR R4.0 Rev 3
- Add API and call back functions related to partial networking according to AUTOSAR R4.0 Rev 3

Module version 6.0.0

2011-09-02

- Initial AUTOSAR 4.0 version.

2.2. New features

- Support of PB variant for configparameter CanIfTrcvDrvCfg in CanIf
- Support of interception of Can messages through configured Tx and Rx Bus-Adapter callout functions.

2.3. Elektrobit-specific enhancements

This chapter lists the enhancements provided by the module.

- [HisCanIf0001] Transmit buffering (extension to AUTOSAR specification)

  Description:

  If no transmit buffers are configured, the corresponding code and data are removed via the C preprocessor.
- [HisCanIf0002, HisCanIf0004] Single controller support (extension to AUTOSAR specification)

  Description:

  The configuration and the code were optimized for the use case of only one CAN controller. Unnecessary configuration data is removed and macros are used, to allow the compiler to better optimize loops.

  Please note however, that the controller configuration is not pre-compile-time and therefore the controller configuration structure is also used in this case.
- [HisCanIf0003, HisCanIf0005] Single driver support (extension to AUTOSAR specification)

  Description:

  This implementation only supports one CAN driver. It contains no code in form of loops, branches, etc. to support several CAN drivers.

  Please note however, that the driver configuration is not pre-compile-time and therefore there is a driver configuration structure in the C code.
- CAN ID parameter support for user specific upper layer callbacks

  Description:

  This implementation supports to provide the CAN ID as parameter to the receive indication callback function of user specific upper layers.

  The vendor specific parameter `CanIfUserUpperLayerConfig/CanIfUpperLayerUseCanId` was added to configure this feature.
- DLC check callout function support

Description:

This implementation supports the configuration of up to two DLC check callout functions for each upper layer. The first function is called, if a DLC check fails while the other is called in the case, that the DLC check succeeds.

The vendor specific parameters `CanIfUserDlcErrorNotification` and `CanIfUserDlcPassedNotification` were added to the container `CanIfRxPduConfig` to configure the new callout functions.

- CAN ID translation callout support

  Description:

  This implementation supports the configuration of CAN ID translation callout functions. During run-time these functions can be used to change the CAN ID used on the CAN bus from the CAN ID that was originally configured (in parameter `CanIfTxPduCanId` and `CanIfRxPduCanId`).

  The vendor specific parameters `CanIfTranslateTxCanIdFunc` and `CanIfTranslateRxCanIdFunc` were added to the container `CanIfDispatchConfig` to configure the new callout functions.

- API and call back functions according to AUTOSAR 4.0 rev 3

  Description:

  The API functions `CanIf_ClearTrcvWufFlag`, `CanIf_CheckTrcvWakeFlag` and the call back functions `CanIf_ConfirmPnAvailability`, `CanIf_ClearTrcvWufFlagIndication`, `CanIf_CheckTrcvWakeFlagIndication` according to AUTOSAR 4.0 rev 3 were added.

- CanTrcv name mangling

  Description:

  This implementation supports to disable name mangling for header files and API functions of CanTrcv if a single CanTrcv driver is used.

  The vendor specific parameter `CanIfPublicCfg/CanIfSingleCanTrcvAPIInfixEnable` was added to configure this feature.

- Partial networking TX filter support (extension to AUTOSAR 4.2.2 specification)

  Description:

  According to AUTOSAR 4.2.2 a configured partial networking TX filter is enabled in the following cases: - at initialization - when CanIf_SetPduMode() is called with CANIF_TX_OFFLINE - when CanIf_SetControllerMode() is called with CANIF_CS_SLEEP The partial networking TX filter shall be deactivated after successful transmission of Wake-Up Frame. In case communication is ongoing and there is an successful reception of frame with PnTxFilter enabled, PnTxFilter shall be disabled. The PnTxFilter is in this case not needed since an Ack will be provided by an already active node.

