

User Manual

for S32K1_S32M24X CAN_43_FLEXCAN Driver

Document Number: UM2CAN_43_FLEXCANASRR21-11 Rev0000R2.0.0 Rev. 1.0

1 Revision History	2
2 Introduction	3
2.1 Supported Derivatives	3
2.2 Overview	4
2.3 About This Manual	5
2.4 Acronyms and Definitions	6
2.5 Reference List	6
3 Driver	7
3.1 Requirements	7
3.2 Driver Design Summary	7
3.3 Hardware Resources	8
3.4 Deviations from Requirements	8
3.5 Driver Limitations	25
3.6 Driver usage and configuration tips	26
3.6.1 Driver APIs usage	26
3.6.2 Can Hardware Object Handle (HOH) configuration	27
3.6.3 Legacy Rx FIFO Configuration	27
3.6.4 RxFifo Configuration with DMA support	28
3.7 Runtime errors	31
3.8 Symbolic Names Disclaimer	32
4 Tresos Configuration Plug-in	33
4.1 Module Can	36
4.2 Container CanGeneral	36
4.3 Parameter CanDevErrorDetect	37
4.4 Parameter CanEnableUserModeSupport	37
4.5 Parameter CanMulticoreSupport	38
4.6 Parameter CanEnableSecurityEventReporting	38
4.7 Parameter CanVersionInfoApi	39
4.8 Parameter CanIndex	39
4.9 Parameter CanMainFunctionBusoffPeriod	40
4.10 Parameter CanMainFunctionWakeupPeriod	40
4.11 Parameter CanMainFunctionModePeriod	41
4.12 Parameter CanMultiplexedTransmission	41
4.13 Parameter CanTimeoutMethod	42
4.14 Parameter CanTimeoutDuration	43
4.15 Parameter CanLPduReceiveCalloutFunction	43
4.16 Parameter CanMBCountExtensionSupport	44
4.17 Parameter CanApiEnableMbAbort	44

4.18 Parameter CanSetBaudrateApi	45
4.19 Parameter CanEnableDualClockMode	45
4.20 Parameter CanListenOnlyModeApi	46
4.21 Parameter CanGlobalTimeSupport	46
4.22 Reference CanEcucPartitionRef	47
4.23 Reference CanOsCounterRef	47
4.24 Reference CanSupportTTCANRef	48
4.25 Container CanMainFunctionRWPeriods	48
4.26 Parameter CanMainFunctionPeriod	48
4.27 Container CanConfigSet	49
4.28 Container CanController	49
4.29 Parameter CanHwChannel	50
4.30 Parameter CanControllerActivation	50
4.31 Parameter CanControllerBaseAddress	51
4.32 Parameter CanControllerId	51
4.33 Parameter CanRxProcessing	52
4.34 Parameter CanTxProcessing	52
4.35 Parameter CanBusoffProcessing	53
4.36 Parameter CanWakeupFunctionalityAPI	53
4.37 Parameter CanWakeupProcessing	54
4.38 Parameter CanWakeupSupport	54
4.39 Parameter CanLoopBackMode	55
4.40 Parameter CanAutoBusOffRecovery	55
4.41 Parameter CanTrippleSamplingEnable	56
4.42 Parameter CanControllerPrExcEn	56
4.43 Parameter CanControllerEdgeFilter	57
4.44 Parameter CanControllerFdISO	57
4.45 Parameter CanClockFromBus	58
4.46 Reference CanControllerDefaultBaudrate	58
4.47 Reference CanControllerEcucPartitionRef	59
4.48 Reference CanCpuClockRef	59
4.49 Reference CanCpuClockRefAlternate	60
4.50 Reference CanWakeupSourceRef	60
4.51 Container CanControllerBaudrateConfig	62
4.52 Parameter CanAdvancedSetting	62
4.53 Parameter CanBusLength	63
4.54 Parameter CanPropDelayTranceiver	63
4.55 Parameter CanTxArbitrationStartDelay	64
4.56 Parameter CanControllerPrescaller	65
4.57 Parameter CanControllerPrescallerAlternate	65

4.58 Parameter CanControllerBaudRateConfigID	66
4.59 Parameter CanControllerBaudRate	66
4.60 Parameter CanControllerSyncSeg	67
4.61 Parameter CanControllerPropSeg	67
4.62 Parameter CanControllerSeg1	68
4.63 Parameter CanControllerSeg2	69
4.64 Parameter CanControllerSyncJumpWidth	69
4.65 Container CanControllerFdBaudrateConfig	70
4.66 Parameter CanControllerFdBaudRate	70
4.67 Parameter CanControllerFdSyncSeg	71
4.68 Parameter CanControllerPropSeg	71
4.69 Parameter CanControllerSeg1	72
4.70 Parameter CanControllerSeg2	72
4.71 Parameter CanControllerSyncJumpWidth	73
4.72 Parameter CanControllerSspOffset	73
4.73 Parameter CanControllerFdPrescaler	73
4.74 Parameter CanControllerPrescalerAlternateFd	74
4.75 Parameter CanControllerTxBitRateSwitch	75
4.76 Container CanTTController	75
4.77 Parameter CanTTControllerApplWatchdogLimit	75
4.78 Parameter CanTTControllerCycleCountMax	76
4.79 Parameter CanTTControllerExpectedTxTrigger	76
4.80 Parameter CanTTControllerExternalClockSynchronisation	77
4.81 Parameter CanTTControllerGlobalTimeFiltering	77
4.82 Parameter CanTTControllerInitialRefOffset	78
4.83 Parameter CanTTControllerInterruptEnable	78
4.84 Parameter CanTTControllerLevel2	79
4.85 Parameter CanTTControllerNTUConfig	79
4.86 Parameter CanTTControllerOperationMode	80
4.87 Parameter CanTTControllerSyncDeviation	80
4.88 Parameter CanTTControllerTURRestore	80
4.89 Parameter CanTTControllerTimeMaster	81
4.90 Parameter CanTTControllerTimeMasterPriority	81
4.91 Parameter CanTTControllerTxEnableWindowLength	82
4.92 Parameter CanTTControllerWatchTriggerGapTimeMark	82
4.93 Parameter CanTTControllerWatchTriggerTimeMark	83
4.94 Parameter CanTTIRQProcessing	83
4.95 Reference CanTTControllerEcucPartitionRef	84
4.96 Container CanRamBlock	84
4.97 Container CanRxFiFo	85

4.98 Container CanHardwareObject	85
4.99 Parameter CanFdPaddingValue	86
4.100 Parameter CanObjectPayloadLength	86
4.101 Parameter CanHandleType	87
4.102 Parameter CanIdType	87
4.103 Parameter CanObjectId	89
4.104 Parameter CanObjectType	90
4.105 Parameter CanHardwareObjectUsesPolling	90
4.106 Parameter CanTriggerTransmitEnable	90
4.107 Parameter CanHwObjectUsesBlock	91
4.108 Parameter CanHwObjectCount	91
4.109 Reference CanControllerRef	92
4.110 Reference CanMainFunctionRWPeriodRef	92
4.111 Container CanHwFilter	93
4.112 Parameter CanHwFilterCode	94
4.113 Parameter CanHwFilterMask	94
4.114 Container CanTTHardwareObjectTrigger	95
4.115 Parameter CanTTHardwareObjectBaseCycle	95
4.116 Parameter CanTTHardwareObjectCycleRepetition	96
4.117 Parameter CanTTHardwareObjectTimeMark	96
4.118 Parameter CanTTHardwareObjectTriggerId	97
4.119 Parameter CanTTHardwareObjectTriggerType	97
4.120 Container CommonPublishedInformation	98
4.121 Parameter ArReleaseMajorVersion	98
4.122 Parameter ArReleaseMinorVersion	99
4.123 Parameter ArReleaseRevisionVersion	99
4.124 Parameter ModuleId	100
4.125 Parameter SwMajorVersion	100
4.126 Parameter SwMinorVersion	101
4.127 Parameter SwPatchVersion	101
4.128 Parameter VendorApiInfix	102
4.129 Parameter VendorId	103
5 Module Index	104
5.1 Software Specification	104
6 Module Documentation	105
6.1 CAN_43_FLEXCAN_DRIVER	105
6.1.1 Detailed Description	105
6.1.2 Data Structure Documentation	106
6.1.3 Macro Definition Documentation	110

6.1.4 Enum Reference	110
6.1.5 Function Reference	112
6.2 FlexCAN	121
6.2.1 Detailed Description	121
6.2.2 Data Structure Documentation	127
6.2.3 Macro Definition Documentation	132
6.2.4 Types Reference	153
6.2.5 Enum Reference	153
6.2.6 Function Reference	159
6.3 Controller Area Network with Flexible Data Rate (FlexCAN)	179
6.4 FlexCAN_driver	181

Chapter 1

Revision History

Revision	Date	Author	Description
1.0	04.08.2023	NXP RTD Team	S32K1_S32M24X Real-Time Drivers AUTOSAR 4.4 & R21-11 Version 2.0.0

Chapter 2

Introduction

- [Supported Derivatives](#)
- [Overview](#)
- [About This Manual](#)
- [Acronyms and Definitions](#)
- [Reference List](#)

This User Manual describes NXP Semiconductor AUTOSAR CAN for S32K1_M24X. AUTOSAR CAN driver configuration parameters and deviations from the specification are described in Driver chapter of this document. AUTOSAR CAN driver requirements and APIs are described in the AUTOSAR CAN driver software specification document.

2.1 Supported Derivatives

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

- s32k116_qfn32
- s32k116_lqfp48
- s32k118_lqfp48
- s32k118_lqfp64
- s32k142_lqfp48
- s32k142_lqfp64
- s32k142_lqfp100
- s32k142w_lqfp48
- s32k142w_lqfp64
- s32k144_lqfp48

- s32k144_lqfp64 / MWCT1014S_lqfp64
- s32k144_lqfp100 / MWCT1014S_lqfp100
- s32k144_mapbga100
- s32k144w_lqfp48
- s32k144w_lqfp64
- s32k146_lqfp64
- s32k146_lqfp100 / MWCT1015S_lqfp100
- s32k146_mapbga100 / MWCT1015S_mapbga100
- s32k146_lqfp144
- s32k148_lqfp100
- s32k148_mapbga100 / MWCT1016S_mapbga100
- s32k148_lqfp144
- s32k148_lqfp176
- s32m241_lqfp64
- s32m242_lqfp64
- s32m243_lqfp64
- s32m244_lqfp64

All of the above microcontroller devices are collectively named as S32K1_S32M24X. Note: MWCT part numbers contain NXP confidential IP for Qi Wireless Power

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
ASM	Assembler
BSMI	Basic Software Make file Interface
CAN	Controller Area Network
C/CPP	C and C++ Source Code
CS	Chip Select
CTU	Cross Trigger Unit
DEM	Diagnostic Event Manager
DET	Development Error Tracer
DMA	Direct Memory Access
ECU	Electronic Control Unit
FIFO	First In First Out
LSB	Least Significant Bit
MCU	Micro Controller Unit
MIDE	Multi Integrated Development Environment
MSB	Most Significant Bit
N/A	Not Applicable
RAM	Random Access Memory
SIU	Systems Integration Unit
SWS	Software Specification
VLE	Variable Length Encoding
XML	Extensible Markup Language

2.5 Reference List

#	Title	Version
1	Specification of CAN Driver	AUTOSAR Release R21-11
2	Reference Manual	S32K1xx Series Reference Manual, Rev. 14, 09/2021 S32M24x Reference Manual, Rev. 2 Draft A, 05/2023
3	Errata	S32K116_0N96V Rev. 22/OCT/2021 S32K118_0N97V Rev. 22/OCT/2021 S32K142_0N33V Rev. 22/OCT/2021 S32K144_0N57U Rev. 22/OCT/2021 S32K144W_0P64A Rev. 22/OCT/2021 S32K146_0N73V Rev. 22/OCT/2021 S32K148_0N20V Rev. 22/OCT/2021 S32M244_P64A+P73G, Rev. 0 S32M242_N33V+P73G, Rev. 0, 6/2023
4	Datasheet	S32K1xx Data Sheet, Rev. 14, 08/2021 S32M2xx Data Sheet, Rev. 3 DraftA, 05/2023

Chapter 3

Driver

- [Requirements](#)
- [Driver Design Summary](#)
- [Hardware Resources](#)
- [Deviations from Requirements](#)
- [Driver Limitations](#)
- [Driver usage and configuration tips](#)
- [Runtime errors](#)
- [Symbolic Names Disclaimer](#)

3.1 Requirements

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

It has vendor-specific requirements and implementation.

3.2 Driver Design Summary

The S32K1_M24X contains up to 3 Controller Area Network (CAN) blocks. Which supports CAN FD. Each IPV_FlexCAN module is a full implementation of the CAN protocol specification, the CAN with Flexible Data rate (CAN FD) protocol and the CAN 2.0 version B protocol. The CAN protocol interface (CPI) sub-module manages the serial communication on the CAN bus, requesting RAM access for receiving and transmitting message frames, validating received messages and performing error handling. The message buffer management (MBM) sub-module handles message buffer selection for reception and transmission, taking care of arbitration and ID matching algorithms. The bus interface unit (BIU) sub-module controls the access to and from the internal interface bus, to establish connection to the CPU and other blocks. The IPV_FlexCAN has these major features:

- Flexible message buffers (MBs) of zero to eight bytes data length. With CAN_FD, this length is from 0 to 64 bytes. Some platform has support the selecting ISO/none-ISO.
- Individual Rx mask registers per message buffer.

- Powerful Legacy Rx FIFO ID filtering, capable of matching incoming IDs against either 128 Extended, 256 Standard, or 512 Partial (8 bits) IDs, with 128 individual masking capability.
- ListenOnly capability.
- Programmable loop-back mode supporting self-test operation.
- Maskable interrupts.
- Low power modes.
- Transceiver Delay Compensation feature when transmitting CAN FD messages at faster data rates.
- Timestamp of the Messages sent or received
- Supports Legacy Time stamp based on 16-bit free-running timer, with an optional external time tick(LPIT channel 0)
- RxFifo Supports DMA transfers depend on the Fifo type support single message transfer.
- Supports Pretended Networking functionality in low power: Stop mode.

3.3 Hardware Resources

The CAN controller number mapping between Reference Manual/microcontroller and our XDM configuration can be done by using the following:

- Reference Manual naming = Configuration Naming
- FlexCAN_0 = FlexCAN_0
- FlexCAN_1 = FlexCAN_1
- FlexCAN_2 = FlexCAN_2

3.4 Deviations from Requirements

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

Requirement	Status	Description	Notes
SWS_Can_00362	N/S	If development error detection for the Can module is enabled: The function Can_CheckWakeup shall raise the error CAN_E_UNINIT if the driver is not yet initialized.	The external application shall call Can_CheckWakeup function only after driver initialization.
SWS_Can_00363	N/S	If development error detection for the Can module is enabled: The function Can_CheckWakeup shall raise the error CAN_E_PARAM←_CONTROLLER if the parameter Controller is out of range.	Platform does not support a WAKE UP mode.

Requirement	Status	Description	Notes
SWS_Can_00447	N/S	Icu_DisableNotification shall be called when "external" Can controllers have been transitioned to STOPPED state.	All controllers are on chip. Our SoC does not support wakeup portde
SWS_Can_00240	N/S	The Mcu module (SPAL see [REF]) shall configure register settings that are 'shared' with other modules.	Not a requirement for CAN module
SWS_Can_00242	N/S	If an off-chip CAN controller is used[3], the Can module shall use services of other MCAL drivers (e.g. SPI).	All current controllers are onchip.
SWS_Can_00244	N/S	The Can module shall use the synchronous APIs of the underlying M↔CAL drivers and shall not provide callback functions that can be called by the MCAL drivers.	All current controllers are onchip.↔ No callback can be called by other drivers.
SWS_Can_00257	N/S	When the CAN hardware supports sleep mode and is triggered to transition into SLEEP state, the Can module shall set the controller to the S↔LEEP state from which the hardware can be woken over CAN Bus.	Controller not support sleep mode.
SWS_Can_00270	N/S	On hardware wakeup (triggered by a wake-up event from CAN bus), the CAN controller shall transition into the state STOPPED.	Platform does not support a WAKE UP mode.
SWS_Can_00271	N/S	On hardware wakeup (triggered by a wake-up event from CAN bus), the Can module shall call the function EcuM_CheckWakeup either in interrupt context or in the context of Can_MainFunction_Wakeup.	Platform does not support a WAKE UP mode.
SWS_Can_00269	N/S	The Can module shall not further process the L-PDU that caused a wake-up.	Platform does not support a WAKE UP mode.
SWS_Can_00048	N/S	In case of a CAN bus wake-up during sleep transition, the function Can_SetControllerMode(CA↔N_CS_STOPPED) shall return E↔_NOT_OK.	For HW not support on-chip wakeup, this event will not occur.
SWS_Can_00274	N/S	The Can module shall disable or suppress automatic bus-off recovery.	It is replaced by CPR_RTD_↔00061: The CAN driver configuration shall allow automatic and also manual CAN bus-off recovery.

Requirement	Status	Description	Notes
SWS_CAN_00490	N/S	Controllers that do not support a hardware FIFO often provide the capabilities to implement a shadow buffer mechanism, where additional hardware objects take over when the primary hardware object is busy. The number of hardware objects is configured via "CanHwObjectCount".	Hardware support FIFO, so this requirement is not applicable.
SWS_Can_00299	N/S	The Can module shall copy the L \leftrightarrow SDU in a shadow buffer after reception, if the RX buffer cannot be protected (locked) by CAN Hardware against overwriting by a newly received message.	The HW support lock of the received Buffers
SWS_Can_00300	N/S	The Can module shall copy the L \leftrightarrow SDU in a shadow buffer, if the CAN Hardware is not globally accessible.	Message buffers are globally accessible are located in system ram.
SWS_Can_00364	N/S	If the ISR for wakeup events is called, it shall call EcuM_Check \leftrightarrow Wakeup in turn. The parameter passed to EcuM_CheckWakeup shall be the ID of the wakeup source referenced by the CanWakeupSourceRef configuration parameter.	Only for platform support wake up
SWS_Can_00294	N/S	The function Can_SetController \leftrightarrow Mode shall disable the wake-up interrupt, while checking the wake-up status.	Platform does not support a WAKE UP mode.
SWS_Can_00360	N/S	Service name: - Can_CheckWakeup - Syntax: - Std_ReturnType Can_CheckWakeup(uint8 Controller) - Service ID[hex]: - 0x0b - Sync/Async: - Synchronous - Reentrancy: - Non Reentrant - Parameters (in): - Controller - Controller to be checked for a wakeup. - Parameters (inout): - None - Parameters (out): - None - Return value: - Std_ReturnType - E_OK: API call has been accepted - E_NOT_OK: API call has not been accepted - Description: - This function checks if a wakeup has occurred for the given controller. - Available via: - Can.h -	The external application shall assure that Can_CheckWakeup does not preempt and is not preempted by any other CAN driver API using the same controller parameter. The external application shall assure that Can_CheckWakeup does not preempt itself.

Requirement	Status	Description	Notes
SWS_Can_00361	N/S	The function Can_CheckWakeup shall check if the requested CAN controller has detected a wakeup. If a wakeup event was successfully detected, reporting shall be done to EcuM via API EcuM_SetWakeup↔Event.	Platform does not support a WAKE UP mode.
SWS_CAN_00485	N/S	The function Can_CheckWakeup shall be pre compile time configurable On/Off by the configuration parameter: CanWakeup↔FunctionalityAPI	This requirement can only apply for platform support on-chip wakeup. Otherwise, it will be always Off.
SWS_Can_00445	N/S	Can driver shall use the following APIs provided by Icu driver, to enable and disable the wakeup event notification:Icu_Enable↔NotificationIcu_DisableNotification	Not implemented for OnChip platform
SWS_Can_00446	N/S	Icu_EnableNotification shall be called when "external" Can controllers have been transitioned to SLEEP state.	Not implemented for OnChip platform
SWS_Can_00110	N/S	There is no requirement regarding the execution order of the CAN main processing functions.	this is not a requirement
SWS_Can_00228	N/S	Service name: - Can_Main↔Function_Wakeup - Syntax: - void Can_MainFunction_Wakeup(void) - Service ID[hex]: - 0x0a - Description: - This function performs the polling of wake-up events that are configured statically as 'to be polled'. - Available via: - SchM_Can.h -	Not implemented for platform not support hardware wake-up
SWS_Can_00112	N/S	The function Can_MainFunction↔_Wakeup shall perform the polling of wake-up events that are configured statically as 'to be polled'.	Not implemented for platform not support hardware wake-up
SWS_Can_00185	N/S	The Can module may implement the function Can_MainFunction_↔Wakeup as empty define in case no polling at all is used.	Not implemented for platform not support hardware wake-up
SWS_Can_00999	N/S	These requirements are not applicable to this specification.	this is not a requirement

Requirement	Status	Description	Notes
ECUC_Can_00357	N/S	Name - CanMainFunctionWakeup Period - Parent Container - Can General - Description - This parameter describes the period for cyclic call to Can_MainFunction_Wakeup. Unit is seconds. - Multiplicity - 0..1 - Type - EcucFloat ParamDef - Range -]0 .. INF[- - 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 - -	Shall be supported only by platforms which include hardware wake-up supported
ECUC_Can_00430	N/S	Name - CanSupportTTCANRef - Parent Container - CanGeneral - Description - The parameter refers to CanIfSupportTTCAN parameter in the CAN Interface Module configuration. The CanIfSupportTTCAN parameter defines whether TTCAN is supported. - Multiplicity - 1 - Type - Reference to [CanIfPrivateCfg] - Post-Build Variant Value - false - Value Configuration Class - Pre-compile time - X - All Variants - Link time - - - - Post-build time - - - - Scope / Dependency - scope: ECU -	This is a HW limitation, some platform dose not support TTCAN feature.
ECUC_Can_00382	N/S	Name - CanControllerBaseAddress - Parent Container - CanController - Description - Specifies the CAN controller base address. - Multiplicity - 1 - Type - EcucIntegerParamDef - Range - 0 .. 4294967295 - - Default value - - - Post-Build Variant Value - false - Value Configuration Class - Pre-compile time - X - All Variants - Link time - - - - Post-build time - - - - Scope / Dependency - scope: local -	Controller base address is taken from Base module

Requirement	Status	Description	Notes
ECUC_Can_00466	N/S	Name - CanWakeupFunctionality API - Parent Container - CanController - Description - Adds / removes the service Can_CheckWakeup() from the code. True: Can_CheckWakeup can be used. False: Can_CheckWakeup cannot be used. - 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: local : H/W should support the wakeup functionality to enable this parameter. -	Shall be supported only by platforms which include hardware wake-up supported
ECUC_Can_00319	N/S	Name - CanWakeupProcessing - Parent Container - CanController - Description - Enables / disables API Can_MainFunction_Wakeup() for handling wakeup events in polling mode. - Multiplicity - 1 - Type - EcucEnumerationParamDef - Range - INTERRUPT - Interrupt Mode of operation. - POLLING - Polling Mode of operation. - Post-Build Variant Value - false - Value Configuration Class - Pre-compile time - X - All Variants - Link time - - - Post-build time - - - Scope / Dependency - scope: local -	This parameter shall be hardcoded always to unchecked(disable) for following all platforms which does not include hardware wake-up supported.
ECUC_Can_00330	N/S	Name - CanWakeupSupport - Parent Container - CanController - Description - CAN driver support for wakeup over CAN Bus. - Multiplicity - 1 - Type - EcucBooleanParamDef - Default value - - - Post-Build Variant Value - false - Value Configuration Class - Pre-compile time - X - All Variants - Link time - - - Post-build time - - - Scope / Dependency - -	Shall be supported only by platforms which include hardware wake-up supported. Always has value = false

Requirement	Status	Description	Notes
ECUC_Can_00359	N/S	Name - CanWakeupSourceRef - Parent Container - CanController - Description - This parameter contains a reference to the Wakeup Source for this controller as defined in the ECU State Manager.Implementation Type: reference to EcuM_WakeupSource Type - 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: local -	This requirement is only applicable for platform support HW wake up.
ECUC_Can_00001	N/S	Container Name - CanTTCController - Description - CanTTCController is specified in the SWS TTCAN and contains the configuration parameters of the TTCAN controller(s) (which are needed in addition to the configuration parameters of the CAN controller(s)).This container is only included and valid if TTCAN is supported by the controller, enabled (see CanSupportTTCANRef, ECUC_Can_00430), and used. - Configuration Parameters -	This is a HW limitation, some platform dose not support TTCAN feature.
ECUC_Can_00139	N/S	Name - CanTTCControllerApplWatchdogLimit - Parent Container - CanTTCController - Description - Defines the maximum time period (unit is 256 times NTU) after which the application has to serve the watchdog. - Multiplicity - 1 - Type - EcuIntegerParamDef - Range - 0 .. 255 - - Default value - - - Post-Build Variant Value - true - Value Configuration Class - Pre-compile time - X - VARIANT-PRE-COMPILE - Link time - - - - Post-build time - X - VARIANT-POST-BUILD - Scope / Dependency - scope: ECU -	This is a HW limitation, some platform dose not support TTCAN feature.

Requirement	Status	Description	Notes
ECUC_Can_00138	N/S	Name - CanTTControllerCycleCountMax - Parent Container - CanTTController - Description - Defines the value for cycle_count_max. Allowed values:0x00: 1 basic cycle0x01: 2 basic cycles0x03: 4 basic cycles0x07: 8 basic cycles0x0F: 16 basic cycles0x1F: 32 basic cycles0x3F: 64 basic cycles - Multiplicity - 1 - Type - EcucIntegerParamDef - Range - 0 .. 63 - - Default value - - - Post-Build Variant Value - true - Value Configuration Class - Pre-compile time - X - VARIANT-PRE-COMPILE - Link time - - - - Post-build time - X - VARIANT-POST-BUILD - Scope / Dependency - scope: ECU -	This is a HW limitation, some platform dose not support TTCan feature.
ECUC_Can_00136	N/S	Name - CanTTControllerExpectedTxTrigger - Parent Container - CanTTController - Description - Number of expected_tx_trigger. - Multiplicity - 1 - Type - EcucIntegerParamDef - Range - 0 .. 255 - - Default value - - - Post-Build Variant Value - true - Value Configuration Class - Pre-compile time - X - VARIANT-PRE-COMPILE - Link time - - - - Post-build time - X - VARIANT-POST-BUILD - Scope / Dependency - scope: local -	This is a HW limitation, some platform dose not support TTCan feature.
ECUC_Can_00135	N/S	Name - CanTTControllerExternalClockSynchronisation - Parent Container - CanTTController - Description - Enables/disables the external clock synchronization.TRUE:External clock synchronization enabled.FALSE:External clock synchronization disabled.This parameter shall only be configurable if parameter CanTTControllerLevel2 equals TRUE. - Multiplicity - 1 - Type - EcucBooleanParamDef - Default value - - - Post-Build Variant Value - true - Value Configuration Class - Pre-compile time - X - VARIANT-PRE-COMPILE - Link time - - - - Post-build time - X - VARIANT-POST-BUILD - Scope / Dependency - scope: ECUdependency: CanTTControllerLevel2 (ECUC_Can_00131) -	This is a HW limitation, some platform dose not support TTCan feature.

Requirement	Status	Description	Notes
ECUC_Can_00134	N/S	<p>Name - CanTTControllerGlobal← TimeFiltering - Parent Container - CanTTController - Description - Enables/disables the global time filtering.TRUE:Global time filtering enabled.FALSE:Global time filtering disabled.This parameter shall only be configurable if parameter Can← TTControllerLevel2 equals TRUE. - Multiplicity - 1 - Type - Ecuc← BooleanParamDef - Default value - - - Post-Build Variant Value - true - Value Configuration Class - Pre-compile time - X - VARIANT-P← RE-COMPILE - Link time - - - - Post-build time - X - VARIANT-P← OST-BUILD - Scope / Dependency - scope: localdependency: CanT← TControllerLevel2 (ECUC_Can_← 00131) -</p>	This is a HW limitation, some palt-form dose not support TTCan feature.
ECUC_Can_00128	N/S	<p>Name - CanTTControllerInitial← RefOffset - Parent Container - Can← TTController - Description - Defines the initial value for ref trigger offset. - Multiplicity - 1 - Type - Ecuc← IntegerParamDef - Range - 0 .. 127 - - Default value - - - Post-Build Variant Value - true - Value Configura- tion Class - Pre-compile time - X - VARIANT-PRE-COMPILE - Link time - - - - Post-build time - X - VA← RIANT-POST-BUILD - Scope / De- pendency - scope: ECU -</p>	This is a HW limitation, some palt-form dose not support TTCan feature.

