MCAL Configuration Verification Manual

32-bit TriCore™ AURIX™ TC3xx microcontroller family

About this document

Scope and purpose

This Configuration Data Reference document is applicable to all TC3xx devices in the TriCore™ AURIX™ family of 32-bit microcontrollers.

The purpose of this document is to facilitate the integrator to verify the generated code based on the input configuration parameters. This document describes details of structures, defines, macros and variables generated from the configuration parameters for ASIL-B modules.

Intended audience

This document is intended for integrators who need to understand the logic of the generated configuration code of AURIX™ AUTOSAR MCAL ASIL-B modules.

Reference documents

This document should be read in conjunction with the following documents:

- AURIX™ TC3xx MCAL User Manual Can_17_McmCan

Table of contents

1 Can_17_McmCan driver

This chapter describes the details of the configuration data generated from the CAN driver.

1.1 File: Can_17_McmCan_Cfg.h

The generated header file contains all pre-compile configuration parameters. Pre-compile time configuration allows decoupling of the static configuration from implementation. The file is generated in ‘inc’ folder.

1.1.1 Macro: CAN_17_MCMCAN_AR_RELEASE_MAJOR_VERSION

Table 1 CAN_17_MCMCAN_AR_RELEASE_MAJOR_VERSION

Name	CAN_17_MCMCAN_AR_RELEASE_MAJOR_VERSION	
Description	Major version number of AUTOSAR release on which the Can_17_McmCan implementation is based on.	
Verification method	The macro is generated with the value present in ‘CommonPublishedInformation/ArMajorVersion’. *Note: The macro is not user configurable.*	
Example(s)	**Action**	**Generated output**
	Generate Can_17_McmCan_Cfg.h file with ArMajorVersion 4	`#define CAN_17_MCMCAN_AR_RELEASE_MAJOR_VERSION (4U)`

1.1.2 Macro: CAN_17_MCMCAN_AR_RELEASE_MINOR_VERSION

Table 2 CAN_17_MCMCAN_AR_RELEASE_MINOR_VERSION

Name	CAN_17_MCMCAN_AR_RELEASE_MINOR_VERSION	
Description	Minor version number of AUTOSAR release on which the Can_17_McmCan implementation is based on.	
Verification method	The macro is generated with the value present in ‘CommonPublishedInformation/ArMinorVersion’. *Note: The macro is not user configurable.*	
Example(s)	**Action**	**Generated output**
	Generate Can_17_McmCan_Cfg.h file with ArMinorVersion 2	`#define CAN_17_MCMCAN_AR_RELEASE_MINOR_VERSION (2U)`

1.1.3 Macro: CAN_17_MCMCAN_AR_RELEASE_REVISION_VERSION

Table 3 CAN_17_MCMCAN_AR_RELEASE_REVISION_VERSION

Name	CAN_17_MCMCAN_AR_RELEASE_REVISION_VERSION	
Description	Revision version number of AUTOSAR release on which the Can_17_McmCan implementation is based on.	
Verification method	The macro is generated with the value present in 'CommonPublishedInformation/ArPatchVersion'. *Note: The macro is not user configurable.*	
Example(s)	**Action**	**Generated output**
	Generate Can_17_McmCan_Cfg.h file with ArPatchVersion 2	`#define CAN_17_MCMCAN_AR_RELEASE_REVISION_VERSION (2U)`

1.1.4 Macro: CAN_17_MCMCAN_SW_MAJOR_VERSION

Table 4 CAN_17_MCMCAN_SW_MAJOR_VERSION

Name	CAN_17_MCMCAN_SW_MAJOR_VERSION	
Description	Major version number of the Can_17_McmCan module.	
Verification method	The macro is generated with the value present in 'CommonPublishedInformation/SwMajorVersion'. *Note: The macro is not user configurable.*	
Example(s)	**Action**	**Generated output**
	Generate Can_17_McmCan_Cfg.h file with SwMajorVersion 10	`#define CAN_17_MCMCAN_SW_MAJOR_VERSION (10U)`

1.1.5 Macro: CAN_17_MCMCAN_SW_MINOR_VERSION

Table 5 CAN_17_MCMCAN_SW_MINOR_VERSION

Name	CAN_17_MCMCAN_SW_MINOR_VERSION	
Description	Minor version number of the Can_17_McmCan module.	
Verification method	The macro is generated with the value present in 'CommonPublishedInformation/SwMinorVersion'. *Note: The macro is not user configurable.*	
Example(s)	**Action**	**Generated output**

	Generate Can_17_McmCan_Cfg.h file with SwMinorVersion 10	`#define CAN_17_MCMCAN_SW_MINOR_VERSION (10U)`

1.1.6 Macro: CAN_17_MCMCAN_SW_PATCH_VERSION

Table 6 CAN_17_MCMCAN_SW_PATCH_VERSION

Name	CAN_17_MCMCAN_SW_PATCH_VERSION	
Description	Patch level version number of the Can_17_McmCan module.	
Verification method	The macro is generated with the value present in 'CommonPublishedInformation/SwPatchVersion'. *Note: The macro is not user configurable.*	
Example(s)	**Action**	**Generated output**
	Generate Can_17_McmCan_Cfg.h file with SwPatchVersion 0	`#define CAN_17_MCMCAN_SW_PATCH_VERSION (0U)`

1.1.7 Macro: CAN_17_MCMCAN_LPDU_RX_CALLOUT

Table 7 CAN_17_MCMCAN_ LPDU_RX_CALLOUT

Name	CAN_17_MCMCAN_ LPDU_RX_CALLOUT	
Description	Indicates if receive L-PDU callout function exist.	
Verification method	The macro is generated as STD_ON if L-PDU callout function is added in CanLPduReceiveCalloutFunction else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	• Configure 1 CAN L-PDU callout function.	`#define CAN_17_MCMCAN_LPDU_RX_CALLOUT (STD_ON)`
	• Do not configure any CAN L-PDU callout function.	`#define CAN_17_MCMCAN_LPDU_RX_CALLOUT (STD_OFF)`

1.1.8 Macro: CAN_17_MCMCAN_MASTER_CORE_ALLOCATION

Table 8 CAN_17_MCMCAN_MASTER_CORE_ALLOCATION

Name	CAN_17_MCMCAN_MASTER_CORE_ALLOCATION
Description	Indicates if the master core has any resources(controllers) allocated to it.
Verification method	The macro is generated as STD_ON if controller is assigned to master core else the macro is generated as STD_OFF.

	Note:	*Note: Controllers configured and not assigned to any core are assigned to master core (ResourceMMasterCore).*
Example(s)	**Action**	**Generated output**
	• Configure 1 CAN controller. • Set ResourceMMasterCore as CORE1. • Do not assign CAN controllers in any ResourceMAllocation	`#define CAN_17_MCMCAN_MASTER_CORE_ALLOCATION (STD_ON)`
	• Configure 4 CAN controllers. • Set ResourceMMasterCore as CORE1. • Assign all 4 controllers configured under ResourceMAllocation with ResourceMCoreID as CORE0.	`#define CAN_17_MCMCAN_MASTER_CORE_ALLOCATION (STD_OFF)`

1.1.9 Macro: CAN_17_MCMCAN_MULTICORE_ERROR_DETECT

Table 9 CAN_17_MCMCAN_MULTICORE_ERROR_DETECT

Name	CAN_17_MCMCAN_MULTICORE_ERROR_DETECT	
Description	Enables/Disables Multicore DET Check	
Verification method	The macro is generated as STD_ON if CanMultiCoreErrorDetect configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	CanMultiCoreErrorDetect = True	`#define CAN_17_MCMCAN_MULTICORE_ERROR_DETECT (STD_ON)`
	CanMultiCoreErrorDetect = False	`#define CAN_17_MCMCAN_MULTICORE_ERROR_DETECT (STD_OFF)`

1.1.10 Macro: CAN_17_MCMCAN_RUNTIME_ERROR_DETECT

Table 10 CAN_17_MCMCAN_ RUNTIME_ERROR_DETECT

Name	CAN_17_MCMCAN_ RUNTIME_ERROR_DETECT
Description	Specifies whether runtime error detection is enabled in case of AUTOSAR 4.4.0. In case of AUTOSAR 4.2.2 runtime error detection is not applicable as there are no runtime errors to be reported. In AUTOSAR 4.2.2 there is no parameter CanRunTimeErrorDetect.