Please note however, that the transmit requests for other PDUs will be rejected until the configured PDU was sent. Only the very first PDU which initiates the Wake-up of the Network has to be the CanIfTxPduPnFilterPdu.

- TX Offline Active mode support

  Description:

  The config parameter `CanIfTxOfflineActiveSupport` was introduced to determines whether TxOfflineActive feature is enabled or not. During CANIF_TX_OFFLINE_ACTIVE mode the upper layer has to handle the execution of the transmit confirmations immediately at the end of the transmit request.

- Support for decoupled processing of reception and transmission confirmation events

  Description:

  This implementation supports the configuration of decoupled processing of reception and transmission confirmation events on the MainFunction. This allows configuration of different PDUs to multiple CanIf_MainFunctionRx/Tx (with different period and priority) in order to limit the CPU load. During run-time these functions (CanIf_MainFunctionRx/Tx) can be used to process the reception/ transmission confirmation events received in interrupt context (CanIf_RxIndication/CanIf_TxConfirmation) using parameters `CanIfRxProcessing` and `CanIfTxProcessing`. The PDUs that are not assigned to a CanIf_MainFunctionRx/Tx() will be handled in interrupt context.

  The vendor specific parameter `CanIfDecoupledProcessingSupport` was added to enable this feature. The vendor specific containers `CanIfRxProcessing` and `CanIfTxProcessing` were added to the container `CanIfPublicCfg` to configure the new CanIf_MainFunctionRx and CanIf_MainFunctionTx functions. Each `CanIfRxProcessing` container contains the following parameters `CanIfRxPduQueueSize` and `CanIfPublicMaxPayloadQueueSize` which define the number of PDUs that can be added to the queue and the size of the data queue. Each `CanIfTxProcessing` container contains the parameter `CanIfTxPduQueueSize` which defines the number of PDUs that can be added to the queue. The vendor specific parameters `CanIfRxPduProcessingRef` and `CanIfTxPduProcessingRef` (added to the containers `CanIfRxProcessing` and `CanIfTxProcessing`)reference the PDUs that shall be processed on the MainFunction. When the queue is full, the incomming events will be ignored.

- MetaData support

  Description:

  Dynamic Tx and Rx PDUs are supported, where either parts or the whole CanId resides in the MetaData memory of the PDU which is managed by EcuC. A PDU is considered to have Metadata if a `MetaDataItem` of type `CAN_ID_32` is configured in EcuC. (see EcuC_Design.pdf)

  During transmission of a PDU with Metadata: CanIf will call `EcuC_GetMetaDataCanId()` (see EcuC_Design.pdf) in order to retrieve the Metadata information. `CanIfTxPduCanIdMask` defines which bits of the CanId reside in `CanIfTxPduCanId` and which are provided via Metadata. The resulting

CanId is calculated in the following way: (`CanIfTxPduCanId` & `CanIfTxPduCanIdMask`) | (Metadata & ~`CanIfTxPduCanIdMask`). For a PDU with Metadata, `CanIfTxPduCanId` can be omitted. This means that the whole CanId is expected to be provided via the Metadata information.

During reception of a PDU with Metadata: CanIf will perform a filtering which shall be done by comparing the incoming CanId with the stored `CanIfRxPduCanId` after applying the `CanIfRxPduCanIdMask` to both IDs. This filtering is applied on the stored `CanIfRxPduCanId` using linear search to get sure that the matching ID can be found if exist. If the filtering is successful, CanIf shall place the CanId in the `MetaDataItem` of type `CAN_ID_32` by calling `EcuC_SetMetaDataCanId()` (see EcuC_Design.pdf).

- Multiple CAN driver support with one driver

  Description:

  It is possible to enable `CanIfPublicMultipleDrvSupport` with only one CAN driver configured. In this case, all the functions in relation to the driver are build using the `vendorId` and `vendorApiInfix` from that CAN driver, just like it would with multiple drivers configured.

