Elektrobit

Can release notes and documentation

product release 8.8.3

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

Technical support

https://www.elektrobit.com/support

Legal disclaimer

Confidential information.

Table of Contents

Preface 5
1. Overview 5
1. Can release notes 6
1.1. New features 6
1.2. Migrating the Can module 6
1.3. Limitations and deviations 6
1.4. EB-specific enhancements 10
1.5. Change log 11
2. Can user's guide 15
2.1. Overview 15
2.2. Background information 15
2.3. Configuring CAN Driver and related modules 16
2.3.1. Configuring CAN Driver 16
2.3.2. Configuring the Os 17
2.3.3. Configuring the Rte 18
2.3.4. Configuring the Dem 18
2.4. Writing application software 18
2.5. Compiling CAN Driver 19
2.6. Running CAN Driver 20
3. Can references 21
3.1. Configuration parameters 21
3.1.1. CanConfigSet 22
3.1.2. CanController 22
3.1.3. CanHardwareEmulation 26
3.1.4. CanControllerBaudrateConfig 29
3.1.5. CanControllerFdBaudrateConfig 31
3.1.6. CanFilterMask 34
3.1.7. CanHardwareObject 34
3.1.8. CanGeneral 37
3.1.9. CanMainFunctionRWPeriods 43
3.1.10. CommonPublishedInformation 44
3.1.11. PublishedInformation 47
3.2. Application programming interface (API) 48
3.2.1. Type definitions 48
3.2.1.1. Can_ConfigType 48
3.2.1.2. Can_ControllerBaudrateConfigType 48
3.2.2. Macro constants 48
3.2.2.1. CAN_API_CBK_CHECK_WAKEUP 48
3.2.2.2. CAN_API_DISABLE_CONTROLLER_INTERRUPTS 49

3.2.2.3. CAN_API_ENABLE_CONTROLLER_INTERRUPTS 49
3.2.2.4. CAN_API_GET_VERSION_INFO 49
3.2.2.5. CAN_API_INIT 49
3.2.2.6. CAN_API_INIT_CONTROLLER 49
3.2.2.7. CAN_API_MAIN_FUNCTION_BUS_OFF 49
3.2.2.8. CAN_API_MAIN_FUNCTION_MODE 50
3.2.2.9. CAN_API_MAIN_FUNCTION_READ 50
3.2.2.10. CAN_API_MAIN_FUNCTION_WAKEUP 50
3.2.2.11. CAN_API_MAIN_FUNCTION_WRITE 50
3.2.2.12. CAN_API_SET_CONTROLLER_MODE 50
3.2.2.13. CAN_API_STORE_IN_RX_BUFFER 50
3.2.2.14. CAN_API_STORE_IN_TX_BUFFER 50
3.2.2.15. CAN_API_WRITE 51
3.2.2.16. CAN_E_DATALOST 51
3.2.2.17. CAN_E_PARAM_CONTROLLER 51
3.2.2.18. CAN_E_PARAM_DLC 51
3.2.2.19. CAN_E_PARAM_HANDLE 51
3.2.2.20. CAN_E_PARAM_POINTER 51
3.2.2.21. CAN_E_TRANSITION 52
3.2.2.22. CAN_E_UNINIT 52
3.2.3. Objects 52
3.2.3.1. CAN_CONFIG_SET 52
3.2.3.2. Can_ControllerConfigDummy 52
3.2.4. Functions 52
3.2.4.1. Can_CheckWakeup 52
3.2.4.2. Can_DisableControllerInterrupts 53
3.2.4.3. Can_EnableControllerInterrupts 53
3.2.4.4. Can_GetVersionInfo 54
3.2.4.5. Can_Init 54
3.2.4.6. Can_InitController 54
3.2.4.7. Can_MainFunction_BusOff 55
3.2.4.8. Can_MainFunction_Mode 55
3.2.4.9. Can_MainFunction_Read 55
3.2.4.10. Can_MainFunction_Read_Internal 55
3.2.4.11. Can_MainFunction_Wakeup 56
3.2.4.12. Can_MainFunction_Write 56
3.2.4.13. Can_MainFunction_Write_Internal 56
3.2.4.14. Can_SetControllerMode 56
3.2.4.15. Can_Write 57

Preface

1. Overview

Welcome to the CAN MCAL release notes and documentation. This document provides:

- Chapter 1, “Can release notes“, which include
  - Section 1.1, “New features”
  - Section 1.2, “Migrating the Can module”
  - Section 1.3, “Limitations and deviations ”
  - Section 1.4, “EB-specific enhancements”
  - Section 1.5, “Change log”
- Chapter 2, “Can user's guide“: concept information and configuration instructions
- Chapter 3, “Can references“: configuration parameters and the application programming

1. Can release notes

- AUTOSAR version and revision: 4.0.3
- AUTOSAR SWS version and revision: 4.0.0
- Module version: 3.1.4
- Supplier: Elektrobit Automotive GmbH

1.1. New features

- No new features have been added since the last release.

1.2. Migrating the Can module

This chapter describes how to migrate a module from one release to another.

If you want to migrate from one release to another, follow the installation instructions according to EB tresos installation guide.

- When migrating, you have to update the controller config and define the underlying CAN hardware from the drop-down list.
- If the controller Baud rate lists are empty, you have to define them by activating the CAN FD switch and you may disable CAN FD if it not used afterwards.

1.3. Limitations and deviations

This chapter lists the limitations of the module and its deviations from the AUTOSAR standard.

- Sleep / wakeup handling is not supported

  Affected AUTOSAR releases:

  - R4.0.0 rev 3

  Description:

  Sleep / wakeup handling is not supported.

  Rationale:

Sleep / wakeup handling is not supported by underlying hardware.

Requirements:

CAN257, CAN265, CAN266, CAN270, CAN271, CAN269, CAN364 CAN048, CAN294, CAN112, CAN319_Conf

- BusOff detection is not supported

Affected AUTOSAR releases:

- R4.0.0 rev 3

Description:

BusOff detection is not supported.

Rationale:

BusOff detection is not supported by underlying hardware.

Requirements:

CAN020, CAN272, CAN273, CAN274, CAN109

- There is no underlying CAN hardware and no asynchronous transmission

Affected AUTOSAR releases:

- R4.0.0 rev 3

Description:

The WinCore CAN driver does not access a particular hardware. All communication is performed via a virtual bus connection controlled by the underlying CIF library. Transmission of data is initiated immediately.

Requirements:

CAN237, CAN236, CAN238, CAN239, CAN240, CAN242, CAN244, CAN280, CAN419, CAN420, CAN077, CAN284, CAN246, CAN245, CAN404, CAN053, CAN407, CAN255, CAN021, CAN291, CAN277, CAN401, CAN402, CAN403, CAN278, CAN286, CAN011, CAN300, CAN384, CAN196, CAN197, CAN434, CAN022, CAN024, CAN408, CAN385, CAN422, CAN069_Conf

- Timing parameters will have no effect

Affected AUTOSAR releases:

- R4.0.0 rev 3

Description:

Timing parameters will have no effect.

Rationale:

The design of the EB tresos Inspector interface is not asynchronous. Functions will either return immediately or block indefinitely. Furthermore, exact timing behaviour is not guaranteed due to the underlying Windows host system.