Verification method	In case of AUTOSAR 4.40, the macro is generated as STD_ON if CanRunTimeErrorDetect configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	• Validate AUTOSAR minor version is 4. • CanRunTimeErrorDetect = True	`#define CAN_17_MCMCAN_ RUNTIME _ERROR_DETECT (STD_ON)`
	• Validate AUTOSAR minor version is 4. • CanRunTimeErrorDetect = False	`#define CAN_17_MCMCAN_ RUNTIME _ERROR_DETECT (STD_OFF)`

1.1.11 Macro: CAN_17_MCMCAN_NOOF_CONTROLLER

Table 11 CAN_17_MCMCAN_NOOF_CONTROLLER

Name	CAN_17_MCMCAN_NOOF_CONTROLLER	
Description	Indicates the total number of controllers configured in the CAN driver	
Verification method	The macro is generated as a numeric value which corresponds to the number of elements in the list 'CanConfigSet/ CanController'.	
Example(s)	**Action**	**Generated output**
	Configure 4 CAN controllers	`#define CAN_17_MCMCAN_NOOF_CONTROLLER (4)`
	Configure 6 CAN controllers	`#define CAN_17_MCMCAN_NOOF_CONTROLLER (6)`

1.1.12 Macro: CAN_17_MCMCAN_CORE<x>_NOOF_CONTROLLER

Table 12 CAN_17_MCMCAN_CORE<x>_NOOF_CONTROLLER

Name	CAN_17_MCMCAN_CORE<x>_NOOF_CONTROLLER	
Description	Indicates the total number of controllers configured for CORE<x>.	
Verification method	The macro is generated as total number of controllers allocated to CORE<x>. *Note:* *Note: Controllers not assigned to any core are assigned to master core (ResourceMMasterCore).*	
Example(s)	**Action**	**Generated output**
	• Configure 4 CAN controllers. • Set ResourceMMasterCore as CORE1. Do not assign CAN controllers in any ResourceMAllocation	`#define CAN_17_MCMCAN_CORE1_NOOF_CONTROLLER (4)`

	- Configure 4 CAN controllers.
	- Set ResourceMMasterCore as CORE1.
	- Assign 3 controllers configured under ResourceMAllocation with ResourceMCoreID as CORE0.
		`#define CAN_17_MCMCAN_CORE1_NOOF_CONTROLLER (1)`
		`#define CAN_17_MCMCAN_CORE0_NOOF_CONTROLLER (3)`

1.1.13 Macro: CAN_17_MCMCAN_NOOF_RX_TX_PERIODS_CONFIG

Table 13 CAN_17_MCMCAN_NOOF_RX_TX_PERIODS_CONFIG

Name	CAN_17_MCMCAN_NOOF_RX_TX_PERIODS_CONFIG	
Description	Indicates the total number of read write periods configured	
Verification method	The macro is generated as a numeric value which corresponds to the number of elements in the list 'CanGeneral/CanMainFunctionRWPeriods'.	
Example(s)	**Action**	**Generated output**
	Configure 4 read write periods	`#define CAN_17_MCMCAN_NOOF_RX_TX_PERIODS_CONFIG (4)`
	Configure 10 read write periods	`#define CAN_17_MCMCAN_NOOF_RX_TX_PERIODS_CONFIG (10)`

1.1.14 Macro: CAN_17_MCMCAN_RX_MULTI_PERIODS_SUPPORT

Table 14 CAN_17_MCMCAN_RX_MULTI_PERIODS_SUPPORT

Name	CAN_17_MCMCAN_RX_MULTI_PERIODS_SUPPORT	
Description	Enables/Disables multi-period read support.	
Verification method	The macro is generated as STD_ON if at least one of the CAN controllers has Rx processing as 'Polling' and the number of elements in the list 'CanGeneral/CanMainFunctionRWPeriods' is greater than 1 else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	- Configure 4 read write periods in list CanMainFunctionRWPeriods - Configure 4 CAN controllers - Configure 1 of the CAN controllers with CanRxProcessing set as 'Polling'.	`#define CAN_17_MCMCAN_RX_MULTI_PERIODS_SUPPORT (STD_ON)`

	• Configure 1 read write periods in list CanMainFunctionRWPeriods • Configure 4 CAN controllers • Configure 1 of the CAN controllers with CanRxProcessing set as 'Polling'.	`#define CAN_17_MCMCAN_RX_MULTI_PERIODS_SUPPORT (STD_OFF)`
	• Configure 4 read write periods in list CanMainFunctionRWPeriods • Configure 4 CAN controllers • Configure all of the CAN controllers with CanRxProcessing set as 'Interrupt'.	`#define CAN_17_MCMCAN_RX_MULTI_PERIODS_SUPPORT (STD_OFF)`

1.1.15 Macro: CAN_17_MCMCAN_TX_MULTI_PERIODS_SUPPORT

Table 15 CAN_17_MCMCAN_TX_MULTI_PERIODS_SUPPORT

Name	CAN_17_MCMCAN_TX_MULTI_PERIODS_SUPPORT	
Description	Enables/Disables multi-period write support.	
Verification method	The macro is generated as STD_ON if at least one of the CAN controllers has Tx processing as 'Polling' and the number of elements in the list 'CanGeneral/CanMainFunctionRWPeriods' is greater than 1 else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	• Configure 4 read write periods in list CanMainFunctionRWPeriods • Configure 4 CAN controllers • Configure 1 of the CAN controllers with CanTxProcessing set as 'Polling'.	`#define CAN_17_MCMCAN_TX_MULTI_PERIODS_SUPPORT (STD_ON)`
	• Configure 1 read write periods in list CanMainFunctionRWPeriods • Configure 4 CAN controllers • Configure 1 of the CAN controllers with CanTxProcessing set as 'Polling'.	`#define CAN_17_MCMCAN_TX_MULTI_PERIODS_SUPPORT (STD_OFF)`

	• Configure 4 read write periods in list CanMainFunctionRWPeriods • Configure 4 CAN controllers • Configure all of the CAN controllers with CanTxProcessing set as 'Interrupt'.	`#define CAN_17_MCMCAN_TX_MULTI_PERIODS_SUPPORT (STD_OFF)`

1.1.16 Macro: CAN_17_MCMCAN_NOOF_ICOM_CONFIGURATIONS

Table 16 CAN_17_MCMCAN_NOOF_ICOM_CONFIGURATIONS

Name	CAN_17_MCMCAN_NOOF_ICOM_CONFIGURATIONS	
Description	Indicates the total number of ICOM configured	
Verification method	The macro is generated as a numeric value which corresponds to the number of elements in the list 'CanIcomConfig'. *Note: Note: This parameter is generated only if 'CanGeneral/CanPublicIcomSupport' is set to 'True'.*	
Example(s)	**Action**	**Generated output**
	Configure 14 CanIcomConfig	`#define CAN_17_MCMCAN_NOOF_ICOM_CONFIGURATIONS (14)`
	Configure 20 CanIcomConfig	`#define CAN_17_MCMCAN_NOOF_ICOM_CONFIGURATIONS (20)`

1.1.17 Macro: CAN_17_MCMCAN_NOOF_ICOM_MSGCONFIGURATIONS

Table 17 CAN_17_MCMCAN_NOOF_ICOM_MSGCONFIGURATIONS

Name	CAN_17_MCMCAN_NOOF_ICOM_MSGCONFIGURATIONS	
Description	Indicates the total number of ICOM messages configured	
Verification method	The macro is generated as a numeric value which corresponds to the sum of the number of elements in the list 'CanIcomWakeupCauses/CanIcomRxMessage'. *Note: Note: This parameter is generated only if 'CanGeneral/CanPublicIcomSupport' is set to 'True'.*	
Example(s)	**Action**	**Generated output**
	• Configure 2 CanIcomConfig.	`#define CAN_17_MCMCAN_NOOF_ICOM_MSGCONFIGURATIONS (7)`