- Support for different AR revisions of Can MCAL modules

  Description:

  The config parameter `CanIfCanDriverCompatibility` was added to support compatibility with different AR revisions of Can MCAL modules.

- Support for Autosar HandleId concept for CDDs

  Description:

  The config parameter `CanIfUseCddHandleIds` was added to support Autosar concept of Cdd handle id usage. If `CanIfUseCddHandleIds` is enabled, for Pdus that have a Cdd as an upper layer, CanIf will use handle ids from the Cdd that references that Pdu (CDD/CddComStackContribution/CddComIfUpperLayerContribution/CddComIfUpperLayerRx(or Tx)Pdu/CddComIfHandleId). If `CanIfUseCddHandleIds` is disabled, handle ids will be manually set using CanIfTxPduSourcePduID for Tx Pdu or CanIfRxPduTargetPduID for Rx Pdus.

- Support for Custom Hook on Reception

  Description:

  The container `CanIfHookOnRxSupport` was added to support Custom Hook on Reception. If `CanIfHookOnReceptionSupport` is enabled, a custom function call configured by CanIfHookOnReceptionFunctionName is enabled, which provides the Can ID, the PDU information and Controller ID of a successful reception and all configuration parameters from container CanIfHookOnRxSupport are configurable. If `CanIfHookOnReceptionSupport` is disabled, reporting a successful reception to a CDD module is disabled.

- Support for Bus Mirroring

Description:

The container `CanIfMirroringSupport` was added to support Autosar Bus Mirroring concept. If `CanIfBusMirroringSupport` is enabled, frame mirroring support through the Mirror module is enabled and all configuration parameters from container CanIfMirroringSupport are configurable. If `CanIfBusMirroringSupport` is disabled, frame mirroring support through the Mirror module is disabled.

- Support for Security Event Reporting

Description:

The config parameter `CanIfEnableSecurityEventReporting` was added to support the Security Event Reporting concept. If `CanIfEnableSecurityEventReporting` is enabled, reporting of security events to IdsM is possible by configuring `CanIfSecurityEventRefs`. If `CanIfEnableSecurityEventReporting` is disabled, reporting of security events to IdsM is disabled.

- Software Filtering optimization

Description:

The Software Filtering can be disabled and consequently the related code is disabled in case all HRHs have parameter "CanIfHrhSoftwareFilter" set to OFF and no Rx Pdu has parameter "CanIfRxPduCanIdMask" enabled, i.e. the case where there is no need for Software Filtering.

The vendor specific parameter `CanIfSoftwareFilteringSupport` was added to disable Software filtering when it is not needed.

- Truncation options for a transmitted PDU

Description:

Truncation for TX PDUs is available as described in the AUTOSAR specification of CAN Interface, version R20-11.

In addition, an over-sized TX Pdu can be truncated either to the size of a CAN frame (8 bytes for CAN 2.-0, 64 bytes for CAN FD) or to the size of the configured length in EcuC. The option can be toggled based on the vendor specific parameter `CanIfTxPduTruncateToFrame`.

- Queue measurement support for decoupled processing

Description:

The decoupled processing of Rx and Tx Pdus can be measured by configuring the vendor specific parameters `CanIfRxDecoupledMeasurementSupport` and `CanIfTxDecoupledMeasurementSupport`. CanIf will forward to CDD the size of the enqueued Pdus in context of the MainFunction through the vendor specific parameters `CanIfNumberOfEnqueuedRxPdusApiName` and `CanIfNumberOfEnqueuedtxPdusApiName`.

In addition, another configurable call can be forwarded to the CDD module through the vendor specific parameters `CanIfNumberOfRxPdusExceedingQueueApiName` and `CanIfNumberOfTxPdusExceedingQueueApiName` in case no Pdu can be enqueued anymore.