Requirement	Status	Description	Notes
ECUC_Can_00140	N/S	<p>Name - CanTTControllerInterrupt</p> <p>Enable - Parent Container - CanTTController</p> <p>Description - Enables/disables the respective interrupts.Bit Position set to 1: Enable respective interrupt.Bit Position set to 0: Disable respective interrupt.Bit Position / Interrupt Source: 10: Application Watchdog. 9: Watch Trigger reached. 8: Initialization Watch Trigger reached. 7: Change of Error Level. 6: Tx Overflow. 5: Tx Underflow. 4: Global Time Error. 3: Gap. 2: Start of Cycle. 1: Time Discontinuity. 0: Master State Change.Bit position "1: Time Discontinuity" and "4: Global Time Error" shall only be configurable if parameter CanTTControllerLevel2 equals TRUE.</p> <p>Multiplicity - 1</p> <p>Type - EcucIntegerParamDef</p> <p>Range - 0 .. 1023</p> <p>Default value - - -</p> <p>Post-Build Variant Value - true</p> <p>Value Configuration Class - Pre-compile time - X - VARIANT-PRE-COMPILE</p> <p>Link time - - - -</p> <p>Post-build time - X - VARIANT-POST-BUILD</p> <p>Scope / Dependency - scope: local</p> <p>dependency: CanTTControllerLevel2 (ECUC_Can_00131)</p>	This is a HW limitation, some platform dose not support TTCAN feature.
ECUC_Can_00131	N/S	<p>Name - CanTTControllerLevel2</p> <p>Parent Container - CanTTController</p> <p>Description - Defines whether Level 2 or Level 1 is used.TRUE: Level 2.FALSE: Level 1.If this parameter is set to FALSE then all parameters with dependency to CanTTControllerLevel2 need not be configured.</p> <p>Multiplicity - 1</p> <p>Type - EcucBooleanParamDef</p> <p>Default value - - -</p> <p>Post-Build Variant Value - true</p> <p>Value Configuration Class - Pre-compile time - X - VARIANT-PRE-COMPILE</p> <p>Link time - - - -</p> <p>Post-build time - X - VARIANT-POST-BUILD</p> <p>Scope / Dependency - scope: local</p>	This is a HW limitation, some platform dose not support TTCAN feature.

Requirement	Status	Description	Notes
ECUC_Can_00141	N/S	<p>Name - CanTTControllerNTUConfig - Parent Container - CanTTController - Description - Defines the config value for NTU (network time unit). Value given in microseconds. The value configured shall be greater than 0. Together with the local oscillator period, the TUR (time unit ratio) can be derived from the NTU. This parameter shall only be configurable if parameter CanTTControllerLevel2 equals TRUE. - Multiplicity - 1 - Type - EcucFloatParamDef - Range - [0 .. 100] - - Default value - - - Post-Build Variant Value - true - Value Configuration Class - Pre-compile time - X - VARIANT-PRE-COMPILE - Link time - - - - Post-build time - X - VARIANT-POST-BUILD - Scope / Dependency - scope: ECUC - Udependency: CanTTControllerLevel2 (ECUC_Can_00131) -</p>	This is a HW limitation, some platform dose not support TTCan feature.
ECUC_Can_00127	N/S	<p>Name - CanTTControllerOperationMode - Parent Container - CanTTController - Description - Defines the operation mode. - Multiplicity - 1 - Type - EcucEnumerationParamDef - Range - CAN_TT_EVENT_SYNC_TIME_TRIGGERED - Event-synchronized time triggered operation - CAN_TT_EVENT_SYNC_TIME_TRIGGERED - Event triggered operation (normal can operation without time schedule) - CAN_TT_TIME_TRIGGERED - Time triggered operation - Post-Build Variant Value - true - Value Configuration Class - Pre-compile time - X - VARIANT-PRE-COMPILE - Link time - - - - Post-build time - X - VARIANT-POST-BUILD - Scope / Dependency - scope: ECU -</p>	This is a HW limitation, some platform dose not support TTCan feature.

Requirement	Status	Description	Notes
ECUC_Can_00132	N/S	Name - CanTTControllerSync↔ Deviation - Parent Container - CanTTController - Description - Defines the maximum synchronization deviation: Given as a percentage value of the NTU (network time unit). The value configured shall be greater than 0. This parameter shall only be configurable if parameter CanTTControllerLevel2 equals TRUE. - Multiplicity - 1 - Type - EcucFloatParamDef - Range - [0 .. 100] - - Default value - - - Post-↔ Build Variant Value - true - Value Configuration Class - Pre-compile time - X - VARIANT-PRE-COM↔ PILE - Link time - - - - Post-build time - X - VARIANT-POST-↔ BUILD - Scope / Dependency - scope: local dependency: CanTT↔ ControllerLevel2 (ECUC_Can_↔ 00131) -	This is a HW limitation, some platform dose not support TTCan feature.
ECUC_Can_00129	N/S	Name - CanTTControllerTime↔ Master - Parent Container - CanT↔ TController - Description - Defines whether the controller acts as a potential time master. TRUE: Potential time master. FALSE: Time slave. - Multiplicity - 1 - Type - EcucBooleanParamDef - Default value - - - Post-Build Variant Value - true - Value Configuration Class - Pre-compile time - X - VARIANT-PRE-COMPILE - Link time - - - - Post-build time - X - VARIANT-POST-BUILD - Scope / Dependency - scope: ECU -	This is a HW limitation, some platform dose not support TTCan feature.
ECUC_Can_00130	N/S	Name - CanTTControllerTime↔ MasterPriority - Parent Container - CanTTController - Description - Defines the time master priority. - Multiplicity - 1 - Type - Ecuc↔ IntegerParamDef - Range - 0 .. 7 - - Default value - - - Post-Build Variant Value - true - Value Configuration Class - Pre-compile time - X - VARIANT-PRE-COMPILE - Link time - - - - Post-build time - X - VARIANT-POST-BUILD - Scope / Dependency - scope: ECU -	This is a HW limitation, some platform dose not support TTCan feature.

Requirement	Status	Description	Notes
ECUC_Can_00133	N/S	<p>Name - CanTTControllerTURRestore - Parent Container - CanTTController - Description - Enables/disables the TUR restore.</p> <p>Note that the value configured for TUR can be derived from the value configured for NTU and the local oscillator period.</p> <p>TRUE: TUR restore enabled. FALSE: TUR restore disabled.</p> <p>This parameter shall only be configurable if parameter CanTTControllerLevel2 equals TRUE.</p> <p>Multiplicity - 1 - Type - EcucBooleanParamDef - Default value - - - Post-Build Variant Value - true - Value Configuration Class - Pre-compile time - X - VARIANT-Pre-Compile - Link time - - - Post-build time - X - VARIANT-T-POST-BUILD - Scope / Dependency - scope: localdependency: CanTTControllerLevel2 (ECUC_Can_00131) -</p>	This is a HW limitation, some platform does not support TTCAN feature.
ECUC_Can_00137	N/S	<p>Name - CanTTControllerTxEnableWindowLength - Parent Container - CanTTController - Description - Length of the tx enable window given in CAN bit times.</p> <p>Definition parameter "CanTTControllerTxEnableWindowlength" is used such that: Length of enable window = CanTTControllerTxEnableWindowLength + 1</p> <p>Multiplicity - 1 - Type - EcucIntegerParamDef - Range - 1 .. 16 - - Default value - - - Post-Build Variant Value - true - Value Configuration Class - Pre-compile time - X - VARIANT-Pre-Compile - Link time - - - Post-build time - X - VARIANT-POST-BUILD - Scope / Dependency - scope: ECU -</p>	This is a HW limitation, some platform does not support TTCAN feature.

Requirement	Status	Description	Notes
ECUC_Can_00158	N/S	Name - CanTTControllerWatchTriggerGapTimeMark - Parent Container - CanTTController - Description - watch trigger time mark after a gap - Multiplicity - 1 - Type - EcucIntegerParamDef - Range - 0 .. 65535 - - Default value - - - Post-Build Variant Value - true - Value Configuration Class - Pre-compile time - X - VARIANT-PRE-COMPILE - Link time - - - Post-build time - X - VARIANT-POST-BUILD - Scope / Dependency - scope: local -	This is a HW limitation, some platform dose not support TTCan feature.
ECUC_Can_00157	N/S	Name - CanTTControllerWatchTriggerTimeMark - Parent Container - CanTTController - Description - watch trigger time mark - Multiplicity - 1 - Type - EcucIntegerParamDef - Range - 0 .. 65535 - - Default value - - - Post-Build Variant Value - true - Value Configuration Class - Pre-compile time - X - VARIANT-PRE-COMPILE - Link time - - - Post-build time - X - VARIANT-POST-BUILD - Scope / Dependency - scope: local -	This is a HW limitation, some platform dose not support TTCan feature.
ECUC_Can_00142	N/S	Name - CanTTIRQProcessing - Parent Container - CanTTController - Description - Enables / disables API Can_MainFunction_BusOff() for handling busoff events in polling mode. - Multiplicity - 1 - Type - EcucEnumerationParamDef - Range - INTERRUPT - Interrupt Mode of operation. - POLLING - Polling Mode of operation. - Post-Build Variant Value - true - Value Configuration Class - Pre-compile time - X - VARIANT-PRE-COMPILE - Link time - - - Post-build time - X - VARIANT-POST-BUILD - Scope / Dependency - scope: ECU -	This is a HW limitation, some platform dose not support TTCan feature.

Requirement	Status	Description	Notes
ECUC_Can_00002	N/S	Container Name - CanTTHardwareObjectTrigger - Description - CanTTHardwareObjectTrigger is specified in the SWS TTCAN and contains the configuration (parameters) of TTCAN triggers for Hardware Objects, which are additional to the configuration (parameters) of CAN Hardware Objects. This container is only included and valid if TTCAN is supported by the controller and, enabled (see CanSupportTTCANRef, ECUC_Can_00430), and used. - Configuration Parameters -	This is a HW limitation, some platform dose not support TTCan feature.
ECUC_Can_00147	N/S	Name - CanTTHardwareObjectBaseCycle - Parent Container - CanTTHardwareObjectTrigger - Description - Defines the cycle_offset. CanTTHardwareObjectBaseCycle must be not greater than cycle_count_max. - Multiplicity - 1 - Type - EcucIntegerParamDef - Range - 0 .. 63 - - Default value - - - Post-Build Variant Value - true - Value Configuration Class - Pre-compile time - X - VARIANT-PRE-COMPILE - Link time - - - Post-build time - X - VARIANT-POST-BUILD - OST-BUILD - Scope / Dependency - scope: ECU -	This is a HW limitation, some platform dose not support TTCan feature.
ECUC_Can_00148	N/S	Name - CanTTHardwareObjectCycleRepetition - Parent Container - CanTTHardwareObjectTrigger - Description - Defines the repeat_factor. CanTTHardwareObjectCycleRepetition shall be a power of two (2), greater than cycle_offset but not greater than cycle_count_max + 1. - Multiplicity - 1 - Type - EcucIntegerParamDef - Range - 1 .. 64 - - Default value - - - Post-Build Variant Value - true - Value Configuration Class - Pre-compile time - X - VARIANT-PRE-COMPILE - Link time - - - Post-build time - X - VARIANT-POST-BUILD - Scope / Dependency - scope: ECU -	This is a HW limitation, some platform dose not support TTCan feature.

Requirement	Status	Description	Notes
ECUC_Can_00146	N/S	Name - CanTTHardwareObject↵ TimeMark - Parent Container - CanTTHardwareObjectTrigger - De- scription - Defines the point in time, when the trigger will be activated.↵ Value is given in cycle time. - Mul- tiplicity - 1 - Type - EcucInteger↵ ParamDef - Range - 0 .. 65535 - - Default value - - - Post-Build Variant Value - true - Value Configuration Class - Pre-compile time - X - VA↵ RIANT-PRE-COMPILE - Link time - - - - Post-build time - X - VARIA↵ NT-POST-BUILD - Scope / Depen- dency - scope: local -	This is a HW limitation, some palt- form dose not support TTCan fea- ture.
ECUC_Can_00155	N/S	Name - CanTTHardwareObject↵ TriggerId - Parent Container - CanTTHardwareObjectTrigger - De- scription - Sequential number which allows separation of different TTC↵ AN triggers configured for one and the same hardware object. - Mul- tiplicity - 1 - Type - EcucInteger↵ ParamDef (Symbolic Name gener- ated for this parameter) - Range - 0 .. 63 - - Default value - - - Post-Build Variant Value - false - Value Config- uration Class - Pre-compile time - X - All Variants - Link time - - - - Post- build time - - - - Scope / Dependency - scope: local -	This is a HW limitation, some palt- form dose not support TTCan fea- ture.

Requirement	Status	Description	Notes
ECUC_Can_00145	N/S	<p>Name - CanTTHardwareObject←</p> <p>TriggerType - Parent Container</p> <p>- CanTTHardwareObjectTrigger</p> <p>- Description - Defines the type of the trigger associated with the hardware object. This parameter depends on plain CAN parameter CAN_OBJECT_TYPE.If CAN_OBJECT_TYPE equals RECEIVE then this parameter is fixed to CAN_TT_RX_TRIGGER.If CAN_OBJECT_TYPE equals TRANSMIT then one of the following literals is configurable←</p> <p>:CAN_TT_TX_REF_TRIGGER←</p> <p>ER,CAN_TT_TX_REF_TRIGGER←</p> <p>GER_GAP,CAN_TT_TX_TRIGGER←</p> <p>MERGED,CAN_TT_TX_TRIGGER←</p> <p>SINGLE,CAN_TT_TX_TRIGGER←</p> <p>EXCLUSIVE.</p> <p>- Multiplicity - 1 - Type - Ecuc←</p> <p>EnumerationParamDef - Range</p> <p>- CAN_TT_RX_TRIGGER - Trigger for verifying the successful reception of messages. - CAN_TT_TX_REF_TRIGGER - Trigger for transmitting the reference message. - CAN_TT_TX_REF_TRIGGER_GAP - Trigger for transmitting the reference message in case no event occurs after a gap. - CAN_TT_TX_TRIGGER_EXCLUSIVE - Trigger for transmitting a message in an exclusive time window. Note, that messages in an exclusive window are transmitted continuously, i.e. regardless whether the same message has been transmitted before, the message, which is currently available, will be transmitted every time the tx trigger occurs. - CAN_TT_TX_TRIGGER_MERGED - Trigger for transmitting a message inside a merged arbitration window (the last tx trigger in a merged arbitration window is of type CAN_TT_TX_TRIGGER_SINGLE). Note, that messages in an arbitration window are transmitted only, if new data is available. When the transmission was not successful, it will be repeated at the next tx trigger for this message. When the transmission was successful, this message will not be transmitted again at the next tx triggers until a new message for this tx trigger</p>	This is a HW limitation, some paltform dose not support TTCan feature.
24	S32K1	S32M24X CAN 43 FLEXCAN Driver	NXP Semiconductors

Requirement	Status	Description	Notes
ECUC_Can_00493	N/S	Name - CanTTCControllerEcuc PartitionRef - Parent Container - CanTTCController - Description - Maps the Time triggered CAN controller to zero or one ECUC partitions. The ECUC partition referenced is a subset of the ECUC partitions where the CAN driver is mapped to. Tags: atp.Status=draft Multiplicity - 0..1 Type - Reference to [EcucPartition] Post-Build Variant Multiplicity - false 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 -	This is a HW limitation, some platform does not support TTCan feature.
SWS_Can_00265	N/F	The function Can_SetControllerMode(CAN_CS_SLEEP) shall set the controller into sleep mode.	Only applicable for platform support hardware wake-up. Otherwise, only logical sleep is implemented.
SWS_Can_00398	N/F	The function Can_SetControllerMode shall use the system service GetCounterValue for timeout monitoring to avoid blocking functions.	Can Driver uses system services(Get - CounterValue,...) indirectly from OsIf Driver.
CPR_RTD_00666.can	N/S	An additional vendor specific API shall be provided to update the current time of the CAN controller Service name: Can_SetCurrentTime Syntax: Std_ReturnType Can_SetCurrentTime(uint8 ControllerId, const Can_TimeStampType* timeStampPtr) Sync/Async: Synchronous Reentrancy: Reentrant Parameters (in): ControllerId Index of the addresses CAN controller. Parameters (in): timeStampPtr target time stamp Return value: Std_ReturnType E_OK: successful E_NOT_OK: failed Description: Sets current time for the CAN controller	This requirement isn't properly applied to this platform.

3.5 Driver Limitations

- FlexCAN has two FIFO options, Legacy Rx FIFO and Enhanced Rx FIFO. However, they cannot be enabled at the same time.

- It is a hardware limitation that the Legacy FIFO feature must not be enabled when the CAN FD feature is enabled. It means the two features must not be used at the same time.
- The Can driver supports the reception Legacy FIFO engine whose size is fixed as 6-message deep due to the hardware limitation.
- The Can driver only supports eleven the API name of `Can_43_FLEXCAN_MainFunction_Write()` for processing transmitted MBs.
The APIs obey the following pattern `Can_43_FLEXCAN_MainFunction_Write_<CanMainFunctionRWPeriods.ShortName>()`, where `<CanMainFunctionRWPeriods.ShortName>` is fixed by default name on configurator.
- The Can driver only supports eleven the API name of `Can_43_FLEXCAN_MainFunction_Read()` for processing received MBs.
The APIs obey the following pattern `Can_43_FLEXCAN_MainFunction_Read_<CanMainFunctionRWPeriods.ShortName>()`, where `<CanMainFunctionRWPeriods.ShortName>` is fixed by default name on configurator.
- The driver does not distinguish between Extended and Mixed MB types for receiving way: All Rx MBs configured as MIXED type will be converted to EXTENDED type. For transmission the CanIf will prepare the message ID with MSB bit set and based on this fact the Can module will send the message as STANDARD or EXTENDED type.
- `CanOsCounterRef` is not used anymore, the using of `OsCounter` is done by selecting `CanTimeoutMethod` to `OSIF_COUNTER_SYSTEM`.
- The base address for the controllers is not user input
- If DMA is used, the User need to input exactly Can Dma callback function name (detailed in the description of `CanLegacyFiFoDmaRef` node) and select exactly corresponding dma hardware channel in Mcl driver.
- Overflow, Warning notifications is not supported for Legacy Rx Fifo Dma.

3.6 Driver usage and configuration tips

This chapter describes how to configure for advanced features which are not (fully) described by AUTOSAR SWS (i.e NON-ASR features).

3.6.1 Driver APIs usage

- `Can_43_FLEXCAN_AbortMb()` API (Non Autosar) is defined if this feature is enabled by `CanApiEnableMbAbort` from the Tresos plugin.
- Multiplex transmission which is supported by `Can_43_FLEXCAN_Write()` API means to send a message from any Tx MB that is free to be used, in the range of the same `HWObjectID`. This means that several Hardware Objects can have the same `HWObjectID`. This feature can be used only if it's enabled by `CanMultiplexedTransmission` from the Tresos Plugin.
- Can driver support loop back mode to verify driver internally. In this case, it is necessary to configure filter in order to receive the transmit message.

- Can driver support listen only mode switch in order only to receive messages and don't be able to transmit, this can be activate at runtime by call `Can_43_FLEXCAN_ListenOnlyMode()` with state `CAN_43_FLEXCAN_LISTEN_MODE`, in order to revert to normal transmission mode need to call `Can_43_FLEXCAN_ListenOnlyMode()` with state `CAN_43_FLEXCAN_NORMAL_MODE`.

3.6.2 Can Hardware Object Handle (HOH) configuration The CanHardwareObject container in Tresos plugin is used to configure the operating for Rx and Tx MBs. The elements of this container is detailed in the chapter Can HardwareObject.

For Tx MBs (HTHs) the difference between Standard and Extended mode is done by the most significant bit of the Can ID.

For Rx MBs (HRHs) the MIXED message buffer type is handled as EXTENDED type. The platform support Legacy FIFO engine with 6 receive buffers storage scheme. This section describes the configuration in the advanced features.

3.6.3 Legacy Rx FIFO Configuration The receive-only FIFO is enabled for specific controller by asserting the FEN bit in the MCR register. The Legacy Rx Fifo configuration in the Tresos plugin is implemented by selection CanRxFiFo tab to CanLegacyFiFo in CanRxFiFo container.

When the Fifo is enabled, the memory region normally occupied by the first 6 MBs is normally reserved for use of the Fifo engine. The CPU can read the received frames sequentially, in the order they were received, by repeatedly accessing the MB0 structure.

The interrupts corresponding to MB0 to 5 have a different behavior when Rx Fifo in enabled. Bit 7 of the IFLAG1 becomes the “Fifo Overflow” flag, bit 6 becomes the “Fifo Warning” flag, bit 5 becomes the “Frame Available in Rx Fifo” flag and bits 4 to 0 are unused. If Legacy Rx Fifo is enabled for a specific controller, the user shall configure at least 1 hardware object which use that controller.

NOTE

The filters of Legacy Rx Fifo are configured in the CanHwFilter list in the first HOH referring to the controller enabling Legacy Rx Fifo.

The number of MBs used by Legacy Rx Fifo = 6 + (Number of Legacy FiFo Filter elements / 4)

Number of Legacy FiFo Filter elements depend on type of Id Acceptance Mode:

- FORMAT A: each 1 element of CanHwFilter list correspond to one Filter elements
- FORMAT B: each 2 consecutive elements of CanHwFilter list correspond to one Filter elements
- FORMAT C: each 4 consecutive elements of CanHwFilter list correspond to one Filter elements

The remain MBs of the controller = total MBs supported by the controller - (The number of MBs used by Legacy Rx Fifo)

If Legacy RxFifo is enabled the user can define proper handlers for overflow and warnings notification events.

Below is presented an example of a mapping between hardware objects and message buffers for a driver configuration which use multiple controllers and the Legacy RxFifo feature is enabled for all of them:

HRH0 id 0, controller A -> Legacy rx fifo of controller A

HRH1 id 1, controller A -> MB8

HRH2 id 2, controller A -> MB9

HRH3 id 3, controller B -> Legacy rx fifo of controller B

HRH4 id 4, controller B -> MB8

HRH5 id 5, controller B -> MB9

HTH0 id 6, controller A -> MB10

HTH1 id 7, controller B -> MB10

In order to understand the differences, below is presented an example of a mapping between hardware objects and message buffers for a driver configuration which use multiple controllers and the RxFifo feature is NOT enabled for any controller:

HRH0 id 0, controller A -> MB0

HRH1 id 1, controller A -> MB1

HRH2 id 2, controller A -> MB2

HRH3 id 3, controller B -> MB0

HRH4 id 4, controller B -> MB1

HRH5 id 5, controller B -> MB2

HTH0 id 6, controller A -> MB3

HTH1 id 7, controller B -> MB4

3.6.4 RxFifo Configuration with DMA support The CAN driver support DMA over Rx Fifo, this is supported together with MCL module. The Tresos project them must contain both plugins MCL and CAN. The CPU can be used only to configure the transmission and to process end of sequence notification (DMA_Can_← Callback), the transfer itself is triggered and done by the FlexCAN and DMA Hardware.

- **Step 1** Config DMA in the MCL module

To activate the DMA transfer from the RxFifo the user should configure the MCL module. By selecting adding a DMA Logic Instance inside which will configure a DMA Logic Channel. The configuration of DMA Logic Channel is done by selecting the Hardware Instance and Hardware Channel the same as the ones selected in the DMA Logic Instance. Then link the Interrupt and Error Interrupt Callback with the Can designed Interrupt. The interrupt name should be DMA_Can_Callback<InstanceIndex>, example DMA_← Can_Callback0 for FlexCAN0, DMA_Can_Callback1 for FlexCAN1

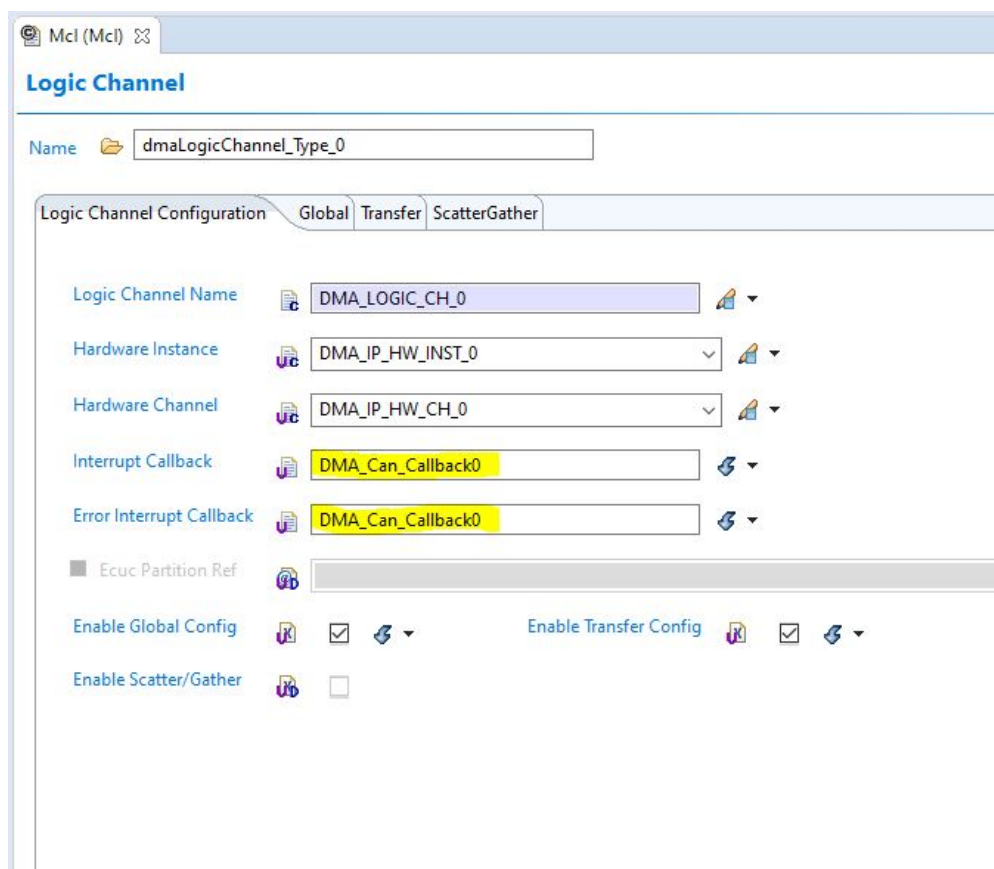


Figure 3.1 MCL Logic Channel Config

Then selection of the DMA Trigger Source of peripheral as FlexCAN instance need to be done on the Global Tab of Logic Channel. The DMAMUX0 Source field must select the peripheral desired from which to transfer the data messages, the user should enable the Enable Error Interrupt, Enable DMAMUX Source and disable the Enable DMA Request, Enable DMAMUX Trigger.