Requirements:

CAN281, CAN398, CAN262, CAN264, CAN275, CAN370, CAN371

- MainFunctions do not call DET if module is not initialized

Affected AUTOSAR releases:

- R4.0.0 rev 3

Description:

MainFunctions do not call DET if module is not initialized.

Rationale:

MainFunctions may be called by SchM before Can has been initialized.

Requirements:

CAN179, CAN181, CAN184, CAN186, CAN379

- Post-Build configuration is not supported

Affected AUTOSAR releases:

- R4.0.0 rev 3

Description:

Post-Build configuration is not supported.

Rationale:

Design decision.

Requirements:

CAN078, CAN056, CAN023, CAN221

- Can_GetVersionInfo is a function

Affected AUTOSAR releases:

- R4.0.0 rev 3

Description:

Can_GetVersionInfo is a function.

Rationale:

It is unknown whether the caller of the function will be available as source code.

Requirements:

CAN251

- Can_MainFunctionRead, Can_MainFunctionWrite and Can_MainFunction_Wakeup are always implemented as functions.

Affected AUTOSAR releases:

- R4.0.0 rev 3

Description:

Can_MainFunctionRead, Can_MainFunctionWrite and Can_MainFunction_Wakeup are always implemented as functions.

Rationale:

Design decision.

Requirements:

CAN178, CAN180, CAN85

- Requirements on the environment are not implemented

Affected AUTOSAR releases:

- R4.0.0 rev 3

Description:

Requirements on the environment are not implemented.

Rationale:

These are no requirements on the CAN driver but on its environment.

Requirements:

CAN256, CAN222, CAN444

- Hardware cancellation of pending Tx frames is not supported

  Affected AUTOSAR releases:

  - R4.0.0 rev 3

  Description:

  Hardware cancellation of pending Tx frames is not supported.

  Rationale:

  Hardware cancellation of pending Tx frames is not implemented yet.

  Requirements:

  CAN287, CAN432, CAN285, CAN281, CAN433, CAN399, CAN400, CAN288, CAN069_Conf, CAN235

- No support for debug and trace

  Affected AUTOSAR releases:

  - R4.0.0 rev 3

  Description:

  Debug and Trace is not supported.

  Rationale:

  The module no longer provide DebugNTrace capabilities.

  Requirements:

  EBREQ_CAN_DebugNTrace0001

1.4. EB-specific enhancements

This chapter list the enhancements provided by the module.

- No enhancements in this release.

1.5. Change log

This chapter lists the changes between different versions.

Module version 3.1.4

2021-06-25

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

Module version 3.1.3

2021-03-05

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

Module version 3.1.2

2020-10-23

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

Module version 3.1.1

2020-06-19

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

Module version 3.1.0

2018-07-11

- Fixed configuration variant
- Use common Can_ControllerStateType from Base plugin
- Use < and > for #include directives
- Include Can_MemMap.h instead of MemMap.h

Module version 3.0.1

2016-11-04

- Adapted resource file for the scheduling of main functions to the split of `IpduM_MainFunction()` into `IpduM_MainFunctionRx()` and `IpduM_MainFunctionTx()`.
- EBAMCALWINCORE-150: Switchted to CIF framework

Module version 2.2.2

2014-03-21

- EBAMCALWINCORE-136: Fixed VSMD violations.

Module version 2.2.1

2014-02-12

- EBAMCALWINCORE-108: Adapted ISR symbols to Os 5.1 needs.

Module version 2.2.0

2013-10-14

- EBAMCALWINCORE-102, EBAMCALWINCORE-118: Cleaned up configuration schema.
- EBAMCALWINCORE-116: Removed MISRA violations.
- EBAMCALWINCORE-57: Removed compiler warning due to "always false" condition.
- EBAMCALWINCORE-133: Updated migration notes.

Module version 2.1.4

2013-06-21

- EBAMCALWINCORE-104: Added support for Debug and Trace
- EBAMCALWINCORE-106: Added support for MemMap generator
- EBAMCALWINCORE-110: Improved robustness of init state handling.
- EBAMCALWINCORE-111: Unified syntax of include directives

- EBAMCALWINCORE-105: Moved optional configuration parameters from list representation to "optional" parameter representation.

Module version 2.1.3

2012-10-18

- Enhance BSWMDs with explicit references from BSW-CALLED-ENTITYs or BSW-SCHEDULABLE-ENTITYs to EXCLUSIVE-AREAS
- Corrected EB tresos Studio online help.
- Updated deviations list.
- Removed declarations of MainFunctions.
- EBAMCALWINCORE-92 Fixed known issue: CAN passes the wrong controller mode to CanIf_ControllerModeIndication().

Module version 2.1.2

2012-06-15

- Corrected module Id.
- Added defines for exported symbols according to new naming scheme.

Module version 2.1.1

2012-04-23

- Removed DET call from MainFunctions.

Module version 2.1.0

2012-03-27

- Added BSWMD support.

Module version 2.0.1

2011-10-06

- Corrected API Ids and documentation.
- Updated MainFunction default periods.

Module version 2.0.0

2011-09-14

- Initial AUTOSAR 4.0 version

2. Can user's guide

2.1. Overview

This users guide provides information which is specific to EB tresos WinCore `CAN Driver` and its successful interaction with CIF. After following the instructions given in this document, you will be able to use EB tresos WinCore `CAN Driver` to send CAN frames to and receive them from a real or virtual CAN Bus. EB tresos WinCore `CAN Driver` behaves as a standard AUTOSAR CAN Driver and acts as MCAL for the upper layer modules of the CAN communication stack. It thus adds full CAN capabilities to EB tresos WinCore.

2.2. Background information

EB tresos WinCore `CAN Driver` communicates with CIF which is a generic interface to different USB CAN-Bus Interfaces over which the CAN frames sent or received by `CAN Driver` are transmitted. The `CAN Driver` module uses a special library, called CIF, to access the DLLs provided by the Hardware vendors. Frames are either transmitted on a real CAN Bus or on a virtual CAN bus inside the driver DLL of the vendor hardware.

Currently CIF supports access to PCAN, EB2100 and all devices connected to the Vector XL Driver Library.

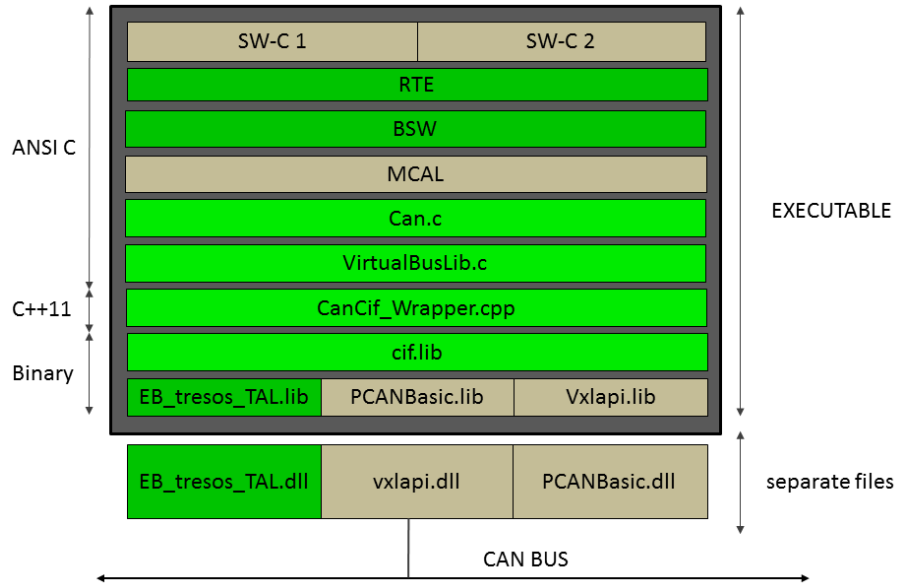

Figure 2.1. Schematic representation of the interaction between EB tresos WinCore CAN Driver and CIF

From `CAN Driver`'s point of view, the actual CAN hardware is abstracted by CIF. Hence, `CAN Driver` may only detect and process errors which are reported by the lower layer. `CAN Driver` Bus Off problems and

transmission problems, which are reported from the hardware appear in the `CAN Driver` if the hardware supports them.