- Shared Bus Mirroring buffer for all TX PDUs

  Description:

  The Bus Mirroring buffer now share the space allocated space among all TX PDUs, the size of the buffer based on the input parameter `canIfTxMirrorBufSize` can be less than the total size of all TX PDUs. If no free space is available for the requested PDU DET error 'CANIF_E_PDU_INSTANCE_LOST' will be reported.

- Support of PB variant for configparameter CanIfTrcvDrvCfg in CanIf

  Description:

  CanIf considers transceivers which are referenced by the Post-Build reference CanSMTransceiverId in CanSM as active. CanIf_CheckWakeup() only checks active CanTrcvs as possible wakeup sources.

- Support for Bus Adapter APIs in CanIf

  Description:

  The configuration parameters `CanIfBusATxIndication` and `CanIfBusARxIndication` were added. By enabling those parameters, the user is able to configure Bus Adapter APIs in CanIf. The data flow of `CanIf_Transmit()` and `CanIf_RxIndication()` will be conditioned by the return of configured callouts.

2.4. Deviations

This chapter lists the deviations of the module from the AUTOSAR standard.

- No inter-module consistency checks

  Description:

  The CanIf module does not perform the inter-module version checks as specified in the CanIf SWS.

  Rationale:

  The module consistency check is not within the responsibility of the basic software but part of the configuration management and delivery process.

  Requirements:

  CANIF021

- Hrh Range Container (`CanIfHrhRangeCfg`) is not used

  Description:

  The software receive range filter as configured by configuration parameter CanIfHrhRangeCfg is not supported.

  Rationale:

  Software filtering is done by mapping received CanIDs to configured Rx LPdus. Putting an extra software range filter in front does not add any functional benefit.

  Requirements:

  CANIF645, CANIF646, CANIF628_Conf, CANIF629_Conf, CANIF644_Conf, CANIF630_Conf

- No Debug and Trace support

  Description:

  CanIf is not instrumented for the usage with Debug and Trace.

  Requirements:

  CANIF565, CANIF566, CANIF567, CANIF568

- `CanIf_Init()` has no default configuration if NULL_PTR is passed

  Description:

  If a NULL_PTR is passed to function CanIf_Init() differs from the SWS:

  - Module PbCfgM used: Post-build-time configuration pointer is requested from PbCfgM module.
  - Module PbCfgM not used: Module initialization is aborted and CANIF_E_PARAM_POINTER is reported to Det_ReportErrorStatus() if CanIfDevErrorDetect is true.

  Requirements:

  CANIF301

- `CanIf_ChangeBaudrate` and `CanIf_CheckBaudrate` API functions are not supported (reference to product description: ASCPD-98)

  Description:

  `CanIf_ChangeBaudrate` and `CanIf_CheckBaudrate` API functions are not implemented. Instead this feature is implemented according to AUTOSAR R4.1 Rev 1. where a new function `CanIf_SetBaudrate` has been introduced while `CanIf_ChangeBaudrate` and `CanIf_CheckBaudrate` are set to deprecated.

Requirements:

CANIF775, CANIF786, CANIF778, CANIF779, CANIF780, CANIF776, CANIF787, CANIF782, CANIF783, CANIF784, CANIF785_Conf, CANIF294

- Only post-build configuration is supported

Description:

The CanIf module only supports configuration variant VARIANT-POST-BUILD. VARIANT-PRE-COMPILE and VARIANT-LINK-TIME are not supported.

Requirements:

CANIF460, CANIF461, CANIF377

- Only binary software filter algorithm is supported

Description:

The CanIf module only supports algorithm BINARY of configuration parameter CanIfPrivateSoftwareFilterType which configures the software filtering algorithm.

Requirements:

CANIF619_Conf

- No TTCan support

Description:

The CanIf implementation does not support TTCan.

Requirements:

CANIF675_Conf

- Parameter `CanIfTxPduDlc` is not used

Description:

The parameter `CanIfInitCfg/CanIfTxPduCfg/CanIfTxPduDlc` is not used. Instead, `CanIf_Transmit()` uses the DLC value given by the caller.