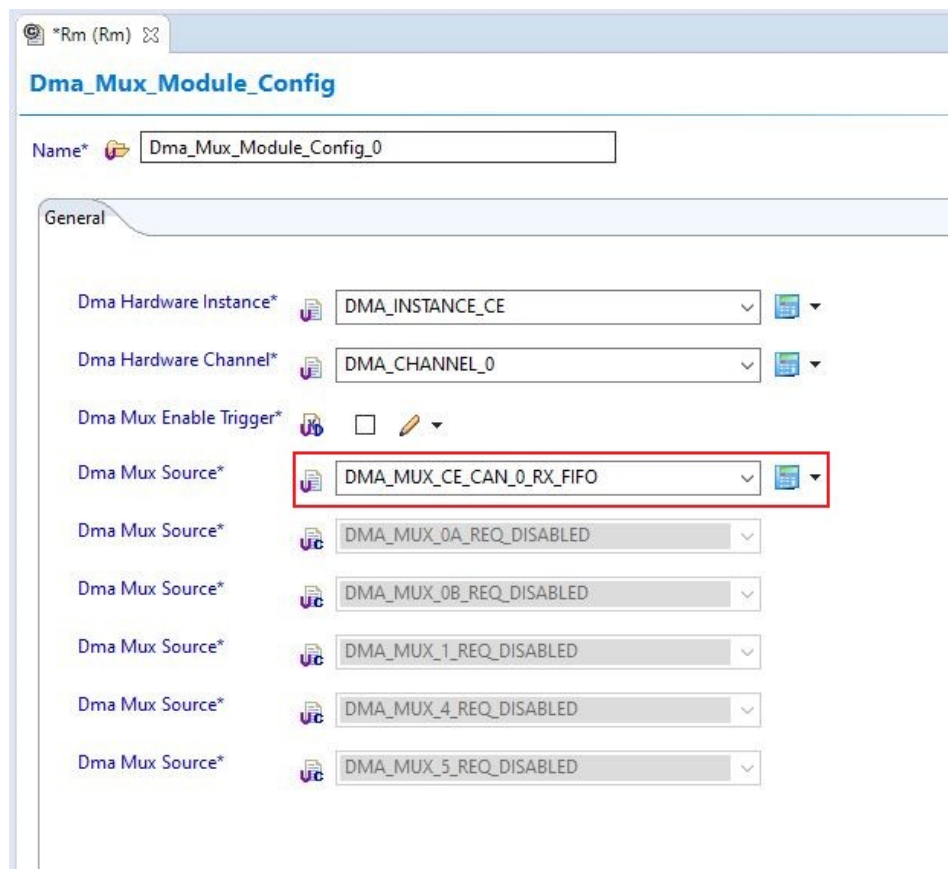


Figure 3.2 MCL Logic Channel Global

- **Step 2** Configure CanRxFiFo in the CAN module

For the usage of Legacy CanRxFiFo In order to activate this feature inside Controller specific by selecting CanRxFiFo Tab and on the CanRxFiFo selection need to be set CanLegacyFiFo from dropdown list, another parameter that need to be selected from this tab is CanLegacyFiFoDmaEnable as true. Then the field CanLegacyFiFoDmaRef should allow to select of MCL configured DMA Logic Channel previous configured at Step 1.

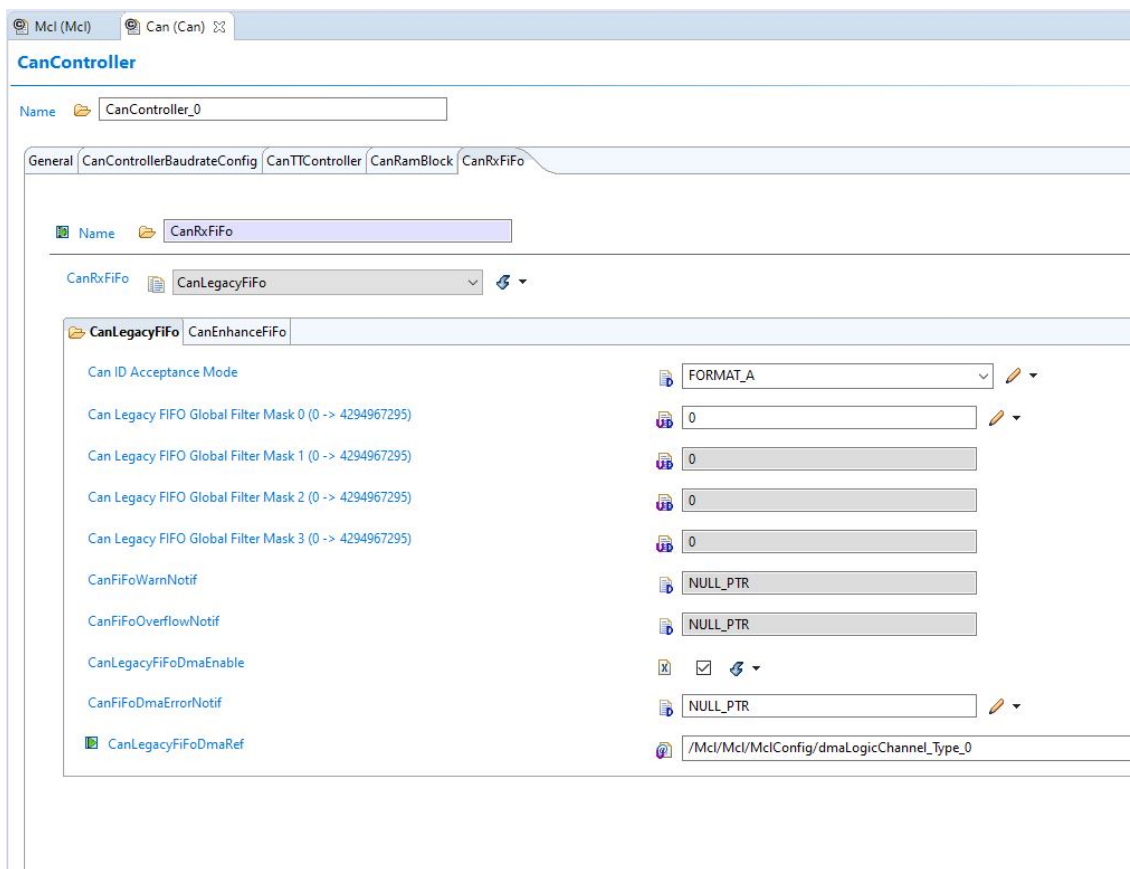


Figure 3.3 Legacy CanRxFiFo Config

3.7 Runtime errors

The driver generates the following DET errors at runtime.

Function	Error Code	Condition triggering the error
Can_43_FLEXCAN_Get↔VersionInfo, Can_43_FLEXCAN_Get↔ControllerErrorState, Can_43_FLEXCAN_Get↔ControllerMode, Can_43_FLEXCAN_Get↔ControllerRxError↔Counter, Can_43_FLEXCAN_Get↔ControllerTxErrorCounter, Can_43_FLEXCAN_Write	CAN_43_FLEXCAN_E_PAR↔AM_POINTER	API Service called with wrong parameter
Can_43_FLEXCAN_Write, Can_43_FLEXCAN_AbortMb	CAN_43_FLEXCAN_E_PAR↔AM_HANDLE	API Service called with wrong parameter
Can_43_FLEXCAN_Write	CAN_43_FLEXCAN_E_PAR↔AM_DATA_LENGTH	The length is exceeded Message Buffer's length

Function	Error Code	Condition triggering the error
Can_43_FLEXCAN_Set↵ Baudrate, Can_43_FLEXC↵ AN_SetControllerMode, Can_↵ 43_FLEXCAN_ListenOnlyMode, Can_43_FLEXCAN_Disable↵ ControllerInterrupts, Can_43_↵ FLEXCAN_EnableController↵ Interrupts, Can_43_FLEXC↵ AN_GetControllerErrorState, Can_43_FLEXCAN_Get↵ ControllerMode, Can_43_FL↵ EXCAN_GetControllerRxError↵ Counter, Can_43_FLEXCAN_↵ GetControllerTxErrorCounter	CAN_43_FLEXCAN_E_PAR↵ AM_CONTROLLER	The parameter Controller is out of range
All functions except main Functions	CAN_43_FLEXCAN_E_UNINIT	The driver is not yet initialized
Can_43_FLEXCAN_Init, Can_↵ 43_FLEXCAN_DeInit, Can_43↵ _FLEXCAN_SetControllerMode, Can_43_FLEXCAN_SetClock↵ Mode	CAN_43_FLEXCAN_E_TRA↵ NSITION	Invalid transition for the current mode
Can_43_FLEXCAN_SetBaudrate	CAN_43_FLEXCAN_E_PAR↵ AM_BAUDRATE	Parameter Baudrate has an invalid value
Can_43_FLEXCAN_Init	CAN_43_FLEXCAN_E_INIT↵ _FAILED	Invalid transition for the current mode
Can_43_FLEXCAN_Main↵ Function_Read	CAN_43_FLEXCAN_E_DAT↵ ALOST	Received CAN message is lost

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 [Can](#)
 - Container [CanGeneral](#)
 - * Parameter [CanDevErrorDetect](#)
 - * Parameter [CanEnableUserModeSupport](#)
 - * Parameter [CanMulticoreSupport](#)
 - * Parameter [CanEnableSecurityEventReporting](#)
 - * Parameter [CanVersionInfoApi](#)
 - * Parameter [CanIndex](#)
 - * Parameter [CanMainFunctionBusoffPeriod](#)
 - * Parameter [CanMainFunctionWakeupPeriod](#)
 - * Parameter [CanMainFunctionModePeriod](#)
 - * Parameter [CanMultiplexedTransmission](#)
 - * Parameter [CanTimeoutMethod](#)
 - * Parameter [CanTimeoutDuration](#)
 - * Parameter [CanLPduReceiveCalloutFunction](#)
 - * Parameter [CanMBCountExtensionSupport](#)
 - * Parameter [CanApiEnableMbAbort](#)
 - * Parameter [CanSetBaudrateApi](#)
 - * Parameter [CanEnableDualClockMode](#)
 - * Parameter [CanListenOnlyModeApi](#)
 - * Parameter [CanGlobalTimeSupport](#)
 - * Reference [CanEcucPartitionRef](#)
 - * Reference [CanOsCounterRef](#)
 - * Reference [CanSupportTTCANRef](#)
 - * Container [CanMainFunctionRWPeriods](#)
 - Parameter [CanMainFunctionPeriod](#)
 - Container [CanConfigSet](#)
 - * Container [CanController](#)
 - Parameter [CanHwChannel](#)
 - Parameter [CanControllerActivation](#)

- Parameter [CanControllerBaseAddress](#)
- Parameter [CanControllerId](#)
- Parameter [CanRxProcessing](#)
- Parameter [CanTxProcessing](#)
- Parameter [CanBusoffProcessing](#)
- Parameter [CanWakeupFunctionalityAPI](#)
- Parameter [CanWakeupProcessing](#)
- Parameter [CanWakeupSupport](#)
- Parameter [CanLoopBackMode](#)
- Parameter [CanAutoBusOffRecovery](#)
- Parameter [CanTrippleSamplingEnable](#)
- Parameter [CanControllerPrExcEn](#)
- Parameter [CanControllerEdgeFilter](#)
- Parameter [CanControllerFdISO](#)
- Parameter [CanClockFromBus](#)
- Reference [CanControllerDefaultBaudrate](#)
- Reference [CanControllerEcucPartitionRef](#)
- Reference [CanCpuClockRef](#)
- Reference [CanCpuClockRefAlternate](#)
- Reference [CanWakeupSourceRef](#)
- Container [CanControllerBaudrateConfig](#)
- Parameter [CanAdvancedSetting](#)
- Parameter [CanBusLength](#)
- Parameter [CanPropDelayTranceiver](#)
- Parameter [CanTxArbitrationStartDelay](#)
- Parameter [CanControllerPrescaler](#)
- Parameter [CanControllerPrescalerAlternate](#)
- Parameter [CanControllerBaudRateConfigID](#)
- Parameter [CanControllerBaudRate](#)
- Parameter [CanControllerSyncSeg](#)
- Parameter [CanControllerPropSeg](#)
- Parameter [CanControllerSeg1](#)
- Parameter [CanControllerSeg2](#)
- Parameter [CanControllerSyncJumpWidth](#)
- Container [CanControllerFdBaudrateConfig](#)
- Parameter [CanControllerFdBaudRate](#)
- Parameter [CanControllerFdSyncSeg](#)
- Parameter [CanControllerPropSeg](#)
- Parameter [CanControllerSeg1](#)
- Parameter [CanControllerSeg2](#)
- Parameter [CanControllerSyncJumpWidth](#)
- Parameter [CanControllerSspOffset](#)
- Parameter [CanControllerFdPrescaler](#)
- Parameter [CanControllerPrescalerAlternateFd](#)
- Parameter [CanControllerTxBitRateSwitch](#)
- Container [CanTTController](#)
- Parameter [CanTTControllerApplWatchdogLimit](#)

- Parameter [CanTTControllerCycleCountMax](#)
- Parameter [CanTTControllerExpectedTxTrigger](#)
- Parameter [CanTTControllerExternalClockSynchronisation](#)
- Parameter [CanTTControllerGlobalTimeFiltering](#)
- Parameter [CanTTControllerInitialRefOffset](#)
- Parameter [CanTTControllerInterruptEnable](#)
- Parameter [CanTTControllerLevel2](#)
- Parameter [CanTTControllerNTUConfig](#)
- Parameter [CanTTControllerOperationMode](#)
- Parameter [CanTTControllerSyncDeviation](#)
- Parameter [CanTTControllerTURRestore](#)
- Parameter [CanTTControllerTimeMaster](#)
- Parameter [CanTTControllerTimeMasterPriority](#)
- Parameter [CanTTControllerTxEnableWindowLength](#)
- Parameter [CanTTControllerWatchTriggerGapTimeMark](#)
- Parameter [CanTTControllerWatchTriggerTimeMark](#)
- Parameter [CanTTIRQProcessing](#)
- Reference [CanTTControllerEcucPartitionRef](#)
- Container [CanRamBlock](#)
- Container [CanRxFiFo](#)
- * Container [CanHardwareObject](#)
 - Parameter [CanFdPaddingValue](#)
 - Parameter [CanObjectPayloadLength](#)
 - Parameter [CanHandleType](#)
 - Parameter [CanIdType](#)
 - Parameter [CanObjectId](#)
 - Parameter [CanObjectType](#)
 - Parameter [CanHardwareObjectUsesPolling](#)
 - Parameter [CanTriggerTransmitEnable](#)
 - Parameter [CanHwObjectUsesBlock](#)
 - Parameter [CanHwObjectCount](#)
 - Reference [CanControllerRef](#)
 - Reference [CanMainFunctionRWPeriodRef](#)
 - Container [CanHwFilter](#)
 - Parameter [CanHwFilterCode](#)
 - Parameter [CanHwFilterMask](#)
 - Container [CanTTHardwareObjectTrigger](#)
 - Parameter [CanTTHardwareObjectBaseCycle](#)
 - Parameter [CanTTHardwareObjectCycleRepetition](#)

- Parameter [CanTTHardwareObjectTimeMark](#)
- Parameter [CanTTHardwareObjectTriggerId](#)
- Parameter [CanTTHardwareObjectTriggerType](#)
- Container [CommonPublishedInformation](#)
 - * Parameter [ArReleaseMajorVersion](#)
 - * Parameter [ArReleaseMinorVersion](#)
 - * Parameter [ArReleaseRevisionVersion](#)
 - * Parameter [ModuleId](#)
 - * Parameter [SwMajorVersion](#)
 - * Parameter [SwMinorVersion](#)
 - * Parameter [SwPatchVersion](#)
 - * Parameter [VendorApiInfix](#)
 - * Parameter [VendorId](#)

4.1 Module Can

This container holds the configuration of a single CAN Driver.

Included containers:

- [CanGeneral](#)
- [CanConfigSet](#)
- [CommonPublishedInformation](#)

Property	Value
type	ECUC-MODULE-DEF
lowerMultiplicity	1
upperMultiplicity	Infinite
postBuildVariantSupport	true
supportedConfigVariants	VARIANT-POST-BUILD, VARIANT-PRE-COMPILE

4.2 Container CanGeneral

This container holds the parameters related each CAN Driver Unit.

Included subcontainers:

- [CanMainFunctionRWPeriods](#)

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

4.3 Parameter CanDevErrorDetect

ECUC_Can_00064: 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-PRE-COMPILE: PRE-COMPILE VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.4 Parameter CanEnableUserModeSupport

When this parameter is enabled, the CAN module will adapt to run from User Mode, with the following measures:

(if applicable) a) configuring REG_PROT for the Can Controllers so that the registers under protection can be accessed from user mode by setting UAA bit in REG_PROT_GCR to 1

(if applicable) b) using 'call trusted function' stubs for all internal function calls that access registers requiring supervisor mode.

(if applicable) c) other module specific measures for more information, please see chapter 5.7 User Mode Support in IM

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

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

4.5 Parameter CanMulticoreSupport

Enable Maps Can driver to multiple EcuC partitions to make the modules API

available in this partition. The Can 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-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
default Value	false

4.6 Parameter CanEnableSecurityEventReporting

ECUC_Can_00496. Switches the reporting of security events to the IdsM:

- true: reporting is enabled.
- false: reporting is disabled.

Tags: atp.Status=draft

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	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.7 Parameter CanVersionInfoApi

ECUC_Can_00106. Switches the Can_GetVersionInfo() API: ON or OFF.

When this option is ON driver supports API for getting Version information for 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-PRE-COMPILE: PRE-COMPILE VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.8 Parameter CanIndex

ECUC_Can_00320. Specifies the InstanceId of this module instance.

If only one instance is present it shall have the Id 0.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false

Property	Value
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	0
max	255
min	0

4.9 Parameter CanMainFunctionBusoffPeriod

ECUC_Can_00355. This parameter describes the period for cyclic call to Can_MainFunction_Busoff.

Unit is seconds.

Property	Value
type	ECUC-FLOAT-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	0.001
max	65.535
min	0.0

4.10 Parameter CanMainFunctionWakeupPeriod

ECUC_Can_00357. This parameter describes the period for cyclic call to Can_MainFunction_Wakeup.

Unit is seconds.

This field is editable if CanConfigSet/CanController/CanWakeupSupport is 'true'.

Property	Value
type	ECUC-FLOAT-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	0.001
max	65.535
min	0.0

4.11 Parameter CanMainFunctionModePeriod

ECUC_Can_00376. This parameter describes the period for cyclic call to Can_MainFunction_Mode.

Unit in seconds.

Property	Value
type	ECUC-FLOAT-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	0.001
max	65.535
min	0.0

4.12 Parameter CanMultiplexedTransmission

ECUC_Can_00095. Specifies if Multiplexed Transmission shall be supported: ON or OFF.

Multiplex transmission means to search for a free MB, that has the same ObjectId with the one transmitted to Can_Write,

if current Hth MB is busy.

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-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	true

4.13 Parameter CanTimeoutMethod

The driver timeout method shall be defined based on the pre-compile configuration parameter named <Module>TimeoutMethod with the following values:

OSIF_COUNTER_DUMMY - timeout based on software implementation in osif

OSIF_COUNTER_SYSTEM - timeout based on system timer - reference found in Base driver

OSIF_COUNTER_CUSTOM - timeout based on custom implementation using Osif APIs

Note: If SystemTimer or CustomTimer are selected make sure the corresponding timer is enabled in OsIf General configuration.

Note: Implementation Specific Parameter.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	OSIF_COUNTER_DUMMY
literals	['OSIF_COUNTER_DUMMY', 'OSIF_COUNTER_SYSTEM', 'OSIF_COUNTER_CUSTOM']

4.14 Parameter CanTimeoutDuration

ECUC_Can_00113. Specifies the maximum time for blocking function until a timeout is detected. Unit is seconds.

This Timeout is used to detect the Hardware Errors/ Production Errors.

When Hardware registers like Controller Register (CTRL) or Module Control Register(MCR) are configured, the Hardware take some time to take effect of these new settings CANuested.

Once timeout has been occured and if hardware could not take effect of the CANu settings, then Error is reported.

So this timeout is used to allow hardware to take effect of the Hardware settings.

For OSIF_COUNTER_DUMMY method, CanTimeoutDuration may not reflect exactly in seconds (but in the terms of loops, 1 us : 1 loop).

Property	Value
type	ECUC-FLOAT-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	1.0
max	65.535
min	1.0E-6

4.15 Parameter CanLPduReceiveCalloutFunction

ECUC_Can_00434:This parameter defines the existence and the name of a callout function that is called after a successful reception of a received CAN Rx L-PDU. If this parameter is omitted no callout shall take place.

Property	Value
type	ECUC-FUNCTION-NAME-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE VARIANT-POST-BUILD: PRE-COMPILE

Property	Value
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	NULL_PTR

4.16 Parameter CanMBCountExtensionSupport

Enables support of more than 255 Can Hardware Objects.

Some platforms have a bigger number of Can controllers and the sum of total MBS for all controllers is bigger than uint8 size (as HTH/HRH is specified in Autosar).

This option should not be enabled for platforms that have a number of MBs smaller than 256 (summing all Can controllers from the platform).

NoteImplementation Specific parameter.

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-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	true

4.17 Parameter CanApiEnableMbAbort

Vendor specific: Can_AbortMb shall be supported if the parameter set to true.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	0
upperMultiplicity	1

Property	Value
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.18 Parameter CanSetBaudrateApi

If the parameter is set to true the Can_SetBaudrate Api shall be supported.

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

4.19 Parameter CanEnableDualClockMode

Enables support for dual clock API. When this parameter is true will generate CAN_DUAL_CLOCK_MODE = STD_ON.

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

Property	Value
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.20 Parameter CanListenOnlyModeApi

Vendor specific: Can_ListenOnlyMode shall be supported if the parameter set to true.

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

4.21 Parameter CanGlobalTimeSupport

Shall enable/disable the Global Time APIs used when hardware timestamping is supported by CAN controller.

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-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.22 Reference CanEcucPartitionRef

ECUC_Can_00491. Maps the CAN driver to zero or multiple ECUC partitions to make the modules API available in this partition. The CAN driver will operate as an independent instance in each of the partitions.

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

4.23 Reference CanOsCounterRef

ECUC_Can_00431. This parameter contains a reference to the counter, which is used by the CAN driver.

Note: This node is unused, the using of OsCounter is done by selecting CanTimeoutMethod to OSIF_COUNTER_SYSTEM.

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

4.24 Reference CanSupportTTCANRef

ECUC_Can_00430: The parameter refers to CanIfSupportTTCAN parameter in the CAN Interface Module configuration. The CanIfSupportTTCAN parameter defines whether TTCAN is supported

Property	Value
type	ECUC-REFERENCE-DEF
origin	AUTOSAR_ECUC
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE VARIANT-POST-BUILD: PRE-COMPILE
requiresSymbolicNameValue	False
destination	/AUTOSAR/EcucDefs/CanIf/CanIfPrivateCfg

4.25 Container CanMainFunctionRWPeriods

ECUC_Can_00437. This container contains the parameter for configuring the period for cyclic call to Can_MainFunction_Read or Can_MainFunction_Write depending on the referring item.

Included subcontainers:

- None

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

4.26 Parameter CanMainFunctionPeriod

ECUC_Can_00484. This parameter describes the period for cyclic call to Can_MainFunction_Read or Can_MainFunction_Write depending on the referring item. Unit is seconds.

Different poll-cycles will be configurable if more than one CanMainFunctionPeriod is configured.

In this case multiple `Can_MainFunction_Read()` or `Can_MainFunction_Write()` will be provided by the CAN Driver module.

Property	Value
type	ECUC-FLOAT-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	0.001
max	65.535
min	0.001

4.27 Container CanConfigSet

ECUC_Can_00343. This is the multiple configuration set container for CAN Driver.

Included subcontainers:

- [CanController](#)
- [CanHardwareObject](#)

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

4.28 Container CanController

ECUC_Can_00354. This container contains the configuration parameters of the CAN controller(s).

Included subcontainers:

- [CanControllerBaudrateConfig](#)

- [CanTTController](#)
- [CanRamBlock](#)
- [CanRxFiFo](#)

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

4.29 Parameter CanHwChannel

Specifies which one of the on-chip FlexCAN interfaces is associated with this controller ID.

NoteImplementation Specific parameter. Not AutoSar Required.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	FLEXCAN_0
literals	['FLEXCAN_0']

4.30 Parameter CanControllerActivation

ECUC_Can_00315. Defines if a CAN controller is used in the configuration.

Deactivation of a particular CAN controller is equivalent to a CAN controller not used in the configuration.

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	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	true

4.31 Parameter CanControllerBaseAddress

ECUC_Can_00382. Specifies the CAN controller base address.

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-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	0
max	4294967295
min	0

4.32 Parameter CanControllerId

ECUC_Can_00316: This parameter provides the controller ID which is unique in a given CAN Driver.

The value for this parameter starts with 0 and continue without any gaps.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	true

Property	Value
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	1
max	255
min	0

4.33 Parameter CanRxProcessing

ECUC_Can_00317. Enables/Disables API Can_MainFunction_Read() for handling PDU reception events in POLLING mode.

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-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	POLLING
literals	['INTERRUPT', 'POLLING', 'MIXED']

4.34 Parameter CanTxProcessing

ECUC_Can_00318. Enables/Disables API Can_MainFunction_Write() for handling PDU transmission events in POLLING mode

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

Property	Value
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	POLLING
literals	['INTERRUPT', 'POLLING', 'MIXED']

4.35 Parameter CanBusoffProcessing

ECUC_Can_00314. Enables/Disables API Can_MainFunction_BusOff() for handling busoff events in POLLING mode.

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-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	POLLING
literals	['INTERRUPT', 'POLLING']

4.36 Parameter CanWakeupFunctionalityAPI

Adds / removes the service Can_CheckWakeup() from the code.

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

Property	Value
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	false

4.37 Parameter CanWakeupProcessing

ECUC_Can_00319. Enables/Disables API Can_MainFunction_Wakeup() for handling wakeup events in POLLING mode.

Note This option is enabled only if global parameter <CanController/CanWakeupSupport> is 'true'.

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-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	POLLING
literals	['INTERRUPT', 'POLLING']

4.38 Parameter CanWakeupSupport

ECUC_Can_00330. CAN driver support for wakeup over CAN Bus.

Every WakeUp process will be ignore if this checkbox is not set to ON.

This parameter enables Internal Wakeup (using controller registers) and External Wakeup (using WKUP module).

This is enabled only if internal Wakeup is supported by the platform.

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-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.39 Parameter CanLoopBackMode

Vendor specific: Enables CAN to operate in Loop Back Mode.

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-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.40 Parameter CanAutoBusOffRecovery

Enable/Disable automatic BusOff recovery (CTRL[BOFF_REC] bit).

0(Checked) = Automatic recovering from Bus Off state occurs according to the CAN Specification 2.0B.

1(Unchecked) = Automatic recovering from Bus Off is disabled and the module remains in Bus Off state until the bit is negated(zero) by the user.

NoteImplementation specific Parameter. Not AutoSar Required.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF

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

4.41 Parameter CanTrippleSamplingEnable

Vendor specific: Defines the sampling mode of CAN bits at the Rx input.

True - Three samples are used to determine the value of the received bit.

False - Just one sample is used to determine the bit value.

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-PRE-COMPILE: PRE-COMPILE VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.42 Parameter CanControllerPrExcEn

Vendor specific: The protocol exception feature.(See Protocol exception event in the CAN Protocol standard (ISO 11898-1) for details)

True - Enable Feature.

False - Disable Feature.

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

4.43 Parameter CanControllerEdgeFilter

Vendor specific: The Edge Filter feature. (See Bus Integration state in the CAN Protocol standard (ISO 11898-1) for details)

True - Enable Feature.

False - Disable Feature.

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

4.44 Parameter CanControllerFdISO

Vendor specific: Specifies Can FD protocol according to ISO or non-ISO (FlexCAN is able to transmit FD frame format according to CAN Protocol standard (ISO11898-1))

True - Controller operates using the ISO CAN FD protocol (ISO 11898-1).

False - Controller operates using the non-ISO CAN FD protocol.

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-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
default Value	true

4.45 Parameter CanClockFromBus

Switches the source clock for the module to the system bus (rather than crystal).

1 = The CAN engine clock source is the bus clock.(from MCU)

0 = The CAN engine clock source is the oscillator clock.

NoteImplementation specific Parameter. Not AutoSar Required.

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-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
default Value	true

4.46 Reference CanControllerDefaultBaudrate

ECUC_Can_00435. Reference to baudrate configuration container configured for the Can Controller.

Property	Value
type	ECUC-REFERENCE-DEF
origin	AUTOSAR_ECUC
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
requiresSymbolicNameValue	False
destination	/AUTOSAR/EcuDefs/Can/CanConfigSet/CanController/CanController↵ BaudrateConfig

4.47 Reference CanControllerEcucPartitionRef

ECUC_Can_00492. Maps the CAN controller to zero or one ECUC partitions. The ECUC partition referenced is a subset of the ECUC partitions where the CAN driver is mapped to.