NOTE

If `CAN Driver` signals a successful transmission, this signals the transmit confirmation received from the underlying Bus hardware.
Note that Bus Off events are *not* correctly handled by some devices.

Likewise, several other CAN features, which are related to physical or electrical properties of a CAN bus, are not available to EB tresos WinCore `CAN Driver`. The following features are not available:

- sleep- and wakeup transitions
- detection and handling of bus-off events
- hardware cancellation of pending Tx frames

All configuration parameters which refer to such unavailable features are deactivated in the EB tresos WinCore `CAN Driver` configuration scheme. All API functions concerning these features are implemented but do not provide any functionality (apart from development error tracing).

2.3. Configuring CAN Driver and related modules

2.3.1. Configuring CAN Driver

The following parameters control the communication between EB tresos WinCore `CAN Driver` and CIF:

Parameter name	Purpose of the parameter	How to configure
`CanHardware`	The type of the bus hardwware running under CIF.	Choose the hardware device which is connected to your PC. Ensure you installed all drivers necessary and you tested it seperately.
`CanHardwareChannel`	The CAN channel used by the CIF. The default channel is 0.	Some devices support multiple CAN channels which start with either 0 or 1. Note that 0 represents the first channel and CIF converts this value to the hardware value for you.
`BitRate`	This ia a list of standard Bit rates used on CAN.	The recommended default is 500k Baud.
`CanFDSupport`	This parameter defines wether the CAN FD interfaces are used.	Chose freely if you need CAN FD Support.

Parameter name	Purpose of the parameter	How to configure
	This switch decides whether the BitRate (see above) is used (non-CAN_FD) or if the CanController-BaudRateConfig has to be defined (see below).	
`CanControllerBaudrateConfig`	This ia a list of CAN Timing Parameters. See the CAN Standard and hardware documentation for details.	The recommended values are set by default
`CanControllerFdBaudrateConfig`	These are timing paremeters for FD timing. The PCAN hardware needs an additional frequncy paramter (see hardware documentation).	The recommended values are set by default.

Table 2.1. Parameters that control the communication

The following parameters control the hardware emulation of EB tresos WinCore `CAN Driver`:

Parameter name	Purpose of the parameter	How to configure
`CanBufferSize`	The size of the buffer (in number of CAN frames) associated with each HRH or HTH (each HRH or HTH has its own, independent buffer).	Number of CAN frames stored in the ring buffer. The default value is 8. If the buffer overflows the strategy configured in `CanBufferMode` is applied to discard messages.
`CanBufferMode`	The behavior of `CAN Driver` if the buffer of a HRH is full: ► `KEEP_OLDEST`: Newly arriving frames are discarded. ► `KEEP_NEWEST`: Newly arriving frames overwrite the oldest frames in the buffer.	the default value is KEEP_OLDEST in order to stop receiving new frames if the receive buffer is full.

Table 2.2. Parameters that control the hardware emulation

2.3.2. Configuring the Os

You need the following settings to configure EB tresos WinCore Os:

- If the Rx policy of `CAN Driver` (configuration parameter `CanRxProcessing`) is `INTERRUPT`, configure an interrupt handler called `Can_Rx_Interrupt`.
- If the Tx policy of `CAN Driver` (configuration parameter `CanTxProcessing`) is `INTERRUPT`, configure an interrupt handler called `Can_Tx_Interrupt`.
- EB tresos WinCore `CAN Driver` needs considerably more stack space than other embedded `CAN Drivers`. Choose an appropriately high value for the Os task which is supposed to run `CAN Driver` (configuration parameter `OsStacksize`).

NOTE

Configuration values may vary, refer to EB tresos AutoCore OS documentation

It depends on your overall setup which values to use in detail for these configuration items (e.g.. interrupt category or priority, stack size etc.). Please refer to the EB tresos AutoCore OS documentation for further information on these parameters.

2.3.3. Configuring the Rte

In order to provide the exclusive area needed by EB tresos WinCore `CAN Driver`, use the `generate_swcd` option in tresos to create an `arxml` file which contains the exclusive areas needed by EB tresos WinCore. You can then import the generated file via `System Description importer` to make these areas visible in the RTE.

- `SchMModuleLiteral`: `Can`
- `SchMUseInstanceId`: `true`

Within this container, add a new instance including an exclusive area with the following properties:

- `SchMInstanceLiteral`: `SCHM_CAN_INSTANCE`
- `SchMExclusiveAreaLiteral`: `SCHM_CAN_EXCLUSIVE_AREA`
- `SchMExclusiveAreaType`: `GLOBAL_IRQ_LOCK`

2.3.4. Configuring the Dem

EB tresos WinCore `CAN Driver` reports certain run-time errors to the `Diagnostic Event Manager` (`Dem`). Thus, add a new Dem event called `CAN_E_TIMEOUT` to the Dem configuration.

2.4. Writing application software

Take care of the following three points when you write software that uses EB tresos WinCore CAN Driver:

1. The CAN Driver implementation maps the name of the CanConfigSet configuration container to a dummy constant of type Can_ConfigType.

NOTE **Do not pass a NULL pointer to Can_Init()**

Do *not* pass a NULL pointer to Can_Init()! Although post-build configuration is not supported, nevertheless a non-NULL pointer must be passed to Can_Init(). Using a NULL pointer is prohibited by AUTOSAR and will result in a CAN_E_PARAM_POINTER error reported to the Development Error Tracer (Det) if development error detection is enabled.

   - You may use the name of the CanConfigSet configuration container as parameter for the Can_Init() function.
   - Alternatively, you may use an own dummy constant of this type.

2. The CAN Driver implementation maps the controller ID that is specified in the CanControllerId configuration parameter to the name CanConf_CanController_[CanController] where [CanController] is the name of the CanController configuration container (see AUTOSAR Bugzilla entry #51555). You may use this name in all API functions which expect a controller ID as function parameter (e.g.. Can_SetControllerMode()).

   Unless the preprocessor define CAN_DONT_PROVIDE_LEGACY_SYMBOLIC_NAMES is set at compile time, the CAN Driver also exports the equivalent symbols Can_[CanController] and [CanController] to retain compatibility with former versions of the driver.

