User Manual

for S32M27X CANTRCV Driver

Document Number: UM34CANTRCVASRR21-11 Rev0000R3.0.0 Rev. 1.0

Chapter 1

Revision History

Revision	Date	Author	Description
1.0	31.03.2023	NXP RTD Team	S32K3 Real-Time Drivers AUTOSAR 4.4 & R21-11 Version 3.0.0

Chapter 2

Introduction

This User Manual describes NXP Semiconductors' AUTOSAR CanTrcv Driver for S32M27x.

AUTOSAR CanTrcv Driver configuration parameters description can be found in the Tresos Configuration Plugin section. Deviations from the specification are described in the Deviations from Requirements section.

AUTOSAR CanTrcv driver requirements and APIs are described in the CanTrcv Driver Software Specification Document (version R21-11).

2.1 Supported Derivatives

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

- s32m274_lqfp64
- s32m276_lqfp64

All of the above microcontroller devices are collectively named as S32K3.

2.2 Overview

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

AUTOSAR:

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

2.3 About This Manual

This Technical Reference employs the following typographical conventions:

- **Boldface** style: Used for important terms, notes and warnings.
- *Italic* style: Used for code snippets in the text. Note that C language modifiers such "const" or "volatile" are sometimes omitted to improve readability of the presented code.

Notes and warnings are shown as below:

Note

This is a note.

Warning

This is a warning

2.4 Acronyms and Definitions

Term	Definition
API	Application Programming Interface
AUTOSAR	Automotive Open System Architecture
DEM	Diagnostic Event Manager
DET	Default Error Tracer
MCU	Micro controller Unit
N/A	Not Available
CANTRCV	Can Transceiver
AE	Application Extension
CAN	Controller Area Network

- The term "CanTrcv Driver" is related to the software handling the CANPHY channel.
- The term "Application" is used for the software utilizing the CanTrcv Driver.

2.5 Reference List

#	Title	Version
1	Specification of Can Transceiver Driver	AUTOSAR Release R21-11
2	Reference Manual	S32M27x Reference Manual, Rev.2, Draft A, — 02/2023
3	Data Sheet	S32M2xx Data Sheet, Rev. 2 RC — 12/2022

Chapter 3

Driver

3.1 Requirements

Requirements for this driver are detailed in the AUTOSAR R21-11 CanTrcv Driver Software Specification document (See Table Reference List)

3.2 Driver Design Summary

The CanTrcv Driver controls the CANPHY in AE of the S32M27x devices. It provides the following features:

- Configuration and initialization of the CANPHY.
- Support Normal and Standby modes.
- Support Wake Up Pattern (WUP) in Standby mode.

3.3 Hardware Resources

Driver has one CANPHY channel in AE which is a companion die that is assembled along with an MCU die.

Note

The user must select properly the chip which supports CanPHY.

3.4 Deviations from Requirements

The driver deviates from the AUTOSAR CANTRCV Driver software specification in some places. The table below identifies the AUTOSAR requirements that are not implemented or out of scope for the CANTRCV Driver.

Term	Definition
N/S	Out of scope
N/I	Not implemented
N/F	Not fully implemented

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

Requirement	Status	Description	Notes
SWS_CanTrcv_00230	N/S	The CAN Tranceiver Driver shall use the Time service Tm↩ _BusyWait1us16bit to realize the wait time for transceiver state changes.	Applicable only for platforms which need timeout monitors
SWS_CanTrcv_00174	N/S	If selective wakeup is supported by the transceiver hardware, it shall be indicated with the configuration parameter CanTrcv↩ HwPnSupport.	Applicable only for platforms which support Selective wakeup feature
SWS_CanTrcv_00175	N/S	The configuration container for selective wakeup functionality (CanTrcvPartialNetwork) and for the following APIs: - 8.4.7 CanTrcv_GetTrcvSystemData, - 8.4.8 CanTrcv_ClearTrcv↩ WufFlag, - 8.4.9 CanTrcv_↩ ReadTrcvTimeoutFlag, - 8.4.10 CanTrcv_ClearTrcvTimeout↩ Flag and - 8.4.11 CanTrcv_↩ ReadTrcvSilenceFlag shall exist only if CanTrcvHwPnSupport = TRUE.	Applicable only for platforms which support Selective wakeup feature

Requirement	Status	Description	Notes
SWS_CanTrcv_00177	N/S	If selective wakeup is supported, CAN transceivers shall be configured to wake up on a particular CAN frame or a group of CAN frames using the parameters CanTrcvPnFrame↩ CanId, CanTrcvPnFrameCan↩ IdMask and CanTrcvPnFrame↩ DataMask.	Applicable only for platforms which support Selective wakeup feature
SWS_CanTrcv_00181	N/S	If selective wakeup is enabled and supported by hardware↩ : POR and SYSERR flags of the transceiver status shall be checked by CanTrcv_Init API.	Applicable only for platforms which support Selective wakeup feature
SWS_CanTrcv_00182	N/S	If the POR flag or SYSERR flag is set, transceiver shall be reconfigured for selective wakeup functionality by running the configuration sequence. If the POR flag or SYSERR flag is not set, the configuration stored in the transceiver memory will be still valid and re-configuration is not necessary.	Applicable only for platforms which support Selective wakeup feature
SWS_CanTrcv_00183	N/I	If the POR flag is set, wakeup shall be reported to EcuM through API EcuM_Set↩ WakeupEvent with a wakeup source value, which has a "1" at the bit position according to the symbolic name value referred by CanTrcvPorWakeupsourceRef, and "0" on all others.	Not implemented yet
SWS_CanTrcv_00184	N/I	If the SYSERR flag is set, wakeup shall be reported to EcuM through API EcuM_↩ SetWakeupEvent with a wakeup source value, which has a "1" at the bit position according to the symbolic name value referred by CanTrcvSyserr↩ WakeupSourceRef, and "0" on all others.	Not implemented yet
SWS_CanTrcv_00168	N/I	If development error detection is enabled for CanTrcv module↩ : the function CanTrcv_Init shall raise the development error CANTRCV_E_BAUDR↩ ATE_NOT_SUPPORTED, if the configured baud rate is not supported by the transceiver.	Not implemented yet

Requirement	Status	Description	Notes
SWS_CanTrcv_00226	N/S	In order to implement the AUTOSAR Partial Networking mechanism CAN transceivers shall support the definition of a data mask for the Wake Up Frame (the configuration structure of CanTrcvPnFrameDataMask is mandatory).	Applicable only for platforms which support Selective wakeup feature
SWS_CanTrcv_00186	N/S	If selective wakeup is supported by hardware: the flags POR and SYSERR of the transceiver status shall be checked by Can_Trcv_SetOpMode API.	Applicable only for platforms which support Selective wakeup feature
SWS_CanTrcv_00187	N/S	If the POR flag is set, transceiver shall be re-initialized to run the transceiver's configuration sequence.	Applicable only for platforms which support Selective wakeup feature
SWS_CanTrcv_00188	N/S	If the SYSERR flag is NOT set and the requested mode is CANTRCV_NORMAL, transceiver shall call the API CanIf_ConfirmPnAvailability() for the corresponding abstract CanIf TransceiverId. CanIf_ConfirmPnAvailability informs CanNm (through CanIf and CanSm) that selective wakeup is enabled.	Applicable only for platforms which support Selective wakeup feature
SWS_CanTrcv_00189	N/S	The function CanTrcv_GetTrcvSystemData shall read the configuration/status of the CAN transceiver and store the read data in the out parameter TrcvSysData. If this is successful, E_OK shall be returned. Hint: This API can be invoked through diagnostic services or during initialization to determine the transceiver status and its availability. Note: Currently an agreement on the parameter set for the transceiver HW specification has not been reached. For this reason, the diagnostic data is now returned as a uint32 (as stored in the transceiver registers). When a definitive and standard parameter set is defined, a data structure may be defined for abstracting the diagnostic data.	Applicable only for platforms which support Selective wakeup feature

Requirement	Status	Description	Notes
SWS_CanTrcv_00190	N/S	If there is no/incorrect communication to the transceiver, the function CanTrcv_GetTrcv↩ SystemData shall report the runtime error code CANTRC↩ V_E_NO_TRCV_CONTROL to the default Error Tracer and return E_NOT_OK.	Applicable only for platforms which support Selective wakeup feature
SWS_CanTrcv_00191	N/S	If development error detection is enabled for the CanTrcv module: if called before the CanTrcv has been initialized, the function CanTrcv_Get↩ TrcvSystemData shall raise development error CANTRCV_↩ E_UNINIT otherwise (if DET is disabled) return E_NOT_OK.	Applicable only for platforms which support Selective wakeup feature
SWS_CanTrcv_00192	N/S	If development error detection is enabled for the CanTrcv module: if called with an invalid transceiver ID for parameter Transceiver, function Can↩ Trcv_GetTrcvSystemData shall raise the development error CANTRCV_E_INVALID↩ _TRANSCEIVER otherwise (if DET is disabled) return E_NOT_OK.	Applicable only for platforms which support Selective wakeup feature
SWS_CanTrcv_00193	N/S	If development error detection is enabled for the CanTrcv module: if called with N↩ ULL pointer for parameter TrcvSysData, function Can↩ Trcv_GetTrcvSystemData shall raise the development error CANTRCV_E_PARAM_P↩ OINTER otherwise (if DET is disabled) return E_NOT_OK.	Applicable only for platforms which support Selective wakeup feature

Requirement	Status	Description	Notes
SWS_CanTrcv_00194	N/S	The function CanTrcv_↩ ClearTrcvWufFlag shall clear the wakeup flag in the C↩ AN transceiver. If successful, E_OK shall be returned. Implementation Hints: This API shall be used by the CanSM module for ensuring that no frame wakeup event is lost, during entering a low-power mode. This API clears the WUF flag. The CAN transceiver shall be shall be put into Standby mode (CANTRCV_STANDBY) after clearing of the WUF flag. If a system error (SYSERR, e.g. configuration error) occurs while selective wakeup functionality is being enabled, transceiver will disable the functionality. Transceiver will wake up on the next CAN wake pattern (WUP). In case of any other hardware error (e.g. frame detection error), transceiver will wake up if the error counter inside the transceiver overflows.	Applicable only for platforms which support Selective wakeup feature
SWS_CanTrcv_00195	N/S	CanTrcv shall inform Can↩ If that the wakeup flag has been cleared for the requested Transceiver, through the callback notification CanIf↩ _ClearTrcvWufFlagIndication referring to the corresponding CAN transceiver with the abstract CanIf TransceiverId.	Applicable only for platforms which support Selective wakeup feature
SWS_CanTrcv_00196	N/S	If there is no/incorrect communication to the transceiver, the function CanTrcv_ClearTrcv↩ WufFlag shall report the runtime error CANTRCV_E_N↩ O_TRCV_CONTROL to the Default Error Tracer and return E_NOT_OK.	Applicable only for platforms which support Selective wakeup feature

Requirement	Status	Description	Notes
SWS_CanTrcv_00197	N/S	If development error detection is enabled for the CanTrcv module: if called before the CanTrcv has been initialized, the function CanTrcv_Clear↩ TrcvWufFlag shall raise development error CANTRCV_E↩ _UNINIT otherwise (if DET is disabled) return E_NOT_OK.	Applicable only for platforms which support Selective wakeup feature
SWS_CanTrcv_00198	N/S	If development error detection is enabled for the Can↩ Trcv module: if called with an invalid transceiver ID for parameter Transceiver, function CanTrcv_ClearTrcvWuf↩ Flag shall raise the development error CANTRCV_E_INVAL↩ ID_TRANSCEIVER otherwise (if DET is disabled) return E↩ _NOT_OK.	Applicable only for platforms which support Selective wakeup feature
SWS_CanTrcv_00199	N/S	If development error detection is enabled for the module Can↩ Trcv: If called with an invalid transceiver ID Transceiver, the function CanTrcv_↩ ReadTrcvTimeoutFlag shall raise the development error CANTRCV_E_INVALID↩ _TRANSCEIVER otherwise (if DET is disabled) return E_NOT_OK.	Applicable only for platforms which support Selective wakeup feature
SWS_CanTrcv_00200	N/S	If development error detection is enabled for the module Can↩ Trcv: If called with FlagState = NULL, the function CanTrcv↩ _ReadTrcvTimeoutFlag shall raise the development error CANTRCV_E_PARAM_P↩ OINTER otherwise (if DET is disabled) return E_NOT_OK.	Applicable only for platforms which support Selective wakeup feature
SWS_CanTrcv_00201	N/S	If development error detection is enabled for the module Can↩ Trcv: If called with an invalid transceiver ID Transceiver, the function CanTrcv_↩ ClearTrcvTimeoutFlag shall raise the development error CANTRCV_E_INVALID↩ _TRANSCEIVER otherwise (if DET is disabled) return E_NOT_OK.	Applicable only for platforms which support Selective wakeup feature