Property	Value
type	ECUC-REFERENCE-DEF
origin	AUTOSAR_ECUC
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	true
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
requiresSymbolicNameValue	False
destination	/AUTOSAR/EcuDefs/EcuC/EcuPartitionCollection/EcuPartition

4.48 Reference CanCpuClockRef

ECUC_Can_00313. Reference to the CPU clock configuration, which is set in the MCU driver configuration. MCU plugin need to be added and then give the reference to it.

Property	Value
type	ECUC-REFERENCE-DEF
origin	AUTOSAR_ECUC
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
requiresSymbolicNameValue	False
destination	/AUTOSAR/EcuDefs/Mcu/McuModuleConfiguration/McuClockSetting↔ Config/McuClockReferencePoint

4.49 Reference CanCpuClockRefAlternate

Vendor specific: Alternative reference to the CPU clock configuration, which is set in the MCU driver configuration.

MCU plugin need to be added and then give the reference to it.

Note: CanEnableDualClockMode must be true to use this node.

Property	Value
type	ECUC-REFERENCE-DEF
origin	NXP
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
requiresSymbolicNameValue	False
destination	/AUTOSAR/EcuDefs/Mcu/McuModuleConfiguration/McuClockSetting↔ Config/McuClockReferencePoint

4.50 Reference CanWakeupSourceRef

ECUC_Can_00359. This parameter contains a reference to the Wakeup Source for this controller as defined in the ECU State Manager.

Type: reference to EcuM_WakeupSourceType

EcuM plugin need to be added and then give the reference to it.

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

4.51 Container CanControllerBaudrateConfig

This container contains bit timing related configuration parameters of the CAN controller(s)

Included subcontainers:

- [CanControllerFdBaudrateConfig](#)

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

4.52 Parameter CanAdvancedSetting

If TRUE initiates the derivation of the CAN bit timing values from the CanControllerBaudRate parameter.

When this option is True the CanControllerPropSeg, CanControllerSeg1, CanControllerSeg2, CanController-SyncJumpWidth are disabled because are background calculated.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF

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

4.53 Parameter CanBusLength

Specifies the CAN Bus length in meters.

This parameter is used for PROPSEG parameter calculation when "CanAdvancedSetting" control is set to true.

NoteImplementation specific Parameter.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE VARIANT-POST-BUILD: POST-BUILD
defaultValue	40
max	5000
min	1

4.54 Parameter CanPropDelayTranceiver

Propagation delay of Tranceiver used in nanoseconds.

NoteImplementation specific Parameter.

The calculation for the CAN bit timing is implemented in the code template.

The Formulas used in the code template for calculation are as follows.

Physical delay of bus = Bus length * Bus propagation delay.

$t_{PROP_SEG} = 2(\text{Physical delay of bus} + \text{CanPropDelayTranceiver})$.

$PROP_SEG = \text{ROUND_UP}(t_{PROP_SEG} / \text{Bus propagation delay})$.

Based on these calculations implemented in the Code template the consistency check is maintained for CanPropDelayTranceiver parameter.

The PROP_SEG parameter need to be a integral value and not fractional value.

Property	Value
type	ECUC-FLOAT-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE VARIANT-POST-BUILD: POST-BUILD
defaultValue	150.0
max	5000.0
min	0.0

4.55 Parameter CanTxArbitrationStartDelay

This 5-bit field indicates how many CAN bits the Tx arbitration process start point can be delayed from the first bit of CRC field on CAN bus.

See Reference Manual to have a calculation method for the optimal T ASD value.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE

Property	Value
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	31
min	0

4.56 Parameter CanControllerPrescaler

Specifies the prescaler for the controller .

The calculation of the resulting CanControllerTimeQuanta value depending on module clocking and prescaler shall be done offline.

$\text{Prescaler} = \text{FreqCanClk} / \text{FreqTq}$; $\text{FreqTq} = 1 / \text{CanControllerTimeQuanta}$.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	10
max	1024
min	1

4.57 Parameter CanControllerPrescalerAlternate

Vendor specific: Specifies the alternate prescaler for the controller .

The calculation of the resulting CanControllerTimeQuanta_Alternate value depending on module clocking and prescaler shall be done offline.

$\text{Prescaler} = \text{FreqCanClk} / \text{FreqTq}$; $\text{FreqTq} = 1 / \text{CanControllerTimeQuanta}$.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP

Property	Value
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE VARIANT-POST-BUILD: POST-BUILD
defaultValue	10
max	1024
min	1

4.58 Parameter CanControllerBaudRateConfigID

Uniquely identifies a specific baud rate configuration. This ID is used by

SetBaudrate API

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-PRE-COMPILE: PRE-COMPILE VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	65535
min	0

4.59 Parameter CanControllerBaudRate

ECUC_Can_00005. Specifies the buadrate of the controller in kbps.

CAN maximum speed is 1Mbps.

Property	Value
type	ECUC-FLOAT-PARAM-DEF

Property	Value
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE VARIANT-POST-BUILD: POST-BUILD
defaultValue	20.0
max	2000.0
min	0.0

4.60 Parameter CanControllerSyncSeg

The Synchronization Segment or SYNC_SEG time interval is used to synchronize all the nodes across the network.

The SYNC_SEG time interval has a fixed period of one Time Quantum (TQ).

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-PRE-COMPILE: PRE-COMPILE VARIANT-POST-BUILD: POST-BUILD
defaultValue	1
max	1
min	1

4.61 Parameter CanControllerPropSeg

ECUC_Can_00073. It is used to compensate the physical delay within the CAN network.

when disable extended CAN bit timing:

The CanControllerPropSeg valid values are 1-8 Tq.

when enable extended CAN bit timing:

The CanControllerPropSeg valid values are 1-64 Tq.

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-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	5
max	255
min	0

4.62 Parameter CanControllerSeg1

ECUC_Can_00074. Specifies the Phase Segment 1 in time quantas.

when disable extended CAN bit timing:

The CanControllerSeg1 valid values are 1-8 Tq.

when enable extended CAN bit timing:

The CanControllerSeg1 valid values are 1-32 Tq.

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-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	5
max	255
min	0

4.63 Parameter CanControllerSeg2

Specifies the Phase Segment 1 in time quantas.

when disable extended CAN bit timing:

The CanControllerSeg2 valid values are 2-8 Tq.

when enable extended CAN bit timing:

The CanControllerSeg2 valid values are 2-32 Tq.

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-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	6
max	255
min	0

4.64 Parameter CanControllerSyncJumpWidth

when disable extended CAN bit timing:

The CanControllerSyncJumpWidth valid values are 1-4 Tq.

when enable extended CAN bit timing:

The CanControllerSyncJumpWidth valid values are 1-32 Tq.

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

Property	Value
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	1
max	255
min	0

4.65 Container CanControllerFdBaudrateConfig

This optional container contains bit timing related configuration parameters of the CAN controller(s) for payload and CRC of a CAN FD frame. If this container exists the controller supports CAN FD frames.

Included subcontainers:

- None

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

4.66 Parameter CanControllerFdBaudRate

ECUC_Can_00481.Specifies the data segment baud rate of the controller in kbps.

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

Property	Value
default Value	250.0
max	8000.0
min	0.0

4.67 Parameter CanControllerFdSyncSeg

The Synchronization Segment or SYNC_SEG time interval is used to synchronize all the nodes across the network.

The SYNC_SEG time interval has a fixed period of one Time Quantum (TQ).

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-PRE-COMPILE: PRE-COMPILE VARIANT-POST-BUILD: POST-BUILD
default Value	1
max	1
min	1

4.68 Parameter CanControllerPropSeg

ECUC_Can_00476.Specifies propagation delay in time quantas.

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-PRE-COMPILE: PRE-COMPILE VARIANT-POST-BUILD: POST-BUILD

Property	Value
defaultValue	1
max	255
min	0

4.69 Parameter CanControllerSeg1

ECUC_Can_00477.Specifies phase segment 1 in time quantas.

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-PRE-COMPILE: PRE-COMPILE VARIANT-POST-BUILD: POST-BUILD
defaultValue	1
max	255
min	0

4.70 Parameter CanControllerSeg2

ECUC_Can_00478.Specifies phase segment 2 in time quantas.

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-PRE-COMPILE: PRE-COMPILE VARIANT-POST-BUILD: POST-BUILD
defaultValue	2
max	255
min	0

4.71 Parameter CanControllerSyncJumpWidth

ECUC_Can_00479.Specifies the synchronization jump width for the controller in time quantas.

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-PRE-COMPILE: PRE-COMPILE VARIANT-POST-BUILD: POST-BUILD
defaultValue	1
max	255
min	0

4.72 Parameter CanControllerSpOffset

ECUC_Can_00494 .Specifies the Transmitter Delay Compensation Offset in minimum time quanta

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

4.73 Parameter CanControllerFdPrescaler

Fd Prescaler Option overwrite the Can Controller Prescaler

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	true
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	1
max	1024
min	1

4.74 Parameter CanControllerPrescalerAlternateFd

Vendor specific: Specifies the alternate prescaler for the controller .

The calculation of the resulting CanControllerTimeQuanta_Alternate value depending on module clocking and prescaler shall be done offline.

$\text{Prescaler} = \text{FreqCanClk} / \text{FreqTq}$; $\text{FreqTq} = 1 / \text{CanControllerTimeQuanta}$.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	10
max	1024
min	1

4.75 Parameter CanControllerTxBitRateSwitch

Specifies if the bit rate switching shall be used for transmissions.

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-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	true

4.76 Container CanTTController

This container is only included and valid if TTCAN SWS is used and TTCAN is enabled.

Included subcontainers:

- None

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

4.77 Parameter CanTTControllerApplWatchdogLimit

Defines the maximum time period (unit is 256 times NTU) after which the application has to serve the watchdog.

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

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

4.78 Parameter CanTTControllerCycleCountMax

Defines the value for cycle_count_max.

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-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	0
max	63
min	0

4.79 Parameter CanTTControllerExpectedTxTrigger

Number of expected_tx_trigger.

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

Property	Value
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	0
max	255
min	0

4.80 Parameter CanTTControllerExternalClockSynchronisation

Enables/disables the external clock synchronization.

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-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	false

4.81 Parameter CanTTControllerGlobalTimeFiltering

Enables/disables the global time filtering.

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

Property	Value
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	false

4.82 Parameter CanTTControllerInitialRefOffset

Defines the initial value for ref trigger offset.

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-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	0
max	127
min	0

4.83 Parameter CanTTControllerInterruptEnable

Enables/disables the respective interrupts.

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-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE

Property	Value
defaultValue	0
max	1023
min	0

4.84 Parameter CanTTControllerLevel2

Defines whether Level 2 or Level 1 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-POST-BUILD: POST-BUILD VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	false

4.85 Parameter CanTTControllerNTUConfig

Defines the config value for NTU (network time unit).

Property	Value
type	ECUC-FLOAT-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: POST-BUILD VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	0.0
max	100.0
min	0.0

4.86 Parameter CanTTControllerOperationMode

Defines the operation mode.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	CAN_TT_EVENT_SYNC_TIME_TRIGGERED
literals	['CAN_TT_EVENT_SYNC_TIME_TRIGGERED', 'CAN_TT_EVENT_↔ TRIGGERED', 'CAN_TT_TIME_TRIGGERED']

4.87 Parameter CanTTControllerSyncDeviation

Defines the maximum synchronization deviation:

Property	Value
type	ECUC-FLOAT-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	0.0
max	100.0
min	0.0

4.88 Parameter CanTTControllerTURRestore

Enables/disables the TUR restore.

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-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
default Value	false

4.89 Parameter CanTTControllerTimeMaster

Defines whether the controller acts as a potential time master.

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-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
default Value	false

4.90 Parameter CanTTControllerTimeMasterPriority

Defines the time master priority.

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

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

4.91 Parameter CanTtControllerTxEnableWindowLength

Length of the tx enable window given in CAN bit times.

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-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	1
max	16
min	1

4.92 Parameter CanTtControllerWatchTriggerGapTimeMark

watch trigger time mark after a gap

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-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	0
max	65535
min	0

4.93 Parameter CanTTControllerWatchTriggerTimeMark

watch trigger time mark

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-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	0
max	65535
min	0

4.94 Parameter CanTTIRQProcessing

Enables / disables API Can_MainFunction_BusOff() for handling busoff events in POLLING mode.

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

Property	Value
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	POLLING
literals	['INTERRUPT', 'POLLING']

4.95 Reference CanTTControllerEcucPartitionRef

ECUC_Can_00493. Maps the Time triggered CAN controller to zero or one ECUC

partitions. The ECUC partition referenced is a subset of the ECUC

partitions where the CAN driver is mapped to.

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

4.96 Container CanRamBlock

Vendor specific: Specify Data size of ram block.

Included choices:

- CanRamBlockUnified
- CanRamBlockSpecified

Property	Value
type	ECUC-CHOICE-CONTAINER-DEF

Property	Value
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE

4.97 Container CanRxFiFo

Vendor specific: Specify the FIFO used.

Legacy FIFO can't be used if FD is activated! Please deactivate CanControllerFdBaudrateConfig optional field if Legacy FIFO is needed.

Included choices:

- CanLegacyFiFo

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

4.98 Container CanHardwareObject

SWS 324. This container contains the configuration (parameters) of CAN Hardware Objects.

This configuration element is used as information for the CAN Interface only.

The relevant CAN driver configuration is done with the filter mask and identifier.

Included subcontainers:

- [CanHwFilter](#)
- [CanTTHardwareObjectTrigger](#)

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

4.99 Parameter CanFdPaddingValue

MBCS[PRI0]: This value it is the padding value when FD it is used.

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

4.100 Parameter CanObjectPayloadLength

ECUC_Can_00495. Specifies the maximum L-PDU payload length in bytes the hardware object can store.

If the parameter is not provided, Can driver configuration generators have to assume the maximum length of the underlying CAN derivate, e.g. 8 bytes for CAN, 64 bytes for CAN-FD.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	0
upperMultiplicity	1

Property	Value
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	CAN_OBJECT_PL_8
literals	['CAN_OBJECT_PL_12', 'CAN_OBJECT_PL_16', 'CAN_OBJECT_PL_20', 'CAN_OBJECT_PL_24', 'CAN_OBJECT_PL_32', 'CAN_OBJECT_PL_48', 'CAN_OBJECT_PL_64', 'CAN_OBJECT_PL_8']

4.101 Parameter CanHandleType

ECUC_Can_00323. Specifies the type (Full-CAN or Basic-CAN) of a hardware object.

NoteAll controllers which the Fifo is enabled shall define at least 1 RECEIVE hardware object.

First RECEIVE hardware object defined for a controller which have the Fifo enabled is configured by CONVENTION to receive data from Fifo.

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

4.102 Parameter CanIdType

ECUC_Can_00065. Specifies whether the IdValue is of type

- standard identifier (ID - 11 bits length)
- extended identifier (ID - 29 bits length)



Tresos Configuration Plug-in

- mixed mode (standard or extended)

NoteMBs configed as MIXED standard and RECEIVE type will be treated as EXTENDED.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
default Value	STANDARD
literals	['EXTENDED', 'MIXED', 'STANDARD']

4.103 Parameter CanObjectId

ECUC_Can_00326. Holds the handle ID of HRH or HTH.

The value of this parameter is unique in a given CAN Driver, and it should start with 0 and continue without any gaps.

The HRH and HTH Ids are defined under two different name-spaces.

Example: HRH0-0, HRH1-1, HTH0-2, HTH1-3

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	true
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
default Value	0
max	65535
min	0

4.104 Parameter CanObjectType

ECUC_Can_00327. Specifies if the HardwareObject is used as Transmit or as Receive object.

NoteMBs configured as MIXED standard and RECEIVE type will be treated as EXTENDED.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
default Value	RECEIVE
literals	['RECEIVE', 'TRANSMIT']

4.105 Parameter CanHardwareObjectUsesPolling

Enables polling of this hardware object. This node shall exist if CanRxProcessing/CanTxProcessing is set to MIXED.

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

4.106 Parameter CanTriggerTransmitEnable

This parameter defines if or if not Can supports the trigger-transmit API for this handle.

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

4.107 Parameter CanHwObjectUsesBlock

Vendor specific: Selects the Block which Hw Object take into.

This field is meaningless for first HRH of controller enabling Enhance FIFO (Enhance FIFO object).

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	False
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	CAN_RAM_BLOCK_0
literals	['CAN_RAM_BLOCK_0', 'CAN_RAM_BLOCK_1']

4.108 Parameter CanHwObjectCount

Number of hardware objects used to implement one HOH. In case of a HRH this parameter defines the number of elements in the hardware FIFO or the number of shadow buffers, in case of a HTH it defines the number of hardware objects used for multiplexed transmission or for a hardware FIFO used by a FullCAN HTH

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-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
default Value	1
max	65535
min	1

4.109 Reference CanControllerRef

ECUC_Can_00322. Reference to CAN Controller to which the HOH is associated to.

Property	Value
type	ECUC-REFERENCE-DEF
origin	AUTOSAR_ECUC
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
requiresSymbolicNameValue	False
destination	/AUTOSAR/EcucDefs/Can/CanConfigSet/CanController

4.110 Reference CanMainFunctionRWPeriodRef

ECUC_Can_00438.Reference to CAN Controller to which the HOH is associated to.

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

Property	Value
upperMultiplicity	1
postBuildVariantMultiplicity	true
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
requiresSymbolicNameValue	False
destination	/AUTOSAR/EcucDefs/Can/CanGeneral/CanMainFunctionRWPeriods

4.111 Container CanHwFilter

ECUC_Can_00468 : This container is only valid for HRHs and contains the configuration (parameters) of one hardware filter.

If the HRH is used for Legacy FIFO, CanHwFilterCode must be considered as below:

Can ID Acceptance Mode :

FORMAT_A :

- STANDARD : All bits (in the total of 11 bits) are used for frame identification
- EXTENDED : All bits (in the total of 29 bits) are used for frame identification

FORMAT_B :

- STANDARD : All bits (in the total of 11 bits) are used for frame identification
- EXTENDED : Only 14 most significant bits (in the total of 29 bits) used for frame identification

FORMAT_C :

- STANDARD : Only 8 most significant bits (in the total of 11 bits) used for frame identification
- EXTENDED : Only 8 most significant bits (in the total of 29 bits) used for frame identification

User need to provide the entire id.

For example: for FORMAT_C, Frame type is STANDARD, user must provide all 11 bits instead of 8 most significant bits only.

Included subcontainers:

- None

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

4.112 Parameter CanHwFilterCode

ECUC_Can_00325. Specifies (together with the filter mask)- the identifiers range that passes the hardware filter for of RX objects.

Parameter ranges from 0 to 0x7FF (11 bits) for Standard IDs and 0 to 0xFFFFFFFF (29 bits) for Extended IDs.

User can assign any code to this parameter, but must to respect the above rule related to Standard/Extended 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-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	4294967295
min	0

4.113 Parameter CanHwFilterMask

ECUC_Can_00469 : Specifies (together with the filter mask) the identifiers range that passes the hardware filter.

EN:

This value is used as acceptance masks for ID filtering in RX MBs and the FIFO.

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-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	4294967295
min	0

4.114 Container CanTTHardwareObjectTrigger

This container is only included and valid if TTCAN SWS is used and TTCAN is enabled.

Included subcontainers:

- None

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

4.115 Parameter CanTTHardwareObjectBaseCycle

Defines the cycle_offset.

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

Property	Value
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	0
max	63
min	0

4.116 Parameter CanTTHardwareObjectCycleRepetition

Defines the repeat_factor.

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-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	1
max	64
min	1

4.117 Parameter CanTTHardwareObjectTimeMark

Defines the point in time, when the trigger will be activated.

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-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	0
max	65535
min	0

4.118 Parameter CanTTHardwareObjectTriggerId

Sequential number which allows separation of different TTCAN triggers configured for one and the same hardware object.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	true
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	0
max	63
min	0

4.119 Parameter CanTTHardwareObjectTriggerType

Defines the type of the trigger associated with the hardware object. This parameter depends on plain CAN parameter CAN_OBJECT_TYPE.

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

Property	Value
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	CAN_TT_RX_TRIGGER
literals	['CAN_TT_RX_TRIGGER', 'CAN_TT_TX_REF_TRIGGER', 'CAN_TT_TX_REF_TRIGGER_GAP', 'CAN_TT_TX_TRIGGER_EXCLUSIVE', 'CAN_TT_TX_TRIGGER_MERGED', 'CAN_TT_TX_TRIGGER_SINGLE']

4.120 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

4.121 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

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

4.122 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-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
defaultValue	7
max	7
min	7

4.123 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
valueConfigClasses	VARIANT-POST-BUILD: PUBLISHED-INFORMATION

Property	Value
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
defaultValue	0
max	0
min	0

4.124 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-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
defaultValue	80
max	80
min	80

4.125 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
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

Property	Value
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
defaultValue	2
max	2
min	2

4.126 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-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
defaultValue	0
max	0
min	0

4.127 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
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1

Property	Value
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
defaultValue	0
max	0
min	0

4.128 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-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
defaultValue	FLEXCAN

4.129 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-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:

CAN_43_FLEXCAN_DRIVER	105
FlexCAN	121
FlexCAN_driver	181
Controller Area Network with Flexible Data Rate (FlexCAN)	179

Chapter 6

Module Documentation

6.1 CAN_43_FLEXCAN_DRIVER

6.1.1 Detailed Description

Data Structures

- struct [Can_43_FLEXCAN_ConfigType](#)
Can Configuration. [More...](#)
- struct [Can_43_FLEXCAN_HwFilterType](#)
Can Hardware Filter. [More...](#)
- struct [Can_43_FLEXCAN_HwObjectConfigType](#)
Can Hardware Object. [More...](#)
- struct [Can_43_FLEXCAN_TimeSegmentType](#)
Can Bit Rate. [More...](#)
- struct [Can_43_FLEXCAN_BaudrateConfigType](#)
Can Baudrate. [More...](#)
- struct [Can_43_FLEXCAN_ControllerConfigType](#)
Can Controller. [More...](#)

Macros

- [#define CAN_43_FLEXCAN_E_DATALOST](#)
Runtime Error ID for "Received CAN message is lost".
- [#define CAN_43_FLEXCAN_SID_MAIN_FUNCTION_READ](#)
Service ID of Can_43_FLEXCAN_MainFunction_Read.

Enum Reference

- enum [Can_43_FLEXCAN_HwObjectHandleType](#)
Can Hardware Object Handle.
- enum [Can_43_FLEXCAN_IdMessageType](#)
Can Id Message.
- enum [Can_43_FLEXCAN_MbType](#)
Message Buffer Type: TX, RX, RX FIFO.
- enum [Can_43_FLEXCAN_LegacyFIFOAcceptanceModeType](#)
Legacy FIFO ID Acceptance Mode.

Function Reference

- void [Can_43_FLEXCAN_Init](#) (const [Can_43_FLEXCAN_ConfigType](#) *Config)
Initialize the CAN driver. SID is 0x00.
- void [Can_43_FLEXCAN_DeInit](#) (void)
De-initialize the CAN driver. SID is 0x10.
- Std_ReturnType [Can_43_FLEXCAN_SetControllerMode](#) (uint8 Controller, Can_ControllerStateType Transition)
Put the controller into a required state. SID is 0x03.
- void [Can_43_FLEXCAN_DisableControllerInterrupts](#) (uint8 Controller)
Disable INTs. SID is 0x04.
- void [Can_43_FLEXCAN_EnableControllerInterrupts](#) (uint8 Controller)
Enable INTs. SID is 0x05.
- Std_ReturnType [Can_43_FLEXCAN_GetControllerErrorState](#) (uint8 ControllerId, Can_ErrorStateType *ErrorStatePtr)
Obtains the error state of the CAN controller.. SID is 0x11.
- Std_ReturnType [Can_43_FLEXCAN_GetControllerMode](#) (uint8 Controller, Can_ControllerStateType *ControllerModePtr)
Reports about the current status of the requested CAN controller. SID is 0x12.
- Std_ReturnType [Can_43_FLEXCAN_GetControllerRxErrorCounter](#) (uint8 ControllerId, uint8 *RxErrorCounterPtr)
Return the Rx error counter for a CAN controller.
- Std_ReturnType [Can_43_FLEXCAN_GetControllerTxErrorCounter](#) (uint8 ControllerId, uint8 *TxErrorCounterPtr)
Return the Tx error counter for a CAN controller.
- Std_ReturnType [Can_43_FLEXCAN_Write](#) (Can_HwHandleType Hth, const Can_PduType *PduInfo)
Transmit information on CAN bus. SID is 0x06.

6.1.2 Data Structure Documentation

6.1.2.1 struct Can_43_FLEXCAN_ConfigType

Can Configuration.

Definition at line 249 of file Can_43_FLEXCAN.h.

Data Fields

Type	Name	Description
const uint32	Can_u32CoreID	Configuration Core ID.
const Can_HwHandleType	Can_uHthFirstIndex	The first Hth after Hrh consecutive.
const uint8 *	Can_pCtrlOffsetToCtrlIDMap	Mapping Controller ID to Controller hardware offset.
const uint8 *	Can_pHwObjIDToCtrlIDMap	Mapping Controller ID to Hardware Object ID.
const Can_43_FLEXCAN_HwObjectConfigType *	Can_pHwObjectConfig	Pointer to Can Hardware Object Config.
const Can_43_FLEXCAN_ControllerConfigType *const *	Can_ppController	Pointer to Can Controller Config.

6.1.2.2 struct Can_43_FLEXCAN_HwFilterType

Can Hardware Filter.

Definition at line 319 of file Can_Flexcan_Types.h.

Data Fields

Type	Name	Description
const uint32	Can_u32HwFilterCode	Specifies (together with the filter mask) the identifiers range that passes the hardware filter.
const uint32	Can_u32HwFilterMask	Describes a mask for hardware-based filtering of CAN identifiers.

6.1.2.3 struct Can_43_FLEXCAN_HwObjectConfigType

Can Hardware Object.

Definition at line 326 of file Can_Flexcan_Types.h.

Data Fields

Type	Name	Description
const Can_HwHandleType	Can_HwObjectID	Can Hardware Object ID.
const Can_43_FLEXCAN_HwObjectHandleType	Can_HohType	Specifies Hardware Object is used as Transmit or as Receive Object.
const Can_43_FLEXCAN_IdMessageType	Can_IdMessage	Specifies the type of Message ID: STANDARD, EXTENDED, MIXED.
const boolean	Can_bHwObjectUsesPolling	Specifies the processing of HOH is Polling or Interrupt.

Data Fields

Type	Name	Description
const boolean	Can_bTriggerTransmit	Specifies the Hw object is enable/disable Trigger Transmit.
const uint8	Can_u8ObjectCount	Number of Hardware Objects used to implement one HOH.
const uint8	Can_MainFuncPeriodIndex	Can MainFunction RW period reference.
const uint8	Can_u8PayloadLength	Specifies the Max data length of Hw Object.
const uint8	Can_u8PaddingValue	Specifies the value which is used to pad unspecified data.
const uint8	Can_u8HwFilterCount	The number of Can Hw Filter Config.
const Can_43_FLEXCAN_HwFilterType *	Can_pHwFilterConfig	Pointer to Hw Filter Config.
const Can_43_FLEXCAN_MbType	Can_eReceiveType	Specifies the Message Buffer is TX, RX or RX FIFO.
const uint8	Can_u8HwBufferIndex	Buffer Index in Message buffer ram.
const uint32 *	Can_pHwBufferAddr	Pointer to Hw Buffer Address.

6.1.2.4 struct Can_43_FLEXCAN_TimeSegmentType

Can Bit Rate.

Definition at line 360 of file Can_Flexcan_Types.h.

Data Fields

Type	Name	Description
const uint8	Can_u8PropSeg	Propagation Segment.
const uint8	Can_u8PhaseSeg1	Phase Segment 1.
const uint8	Can_u8PhaseSeg2	Phase Segment 2.
const uint16	Can_u16Prescaler	Prescaler Divider.
const uint8	Can_u8ResyncJumpWidth	Synchronization Jump Width.