3. Can_InitController() expects a pointer to a controller configuration as parameter.

NOTE **Do not pass a NULL pointer to Can_InitController()**

Do *not* pass a NULL pointer to Can_InitController()! Although post-build configuration is not supported and only one CAN controller is emulated, nevertheless a non-NULL pointer must be passed to Can_InitController(). Using a NULL pointer is prohibited by AUTOSAR and will result in a CAN_E_PARAM_POINTER error reported to the Development Error Tracer (Det) if development error detection is enabled.

   - You may use a pointer to the Can_ControllerConfigDummy dummy configuration structure as parameter for the Can_InitController() function.
   - Alternatively, you may use an own dummy constant of this type.

2.5. Compiling CAN Driver

EB tresos WinCore CAN Driver needs to access some MS Windows API functions.

- The EB tresos WinCore has been release for the compiler defined in the Quality Statement. exclusively. The compiler and all its sources are distributed along with the Platforms plugin under the GPL license (see license documentation for further information).
- Set the environment variable `PATH` to the folder containing gcc.exe

2.6. Running CAN Driver

To successfully run EB tresos WinCore `CAN Driver`, follow this advice:

1. Ensure the driver DLLs are either stored in the same folder as the executable or in a folder registered in the `PATH` variable.
   - `EB_tresos_TAL.dll`
   - `PCANBasic.dll`
   - `vxlapi.dll`
2. Ensure the compiler directory which contains the gcc.exe is stored in the `PATH` variable.
3. If the `CAN Driver` crashes without an error message start `gdb` with the exectuable name as first parameter on the commandline. Then type `run` to determine the instruction causing the crash.

3. Can references

3.1. Configuration parameters

Containers included		
Container name	**Multiplicity**	**Description**
CanConfigSet	1..n	This is the multiple configuration set container for CAN Driver
CanGeneral	1..1	This container contains the parameters related each CAN Driver Unit.
CommonPublishedInformation	1..1	**Label:** Common Published Information Common container, aggregated by all modules. It contains published information about vendor and versions.
PublishedInformation	1..1	**Label:** EB Published Information Additional published parameters not covered by CommonPublishedInformation container.

Parameters included	
Parameter name	**Multiplicity**
IMPLEMENTATION_CONFIG_VARIANT	1..1

Parameter Name	**IMPLEMENTATION_CONFIG_VARIANT**	
Label	Config Variant	
Multiplicity	1..1	
Type	ENUMERATION	
Default value	VariantPreCompile	
Range	VariantPostBuild	
	VariantPreCompile	
Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile

3.1.1. CanConfigSet

Containers included		
Container name	**Multiplicity**	**Description**
CanController	1..n	This container contains the configuration parameters of the CAN controller(s).
CanHardwareObject	1..n	This container contains the configuration (parameters) of CAN Hardware Objects.

3.1.2. CanController

Containers included		
Container name	**Multiplicity**	**Description**
CanHardwareEmulation	1..1	This container contains the parameters related to the CAN hardware emulation using the EB tresos Inspector environment.
CanControllerBaudrateConfig	1..1	This container contains bit timing related configuration parameters of the CAN controller(s).
CanControllerFdBaudrate-Config	1..1	This container contains bit timing related configuration parameters of the CAN controller(s) for FD frame.
CanFilterMask	0..n	This container contains the configuration (parameters) of the CAN Filter Mask(s).

Parameters included	
Parameter name	**Multiplicity**
CanBusoffProcessing	1..1
CanControllerActivation	1..1
CanControllerBaseAddress	1..1
CanControllerId	1..1
CanRxProcessing	1..1
CanTxProcessing	1..1
CanWakeupProcessing	1..1
CanWakeupSupport	1..1
CanControllerDefaultBaudrate	1..1

Parameters included	
CanCpuClockRef	1..1
CanWakeupSourceRef	0..1
CanFDSupport	1..1

Parameter Name	CanBusoffProcessing	
Description	Enables / disables API Can_MainFunction_BusOff() for handling busoff events in polling mode.	
Multiplicity	1..1	
Type	ENUMERATION	
Default value	INTERRUPT	
Range	INTERRUPT	
	POLLING	
Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	CanControllerActivation	
Description	Defines if a CAN controller is used in the configuration.	
Multiplicity	1..1	
Type	BOOLEAN	
Default value	true	
Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	CanControllerBaseAddress
Description	Specifies the CAN controller base address.
Multiplicity	1..1
Type	INTEGER
Default value	0
Range	<=4294967295
	>=0

Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CanControllerId**	
Description	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.	
Multiplicity	1..1	
Type	INTEGER	
Default value	0	
Range	<=255	
	>=0	
Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CanRxProcessing**	
Description	Enables / disables API Can_MainFunction_Read() for handling PDU reception events in polling mode.	
Multiplicity	1..1	
Type	ENUMERATION	
Default value	INTERRUPT	
Range	INTERRUPT	
	POLLING	
Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CanTxProcessing**
Description	Enables / disables API Can_MainFunction_Write() for handling PDU transmission events in polling mode.
Multiplicity	1..1
Type	ENUMERATION
Default value	INTERRUPT

Range	INTERRUPT	
	POLLING	
Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CanWakeupProcessing**	
Description	Enables / disables API Can_MainFunction_Wakeup() for handling wakeup events in polling mode.	
Multiplicity	1..1	
Type	ENUMERATION	
Default value	INTERRUPT	
Range	INTERRUPT	
	POLLING	
Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CanWakeupSupport**	
Description	CAN driver support for wakeup over CAN Bus.	
Multiplicity	1..1	
Type	BOOLEAN	
Default value	true	
Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CanControllerDefaultBaudrate**	
Description	Reference to baudrate configuration container configured for the Can Controller.	
Multiplicity	1..1	
Type	REFERENCE	
Configuration class	**VariantPreCompile:**	VariantPreCompile
	VariantPostBuild:	VariantPostBuild

Origin	AUTOSAR_ECUC

Parameter Name	**CanCpuClockRef**	
Description	Reference to the CPU clock configuration, which is set in the MCU driver configuration	
Multiplicity	1..1	
Type	REFERENCE	
Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CanWakeupSourceRef**	
Description	This parameter contains a reference to the Wakeup Source for this controller as defined in the ECU State Manager.	
Multiplicity	0..1	
Type	SYMBOLIC-NAME-REFERENCE	
Configuration class	**PreCompile:**	VariantPostBuild
	PreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CanFDSupport**	
Description	CAN driver support for FD.	
Multiplicity	1..1	
Type	BOOLEAN	
Default value	false	
Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

3.1.3. CanHardwareEmulation

Parameters included	
Parameter name	**Multiplicity**
CanHardware	1..1

Parameters included	
CanHardwareChannel	1..1
BitRate	1..1
CanBufferSize	1..1

Parameter Name	CanHardware	
Description	Specifies the underlying USB Interface	
Multiplicity	1..1	
Type	ENUMERATION	
Default value	CIF_HWTYPE_VIRTUAL	
Range	CIF_HWTYPE_VIRTUAL	
	CIF_HWTYPE_CANCARDX	
	CIF_HWTYPE_CANAC2PCI	
	CIF_HWTYPE_CANCARDXL	
	CIF_HWTYPE_CANCASEXL	
	CIF_HWTYPE_CANBOARDXL	
	CIF_HWTYPE_VN2600	
	CIF_HWTYPE_VN2610	
	CIF_HWTYPE_VN3300	
	CIF_HWTYPE_VN3600	
	CIF_HWTYPE_VN7600	
	CIF_HWTYPE_VN8900	
	CIF_HWTYPE_VN2640	
	CIF_HWTYPE_VN1610	
	CIF_HWTYPE_VN1630	
	CIF_HWTYPE_VN1640	
	CIF_HWTYPE_VN1611	
	CIF_PCAN_CT_USB	
Configuration class	**VariantPreCompile:**	VariantPreCompile
	VariantPostBuild:	VariantPostBuild
Origin	Elektrobit Automotive Software	