Requirement	Status	Description	Notes
SWS_CanTrcv_00202	N/S	If development error detection is enabled for the module Can↩ Trcv: If called with an invalid transceiver ID Transceiver, the function CanTrcv_↩ ReadTrcvSilenceFlag shall raise the development error CANTRCV_E_INVALID↩ _TRANSCEIVER otherwise (if DET is disabled) return E_NOT_OK.	Applicable only for platforms which support Selective wakeup feature
SWS_CanTrcv_00203	N/S	If development error detection is enabled for the module Can↩ Trcv: If called with FlagState = NULL, the function CanTrcv↩ _ReadTrcvSilenceFlag shall raise the development error CANTRCV_E_PARAM_P↩ OINTER otherwise (if DET is disabled) return E_NOT_OK.	Applicable only for platforms which support Selective wakeup feature
SWS_CanTrcv_00220	N/S	If development error detection for the module CanTrcv is enabled: If called before the CanTrcv module has been initialized, the function Can↩ Trcv_SetPNActivationState shall raise the development error CANTRCV_E_UNINIT otherwise (if DET is disabled) return E_NOT_OK.	Applicable only for platforms which support Selective wakeup feature
SWS_CanTrcv_00221	N/S	CanTrcv shall enable the PN wakeup functionality when function CanTrcv_SetPN↩ ActivationState is called with ActivationState= PN_ENAB↩ LED and return E_OK.	Applicable only for platforms which support Selective wakeup feature
SWS_CanTrcv_00222	N/S	CanTrcv shall disable the PN wakeup functionality when function CanTrcv_SetPN↩ ActivationState is called with ActivationState= PN_DISA↩ BLED and return E_OK.	Applicable only for platforms which support Selective wakeup feature
SWS_CanTrcv_00231	N/S	The Can Transceiver Driver module shall reject configurations with partition mappings which are not supported by the implementation.	Applicable only for platforms which support Multicore feature

Requirement	Status	Description	Notes
SWS_CanTrcv_00233	N/S	The ECUC partitions referenced by CanTrcvChannelEcuc↩ PartitionRef shall be a subset of the ECUC partitions referenced by CanTrcvEcucPartitionRef.	Applicable only for platforms which support Multicore feature
SWS_CanTrcv_CONSTR_↩ 00235	N/S	If CanTrcvEcucPartitionRef references one or more ECUC partitions, CanTrcvChannelEcuc↩ PartitionRef shall have a multiplicity of one and reference one of these ECUC partitions as well.	Applicable only for platforms which support Multicore feature
SWS_CanTrcv_00234	N/S	CanTrcvChannel and Can↩ Controller of one communication channel shall all reference the same ECUC partition.	Applicable only for platforms which support Multicore feature. It's responsibility of user to configure Can and CanTrcv to refer to the same partition
SWS_CanTrcv_00216	N/S	Service Name: CanTrcv↩ _ClearTrcvTimeoutFlag Syntax: Std_ReturnType CanTrcv_ClearTrcvTimeout↩ Flag (uint8 Transceiver) Service ID [hex]: 0x0c Sync/Async: Synchronous Reentrancy: Non Reentrant Parameters (in): Transceiver: CAN transceiver ID. Parameters (inout): None Parameters (out): None Return value: Std_ReturnType: E_OK: Will be returned, if the timeout flag is successfully cleared. E_N↩ OT_OK: Will be returned, if the timeout flag could not be cleared. Description: Clears the status of the timeout flag in the transceiver hardware. This API shall exist only if CanTrcvHw↩ PnSupport = TRUE. Available via: CanTrcv.h	Applicable only for platforms which support Selective wakeup feature

Requirement	Status	Description	Notes
SWS_CanTrcv_00214	N/S	Service Name: CanTrcv_↩ ClearTrcvWufFlag Syntax: Std_ReturnType CanTrcv↩ _ClearTrcvWufFlag (uint8 Transceiver) Service ID [hex]: 0x0a Sync/Async: Synchronous Reentrancy: Reentrant for different transceivers Parameters (in): Transceiver: CAN Transceiver ID. Parameters (inout): None Parameters (out): None Return value: Std_ReturnType: E_OK: will be returned if the WUF flag has been cleared. E_NOT_OK: will be returned if the WUF flag has not been cleared or a development error occurs. Description: Clears the WUF flag in the transceiver hardware. This API shall exist only if CanTrcvHwPnSupport = TR↩ UE. Available via: CanTrcv.h	Applicable only for platforms which support Selective wakeup feature
SWS_CanTrcv_00213	N/S	Service Name: CanTrcv_↩ GetTrcvSystemData Syntax: Std_ReturnType CanTrcv↩ _GetTrcvSystemData (uint8 Transceiver, uint32* Trcv↩ SysData) Service ID [hex]: 0x09 Sync/Async: Synchronous Reentrancy: Non Reentrant Parameters (in): Transceiver: CAN transceiver ID. Parameters (inout): None Parameters (out): TrcvSysData: Configuration/Status data of the transceiver. Return value: Std_ReturnType: E_OK: will be returned if the transceiver status is successfully read. E↩ _NOT_OK: will be returned if the transceiver status data is not available or a development error occurs. Description: Reads the transceiver configuration/status data and returns it through parameter TrcvSysData. This API shall exist only if Can↩ TrcvHwPnSupport = TRUE. Available via: CanTrcv.h	Applicable only for platforms which support Selective wakeup feature

Requirement	Status	Description	Notes
SWS_CanTrcv_00217	N/S	Service Name: CanTrcv_↩ ReadTrcvSilenceFlag Syntax: Std_ReturnType CanTrcv↩ _ReadTrcvSilenceFlag (uint8 Transceiver, CanTrcv_Trcv↩ FlagStateType* FlagState) Service ID [hex]: 0x0d Sync/Async: Synchronous Reentrancy: Non Reentrant Parameters (in): Transceiver: CAN transceiver ID. Parameters (inout): None Parameters (out): FlagState: State of the silence flag. Return value↩ : Std_ReturnType: E_OK: Will be returned, if status of the silence flag is success-fully read. E_NOT_OK: Will be returned, if status of the silence flag could not be read. Description: Reads the status of the silence flag from the transceiver hardware. This API shall exist only if CanTrcvHw↩ PnSupport = TRUE. Available via: CanTrcv.h	Applicable only for platforms which support Selective wakeup feature
SWS_CanTrcv_00215	N/S	Service Name: CanTrcv↩ _ReadTrcvTimeoutFlag Syntax: Std_ReturnType CanTrcv_ReadTrcvTimeout↩ Flag (uint8 Transceiver, CanTrcv_TrcvFlagStateType* FlagState) Service ID [hex]: 0x0b Sync/Async: Synchronous Reentrancy: Non Reentrant Parameters (in): Transceiver: CAN transceiver ID. Parameters (inout): None Parameters (out): FlagState: State of the timeout flag. Return value: Std_ReturnType: E_OK: Will be returned, if status of the timeout flag is success-fully read. E_NOT_OK: Will be returned, if status of the timeout flag could not be read. Description: Reads the status of the timeout flag from the transceiver hardware. This API shall exist only if CanTrcvHw↩ PnSupport = TRUE. Available via: CanTrcv.h	Applicable only for platforms which support Selective wakeup feature

Requirement	Status	Description	Notes
SWS_CanTrcv_00219	N/S	Service Name: CanTrcv_↩ SetPNActivationState Syntax: Std_ReturnType CanTrcv↩ _SetPNActivationState (CanTrcv_PNActivationType ActivationState) Service ID [hex]: 0x0f Sync/Async: Synchronous Reentrancy: Non Reentrant Parameters (in): ActivationState: PN_ENAB↩ LED: PN wakeup functionality in CanTrcv shall be enabled. PN_DIABLED: PN wakeup functionality in CanTrcv shall be disabled. Parameters (inout): None Parameters (out): None Return value: Std_ReturnType: E_OK: Will be returned, if the PN has been changed to the requested configuration. E_NOT_OK: Will be returned, if the PN configuration change has failed. The previous configuration has not been changed. Description: The API configures the wake-up of the transceiver for Standby and Sleep Mode: Either the CAN transceiver is woken up by a remote wake-up pattern (standard CAN wake-up) or by the configured remote wake-up frame. Available via: CanTrcv.h	Applicable only for platforms which support Selective wakeup feature
SWS_CanTrcv_00210	N/S	Name: CanTrcv_PN↩ ActivationType Kind: Enumeration Range: PN_ENABLED: –: PN wakeup functionality in CanTrcv is enabled. PN↩ _DISABLED: –: PN wakeup functionality in CanTrcv is disabled. Description: Datatype used for describing whether PN wakeup functionality in CanTrcv is enabled or disabled. Available via: CanTrcv.h	Applicable only for platforms which support Selective wakeup feature

Requirement	Status	Description	Notes
SWS_CanTrcv_00211	N/S	Name: CanTrcv_TrcvFlag↩ StateType Kind: Enumeration Range: CANTRCV_F↩ LAG_SET: –: The flag is set in the transceiver hardware. CANTRCV_FLAG_↩ CLEARED: –: The flag is cleared in the transceiver hardware. Description: Provides the state of a flag in the transceiver hardware. Available via: Can↩ Trcv.h	Applicable only for platforms which support Selective wakeup feature
SWS_CanTrcv_00163	N/S	Name: CanTrcv_TrcvMode↩ Type Kind: Enumeration Range: CANTRCV_TRCV↩ MODE_SLEEP: –: Transceiver mode SLEEP CANTRCV_↩ TRCVMODE_STANDBY: –: Transceiver mode STANDBY CANTRCV_TRCVMODE_↩ NORMAL: 0x00: Transceiver mode NORMAL Description: Operating modes of the CAN Transceiver Driver. Available via: Can_GeneralTypes.h	Implemented in Base module
SWS_CanTrcv_00164	N/S	Name: CanTrcv_Trcv↩ WakeupModeType Kind: Enumeration Range: CANT↩ RCV_WUMODE_ENABLE: 0x00: The notification for wakeup events is enabled on the addressed transceiver. CANTRCV_WUMODE_DI↩ SABLE: 0x01: The notification for wakeup events is disabled on the addressed transceiver. CANTRCV_WUMODE_C↩ LEAR: 0x02: A stored wakeup event is cleared on the addressed transceiver. Description: This type shall be used to control the CAN transceiver concerning wake up events and wake up notifications. Available via: Can_GeneralTypes.h	Implemented in Base module

Requirement	Status	Description	Notes
SWS_CanTrcv_00165	N/S	Name: CanTrcv_Trcv↩ WakeupReasonType Kind: Enumeration Range: CAN↩ TRCV_WU_ERROR: 0x00: Due to an error wake up reason was not detected. This value may only be reported when error was reported to DEM before. CANTRCV_WU_N↩ OT_SUPPORTED: 0x01: The transceiver does not support any information for the wake up reason. CANTRCV_↩ WU_BY_BUS: 0x02: The transceiver has detected, that the network has caused the wake up of the ECU. CANTRCV↩ _WU_INTERNALLY: 0x03: The transceiver has detected, that the network has woken up by the ECU via a request to NORMAL mode. CAN↩ TRCV_WU_RESET: 0x04: The transceiver has detected, that the "wake up" is due to an ECU reset. CANTRC↩ V_WU_POWER_ON: 0x05: The transceiver has detected, that the "wake up" is due to an ECU reset after power on. CANTRCV_WU_BY_PIN: 0x06: The transceiver has detected a wake-up event at one of the transceiver's pins (not at the CAN bus). CANTRC↩ V_WU_BY_SYSERR: 0x07: The transceiver has detected, that the wake up of the ECU was caused by a HW related device failure. Description: This type denotes the wake up reason detected by the CAN transceiver in detail. Available via: Can_GeneralTypes.h	Implemented in Base module