Requirements:

CANIF594_Conf

- Tx buffer handling according to AUTOSAR R4.0 Rev 2 (reference to product description: ASCPD-97)

Description:

If transmit buffering is enabled via parameter `CanIfPublicCfg/CanIfPublicTxBuffering`, the CanIf provides one buffer for each Tx PDU independent on the type of the associated HTH. I.e. the CanIf provides buffers for Tx PDUs assigned for BasicCAN as well as FullCAN transmission. Configuration parameter CanIfBufferSize is not used.

If a dynamic Tx PDU shall be buffered and the buffer for that PDU is already in use, the old message will only be overwritten, if the CAN ID of the new PDU is still the same as the one of the buffered message. If the CAN ID was changed in the meantime, the new message will be discarded and the old one remains in the buffer.

Rationale:

Resorting of pending Tx messages in case of an overwritten buffer is not implemented. Therefore overwriting buffers with messages that have a different CAN ID is rejected to prevent priority inversion in the CanIf Tx buffers. Also compare the requirement CANIF282 from previous AUTOSAR releases (e.g. R3.1).

Requirements:

CANIF837, CANIF833_Conf, CANIF834_Conf

- CanIf upper layer XCP must be configured as CDD module

Description:

The standard upper layer XCP is not contained in the enumeration range of the configuration parameters `CanIfRxPduUserRxIndicationUL` and `CanIfTxPduUserTxConfirmationUL`.

If this upper layer shall be used, configure it as complex device drivers (CDD).

Requirements:

CANIF556, CANIF555

- Symbolic names for CanIfCtrlIds and CanIfTrcvIds do not follow the AUTOSAR naming scheme

Description:

Some of the implemented symbolic name macros do not follow the AUTOSAR R4.0 Rev 3 naming scheme. AUTOSAR only specifies the inclusion of the name of direct parents in the symbolic macros but it is getting violated for CanIfCtrlIds and CanIfTrcvIds. CanIfCtrlIds: "CanIfConf_[CanIfCtrlDrvCfg]_[Controllername]" CanIfTrcvIds: "CanIfConf_[CanIfTrcvDrvCfg]_[CanIfTrcvCfg]"

Rationale:

CanIfCtrlIds short names are only distinct within the context of the superior CanIfCtrlDrv. CanIfTrcvIds short names are only distinct within the context of the superior CanIfTrcvDrv. The generation of SymbolicName macros as specified within the ECU configuration specification could lead to multiple macro redefinitions.

- Some configuration parameters have a lower config variant than specified

  Description:

  The following configuration parameters are specified to implement config variant link-time but actually implement config variant pre-compile:

  - CanIfDispatchUserCtrlBusOffName
  - CanIfDispatchUserCtrlModeIndicationName
  - CanIfDispatchUserCheckTrcvWakeFlagIndicationName
  - CanIfDispatchUserCheckTrcvWakeFlagIndicationUL
  - CanIfDispatchUserClearTrcvWufFlagIndicationName
  - CanIfDispatchUserClearTrcvWufFlagIndicationUL
  - CanIfDispatchUserConfirmPnAvailabilityName
  - CanIfDispatchUserConfirmPnAvailabilityUL
  - CanIfDispatchUserTrcvModeIndicationName
  - CanIfDispatchUserValidateWakeupEventName
  - CanIfDispatchUserCtrlBusOffUL
  - CanIfDispatchUserCtrlModeIndicationUL
  - CanIfDispatchUserValidateWakeupEventUL

  The following configuration parameters are specified to implement config variant post-build but actually implement config variant link-time:

  - CanIfRxPduUserRxIndicationUL
  - CanIfTxPduUserTxConfirmationUL
- When CanIfValidateWakeupOnStartedCtrlOnly is disabled, wakeup validation is processed in CANIF_CS_SLEEP, CANIF_CS_STOPPED, and CANIF_CS_STARTED.