6.1.2.5 struct Can_43_FLEXCAN_BaudrateConfigType

Can Baudrate.

Definition at line 379 of file Can_Flexcan_Types.h.

Data Fields

Type	Name	Description
const boolean	Can_bBitRateSwitch	Tx Bit Rate Switch.
const boolean	Can_bFDFrame	Can FD support.
const Can_43_FLEXCAN_TimeSegmentType	Can_NominalBitRate	Nominal Bit Rate.
const Can_43_FLEXCAN_TimeSegmentType	Can_DataBitRate	Data Bit Rate (using when support FD and Bit Rate Switch is set)
const uint8	Can_u8TxArbitrationStartDelay	Specifies the Transmission Arbitration start delay.
const boolean	Can_bTrcvDelayEnable	Transmitter Delay Compensation Enable.
const uint8	Can_u8TrcvDelayCompOffset	Specifies the Transmitter Delay Compensation Offset.

6.1.2.6 struct Can_43_FLEXCAN_ControllerConfigType

Can Controller.

Definition at line 398 of file Can_Flexcan_Types.h.

Data Fields

Type	Name	Description
const uint8	Can_u8AbstControllerID	Abstracted CanIf Controller ID.
const uint8	Can_u8ControllerID	Controller ID.
const uint8	Can_u8ControllerOffset	Controller Offset.
const uint32	Can_u32BaseAddress	Controller Base Address.
const boolean	Can_bActivation	Define Controller is used in Config.
const boolean	Can_bBusOffUsesPolling	Bus Off uses Polling.
const uint32	Can_u32LegacyGlobalMask	Specifies the Global mask of Legacy FIFO.
const Can_43_FLEXCAN_LegacyFIFOAcceptanceModeType	Can_eLegacyAcceptanceMode	ID Acceptance Mode.
const Can_43_FLEXCAN_NotifyType	Can_pLegacyFiFoWarnNotif	Legacy FIFO Warning Notification.
const Can_43_FLEXCAN_NotifyType	Can_pLegacyFiFoOvfNotif	Legacy FIFO Overflow Notification.
const uint16	Can_u16DefaultBaudrateID	Default Baudrate ID.
const uint16	Can_u16BaudrateConfigCount	Number of Baurate Configured.
const Can_43_FLEXCAN_BaudrateConfigType	Can_pBaudrateConfig	Pointer to Baudrate Config.
* const Can_43_FLEXCAN_Ipw_HwChannelConfigType	HwChannelIpConfig	Pointer to Controller config.
*		

Module Documentation

Data Fields

Type	Name	Description
const uint8	Can_u8HwObjectRefCount	The number of Hw Objects referred to Controller.
const Can_43_FLEXCAN_HwObjectConfigType *const *	Can_ppHwObject	Pointer point to Pointer to Hw Object that refer to Controller.

6.1.3 Macro Definition Documentation

6.1.3.1 CAN_43_FLEXCAN_E_DATALOST

```
#define CAN_43_FLEXCAN_E_DATALOST
```

Runtime Error ID for "Received CAN message is lost".

Definition at line 179 of file Can_Flexcan_Types.h.

6.1.3.2 CAN_43_FLEXCAN_SID_MAIN_FUNCTION_READ

```
#define CAN_43_FLEXCAN_SID_MAIN_FUNCTION_READ
```

Service ID of Can_43_FLEXCAN_MainFunction_Read.

Definition at line 243 of file Can_Flexcan_Types.h.

6.1.4 Enum Reference

6.1.4.1 Can_43_FLEXCAN_HwObjectHandleType

```
enum Can_43_FLEXCAN_HwObjectHandleType
```

Can Hardware Object Handle.

Enumerator

CAN_RECEIVE	Specifies the HardwareObject is used as Receive.
CAN_TRANSMIT	Specifies the HardwareObject is used as Transmit.

Definition at line 273 of file Can_Flexcan_Types.h.

6.1.4.2 Can_43_FLEXCAN_IdMessageType

```
enum Can_43_FLEXCAN_IdMessageType
```

Can Id Message.

Enumerator

CAN_STANDARD	All the CANIDs are of type standard only (11bit).
CAN_EXTENDED	All the CANIDs are of type extended only (29 bit)
CAN_MIXED	All the CANIDs are of type extended only (29 bit)

Definition at line 280 of file Can_Flexcan_Types.h.

6.1.4.3 Can_43_FLEXCAN_MbType

```
enum Can_43_FLEXCAN_MbType
```

Message Buffer Type: TX, RX, RX FIFO.

Enumerator

CAN_RX_NORMAL	Specifies the HardwareObject is used as Normal Receive Object.
CAN_RX_LEGACY_FIFO	Specifies the HardwareObject is used as Legacy FIFO Receive Object.
CAN_RX_ENHANCED_FIFO	Specifies the HardwareObject is used as Enhanced FIFO Receive Object.
CAN_TX_NORMAL	Specifies the HardwareObject is used as Normal Transmit Object.

Definition at line 288 of file Can_Flexcan_Types.h.

6.1.4.4 Can_43_FLEXCAN_LegacyFIFOAcceptanceModeType

```
enum Can_43_FLEXCAN_LegacyFIFOAcceptanceModeType
```

Legacy FIFO ID Acceptance Mode.

Enumerator

CAN_LEGACY_FIFO_FORMAT_A	One full ID (standard and extended) per ID filter table element.
CAN_LEGACY_FIFO_FORMAT_B	Two full standard IDs or two partial 14-bit (standard and extended) IDs per ID filter table element.
CAN_LEGACY_FIFO_FORMAT_C	Four partial 8-bit standard IDs per ID filter table element.

Definition at line 297 of file Can_Flexcan_Types.h.

6.1.5 Function Reference

6.1.5.1 Can_43_FLEXCAN_Init()

```
void Can_43_FLEXCAN_Init (
    const Can_43_FLEXCAN_ConfigType * Config )
```

Initialize the CAN driver. SID is 0x00.

Initialize all the controllers. The CAN module shall be initialized by Can_Init(<&Can_Configuration>) service call during the start-up. This routine is called by:

- CanIf or an upper layer according to Autosar requirements.

Parameters

in	Config	Pointer to driver configuration.
----	--------	----------------------------------

Returns

void

Precondition

Can_Init shall be called at most once during runtime.

Postcondition

Can_Init shall initialize all the controllers and set the driver in READY state.

6.1.5.2 Can_43_FLEXCAN_DeInit()

```
void Can_43_FLEXCAN_DeInit (
    void )
```

De-initialize the CAN driver. SID is 0x10.

De-initialize all the controllers. The CAN module shall be de-initialized by Can_DeInit() service call during the start-up. This routine is called by:

- CanIf or an upper layer according to Autosar requirements.

Parameters

in	None	
----	------	--

Returns

void

Precondition

Before controller de-initialization, the driver must be initialized and the controllers must be not in Start state.

Postcondition

Can_DeInit shall de-initialize all the controllers and set the driver in UNINIT state.

6.1.5.3 Can_43_FLEXCAN_SetControllerMode()

```
Std_ReturnType Can_43_FLEXCAN_SetControllerMode (
    uint8 Controller,
    Can_ControllerStateType Transition )
```

Put the controller into a required state. SID is 0x03.

Switch the controller from one state to another. This routine is called by:

- CanIf or an upper layer according to Autosar requirements.

Module Documentation

Parameters

in	<i>Controller</i>	- Can controller for which the status shall be changed - based on configuration order list (CanControllerId).
in	<i>Transition</i>	- Possible transitions (CAN_CS_STOPPED , CAN_CS_STARTED , CAN_CS_SLEEP)

Returns

Std_ReturnType Result of the transition.

Return values

<i>E_OK</i>	request accepted.
<i>E_NOT_OK</i>	request not accepted, a development error occurred.

Precondition

Before changing the controller state the driver must be initialized.

Postcondition

After the transition to the new state the interrupts required for that state must be enebaled.

6.1.5.4 Can_43_FLEXCAN_DisableControllerInterrupts()

```
void Can_43_FLEXCAN_DisableControllerInterrupts (  
    uint8 Controller )
```

Disable INTs. SID is 0x04.

Switch OFF the controller's interrupts. This routine is called by:

- CanIf or an upper layer according to Autosar requirements.

Parameters

in	<i>Controller</i>	Can controller for which interrupts shall be disabled - based on configuration order list (CanControllerId).
----	-------------------	--

Returns

void

Precondition

Driver must be initialized before changing the interrupts state (en or dis).

Postcondition

Controller must not respond to any interrupt assertion.

6.1.5.5 Can_43_FLEXCAN_EnableControllerInterrupts()

```
void Can_43_FLEXCAN_EnableControllerInterrupts (
    uint8 Controller )
```

Enable INTs. SID is 0x05.

Switch ON the controller's interrupts. This routine is called by:

- CanIf or an upper layer according to Autosar requirements.

Parameters

in	<i>Controller</i>	Can controller for which interrupts shall be disabled - based on configuration order list (CanControllerId).
----	-------------------	--

Returns

void

Precondition

Driver must be initialized before changing the interrupts state (en or dis).

Postcondition

Controller must respond to interrupt assertion.

6.1.5.6 Can_43_FLEXCAN_GetControllerErrorState()

```
Std_ReturnType Can_43_FLEXCAN_GetControllerErrorState (
    uint8 ControllerId,
    Can_ErrorStateType * ErrorStatePtr )
```

Obtains the error state of the CAN controller.. SID is 0x11.

This routine is called by:

- CanIf or an upper layer according to Autosar requirements.

Parameters

in	<i>ControllerId</i>	Abstracted CanIf ControllerId which is assigned to a CAN controller, which is requested for ErrorState.
out	<i>ErrorStatePtr</i>	Pointer to a memory location, where the error state of the CAN controller will be stored.

Returns

Std_ReturnType Result of the transition.

Return values

<i>E_OK</i>	: Error state request has been accepted.
<i>E_NOT_OK</i>	: Error state request has not been accepted.

Precondition

Postcondition

6.1.5.7 Can_43_FLEXCAN_GetControllerMode()

```
Std_ReturnType Can_43_FLEXCAN_GetControllerMode (
    uint8 Controller,
    Can_ControllerStateType * ControllerModePtr )
```

Reports about the current status of the requested CAN controller. SID is 0x12.

This routine is called by:

- CanIf or an upper layer according to Autosar requirements.

Parameters

in	<i>Controller</i>	CAN controller for which the status shall be requested.
out	<i>ControllerModePtr</i>	Pointer to a memory location, where the current mode of the CAN controller will be stored.

Returns

Std_ReturnType Result of the transition.

Return values

<i>E_OK</i>	: Controller mode request has been accepted.
<i>E_NOT_OK</i>	: Controller mode request has not been accepted.

Precondition

Postcondition

6.1.5.8 Can_43_FLEXCAN_GetControllerRxErrorCounter()

```
Std_ReturnType Can_43_FLEXCAN_GetControllerRxErrorCounter (
    uint8 ControllerId,
    uint8 * RxErrorCounterPtr )
```

Return the Rx error counter for a CAN controller.

Return the Rx error counter for a CAN controller. This value might not be available for all CAN controller, in which case E_NOT_OK would be returned. Please note that the value of the counter might not be correct at the moment the API returns it, because the Rx counter is handled asynchronously in hardware. Applications should not trust this value for any assumption about the current bus state.

Parameters

in	<i>ControllerId</i>	CAN controller, whose current Rx error counter shall be acquired.
out	<i>RxErrorCounterPtr</i>	Pointer to a memory location, where the current Rx error counter of the CAN controller will be stored.

Returns

Std_ReturnType Result of the transition.

Return values

<i>E_OK</i>	Rx error counter available.
<i>E_NOT_OK</i>	Wrong ControllerId, or Rx error counter not available.

Precondition

Postcondition

6.1.5.9 Can_43_FLEXCAN_GetControllerTxErrorCounter()

```
Std_ReturnType Can_43_FLEXCAN_GetControllerTxErrorCounter (
    uint8 ControllerId,
    uint8 * TxErrorCounterPtr )
```

Return the Tx error counter for a CAN controller.

Return the Tx error counter for a CAN controller. This value might not be available for all CAN controller, in which case E_NOT_OK would be returned. Please note that the value of the counter might not be correct at the moment the API returns it, because the Tx counter is handled asynchronously in hardware. Applications should not trust this value for any assumption about the current bus state.

Parameters

in	<i>ControllerId</i>	CAN controller, whose current Tx error counter shall be acquired.
out	<i>TxErrorCounterPtr</i>	Pointer to a memory location, where the current Tx error counter of the CAN controller will be stored.

Returns

Std_ReturnType Result of the transition.

Return values

<i>E_OK</i>	Tx error counter available.
<i>E_NOT_OK</i>	Wrong ControllerId, or Tx error counter not available.

Precondition

Postcondition

6.1.5.10 Can_43_FLEXCAN_Write()

```
Std_ReturnType Can_43_FLEXCAN_Write (
    Can_HwHandleType Hth,
    const Can_PduType * PduInfo )
```

Transmit information on CAN bus. SID is 0x06.

Can_Write checks if hardware transmit object that is identified by the HTH is free. Can_Write checks if another Can_Write is ongoing for the same HTH. a) hardware transmit object is free: The mutex for that HTH is set to 'signaled' the ID, DLC and SDU are put in a format appropriate for the hardware (if necessary) and copied in the appropriate hardware registers or buffers. All necessary control operations to initiate the transmit are done. The mutex for that HTH is released. The function returns with E_OK. b) hardware transmit object is busy with another transmit request. The function returns with CAN_BUSY. c) A preemptive call of Can_Write has been issued, that could not be handled reentrant (i.e. a call with the same HTH). The function returns with CAN_BUSY the function is non blocking d) The hardware transmit object is busy with another transmit request for an L-PDU that has lower priority than that for the current request The transmission of the previous L-PDU is cancelled (asynchronously). The function returns with CAN_BUSY. This routine is called by:

- CanIf or an upper layer according to Autosar requirements.

Parameters

in	<i>Hth</i>	Information which HW-transmit handle shall be used for transmit. Implicitly this is also the information about the controller to use because the Hth numbers are unique inside one hardware unit.
in	<i>PduInfo</i>	Pointer to SDU user memory, DLC and Identifier.

Returns

Std_ReturnType Result of the write operation.

Return values

<i>E_OK</i>	Write command has been accepted.
<i>E_NOT_OK</i>	Development error occurred.
<i>CAN_BUSY</i>	No of TX hardware buffer available or preemptive call of Can_Write() that can't be implemented reentrant.

Module Documentation

Precondition

Driver must be initialized and MB must be configured for Tx.

Postcondition

The data can be transmitted or rejected because of another data with a higher priority.

6.2 FlexCAN

6.2.1 Detailed Description

Modules

- [FlexCAN_driver](#)

Data Structures

- struct [Flexcan_Ip_MsbuffCodeStatusType](#)
FlexCAN Message Buffer code and status for transmit and receive. [More...](#)
- struct [Flexcan_Ip_TimeSegmentType](#)
FlexCAN bitrate related structures. [More...](#)
- struct [Flexcan_Ip_PayloadSizeType](#)
FlexCAN Blocks payload sizes structure. [More...](#)
- struct [Flexcan_Ip_MsgBuffType](#)
FlexCAN message buffer structure. [More...](#)
- struct [Flexcan_Ip_MBhandleType](#)
Information needed for internal handling of a given MB. [More...](#)
- struct [Flexcan_Ip_StateType](#)
Internal driver state information. [More...](#)
- struct [Flexcan_Ip_ConfigType](#)
FlexCAN configuration. [More...](#)
- struct [Flexcan_Ip_IdTableType](#)
FlexCAN Rx FIFO ID filter table structure. [More...](#)
- struct [Flexcan_Ip_DataInfoType](#)
FlexCAN data info from user. [More...](#)

Macros

- `#define FlexCAN_Ip_Init(Flexcan_Ip_u8Instance, Flexcan_Ip_pState, Flexcan_Ip_pData)`
Initializes the FlexCAN peripheral.
- `#define FlexCAN_Ip_ConfigRxFifo(instance, id_format, id_filter_table)`
FlexCAN Rx FIFO field configuration.
- `#define FlexCAN_Ip_SetRxIndividualMask(instance, mb_idx, mask)`
Sets the FlexCAN Rx individual mask.
- `#define FlexCAN_Ip_SetRxMbGlobalMask(instance, mask)`
Sets the FlexCAN Rx MB global mask.
- `#define FlexCAN_Ip_SetRxFifoGlobalMask(instance, mask)`
Sets the FlexCAN Rx FIFO global mask. This mask is applied to all filters ID regardless the ID Filter format.
- `#define FlexCAN_Ip_MainFunctionBusOff(instance)`
Check a bus-off event.
- `#define FlexCAN_Ip_EnterFreezeMode(instance)`

- *Enter FlexCAN Module in Freeze Mode.*
- #define `FlexCAN_Ip_ExitFreezeMode(instance)`
- *Exit FlexCAN Module from Freeze Mode.*
- #define `FlexCAN_Ip_Deinit(instance)`
- *DeInitilize the FlexCAN instance driver.*
- #define `FlexCAN_Ip_GetStartMode(instance)`
- *Get Start Mode Status.*
- #define `FlexCAN_Ip_SetStartMode(instance)`
- *Set the FlexCAN instance in START mode.*
- #define `FlexCAN_Ip_SetStopMode(instance)`
- *Set the FlexCAN instance in STOP mode.*
- #define `FlexCAN_Ip_SetListenOnlyMode(instance, listenonlystate)`
- *Enable\Disable listen Only Mode.*
- #define `FlexCAN_Ip_SetRxMaskType(instance, type)`
- *Set RX masking type.*
- #define `FlexCAN_Ip_SetRxMb14Mask(instance, mask)`
- *Set Rx14Mask filter for message buffer 14.*
- #define `FlexCAN_Ip_SetRxMb15Mask(instance, mask)`
- *Set Rx15Mask filter for message buffer 15.*
- #define `FlexCAN_Ip_SetBitrate(instance, bitrate, enhExt)`
- *Sets the FlexCAN bit rate for standard frames or the arbitration phase of FD frames.*
- #define `FlexCAN_Ip_EnableInterrupts(u8Instance)`
- *Enable all interrupts configured.*
- #define `FlexCAN_Ip_DisableInterrupts(u8Instance)`
- *Disable all interrupts.*
- #define `FlexCAN_Ip_SetErrorInt(u8Instance, type, enable)`
- *Enable\Disable Error or BusOff Interrupt.*
- #define `FlexCAN_Ip_GetStopMode(instance)`
- *Get Stop Mode Status.*
- #define `FLEXCAN_IP_MCR_DEFAULT_VALUE_U32`
- *Default value for the MCR register.*
- #define `FLEXCAN_IP_CTRL1_DEFAULT_VALUE_U32`
- *Default value for the CTRL1 register.*
- #define `FLEXCAN_IP_TIMER_DEFAULT_VALUE_U32`
- *Default value for the TIMER register.*
- #define `FLEXCAN_IP_ECR_DEFAULT_VALUE_U32`
- *Default value for the ECR register.*
- #define `FLEXCAN_IP_ESR1_DEFAULT_VALUE_U32`
- *Default value for the ESR1 register.*
- #define `FLEXCAN_IP_IMASK_DEFAULT_VALUE_U32`
- *Default value for the IMASK2 register.*
- #define `FLEXCAN_IP_IFLAG_DEFAULT_VALUE_U32`
- *Default value for the IFLAG4 register.*
- #define `FLEXCAN_IP_CTRL2_DEFAULT_VALUE_U32`
- *Default value for the CTRL2 register.*
- #define `FLEXCAN_IP_CBT_DEFAULT_VALUE_U32`
- *Default value for the CTRL2 register.*

- `#define FLEXCAN_IP_FDCTRL_DEFAULT_VALUE_U32`
Default value for the FDCTRL register.
- `#define FLEXCAN_IP_FDCBT_DEFAULT_VALUE_U32`
Default value for the FDCBT register.
- `#define CAN_BASE_PTRS_HAS_FD`
- `#define FLEXCAN_IP_FEATURE_RAM_OFFSET`
FlexCAN Embedded RAM address offset.
- `#define FLEXCAN_IP_ALL_INT`
- `#define FLEXCAN_IP_BUS_OFF_INT`
- `#define FLEXCAN_IP_ERROR_INT`
- `#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATAB_RTR_SHIFT`
- `#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATAB_IDE_SHIFT`
- `#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATB_RTR_SHIFT`
- `#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATB_IDE_SHIFT`
- `#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATA_EXT_MASK`
- `#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATA_EXT_SHIFT`
- `#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATA_STD_MASK`
- `#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATA_STD_SHIFT`
- `#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATB_EXT_MASK`
- `#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATB_EXT_SHIFT1`
- `#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATB_EXT_SHIFT2`
- `#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATB_STD_MASK`
- `#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATB_STD_SHIFT1`
- `#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATB_STD_SHIFT2`
- `#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATB_EXT_CMP_SHIFT`
- `#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATC_EXT_MASK`
- `#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATC_STD_MASK`
- `#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATC_SHIFT1`
- `#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATC_SHIFT2`
- `#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATC_SHIFT3`
- `#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATC_SHIFT4`
- `#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATC_EXT_CMP_SHIFT`
- `#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATC_STD_CMP_SHIFT`
- `#define FLEXCAN_IP_REM_STORE_U32`
Remote Request Store enable.
- `#define FLEXCAN_IP_THREE_SAMPLES_U32`
Three samples to determine the value of received bit.
- `#define FLEXCAN_IP_BUSOFF_RECOVERY_U32`
Define how controller recover from bus off state.
- `#define FLEXCAN_IP_PROTOCOL_EXCEPTION_U32`
Protocol Exception.
- `#define FLEXCAN_IP_EDGE_FILTER_U32`
Edge Filter.
- `#define FLEXCAN_IP_ISO_U32`
CAN FD protocol according to ISO specification (ISO 11898-1)
- `#define FLEXCAN_IP_EACEN_U32`
Entire Frame Arbitration Field Comparison.

Types Reference

- typedef void(* [FlexCAN_Ip_CallbackType](#)) (uint8 instance, [Flexcan_Ip_EventType](#) eventType, uint32 buff←Idx, const [Flexcan_Ip_StateType](#) *flexcanState)
FlexCAN Driver callback function type.
- typedef void(* [FlexCAN_Ip_ErrorCallbackType](#)) (uint8 instance, [Flexcan_Ip_EventType](#) eventType, uint32 u32ErrStatus, const [Flexcan_Ip_StateType](#) *flexcanState)
FlexCAN Driver error callback function type.

Enum Reference

- enum
FlexCAN message buffer CODE for Rx buffers.
- enum
FlexCAN message buffer CODE FOR Tx buffers.
- enum [flexcan_int_type_t](#)
FlexCAN error interrupt types.
- enum [Flexcan_Ip_RxFifoTransferType](#)
The type of the RxFIFO transfer (interrupts/DMA).
- enum [Flexcan_Ip_RxFifoIdFilterNumType](#)
FlexCAN Rx FIFO filters number.
- enum [Flexcan_Ip_RxMaskType](#)
FlexCAN Rx mask type.
- enum [Flexcan_Ip_FdPayloadSizeType](#)
FlexCAN payload sizes.
- enum [Flexcan_Ip_ModesType](#)
FlexCAN operation modes.
- enum [Flexcan_Ip_MbStateType](#)
The state of a given MB (idle/Rx busy/Tx busy).
- enum [Flexcan_Ip_EventType](#)
The type of the event which occurred when the callback was invoked.
- enum [Flexcan_Ip_ErrorIntType](#)
FlexCAN error interrupt types.
- enum [Flexcan_Ip_MsgBuffIdType](#)
FlexCAN Message Buffer ID type.
- enum [Flexcan_Ip_RxFifoIdElementFormatType](#)
ID formats for Rx FIFO.
- enum [Flexcan_Ip_StatusType](#)
The status used and reported by FlexCAN Ip driver.

Function Reference

- [Flexcan_Ip_StatusType FlexCAN_Ip_Send](#) (uint8 instance, uint8 mb_idx, const [Flexcan_Ip_DataInfoType](#) *tx_info, uint32 msg_id, const uint8 *mb_data)
Sends a CAN frame using the specified message buffer.
- [Flexcan_Ip_StatusType FlexCAN_Ip_SendBlocking](#) (uint8 instance, uint8 mb_idx, const [Flexcan_Ip_DataInfoType](#) *tx_info, uint32 msg_id, const uint8 *mb_data, uint32 timeout_ms)
Sends a CAN frame using the specified message buffer, in a blocking manner.
- [Flexcan_Ip_StatusType FlexCAN_Ip_Receive](#) (uint8 instance, uint8 mb_idx, [Flexcan_Ip_MsgBuffType](#) *data, boolean isPolling)
Receives a CAN frame using the specified message buffer.
- [Flexcan_Ip_StatusType FlexCAN_Ip_RxFifo](#) (uint8 instance, [Flexcan_Ip_MsgBuffType](#) *data)
Receives a CAN frame using the message FIFO.
- [Flexcan_Ip_StatusType FlexCAN_Ip_RxFifoBlocking](#) (uint8 instance, [Flexcan_Ip_MsgBuffType](#) *data, uint32 timeout)
Receives a CAN frame using the message FIFO, in a blocking manner.
- [Flexcan_Ip_StatusType FlexCAN_Ip_ConfigRxMb](#) (uint8 instance, uint8 mb_idx, const [Flexcan_Ip_DataInfoType](#) *rx_info, uint32 msg_id)
FlexCAN receive message buffer field configuration.
- void [FlexCAN_Ip_MainFunctionRead](#) (uint8 instance, uint8 mb_idx)
Check a receive event.
- void [FlexCAN_Ip_MainFunctionWrite](#) (uint8 instance, uint8 mb_idx)
Check a Transmission event.
- [Flexcan_Ip_StatusType FlexCAN_Ip_GetTransferStatus](#) (uint8 instance, uint8 mb_idx)
Returns whether the previous FlexCAN transfer has finished.
- uint32 [FlexCAN_Ip_GetErrorStatus](#) (uint8 instance)
Get Error Status of FlexCAN.
- uint8 [FlexCAN_Ip_GetControllerTxErrorCounter](#) (uint8 instance)
Get Transmit error counter of FlexCAN.
- uint8 [FlexCAN_Ip_GetControllerRxErrorCounter](#) (uint8 instance)
Get Receive error counter of FlexCAN.
- void [FlexCAN_Ip_ClearErrorStatus](#) (uint8 instance, uint32 error)
Clear Error Status of FlexCAN.
- boolean [FlexCAN_Ip_GetBtrRate](#) (uint8 instance, [Flexcan_Ip_TimeSegmentType](#) *btrRate)
Gets the FlexCAN bit rate for standard frames or the arbitration phase of FD frames.
- boolean [FlexCAN_Ip_GetBuffStatusFlag](#) (uint8 instance, uint8 msgBuffIdx)
Get the Status of Message Buffer.
- void [FlexCAN_Ip_ClearBuffStatusFlag](#) (uint8 instance, uint8 msgBuffIdx)
Clear Message Buffer Status Flag.
- [Flexcan_Ip_StatusType FlexCAN_Ip_AbortTransfer](#) (uint8 u8Instance, uint8 mb_idx)
Ends a non-blocking FlexCAN transfer early.
- boolean [FlexCAN_Ip_GetListenOnlyMode](#) (uint8 instance)
Get the Status of Listen Only Mode.
- [Flexcan_Ip_StatusType FlexCAN_Ip_ReceiveBlocking](#) (uint8 instance, uint8 mb_idx, [Flexcan_Ip_MsgBuffType](#) *data, boolean isPolling, uint32 u32TimeoutMs)
Receives a CAN frame using the specified message buffer, in a blocking manner.
- [Flexcan_Ip_StatusType FlexCAN_Ip_ConfigRemoteResponseMb](#) (uint8 instance, uint8 mb_idx, const [Flexcan_Ip_DataInfoType](#) *tx_info, uint32 msg_id, const uint8 *mb_data)

- Configures a transmit message buffer for remote frame response.*
- [Flexcan_Ip_StatusType FlexCAN_Ip_ManualBusOffRecovery](#) (uint8 Instance)
Recover manually from bus-off if possible.
- void [FlexCAN_SetRxFifoFilter](#) (FLEXCAN_Type *base, [Flexcan_Ip_RxFifoIdElementFormatType](#) idFormat, const [Flexcan_Ip_IdTableType](#) *idFilterTable)
Sets the FlexCAN Rx FIFO fields.
- void [FlexCAN_ReadRxFifo](#) (const FLEXCAN_Type *base, [Flexcan_Ip_MsgBuffType](#) *rxFifo)
Gets the FlexCAN Rx FIFO data.
- [Flexcan_Ip_StatusType FlexCAN_ExitFreezeMode](#) (FLEXCAN_Type *base)
Unfreezes the FlexCAN module.
- void [FlexCAN_LockRxMsgBuff](#) (const FLEXCAN_Type *base, uint32 msgBuffIdx)
Locks the FlexCAN Rx message buffer.
- [Flexcan_Ip_StatusType FlexCAN_SetMsgBuffIntCmd](#) (FLEXCAN_Type *base, uint8 u8Instance, uint32 msgBuffIdx, boolean enable, boolean bIsIntActive)
Enables/Disables the FlexCAN Message Buffer interrupt.
- void [FlexCAN_DisableInterrupts](#) (FLEXCAN_Type *pBase)
Disable all interrupts.
- void [FlexCAN_EnableInterrupts](#) (FLEXCAN_Type *pBase, uint8 u8Instance)
Enable all interrupts configured.
- void [FlexCAN_SetTxMsgBuff](#) (volatile uint32 *const pMbAddr, const [Flexcan_Ip_MsbufCodeStatusType](#) *cs, uint32 msgId, const uint8 *msgData, const boolean isRemote)
Sets the FlexCAN message buffer fields for transmitting.
- [Flexcan_Ip_StatusType FlexCAN_EnableRxFifo](#) (FLEXCAN_Type *base, uint32 numOffFilters)
Enables the Rx FIFO.
- [Flexcan_Ip_StatusType FlexCAN_SetMaxMsgBuffNum](#) (FLEXCAN_Type *base, uint32 maxMsgBuffNum)
Sets the maximum number of Message Buffers.
- void [FlexCAN_SetRxMsgBuff](#) (const FLEXCAN_Type *base, uint32 msgBuffIdx, const [Flexcan_Ip_MsbufCodeStatusType](#) *cs, uint32 msgId)
Sets the FlexCAN message buffer fields for receiving.
- uint32 [FlexCAN_GetMsgBuffTimestamp](#) (const FLEXCAN_Type *base, uint32 msgBuffIdx)
Gets the message buffer timestamp value.
- void [FlexCAN_GetMsgBuff](#) (const FLEXCAN_Type *base, uint32 msgBuffIdx, [Flexcan_Ip_MsgBuffType](#) *msgBuff)
Gets the FlexCAN message buffer fields.
- uint8 [FlexCAN_GetMbPayloadSize](#) (const FLEXCAN_Type *base, uint32 maxMsgBuffNum)
Gets the payload size of the MBs.
- [Flexcan_Ip_StatusType FlexCAN_Init](#) (FLEXCAN_Type *base)
Initializes the FlexCAN controller.
- uint32 [FlexCAN_GetMaxMbNum](#) (const FLEXCAN_Type *base)
Get The Max no of MBs allowed on CAN instance.
- void [FlexCAN_SetOperationMode](#) (FLEXCAN_Type *base, [Flexcan_Ip_ModesType](#) mode)
Set operation mode.
- volatile uint32 * [FlexCAN_GetMsgBuffRegion](#) (const FLEXCAN_Type *base, uint32 msgBuffIdx)
Sets the FlexCAN message buffer fields for transmitting.
- void [FlexCAN_ConfigCtrlOptions](#) (FLEXCAN_Type *pBase, uint32 u32Options)
configure controller depending on options.
- void [FlexCAN_ResetImaskBuff](#) (uint8 Instance)
Reset Imask Buffers.