Requirement	Status	Description	Notes
ECUC_CanTrcv_00097	N/S	Name: CanTrcvControls↩ PowerSupply Parent Container: CanTrcvChannel Description: Is ECU power supply controlled by this transceiver? TRUE = Controlled by transceiver. FALSE = Not controlled by transceiver. Multiplicity: 1 Type: Ecuc↩ BooleanParamDef Default value: false Post-Build Variant Value: false Value Configuration Class: Pre-compile time: X: All Variants Link time: – Post-build time: – Scope / Dependency: scope: local	The CanTrcv Drvier will not control Power Supply as it's done in other Driver
ECUC_CanTrcv_00160	N/S	Name: CanTrcvHwPnSupport Parent Container: CanTrcv↩ Channel Description: Indicates whether the HW supports the selective wake-up function: T↩ RUE = Selective wakeup feature is supported by the transceiver FALSE = Selective wakeup functionality is not available in transceiver Multiplicity: 1 Type: EcucBooleanParamDef Default value: false Post-Build Variant Value: false Value Configuration Class: Pre-compile time: X: All Variants Link time: – Post-build time: – Scope / Dependency: scope: local↩ : dependency: CanTrcvWake↩ UpSupport	Applicable only for platforms which support Selective wakeup feature

Requirement	Status	Description	Notes
ECUC_CanTrcv_00194	N/S	Name: CanTrcvChannelEcuc↩ PartitionRef Parent Container: CanTrcvChannel Description↩ : Maps the CAN transceiver channel to zero or one EC↩ UC partitions. The ECUC partition referenced is a subset of the ECUC partitions where the CAN transceiver driver is mapped to. Multiplicity: 0..1 Type: Reference to [Ecuc↩ Partition] Post-Build Variant Multiplicity: true Post-Build Variant Value: true Multiplicity Configuration Class: Pre-compile time: X: All Variants Link time: – Post-build time: – Value Configuration Class: Pre-compile time: X: All Variants Link time: – Post-build time: – Scope / Dependency: scope: ECU	Applicable only for platforms which support multicore feature
ECUC_CanTrcv_00181	N/I	Name: CanTrcvPorWakeup↩ SourceRef Parent Container: CanTrcvChannel Description: Symbolic name reference to specify the wakeup sources that should be used in the calls to EcuM_SetWakeup↩ Event as specified in [S↩ WS_CanTrcv_00183] and [SWS_CanTrcv_00184].: This reference is mandatory if the HW supports POR or SYSERR flags Multiplicity: 0..1 Type: Symbolic name reference to [EcuMWakeupSource] Post-↩ Build Variant Multiplicity: false Post-Build Variant Value: false Multiplicity Configuration Class: Pre-compile time: X: All Variants Link time: – Post-build time: – Value Configuration Class: Pre-compile time: X: All Variants Link time: – Post-build time: – Scope / Dependency: scope: ECU	Not implemented yet

Requirement	Status	Description	Notes
ECUC_CanTrcv_00182	N/I	Name: CanTrcvSyserr↩ WakeupSourceRef Parent Container: CanTrcvChannel Description: Symbolic name reference to specify the wakeup sources that should be used in the calls to EcuM_Set↩ WakeupEvent as specified in [SWS_CanTrcv_00183] and [SWS_CanTrcv_00184]: This reference is mandatory if the HW supports POR or SYSERR flags Multiplicity: 0..1 Type: Symbolic name reference to [EcuMWakeupSource] Post-↩ Build Variant Multiplicity: false Post-Build Variant Value: false Multiplicity Configuration Class: Pre-compile time: X: All Variants Link time: – Post-build time: – Value Configuration Class: Pre-compile time: X: All Variants Link time: – Post-build time: – Scope / Dependency: scope: ECU	Not implemented yet

Requirement	Status	Description	Notes
ECUC_CanTrcv_00174	N/S	Name: CanTrcvSPI↩ CommTimeout Parent Container: CanTrcvConfigSet Description: Indicates the maximum time allowed to the CanTrcv for replying (either positively or negatively) to a SPI command. Timeout is configured in milliseconds. Timeout value of '0' means that no specific timeout is to be used by CanTrcv and the communication is executed at the best of the SPI HW capacity. Multiplicity: 1 Type: EcucIntegerParamDef Range: 0 .. 100 Default value: 0 Post-Build Variant Value↩ : true Value Configuration Class: Pre-compile time: X: VARIANT-PRE-COMPILE Link time: X: VARIANT-↩ LINK-TIME Post-build time: X: VARIANT-POST-BUILD Scope / Dependency: scope: local: dependency: This parameter exists only if atleast one SPI Sequence is referenced in CanTrcvSpiSequence.	This configuration is done indirectly in Ae Driver
ECUC_CanTrcv_00145	N/S	Container Name: CanTrcvDio↩ Access Parent Container: Can↩ TrcvAccess Description: Container gives CAN transceiver driver information about accessing ports and port pins. In addition relation between CAN transceiver hardware pin names and Dio port access information is given. If a CAN transceiver hardware has no Dio interface, there is no instance of this container. Configuration Parameters	Only apply for Hardware support access via DIO
ECUC_CanTrcv_00157	N/S	Container Name: CanTrcv↩ DioChannelAccess Parent Container: CanTrcvDioAccess Description: Container gives DIO channel access by single Can transceiver channel. Configuration Parameters	Only apply for Hardware support access via DIO

Requirement	Status	Description	Notes
ECUC_CanTrcv_00150	N/S	Name: CanTrcvHardware↩ InterfaceName Parent Container: CanTrcvDio↩ ChannelAccess Description: CAN transceiver hardware interface name. It is typically the name of a pin. From a Dio point of view it is either a port, a single channel or a channel group. Depending on this fact either CANTRCV_DIO_↩ PORT_SYMBOLIC_NAME or CANTRCV_DIO_CHA↩ NNEL_SYMBOLIC_NAME or CANTRCV_DIO_CHA↩ NNEL_GROUP_SYMBO↩ LIC_NAME shall reference a Dio configuration. The CAN transceiver driver implementation description shall list up this name for the appropriate CAN transceiver hardware. Multiplicity: 1 Type: Ecuc↩ StringParamDef Default value: – maxLength: – minLength: – regularExpression: – Post-Build Variant Value: false Value Configuration Class: Pre-compile time: X: All Variants Link time: – Post-build time: – Scope / Dependency: scope: local	Only apply for Hardware support access via DIO
ECUC_CanTrcv_00149	N/S	Name: CanTrcvDioSym↩ NameRef Parent Container: CanTrcvDioChannelAccess Description: Choice Reference to a DIO Port, DIO Channel or DIO Channel Group. This reference replaces the CANTRCV↩ _DIO_PORT_SYM_NAME, CANTRCV_DIO_CHA↩ NNEL_SYM_NAME and CANTRCV_DIO_GROUP_↩ SYM_NAME references in the Can Trcv SWS. Multiplicity: 1 Type: Choice reference to [DioChannel , DioChannelGroup , DioPort] Post-Build Variant Value: false Value Configuration Class: Pre-compile time: X: All Variants Link time: – Post-build time: – Scope / Dependency	Only apply for Hardware support access via DIO

Requirement	Status	Description	Notes
ECUC_CanTrcv_00190	N/S	Name: CanTrcvTimerType Parent Container: CanTrcvGeneral Description: Type of the Time Service Predefined Timer. Multiplicity: 0..1 Type: Ecuc↩ EnumerationParamDef Range: None: None Timer_1us16bit: 16 bit 1us timer Post-Build Variant Multiplicity: false Post-Build Variant Value: false Multiplicity Configuration Class: Pre-compile time: X: All Variants Link time: – Post-build time: – Value Configuration Class: Pre-compile time: X: All Variants Link time: – Post-build time: – Scope / Dependency: scope: local	Applicable only for platforms which need timeout monitors
ECUC_CanTrcv_00191	N/S	Name: CanTrcvWaitTime Parent Container: CanTrcvGeneral Description: Wait time for transceiver state changes in seconds. Multiplicity: 0..1 Type↩ : EcucFloatParamDef Range↩ : [0 .. 2.55E-4] Default value: – Post-Build Variant Multiplicity: false Post-Build Variant Value: false Multiplicity Configuration Class: Pre-compile time: X: All Variants Link time: – Post-build time: – Value Configuration Class: Pre-compile time: X: All Variants Link time: – Post-build time: – Scope / Dependency: scope: local	Applicable only for platforms which need timeout monitors

Requirement	Status	Description	Notes
ECUC_CanTrcv_00193	N/S	Name: CanTrcvEcucPartition↩ Ref Parent Container: Can↩ TrcvGeneral Description: Maps the CAN transceiver driver to zero or multiple ECUC partitions to make the modules API available in this partition. The module will operate as an independent instance in each of the partitions. Multiplicity↩ : 0..∗ Type: Reference to [EcucPartition] Post-Build Variant Multiplicity: true Post-↩ Build Variant Value: true Multiplicity Configuration Class: Precompile time: X: All Variants Link time: – Post-build time: – Value Configuration Class: Precompile time: X: All Variants Link time: – Post-build time: – Scope / Dependency: scope: ECU	Applicable only for platforms which support multicore feature
ECUC_CanTrcv_00169	N/S	Name: CanTrcvBaudRate Parent Container: CanTrcv↩ PartialNetwork Description: Indicates the data transfer rate in kbps. Multiplicity: 1 Type: EcucIntegerParamDef Range: 0 .. 12000 Default value: – Post-Build Variant Value: true Value Configuration Class: Pre-compile time: X: VARIANT-PRE-COMPILE Link time: X: VARIANT-↩ LINK-TIME Post-build time: X: VARIANT-POST-BUILD Scope / Dependency: scope: local: dependency: Although WUF with DLC=0 is technically possible, it is explicitly not wanted.	Applicable only for platforms which support Selective wakeup feature

Requirement	Status	Description	Notes
ECUC_CanTrcv_00164	N/S	Name: CanTrcvPnCanIdIs↩ Extended Parent Container↩ : CanTrcvPartialNetwork Description: Indicates whether extended or standard ID is used. TRUE = Extended Can identifier is used. FALSE = Standard Can identifier is used Multiplicity: 1 Type: EcucBooleanParamDef Default value: false Post-Build Variant Value: true Value Configuration Class: Pre-compile time: X: VARIANT-PRE-COMPILE Link time: X: VARIANT-↩ LINK-TIME Post-build time: X: VARIANT-POST-BUILD Scope / Dependency: scope: local	Only apply for Hardware support Selective Wakeup
ECUC_CanTrcv_00172	N/S	Name: CanTrcvPnEnabled Parent Container: CanTrcv↩ PartialNetwork Description: Indicates whether the selective wake-up function is enabled or disabled in HW.: TRUE = Selective wakeup feature is enabled in the transceiver hardware FALSE = Selective wakeup feature is disabled in the transceiver hardware Multiplicity: 1 Type: Ecuc↩ BooleanParamDef Default value: false Post-Build Variant Value: true Value Configuration Class: Pre-compile time: X: VARIANT-PRE-COMPILE Link time: X: VARIANT-↩ LINK-TIME Post-build time: X: VARIANT-POST-BUILD Scope / Dependency: scope: local	Only apply for Hardware support Selective Wakeup

Requirement	Status	Description	Notes
ECUC_CanTrcv_00163	N/S	Name: CanTrcvPnFrameCanId Parent Container: CanTrcv↩ PartialNetwork Description: CAN ID of the Wake-up Frame (WUF). Multiplicity: 1 Type: EcucIntegerParamDef Range: 0 .. 4294967295 Default value: – Post-Build Variant Value↩ : true Value Configuration Class: Pre-compile time: X: VARIANT-PRE-COMPILE Link time: X: VARIANT-↩ LINK-TIME Post-build time: X: VARIANT-POST-BUILD Scope / Dependency: scope: local	Only apply for Hardware support Selective Wakeup
ECUC_CanTrcv_00162	N/S	Name: CanTrcvPnFrameCan↩ IdMask Parent Container↩ : CanTrcvPartialNetwork Description: ID Mask for the selective activation of the transceiver. It is used to enableFrame Wake-up (WUF) on a group of IDs. Multiplicity: 1 Type: EcucIntegerParamDef Range: 0 .. 4294967295 Default value: – Post-Build Variant Value: true Value Configuration Class: Pre-compile time: X: VARIANT-PRE-COMPILE Link time: X: VARIANT-↩ LINK-TIME Post-build time: X: VARIANT-POST-BUILD Scope / Dependency: scope: local	Only apply for Hardware support Selective Wakeup
ECUC_CanTrcv_00168	N/S	Name: CanTrcvPnFrameDlc Parent Container: CanTrcv↩ PartialNetwork Description: Data Length of the Wake-up Frame (WUF). Multiplicity: 1 Type: EcucIntegerParamDef Range: 0 .. 8 Default value: – Post-Build Variant Value↩ : true Value Configuration Class: Pre-compile time: X: VARIANT-PRE-COMPILE Link time: X: VARIANT-↩ LINK-TIME Post-build time: X: VARIANT-POST-BUILD Scope / Dependency: scope: local	Only apply for Hardware support Selective Wakeup