	• Configure 1st CanIcomConfig having 5 CanIcomRxMessage'. • Configure 2nd CanIcomConfig having 2 CanIcomRxMessage'	
	• Configure 3 CanIcomConfig. • Configure 1st CanIcomConfig having 1 CanIcomRxMessage'. • Configure 2nd CanIcomConfig having 2 CanIcomRxMessage' • Configure 3rd CanIcomConfig having no CanIcomRxMessage'.	`#define CAN_17_MCMCAN_NOOF_ICOM_MSGCONFIGURATIONS (3)`

1.1.18 Macro: CAN_17_MCMCAN_NOOF_ICOM_SIGNALCONFIGURATIONS

Table 18 CAN_17_MCMCAN_NOOF_ICOM_SIGNALCONFIGURATIONS

Name	CAN_17_MCMCAN_NOOF_ICOM_SIGNALCONFIGURATIONS	
Description	Indicates the total number of ICOM signals configured	
Verification method	The macro is generated as a numeric value which corresponds to the sum of the number of elements in the list 'CanIcomWakeupCauses/CanIcomRxMessage/*/CanIcomRxMessageSignalConfig'. *Note: Note: This parameter is generated only if 'CanGeneral/CanPublicIcomSupport' is set to 'True'.*	
Example(s)	**Action**	**Generated output**
	• Configure 2 CanIcomConfig. • Configure 1st CanIcomConfig having 1 CanIcomRxMessage with 2 CanIcomRxMessageSignalConfig. • Configure 2nd CanIcomConfig having 2 CanIcomRxMessage with 2 and 3 CanIcomRxMessageSignalConfig	`#define CAN_17_MCMCAN_NOOF_ICOM_SIGNALCONFIGURATIONS (7)`

	- Configure 3 CanIcomConfig. - Configure 1st CanIcomConfig having 1 CanIcomRxMessage with 5 CanIcomRxMessageSignalConfig. - Configure 2nd CanIcomConfig having 2 CanIcomRxMessage with 2 and 6 CanIcomRxMessageSignalConfig - Configure 3rd CanIcomConfig having no CanIcomRxMessage'.	`#define CAN_17_MCMCAN_NOOF_ICOM_SIGNALCONFIGURATIONS (13)`

1.1.19 Macro: CAN_17_MCMCAN_INIT_DEINIT_API_MODE

Table 19 CAN_17_MCMCAN_INIT_DEINIT_API_MODE

Name	CAN_17_MCMCAN_INIT_DEINIT_API_MODE	
Description	Decides the mode of execution of Init and DeInit API's.	
Verification method	The macro is generated as CAN_17_MCMCAN_MCAL_USER1 if CanInitDeInitApiMode configuration parameter is set to 'CAN_17_MCMCAN_MCAL_USER1' else the macro is generated as CAN_17_MCMCAN_MCAL_SUPERVISOR.	
Example(s)	**Action**	**Generated output**
	CanInitDeInitApiMode = CAN_17_MCMCAN_MCAL_USER1	`#define CAN_17_MCMCAN_INIT_DEINIT_API_MODE (CAN_17_MCMCAN_MCAL_USER1)`
	CanInitDeInitApiMode = CAN_17_MCMCAN_MCAL_SUPERVISOR	`#define CAN_17_MCMCAN_INIT_DEINIT_API_MODE (CAN_17_MCMCAN_MCAL_SUPERVISOR)`

1.1.20 Macro: CAN_17_MCMCAN_INSTANCE_ID

Table 20 CAN_17_MCMCAN_INSTANCE_ID

Name	CAN_17_MCMCAN_INSTANCE_ID	
Description	Instance ID of CAN module.	
Verification method	The macro is generated as a numeric value set in the configuration parameter 'CanGeneral/CanIndex'	
Example(s)	**Action**	**Generated output**
	Set CanIndex as 0	`#define CAN_17_MCMCAN_INSTANCE_ID (0U)`
	Set CanIndex as 42	`#define CAN_17_MCMCAN_INSTANCE_ID (42U)`

1.1.21 Macro: CAN_17_MCMCAN_DEV_ERROR_DETECT

Table 21 CAN_17_MCMCAN_DEV_ERROR_DETECT

Name	CAN_17_MCMCAN_DEV_ERROR_DETECT	
Description	Enables/Disables the Development Error Detection.	
Verification method	The macro is generated as STD_ON if CanDevErrorDetect configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	CanDevErrorDetection = True	`#define CAN_17_MCMCAN_DEV_ERROR_DETECT (STD_ON)`
	CanDevErrorDetection = False	`#define CAN_17_MCMCAN_DEV_ERROR_DETECT (STD_OFF)`

1.1.22 Macro: CAN_17_MCMCAN_VERSION_INFO_API

Table 22 CAN_17_MCMCAN_VERSION_INFO_API

Name	CAN_17_MCMCAN_VERSION_INFO_API	
Description	Enables/Disables Can_17_McmCan_GetVersionInfo API	
Verification method	The macro is generated as STD_ON if CanVersionInfoApi configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	CanVersionInfoApi = True	`#define IC CAN_17_MCMCAN_VERSION_INFO_API (STD_ON)`
	CanVersionInfoApi = False	`#define CAN_17_MCMCAN_VERSION_INFO_API (STD_OFF)`

1.1.23 Macro: CAN_17_MCMCAN_MULTIPLEXED_TRANSMISSION

Table 23 CAN_17_MCMCAN_MULTIPLEXED_TRANSMISSION

Name	CAN_17_MCMCAN_MULTIPLEXED_TRANSMISSION	
Description	Enables/Disables multiplexed transmission support	
Verification method	The macro is generated as STD_ON if CanMultiplexedTransmission configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	CanMultiplexedTransmission = True	`#define CAN_17_MCMCAN_MULTIPLEXED_TRANSMISSION (STD_ON)`

	CanMultiplexedTransmission = False	`#define CAN_17_MCMCAN_MULTIPLEXED_TRANSMISSION (STD_OFF)`

1.1.24 Macro: CAN_17_MCMCAN_TIMEOUT_DURATION

Table 24 CAN_17_MCMCAN_TIMEOUT_DURATION

Name	CAN_17_MCMCAN_TIMEOUT_DURATION	
Description	Indicates the maximum amount of time allocated for a blocking function before timeout is raised.	
Verification method	The macro is generated as a numeric value which is the product of the configured value in container 'CanGeneral/CanTimeoutDuration' with 10000000.	
Example(s)	**Action**	**Generated output**
	Configure CanTimeoutDuration as 1	`#define CAN_17_MCMCAN_TIMEOUT_DURATION (10000000)`
	Configure CanTimeoutDuration as 20	`#define CAN_17_MCMCAN_TIMEOUT_DURATION (200000000)`

1.1.25 Macro: CAN_17_MCMCAN_SET_BAUDRATE_API

Table 25 CAN_17_MCMCAN_SET_BAUDRATE_API

Name	CAN_17_MCMCAN_SET_BAUDRATE_API	
Description	Enables/Disables Can_17_McmCan_CheckBaudrate and Can_17_McmCan_SetBaudrate API	
Verification method	The macro is generated as STD_ON if CanSetBaudrateApi configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	CanSetBaudrateApi= True	`#define CAN_17_MCMCAN_SET_BAUDRATE_API (STD_ON)`
	CanSetBaudrateApi= False	`#define CAN_17_MCMCAN_SET_BAUDRATE_API (STD_OFF)`

1.1.26 Macro: CAN_17_MCMCAN_FD_ENABLE

Table 26 CAN_17_MCMCAN_FD_ENABLE

Name	CAN_17_MCMCAN_FD_ENABLE
Description	Enables/Disables CANFD support

Verification method	The macro is generated as STD_ON if at least one of the CAN controllers have FD baudrate configured else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	• Configure 4 CAN controllers. • Configure CAN controller 1 with FD baudrate.	`#define CAN_17_MCMCAN_FD_ENABLE (STD_ON)`
	• Configure 4 CAN controllers. • Configure all CAN controllers without FD baudrate.	`#define CAN_17_MCMCAN_FD_ENABLE (STD_OFF)`