6.2.2 Data Structure Documentation

6.2.2.1 struct Flexcan_Ip_MsbuffCodeStatusType

FlexCAN Message Buffer code and status for transmit and receive.

Definition at line 317 of file FlexCAN_Ip_HwAccess.h.

Data Fields

Type	Name	Description
uint32	code	MB code for TX or RX buffers.
Flexcan_Ip_MsgBuffIdType	msgIdType	Defined by flexcan_mb_code_rx_t and flexcan_mb_code_tx_t Type of message ID (standard or extended)
uint32	dataLen	Length of Data in Bytes
boolean	fd_enable	
uint8	fd_padding	
boolean	enable_brs	

6.2.2.2 struct Flexcan_Ip_TimeSegmentType

FlexCAN bitrate related structures.

Definition at line 320 of file FlexCAN_Ip_Types.h.

Data Fields

Type	Name	Description
uint32	propSeg	Propagation segment
uint32	phaseSeg1	Phase segment 1
uint32	phaseSeg2	Phase segment 2
uint32	preDivider	Clock prescaler division factor
uint32	rJumpwidth	Resync jump width

6.2.2.3 struct Flexcan_Ip_PayloadSizeType

FlexCAN Blocks payload sizes structure.

Definition at line 332 of file FlexCAN_Ip_Types.h.

Data Fields

Type	Name	Description
Flexcan_Ip_FdPayloadSizeType	payloadBlock0	Payload for Ram Block 0

6.2.2.4 struct Flexcan_Ip_MsgBuffType

FlexCAN message buffer structure.

Definition at line 352 of file FlexCAN_Ip_Types.h.

Data Fields

Type	Name	Description
uint32	cs	Code and Status
uint32	msgId	Message Buffer ID
uint8	data[64]	Data bytes of the FlexCAN message
uint8	dataLen	Length of data in bytes
uint8	id_hit	Identifier Acceptance Filter Hit Indicator
uint32	time_stamp	Free-Running Counter Time Stamp

6.2.2.5 struct Flexcan_Ip_MBhandleType

Information needed for internal handling of a given MB.

Definition at line 364 of file FlexCAN_Ip_Types.h.

Data Fields

Type	Name	Description
Flexcan_Ip_MsgBuffType *	pMBmessage	The FlexCAN MB structure
volatile Flexcan_Ip_MbStateType	state	The state of the current MB (idle/Rx busy/Tx busy)
boolean	isPolling	True if the transfer is Polling Mode
boolean	isRemote	True if the frame is a remote frame
uint32	time_stamp	TimeStamp of the Message

6.2.2.6 struct Flexcan_Ip_StateType

Internal driver state information.

Note

The contents of this structure are internal to the driver and should not be modified by users. Also, contents of the structure are subject to change in future releases.

Definition at line 384 of file FlexCAN_Ip_Types.h.

Data Fields

- [Flexcan_Ip_MBhandleType](#) mbs [FLEXCAN_IP_FEATURE_MAX_MB_NUM]
- void(* [callback](#))(uint8 instance, [Flexcan_Ip_EventType](#) eventType, uint32 buffIdx, const struct FlexCANState *driverState)
- void * [callbackParam](#)
- void(* [error_callback](#))(uint8 instance, [Flexcan_Ip_EventType](#) eventType, uint32 u32ErrStatus, const struct FlexCANState *driverState)
- void * [errorCallbackParam](#)
- [Flexcan_Ip_RxFifoTransferType](#) transferType
- boolean [bIsLegacyFifoEn](#)
- uint32 [u32MaxMbNum](#)
- boolean [isIntActive](#)

6.2.2.6.1 Field Documentation

6.2.2.6.1.1 mbs [Flexcan_Ip_MBhandleType](#) mbs [FLEXCAN_IP_FEATURE_MAX_MB_NUM]

Array containing information related to each MB

Definition at line 386 of file FlexCAN_Ip_Types.h.

6.2.2.6.1.2 **callback** void(* [callback](#))(uint8 instance, [Flexcan_Ip_EventType](#) eventType, uint32 buffIdx, const struct FlexCANState *driverState)

IRQ handler callback function.

Definition at line 392 of file FlexCAN_Ip_Types.h.

6.2.2.6.1.3 **callbackParam** void* [callbackParam](#)

Parameter used to pass user data when invoking the callback function.

Definition at line 397 of file FlexCAN_Ip_Types.h.

6.2.2.6.1.4 error_callback void(* error_callback) (uint8 instance, Flexcan_Ip_EventType eventType, uint32 u32ErrStatus, const struct FlexCANState *driverState)

Error IRQ handler callback function.

Definition at line 400 of file FlexCAN_Ip_Types.h.

6.2.2.6.1.5 errorCallbackParam void* errorCallbackParam

Parameter used to pass user data when invoking the error callback function.

Definition at line 406 of file FlexCAN_Ip_Types.h.

6.2.2.6.1.6 transferType Flexcan_Ip_RxFifoTransferType transferType

Type of Rx FIFO transfer.

Definition at line 415 of file FlexCAN_Ip_Types.h.

6.2.2.6.1.7 bIsLegacyFifoEn boolean bIsLegacyFifoEn

This controls whether the Rx FIFO feature is enabled or not.

Definition at line 416 of file FlexCAN_Ip_Types.h.

6.2.2.6.1.8 u32MaxMbNum uint32 u32MaxMbNum

The maximum number of Message Buffers.

Definition at line 420 of file FlexCAN_Ip_Types.h.

6.2.2.6.1.9 isIntActive boolean isIntActive

Save status of enabling/disabling interrupts in runtime.

Definition at line 421 of file FlexCAN_Ip_Types.h.

6.2.2.7 struct Flexcan_Ip_ConfigType

FlexCAN configuration.

Definition at line 446 of file FlexCAN_Ip_Types.h.

Data Fields

Type	Name	Description
uint32	max_num_mb	The maximum number of Message Buffers
Flexcan_Ip_RxFifoIdFilterNumType	num_id_filters	The number of RX FIFO ID filters needed
boolean	is_rx_fifo_needed	1 if needed; 0 if not. This controls whether the Rx FIFO feature is enabled or not.
Flexcan_Ip_ModesType	flexcanMode	User configurable FlexCAN operation modes.
uint32	ctrlOptions	Use of different features support like ISO-FD, EDGE_FILTER, AUTO_BussOffRecovery, Protocol_Exception.
Flexcan_Ip_PayloadSizeType	payload	The payload size of the mailboxes specified in bytes for every partition block
boolean	fd_enable	Enable/Disable the Flexible Data Rate feature.
boolean	enhCbtEnable	The use of enhanced bit time segments format from ExCBT register, instead of CTRL1 or CBT register
boolean	bitRateSwitch	Enable of BitRate Switch support for FD frames
Flexcan_Ip_TimeSegmentType	bitrate	The bitrate used for standard frames or for the arbitration phase of FD frames.
Flexcan_Ip_TimeSegmentType	bitrate_cbt	The bitrate used for the data phase of FD frames.
Flexcan_Ip_RxFifoTransferType	transfer_type	Specifies if the Rx FIFO uses interrupts or DMA.
FlexCAN_Ip_CallbackType	Callback	The Callback for Rx or Tx DMA Events
FlexCAN_Ip_ErrorCallbackType	ErrorCallback	The ErrorCallback for Error Events

6.2.2.8 struct Flexcan_Ip_IdTableType

FlexCAN Rx FIFO ID filter table structure.

Structure Used to configure and add filters to Legacy RxFIFO

Definition at line 495 of file FlexCAN_Ip_Types.h.

Data Fields

Type	Name	Description
boolean	isRemoteFrame	Remote frame
boolean	isExtendedFrame	Extended frame
uint32	id	Rx FIFO ID filter element

6.2.2.9 struct Flexcan_Ip_DataInfoType

FlexCAN data info from user.

Module Documentation

This structure defines the members used to configure the Frame Parameters used to be Send or Receive. Some parameters are available based on configuration of driver like: `fd_enable`, `fd_padding`, `enable_brs`.

Definition at line 509 of file `FlexCAN_Ip_Types.h`.

Data Fields

Type	Name	Description
Flexcan_Ip_MsgBuffIdType	<code>msg_id_type</code>	Type of message ID (standard or extended)
<code>uint32</code>	<code>data_length</code>	Length of Data in Bytes
<code>boolean</code>	<code>is_remote</code>	Specifies if the frame is standard or remote
<code>boolean</code>	<code>is_polling</code>	Specifies if the MB is in polling mode

6.2.3 Macro Definition Documentation

6.2.3.1 FlexCAN_Ip_Init

```
#define FlexCAN_Ip_Init(  
    Flexcan_Ip_u8Instance,  
    Flexcan_Ip_pState,  
    Flexcan_Ip_pData )
```

Initializes the FlexCAN peripheral.

This function will config FlexCAN module and will leave the module in freeze mode.

Parameters

in	<i>Flexcan_Ip_u8Instance</i>	A FlexCAN instance number
	<i>[in]</i>	

Definition at line 212 of file `FlexCAN_Ip.h`.

6.2.3.2 FlexCAN_Ip_ConfigRxFifo

```
#define FlexCAN_Ip_ConfigRxFifo(  
    instance,  
    id_format,  
    id_filter_table )
```

FlexCAN Rx FIFO field configuration.

Each element in the ID filter table specifies an ID to be used as acceptance criteria for the FIFO as follows:

- for format A: In the standard frame format, bits 10 to 0 of the ID are used for frame identification. In the extended frame format, bits 28 to 0 are used.
- for format B: In the standard frame format, bits 10 to 0 of the ID are used for frame identification. In the extended frame format, only the 14 most significant bits (28 to 15) of the ID are compared to the 14 most significant bits (28 to 15) of the received ID.
- for format C: In both standard and extended frame formats, only the 8 most significant bits (7 to 0 for standard, 28 to 21 for extended) of the ID are compared to the 8 most significant bits (7 to 0 for standard, 28 to 21 for extended) of the received ID.

Parameters

in	<i>instance</i>	A FlexCAN instance number
in	<i>id_format</i>	The format of the Rx FIFO ID Filter Table Elements
in	<i>id_filter_table</i>	The ID filter table elements which contain RTR bit, IDE bit, and Rx message ID

Returns

FLEXCAN_STATUS_SUCCESS if successful;
 FLEXCAN_STATUS_ERROR if fail to set;
 FLEXCAN_STATUS_TIMEOUT if fail to configure in configured timeout value.

Note

The number of elements in the ID filter table is defined by the following formula:

- for format A: the number of Rx FIFO ID filters
- for format B: twice the number of Rx FIFO ID filters
- for format C: four times the number of Rx FIFO ID filters The user must provide the exact number of elements in order to avoid any misconfiguration. This function should be called from StopMode or FreezeMode.

Definition at line 304 of file FlexCAN_Ip.h.

6.2.3.3 FlexCAN_Ip_SetRxIndividualMask

```
#define FlexCAN_Ip_SetRxIndividualMask(  
    instance,  
    mb_idx,  
    mask )
```

Sets the FlexCAN Rx individual mask.

This function will set directly the mask value as is provided.

Parameters

in	<i>instance</i>	A FlexCAN instance number
in	<i>mb_idx</i>	Index of the message buffer
in	<i>mask</i>	Mask value

Note

This function should be called from StopMode or FreezeMode.

Returns

FLEXCAN_STATUS_SUCCESS if successful;
FLEXCAN_STATUS_BUFF_OUT_OF_RANGE if the index of a message buffer is invalid.

Definition at line 409 of file FlexCAN_Ip.h.

6.2.3.4 FlexCAN_Ip_SetRxMbGlobalMask

```
#define FlexCAN_Ip_SetRxMbGlobalMask(  
    instance,  
    mask )
```

Sets the FlexCAN Rx MB global mask.

This function will set directly the mask value as is provided.

Parameters

in	<i>instance</i>	A FlexCAN instance number
in	<i>mask</i>	Mask value

Note

This function should be called from StopMode or FreezeMode.

Returns

FLEXCAN_STATUS_SUCCESS if successful;
FLEXCAN_STATUS_ERROR if fail to set;
FLEXCAN_STATUS_TIMEOUT if fail to configure in configured timeout value.

Definition at line 423 of file FlexCAN_Ip.h.

6.2.3.5 FlexCAN_Ip_SetRxFifoGlobalMask

```
#define FlexCAN_Ip_SetRxFifoGlobalMask(  
    instance,  
    mask )
```

Sets the FlexCAN Rx FIFO global mask. This mask is applied to all filters ID regardless the ID Filter format.

Parameters

in	<i>instance</i>	A FlexCAN instance number
in	<i>mask</i>	Mask Value.

Note

This function should be called from StopMode or FreezeMode.

Returns

FLEXCAN_STATUS_SUCCESS if successful;
FLEXCAN_STATUS_ERROR if fail to set;
FLEXCAN_STATUS_TIMEOUT if fail to configure in configured timeout value.

Definition at line 437 of file FlexCAN_Ip.h.

6.2.3.6 FlexCAN_Ip_MainFunctionBusOff

```
#define FlexCAN_Ip_MainFunctionBusOff(  
    instance )
```

Check a bus-off event.

This function will check bus activity of FlexCAN module and if a bus off event is detected will suspend the future bus activities by setting module in stop mode.

Parameters

in	<i>instance</i>	A FlexCAN instance number
----	-----------------	---------------------------

Returns

FLEXCAN_STATUS_SUCCESS if successful busoff and set on stop;
FLEXCAN_STATUS_ERROR if no busoff event detected;
FLEXCAN_STATUS_TIMEOUT if fail to configure in the configured timeout value.

Definition at line 469 of file FlexCAN_Ip.h.

6.2.3.7 FlexCAN_Ip_EnterFreezeMode

```
#define FlexCAN_Ip_EnterFreezeMode(  
    instance )
```

Enter FlexCAN Module in Freeze Mode.

This function will suspend bus activity of FlexCAN module and set it to Freeze Mode to allow module configuration.

Parameters

in	<i>instance</i>	A FlexCAN instance number
----	-----------------	---------------------------

Returns

FLEXCAN_STATUS_SUCCESS if successful;
FLEXCAN_STATUS_TIMEOUT if fail to configure in the configured timeout value.

Definition at line 481 of file FlexCAN_Ip.h.

6.2.3.8 FlexCAN_Ip_ExitFreezeMode

```
#define FlexCAN_Ip_ExitFreezeMode(  
    instance )
```

Exit FlexCAN Module from Freeze Mode.

This function will allow FlexCAN module to participate to the BUS activity and restore normal operation of the driver.

Parameters

in	<i>instance</i>	A FlexCAN instance number
----	-----------------	---------------------------

Note

This function should be called from FreezeMode.

Returns

FLEXCAN_STATUS_SUCCESS if successful;
FLEXCAN_STATUS_TIMEOUT if fail to configure in the configured timeout value.

Definition at line 494 of file FlexCAN_Ip.h.

6.2.3.9 FlexCAN_Ip_Deinit

```
#define FlexCAN_Ip_Deinit(  
    instance )
```

DeInitilize the FlexCAN instance driver.

This function will make future operataions of FlexCAN instance imposible and will restore it's state to default value as before initialization.

Parameters

in	<i>instance</i>	A FlexCAN instance number
----	-----------------	---------------------------

Returns

FLEXCAN_STATUS_SUCCESS if successful;
FLEXCAN_STATUS_TIMEOUT if fail to configure in configured timeout value.

Definition at line 506 of file FlexCAN_Ip.h.

6.2.3.10 FlexCAN_Ip_GetStartMode

```
#define FlexCAN_Ip_GetStartMode(  
    instance )
```

Get Start Mode Status.

Return if the instance is in Start Mode

Parameters

in	<i>instance</i>	A FlexCAN instance number
----	-----------------	---------------------------

Returns

True instance is in START Mode False instance is not in START Mode

Definition at line 517 of file FlexCAN_Ip.h.

6.2.3.11 FlexCAN_Ip_SetStartMode

```
#define FlexCAN_Ip_SetStartMode(  
    instance )
```

Set the FlexCAN instance in START mode.

Set the FlexCAN instance in START mode, allowing to participate to bus transfers.

Parameters

in	<i>instance</i>	A FlexCAN instance number
----	-----------------	---------------------------

Returns

FLEXCAN_STATUS_SUCCESS if successful;
FLEXCAN_STATUS_TIMEOUT if fail to configure in configured timeout value.

Definition at line 527 of file FlexCAN_Ip.h.

6.2.3.12 FlexCAN_Ip_SetStopMode

```
#define FlexCAN_Ip_SetStopMode(  
    instance )
```

Set the FlexCAN instance in STOP mode.

Set the FlexCAN instance in START mode, this will prevent instance to participate to bus transactions and disable module clocks.

Parameters

in	<i>instance</i>	A FlexCAN instance number
----	-----------------	---------------------------

Returns

FLEXCAN_STATUS_SUCCESS if successful;
FLEXCAN_STATUS_TIMEOUT if fail to configure in configured timeout value.

Definition at line 539 of file FlexCAN_Ip.h.

6.2.3.13 FlexCAN_Ip_SetListenOnlyMode

```
#define FlexCAN_Ip_SetListenOnlyMode(  
    instance,  
    listenonlystate )
```

Enable\Disable listen Only Mode.

This function will Enable or Disable listen Only Mode.

Note

This function should be called from StopMode or FreezeMode.

Parameters

in	<i>u8Instance</i>	A FlexCAN instance number
in	<i>listenonlystate</i>	Enable\Disable interrupt selected

Returns

FLEXCAN_STATUS_SUCCESS if successful;
FLEXCAN_STATUS_ERROR if fail to set;

Definition at line 552 of file FlexCAN_Ip.h.

6.2.3.14 FlexCAN_Ip_SetRxMaskType

```
#define FlexCAN_Ip_SetRxMaskType(  
    instance,  
    type )
```

Set RX masking type.

This function will set RX masking type as RX global mask or RX individual mask

Parameters

in	<i>instance</i>	A FlexCAN instance number
in	<i>type</i>	FlexCAN Rx mask type

Note

This function should be called from StopMode or FreezeMode.

Returns

FLEXCAN_STATUS_SUCCESS if successful

FLEXCAN_STATUS_ERROR if controller is not in freeze mode

Definition at line 623 of file FlexCAN_Ip.h.

6.2.3.15 FlexCAN_Ip_SetRxMb14Mask

```
#define FlexCAN_Ip_SetRxMb14Mask(  
    instance,  
    mask )
```

Set Rx14Mask filter for message buffer 14.

This function will set directly the mask value as is provided.

Parameters

in	<i>instance</i>	A FlexCAN instance number
in	<i>mask</i>	The value applied for mask

Note

This function should be called from StopMode or FreezeMode.

Returns

FLEXCAN_STATUS_SUCCESS if successful

FLEXCAN_STATUS_ERROR if controller is not in freeze mode

Definition at line 636 of file FlexCAN_Ip.h.

6.2.3.16 FlexCAN_Ip_SetRxMb15Mask

```
#define FlexCAN_Ip_SetRxMb15Mask(  
    instance,  
    mask )
```

Set Rx15Mask filter for message buffer 15.

This function will set directly the mask value as is provided.

Parameters

in	<i>instance</i>	A FlexCAN instance number
in	<i>mask</i>	The value applied for mask

Note

This function should be called from StopMode or FreezeMode.

Returns

FLEXCAN_STATUS_SUCCESS if successful
FLEXCAN_STATUS_ERROR if controller is not in freeze mode

Definition at line 649 of file FlexCAN_Ip.h.

6.2.3.17 FlexCAN_Ip_SetBtrate

```
#define FlexCAN_Ip_SetBtrate(  
    instance,  
    btrate,  
    enhExt )
```

Sets the FlexCAN bit rate for standard frames or the arbitration phase of FD frames.

This function request the FlexCAN module to be in Stop Mode or in Freeze Mode.

Parameters

in	<i>instance</i>	A FlexCAN instance number
in	<i>btrate</i>	A pointer to the FlexCAN bit rate settings.
in	<i>enhExt</i>	The time segments used are set in Enhanced Time Seg Registers

Returns

FLEXCAN_STATUS_SUCCESS if successful;
FLEXCAN_STATUS_ERROR if fail to set;
FLEXCAN_STATUS_TIMEOUT if fail to configure in configured timeout value.

Definition at line 674 of file FlexCAN_Ip.h.

6.2.3.18 FlexCAN_Ip_EnableInterrupts

```
#define FlexCAN_Ip_EnableInterrupts(  
    u8Instance )
```

Enable all interrupts configured.

Enable all interrupts configured.

Parameters

in	<i>u8Instance</i>	A FlexCAN instance number
----	-------------------	---------------------------

Returns

FLEXCAN_STATUS_SUCCESS if successful
FLEXCAN_STATUS_ERROR if fail to set

Definition at line 791 of file FlexCAN_Ip.h.

6.2.3.19 FlexCAN_Ip_DisableInterrupts

```
#define FlexCAN_Ip_DisableInterrupts(  
    u8Instance )
```

Disable all interrupts.

Disable all interrupts.

Parameters

in	<i>u8Instance</i>	A FlexCAN instance number
----	-------------------	---------------------------

Returns

FLEXCAN_STATUS_SUCCESS if successful
FLEXCAN_STATUS_ERROR if fail to set

Definition at line 801 of file FlexCAN_Ip.h.

6.2.3.20 FlexCAN_Ip_SetErrorInt

```
#define FlexCAN_Ip_SetErrorInt(  
    u8Instance,  
    type,  
    enable )
```

Enable\Disable Error or BusOff Interrupt.

This function will set Error or BusOff interrupt, Error Fast is available only if FD CAN support is active. @Note This function should be called from StopMode or FreezeMode. When an error interrupt is set and error callback function is installed, The error callback function will be invoked with a respective event occurred and status of ESR1 register: In the callback, if another event(got from ESR1 register) recognized(Error, Error Fast, Bus Off, Tx/Rx warning) Then it should be cleared by FlexCAN_Ip_ClearErrorStatus with a respective mask to avoid duplication.

Parameters

in	<i>u8Instance</i>	A FlexCAN instance number
in	<i>type</i>	Interrupt Type
in	<i>enable</i>	Enable\Disable interrupt selected

Returns

FLEXCAN_STATUS_SUCCESS if successful;
FLEXCAN_STATUS_ERROR if fail to set;
FLEXCAN_STATUS_TIMEOUT if fail to configure in configured timeout value.

Definition at line 821 of file FlexCAN_Ip.h.

6.2.3.21 FlexCAN_Ip_GetStopMode

```
#define FlexCAN_Ip_GetStopMode(  
    instance )
```

Get Stop Mode Status.

Return if the instance is in Stop Mode

Parameters

in	<i>instance</i>	A FlexCAN instance number
----	-----------------	---------------------------

Returns

True instance is in STOP Mode False instance is not in STOP Mode

Definition at line 856 of file FlexCAN_Ip.h.

6.2.3.22 FLEXCAN_IP_MCR_DEFAULT_VALUE_U32

```
#define FLEXCAN_IP_MCR_DEFAULT_VALUE_U32
```

Default value for the MCR register.

Definition at line 93 of file FlexCAN_Ip_DeviceReg.h.

6.2.3.23 FLEXCAN_IP_CTRL1_DEFAULT_VALUE_U32

```
#define FLEXCAN_IP_CTRL1_DEFAULT_VALUE_U32
```

Default value for the CTRL1 register.

Definition at line 98 of file FlexCAN_Ip_DeviceReg.h.

6.2.3.24 FLEXCAN_IP_TIMER_DEFAULT_VALUE_U32

```
#define FLEXCAN_IP_TIMER_DEFAULT_VALUE_U32
```

Default value for the TIMER register.

Definition at line 103 of file FlexCAN_Ip_DeviceReg.h.

6.2.3.25 FLEXCAN_IP_ECR_DEFAULT_VALUE_U32

```
#define FLEXCAN_IP_ECR_DEFAULT_VALUE_U32
```

Default value for the ECR register.

Definition at line 108 of file FlexCAN_Ip_DeviceReg.h.

6.2.3.26 FLEXCAN_IP_ESR1_DEFAULT_VALUE_U32

```
#define FLEXCAN_IP_ESR1_DEFAULT_VALUE_U32
```

Default value for the ESR1 register.