Requirement	Status	Description	Notes
ECUC_CanTrcv_00170	N/I	Name: CanTrcvPowerOnFlag Parent Container: CanTrcv↩ PartialNetwork Description: Description: Indicates if the Power On Reset (POR) flag is available and is managed by the transceiver.: TRUE = Supported by Hardware. FALSE = Not supported by Hardware Multiplicity: 1 Type: EcucBooleanParamDef Default value: false Post-Build Variant Value: true Value Configuration Class: Pre-compile time: X: VARIANT-PRE-COMPILE Link time: X: VARIANT-↩ LINK-TIME Post-build time: X: VARIANT-POST-BUILD Scope / Dependency: scope: local	Not implemented yet
ECUC_CanTrcv_00165	N/S	Container Name: CanTrcv↩ PnFrameDataMaskSpec Parent Container: CanTrcvPartial↩ Network Description: Defines data payload mask to be used on the received payload in order to determine if the transceiver must be woken up by the received Wake-up Frame (WUF). Configuration Parameters	Applicable only for platforms which support Selective wakeup feature
ECUC_CanTrcv_00166	N/S	Name: CanTrcvPnFrame↩ DataMask Parent Container: CanTrcvPnFrameDataMask↩ Spec Description: Defines the n byte (Byte0 = LSB) of the data payload mask to be used on the received payload in order to determine if the transceiver must be woken up by the received Wake-up Frame (WUF). Multiplicity: 1 Type: EcucIntegerParam↩ Def Range: 0 .. 255 Default value: – Post-Build Variant Value: true Value Configuration Class: Pre-compile time: X: VARIANT-PRE-COMPILE Link time: X: VARIANT-↩ LINK-TIME Post-build time: X: VARIANT-POST-BUILD Scope / Dependency: scope: local	Applicable only for platforms which support Selective wakeup feature

Requirement	Status	Description	Notes
ECUC_CanTrcv_00167	N/S	Name: CanTrcvPnFrameData↩ MaskIndex Parent Container↩ : CanTrcvPnFrameDataMask↩ Spec Description: holds the position n in frame of the mask-part Multiplicity: 1 Type: EcucIntegerParamDef Range: 0 .. 7 Default value: – Post-Build Variant Value: true Value Configuration Class: Pre-compile time: X: VARIANT-PRE-C↩ OMPILE Link time: X: VA↩ RIANT-LINK-TIME Post-build time: X: VARIANT-POST-↩ BUILD Scope / Dependency↩ : scope: local	Applicable only for platforms which support Selective wakeup feature
ECUC_CanTrcv_00183	N/S	Container Name: CanTrcvSpi↩ Access Parent Container: Can↩ TrcvAccess Description: Container gives CAN transceiver driver information about accessing Spi. If a CAN transceiver hardware has no Spi interface, there is no instance of this container. Configuration Parameters	This configuration is done indirectly in Ae Driver
ECUC_CanTrcv_00144	N/S	Container Name: Can↩ TrcvSpiSequence Parent Container: CanTrcvSpiAccess Description: Container gives CAN transceiver driver information about one SPI sequence. One SPI sequence used by CAN transceiver driver is in exclusive use for it. No other driver is allowed to access this sequence. CAN transceiver driver may use one sequence to access n CAN transceiver hardwares chips of the same type or n sequences are used to access one single CAN transceiver hardware chip. If a CAN transceiver hardware has no SPI interface, there is no instance of this container. Configuration Parameters	This configuration is done indirectly in Ae Driver

Requirement	Status	Description	Notes
ECUC_CanTrcv_00176	N/S	Name: CanTrcvSpi↩ AccessSynchronous Parent Container: CanTrcvSpi↩ Sequence Description: This parameter is used to define whether the access to the Spi sequence is synchronous or asynchronous.: true: SPI access is synchronous. false: SPI access is asynchronous. Multiplicity: 0..1 Type: Ecuc↩ BooleanParamDef Default value: false Post-Build Variant Multiplicity: false Post-Build Variant Value: false Multiplicity Configuration Class: Pre-compile time: X: All Variants Link time: – Post-build time: – Value Configuration Class: Pre-compile time: X: All Variants Link time: – Post-build time: – Scope / Dependency: scope: local	This configuration is done indirectly in Ae Driver
ECUC_CanTrcv_00151	N/S	Name: CanTrcvSpiSequence↩ Name Parent Container: Can↩ TrcvSpiSequence Description↩ : Reference to a Spi sequence configuration container. Multiplicity: 0..∗ Type: Symbolic name reference to [Spi↩ Sequence] Post-Build Variant Multiplicity: false Post-Build Variant Value: false Multiplicity Configuration Class: Precompile time: X: All Variants Link time: – Post-build time: – Value Configuration Class: Precompile time: X: All Variants Link time: – Post-build time: – Scope / Dependency: scope: local: dependency: SpiSequence	This configuration is done indirectly in Ae Driver

3.5 Driver Limitations

The CanTrcv Driver has the following limitations:

- Driver doesn't support Sleep mode.

3.6 Driver usage and configuration tips

3.6.1 How to configure the SPI sequence via AE module

Ae module is used to access CAN transceiver hardware connected via SPI. This SPI sequence shalln't configure in CanTrcv module, It have to configure in Ae module.

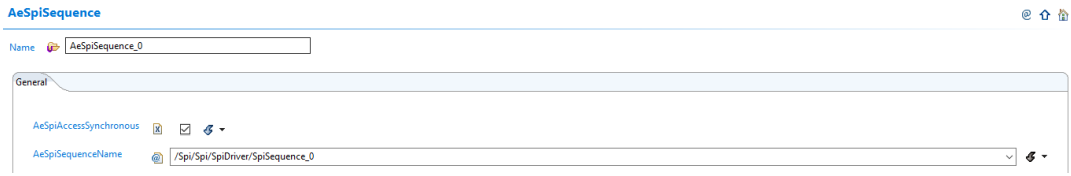

Figure 3.1 Reference to a Spi sequence configuration in Ae module over EB tresos.

SPI Configuration:
Configuration SPI sequence:

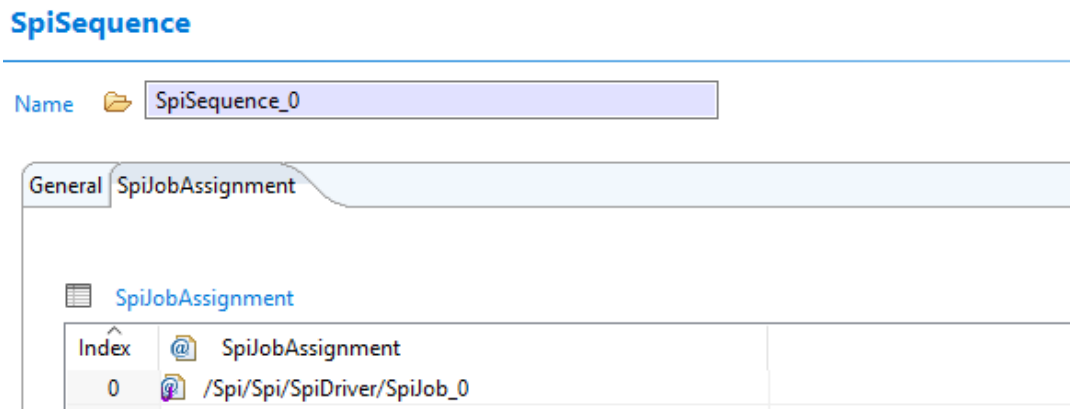

Figure 3.2 Configuration SPI sequence.

Configuration SPI Job:
In Spi Job tab, Need to configure the external device and asignment to Spi channel.

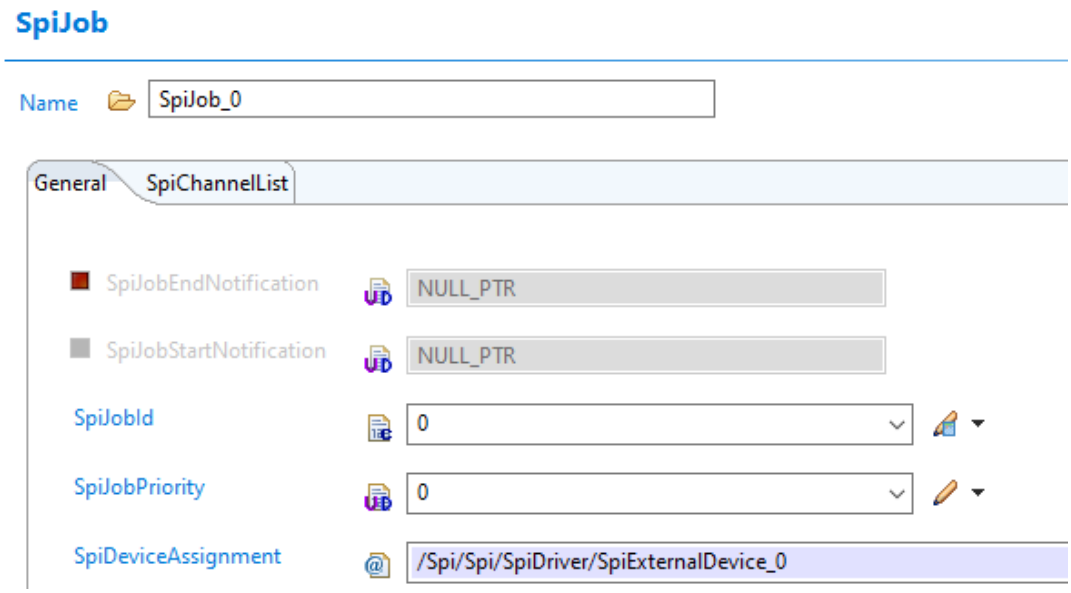

Figure 3.3 Configuration SPI job.

In Spi External Device tab:

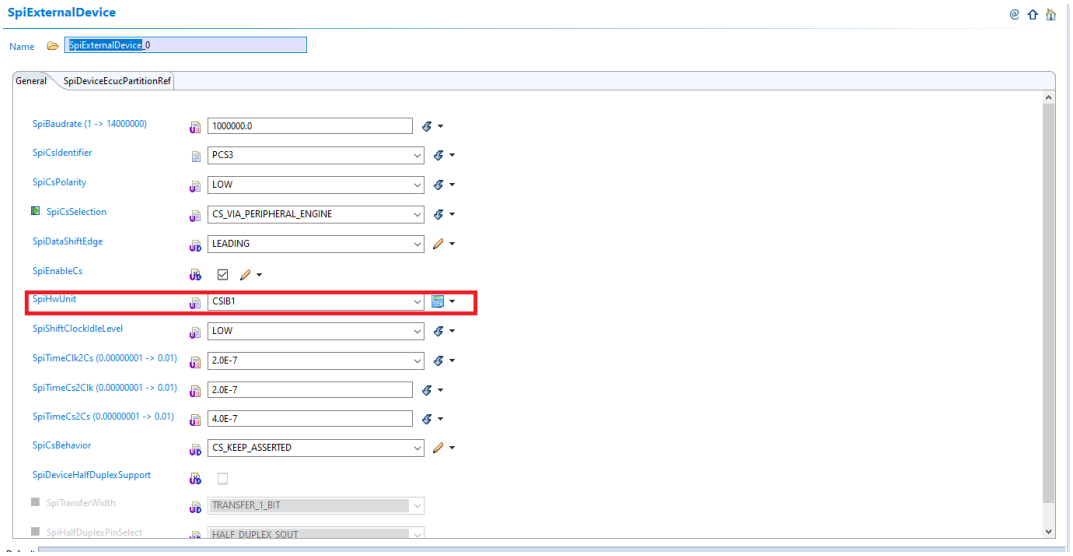

Figure 3.4 Configuration SPI External Device.