1.1.27 Macro: CAN_17_MCMCAN_DEINIT_API

Table 27 CAN_17_MCMCAN_DEINIT_API

Name	CAN_17_MCMCAN_DEINIT_API	
Description	Enables/Disables Can_17_McmCan_DeInit API. In case of AUTOSAR 4.4.0 the macro is always generated as STD_ON because Can_17_McmCan_DeInit is a mandatory interface.	
Verification method	The macro is generated as STD_ON if CanDeInitApi configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	CanDeInitApi = True	`#define CAN_17_MCMCAN_DEINIT_API (STD_ON)`
	CanDeInitApi = False	`#define CAN_17_MCMCAN_DEINIT_API (STD_OFF)`

1.1.28 Macro: CAN_17_MCMCAN_PUBLIC_ICOM_SUPPORT

Table 28 CAN_17_MCMCAN_PUBLIC_ICOM_SUPPORT

Name	CAN_17_MCMCAN_PUBLIC_ICOM_SUPPORT	
Description	Enables/Disables Can_17_McmCan_SetIcomConfiguration API and also pretended network overall support.	
Verification method	The macro is generated as STD_ON if CanPublicIcomSupport configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	CanPublicIcomSupport = True	`#define CAN_17_MCMCAN_PUBLIC_ICOM_SUPPORT (STD_ON)`
	CanPublicIcomSupport = False	`#define CAN_17_MCMCAN_PUBLIC_ICOM_SUPPORT (STD_OFF)`

1.1.29 Macro: CAN_17_MCMCAN_MAX_SWOBJECT_INDEX

Table 29 CAN_17_MCMCAN_MAX_SWOBJECT_INDEX

Name	CAN_17_MCMCAN_MAX_SWOBJECT_INDEX	
Description	The macro specifies the number of transmit hardware object buffers present.	
Verification method	The macro is generated by multiplying 32 with the value of Can.MaxControllers which is given in the property file.	
Example(s)	**Action**	**Generated output**
	Can.MaxControllers: 12	`#define CAN_17_MCMCAN_MAX_SWOBJECT_INDEX ((uint8)(384U))`

1.1.30 Macro: CAN_17_MCMCAN_NOOF_NODES_PER_KERNEL

Table 30 CAN_17_MCMCAN_NOOF_NODES_PER_KERNEL

Name	CAN_17_MCMCAN_NOOF_NODES_PER_KERNEL	
Description	The macro specifies the maximum number of controllers associated with a kernel. Eg: In a derivative if Kernel 0 has 3 controllers and Kernel 1 has 4, then the value generated would be 4.	
Verification method	The macro is generated based on the value of Can.MaxCtrlKer given in the property file.	
Example(s)	**Action**	**Generated output**
	Can.MaxCtrlKer: 4	`#define CAN_17_MCMCAN_NOOF_NODES_PER_KERNEL ((uint32)(4U))`

1.1.31 Macro: CAN_17_MCMCAN_CORE<x>_ACTIVATION

Table 31 CAN_17_MCMCAN_CORE<x>_ACTIVATION

Name	CAN_17_MCMCAN_CORE<x>_ACTIVATION	
Description	Indicates the configuration of the CORE<x>.	
Verification method	The macro is generated as STD_ON if atleast one Can controller is allocated to CORE<x>. *Note: Note: Channels not assigned to any core are assigned to master core (ResourceMMasterCore).*	
Example(s)	**Action**	**Generated output**

	• Configure 4 CAN controllers.
	• Set ResourceMMasterCore as CORE1.
	Do not assign CAN controllers in any ResourceMAllocation
		`#define CAN_17_MCMCAN_CORE0_ACTIVATION (STD_OFF)`
		`#define CAN_17_MCMCAN_CORE1_ACTIVATION (STD_ON)`

		#define CAN_17_MCMCAN_CORE2_ACTIVATION (STD_OFF) #define CAN_17_MCMCAN_CORE3_ACTIVATION (STD_OFF) #define CAN_17_MCMCAN_CORE4_ACTIVATION (STD_OFF) #define CAN_17_MCMCAN_CORE5_ACTIVATION (STD_OFF)
	• Configure 4 CAN controllers. • Set ResourceMMasterCore as CORE1. • Assign 3 controllers configured under ResourceMAllocation with ResourceMCoreID as CORE0.	#define CAN_17_MCMCAN_CORE0_ACTIVATION (STD_ON) #define CAN_17_MCMCAN_CORE1_ACTIVATION (STD_ON) #define CAN_17_MCMCAN_CORE2_ACTIVATION (STD_OFF) #define CAN_17_MCMCAN_CORE3_ACTIVATION (STD_OFF) #define CAN_17_MCMCAN_CORE4_ACTIVATION (STD_OFF) #define CAN_17_MCMCAN_CORE5_ACTIVATION (STD_OFF)

1.1.32 Macro: CAN_17_MCMCAN_BO_INTERRUPT_PROCESSING

Table 32 CAN_17_MCMCAN_BO_INTERRUPT_PROCESSING

Name	CAN_17_MCMCAN_BO_INTERRUPT_PROCESSING	
Description	Enables/Disables bus off processing through interrupt.	
Verification method	The macro is generated as STD_ON if atleast one of the CAN controllers configured has the container CanBusoffProcessing set as ‘INTERRUPT’ else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	• Configure 2 CAN controllers • Configure CAN controller 1 with CanBusoffProcessing set as ‘INTERRUPT’.	#define CAN_17_MCMCAN_BO_INTERRUPT_PROCESSING (STD_ON)

	- Configure 2 CAN controllers - Configure the 2 CAN controllers with CanBusoffProcessing set as 'POLLING'.	#define CAN_17_MCMCAN_BO_INTERRUPT_PROCESSING (STD_OFF)

1.1.33 Macro: CAN_17_MCMCAN_TX_INTERRUPT_PROCESSING

Table 33 CAN_17_MCMCAN_TX_INTERRUPT_PROCESSING

Name	CAN_17_MCMCAN_TX_INTERRUPT_PROCESSING	
Description	Enables/Disables CAN Tx processing through interrupt.	
Verification method	The macro is generated as STD_ON if atleast one of the CAN controllers configured has the container CanTxProcessing set as 'INTERRUPT' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	- Configure 2 CAN controllers - Configure CAN controller 1 with CanTxProcessing set as 'INTERRUPT'.	#define CAN_17_MCMCAN_TX_INTERRUPT_PROCESSING (STD_ON)
	- Configure 2 CAN controllers - Configure the 2 CAN controllers with CanTxProcessing set as 'POLLING'.	#define CAN_17_MCMCAN_TX_INTERRUPT_PROCESSING (STD_OFF)

1.1.34 Macro: CAN_17_MCMCAN_RX_INTERRUPT_PROCESSING

Table 34 CAN_17_MCMCAN_RX_INTERRUPT_PROCESSING

Name	CAN_17_MCMCAN_RX_INTERRUPT_PROCESSING	
Description	Enables/Disables CAN Rx processing through interrupt.	
Verification method	The macro is generated as STD_ON if atleast one of the CAN controllers configured has the container CanRxProcessing set as 'INTERRUPT' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	- Configure 2 CAN controllers - Configure CAN controller 1 with CanRxProcessing set as 'INTERRUPT'.	#define CAN_17_MCMCAN_RX_INTERRUPT_PROCESSING (STD_ON)
	- Configure 2 CAN controllers - Configure the 2 CAN controllers with CanRxProcessing set as 'POLLING'.	#define CAN_17_MCMCAN_RX_INTERRUPT_PROCESSING (STD_OFF)

1.1.35 Macro: CAN_17_MCMCAN_BO_POLLING_PROCESSING

Table 35 CAN_17_MCMCAN_BO_POLLING_PROCESSING

Name	CAN_17_MCMCAN_BO_POLLING_PROCESSING	
Description	Enables/Disables bus off processing through polling.	
Verification method	The macro is generated as STD_ON if atleast one of the CAN controllers configured has the container CanBusoffProcessing set as ‘POLLING’ else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	• Configure 2 CAN controllers • Configure CAN controller 1 with CanBusoffProcessing set as ‘POLLING’.	`#define CAN_17_MCMCAN_BO_POLLING_PROCESSING (STD_ON)`
	• Configure 2 CAN controllers • Configure the 2 CAN controllers with CanBusoffProcessing set as ‘INTERRUPT’.	`#define CAN_17_MCMCAN_BO_POLLING_PROCESSING (STD_OFF)`

1.1.36 Macro: CAN_17_MCMCAN_TX_POLLING_PROCESSING

Table 36 CAN_17_MCMCAN_TX_POLLING_PROCESSING

Name	CAN_17_MCMCAN_TX_POLLING_PROCESSING	
Description	Enables/Disables CAN Tx processing through polling.	
Verification method	The macro is generated as STD_ON if atleast one of the CAN controllers configured has the container CanTxProcessing set as ‘POLLING’ else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	• Configure 2 CAN controllers • Configure CAN controller 1 with CanTxProcessing set as ‘POLLING’.	`#define CAN_17_MCMCAN_TX_POLLING_PROCESSING (STD_ON)`
	• Configure 2 CAN controllers • Configure the 2 CAN controllers with CanTxProcessing set as ‘INTERRUPT’.	`#define CAN_17_MCMCAN_TX_POLLING_PROCESSING (STD_OFF)`

1.1.37 Macro: CAN_17_MCMCAN_RX_POLLING_PROCESSING

Table 37 CAN_17_MCMCAN_RX_POLLING_PROCESSING

Name	CAN_17_MCMCAN_RX_POLLING_PROCESSING
Description	Enables/Disables CAN Rx processing through polling.