Parameter Name	CanHardwareChannel

Description	Specifies the CAN channel on the hardware interface.	
Multiplicity	1..1	
Type	ENUMERATION	
Default value	CIF_HW_INDEX_CAN0	
Range	CIF_HW_INDEX_CAN0	
	CIF_HW_INDEX_CAN1	
	CIF_HW_INDEX_CAN2	
	CIF_HW_INDEX_CAN3	
Configuration class	**VariantPreCompile:**	VariantPreCompile
	VariantPostBuild:	VariantPostBuild
Origin	Elektrobit Automotive Software	

Parameter Name	**BitRate**	
Description	Specifies the controller bit rate.	
Multiplicity	1..1	
Type	ENUMERATION	
Default value	CIF_BITRATE_500K	
Range	CIF_BITRATE_5K	
	CIF_BITRATE_10K	
	CIF_BITRATE_15K	
	CIF_BITRATE_20K	
	CIF_BITRATE_50K	
	CIF_BITRATE_100K	
	CIF_BITRATE_125K	
	CIF_BITRATE_250K	
	CIF_BITRATE_500K	
	CIF_BITRATE_800K	
	CIF_BITRATE_1M	
Configuration class	**PreCompile:**	VariantPreCompile
	PreCompile:	VariantPostBuild
Origin	Elektrobit Automotive Software	

Parameter Name	**CanBufferSize**

Description	Specifies the size of the receive and transmit buffers (number of frames per buffer).	
Multiplicity	1..1	
Type	INTEGER	
Default value	8	
Range	<=255 >0	
Configuration class	**VariantPreCompile:**	VariantPreCompile
	VariantPostBuild:	VariantPostBuild
Origin	Elektrobit Automotive Software	

3.1.4. CanControllerBaudrateConfig

Parameters included	
Parameter name	**Multiplicity**
CanControllerBaudRate	1..1
CanControllerPropSeg	1..1
CanControllerSeg1	1..1
CanControllerSeg2	1..1
CanControllerSyncJumpWidth	1..1

Parameter Name	**CanControllerBaudRate**	
Description	Specifies the buadrate of the controller in kbps.	
Multiplicity	1..1	
Type	INTEGER	
Default value	0	
Range	<=1000000 >=1	
Configuration class	**VariantPostBuild:**	VariantPostBuild
Origin	AUTOSAR_ECUC	

Parameter Name	**CanControllerPropSeg**
Description	Specifies propagation delay in time quantas. NOTE: this value is not used

Multiplicity	1..1
Type	INTEGER
Default value	0
Range	<=15 >=0
Configuration class	**VariantPostBuild:** VariantPostBuild
Origin	AUTOSAR_ECUC

Parameter Name	**CanControllerSeg1**
Description	Specifies phase segment 1 in time quantas.
Multiplicity	1..1
Type	INTEGER
Default value	0
Range	<=16 >=1
Configuration class	**VariantPostBuild:** VariantPostBuild
Origin	AUTOSAR_ECUC

Parameter Name	**CanControllerSeg2**
Description	Specifies phase segment 2 in time quantas.
Multiplicity	1..1
Type	INTEGER
Default value	0
Range	<=16 >=1
Configuration class	**VariantPostBuild:** VariantPostBuild
Origin	AUTOSAR_ECUC

Parameter Name	**CanControllerSyncJumpWidth**
Description	Specifies the synchronization jump width for the controller in time quantas. This parameter is not used by the CAN driver.
Multiplicity	1..1
Type	INTEGER

Default value	0	
Range	<=10 >=1	
Configuration class	**VariantPostBuild:**	VariantPostBuild
Origin	AUTOSAR_ECUC	

3.1.5. CanControllerFdBaudrateConfig

Parameters included	
Parameter name	**Multiplicity**
CanControllerFdBaudRate	1..1
CanControllerPropSeg	1..1
CanControllerSeg1	1..1
CanControllerSeg2	1..1
CanControllerSyncJumpWidth	1..1
CanControllerTrcvDelayCompensationOffset	1..1
CanControllerTxBitRateSwitch	1..1
PCAN_frequency	1..1

Parameter Name	**CanControllerFdBaudRate**	
Description	Specifies the buadrate of the controller in kbps.	
Multiplicity	1..1	
Type	INTEGER	
Default value	2500	
Range	<=10000000 >=1	
Configuration class	**VariantPostBuild:**	VariantPostBuild
Origin	INFINEON_AURIX	

Parameter Name	**CanControllerPropSeg**
Description	Specifies propagation delay in time quantas. NOTE: this value is not used
Multiplicity	1..1

Type	INTEGER
Default value	0
Range	<=15 >=0
Configuration class	**VariantPostBuild:** VariantPostBuild
Origin	INFINEON_AURIX

Parameter Name	**CanControllerSeg1**
Description	Specifies phase segment 1 in time quantas.
Multiplicity	1..1
Type	INTEGER
Default value	5
Range	<=16 >=1
Configuration class	**VariantPostBuild:** VariantPostBuild
Origin	INFINEON_AURIX

Parameter Name	**CanControllerSeg2**
Description	Specifies phase segment 2 in time quantas.
Multiplicity	1..1
Type	INTEGER
Default value	4
Range	<=16 >=1
Configuration class	**VariantPostBuild:** VariantPostBuild
Origin	INFINEON_AURIX

Parameter Name	**CanControllerSyncJumpWidth**
Description	Specifies the synchronization jump width for the controller in time quantas. This parameter is not used by the CAN driver.
Multiplicity	1..1
Type	INTEGER
Default value	1

Range	<=10	
	>=1	
Configuration class	**VariantPostBuild:**	VariantPostBuild
Origin	INFINEON_AURIX	

Parameter Name	**CanControllerTrcvDelayCompensationOffset**	
Description	Specifies the Transceiver Delay Compensation Offset in ns.	
Multiplicity	1..1	
Type	INTEGER	
Default value	0	
Range	<=400	
	>=0	
Configuration class	**VariantPostBuild:**	VariantPostBuild
Origin	INFINEON_AURIX	

Parameter Name	**CanControllerTxBitRateSwitch**	
Description	Specifies if the bit rate switching shall be used for transmissions.	
Multiplicity	1..1	
Type	BOOLEAN	
Default value	true	
Configuration class	**VariantPostBuild:**	VariantPostBuild
Origin	INFINEON_AURIX	

Parameter Name	**PCAN_frequency**
Description	defines the frequency for pcan devices Note: this switch is disabled if CanController/CanHardwareEmulation/CanHardware is not set to CIF_PCAN_CT_USB
Multiplicity	1..1
Type	ENUMERATION
Default value	PCAN_FD_20_MHZ
Range	PCAN_FD_20_MHZ
	PCAN_FD_24_MHZ
	PCAN_FD_30_MHZ
	PCAN_FD_40_MHZ