Need to make sure that the SpiHwUnit must refer to the Spi Phy Uint tab and the SpiPhyUnitMapping is LPSPI_1

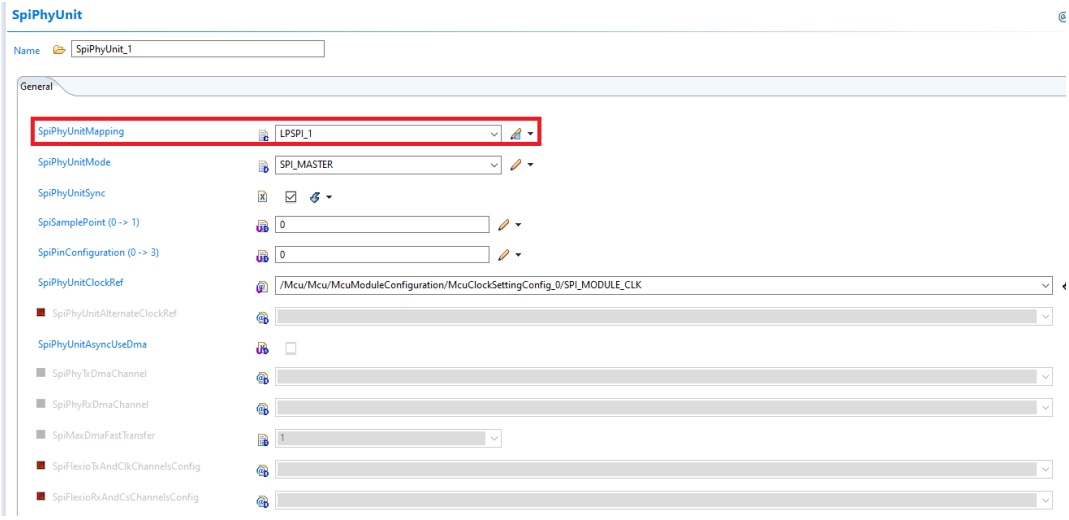

Figure 3.5 Configuration SPI channel.

In SpiChannelList tab:
Need to assignment to the Spi Channel and configure for SpiChannel as below:

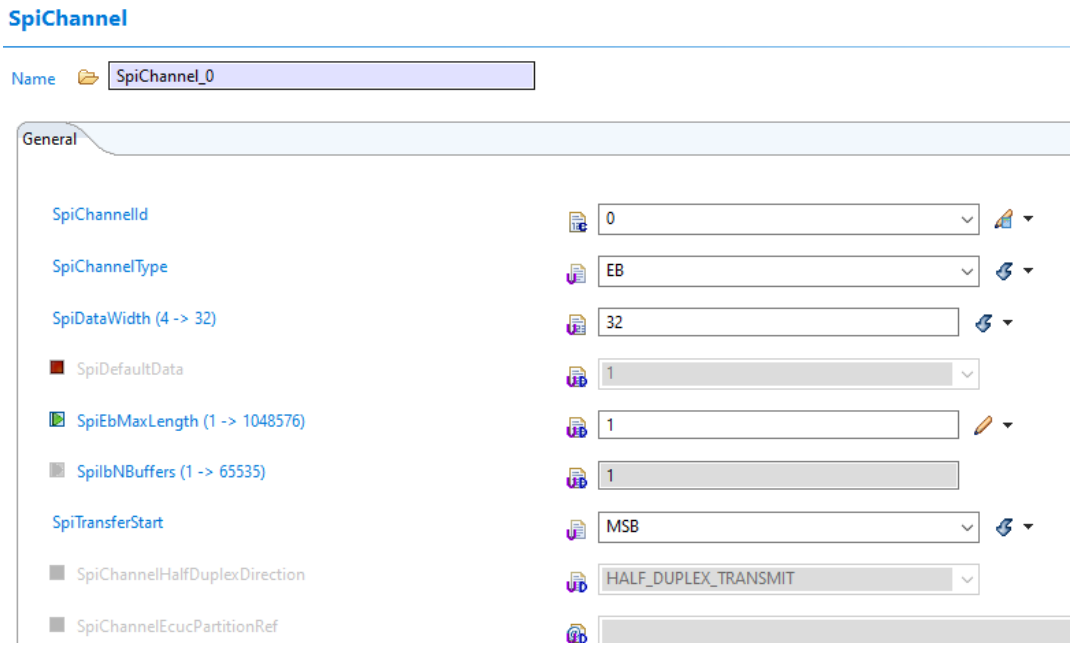

Figure 3.6 Configuration SPI channel.

Port Configuration:
Need to configure Pins for SPI in order communicates with CAN PHY interface.

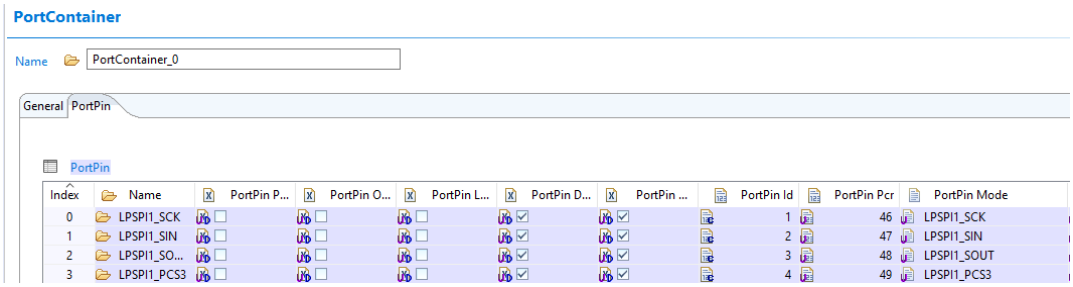

Figure 3.7 Configuration SPI Pins.

MCU Configuration:
Make sure that LPSPI_1 must enable clock.

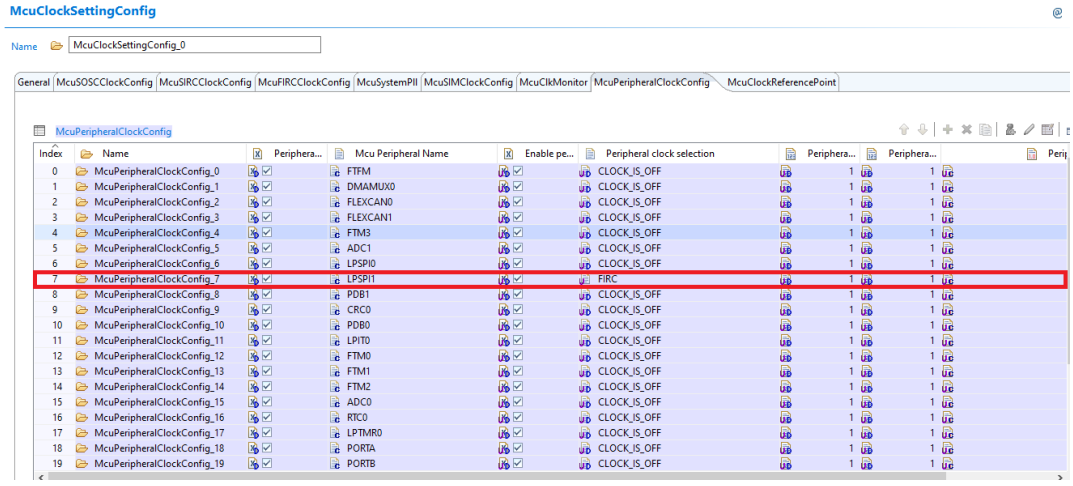

Figure 3.8 Enalbe the clock for LPSPI_1.

3.7 Runtime errors

The CanTrcv driver generates the following errors at runtime:

3.7.1 Runtime Errors

Function	Error code	Condition triggering the error
CanTrcv_Init	CANTRCV_E_NO_TRCV_C↩ ONTROL	No/incorrect communication to transceiver.
CanTrcv_SetOpMode	CANTRCV_E_NO_TRCV_C↩ ONTROL	No/incorrect communication to transceiver.
CanTrcv_GetOpMode	CANTRCV_E_NO_TRCV_C↩ ONTROL	No/incorrect communication to transceiver.

Function	Error code	Condition triggering the error
CanTrcv_GetBusWuReason	CANTRCV_E_NO_TRCV_C↩ ONTROL	No/incorrect communication to transceiver.
CanTrcv_ SetWakeupMode	CANTRCV_E_NO_TRCV_C↩ ONTROL	No/incorrect communication to transceiver.
CanTrcv_DeInit	CANTRCV_E_NO_TRCV_C↩ ONTROL	No/incorrect communication to transceiver.
CanTrcv_MainFunction↩ Diagnostics	CANTRCV_E_BUS_ERROR	A CAN bus error occured during communication.

3.8 Symbolic Names Disclaimer

All containers having symbolicNameValue set to TRUE in the AUTOSAR schema will generate defines like:

#define <Mip>Conf_<Container_ShortName>_<Container_ID>

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

Chapter 4

Tresos Configuration Plug-in

This chapter describes the Tresos configuration plug-in for the driver. All the parameters are described below.

- Module CanTrcv
  - Container CanTrcvGeneral
    - Parameter CanTrcvWakeUpSupport
    - Parameter CanTrcvMainFunctionDiagnosticsPeriod
    - Parameter CanTrcvMainFunctionPeriod
    - Parameter CanTrcvDevErrorDetect
    - Parameter CanTrcvMulticoreSupport
    - Parameter CanTrcvTimerType
    - Parameter CanTrcvWaitTime
    - Parameter CanTrcvVersionInfoApi
    - Parameter CanTrcvIndex
    - Reference CanTrcvEcucPartitionRef
  - Container CanTrcvConfigSet
    - Parameter CanTrcvSPICommRetries
    - Parameter CanTrcvSPICommTimeout
    - Container CanTrcvChannel
      - Parameter CanTrcvInitState
      - Parameter CanTrcvChannelId
      - Parameter CanTrcvAbstractCanIfId
      - Parameter CanTrcvMaxBaudrate
      - Parameter CanTrcvChannelUsed
      - Parameter CanTrcvControlsPowerSupply
      - Parameter CanTrcvWakeupByBusUsed
      - Parameter CanTrcvHwPnSupport
      - Reference CanTrcvIcuChannelRef
      - Reference CanTrcvPorWakeupSourceRef
      - Reference CanTrcvSyserrWakeupSourceRef
      - Reference CanTrcvWakeupSourceRef
      - Reference CanTrcvChannelEcucPartitionRef
      - Container CanTrcvAccess

- Container CanTrcvDemEventParameterRefs
- Reference CANTRCV_E_BUS_ERROR
- Container CanTrcvPartialNetwork
- Parameter CanTrcvBaudRate
- Parameter CanTrcvPnFrameCanId
- Parameter CanTrcvPnFrameCanIdMask
- Parameter CanTrcvPnFrameDlc
- Parameter CanTrcvPnCanIdIsExtended
- Parameter CanTrcvPnEnabled
- Parameter CanTrcvBusErrFlag
- Parameter CanTrcvPowerOnFlag
- Container CanTrcvPnFrameDataMaskSpec
- Parameter CanTrcvPnFrameDataMask
- Parameter CanTrcvPnFrameDataMaskIndex

– Container CommonPublishedInformation

* Parameter ArReleaseMajorVersion
* Parameter ArReleaseMinorVersion
* Parameter ArReleaseRevisionVersion
* Parameter ModuleId
* Parameter SwMajorVersion
* Parameter SwMinorVersion
* Parameter SwPatchVersion
* Parameter VendorApiInfix
* Parameter VendorId

4.1 Module CanTrcv

This container holds the configuration of a single CanTrcv Driver.

Included containers:

- CanTrcvGeneral
- CanTrcvConfigSet
- CommonPublishedInformation

Property	Value
type	ECUC-MODULE-DEF
lowerMultiplicity	0
upperMultiplicity	Infinite
postBuildVariantSupport	true
supportedConfigVariants	VARIANT-LINK-TIME, VARIANT-POST-BUILD, VARIANT-PRE-COMPILE

4.2 Container CanTrcvGeneral

This container holds parameters related to each CanTrcv Driver.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.3 Parameter CanTrcvWakeUpSupport

CANTRCV_WAKEUP_BY_POLLING: Wake up by polling.