Verification method	The macro is generated as STD_ON if atleast one of the CAN controllers configured has the container CanRxProcessing set as ‘POLLING’ else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	• Configure 2 CAN controllers • Configure CAN controller 1 with CanRxProcessing set as ‘POLLING’.	`#define CAN_17_MCMCAN_RX_POLLING_PROCESSING (STD_ON)`
	• Configure 2 CAN controllers • Configure the 2 CAN controllers with CanRxProcessing set as ‘INTERRUPT’.	`#define CAN_17_MCMCAN_RX_POLLING_PROCESSING (STD_OFF)`

1.1.38 Macro: CAN_17_MCMCAN_WU_POLLING_PROCESSING

Table 38 CAN_17_MCMCAN_WU_POLLING_PROCESSING

Name	CAN_17_MCMCAN_WU_POLLING_PROCESSING	
Description	Enables/Disables CAN Wake-up processing through polling.	
Verification method	The macro is generated as STD_ON if atleast one of the CAN controllers configured has the container CanWakeupProcessing set as ‘POLLING’ else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	• Configure 2 CAN controllers • Configure CAN controller 1 with CanWakeupProcessing set as ‘POLLING’.	`#define CAN_17_MCMCAN_WU_POLLING_PROCESSING (STD_ON)`
	• Configure 2 CAN controllers • Configure the 2 CAN controllers with CanWakeupProcessing set as ‘INTERRUPT’.	`#define CAN_17_MCMCAN_WU_POLLING_PROCESSING (STD_OFF)`

1.1.39 Macro: CAN_17_MCMCAN_RX_MIXED_PROCESSING

Table 39 CAN_17_MCMCAN_ RX_MIXED_PROCESSING

Name	CAN_17_MCMCAN_ RX_MIXED_PROCESSING	
Description	Enables/Disables CAN Rx processing when controller is configured as ‘MIXED’.	
Verification method	The macro is generated as STD_ON if atleast one of the CAN controllers configured has the container CanRxProcessing set as ‘MIXED’ else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**

	• Configure 2 CAN controllers • Configure CAN controller 1 with CanRxProcessing set as 'MIXED'.	`#define CAN_17_MCMCAN_RX_MIXED_PROCESSING (STD_ON)`
	• Configure 2 CAN controllers • Configure the 2 CAN controllers with CanRxProcessing set as 'INTERRUPT'.	`#define CAN_17_MCMCAN_RX_MIXED_PROCESSING (STD_OFF)`

1.1.40 Macro: CAN_17_MCMCAN_ TX_MIXED_PROCESSING

Table 40 CAN_17_MCMCAN_ TX_MIXED_PROCESSING

Name	CAN_17_MCMCAN_ TX_MIXED_PROCESSING	
Description	Enables/Disables CAN Tx processing when controller is configured as 'MIXED'.	
Verification method	The macro is generated as STD_ON if atleast one of the CAN controllers configured has the container CanTxProcessing set as 'MIXED' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	• Configure 2 CAN controllers • Configure CAN controller 1 with CanTxProcessing set as 'MIXED'.	`#define CAN_17_MCMCAN_TX_MIXED_PROCESSING (STD_ON)`
	• Configure 2 CAN controllers • Configure the 2 CAN controllers with CanTxProcessing set as 'INTERRUPT'.	`#define CAN_17_MCMCAN_TX_MIXED_PROCESSING (STD_OFF)`

1.1.41 Macro: Can_17_McmCanConf_CanController_<controller name>

Table 41 Can_17_McmCanConf_CanController_<controller name>

Name	Can_17_McmCanConf_CanController_<controller name>	
Description	The macro is the symbolic name generated for the configuration parameter 'CanConfigSet/CanController/CanControllerId'	
Verification method	The macro is generated as a numeric value which is configured in 'CanConfigSet/CanController/CanControllerId'. < controller name> is the name of the CAN controller's container name.	
Example(s)	**Action**	**Generated output**

	- Configure 2 CAN controllers.
	- Container for CAN controller ID 0 is named Int.
	- Container for CAN controller ID 1 is named Ext.
		`#define Can_17_McmCanConf_CanController_Int (0U)`
		`#define Can_17_McmCanConf_CanController_Ext (1U)`

1.1.42 Macro: Can_17_McmCanConf_CanHardwareObject_<hardware object name>

Table 42 Can_17_McmCanConf_CanHardwareObject_<hardware object name>

Name	Can_17_McmCanConf_CanHardwareObject_<hardware object name>	
Description	The macro is the symbolic name generated for the configuration parameter 'CanConfigSet/CanHardwareObject/CanObjectId'	
Verification method	The macro is generated as a numeric value which is configured in 'CanConfigSet/CanHardwareObject/CanObjectId'. < hardware object name> is the name of the CAN hardware object's container name.	
Example(s)	**Action**	**Generated output**

	- Configure 2 hardware objects.
	- Container for hardware object ID 0 is named VMSVoltage.
	- Container for hardware object ID 1 is named VMSCurrent.
		`#define Can_17_McmCanConf_CanHardwareObject_VMSVoltage (0U)`
		`#define Can_17_McmCanConf_CanHardwareObject_VMSCurrent (1U)`

1.1.43 Macro: Can_17_McmCanConf_CanIcomConfigIndex_Deactivate

Table 43 Can_17_McmCanConf_CanIcomConfigIndex_Deactivate

Name	Can_17_McmCanConf_CanIcomConfigIndex_Deactivate	
Description	The macro is the symbolic name generated for the deactivation of pretended network.	
Verification method	The macro is generated as the numeric value 0. *Note: This macro is generated only when container 'CanGeneral/CanPublicIcomSupport' is set to 'True' else this macro is not generated.*	
Example(s)	**Action**	**Generated output**
	Configure CanPublicIcomSupport as 'True'	`#define Can_17_McmCanConf_CanIcomConfigIndex_Deactivate (0U)`

1.1.44 Macro: Can_17_McmCanConf_CanIcomConfigIndex_<Icom name>

Table 44 Can_17_McmCanConf_CanIcomConfigIndex_<Icom name>

Name	Can_17_McmCanConf_CanIcomConfigIndex_<Icom name>	
Description	The macro is the symbolic name generated for the configuration parameter 'CanIcomConfig/CanIcomConfigId'	
Verification method	The macro is generated as a numeric value which is configured in 'CanIcomConfig/CanIcomConfigId'. < Icom name> is the name of the CAN Icom configuration container name. Note: *This macro is generated only when container 'CanGeneral/CanPublicIcomSupport' is set to 'True' else this macro is not generated.*	
Example(s)	**Action**	**Generated output**

	• Configure 2 Icom configurations.
	• Container for Icom config ID 0 is named McuWakeup.
	• Container for Icom config ID 1 is named SlaveWakeup.
		`#define Can_17_McmCanConf_CanIcomConfigIndex_McuWakeup (0U)`
		`#define Can_17_McmCanConf_CanIcomConfigIndex_SlaveWakeup (1U)`