	PCAN_FD_60_MHZ	
	PCAN_FD_80_MHZ	
Configuration class	**PreCompile:**	VariantPreCompile
	PreCompile:	VariantPostBuild
Origin	Elektrobit Automotive Software	

3.1.6. CanFilterMask

Parameters included	
Parameter name	**Multiplicity**
CanFilterMaskValue	1..1

Parameter Name	**CanFilterMaskValue**	
Description	Describes a mask for hardware-based filtering of CAN identifiers. The CAN identifiers of incoming messages are masked with the appropriate CanFilterMaskValue. Bits holding a 0 mean don't care, i.e. do not compare the message's identifier in the respective bit position.	
Multiplicity	1..1	
Type	INTEGER	
Default value	0	
Range	<=4294967295	
	>=0	
Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

3.1.7. CanHardwareObject

Parameters included	
Parameter name	**Multiplicity**
CanHandleType	1..1
CanIdType	1..1

Parameters included	
CanIdValue	1..1
CanObjectId	1..1
CanObjectType	1..1
CanControllerRef	1..1
CanFilterMaskRef	1..1
CanMainFunctionRWPeriodRef	0..1

Parameter Name	CanHandleType	
Description	Specifies the type (Full-CAN or Basic-CAN) of a hardware object.	
Multiplicity	1..1	
Type	ENUMERATION	
Default value	BASIC	
Range	BASIC	
	FULL	
Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	CanIdType	
Description	Specifies whether the IdValue is of type	
Multiplicity	1..1	
Type	ENUMERATION	
Range	EXTENDED	
	MIXED	
	STANDARD	
Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	CanIdValue
Description	Specifies (together with the filter mask) the identifiers range that passes the hardware filter.
Multiplicity	1..1

Type	INTEGER	
Default value	0	
Range	<=4294967295	
	>=0	
Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CanObjectId**	
Description	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.	
Multiplicity	1..1	
Type	INTEGER	
Default value	0	
Range	<=65535	
	>=0	
Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CanObjectType**	
Description	Specifies if the HardwareObject is used as Transmit or as Receive object	
Multiplicity	1..1	
Type	ENUMERATION	
Range	RECEIVE	
	TRANSMIT	
Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CanControllerRef**
Description	Reference to CAN Controller to which the HOH is associated to.
Multiplicity	1..1

Type	REFERENCE	
Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CanFilterMaskRef**	
Description	Reference to the filter mask that is used for hardware filtering together with the CAN_ID_VALUE	
Multiplicity	1..1	
Type	REFERENCE	
Configuration class	**PostBuild:**	VariantPostBuild
	PreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CanMainFunctionRWPeriodRef**	
Description	Reference to CAN Controller to which the HOH is associated to.	
Multiplicity	0..1	
Type	REFERENCE	
Configuration class	**PreCompile:**	VariantPreCompile
	PostBuild:	VariantPostBuild
Origin	AUTOSAR_ECUC	

3.1.8. CanGeneral

Containers included		
Container name	**Multiplicity**	**Description**
CanMainFunctionRWPeriods	1..1	This container contains the parameters related each CAN Driver Unit.

Parameters included	
Parameter name	**Multiplicity**
CanChangeBaudrateApi	1..1
CanDevErrorDetection	1..1
CanHardwareCancellation	1..1

Parameters included	
CanIdenticalIdCancellation	1..1
CanIndex	1..1
CanLPduReceiveCalloutFunction	0..1
CanMainFunctionBusoffPeriod	0..1
CanMainFunctionModePeriod	1..1
CanMainFunctionWakeupPeriod	0..1
CanMultiplexedTransmission	1..1
CanTimeoutDuration	1..1
CanVersionInfoApi	1..1
CanCounterRef	1..1
CanMainFunctionReadPeriodRef	0..1
CanMainFunctionWritePeriodRef	0..1
CanSupportTTCANRef	1..1
CanBufferMode	1..1

Parameter Name	CanChangeBaudrateApi	
Description	The support of the Can_ChangeBaudrate API is optional. If this parameter is set to true the Can_ChangeBaudrate API shall be supported. Otherwise the API is not supported.	
Multiplicity	1..1	
Type	BOOLEAN	
Default value	false	
Configuration class	**VariantPreCompile:**	VariantPreCompile
	VariantPostBuild:	VariantPostBuild
Origin	AUTOSAR_ECUC	

Parameter Name	CanDevErrorDetection	
Description	Switches the Development Error Detection and Notification ON or OFF.	
Multiplicity	1..1	
Type	BOOLEAN	
Default value	true	
Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile

Origin	AUTOSAR_ECUC

Parameter Name	**CanHardwareCancellation**	
Description	Specifies if hardware cancellation shall be supported.ON or OFF	
Multiplicity	1..1	
Type	BOOLEAN	
Default value	true	
Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CanIdenticalIdCancellation**	
Description	Enables/disables cancellation of pending PDUs with identical ID.	
Multiplicity	1..1	
Type	BOOLEAN	
Default value	true	
Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CanIndex**	
Description	Specifies the InstanceId of this module instance. If only one instance is present it shall have the Id 0.	
Multiplicity	1..1	
Type	INTEGER	
Default value	0	
Range	<=255	
	>=0	
Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CanLPduReceiveCalloutFunction**
Description	This parameter defines the existence and the name of a callout function that is called after a successful

Multiplicity	0..1	
Type	FUNCTION-NAME	
Configuration class	**PreCompile:**	VariantPostBuild
	PreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CanMainFunctionBusoffPeriod**	
Description	This parameter describes the period for cyclic call to Can_MainFunction_Busoff. Unit is seconds.	
Multiplicity	0..1	
Type	FLOAT	
Default value	0.005	
Range	<=65.535	
	>=0.0010	
Configuration class	**PreCompile:**	VariantPostBuild
	PreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CanMainFunctionModePeriod**	
Description	This parameter describes the period for cyclic call to Can_MainFunction_Mode. Unit is seconds.	
Multiplicity	1..1	
Type	FLOAT	
Default value	0.005	
Range	<=65.535	
	>=0.0010	
Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CanMainFunctionWakeupPeriod**
Description	This parameter describes the period for cyclic call to Can_MainFunction_Wake-up. Unit is seconds.
Multiplicity	0..1

Type	FLOAT	
Default value	0.005	
Range	<=65.535	
	>=0.0010	
Configuration class	**PreCompile:**	VariantPostBuild
	PreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CanMultiplexedTransmission**	
Description	Specifies if multiplexed transmission shall be supported.ON or OFF	
Multiplicity	1..1	
Type	BOOLEAN	
Default value	true	
Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CanTimeoutDuration**	
Description	Specifies the maximum time for blocking function until a timeout is detected. Unit is seconds.	
Multiplicity	1..1	
Type	FLOAT	
Default value	0.001	
Range	<=65.535	
	>=0.0010	
Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CanVersionInfoApi**
Description	Switches the Can_GetVersionInfo() API ON or OFF.
Multiplicity	1..1
Type	BOOLEAN