Definition at line 113 of file FlexCAN_Ip_DeviceReg.h.

6.2.3.27 FLEXCAN_IP_IMASK_DEFAULT_VALUE_U32

```
#define FLEXCAN_IP_IMASK_DEFAULT_VALUE_U32
```

Default value for the IMASK2 register.

Definition at line 118 of file FlexCAN_Ip_DeviceReg.h.

6.2.3.28 FLEXCAN_IP_IFLAG_DEFAULT_VALUE_U32

```
#define FLEXCAN_IP_IFLAG_DEFAULT_VALUE_U32
```

Default value for the IFLAG4 register.

Definition at line 123 of file FlexCAN_Ip_DeviceReg.h.

6.2.3.29 FLEXCAN_IP_CTRL2_DEFAULT_VALUE_U32

```
#define FLEXCAN_IP_CTRL2_DEFAULT_VALUE_U32
```

Default value for the CTRL2 register.

Definition at line 128 of file FlexCAN_Ip_DeviceReg.h.

6.2.3.30 FLEXCAN_IP_CBT_DEFAULT_VALUE_U32

```
#define FLEXCAN_IP_CBT_DEFAULT_VALUE_U32
```

Default value for the CTRL2 register.

Definition at line 133 of file FlexCAN_Ip_DeviceReg.h.

6.2.3.31 FLEXCAN_IP_FDCTRL_DEFAULT_VALUE_U32

```
#define FLEXCAN_IP_FDCTRL_DEFAULT_VALUE_U32
```

Default value for the FDCTRL register.

Definition at line 138 of file FlexCAN_Ip_DeviceReg.h.

6.2.3.32 FLEXCAN_IP_FDCBT_DEFAULT_VALUE_U32

```
#define FLEXCAN_IP_FDCBT_DEFAULT_VALUE_U32
```

Default value for the FDCBT register.

Definition at line 143 of file FlexCAN_Ip_DeviceReg.h.

6.2.3.33 CAN_BASE_PTRS_HAS_FD

```
#define CAN_BASE_PTRS_HAS_FD
```

Array init has of CAN Peripheral base address has FD capability

Definition at line 155 of file FlexCAN_Ip_DeviceReg.h.

6.2.3.34 FLEXCAN_IP_FEATURE_RAM_OFFSET

```
#define FLEXCAN_IP_FEATURE_RAM_OFFSET
```

FlexCAN Embedded RAM address offset.

Definition at line 130 of file FlexCAN_Ip_HwAccess.h.

6.2.3.35 FLEXCAN_IP_ALL_INT

```
#define FLEXCAN_IP_ALL_INT
```

Masks for wakeup, error, bus off

Definition at line 137 of file FlexCAN_Ip_HwAccess.h.

6.2.3.36 FLEXCAN_IP_BUS_OFF_INT

```
#define FLEXCAN_IP_BUS_OFF_INT
```

Masks for busOff, Tx/Rx Warning

Definition at line 140 of file FlexCAN_Ip_HwAccess.h.

6.2.3.37 FLEXCAN_IP_ERROR_INT

```
#define FLEXCAN_IP_ERROR_INT
```

Masks for ErrorOvr, ErrorFast, Error

Definition at line 141 of file FlexCAN_Ip_HwAccess.h.

6.2.3.38 FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATAB_RTR_SHIFT

```
#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATAB_RTR_SHIFT
```

FlexCAN RX FIFO ID filter

Definition at line 188 of file FlexCAN_Ip_HwAccess.h.

6.2.3.39 FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATAB_IDE_SHIFT

```
#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATAB_IDE_SHIFT
```

format A&B RTR mask. FlexCAN RX FIFO ID filter

Definition at line 190 of file FlexCAN_Ip_HwAccess.h.

6.2.3.40 FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATB_RTR_SHIFT

```
#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATB_RTR_SHIFT
```

format A&B IDE mask. FlexCAN RX FIFO ID filter

Definition at line 192 of file FlexCAN_Ip_HwAccess.h.

6.2.3.41 FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATB_IDE_SHIFT

```
#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATB_IDE_SHIFT
```

format B RTR-2 mask. FlexCAN RX FIFO ID filter

Definition at line 194 of file FlexCAN_Ip_HwAccess.h.

6.2.3.42 FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATA_EXT_MASK

```
#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATA_EXT_MASK
```

format B IDE-2 mask. FlexCAN RX FIFO ID filter

Definition at line 196 of file FlexCAN_Ip_HwAccess.h.

6.2.3.43 FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATA_EXT_SHIFT

```
#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATA_EXT_SHIFT
```

format A extended mask. FlexCAN RX FIFO ID filter

Definition at line 198 of file FlexCAN_Ip_HwAccess.h.

6.2.3.44 FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATA_STD_MASK

```
#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATA_STD_MASK
```

format A extended shift. FlexCAN RX FIFO ID filter

Definition at line 200 of file FlexCAN_Ip_HwAccess.h.

6.2.3.45 FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATA_STD_SHIFT

```
#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATA_STD_SHIFT
```

format A standard mask. FlexCAN RX FIFO ID filter

Definition at line 202 of file FlexCAN_Ip_HwAccess.h.

6.2.3.46 FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATB_EXT_MASK

```
#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATB_EXT_MASK
```

format A standard shift. FlexCAN RX FIFO ID filter

Definition at line 204 of file FlexCAN_Ip_HwAccess.h.

6.2.3.47 FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATB_EXT_SHIFT1

```
#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATB_EXT_SHIFT1
```

format B extended mask1. FlexCAN RX FIFO ID filter

Definition at line 206 of file FlexCAN_Ip_HwAccess.h.

6.2.3.48 FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATB_EXT_SHIFT2

```
#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATB_EXT_SHIFT2
```

format B extended shift 1. FlexCAN RX FIFO ID filter

Definition at line 208 of file FlexCAN_Ip_HwAccess.h.

6.2.3.49 FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATB_STD_MASK

```
#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATB_STD_MASK
```

format B extended shift 2. FlexCAN RX FIFO ID filter

Definition at line 210 of file FlexCAN_Ip_HwAccess.h.

6.2.3.50 FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATB_STD_SHIFT1

```
#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATB_STD_SHIFT1
```

format B standard mask. FlexCAN RX FIFO ID filter

Definition at line 212 of file FlexCAN_Ip_HwAccess.h.

6.2.3.51 FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATB_STD_SHIFT2

```
#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATB_STD_SHIFT2
```

format B standard shift1. FlexCAN RX FIFO ID filter

Definition at line 214 of file FlexCAN_Ip_HwAccess.h.

6.2.3.52 FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATB_EXT_CMP_SHIFT

```
#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATB_EXT_CMP_SHIFT
```

format B standard shift2. FlexCAN RX FIFO ID filter

Definition at line 216 of file FlexCAN_Ip_HwAccess.h.

6.2.3.53 FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATC_EXT_MASK

```
#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATC_EXT_MASK
```

format B extended compare shift. FlexCAN RX FIFO ID filter

Definition at line 218 of file FlexCAN_Ip_HwAccess.h.

6.2.3.54 FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATC_STD_MASK

```
#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATC_STD_MASK
```

format C mask. FlexCAN RX FIFO ID filter

Definition at line 220 of file FlexCAN_Ip_HwAccess.h.

6.2.3.55 FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATC_SHIFT1

```
#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATC_SHIFT1
```

format C mask. FlexCAN RX FIFO ID filter

Definition at line 222 of file FlexCAN_Ip_HwAccess.h.

6.2.3.56 FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATC_SHIFT2

```
#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATC_SHIFT2
```

format C shift1. FlexCAN RX FIFO ID filter

Definition at line 224 of file FlexCAN_Ip_HwAccess.h.

6.2.3.57 FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATC_SHIFT3

```
#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATC_SHIFT3
```

format C shift2. FlexCAN RX FIFO ID filter

Definition at line 226 of file FlexCAN_Ip_HwAccess.h.

6.2.3.58 FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATC_SHIFT4

```
#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATC_SHIFT4
```

format C shift3. FlexCAN RX FIFO ID filter

Definition at line 228 of file FlexCAN_Ip_HwAccess.h.

6.2.3.59 FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATC_EXT_CMP_SHIFT

```
#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATC_EXT_CMP_SHIFT
```

format C shift4. FlexCAN RX FIFO ID filter

Definition at line 230 of file FlexCAN_Ip_HwAccess.h.

6.2.3.60 FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATC_STD_CMP_SHIFT

```
#define FLEXCAN_IP_RX_FIFO_ID_FILTER_FORMATC_STD_CMP_SHIFT
```

format C extended compare shift. FlexCAN RX FIFO ID filter

Definition at line 232 of file FlexCAN_Ip_HwAccess.h.

6.2.3.61 FLEXCAN_IP_REM_STORE_U32

```
#define FLEXCAN_IP_REM_STORE_U32
```

Remote Request Store enable.

Definition at line 87 of file FlexCAN_Ip_Types.h.

6.2.3.62 FLEXCAN_IP_THREE_SAMPLES_U32

```
#define FLEXCAN_IP_THREE_SAMPLES_U32
```

Three samples to determine the value of received bit.

Definition at line 89 of file FlexCAN_Ip_Types.h.

6.2.3.63 FLEXCAN_IP_BUSOFF_RECOVERY_U32

```
#define FLEXCAN_IP_BUSOFF_RECOVERY_U32
```

Define how controller recover from bus off state.

Definition at line 91 of file FlexCAN_Ip_Types.h.

6.2.3.64 FLEXCAN_IP_PROTOCOL_EXCEPTION_U32

```
#define FLEXCAN_IP_PROTOCOL_EXCEPTION_U32
```

Protocol Exception.

Definition at line 93 of file FlexCAN_Ip_Types.h.

6.2.3.65 FLEXCAN_IP_EDGE_FILTER_U32

```
#define FLEXCAN_IP_EDGE_FILTER_U32
```

Edge Filter.

Definition at line 95 of file FlexCAN_Ip_Types.h.

6.2.3.66 FLEXCAN_IP_ISO_U32

```
#define FLEXCAN_IP_ISO_U32
```

CAN FD protocol according to ISO specification (ISO 11898-1)

Definition at line 97 of file FlexCAN_Ip_Types.h.

6.2.3.67 FLEXCAN_IP_EACEN_U32

```
#define FLEXCAN_IP_EACEN_U32
```

Entire Frame Arbitration Field Comparison.

Definition at line 99 of file FlexCAN_Ip_Types.h.

6.2.4 Types Reference

6.2.4.1 FlexCAN_Ip_CallbackType

```
typedef void(* FlexCAN_Ip_CallbackType) (uint8 instance, Flexcan_Ip_EventType eventType, uint32 buffIdx,
const Flexcan_Ip_StateType *flexcanState)
```

FlexCAN Driver callback function type.

Definition at line 427 of file FlexCAN_Ip_Types.h.

6.2.4.2 FlexCAN_Ip_ErrorCallbackType

```
typedef void(* FlexCAN_Ip_ErrorCallbackType) (uint8 instance, Flexcan_Ip_EventType eventType, uint32
u32ErrStatus, const Flexcan_Ip_StateType *flexcanState)
```

FlexCAN Driver error callback function type.

Definition at line 436 of file FlexCAN_Ip_Types.h.

6.2.5 Enum Reference

6.2.5.1 anonymous enum

```
anonymous enum
```

FlexCAN message buffer CODE for Rx buffers.

Enumerator

FLEXCAN_RX_INACTIVE	MB is not active.
FLEXCAN_RX_FULL	MB is full.
FLEXCAN_RX_EMPTY	MB is active and empty.
FLEXCAN_RX_OVERRUN	MB is overwritten into a full buffer.
FLEXCAN_RX_BUSY	FlexCAN is updating the contents of the MB.
FLEXCAN_RX_RANSWER	The CPU must not access the MB. A frame was configured to recognize a Remote Request Frame
FLEXCAN_RX_NOT_USED	and transmit a Response Frame in return. Not used

Definition at line 276 of file FlexCAN_Ip_HwAccess.h.

6.2.5.2 anonymous enum

anonymous enum

FlexCAN message buffer CODE FOR Tx buffers.

Enumerator

FLEXCAN_TX_INACTIVE	MB is not active.
FLEXCAN_TX_ABORT	MB is aborted.
FLEXCAN_TX_DATA	MB is a TX Data Frame(MB RTR must be 0).
FLEXCAN_TX_REMOTE	MB is a TX Remote Request Frame (MB RTR must be 1).
FLEXCAN_TX_TANSWER	MB is a TX Response Request Frame from.
FLEXCAN_TX_NOT_USED	an incoming Remote Request Frame. Not used

Definition at line 290 of file FlexCAN_Ip_HwAccess.h.

6.2.5.3 flexcan_int_type_t

enum flexcan_int_type_t

FlexCAN error interrupt types.

Enumerator

FLEXCAN_INT_RX_WARNING	RX warning interrupt
FLEXCAN_INT_TX_WARNING	TX warning interrupt
FLEXCAN_INT_ERR	Error interrupt
FLEXCAN_INT_ERR_FAST	Error Fast interrupt
154 FLEXCAN_INT_BESOFF	Bus off interrupt

Definition at line 303 of file FlexCAN_Ip_HwAccess.h.

6.2.5.4 Flexcan_Ip_RxFifoTransferType

enum `Flexcan_Ip_RxFifoTransferType`

The type of the RxFIFO transfer (interrupts/DMA).

Enumerator

<code>FLEXCAN_RXFIFO_USING_INTERRUPTS</code>	Use interrupts for RxFIFO.
<code>FLEXCAN_RXFIFO_USING_POLLING</code>	Use polling method for RxFIFO

Definition at line 107 of file FlexCAN_Ip_Types.h.

6.2.5.5 Flexcan_Ip_RxFifoIdFilterNumType

enum `Flexcan_Ip_RxFifoIdFilterNumType`

FlexCAN Rx FIFO filters number.

Enumerator

<code>FLEXCAN_RX_FIFO_ID_FILTERS_8</code>	8 Rx FIFO Filters.
<code>FLEXCAN_RX_FIFO_ID_FILTERS_16</code>	16 Rx FIFO Filters.
<code>FLEXCAN_RX_FIFO_ID_FILTERS_24</code>	24 Rx FIFO Filters.
<code>FLEXCAN_RX_FIFO_ID_FILTERS_32</code>	32 Rx FIFO Filters.
<code>FLEXCAN_RX_FIFO_ID_FILTERS_40</code>	40 Rx FIFO Filters.
<code>FLEXCAN_RX_FIFO_ID_FILTERS_48</code>	48 Rx FIFO Filters.
<code>FLEXCAN_RX_FIFO_ID_FILTERS_56</code>	56 Rx FIFO Filters.
<code>FLEXCAN_RX_FIFO_ID_FILTERS_64</code>	64 Rx FIFO Filters.
<code>FLEXCAN_RX_FIFO_ID_FILTERS_72</code>	72 Rx FIFO Filters.
<code>FLEXCAN_RX_FIFO_ID_FILTERS_80</code>	80 Rx FIFO Filters.
<code>FLEXCAN_RX_FIFO_ID_FILTERS_88</code>	88 Rx FIFO Filters.
<code>FLEXCAN_RX_FIFO_ID_FILTERS_96</code>	96 Rx FIFO Filters.
<code>FLEXCAN_RX_FIFO_ID_FILTERS_104</code>	104 Rx FIFO Filters.
<code>FLEXCAN_RX_FIFO_ID_FILTERS_112</code>	112 Rx FIFO Filters.
<code>FLEXCAN_RX_FIFO_ID_FILTERS_120</code>	120 Rx FIFO Filters.
<code>FLEXCAN_RX_FIFO_ID_FILTERS_128</code>	128 Rx FIFO Filters.

Definition at line 119 of file FlexCAN_Ip_Types.h.

6.2.5.6 Flexcan_Ip_RxMaskType

enum `Flexcan_Ip_RxMaskType`

FlexCAN Rx mask type.

Enumerator

FLEXCAN_RX_MASK_GLOBAL	Rx global mask
FLEXCAN_RX_MASK_INDIVIDUAL	Rx individual mask

Definition at line 142 of file FlexCAN_Ip_Types.h.

6.2.5.7 Flexcan_Ip_FdPayloadSizeType

enum `Flexcan_Ip_FdPayloadSizeType`

FlexCAN payload sizes.

Enumerator

FLEXCAN_PAYLOAD_SIZE_8	FlexCAN message buffer payload size in bytes
FLEXCAN_PAYLOAD_SIZE_16	FlexCAN message buffer payload size in bytes
FLEXCAN_PAYLOAD_SIZE_32	FlexCAN message buffer payload size in bytes
FLEXCAN_PAYLOAD_SIZE_64	FlexCAN message buffer payload size in bytes

Definition at line 152 of file FlexCAN_Ip_Types.h.

6.2.5.8 Flexcan_Ip_ModesType

enum `Flexcan_Ip_ModesType`

FlexCAN operation modes.

Enumerator

FLEXCAN_NORMAL_MODE	Normal mode or user mode
FLEXCAN_LISTEN_ONLY_MODE	Listen-only mode
FLEXCAN_LOOPBACK_MODE	Loop-back mode

Definition at line 164 of file FlexCAN_Ip_Types.h.

6.2.5.9 Flexcan_Ip_MbStateType

enum `Flexcan_Ip_MbStateType`

The state of a given MB (idle/Rx busy/Tx busy).

Enumerator

FLEXCAN_MB_IDLE	The MB is not used by any transfer.
FLEXCAN_MB_RX_BUSY	The MB is used for a reception.
FLEXCAN_MB_TX_BUSY	The MB is used for a transmission.

Definition at line 213 of file FlexCAN_Ip_Types.h.

6.2.5.10 Flexcan_Ip_EventType

enum `Flexcan_Ip_EventType`

The type of the event which occurred when the callback was invoked.

Enumerator

FLEXCAN_EVENT_RX_COMPLETE	A frame was received in the configured Rx MB.
FLEXCAN_EVENT_RXFIFO_COMPLETE	A frame was received in the Rx FIFO.
FLEXCAN_EVENT_RXFIFO_WARNING	Rx FIFO is almost full (5 frames).
FLEXCAN_EVENT_RXFIFO_OVERFLOW	Rx FIFO is full (incoming message was lost).
FLEXCAN_EVENT_TX_COMPLETE	A frame was sent from the configured Tx MB.
FLEXCAN_EVENT_ERROR	Errors detected in CAN frames of any format (interrupt mode only)
FLEXCAN_EVENT_BUSOFF	FlexCAN module entered Bus Off state
FLEXCAN_EVENT_RX_WARNING	The Rx error counter transitioned from less than 96 to greater than or equal to 96 (interrupt mode only)
FLEXCAN_EVENT_TX_WARNING	The Tx error counter transitioned from less than 96 to greater than or equal to 96 (interrupt mode only)

Definition at line 226 of file FlexCAN_Ip_Types.h.

6.2.5.11 Flexcan_Ip_ErrorIntType

enum `Flexcan_Ip_ErrorIntType`

FlexCAN error interrupt types.

Enumerator

FLEXCAN_IP_INT_RX_WARNING	RX warning interrupt
FLEXCAN_IP_INT_TX_WARNING	TX warning interrupt
FLEXCAN_IP_INT_ERR	Error interrupt
FLEXCAN_IP_INT_ERR_FAST	Error Fast interrupt
FLEXCAN_IP_INT_BUSOFF	Bus off interrupt

Definition at line 260 of file FlexCAN_Ip_Types.h.

6.2.5.12 Flexcan_Ip_MsgBuffIdType

enum `Flexcan_Ip_MsgBuffIdType`

FlexCAN Message Buffer ID type.

FlexCAN Id Type, Standard or Extended

Enumerator

FLEXCAN_MSG_ID_STD	Standard ID
FLEXCAN_MSG_ID_EXT	Extended ID

Definition at line 273 of file FlexCAN_Ip_Types.h.

6.2.5.13 Flexcan_Ip_RxFifoIdElementFormatType

enum `Flexcan_Ip_RxFifoIdElementFormatType`

ID formats for Rx FIFO.

Legacy RxFIFO Id Format Types

Enumerator

FLEXCAN_RX_FIFO_ID_FORMAT_A	One full ID (standard and extended) per ID Filter Table element.
FLEXCAN_RX_FIFO_ID_FORMAT_B	Two full standard IDs or two partial 14-bit (standard and extended) IDs per ID Filter Table element.
FLEXCAN_RX_FIFO_ID_FORMAT_C	Four partial 8-bit Standard IDs per ID Filter Table element.
FLEXCAN_RX_FIFO_ID_FORMAT_D	All frames rejected.

Definition at line 283 of file FlexCAN_Ip_Types.h.

6.2.5.14 Flexcan_Ip_StatusType

```
enum Flexcan_Ip_StatusType
```

The status used and reported by FlexCAN Ip driver.

The FlexCAN specific error codes

Enumerator

FLEXCAN_STATUS_SUCCESS	Successfull Operation Completed
FLEXCAN_STATUS_ERROR	Error Operation Completed
FLEXCAN_STATUS_BUSY	Busy Operation Completed
FLEXCAN_STATUS_TIMEOUT	TimeOut Operation Completed
FLEXCAN_STATUS_BUFF_OUT_OF_RANGE	The specified MB index is out of the configurable range
FLEXCAN_STATUS_NO_TRANSFER_IN_PROGRESS	There is no transmission or reception in progress

Definition at line 296 of file FlexCAN_Ip_Types.h.

6.2.6 Function Reference

6.2.6.1 FlexCAN_Ip_Send()

```
Flexcan_Ip_StatusType FlexCAN_Ip_Send (
    uint8 instance,
    uint8 mb_idx,
    const Flexcan_Ip_DataInfoType * tx_info,
```

```
uint32 msg_id,
const uint8 * mb_data )
```

Sends a CAN frame using the specified message buffer.

This function configure parameters form [Flexcan_Ip_DataInfoType](#), ID and sends data as CAN frame using a message buffer.

Parameters

in	<i>instance</i>	A FlexCAN instance number
in	<i>mb_idx</i>	Index of the message buffer
in	<i>tx_info</i>	Data info
in	<i>msg_id</i>	ID of the message to transmit
in	<i>mb_data</i>	Data Bytes of the FlexCAN message.

Returns

FLEXCAN_STATUS_BUFF_OUT_OF_RANGE if the index of a message buffer is invalid;
FLEXCAN_STATUS_BUSY if the message buffer is used for other operation;
FLEXCAN_STATUS_SUCCESS if successfull.

6.2.6.2 FlexCAN_Ip_SendBlocking()

```
Flexcan_Ip_StatusType FlexCAN_Ip_SendBlocking (
    uint8 instance,
    uint8 mb_idx,
    const Flexcan_Ip_DataInfoType * tx_info,
    uint32 msg_id,
    const uint8 * mb_data,
    uint32 timeout_ms )
```

Sends a CAN frame using the specified message buffer, in a blocking manner.

This function sends a CAN frame using a configured message buffer. The function blocks until either the frame was sent, or the specified timeout expired.

Parameters

in	<i>instance</i>	A FlexCAN instance number
in	<i>mb_idx</i>	Index of the message buffer
in	<i>tx_info</i>	Data info
in	<i>msg_id</i>	ID of the message to transmit
in	<i>mb_data</i>	Data bytes of the FlexCAN message
in	<i>timeout_ms</i>	A timeout for the transfer in milliseconds.

Returns

FLEXCAN_STATUS_SUCCESS if successful;
 FLEXCAN_STATUS_TIMEOUT if the timeout is reached;
 FLEXCAN_STATUS_BUFF_OUT_OF_RANGE if the index of a message buffer is invalid;
 FLEXCAN_STATUS_BUSY if the message buffer is used for other operation.

6.2.6.3 FlexCAN_Ip_Receive()

```

Flexcan_Ip_StatusType FlexCAN_Ip_Receive (
    uint8 instance,
    uint8 mb_idx,
    Flexcan_Ip_MsgBuffType * data,
    boolean isPolling )
  
```

Receives a CAN frame using the specified message buffer.

This function receives a CAN frame using a configured message buffer. The function returns immediately.

Parameters

in	<i>instance</i>	A FlexCAN instance number
in	<i>mb_idx</i>	Index of the message buffer
out	<i>data</i>	The FlexCAN receive message buffer data.
in	<i>isPolling</i>	If the message will be send using pooling(true) or interrupt(false).

Returns

FLEXCAN_STATUS_SUCCESS if successfull operation;
 FLEXCAN_STATUS_BUFF_OUT_OF_RANGE if the index of a message buffer is invalid;
 FLEXCAN_STATUS_BUSY if the message buffer is used for other operation.

6.2.6.4 FlexCAN_Ip_RxFifo()

```

Flexcan_Ip_StatusType FlexCAN_Ip_RxFifo (
    uint8 instance,
    Flexcan_Ip_MsgBuffType * data )
  
```

Receives a CAN frame using the message FIFO.

This function receives a CAN frame using the Rx FIFO. The function returns immediately.

Parameters

in	<i>instance</i>	A FlexCAN instance number
out	<i>data</i>	The FlexCAN receive message buffer data.

Returns

FLEXCAN_STATUS_SUCCESS if successfull operation;
FLEXCAN_STATUS_ERROR if FiFO feature wasn't enable;
FLEXCAN_STATUS_BUSY if the message buffer is used by other operation.

6.2.6.5 FlexCAN_Ip_RxFifoBlocking()

```
Flexcan_Ip_StatusType FlexCAN_Ip_RxFifoBlocking (
    uint8 instance,
    Flexcan_Ip_MsgBuffType * data,
    uint32 timeout )
```

Receives a CAN frame using the message FIFO, in a blocking manner.

This function receives a CAN frame using the Rx FIFO or Enhanced Rx FIFO (if available and enabled). If using Enhanced Rx FIFO, the size of the data array will be considered the same as the configured FIFO watermark. The function blocks until either a frame was received, or the specified timeout expired. FlexCAN_Ip_RxFifoBlocking/↵ FlexCAN_Ip_RxFifo must not be called in callback invocation while FlexCAN_Ip_RxFifoBlocking is running to avoid unexpected behaviour.

Parameters

<i>instance</i>	A FlexCAN instance number
<i>data</i>	The FlexCAN receive message buffer data.
<i>timeout</i>	A timeout for the transfer in milliseconds.

Returns

FLEXCAN_STATUS_SUCCESS if successful; FLEXCAN_STATUS_BUSY if a resource is busy; FLEXCAN_STATUS_TIMEOUT if the timeout is reached; FLEXCAN_STATUS_ERROR if other error occurred

6.2.6.6 FlexCAN_Ip_ConfigRxMb()

```
Flexcan_Ip_StatusType FlexCAN_Ip_ConfigRxMb (
    uint8 instance,
```

```
uint8 mb_idx,
const Flexcan_Ip_DataInfoType * rx_info,
uint32 msg_id )
```

FlexCAN receive message buffer field configuration.

This function will config receive parameters form `Flexcan_Ip_DataInfoType` and the message Id, and can overwrite another MB status.

Parameters

in	<i>instance</i>	A FlexCAN instance number
in	<i>mb_idx</i>	Index of the message buffer
in	<i>rx_info</i>	Data info
in	<i>msg_id</i>	ID of the message to transmit

Returns

FLEXCAN_STATUS_SUCCESS if successful;
FLEXCAN_STATUS_BUFF_OUT_OF_RANGE if the index of a message buffer is invalid.

6.2.6.7 FlexCAN_Ip_MainFunctionRead()

```
void FlexCAN_Ip_MainFunctionRead (
    uint8 instance,
    uint8 mb_idx )
```

Check a receive event.

This will check if message is received and read the message buffer or RxFifo.

Parameters

in	<i>instance</i>	A FlexCAN instance number
in	<i>mb_idx</i>	Index of the message buffer

6.2.6.8 FlexCAN_Ip_MainFunctionWrite()

```
void FlexCAN_Ip_MainFunctionWrite (
    uint8 instance,
    uint8 mb_idx )
```


Check a Transmission event.

This function will check a specific MB have been sent of FlexCAN module and if was sent will reset the status of Mb and clear the status flag.