1.1.45 Function Declaration: Can_17_McmCan_MainFunction_Write_<Period Index>

Table 45 Can_17_McmCan_MainFunction_Write_<Period Index>

Name	Can_17_McmCan_MainFunction_Write_<Period Index>	
Description	The external function declaration generated based on the configuration parameter 'CanMainFunctionRWPeriods' for the multi-period polling based write operation.	
Verification method	The number of external function declaration generated is based on the number of elements in the list configured in 'CanMainFunctionRWPeriods'. < Period Index > is the element number of the 'CanMainFunctionRWPeriods'. Note: *This external function declaration is generated only when atleast one CAN controller container 'CanTxProcessing' is set to 'Polling' and number of elements configured in list 'CanMainFunctionRWPeriods' is greater than 1 else this external function declaration is not generated.*	
Example(s)	**Action**	**Generated output**

	• Configure 5 read write periods.
	• Configure 3 CAN controllers.
	• Configure 1 Can controller with 'CanTxProcessing' set as 'Polling'.
		`extern void Can_17_McmCan_MainFunction_Write_0(void);`
		`extern void Can_17_McmCan_MainFunction_Write_1(void);`
		`extern void Can_17_McmCan_MainFunction_Write_2(void);`
		`extern void Can_17_McmCan_MainFunction_Write_3(void);`
		`extern void Can_17_McmCan_MainFunction_Write_4(void);`

1.1.46 Function Declaration: Can_17_McmCan_MainFunction_Read_<Period Index>

Table 46 Can_17_McmCan_MainFunction_Read_<Period Index>

Name	Can_17_McmCan_MainFunction_Read_<Period Index>	
Description	The external function declaration generated based on the configuration parameter 'CanMainFunctionRWPeriods' for the multi-period polling based read operation.	
Verification method	The number of external function declaration generated is based on the number of elements in the list configured in 'CanMainFunctionRWPeriods'. < Period Index > is the element number of the 'CanMainFunctionRWPeriods'. *Note: This external function declaration is generated only when atleast one CAN controller container 'CanRxProcessing' is set to 'Polling' and number of elements configured in list 'CanMainFunctionRWPeriods' is greater than 1 else this external function declaration is not generated.*	
Example(s)	**Action**	**Generated output**

	• Configure 5 read write periods.
	• Configure 3 CAN controllers.
	• Configure 1 Can controller with 'CanRxProcessing' set as 'Polling'.
		`extern void Can_17_McmCan_MainFunction_Read_0(void);`
		`extern void Can_17_McmCan_MainFunction_Read_1(void);`
		`extern void Can_17_McmCan_MainFunction_Read_2(void);`
		`extern void Can_17_McmCan_MainFunction_Read_3(void);`
		`extern void Can_17_McmCan_MainFunction_Read_4(void);`

1.2 File: Can_17_McmCan[_<variant>]_PBcfg.c

The generated source file contains all post-build configuration parameters. Post-build time configuration mechanism allows configurable functionality of CAN driver that is deployed as object code. The file is generated in 'src' folder.

1.2.1 Structure: Can_17_McmCan_Config[_<variant>]

Table 47 Can_17_McmCan_Config[_<variant>]

Name	Can_17_McmCan_Config[_<variant>]	
Type	Can_17_McmCan_ConfigType	
Description	Root configuration structure of CAN driver which will be used during initialization.	
Verification method	The generated structure is present in Can_17_McmCan[_<variant>]_PBcfg.c file. The <variant> indicates the name of the post-build variant. For a variant-aware configuration the structure name is appended with the variant name. For variant-unaware configuration <variant> is ignored.	
Example(s)	**Action**	**Generated output**

	• Configure 4 CAN controllers atleast 1 from all the 3 kernels
	• Allocate the 4 CAN controllers to Core0
	• Configure 60 hardware objects for the 2 CAN controllers with 44 of the hardware objects being of RECEIVE type.
	• CanPublicIcomSupport set as 'True'
	• variant-aware. Variant name is 'Petrol'
		const Can_17_McmCan_ConfigType \\
		  Can_17_McmCan_Config_Petrol =
		{
		  /******************** Core specific configuration set *********************/
		
		  /* Pointer to the Core specific CAN configuration set */
		  {
		    &Can_17_McmCan_kMcmCanConfigCore0_Petrol,
		    NULL_PTR,
		    NULL_PTR,
		    NULL_PTR,
		    NULL_PTR,
		    NULL_PTR
		  },
		
		  /****************** Global data shared amongst all cores ********************/
		
		  /* Number of Kernels configured */
		  3,
		  /* Number of Hrh configured */
		  44,
		  /* Pointer to CAN Kernel configuration */
		 &Can_17_McmCan_kMcmCanModuleConfig_Petrol[0],

	/* Pointer holding physical controller index data */ &Can_17_McmCan_kMcmCanPhyContIndexConfig_Petrol[0], /* Pointer holding logical controller index data */ &Can_17_McmCan_kMcmCanLogicContIndexConfig_Petrol[0], /* Pointer holding configured Hth index data */ &Can_17_McmCan_kMcmCanHthIndexConfig_Petrol[0], /* Pointer to the ICOM configurations set */ &Can_17_McmCan_kMcmCanIcomConfig_Petrol[0], /* Pointer to the ICOM Rx message configurations */ &Can_17_McmCan_kMcmCanIcomRxMsgConfig_Petrol[0], /* Pointer to the ICOM Rx message signal configurations */ &Can_17_McmCan_kMcmCanIcomRxMsgSignalConfig_Petrol[0] };
• Configure 4 CAN controllers atleast 1 from all the 3 kernels • Allocate the 4 CAN controllers to Core0 • Configure 40 hardware objects for the 2 CAN controllers all 40 of the hardware objects being of RECEIVE type. • CanPublicIcomSupport set as 'False'	const Can_17_McmCan_ConfigType \ Can_17_McmCan_Config= { /******************** Core specific configuration set *********************/ /* Pointer to the Core specific CAN configuration set */ { &Can_17_McmCan_kMcmCanConfigCore0, NULL_PTR, NULL_PTR, NULL_PTR, NULL_PTR, NULL_PTR }, /****************** Global data shared amongst all cores ********************/ /* Number of Kernels configured */ 3, /* Number of Hrh configured */ 40, /* Pointer to CAN Kernel configuration */

		&Can_17_McmCan_kMcmCanModuleConfig[0], /* Pointer holding physical controller index data */ &Can_17_McmCan_kMcmCanPhyContIndexConfig[0], /* Pointer holding logical controller index data */ &Can_17_McmCan_kMcmCanLogicContIndexConfig[0], /* Pointer holding configured Hth index data */ NULL_PTR };

1.2.1.1 Member: CanCoreConfigPtr[6]

Table 48 CanCoreConfigPtr[6]

Name	CanCoreConfigPtr[6]	
Type	Can_17_McmCan_CoreConfigType *	
Description	Array of core-specific configuration.	
Verification method	The generated structure member is present in the Can_17_McmCan_Config[_<variant>] structure. If a Core<x> is allocated at least one controller, then the element <x> shall be generated as '&Can_17_McmCan_kMcmCanConfigCore<x>[_<variant>]' else 'NULL_PTR' is generated. (x in range 0 to 5).	
Example(s)	**Action**	**Generated output**

	All the CAN controllers are allocated to Core 0 (variant-unaware)	{
		&Can_17_McmCan_kMcmCanConfigCore0,
		NULL_PTR,
		NULL_PTR,
		NULL_PTR,
		NULL_PTR,
		NULL_PTR
		}

	All the CAN controllers are allocated to Core 0 (variant-aware. Variant name is 'Petrol')	{
		&Can_17_McmCan_kMcmCanConfigCore0_Petrol,
		NULL_PTR,
		NULL_PTR,
		NULL_PTR,
		NULL_PTR,
		NULL_PTR
		}

	All the CAN controllers are split between all cores	{ NULL_PTR,

	except Core 0. (variant-unaware)	`& Can_17_McmCan_kMcmCanConfigCore1,`
		`& Can_17_McmCan_kMcmCanConfigCore2,`
		`& Can_17_McmCan_kMcmCanConfigCore3,`
		`& Can_17_McmCan_kMcmCanConfigCore4,`
		`& Can_17_McmCan_kMcmCanConfigCore5`
		`}`