Default value	false	
Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CanCounterRef**	
Description	This parameter contains a reference to the counter which is used by the CAN driver.	
Multiplicity	1..1	
Type	REFERENCE	
Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CanMainFunctionReadPeriodRef**	
Description	Reference to CAN Hardware Object which shall be polled with the configured CanMainFunctionReadPeriod. This reference shall only be configurable if more than one period is configured via CanMainFunctionReadPeriod.	
Multiplicity	0..1	
Type	REFERENCE	
Configuration class	**PreCompile:**	VariantPostBuild
	PreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CanMainFunctionWritePeriodRef**	
Description	Reference to CAN Hardware Object which shall be polled with the configured CanMainFunctionWritePeriod. This reference shall only be configurable if more than one period is configured via CanMainFunctionWritePeriod.	
Multiplicity	0..1	
Type	REFERENCE	
Configuration class	**PreCompile:**	VariantPostBuild
	PreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CanSupportTTCANRef**

Description	The parameter refers to CanIfSupportTTCAN parameter in the CAN Interface Module configuration.	
Multiplicity	1..1	
Type	REFERENCE	
Configuration class	**VariantPostBuild:**	VariantPostBuild
	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CanBufferMode**
Description	Specifies the behaviour in case of a full receive buffer. ► KEEP_OLDEST: New frames are discarded. ► KEEP_NEWEST: New frames overwrite the oldest frames in the buffer.
Multiplicity	1..1
Type	ENUMERATION
Default value	KEEP_OLDEST
Range	KEEP_OLDEST KEEP_NEWEST
Configuration class	**VariantPreCompile:** VariantPreCompile **VariantPostBuild:** VariantPostBuild
Origin	Elektrobit Automotive Software

3.1.9. CanMainFunctionRWPeriods

Parameters included	
Parameter name	**Multiplicity**
CanMainFunctionReadPeriod	0..n
CanMainFunctionWritePeriod	0..n

Parameter Name	**CanMainFunctionReadPeriod**
Description	This parameter describes the period for cyclic call to Can_MainFunction_Read. Unit is seconds. Different poll-cycles will be configurable if more than one CanMainFunctionReadPeriod is configured. In this case multiple Can_MainFunction_Read() will be provided by the CAN Driver module.

Multiplicity	0..n	
Type	FLOAT	
Default value	0.005	
Range	<=65.535 >=0.0010	
Configuration class	**PreCompile:**	VariantPreCompile
	PreCompile:	VariantPostBuild
Origin	AUTOSAR_ECUC	

Parameter Name	**CanMainFunctionWritePeriod**	
Description	This parameter describes the period for cyclic call to Can_MainFunction_Write. Unit is seconds. Different poll-cycles will be configurable if more than one CanMainFunctionWritePeriod is configured. In this case multiple Can_MainFunction_Write() will be provided by the CAN Driver module.	
Multiplicity	0..n	
Type	FLOAT	
Default value	0.005	
Range	<=65.535 >=0.0010	
Configuration class	**PreCompile:**	VariantPreCompile
	PreCompile:	VariantPostBuild
Origin	AUTOSAR_ECUC	

3.1.10. CommonPublishedInformation

Parameters included	
Parameter name	**Multiplicity**
ArMajorVersion	1..1
ArMinorVersion	1..1
ArPatchVersion	1..1
SwMajorVersion	1..1
SwMinorVersion	1..1
SwPatchVersion	1..1

Parameters included	
ModuleId	1..1
VendorId	1..1
Release	1..1

Parameter Name	ArMajorVersion
Label	AUTOSAR Major Version
Description	Major version number of AUTOSAR specification on which the appropriate implementation is based on.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	4
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	ArMinorVersion
Label	AUTOSAR Minor Version
Description	Minor version number of AUTOSAR specification on which the appropriate implementation is based on.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	0
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	ArPatchVersion
Label	AUTOSAR Patch Version
Description	Patch level version number of AUTOSAR specification on which the appropriate implementation is based on.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	0
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	SwMajorVersion
Label	Software Major Version
Description	Major version number of the vendor specific implementation of the module.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	3
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	SwMinorVersion
Label	Software Minor Version
Description	Minor version number of the vendor specific implementation of the module. The numbering is vendor specific.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	1
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	SwPatchVersion
Label	Software Patch Version
Description	Patch level version number of the vendor specific implementation of the module. The numbering is vendor specific.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	4
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	ModuleId
Label	Numeric Module ID
Description	Module ID of this module from Module List
Multiplicity	1..1
Type	INTEGER_LABEL

Default value	80
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	**VendorId**
Label	Vendor ID
Description	Vendor ID of the dedicated implementation of this module according to the AUTOSAR vendor list
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	1
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	**Release**
Label	Release Information
Multiplicity	1..1
Type	STRING_LABEL
Default value	
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

3.1.11. PublishedInformation

Parameters included	
Parameter name	**Multiplicity**
PbcfgMSupport	1..1

Parameter Name	**PbcfgMSupport**
Label	PbcfgM support
Description	Specifies whether or not the Can can use the PbcfgM module for post-build support.
Multiplicity	1..1

Type	BOOLEAN
Default value	false
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

3.2. Application programming interface (API)

3.2.1. Type definitions

3.2.1.1. Can_ConfigType

Purpose	Overall initialization data for all controllers.
Type	struct
Members	`uint8 dummy`
Description	This is the type of the external data structure containing the overall initialization data for the CAN driver and SFR settings affecting all controllers. Furthermore it contains pointers to controller structures. The contents of the initialization data structure are CAN hardware specific.

3.2.1.2. Can_ControllerBaudrateConfigType

Purpose	
Type	`Can_ControllerType`

3.2.2. Macro constants

3.2.2.1. CAN_API_CBK_CHECK_WAKEUP

Purpose	Service ID of Can_Init().

Value	0x0bU

3.2.2.2. CAN_API_DISABLE_CONTROLLER_INTERRUPTS

Purpose	Service ID of Can_Init().
Value	0x04U

3.2.2.3. CAN_API_ENABLE_CONTROLLER_INTERRUPTS

Purpose	Service ID of Can_Init().
Value	0x05U

3.2.2.4. CAN_API_GET_VERSION_INFO

Purpose	Service ID of Can_Init().
Value	0x07U

3.2.2.5. CAN_API_INIT

Purpose	Service ID of Can_Init().
Value	0x00U

3.2.2.6. CAN_API_INIT_CONTROLLER

Purpose	Service ID of Can_Init().
Value	0x02U

3.2.2.7. CAN_API_MAIN_FUNCTION_BUS_OFF

Purpose	Service ID of Can_MainFunction_Busoff().
Value	0x09U

3.2.2.8. CAN_API_MAIN_FUNCTION_MODE

Purpose	Service ID of Can_MainFunction_Mode().
Value	0x0cU

3.2.2.9. CAN_API_MAIN_FUNCTION_READ

Purpose	Service ID of Can_Init().
Value	0x08U

3.2.2.10. CAN_API_MAIN_FUNCTION_WAKEUP

Purpose	Service ID of Can_MainFunction_Wakeup().
Value	0x0aU

3.2.2.11. CAN_API_MAIN_FUNCTION_WRITE

Purpose	Service ID of Can_Init().
Value	0x01U

3.2.2.12. CAN_API_SET_CONTROLLER_MODE

Purpose	Service ID of Can_Init().
Value	0x03U

3.2.2.13. CAN_API_STORE_IN_RX_BUFFER

Purpose	Service ID of Can_StoreInRxBuffer().
Value	0x10U

3.2.2.14. CAN_API_STORE_IN_TX_BUFFER

Purpose	Service ID of Can_StoreInTxBuffer().