CANTRCV_WAKEUP_NOT_SUPPORTED: Wake up not supported.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	CANTRCV_WAKEUP_NOT_SUPPORTED
literals	['CANTRCV_WAKEUP_BY_POLLING', 'CANTRCV_WAKEUP_NOT_↩ SUPPORTED']

4.4 Parameter CanTrcvMainFunctionDiagnosticsPeriod

This parameter describes the period for cyclic call to CanTrcv_MainFunctionDiagnostics.

Property	Value
type	ECUC-FLOAT-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	False
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	0.0
max	255.0
min	0.0

4.5 Parameter CanTrcvMainFunctionPeriod

This parameter describes the period for cyclic call to CanTrcv_MainFunction.

Property	Value
type	ECUC-FLOAT-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	False
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	0.0
max	255.0
min	0.0

4.6 Parameter CanTrcvDevErrorDetect

Switches the Development Error Detection and Notification: ON or OFF.

When this option is OFF code size is reduced, but no error detection is available.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	false

4.7 Parameter CanTrcvMulticoreSupport

Enable Maps CanTrcv driver to multiple EcuC partitions to make the modules API available in this partition. The CanTrcv driver will operate as an independent instance in each of the partitions.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	false

4.8 Parameter CanTrcvTimerType

Type of the Time Service Predefined Timer.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	None
literals	['None', 'Timer_1us16bit']

4.9 Parameter CanTrcvWaitTime

Wait time for transceiver state changes in seconds.

Property	Value
type	ECUC-FLOAT-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	False
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	0.0
max	2.55E-4
min	0.0

4.10 Parameter CanTrcvVersionInfoApi

Switches the CanTrcv_GetVersionInfo() API: ON or OFF.

When this option is ON, the driver supports API for getting Version information of the Driver.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	false

4.11 Parameter CanTrcvIndex

Specifies the InstanceId of this module instance.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	0
max	255
min	0

4.12 Reference CanTrcvEcucPartitionRef

Maps the CAN transceiver driver to zero or multiple ECUC partitions to make the modules API available in this partition.

Property	Value
type	ECUC-REFERENCE-DEF
origin	AUTOSAR_ECUC
lowerMultiplicity	0
upperMultiplicity	Infinite
postBuildVariantMultiplicity	true
multiplicityConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
postBuildVariantValue	true
valueConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
requiresSymbolicNameValue	False
destination	/AUTOSAR/EcucDefs/EcuC/EcucPartitionCollection/EcucPartition

4.13 Container CanTrcvConfigSet

This is the multiple configuration set container for CanTrcv Driver.

Included subcontainers:

- CanTrcvChannel

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.14 Parameter CanTrcvSPICommRetries

Indicates the maximum number of communication retries in case of a failed SPI communication (applies both to timed out communication and to errors/NACK in the response data).

If configured value is '0', no retry is allowed (communication is expected to succeed at first try).

Property	Value
type	ECUC-INTEGER-PARAM-DEF

Property	Value
origin	AUTOSAR_ECUC
symbolicNameValue	False
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-LINK-TIME: LINK
	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	0
max	255
min	0

4.15 Parameter CanTrcvSPICommTimeout

Indicates the maximum time allowed to the CanTrcv to reply (either positively or negatively) to a SPI command.

Timeout is configured in milliseconds. Timeout value of '0' means no specific timeout is to be used by CanTrcv and the communication is executed at the best of the SPI HW capacity.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	False
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-LINK-TIME: LINK
	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	0
max	100
min	0

4.16 Container CanTrcvChannel

Defines a structure to initialize the selected CAN transceiver (channel).

Included subcontainers:

- CanTrcvAccess
- CanTrcvDemEventParameterRefs
- CanTrcvPartialNetwork

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	Infinite
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE

4.17 Parameter CanTrcvInitState

State of CAN transceiver after call to CanTrcv_Init.

- Normal mode after Init

- Sleep mode after Init

- Standby mode after Init (Default)

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	CANTRCV_OP_MODE_STANDBY
literals	['CANTRCV_OP_MODE_SLEEP', 'CANTRCV_OP_MODE_STANDBY']

4.18 Parameter CanTrcvChannelId

Unique identifier of the CAN Transceiver Channel.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	1
max	255
min	0

4.19 Parameter CanTrcvAbstractCanIfId

Vendor specific:The abstract CanIf TransceiverId used when Driver invokes the callback functions from CanIf.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	1
max	255
min	0

4.20 Parameter CanTrcvMaxBaudrate

Indicates the data transfer rate in kbps. Maximum data transfer rate in kbps for transceiver hardware type.

For CAN the maximum data transfer rate is 500 kbps, for CAN FD 2000 kbps.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	5000
max	12000
min	0

4.21 Parameter CanTrcvChannelUsed

Shall the related CAN transceiver be used?

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	true

4.22 Parameter CanTrcvControlsPowerSupply

Is ECU power supply controlled by this transceiver?

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	AUTOSAR_ECUC

Property	Value
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	false

4.23 Parameter CanTrcvWakeupByBusUsed

Is wake up by bus supported?

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	true

4.24 Parameter CanTrcvHwPnSupport

Indicates whether the HW supports the selective wake-up functionality.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1

Property	Value
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	false

4.25 Reference CanTrcvIcuChannelRef

Reference to the IcuChannel to enable/disable the interrupts for wakeups.

Property	Value
type	ECUC-REFERENCE-DEF
origin	AUTOSAR_ECUC
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
requiresSymbolicNameValue	False
destination	/AUTOSAR/EcucDefs/Icu/IcuConfigSet/IcuChannel

4.26 Reference CanTrcvPorWakeupSourceRef

Symbolic name reference to specify the wakeup sources that should be used in the calls to EcuM_SetWakeupEvent.

Property	Value
type	ECUC-REFERENCE-DEF
origin	AUTOSAR_ECUC
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	false

Property	Value
multiplicityConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
requiresSymbolicNameValue	False
destination	/AUTOSAR/EcucDefs/EcuM/EcuMConfiguration/EcuMCommon↩ Configuration/EcuMWakeupSource

4.27 Reference CanTrcvSyserrWakeupSourceRef

Symbolic name reference to specify the wakeup sources that should be used in the calls to EcuM_SetWakeupEvent.

Property	Value
type	ECUC-REFERENCE-DEF
origin	AUTOSAR_ECUC
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
requiresSymbolicNameValue	False
destination	/AUTOSAR/EcucDefs/EcuM/EcuMConfiguration/EcuMCommon↩ Configuration/EcuMWakeupSource

4.28 Reference CanTrcvWakeupSourceRef

Reference to a wakeup source in the EcuM configuration.

Property	Value
type	ECUC-REFERENCE-DEF
origin	AUTOSAR_ECUC
lowerMultiplicity	0

Property	Value
upperMultiplicity	1
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
requiresSymbolicNameValue	False
destination	/AUTOSAR/EcucDefs/EcuM/EcuMConfiguration/EcuMCommon↩ Configuration/EcuMWakeupSource

4.29 Reference CanTrcvChannelEcucPartitionRef

Maps the CAN transceiver configuration to zero or one ECUC partitions.

Property	Value
type	ECUC-REFERENCE-DEF
origin	AUTOSAR_ECUC
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	true
multiplicityConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
postBuildVariantValue	true
valueConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
requiresSymbolicNameValue	False
destination	/AUTOSAR/EcucDefs/EcuC/EcucPartitionCollection/EcucPartition

4.30 Container CanTrcvAccess

Gives CanTrcv Driver information about access to a single CAN transceiver.

Included choices:

- CanTrcvSpiAccess
- CanTrcvDioAccess

Property	Value
type	ECUC-CHOICE-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.31 Container CanTrcvDemEventParameterRefs

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

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.32 Reference CANTRCV_E_BUS_ERROR

Reference to the DemEventParameter which shall be issued when bus error has occurred.

Property	Value
type	ECUC-REFERENCE-DEF
origin	AUTOSAR_ECUC
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE

Property	Value
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
requiresSymbolicNameValue	False
destination	/AUTOSAR/EcucDefs/Dem/DemConfigSet/DemEventParameter

4.33 Container CanTrcvPartialNetwork

Container gives CAN transceiver driver information about the configuration of Partial Networking functionality.

Included subcontainers:

- CanTrcvPnFrameDataMaskSpec

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE

4.34 Parameter CanTrcvBaudRate

Indicates the data transfer rate in kbps.

Supported values:

- 50 kBits

- 100 kBits

- 125 kBits

- 250 kBits

- 500 kBits

- 1000 kBits

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-LINK-TIME: LINK
	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	50
max	12000
min	0

4.35 Parameter CanTrcvPnFrameCanId

CAN ID of the Wake-up Frame (WUF).

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-LINK-TIME: LINK
	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	0
max	4294967295
min	0

4.36 Parameter CanTrcvPnFrameCanIdMask

ID Mask for the selective activation of the transceiver. It is used to enable-Frame Wake-up (WUF) on a group of IDs.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-LINK-TIME: LINK
	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	0
max	4294967295
min	0

4.37 Parameter CanTrcvPnFrameDlc

Data Length of the Wake-up Frame (WUF).

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-LINK-TIME: LINK
	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	1
max	8
min	0

4.38 Parameter CanTrcvPnCanIdIsExtended

Indicates whether extended or standard ID is used.

TRUE = Extended Can identifier is used.

FALSE = Standard Can identifier is used

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-LINK-TIME: LINK
	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	false

4.39 Parameter CanTrcvPnEnabled

Indicates whether the selective wake-up function is enabled or disabled in HW.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-LINK-TIME: LINK
	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	false

4.40 Parameter CanTrcvBusErrFlag

Indicates if the Bus Error (BUSERR) flag is managed by the BSW. This flag is set if a bus failure is detected by the transceiver.

TRUE = Supported by transceiver and managed by BSW. FALSE = Not managed by BSW.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF

Property	Value
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-LINK-TIME: LINK
	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	true

4.41 Parameter CanTrcvPowerOnFlag

Description: Indicates if the Power On Reset (POR) flag is available and is managed by the transceiver.

TRUE = Supported by Hardware.

FALSE = Not supported by Hardware

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-LINK-TIME: LINK
	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	true

4.42 Container CanTrcvPnFrameDataMaskSpec

Defines data payload mask to be used on the received payload in order to determine if the transceiver must be woken up by the received wake-up frame (WUF).

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	0
upperMultiplicity	8
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-LINK-TIME: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE

4.43 Parameter CanTrcvPnFrameDataMask

Defines the byte 0 of the data payload mask to be used on the received payload in order to determine if the transceiver must be woken up by the received Wake-up frame (WUF)..

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-LINK-TIME: LINK
	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	0
max	255
min	0

4.44 Parameter CanTrcvPnFrameDataMaskIndex

Holds the position n in frame of the mask-part.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A

Property	Value
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-LINK-TIME: LINK
	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	0
max	7
min	0

4.45 Container CommonPublishedInformation

Common container, aggregated by all modules.

It contains published information about vendor and versions.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.46 Parameter ArReleaseMajorVersion

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

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false

Property	Value
valueConfigClasses	VARIANT-LINK-TIME: PUBLISHED-INFORMATION
	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
defaultValue	4
max	4
min	4

4.47 Parameter ArReleaseMinorVersion

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

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PUBLISHED-INFORMATION
	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
defaultValue	7
max	7
min	7

4.48 Parameter ArReleaseRevisionVersion

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

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false

Property	Value
valueConfigClasses	VARIANT-LINK-TIME: PUBLISHED-INFORMATION
	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
defaultValue	0
max	0
min	0

4.49 Parameter ModuleId

Module ID of this module from Module List.

Note: Implementation Specific Parameter

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PUBLISHED-INFORMATION
	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
defaultValue	70
max	70
min	70

4.50 Parameter SwMajorVersion

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

Note: Implementation Specific Parameter

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1

Property	Value
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PUBLISHED-INFORMATION
	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
defaultValue	3
max	3
min	3

4.51 Parameter SwMinorVersion

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

Note: Implementation Specific Parameter

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PUBLISHED-INFORMATION
	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
defaultValue	0
max	0
min	0

4.52 Parameter SwPatchVersion

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

Note: Implementation Specific Parameter

Property	Value
type	ECUC-INTEGER-PARAM-DEF

Property	Value
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PUBLISHED-INFORMATION
	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
defaultValue	0
max	0
min	0

4.53 Parameter VendorApiInfix

In driver modules which can be instantiated several times on a single ECU, BSW00347 requires that the name of APIs is extended by the VendorId and a vendor specific name.