  Description:

  According to AUTOSAR R4.0 Rev 3 and later revisions of the specification successful wakeup validation shall be stored only if the Can controller is in state CANIF_CS_STARTED. Instead the current implementation validates wakeup successfully if the following conditions apply when the configuration parameter CanIfValidateWakeupOnStartedCtrlOnly is disabled:

  - a) The Can controller is in state CANIF_CS_SLEEP.
  - b) The Can controller is in state CANIF_CS_STARTED or CANIF_CS_STOPPED and a wakeup has been previously detected.

Requirements:

CANIF286

- Transmit confirmations always forwarded to upper layer module

Description:

Transmit confirmations are always forwarded to the upper layer regardless of the PDU mode.

The CanIf SWS prohibits calling the transmit confirmation callback service of the upper layer module if the PDU mode is equal to CANIF_SET_OFFLINE or CANIF_SET_TX_OFFLINE.

Rationale:

This mechanism ensures that pending confirmations are not blocked. A confirmation may be pending, if a transmission of LPDU occured before PDU mode was set to CANIF_SET_OFFLINE or CANIF_SET_TX_OFFLINE. Confirmations for transmissions triggered in PDU mode CANIF_SET_OFFLINE or CANIF_SET_TX_OFFLINE cannot occur, because transmission is prohibited in this case. Unexpected confirmations which may be triggered because of an erroneous CAN driver (hardware) are not taken into account.

Requirements:

CANIF073, CANIF489

- API service `CanIf_ReadRxPduData` always accepted

Description:

API service `CanIf_ReadRxPduData` accepts a request even if the corresponding CCMSM is not equal to CANIF_CS_STARTED or the corresponding PDU channel mode is not equal to CANIF_SET_ONLINE or CANIF_SET_TX_ONLINE.

Rationale:

The AUTOSAR configuration schema allows multiple HRHs and therefore multiple CanIf controller assigned to a single Rx-LPdu. This would force to merge the status of multiple CCMSM and PDU channel modes and is not specified. In addition it would increase the configuration complexity drastically.

Requirements:

CANIF324

- Reception of L-PDU is not limited to a single CAN controller

Description:

CanIf allows the configuration of a Rx-PDU with an assignment to multiple HRHs with different underlying CAN controller. The assignment of a Tx-PDUs and its corresponding Tx-Buffer is limited to a single HTH only.

Rationale:

The limitation, as described in the requirement, would prevent the reception of PDUs assigned to HRHs of a different CAN controller.

Requirements:

CANIF046

- DET error `CANIF_E_STOPPED` not supported

Description:

CanIf does not support the DET error `CANIF_E_STOPPED`. In detail this means:

- `CanIf_Transmit` does not report `CANIF_E_STOPPED` if invoked in CCMSM `CANIF_CS_STOPPED` .
- `CanIf_Transmit` does not report `CANIF_E_STOPPED` if invoked in PDU channel mode `CANIF_OFFLINE` .

Rationale:

In case of a bus-off, CanIf switches the CCMSM to `CANIF_E_STOPPED` and informs CanSM about the bus-off. But informing the upper layers does not happen in an atomic fashion. This leads to a small time slot, where a transmission by the upper layers is a regular use-case. CanIf handles this scenario by rejecting the transmit request, but without reporting the DET error.

Requirements:

CANIF382, CANIF723

- CanIf calls `EcuM_ValidateWakeupEvent()` only if wake-up has been successfully validated

Description:

CanIf calls `EcuM_ValidateWakeupEvent()` within the context of `CanIf_CheckValidation()` only if wake-up has been successfully validated.

Rationale:

EcuM performs an action triggered by `EcuM_ValidateWakeupEvent()` no matter of the value of the parameter `sources`. According to Bugzilla (https://www.autosar.org/bugzilla/show_bug.cgi?id=57883), CanIf must NOT call `<User_ValidateWakeupEvent>(sources)` if wakeup has not been validated.