Parameters

in	<i>instance</i>	A FlexCAN instance number
in	<i>mb_idx</i>	message buffer number

6.2.6.9 FlexCAN_Ip_GetTransferStatus()

```
Flexcan_Ip_StatusType FlexCAN_Ip_GetTransferStatus (
    uint8 instance,
    uint8 mb_idx )
```

Returns whether the previous FlexCAN transfer has finished.

When performing an async transfer, call this function to ascertain the state of the current transfer: in progress (or busy) or complete (success).

Parameters

in	<i>instance</i>	The FlexCAN instance number.
in	<i>mb_idx</i>	The index of the message buffer.

Returns

FLEXCAN_STATUS_SUCCESS if successful; FLEXCAN_STATUS_BUSY if a resource is busy; FLEXCAN_STATUS_ERROR in case of a DMA error transfer;

6.2.6.10 FlexCAN_Ip_GetErrorStatus()

```
uint32 FlexCAN_Ip_GetErrorStatus (
    uint8 instance )
```

Get Error Status of FlexCAN.

This function will return the error status from ESR1 register. For exact mapping of errors please refer to RM(Reference Manual) on FLEXCAN ESR1 register description.

Parameters

in	<i>instance</i>	A FlexCAN instance number
----	-----------------	---------------------------

Returns

The errors flags stored by register ESR1

6.2.6.11 FlexCAN_Ip_GetControllerTxErrorCounter()

```
uint8 FlexCAN_Ip_GetControllerTxErrorCounter (
    uint8 instance )
```

Get Transmit error counter of FlexCAN.

This function will return the Transmit error counter for all errors detected in transmitted messages from ECR register. For exact mapping of errors please refer to RM(Reference Manual) on FLEXCAN ECR register description.

Parameters

in	<i>instance</i>	A FlexCAN instance number
----	-----------------	---------------------------

Returns

The Transmit error counter stored by TXERRCNT in register ECR

6.2.6.12 FlexCAN_Ip_GetControllerRxErrorCounter()

```
uint8 FlexCAN_Ip_GetControllerRxErrorCounter (
    uint8 instance )
```

Get Receive error counter of FlexCAN.

This function will return the Receive error counter for all errors detected in transmitted messages from ECR register. For exact mapping of errors please refer to RM(Reference Manual) on FLEXCAN ECR register description.

Parameters

in	<i>instance</i>	A FlexCAN instance number
----	-----------------	---------------------------

Returns

The Receive error counter stored by RXERRCNT in register ECR

6.2.6.13 FlexCAN_Ip_ClearErrorStatus()

```
void FlexCAN_Ip_ClearErrorStatus (
    uint8 instance,
    uint32 error )
```

Clear Error Status of FlexCAN.

This function will clear the error status from ESR1 register. For exact mapping of errors please refer to RM(← Reference Manual) on FLEXCAN ESR1 register description.

Parameters

in	<i>instance</i>	A FlexCAN instance number
in	<i>error</i>	errors flags to be cleared

6.2.6.14 FlexCAN_Ip_GetBitrate()

```
boolean FlexCAN_Ip_GetBitrate (
    uint8 instance,
    Flexcan_Ip_TimeSegmentType * bitrate )
```

Gets the FlexCAN bit rate for standard frames or the arbitration phase of FD frames.

Note

In case is used Enhanced Time Segments the PhaseSeg1 is the sum of PropSeg +1+ PhaseSeg1, and the PropSeg will be 0;

Parameters

in	<i>instance</i>	A FlexCAN instance number
out	<i>bitrate</i>	A pointer to a variable for returning the FlexCAN bit rate settings

Returns

true if Enhanced Time segments are used; false if Enhanced Time segments are not used.

6.2.6.15 FlexCAN_Ip_GetBuffStatusFlag()

```
boolean FlexCAN_Ip_GetBuffStatusFlag (
    uint8 instance,
    uint8 msgBuffIdx )
```

Get the Status of Message Buffer.

This function will return True if Message Buffer Flag is Set or False if is not set.

Parameters

in	<i>instance</i>	A FlexCAN instance number
in	<i>msgBuffIdx</i>	Index of the message buffer

Returns

True if is set False if is clear.

6.2.6.16 FlexCAN_Ip_ClearBuffStatusFlag()

```
void FlexCAN_Ip_ClearBuffStatusFlag (
    uint8 instance,
    uint8 msgBuffIdx )
```

Clear Message Buffer Status Flag.

This function will clear the status of the message buffer

Parameters

in	<i>instance</i>	A FlexCAN instance number
in	<i>msgBuffIdx</i>	Index of the message buffer

6.2.6.17 FlexCAN_Ip_AbortTransfer()

```
Flexcan_Ip_StatusType FlexCAN_Ip_AbortTransfer (
    uint8 u8Instance,
    uint8 mb_idx )
```

Ends a non-blocking FlexCAN transfer early.

Full description

Parameters

in	<i>u8Instance</i>	A FlexCAN instance number
in	<i>mb_idx</i>	The index of the message buffer

Returns

FLEXCAN_STATUS_SUCCESS if successful; FLEXCAN_STATUS_NO_TRANSFER_IN_PROGRESS if no transfer was running, FLEXCAN_STATUS_TIMEOUT if fail to configure in configured timeout value.

6.2.6.18 FlexCAN_Ip_GetListenOnlyMode()

```
boolean FlexCAN_Ip_GetListenOnlyMode (
    uint8 instance )
```

Get the Status of Listen Only Mode.

This function will return True if Listen Only Mode is Enable or False if is Disable.

Parameters

in	<i>instance</i>	A FlexCAN instance number
----	-----------------	---------------------------

Returns

True if Listen Only Mode is Enable False if Listen Only Mode is Disable.

6.2.6.19 FlexCAN_Ip_ReceiveBlocking()

```
Flexcan_Ip_StatusType FlexCAN_Ip_ReceiveBlocking (
    uint8 instance,
    uint8 mb_idx,
    Flexcan_Ip_MsgBuffType * data,
    boolean isPolling,
    uint32 u32TimeoutMs )
```

Receives a CAN frame using the specified message buffer, in a blocking manner.

This function receives a CAN frame using a configured message buffer. The function blocks until either a frame was received, or the specified timeout expired.

Parameters

in	<i>instance</i>	A FlexCAN instance number
in	<i>mb_idx</i>	Index of the message buffer
out	<i>data</i>	The FlexCAN receive message buffer data.
in	<i>isPolling</i>	If the message will be send using pooling(true) or interrupt(false).
in	<i>timeout_ms</i>	A timeout for the transfer in milliseconds.

Returns

FLEXCAN_STATUS_SUCCESS if successfull operation;
 FLEXCAN_STATUS_BUFF_OUT_OF_RANGE if the index of a message buffer is invalid;
 FLEXCAN_STATUS_BUSY if the message buffer is used for other operation.
 FLEXCAN_STATUS_TIMEOUT if the timeout is reached.

6.2.6.20 FlexCAN_Ip_ConfigRemoteResponseMb()

```

Flexcan_Ip_StatusType FlexCAN_Ip_ConfigRemoteResponseMb (
    uint8 instance,
    uint8 mb_idx,
    const Flexcan_Ip_DataInfoType * tx_info,
    uint32 msg_id,
    const uint8 * mb_data )
  
```

Configures a transmit message buffer for remote frame response.

@Note In case of using this function as polling mode the user should call FlexCAN_Ip_MainFunctionWrite to check it. @Note In case of enable the option Remote Request Store by setting corresponding bit for FLEXCAN_IP_REM_STORE_U32 in the ctrlOptions structure member of the Flexcan platform configuration data from FlexCAN_Ip_Init function, will disable Automatic Response Request feature, in this case is not allowed use of this function.

Parameters

in	<i>instance</i>	A FlexCAN instance number
in	<i>mb_idx</i>	Index of the message buffer
in	<i>tx_info</i>	Data info
in	<i>msg_id</i>	ID of the message to transmit
in	<i>mb_data</i>	Bytes of the FlexCAN message

Returns

FLEXCAN_STATUS_SUCCESS if successful; FLEXCAN_STATUS_BUFF_OUT_OF_RANGE if the index of the message buffer is invalid

6.2.6.21 FlexCAN_Ip_ManualBusOffRecovery()

```
Flexcan_Ip_StatusType FlexCAN_Ip_ManualBusOffRecovery (
    uint8 Instance )
```

Recover manually from bus-off if possible.

Note

This function should be used when bus-off auto recovery disabled and controller is in START mode. The function FlexCAN_Ip_GetErrorStatus can be used to check FLTCONF bits to check if bus-off state is exited or not.

Parameters

in	Instance	The FlexCAN instance number.
----	----------	------------------------------

Returns

- FLEXCAN_STATUS_SUCCESS if successful operation or the controller wasn't in bus-off.
- FLEXCAN_STATUS_ERROR if bus-off auto recovery enabled.
- FLEXCAN_STATUS_TIMEOUT if the timeout is reached.

6.2.6.22 FlexCAN_SetRxFifoFilter()

```
void FlexCAN_SetRxFifoFilter (
    FLEXCAN_Type * base,
    Flexcan_Ip_RxFifoIdElementFormatType idFormat,
    const Flexcan_Ip_IdTableType * idFilterTable )
```

Sets the FlexCAN Rx FIFO fields.

Parameters

base	The FlexCAN base address
idFormat	The format of the Rx FIFO ID Filter Table Elements
idFilterTable	The ID filter table elements which contain RTR bit, IDE bit, and RX message ID.

6.2.6.23 FlexCAN_ReadRxFifo()

```
void FlexCAN_ReadRxFifo (
    const FLEXCAN_Type * base,
    Flexcan_Ip_MsgBuffType * rxFifo )
```

Gets the FlexCAN Rx FIFO data.

Parameters

<i>base</i>	The FlexCAN base address
<i>rxFifo</i>	The FlexCAN receive FIFO data

6.2.6.24 FlexCAN_ExitFreezeMode()

```
Flexcan_Ip_StatusType FlexCAN_ExitFreezeMode (
    FLEXCAN_Type * base )
```

Un freezes the FlexCAN module.

Parameters

<i>base</i>	The FlexCAN base address
-------------	--------------------------

Returns

FLEXCAN_STATUS_SUCCESS successfully exit from freeze FLEXCAN_STATUS_TIMEOUT fail to exit from freeze

6.2.6.25 FlexCAN_LockRxMsgBuff()

```
void FlexCAN_LockRxMsgBuff (
    const FLEXCAN_Type * base,
    uint32 msgBuffIdx )
```

Locks the FlexCAN Rx message buffer.

Parameters

<i>base</i>	The FlexCAN base address
<i>msgBuffIdx</i>	Index of the message buffer

6.2.6.26 FlexCAN_SetMsgBuffIntCmd()

```
Flexcan_Ip_StatusType FlexCAN_SetMsgBuffIntCmd (
    FLEXCAN_Type * base,
    uint8 u8Instance,
    uint32 msgBuffIdx,
    boolean enable,
    boolean bIsIntActive )
```

Enables/Disables the FlexCAN Message Buffer interrupt.

Parameters

<i>base</i>	The FlexCAN base address
<i>msgBuffIdx</i>	Index of the message buffer
<i>enable</i>	choose enable or disable

Returns

FLEXCAN_STATUS_SUCCESS if successful; FLEXCAN_STATUS_CAN_BUFF_OUT_OF_RANGE if the index of the message buffer is invalid

6.2.6.27 FlexCAN_DisableInterrupts()

```
void FlexCAN_DisableInterrupts (
    FLEXCAN_Type * pBase )
```

Disable all interrupts.

Parameters

<i>pBase</i>	The FlexCAN base address
--------------	--------------------------

6.2.6.28 FlexCAN_EnableInterrupts()

```
void FlexCAN_EnableInterrupts (
    FLEXCAN_Type * pBase,
    uint8 u8Instance )
```

Enable all interrupts configured.

Parameters

<i>pBase</i>	The FlexCAN base address
<i>u8Instance</i>	A FlexCAN instance number

6.2.6.29 FlexCAN_SetTxMsgBuff()

```
void FlexCAN_SetTxMsgBuff (
    volatile uint32 *const pMbAddr,
    const Flexcan_Ip_MsbuffCodeStatusType * cs,
    uint32 msgId,
    const uint8 * msgData,
    const boolean isRemote )
```

Sets the FlexCAN message buffer fields for transmitting.

Parameters

<i>pMbAddr</i>	The Message buffer address
<i>cs</i>	CODE/status values (TX)
<i>msgId</i>	ID of the message to transmit
<i>msgData</i>	Bytes of the FlexCAN message
<i>isRemote</i>	Will set RTR remote Flag

Returns

FLEXCAN_STATUS_SUCCESS if successful; FLEXCAN_STATUS_CAN_BUFF_OUT_OF_RANGE if the index of the message buffer is invalid

6.2.6.30 FlexCAN_EnableRxFifo()

```
Flexcan_Ip_StatusType FlexCAN_EnableRxFifo (
    FLEXCAN_Type * base,
    uint32 numOfFilters )
```

Enables the Rx FIFO.

Parameters

<i>base</i>	The FlexCAN base address
<i>numOfFilters</i>	The number of Rx FIFO filters

Returns

The status of the operation

Return values

<i>FLEXCAN_STATUS_SUCCESS</i>	RxFIFO was successfully enabled
<i>FLEXCAN_STATUS_ERROR</i>	RxFIFO could not be enabled (e.g. the FD feature was enabled, and these two features are not compatible)

6.2.6.31 FlexCAN_SetMaxMsgBuffNum()

```
Flexcan_Ip_StatusType FlexCAN_SetMaxMsgBuffNum (
    FLEXCAN_Type * base,
    uint32 maxMsgBuffNum )
```

Sets the maximum number of Message Buffers.

Parameters

<i>base</i>	The FlexCAN base address
<i>maxMsgBuffNum</i>	Maximum number of message buffers

Returns

FLEXCAN_STATUS_SUCCESS if successful; FLEXCAN_STATUS_BUFF_OUT_OF_RANGE if the index of the message buffer is invalid

6.2.6.32 FlexCAN_SetRxMsgBuff()

```
void FlexCAN_SetRxMsgBuff (
    const FLEXCAN_Type * base,
    uint32 msgBuffIdx,
    const Flexcan_Ip_MsbuffCodeStatusType * cs,
    uint32 msgId )
```

Sets the FlexCAN message buffer fields for receiving.

Parameters

<i>base</i>	The FlexCAN base address
<i>msgBuffIdx</i>	Index of the message buffer
<i>cs</i>	CODE/status values (RX)
<i>msgId</i>	ID of the message to receive

Returns

FLEXCAN_STATUS_SUCCESS if successful; FLEXCAN_STATUS_BUFF_OUT_OF_RANGE if the index of the message buffer is invalid

6.2.6.33 FlexCAN_GetMsgBuffTimestamp()

```
uint32 FlexCAN_GetMsgBuffTimestamp (
    const FLEXCAN_Type * base,
    uint32 msgBuffIdx )
```

Gets the message buffer timestamp value.

Parameters

<i>base</i>	The FlexCAN base address
<i>msgBuffIdx</i>	Index of the message buffer

Returns

value of timestamp for selected message buffer.

6.2.6.34 FlexCAN_GetMsgBuff()

```
void FlexCAN_GetMsgBuff (
    const FLEXCAN_Type * base,
    uint32 msgBuffIdx,
    Flexcan_Ip_MsgBuffType * msgBuff )
```

Gets the FlexCAN message buffer fields.

Parameters

<i>base</i>	The FlexCAN base address
<i>msgBuffIdx</i>	Index of the message buffer
<i>msgBuff</i>	The fields of the message buffer

6.2.6.35 FlexCAN_GetMbPayloadSize()

```
uint8 FlexCAN_GetMbPayloadSize (
```

```
const FLEXCAN_Type * base,  
uint32 maxMsgBuffNum )
```

Gets the payload size of the MBs.

Parameters

<i>base</i>	The FlexCAN base address
-------------	--------------------------

Returns

The payload size in bytes

6.2.6.36 FlexCAN_Init()

```
Flexcan_Ip_StatusType FlexCAN_Init (  
    FLEXCAN_Type * base )
```

Initializes the FlexCAN controller.

Parameters

<i>base</i>	The FlexCAN base address
-------------	--------------------------

6.2.6.37 FlexCAN_GetMaxMbNum()

```
uint32 FlexCAN_GetMaxMbNum (  
    const FLEXCAN_Type * base )
```

Get The Max no of MBs allowed on CAN instance.

Parameters

<i>base</i>	The FlexCAN base address
-------------	--------------------------

Returns

The Max No of MBs on the CAN instance;

6.2.6.38 FlexCAN_SetOperationMode()

```
void FlexCAN_SetOperationMode (
    FLEXCAN_Type * base,
    Flexcan_Ip_ModesType mode )
```

Set operation mode.

Parameters

<i>base</i>	The FlexCAN base address
<i>mode</i>	Set an operation mode

6.2.6.39 FlexCAN_GetMsgBuffRegion()

```
volatile uint32* FlexCAN_GetMsgBuffRegion (
    const FLEXCAN_Type * base,
    uint32 msgBuffIdx )
```

Sets the FlexCAN message buffer fields for transmitting.

Parameters

<i>base</i>	The FlexCAN base address
<i>msgBuffIdx</i>	Index of the message buffer

Returns

Pointer to the beginning of the MBs space address

6.2.6.40 FlexCAN_ConfigCtrlOptions()

```
void FlexCAN_ConfigCtrlOptions (
    FLEXCAN_Type * pBase,
    uint32 u32Options )
```

configure controller depending on options.

Parameters

<i>pBase</i>	The FlexCAN base address.
<i>u32Options</i>	Controller Options.

6.2.6.41 FlexCAN_ResetImaskBuff()

```
void FlexCAN_ResetImaskBuff (
    uint8 Instance )
```

Reset Imask Buffers.

Parameters

<i>Instance</i>	The FlexCAN instance
-----------------	----------------------

6.3 Controller Area Network with Flexible Data Rate (FlexCAN)

The S32 RTD provides a Peripheral Driver for the FlexCAN module of S32 devices.

6.3.0.1 Hardware background

The FlexCAN module is a communication controller implementing the CAN protocol according to the ISO 11898-1 standard and CAN 2.0 B protocol specifications. The FlexCAN module is a full implementation of the CAN protocol specification, the CAN with Flexible Data rate (CAN FD) protocol and the CAN 2.0 version B protocol, which supports both standard and extended message frames and long payloads up to 64 bytes transferred at faster rates up to 8 Mbps. The message buffers are stored in an embedded RAM dedicated to the FlexCAN module.

The FlexCAN module includes these distinctive features:

- Full implementation of the CAN with Flexible Data Rate (CAN FD) protocol specification and CAN protocol specification, Version 2.0 B (see the `FEATURE_CAN_HAS_FD` define for the availability of this feature on each platform)
 - Standard data frames
 - Extended data frames
 - Zero to sixty four bytes data length
 - Programmable bit rate (see the chip-specific FlexCAN information for the specific maximum bit rate configuration)
 - Content-related addressing
- Compliant with the ISO 11898-1 standard
- Flexible mailboxes configurable to store 0 to 8, 16, 32 or 64 bytes data length (payloads longer than 8 bytes are available only for some platforms, see the `FEATURE_CAN_HAS_FD` define)
- Each mailbox configurable as receive or transmit, all supporting standard and extended messages
- Individual Rx Mask registers per mailbox
- Transmission abort capability
- Flexible message buffers (MBs) configurable as Rx or Tx (see the `FEATURE_CAN_MAX_MB_NUM` define for the specific maximum number of message buffers configurable on each platform) define for the availability of this feature on each platform)
- RAM not used by reception or transmission structures can be used as general purpose RAM space
- Listen-Only mode capability
- Programmable Loop-Back mode supporting self-test operation
- Maskable interrupts
- Short latency time due to an arbitration scheme for high-priority messages
- Transceiver Delay Compensation feature when transmitting CAN FD messages at faster data rates (see the `FEATURE_CAN_HAS_FD` define for the availability of this feature on each platform)

- Remote request frames may be handled automatically or by software
- CAN bit time settings and configuration bits can only be written in Freeze mode
- SYNCH bit available in Error in Status 1 register to inform that the module is synchronous with CAN bus
- CRC status for transmitted message
- Rx FIFO Global Mask register
- Selectable priority between mailboxes and Rx FIFO during matching process
- Powerful Rx FIFO ID filtering, capable of matching incoming IDs against either 128 extended, 256 standard, or 512 partial (8 bit) IDs, with up to 32 individual masking capability
- 100% backward compatibility with previous FlexCAN version
- Supports detection and correction of errors in memory read accesses. Errors in one bit can be corrected and errors in 2 bits can be detected but not corrected (this feature might not be available on some platforms, see chip-specific FlexCAN information for details)
- Disable Detection and Correction of Memory Errors Feature for devices that supports it. This feature can cause Freeze Mode of CAN interface. (see `FEATURE_CAN_HAS_MEM_ERR_DET` define availability of the feature in module)
- Identifier Acceptance Filter Hit Indicator (IDHIT) register for received frames in RxFIFO
- Time stamp based on 32-bit free running timer (see `FEATURE_CAN_HAS_HR_TIMER` define availability of the feature in module)

6.4 FlexCAN_driver

6.4.0.1 How to use the FlexCAN driver in your application

In order to be able to use the FlexCAN in your application, the first thing to do is initializing it with the desired configuration. This is done by calling the **FLEXCAN_DRV_Init** function. One of the arguments passed to this function is the configuration which will be used for the FlexCAN module, specified by the **Flexcan_Ip_ConfigType** structure.

The **Flexcan_Ip_ConfigType** structure allows you to configure the following:

- the number of message buffers needed;
- the number of Rx FIFO ID filters needed;
- enable/disable the Rx FIFO feature;
- the operation mode, which can be one of the following:
 - normal mode;
 - listen-only mode;
 - loopback mode;
 - freeze mode;
 - disable mode;
- Control Options like use of different features support like ISO-FD, EDGE_FILTER, AUTO_BussOffRecovery, Protocol_Exception.
- the payload size of the message buffers:
 - 8 bytes;
 - 16 bytes (only available with the FD feature enabled);
 - 32 bytes (only available with the FD feature enabled);
 - 64 bytes (only available with the FD feature enabled);
- enable/disable the Flexible Data-rate feature;
- The use of extended bit time segments format from CBT register, instead of CTRL1 register
- Enable of BitRate Switch support for FD frames
- the bitrate used for standard frames or for the arbitration phase of FD frames;
- the bitrate used for the data phase of FD frames;
- the Rx FIFO transfer type, which can be one of the following:
 - using interrupts;
 - using DMA, only on supported platforms;
- the DMA channel number to be used for DMA transfers, only on supported platforms;

- the number of words to transfer for each Enhanced data element, only available with the Enhanced and DMA feature enabled

The bitrate is represented by a **Flexcan_Ip_TimeSegmentType** structure, with the following fields:

- propagation segment;
- phase segment 1;
- phase segment 2;
- clock prescaler division factor;
- resync jump width.

Details about these fields can be found in the reference manual.

In order to use a mailbox for reception, it should be initialized using either **FLEXCAN_DRV_ConfigRxMb**, **FLEXCAN_DRV_ConfigRxFifo**.

After having the mailbox configured, you can start sending/receiving data using the specified mailbox, by calling one of the following functions:

- **FLEXCAN_Ip_Send**;
- **FLEXCAN_Ip_SendBlocking**;
- **FLEXCAN_Ip_Receive**;
- **FLEXCAN_Ip_RxFifo**;

6.4.0.1.1 FlexCAN Rx FIFO configuration The Rx FIFO is receive-only and 6-message deep. The user can read the received messages sequentially, in the order they were received, by repeatedly reading Message Buffer 0 (zero). The Rx FIFO ID filter table (configurable from 8 to 128 table elements) specifies filtering criteria for accepting frames into the FIFO. This table is represented through a structure of **Flexcan_Ip_IdTableType** type, which specifies if Remote Frames are accepted into the FIFO if they match the target ID, whether extended or standard frames are accepted into the FIFO if they match the target ID and the target ID.

```
/* ID Filter table */  
  
const Flexcan_Ip_IdTableType filterTable[] = {  
  
{  
  
.isExtendedFrame = false,  
.isRemoteFrame = false,  
.id = 1U  
},  
...  
};  
  
FlexCAN_Ip_ConfigRxFifo(INST_CANCOM1, FLEXCAN_RX_FIFO_ID_FORMAT_A, filterTable);
```

The number of elements in the ID filter table is defined by the following formula:

- for format A: the number of Rx FIFO ID filters
- for format B: twice the number of Rx FIFO ID filters
- for format C: four times the number of Rx FIFO ID filters The user must provide the exact number of elements in order to avoid any misconfiguration.

Each element in the ID filter table specifies an ID to be used as acceptance criteria for the FIFO, as follows:

- for format A: In the standard frame format, bits 10 to 0 of the ID are used for frame identification. In the extended frame format, bits 28 to 0 are used.
- for format B: In the standard frame format, bits 10 to 0 of the ID are used for frame identification. In the extended frame format, only the 14 most significant bits (28 to 15) of the ID are compared to the 14 most significant bits (28 to 15) of the received ID.
- for format C: In both standard and extended frame formats, only the 8 most significant bits (7 to 0 for standard, 28 to 21 for extended) of the ID are compared to the 8 most significant bits (7 to 0 for standard, 28 to 21 for extended) of the received ID.

6.4.0.2 Important Notes

In order to use driver in interrupt mode the user should enable and register the driver Interrupts through Interrupt Controller Module;

6.4.0.3 Integration guideline

6.4.0.3.1 Compilation units

The following files need to be compiled in the project:

```

${S32RTD_PATH}\Can_43_FLEXCAN_<asr_package_name>\src\FlexCAN_Ip.c
${S32RTD_PATH}\Can_43_FLEXCAN_<asr_package_name>\src\FlexCAN_Ip_HwAccess.c
${S32RTD_PATH}\Can_43_FLEXCAN_<asr_package_name>\src\FlexCAN_Ip_Irq.c

```

6.4.0.3.2 Include path

The following paths need to be added to the include path of the toolchain:

```

${S32RTD_PATH}\Can_43_FLEXCAN_<asr_package_name>\include\

```

6.4.0.3.3 Preprocessor symbols

No special symbols are required for this component

6.4.0.3.4 Dependencies

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.

NXP, the NXP logo, NXP SECURE CONNECTIONS FOR A SMARTER WORLD, COOLFLUX, EMBRACE, GREENCHIP, HITAG, I2C BUS, ICODE, JCOP, LIFE VIBES, MIFARE, MIFARE CLASSIC, MIFARE DESFire, MIFARE PLUS, MIFARE FLEX, MANTIS, MIFARE ULTRALIGHT, MIFARE4MOBILE, MIGLO, NTAG, ROADLINK, SMARTLX, SMARTMX, STARPLUG, TOPFET, TRENCHMOS, UCODE, Freescale, the Freescale logo, Altivec, C-5, CodeTEST, CodeWarrior, ColdFire, ColdFire+, C-Ware, the Energy Efficient Solutions logo, Kinetis, Layerscape, MagniV, mobileGT, PEG, PowerQUICC, Processor Expert, QorIQ, QorIQ Qonverge, Ready Play, SafeAssure, the SafeAssure logo, StarCore, Symphony, VortiQa, Vybrid, Airfast, BeeKit, BeeStack, CoreNet, Flexis, MXC, Platform in a Package, QUICC Engine, SMARTMOS, Tower, TurboLink, and UMEMS are trademarks of NXP B.V. All other product or service names are the property of their respective owners. ARM, AMBA, ARM Powered, Artisan, Cortex, Jazelle, Keil, SecurCore, Thumb, TrustZone, and Vision are registered trademarks of ARM Limited (or its subsidiaries) in the EU and/or elsewhere. ARM7, ARM9, ARM11, big.LITTLE, CoreLink, CoreSight, DesignStart, Mali, mbed, NEON, POP, Sensinode, Socrates, ULINK and Versatile are trademarks of ARM Limited (or its subsidiaries) in the EU and/or elsewhere. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. The Power Architecture and Power.org word marks and the Power and Power.org logos and related marks are trademarks and service marks licensed by Power.org.

© 2023 NXP B.V.