Value	0x11U

3.2.2.15. CAN_API_WRITE

Purpose	Service ID of Can_Init().
Value	0x06U

3.2.2.16. CAN_E_DATALOST

Purpose	DET error code CAN_E_DATALOST.
Value	0x07U

3.2.2.17. CAN_E_PARAM_CONTROLLER

Purpose	DET error code CAN_E_PARAM_CONTROLLER.
Value	0x04U

3.2.2.18. CAN_E_PARAM_DLC

Purpose	DET error code CAN_E_PARAM_DLC.
Value	0x03U

3.2.2.19. CAN_E_PARAM_HANDLE

Purpose	DET error code CAN_E_PARAM_HANDLE.
Value	0x02U

3.2.2.20. CAN_E_PARAM_POINTER

Purpose	DET error code CAN_E_PARAM_POINTER.
Value	0x01U

3.2.2.21. CAN_E_TRANSITION

Purpose	DET error code CAN_E_TRANSITION.
Value	0x06U

3.2.2.22. CAN_E_UNINIT

Purpose	DET error code CAN_E_UNINIT.
Value	0x05U

3.2.3. Objects

3.2.3.1. CAN_CONFIG_SET

Purpose	Can Config.
Type	`const Can_ConfigType`
Description	This is the dummy constant to be used as parameter for Can_Init();

3.2.3.2. Can_ControllerConfigDummy

Purpose	Can Controller Config.
Type	`const Can_ControllerBaudrateConfigType`
Description	This is the dummy constant to be used as parameter for Can_InitController();

3.2.4. Functions

3.2.4.1. Can_CheckWakeup

Purpose	Check wakeup callback.

Synopsis	`Can_ReturnType` **`Can_CheckWakeup`** `( uint8 Controller );`	
Service ID	0x0b	
Sync/Async	Synchronous	
Reentrancy	Non-Reentrant	
Parameters (in)	`Controller`	Controller to be checked for a wakeup
Return Value	Result of the check for a wakeup	
	`E_OK`	A wakeup was detected for given controller.
	`E_NOT_OK`	No wakeup was detected for the given controller.
Description	This function checks if a wakeup has occured for the given controller.	

3.2.4.2. Can_DisableControllerInterrupts

Purpose	Disable controller interrupts.	
Synopsis	`void` **`Can_DisableControllerInterrupts`** `( uint8 Controller );`	
Service ID	0x04	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	`Controller`	CAN controller for which interrupts shall be disabled.
Description	This function disables all interrupts for this CAN controller	

3.2.4.3. Can_EnableControllerInterrupts

Purpose	Enable controller interrupts.	
Synopsis	`void` **`Can_EnableControllerInterrupts`** `( uint8 Controller );`	
Service ID	0x05	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	`Controlelr`	CAN controller for which interrupts shall be re-enabled.

Description	This function enables all allowed interrupts.

3.2.4.4. Can_GetVersionInfo

Purpose	Get version info.	
Synopsis	`void` **`Can_GetVersionInfo`** `( Std_VersionInfoType * versioninfo );`	
Service ID	0x07	
Sync/Async	Synchronous	
Reentrancy	Non-Reentrant	
Parameters (out)	`versioninfo`	Pointer to where to store the version information of this module.
Description	This function returns the version information of this module.	

3.2.4.5. Can_Init

Purpose	Init.	
Synopsis	`void` **`Can_Init`** `( const Can_ConfigType * Config );`	
Service ID	0x00	
Sync/Async	Synchronous	
Reentrancy	Non-Reentrant	
Parameters (in)	`Config`	Pointer to driver configuration.
Description	This function initialized the module.	

3.2.4.6. Can_InitController

Purpose	Init controller.
Synopsis	`void` **`Can_InitController`** `( uint8 Controller , const Can_ControllerBaudrateConfigType * Config );`
Service ID	0x02
Sync/Async	Synchronous
Reentrancy	Non-Reentrant

Parameters (in)	`Controller`	CAN controller to be initialized
	`Config`	Pointer to controller configuration
Description	This function initializes only CAN controller specific settings.	

3.2.4.7. Can_MainFunction_BusOff

Purpose	BusOff Main Function.
Synopsis	`void` **`Can_MainFunction_BusOff`** `( void );`
Service ID	0x09
Description	This function performs the polling of bus-off events that are configured statically as 'to be polled'.

3.2.4.8. Can_MainFunction_Mode

Purpose	Mode Main Function This function performs the polling of CAN coltroller mode transitions .
Synopsis	`void` **`Can_MainFunction_Mode`** `( void );`

3.2.4.9. Can_MainFunction_Read

Purpose	Read Main Function.
Synopsis	`void` **`Can_MainFunction_Read`** `( void );`
Service ID	0x08
Description	This function performs the polling of RX indications when CAN_RX_PROCESSING is set to POLLING. For each configured cycle time, a dedicated function is generated with a zero-based consecutive value appended to its name.

3.2.4.10. Can_MainFunction_Read_Internal

Purpose	
Synopsis	`void` **`Can_MainFunction_Read_Internal`** `( void );`

3.2.4.11. Can_MainFunction_Wakeup

Purpose	Wakeup Main Function.
Synopsis	`void` **`Can_MainFunction_Wakeup`** `( void );`
Service ID	0x0a
Description	This function performs the polling of wake-up events that are configured statically as 'to be polled'.

3.2.4.12. Can_MainFunction_Write

Purpose	Write Main Function.
Synopsis	`void` **`Can_MainFunction_Write`** `( void );`
Service ID	0x01
Description	This function performs the polling of TX confirmation and TX cancellation confirmation when CAN_TX_PROCESSING is set to POLLING. For each configured cycle time, a dedicated function is generated with a zero-based consecutive value appended to its name.

3.2.4.13. Can_MainFunction_Write_Internal

Purpose	
Synopsis	`void` **`Can_MainFunction_Write_Internal`** `( void );`

3.2.4.14. Can_SetControllerMode

Purpose	Set controller mode.	
Synopsis	`Can_ReturnType` **`Can_SetControllerMode`** `( uint8 Controller , Can_StateTransitionType Transition );`	
Service ID	0x03	
Sync/Async	Asynchronous	
Reentrancy	Non-Reentrant	
Parameters (in)	`Controller`	CAN controller for which the status shall be changed

	`Transition`	State transition to be performed
Parameters (in,out)		
Return Value	Status of the state change	
	`CAN_OK`	transition initiated
	`CAN_NOT_OK`	Development or production or a wakeup during transition to 'sleep' occured.
Description	This function performs software triggered state transitions of the CAN controller state machine.	

3.2.4.15. Can_Write

Purpose	Write.	
Synopsis	`Can_ReturnType` **`Can_Write`** `( uint8 Hth , const Can_PduType * PduInfo );`	
Service ID	0x06	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	`Hth`	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.
	`PduInfo`	Pointer to SDU user memory, DLC and identifier.
Return Value	Success of transmission request	
	`CAN_OK`	Write command has been accepted.
	`CAN_NOT_OK`	Development error occured.
	`CAN_BUSY`	No TX hardware buffer available or pre-emtive call of Can_Write that can't be implemented reentrant.
Description	This function triggers the transmission of a CAN frame.	