1.2.1.2 Member: CanNoOfKernel

Table 49 CanNoOfKernel

<table>
<tr><td><b>Name</b></td><td colspan="2">CanNoOfKernel</td></tr>
<tr><td><b>Type</b></td><td colspan="2">Uint8</td></tr>
<tr><td><b>Description</b></td><td colspan="2">Indicates the total number of kernels configured
<i>Note: Kernel is a CAN hardware unit consisting of 4 nodes (controllers).</i>
<i>Example:</i>
<i>Kernel 0 shall contain 4 nodes, controllers 0, 1, 2, 3.</i>
<i>Kernel 1 shall contain 4 nodes, controllers 4, 5, 6, 7</i>
<i>Kernel 2 shall contain 4 nodes, controllers 8,9,10,11</i>
<i>Note: The number of Kernels and nodes per kernel are device dependent.</i></td></tr>
<tr><td><b>Verification method</b></td><td colspan="2">The generated numeric value is the total number of kernels configured by the user. This structure element is generated by analysing the CanControllerBaseAddress to bring out which kernel the CanController configured belongs to.</td></tr>
<tr><td><b>Example(s)</b></td><td><b>Action</b></td><td><b>Generated output</b></td></tr>
<tr><td></td><td>Configure 6 CAN controller. Configure 3 CAN controller of Kernel0 and 3 CAN controllers of kernel1.</td><td>2</td></tr>
<tr><td></td><td>Configure 4 CAN controllers that belong to Kernel0.</td><td>1</td></tr>
</table>

1.2.1.3 Member: CanNoOfHrh

Table 50 CanNoOfHrh

Name	CanNoOfHrh
Type	Can_HwHandleType
Description	Indicates the total number of receive hardware objects configured

Verification method	The generated numeric value is the total number of receive hardware objects configured by the user. This structure element is generated by counting the number of CanHardwareObject with CanObjectType set as 'RECEIVE'.	
Example(s)	**Action**	**Generated output**
	Configure 4 Hardware Objects. Configure 2 hardware objects with CanObjectType as RECEIVE and the other 2 as TRANSMIT	2
	Configure 4 Hardware Objects. Configure all 4 hardware objects with CanObjectType as TRANSMIT	0

1.2.1.4 Member: CanTotalHwObj

Table 51 CanTotalHwObj

Name	CanTotalHwObj	
Type	Can_HwHandleType	
Description	The generated structure member is present in the Can_17_McmCan_Config[_<variant>] structure. Indicates the number of hardware objects (includes Tx and Rx) configured in a ConfigSet.	
Verification method	The generated numeric value is the total number of hardware objects configured by the user. This structure element is generated by counting the number of CanHardwareObject.	
Example(s)	**Action**	**Generated output**
	Configure 3 Hardware Objects with CanObjectType as RECEIVE for all.	3
	Configure 2 hardware objects with CanObjectType as RECEIVE and another 2 hardware objects with CanObjectType as TRANSMIT	4

1.2.1.5 Member: CanTriggerTransmitEnable

Table 52 CanTriggerTransmitEnable

Name	CanTriggerTransmitEnable
Type	boolean
Description	The generated structure member is present in the Can_17_McmCan_Config[_<variant>] structure. Indicates if trigger transmit is enabled for the configuration for any of the hardware object
Verification method	The structure member is generated as a 'TRUE' if CanTriggerTransmitEnable is checked for at least one CanHardwareObject else it is generated as 'FALSE'.

Example(s)	**Action**	**Generated output**
	Configure 4 Hardware Objects. Configure 1 hardware objects with CanTriggerTransmitEnable checked.	`TRUE`
	Configure 4 Hardware Objects. Configure all 4 hardware objects with CanTriggerTransmitEnable unchecked	`FALSE`

1.2.1.6 Member: CanMCMModuleConfigPtr

Table 53 CanMCMModuleConfigPtr

Name	CanMCMModuleConfigPtr	
Type	Can_17_McmCan_McmModuleConfigType	
Description	Pointer to kernel specifc configurations	
Verification method	The generated structure member is present in the Can_17_McmCan_Config[_<variant>] structure. The element shall be generated as '&Can_17_McmCan_kMcmCanModuleConfig>[_<variant>][0]', pointing to the first element of the kernel specific configuration array.	
Example(s)	**Action**	**Generated output**
	CAN configured with basic generation package	`&Can_17_McmCan_kMcmCanModuleConfig[0]`
	CAN configured with basic generation package (variant-aware. Variant name is 'Petrol')	`&Can_17_McmCan_kMcmCanModuleConfig_Petrol[0]`

1.2.1.7 Member: CanPhyControllerIndexPtr

Table 54 CanPhyControllerIndexPtr

Name	CanPhyControllerIndexPtr	
Type	Can_17_McmCan_PhyControllerIndexType	
Description	Pointer to CAN hardware (example, kernel1 node 3 will be indexed at '4*KernelId(1) + NodeId(3)' which is 7) controller Id indexing based CAN controller mapping array	
Verification method	The generated structure member is present in the Can_17_McmCan_Config[_<variant>] structure. The element shall be generated as '&Can_17_McmCan_kMcmCanPhyContIndexConfig[_<variant>][0]', pointing to the CAN hardware based controller Id 0's core specific and logical CAN controller Id configuration array.	
Example(s)	**Action**	**Generated output**

	CAN configured with basic generation package	`& Can_17_McmCan_kMcmCanPhyContIndexConfig[0]`

	CAN configured with basic generation package (variant-aware. Variant name is 'Petrol')	`&`
		`Can_17_McmCan_kMcmCanPhyContIndexConfig_Petrol[0]`

1.2.1.8 Member: CanLogicalControllerIndexPtr

Table 55 CanLogicalControllerIndexPtr

Name	CanLogicalControllerIndexPtr	
Type	Can_17_McmCan_LogicalControllerIndexType	
Description	Pointer to logical (controller Id configured) controller Id indexing based CAN controller mapping array	
Verification method	The generated structure member is present in the Can_17_McmCan_Config[_<variant>] structure. The element shall be generated as '&Can_17_McmCan_kMcmCanLogicContIndexConfig[_<variant>][0]', pointing to the logical controller Id 0's core specific and CAN hardware controller Id configuration array.	
Example(s)	**Action**	**Generated output**
	CAN configured with basic generation package	`&Can_17_McmCan_kMcmCanLogicContIndexConfig[0]`
	CAN configured with basic generation package (variant-aware. Variant name is 'Petrol')	`&Can_17_McmCan_kMcmCanLogicContIndexConfig_Petrol[0]`

1.2.1.9 Member: CanHthIndexPtr

Table 56 CanHthIndexPtr

Name	CanHthIndexPtr
Type	Can_17_McmCan_HthIndexType
Description	Pointer to logical (Hth object Id configured) hardware object indexing based core specific Hth mapping array
Verification method	The generated structure member is present in the Can_17_McmCan_Config[_<variant>] structure. The element shall be generated as '&Can_17_McmCan_kMcmCanHthIndexConfig[_<variant>][0]', pointing to the logical Hth object Id 0's core specific based Hth Id configuration array. *Note: This structure member is generated as '&Can_17_McmCan_kMcmCanHthIndexConfig[_<variant>][0]' only when atleast one*

	CAN hardware object is of 'CanObjectType' is set to 'RECEIVE' else this structure member is generated as a NULL_PTR.	
Example(s)	**Action**	**Generated output**
	CAN configured with hardware objects of type 'TRANSMIT' and basic generation package (variant-aware. Variant name is 'Petrol')	`&Can_17_McmCan_kMcmCanHthIndexConfig_Petrol[0]`
	CAN configured with hardware objects only of type 'RECEIVE'.	`NULL_PTR`

1.2.1.10 Member: CanIcomConfigPtr

Table 57 CanIcomConfigPtr

Name	CanIcomConfigPtr	
Type	Can_17_McmCan_IcomConfigType	
Description	Pointer to array of structures holding information on the different pretended networking configurations.	
Verification method	The generated structure member is present in the Can_17_McmCan_Config[_<variant>] structure. The element shall be generated as '&Can_17_McmCan_kMcmCanIcomConfig[_<variant>][0]', pointing to the first element in the array for different pretended networking configurations. *Note: This structure member is generated as '&Can_17_McmCan_kMcmCanIcomConfig[_<variant>][0]' only when atleast one CanIcom element is configured and CanPublicIcomSupport is 'True' else this structure member is not generated.*	
Example(s)	**Action**	**Generated output**
	CAN configured with CanIcom configured and CanPublicIcomSupport set as 'True'	`&Can_17_McmCan_kMcmCanIcomConfig[0]`
	CAN configured with CanIcom configured and CanPublicIcomSupport set as 'True' (variant-aware. Variant name is 'Petrol')	`&Can_17_McmCan_kMcmCanIcomConfig_Petrol[0]`