This parameter is used to specify the vendor specific name. In total, the Implementation specific name is generated as follows:

<ModuleName>_>VendorId>_<VendorApiInfix>.

E.g. assuming that the VendorId of the implementor is 123 and the implementer chose a VendorApiInfix of "v11r456" a api name

Can_Write defined in the SWS will translate to Can_123_v11r456Write.

This parameter is mandatory for all modules with upper multiplicity >

1. It shall not be used for modules with upper multiplicity =1.

Note: Implementation Specific Parameter

Property	Value
type	ECUC-STRING-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PUBLISHED-INFORMATION
	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
defaultValue	AE

4.54 Parameter VendorId

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

Note: Implementation Specific Parameter

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-LINK-TIME: PUBLISHED-INFORMATION
	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
defaultValue	43
max	43
min	43

Chapter 5

Module Index

5.1 Software Specification

Here is a list of all modules:

Chapter 6

Module Documentation

6.1 CANTRCV_DRIVER

6.1.1 Detailed Description

Data Structures

- struct CanTrcv_43_AE_ConfigType

  Can Transceiver Configuration. *More...*
- struct CanTrcv_43_AE_TransceiverConfigType

  CanTrcv_43_AE_TransceiverConfigType. *More...*

Macros

- #define CANTRCV_43_AE_E_INVALID_TRANSCEIVER

  Development Error ID for API called with wrong parameter for the CAN transceiver.
- #define CANTRCV_43_AE_E_PARAM_POINTER

  Development Error ID for API called with null pointer parameter.
- #define CANTRCV_43_AE_E_UNINIT

  Development Error ID for API service used without initialization.
- #define CANTRCV_43_AE_E_TRCV_NOT_STANDBY

  Development Error ID for API service called in wrong transceiver operation mode (STANDBY expected)
- #define CANTRCV_43_AE_E_TRCV_NOT_NORMAL

  Development Error ID for API service called in wrong transceiver operation mode (NORMAL expected)
- #define CANTRCV_43_AE_E_PARAM_TRCV_WAKEUP_MODE

  Development Error ID for API service called with invalid parameter for TrcvWakeupMode.
- #define CANTRCV_43_AE_E_PARAM_TRCV_OPMODE

  Development Error ID for API service called with invalid parameter for OpMode.
- #define CANTRCV_43_AE_E_BAUDRATE_NOT_SUPPORTED

  Development Error ID for Configured baud rate is not supported by the transceiver.
- #define CANTRCV_43_AE_E_INIT_FAILED

Development Error ID for Module initialization has failed, e.g. CanTrcv_Init() called with an invalid pointer in postbuild.

- #define CANTRCV_43_AE_E_NO_TRCV_CONTROL
  Runtime Error ID when No/incorrect communication to transceiver.
- #define CANTRCV_43_AE_SID_INIT
  Service ID of CanTrcv_43_AE_Init.
- #define CANTRCV_43_AE_SID_SET_OPMODE
  Service ID of CanTrcv_43_AE_SetOpMode.
- #define CANTRCV_43_AE_SID_GET_OPMODE
  Service ID of CanTrcv_43_AE_GetOpMode.
- #define CANTRCV_43_AE_SID_GET_BUS_WU_REASON
  Service ID of CanTrcv_43_AE_GetBusWuReason.
- #define CANTRCV_43_AE_SID_GET_VERSION_INFO
  Service ID of CanTrcv_43_AE_GetVersionInfo.
- #define CANTRCV_43_AE_SID_SET_WAKEUP_MODE
  Service ID of CanTrcv_43_AE_SetWakeupMode.
- #define CANTRCV_43_AE_SID_CHECK_WAKEUP
  Service ID of CanTrcv_43_AE_CheckWakeup.
- #define CANTRCV_43_AE_SID_CHECK_WAKE_FLAG
  Service ID of CanTrcv_43_AE_CheckWakeFlag.
- #define CANTRCV_43_AE_SID_DEINIT
  Service ID of CanTrcv_43_AE_DeInit.

Enum Reference

- enum CanTrcv_43_AE_eDriverStatusType
  CanTrcv_43_AE_eDriverStatusType.

Function Reference

- void CanTrcv_43_AE_GetVersionInfo (Std_VersionInfoType *VersionInfo)
  CAN transceiver driver get version info function. SID is 0x04.
- void CanTrcv_43_AE_Init (const CanTrcv_43_AE_ConfigType *ConfigPtr)
  Initializes CanTrcv module. SID 0x00.
- Std_ReturnType CanTrcv_43_AE_SetOpMode (uint8 Transceiver, CanTrcv_TrcvModeType OpMode)
  Sets the mode of the Transceiver to the value OpMode. SID 0x01.
- Std_ReturnType CanTrcv_43_AE_GetOpMode (uint8 Transceiver, CanTrcv_TrcvModeType *OpMode)
  Gets current operation mode. SID 0x02.
- Std_ReturnType CanTrcv_43_AE_GetBusWuReason (uint8 Transceiver, CanTrcv_TrcvWakeupReason↩ Type *Reason)
  Gets the wakeup reason of the Transceiver and returns it in parameter Reason. SID 0x03.
- Std_ReturnType CanTrcv_43_AE_SetWakeupMode (uint8 Transceiver, CanTrcv_TrcvWakeupModeType TrcvWakeupMode)
  Enables, disables or clears wake up events of the Transceiver according to TrcvWakeupMode. SID 0x05.
- Std_ReturnType CanTrcv_43_AE_CheckWakeup (uint8 Transceiver)
  Service is called by underlying CANIF in case a wake up interrupt is detected. SID 0x07.
- Std_ReturnType CanTrcv_43_AE_CheckWakeFlag (uint8 Transceiver)
  Requests to check the status of the wakeup flag from the transceiver hardware. SID 0x0E.
- void CanTrcv_43_AE_DeInit (void)
  De-initializes the CanTrcv module. SID 0x10.

Variables

- const uint32 CanTrcv_u32CoreID
  Configuration Core ID.
- const CanTrcv_43_AE_TransceiverConfigType *const * CanTrcv_ppTransceivers
  Pointer to all Tranceiver channels asigned to the partition.
- const uint8 CanTrcv_u8SwTransceiverId
  The Id of Transceiver channel configured by tool configuration.
- const uint8 CanTrcv_u8HwTransceiverId
  The Id of Transceiver channel mapped in Hardware.
- const uint8 CanTrcv_u8CanIfTransceiverId
  The Id referred by CanIf to CanTrcv.
- const uint32 CanTrcv_WakeupSourceMask
  The value report to EcuM for a wakeup event.
- const uint16 CanTrcv_DemEventId
  The value report to Dem for a bus error.
- const boolean CanTrcv_bWakeupByBusUsed
  Specifies if wakeup is enabled on the transceiver.
- const CanTrcv_TrcvModeType CanTrcv_eInitState
  The mode of Transceiver after Init.
- const CanTrcv_43_AE_Ipw_TransceiverConfigType *const CanTrcv_IpwTransceiverConfig
  Pointer to Ipw configuration wrapped to specific IPV configuration.

6.1.2 Data Structure Documentation

6.1.2.1 struct CanTrcv_43_AE_ConfigType

Can Transceiver Configuration.

Definition at line 147 of file CanTrcv_43_AE.h.

Data Fields

- const uint32 CanTrcv_u32CoreID
  Configuration Core ID.
- const CanTrcv_43_AE_TransceiverConfigType *const * CanTrcv_ppTransceivers
  Pointer to all Tranceiver channels asigned to the partition.

6.1.2.2 struct CanTrcv_43_AE_TransceiverConfigType

CanTrcv_43_AE_TransceiverConfigType.

Definition at line 76 of file CanTrcv_43_AE_Types.h.

Data Fields

Type	Name	Description
const uint8	CanTrcv_u8SwTransceiverId	The Id of Transceiver channel configured by tool configuration.
const uint8	CanTrcv_u8HwTransceiverId	The Id of Transceiver channel mapped in Hardware.
const uint8	CanTrcv_u8CanIfTransceiverId	The Id referred by CanIf to CanTrcv.
const uint32	CanTrcv_WakeupSourceMask	The value report to EcuM for a wakeup event.
const uint16	CanTrcv_DemEventId	The value report to Dem for a bus error.
const boolean	CanTrcv_bWakeupByBusUsed	Specifies if wakeup is enabled on the transceiver.
const CanTrcv_TrcvModeType	CanTrcv_eInitState	The mode of Transceiver after Init.
const CanTrcv_43_AE_Ipw_TransceiverConfigType *const	CanTrcv_IpwTransceiverConfig	Pointer to Ipw configuration wrapped to specific IPV configuration.

6.1.3 Macro Definition Documentation

6.1.3.1 CANTRCV_43_AE_E_INVALID_TRANSCEIVER

```
#define CANTRCV_43_AE_E_INVALID_TRANSCEIVER
```

Development Error ID for API called with wrong parameter for the CAN transceiver.

Development errors. The Can module shall be able to detect the following errors and exceptions depending on its configuration (development/production). Development Errors shall indicate errors that are caused by erroneous usage of the Can module API. This covers API parameter checks and call sequence errors. Development error values are of type uint8.

Definition at line 86 of file CanTrcv_43_AE.h.

6.1.3.2 CANTRCV_43_AE_E_PARAM_POINTER

```
#define CANTRCV_43_AE_E_PARAM_POINTER
```

Development Error ID for API called with null pointer parameter.

Definition at line 88 of file CanTrcv_43_AE.h.

6.1.3.3 CANTRCV_43_AE_E_UNINIT

```
#define CANTRCV_43_AE_E_UNINIT
```

Development Error ID for API service used without initialization.

Definition at line 90 of file CanTrcv_43_AE.h.

6.1.3.4 CANTRCV_43_AE_E_TRCV_NOT_STANDBY

```
#define CANTRCV_43_AE_E_TRCV_NOT_STANDBY
```

Development Error ID for API service called in wrong transceiver operation mode (STANDBY expected)

Definition at line 92 of file CanTrcv_43_AE.h.

6.1.3.5 CANTRCV_43_AE_E_TRCV_NOT_NORMAL

```
#define CANTRCV_43_AE_E_TRCV_NOT_NORMAL
```

Development Error ID for API service called in wrong transceiver operation mode (NORMAL expected)

Definition at line 94 of file CanTrcv_43_AE.h.

6.1.3.6 CANTRCV_43_AE_E_PARAM_TRCV_WAKEUP_MODE

```
#define CANTRCV_43_AE_E_PARAM_TRCV_WAKEUP_MODE
```

Development Error ID for API service called with invalid parameter for TrcvWakeupMode.

Definition at line 96 of file CanTrcv_43_AE.h.

6.1.3.7 CANTRCV_43_AE_E_PARAM_TRCV_OPMODE

```
#define CANTRCV_43_AE_E_PARAM_TRCV_OPMODE
```

Development Error ID for API service called with invalid parameter for OpMode.

Definition at line 98 of file CanTrcv_43_AE.h.

6.1.3.8 CANTRCV_43_AE_E_BAUDRATE_NOT_SUPPORTED

```
#define CANTRCV_43_AE_E_BAUDRATE_NOT_SUPPORTED
```

Development Error ID for Configured baud rate is not supported by the transceiver.

Definition at line 100 of file CanTrcv_43_AE.h.

6.1.3.9 CANTRCV_43_AE_E_INIT_FAILED

```
#define CANTRCV_43_AE_E_INIT_FAILED
```

Development Error ID for Module initialization has failed, e.g. CanTrcv_Init() called with an invalid pointer in postbuild.

Definition at line 102 of file CanTrcv_43_AE.h.

6.1.3.10 CANTRCV_43_AE_E_NO_TRCV_CONTROL

```
#define CANTRCV_43_AE_E_NO_TRCV_CONTROL
```

Runtime Error ID when No/incorrect communication to transceiver.

Definition at line 105 of file CanTrcv_43_AE.h.