Requirements:

CANIF681

- CanIf does not support the API service `CanIf_TriggerTransmit()`.

Description:

CanIf does not support the API service `CanIf_TriggerTransmit()` introduced by AUTOSAR 4.2.1 along with CAN-FD.

Requirements:

SWS_CANIF_00883

- Due to "https://www.autosar.org/bugzilla/show_bug.cgi?id=52225" and "https://www.autosar.org/bugzilla/show_bug.cgi?id=61651" RfCs, CanIf_PduModeType was introduced, according with AUTOSAR 4.2.2 specification.

Description:

Due to "https://www.autosar.org/bugzilla/show_bug.cgi?id=52225" and "https://www.autosar.org/bugzilla/show_bug.cgi?id=61651" RfCs, CanIf_PduGetModeType and CanIf_PduSetModeType shall no longer be used. CanIf_PduModeType was introduced and shall be used accordingly: -CANIF_SET_TX_OFFLINE and CANIF_SET_TX_OFFLINE are not used anymore -CANIF_SET_TX_ONLINE and CANIF_SET_RX_ONLINE are not used anymore -CANIF_ONLINE shall be used instead of CANIF_SET_ONLINE and CANIF_GET_ONLINE -CANIF_TX_OFFLINE_ACTIVE shall be used instead of CANIF_SET_TX_OFFLINE_ACTIVE, CANIF_GET_OFFLINE_ACTIVE -CANIF_TX_OFFLINE shall be used instead of CANIF_SET_TX_OFFLINE -CANIF_GET_RX_ONLINE and CANIF_GET_TX_ONLINE are not used anymore -CANIF_OFFLINE shall be used instead of CANIF_GET_OFFLINE -CANIF_GET_OFFLINE_ACTIVE_RX_ONLINE is not used anymore PnTxFilter will be enabled if CanIf_SetControllerMode(controllerId, CANIF_CS_CLEEP) will be called. CanIf_PduModeType was introduced, according with AUTOSAR 4.2.2 specification.

Requirements:

CANIF490, CANIF491, CANIF492, SWS_CANIF_00073, SWS_CANIF_00483

- CanIf does not include the header Mirror_Cbk.h.

Description:

According to the SWS of the Mirror module the file does not exist. All the interfaces are exposed through the Mirror.h header.

Requirements:

SWS_CANIF_00903

- The signature of the CanIf_GetControllerMode API depends on the configuration

  Description:

  With the introduction of the BusMirroring feature (CONC_634), respecting the signature change of the API according to the AUTOSAR 4.4.0 SWS became mandatory.

  Requirements:

  CANIF_229
- The signature of the Tx Confirmation callback <User_TxConfirmation> depends on the configuration

  Description:

  With the introduction of the requirement SWS_CANIF_00739 for returning the result within the Tx Confirmation, respecting the signature change of the callback according to the AUTOSAR R20-11 SWS became mandatory.

  Requirements:

  CANIF011
- The signature of the CanIf_GetTrcvMode API depends on the configuration

  Description:

  With the introduction of the BusMirroring feature (CONC_634), respecting the signature change of the API according to the AUTOSAR 4.4.0 SWS became mandatory.

  Requirements:

  CANIF_288 SWS_CANIF_00288
- Wakeup event referenced in CanIfCtrlWakeupSourceOutRef shall be validated by CanIf with reception indications from that Controller

  Description:

  CanIfCtrlWakeupSourceOutRef can be configured even if that particular controller does not support wake-up and can be used for validation (e.g. the transceiver supports wake-up but the controller does not). The validation will be done on a reception on that controller.

  Rationale:

  Even though starting with CanIf ASR 4.0.3 CanIfCtrlWakeupSourceOutRef was not used anymore, it will be used to allow transceiver wake-up events to be validated through a controller that does not support wake-up.
- The boundary value of 127 for TxErrorCounter and RxErrorCounter should be taken in consideration.