6.1.3.11 CANTRCV_43_AE_SID_INIT

```
#define CANTRCV_43_AE_SID_INIT
```

Service ID of CanTrcv_43_AE_Init.

Service ID (APIs) for Det reporting

Definition at line 111 of file CanTrcv_43_AE.h.

6.1.3.12 CANTRCV_43_AE_SID_SET_OPMODE

```
#define CANTRCV_43_AE_SID_SET_OPMODE
```

Service ID of CanTrcv_43_AE_SetOpMode.

Definition at line 113 of file CanTrcv_43_AE.h.

6.1.3.13 CANTRCV_43_AE_SID_GET_OPMODE

```
#define CANTRCV_43_AE_SID_GET_OPMODE
```

Service ID of CanTrcv_43_AE_GetOpMode.

Definition at line 115 of file CanTrcv_43_AE.h.

6.1.3.14 CANTRCV_43_AE_SID_GET_BUS_WU_REASON

```
#define CANTRCV_43_AE_SID_GET_BUS_WU_REASON
```

Service ID of CanTrcv_43_AE_GetBusWuReason.

Definition at line 117 of file CanTrcv_43_AE.h.

6.1.3.15 CANTRCV_43_AE_SID_GET_VERSION_INFO

```
#define CANTRCV_43_AE_SID_GET_VERSION_INFO
```

Service ID of CanTrcv_43_AE_GetVersionInfo.

Definition at line 119 of file CanTrcv_43_AE.h.

6.1.3.16 CANTRCV_43_AE_SID_SET_WAKEUP_MODE

```
#define CANTRCV_43_AE_SID_SET_WAKEUP_MODE
```

Service ID of CanTrcv_43_AE_SetWakeupMode.

Definition at line 121 of file CanTrcv_43_AE.h.

6.1.3.17 CANTRCV_43_AE_SID_CHECK_WAKEUP

```
#define CANTRCV_43_AE_SID_CHECK_WAKEUP
```

Service ID of CanTrcv_43_AE_CheckWakeup.

Definition at line 123 of file CanTrcv_43_AE.h.

6.1.3.18 CANTRCV_43_AE_SID_CHECK_WAKE_FLAG

```
#define CANTRCV_43_AE_SID_CHECK_WAKE_FLAG
```

Service ID of CanTrcv_43_AE_CheckWakeFlag.

Definition at line 125 of file CanTrcv_43_AE.h.

6.1.3.19 CANTRCV_43_AE_SID_DEINIT

```
#define CANTRCV_43_AE_SID_DEINIT
```

Service ID of CanTrcv_43_AE_DeInit.

Definition at line 127 of file CanTrcv_43_AE.h.

6.1.4 Enum Reference

6.1.4.1 CanTrcv_43_AE_eDriverStatusType

```
enum CanTrcv_43_AE_eDriverStatusType
```

CanTrcv_43_AE_eDriverStatusType.

Driver status helps to prevent double driver initialization.

Enumerator

CANTRCV_43_AE_NOT_ACTIVE	Driver not initialized
CANTRCV_43_AE_ACTIVE	Driver ready

Definition at line 137 of file CanTrcv_43_AE.h.

6.1.5 Function Reference

6.1.5.1 CanTrcv_43_AE_GetVersionInfo()

```
void CanTrcv_43_AE_GetVersionInfo (
            Std_VersionInfoType * VersionInfo )
```

CAN transceiver driver get version info function. SID is 0x04.

Returns the version information of this module.

Parameters

out	*versioninfo*	Pointer to where to store the version information of this module.

Returns

void

6.1.5.2 CanTrcv_43_AE_Init()

```
void CanTrcv_43_AE_Init (
            const CanTrcv_43_AE_ConfigType * ConfigPtr )
```

Initializes CanTrcv module. SID 0x00.

Initializes all transceivers configured in ConfigPtr parameter. The CANTRCV module shall be initialized by CanTrcv_43_AE_Init() service call during the start-up.

Parameters

in	*ConfigPtr*	Pointer to driver configuration structure.

Returns

void

Precondition

CanTrcv_43_AE_Init shall be called at most once during runtime.

Postcondition

CanTrcv_43_AE_Init shall initialize all the transceivers and set the driver in READY state.

6.1.5.3 CanTrcv_43_AE_SetOpMode()

```
Std_ReturnType CanTrcv_43_AE_SetOpMode (
            uint8 Transceiver,
            CanTrcv_TrcvModeType OpMode )
```

Sets the mode of the Transceiver to the value OpMode. SID 0x01.

Puts the device in one of following modes: normal, standby, sleep.

Parameters

in	*Transceiver*	Index of the transceiver.
out	*OpMode*	Desired operating mode.

Returns

Std_ReturnType Result of the transition.

Return values

E_OK	Operation executed successfully.
E_NOT_OK	Operation failed.

Precondition

CanTrcv module should be initialized before calling the CanTrcv_43_AE_SetOpMode method.

Postcondition

CanTrcv_43_AE_SetOpMode shall set the transceiver in the desired state.

6.1.5.4 CanTrcv_43_AE_GetOpMode()

```
Std_ReturnType CanTrcv_43_AE_GetOpMode (
            uint8 Transceiver,
            CanTrcv_TrcvModeType * OpMode )
```

Gets current operation mode. SID 0x02.

Gets the mode of the Transceiver and returns it in OpMode. The device is in one of following modes: normal, standby, sleep.

Parameters

in	*Transceiver*	CAN transceiver ID.
out	*OpMode*	Current operating mode.

Returns

Std_ReturnType Result of the transition.

Return values

E_OK	Operation executed successfully.
E_NOT_OK	Operation failed.

Precondition

CanTrcv module should be initialized before calling the CanTrcv_43_AE_GetOpMode method.

Postcondition

CanTrcv_43_AE_GetOpMode shall return the currently working mode of the transceiver.

6.1.5.5 CanTrcv_43_AE_GetBusWuReason()

```
Std_ReturnType CanTrcv_43_AE_GetBusWuReason (
            uint8 Transceiver,
            CanTrcv_TrcvWakeupReasonType * Reason )
```

Gets the wakeup reason of the Transceiver and returns it in parameter Reason. SID 0x03.

The device can be woken up by Wake Up Pattern

Parameters

in	*Transceiver*	CAN transceiver to which API call has to be applied.
out	*Reason*	Pointer to wake up reason.

Returns

Std_ReturnType Result of the transition.

Return values

E_OK	Operation executed successfully.
E_NOT_OK	Operation failed.

Precondition

CanTrcv module should be initialized before calling the CanTrcv_43_AE_GetBusWuReason method.

Postcondition

CanTrcv_43_AE_GetBusWuReason shall return wake up reason.

6.1.5.6 CanTrcv_43_AE_SetWakeupMode()

```
Std_ReturnType CanTrcv_43_AE_SetWakeupMode (
            uint8 Transceiver,
            CanTrcv_TrcvWakeupModeType TrcvWakeupMode )
```

Enables, disables or clears wake up events of the Transceiver according to TrcvWakeupMode. SID 0x05.

Enables, disables or clears wake up functionality. If WU mode is disabled all wake up sources and interrupts are off. If WU mode is enabled, all wake up sources and interrupts are set according to configuration. If WU mode is clear, pending wake up flag is cleared.

Parameters

in	*Transceiver*	CAN Transceiver ID.
in	*TrcvWakeupMode*	Mode of wake up functionality (enabled, disabled, cleared).

Returns

Std_ReturnType Result of the transition.

Return values

E_OK	Operation executed successfully.
E_NOT_OK	Operation failed.

Precondition

CanTrcv module should be initialized before calling the CanTrcv_43_AE_SetWakeupMode method.

Postcondition

CanTrcv_43_AE_SetWakeupMode shall return status of the transceiver.

6.1.5.7 CanTrcv_43_AE_CheckWakeup()

```
Std_ReturnType CanTrcv_43_AE_CheckWakeup (
            uint8 Transceiver )
```

Service is called by underlying CANIF in case a wake up interrupt is detected. SID 0x07.

Reads wake up source from the device and reports it to ECUM.

Parameters

in	*Transceiver*	CAN transceiver ID.

Returns

Std_ReturnType Result of the transition.

Return values

E_OK	Operation executed successfully.
E_NOT_OK	Operation failed.

Precondition

CanTrcv module should be initialized before calling the CanTrcv_43_AE_CheckWakeup method.

Postcondition

CanTrcv_43_AE_CheckWakeup shall read and report wake up reason.

6.1.5.8 CanTrcv_43_AE_CheckWakeFlag()

```
Std_ReturnType CanTrcv_43_AE_CheckWakeFlag (
            uint8 Transceiver )
```

Requests to check the status of the wakeup flag from the transceiver hardware. SID 0x0E.

Checks wake up event and if WU occurred, reports it.

Parameters

in	*Transceiver*	CAN transceiver ID.

Returns

Std_ReturnType Result of the transition.

Return values

E_OK	Operation executed successfully.
E_NOT_OK	Operation failed.

Precondition

CanTrcv module should be initialized before calling the CanTrcv_43_AE_CheckWakeFlag method.

Postcondition

CanTrcv_43_AE_CheckWakeFlag shall check for wake up event, and if such event occurred, report it.

6.1.5.9 CanTrcv_43_AE_DeInit()

```
void CanTrcv_43_AE_DeInit (
            void  )
```

De-initializes the CanTrcv module. SID 0x10.

De-initialize all the transceivers. The CANTRCV module shall be de-initialized by CanTrcv_43_AE_DeInit() service call. This routine is called by:

- CanIf or an upper layer according to Autosar requirements.

Parameters

in	*None*	

Returns

void

Precondition

Before transceiver de-initalization, the driver must be initialized and the transceivers must not be in Start state.

Postcondition

CanTrcv_43_AE_DeInit shall de-initialize all the transceivers and set the driver in UNINIT state.

6.1.6 Variable Documentation

6.1.6.1 CanTrcv_u32CoreID

```
const uint32 CanTrcv_u32CoreID
```

Configuration Core ID.

Definition at line 150 of file CanTrcv_43_AE.h.

6.1.6.2 CanTrcv_ppTransceivers

```
const CanTrcv_43_AE_TransceiverConfigType* const* CanTrcv_ppTransceivers
```

Pointer to all Tranceiver channels asigned to the partition.

Definition at line 152 of file CanTrcv_43_AE.h.

6.1.6.3 CanTrcv_u8SwTransceiverId

```
const uint8 CanTrcv_u8SwTransceiverId
```

The Id of Transceiver channel configured by tool configuration.

Definition at line 79 of file CanTrcv_43_AE_Types.h.

6.1.6.4 CanTrcv_u8HwTransceiverId

```
const uint8 CanTrcv_u8HwTransceiverId
```

The Id of Transceiver channel mapped in Hardware.

Definition at line 81 of file CanTrcv_43_AE_Types.h.

6.1.6.5 CanTrcv_u8CanIfTransceiverId

```
const uint8 CanTrcv_u8CanIfTransceiverId
```

The Id referred by CanIf to CanTrcv.

Definition at line 83 of file CanTrcv_43_AE_Types.h.

6.1.6.6 CanTrcv_WakeupSourceMask

```
const uint32 CanTrcv_WakeupSourceMask
```

The value report to EcuM for a wakeup event.

Definition at line 92 of file CanTrcv_43_AE_Types.h.

6.1.6.7 CanTrcv_DemEventId

```
const uint16 CanTrcv_DemEventId
```

The value report to Dem for a bus error.

Definition at line 96 of file CanTrcv_43_AE_Types.h.

6.1.6.8 CanTrcv_bWakeupByBusUsed

```
const boolean CanTrcv_bWakeupByBusUsed
```

Specifies if wakeup is enabled on the transceiver.

Definition at line 99 of file CanTrcv_43_AE_Types.h.

6.1.6.9 CanTrcv_eInitState

```
const CanTrcv_TrcvModeType CanTrcv_eInitState
```

The mode of Transceiver after Init.

Definition at line 101 of file CanTrcv_43_AE_Types.h.

6.1.6.10 CanTrcv_IpwTransceiverConfig

```
const CanTrcv_43_AE_Ipw_TransceiverConfigType* const CanTrcv_IpwTransceiverConfig
```

Pointer to Ipw configuration wrapped to specific IPV configuration.

Definition at line 103 of file CanTrcv_43_AE_Types.h.

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

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

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