Description:

Requirement SWS_CANIF_00917 for reporting of the security event CANIF_SEV_ERRORS-TATE_PASSIVE don't take in consideration boundary value of 127 for TxErrorCounter and RxErrorCounter. This requirement will be deviated as follows : – TxErrorCounter greater than 127 AND RxErrorCounter smaller than 127(0x0) – TxErrorCounter greater than 127 AND RxErrorCounter smaller or equal to 127 (0x1) – RxErrorCounter greater than 127 AND TxErrorCounter smaller or equal to 127 (0x2)

Requirements:

SWS_CANIF_00917

2.5. Limitations

This chapter lists the limitations of the module. Refer to the module references chapter *Integration notes*, subsection *Integration requirements* for requirements on integrating this module.

- No support of TTCan (reference to product description: ASCPD-3)

  Description:

  TTCan controllers are not supported by this implementation of CanIf.

- Number of supported CAN controllers

  Description:

  The CanIf only supports configurations with CAN controllers that have a controller ID less than 255.

  Rationale:

  The number 255 is used by the CanIf to classify a controller ID as invalid.

- Number of supported CAN Transceiver Drivers

  Description:

  Up to 255 CAN Transceiver Drivers are supported.

  Rationale:

  This restriction allows to use smaller types (uint8 instead of uint16) for internal calculations.

- No support for multiple configurations

  Description:

  Only one configuration is supported. Multiple configurations are not allowed.

- Supported software filtering mechanisms

  Description:

  Binary search is the only supported software filtering method.

- Only one upper layer callback function is supported (reference to product description: ASCPD-2)

  Description:

  Only one RX indication and one TX confirmation callback function is supported for each upper layer.

- Code optimizations may result in compiler warnings

  Description:

  The `CanIf` implicitly uses code optimizations for certain configurations. This may, however, lead to compiler warnings such as "if-statement always evaluates to true".

  Rationale:

  Excluding all potentially redundant control flow statements would clutter code.

- Reconfiguration of CAN identifier limited for Flexible Data-Rate

  Description:

  The reconfiguration of a CAN identifier via the API service `CanIf_SetDynamicTxId` is restricted to a separate range for CAN 2.0 and CAN Flexible Data-Rate messages.

  A Tx-PDU with a CAN 2.0 identifier (`STANDARD_CAN` or `EXTENDED_CAN`) can not be reconfigured to a CAN Flexible Data-Rate identifier (`STANDARD_FD_CAN` or `EXTENDED_FD_CAN`) and vice versa.

  Rationale:

  The `CanIf` reserves a transmit buffer of 8 bytes for CAN 2.0 messages. A reconfiguration to a CAN Flexible Data-Rate message offers the possibility to send up to 64 bytes without having an underlying buffer of sufficient size.

- ASR versions of CAN drivers with Multiple CAN driver support enabled

  Description:

  When Multiple CAN driver support is enabled, the configured CAN drivers must have the same AUTOSAR SW release version and must be one which is supported by the CanIf module.

  Rationale:

  The behavior in relationship to the CAN drivers is treated in an unitary manner.

- J1939 PDUs mapping to different cores not supported

Description:

All J1939 PDUs must be mapped on the same core (the same one on which the J1939 stack is located) if the CAN stack is multicore distributed along network boundaries.

- Security event reporting and Multiple Driver support not supported concomitant

  Description:

  CanIfEnableSecurityEventReporting enabled and CanIfPublicMultipleDrvSupport enabled with CanIfCanDriverCompatibility set to 402 or 403 is not supported.

- Security event reporting and Multicore supported concomitant

  Description:

  CanIfEnableSecurityEventReporting enabled and Multicore enabled is not supported.

2.6. Open-source software

CanIf does not use open-source software.