1.2.1.11 Member: CanIcomMsgConfigPtr

Table 58 CanIcomMsgConfigPtr

Name	CanIcomMsgConfigPtr

Type	Can_17_McmCan_IcomRxMsgConfigType	
Description	Pointer to array of structures holding information on the different messages configured for pretended networking.	
Verification method	The generated structure member is present in the Can_17_McmCan_Config[_<variant>] structure. The element shall be generated as '&Can_17_McmCan_kMcmCanIcomRxMsgConfig[_<variant>][0]', pointing to the first element in the array for different messages configured for pretended networking. *Note: This structure member is generated as '&Can_17_McmCan_kMcmCanIcomRxMsgConfig[_<variant>][0]' only when atleast one CanIcom element is configured and CanPublicIcomSupport is 'True' else this structure member is not generated.*	
Example(s)	**Action**	**Generated output**
	CAN configured with CanIcom configured and CanPublicIcomSupport set as 'True'	`&Can_17_McmCan_kMcmCanIcomRxMsgConfig[0]`
	CAN configured with CanIcom configured and CanPublicIcomSupport set as 'True' (variant-aware. Variant name is 'Petrol')	`&Can_17_McmCan_kMcmCanIcomRxMsgConfig_Petrol[0]`

1.2.1.12 Member: CanIcomRxSignalConfigPtr

Table 59 CanIcomRxSignalConfigPtr

Name	CanIcomRxSignalConfigPtr	
Type	Can_17_McmCan_IcomRxMsgSignalConfigType	
Description	Pointer to array of structures holding information on the different signals configured for the messages in pretended networking.	
Verification method	The generated structure member is present in the Can_17_McmCan_Config[_<variant>] structure. The element shall be generated as '&Can_17_McmCan_kMcmCanIcomRxMsgSignalConfig[_<variant>][0]', pointing to the first element in the array for different signals for the messages in pretended networking. *Note: This structure member is generated as '&Can_17_McmCan_kMcmCanIcomRxMsgSignalConfig[_<variant>][0]' only when atleast one CanIcom element with atleast one Icom signal for the Icom message is configured and CanPublicIcomSupport is 'True' else this structure member is not generated.*	
	Action	**Generated output**

Example(s)	CAN configured with CanIcom configured with atleast 1 signal configured for a message configuration and CanPublicIcomSupport set as 'True'	`&Can_17_McmCan_kMcmCanIcomRxMsgSignalConfig[0]`
	CAN configured with CanIcom configured with atleast 1 signal configured for a message configuration and CanPublicIcomSupport set as 'True' (variant-aware. Variant name is 'Petrol')	`&Can_17_McmCan_kMcmCanIcomRxMsgSignalConfig_Petrol[0]`

1.2.1.13 Member: CanLPduRxCalloutFuncPtr

Table 60 CanLPduRxCalloutFuncPtr

Name	CanLPduRxCalloutFuncPtr	
Type	Can_17_McmCan_LPduRxCalloutFnPtrType	
Description	Pointer to L-PDU Callout function. The name is configured by the user	
Verification method	The generated structure member is present in the Can_17_McmCan_Config[_<variant>] structure. The member is generated if an element is added in CanLPduReceiveCalloutFunction else it is not generated.	
Example(s)	**Action**	**Generated output**
	Add an element 'Appl_LPduRxCalloutFunction' in CanLPduReceiveCalloutFunction	`(Can_17_McmCan_LPduRxCalloutFnPtrType) Appl_LPduRxCalloutFunction`

1.2.2 Structure: Can_17_McmCan_kMcmCanConfigCore<x>[_<variant>]

Table 61 Can_17_McmCan_kMcmCanConfigCore<x>[_<variant>]

Name	Can_17_McmCan_kMcmCanConfigCore<x>[_<variant>]
Type	Can_17_McmCan_CoreConfigType
Description	Configuration structure of CAN driver for Core <x> which will be referenced in root configuration structure. (x ranges from 0 to 5)
Verification method	The generated file has this structure if at least one controller is assigned to Core <x>. <Variant> indicates the name of the post-build variant. For a variant aware configuration the structure name is appended with the variant name. For variant unaware configuration <variant> is ignored.

Example(s)	Action	Generated output

	Configure 2 controllers to Core5	static const Can_17_McmCan_CoreConfigType \
		  Can_17_McmCan_kMcmCanConfigCore5=
		{
		    /* Number of controllers configured for the core */
		    2,
		    /* Array of all the controllers configured */
		    &Can_17_McmCan_kControllerIndexingCore5[0],
		    /* Pointer to CAN controller configuration settings */
		    &Can_17_McmCan_kControllerConfigCore5[0],
		    /* Pointer to Message RAM configuration settings */
		&Can_17_McmCan_kControllerMsgRAMMapConfigCore5[0],
		    /* Pointer to CAN Controller Handling of Events : Interrupt/Polling */
		    &Can_17_McmCan_kEventHandlingConfigCore5[0],
		    /* Pointer to Baudrate configuration settings */
		    &Can_17_McmCan_kBaudrateConfigCore5[0],
		    /* Pointer to FDBaudrate configuration settings */
		    &Can_17_McmCan_kFDBaudrateConfigCore0[0],
		/* Pointer to CAN Controller <-> Tx Hardware Objects Mapping */
		    &Can_17_McmCan_kTxHwObjectConfigCore5[0],
		    /* Pointer to CAN Controller <-> Rx Hardware Objects Mapping for Standard
		       messages */
		    &Can_17_McmCan_kSIDFilterConfigCore5[0],
		    /* Pointer to CAN Controller <-> Rx Hardware Objects Mapping for Extended
		    messages */
		    &Can_17_McmCan_kXIDFilterConfigCore5[0],
		    /* Transmit Period to core specific period mapping*/
		    &Can_17_McmCan_kHthPeriodIndexCore5[0],
		    /* Pointer to CAN Controller <-> Tx Hardware Objects Mapping for Multiple
		    period */

	`&Can_17_McmCan_kHthMaskObjectConfigCore5[0],`
	`/* Pointer to CAN Controller <-> Tx Hardware Index Mapping for Multiple`
	`period */`
	`&Can_17_McmCan_kPeriodHthMaskConfigCore5[0],`
	`/* Recieve Period to core specific period mapping*/`
	`&Can_17_McmCan_kHrhPeriodIndexCore5[0],`
	`/* Pointer to CAN Controller <-> Rx Hardware Objects Mapping for Multiple`
	`period */`
	`&Can_17_McmCan_kHrhMaskObjectConfigCore5[0],`
	`/* Pointer to CAN Controller <-> Rx Hardware Index Mapping for Multiple`
	`period */`
	`&Can_17_McmCan_kPeriodHrhMaskConfigCore5[0]`
	`};`

1.2.2.1 Member: CanCoreContCnt

Table 62 CanCoreContCnt

Name	CanCoreContCnt	
Type	uint8	
Description	The total number of controllers allocated to current core.	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanConfigCore<x>[_<variant>] structure. The structure member is generated as a numeric value representing the total number of configured controllers assigned to current core<x>.	
Example(s)	**Action**	**Generated output**
	Configure 5 controllers assign 3 controllers to core 3 and 2 controllers to core 4. The CanCoreContCnt element in core 3	`3U`
	Configure 5 controllers assign 3 controllers to core 3 and 2 controllers to core 4. The CanCoreContCnt element in core 4	`2U`

1.2.2.2 Member: CanControllerIndexingPtr

Table 63 CanControllerIndexingPtr

Name	CanControllerIndexingPtr	
Type	Can_17_McmCan_ControllerIndexType*	
Description	Pointer to the base of array which stores the mapping of the configured controller Id for the controllers configured to Core<x>.	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanConfigCore<x>[_<variant>] structure. The structure member is generated as the pointer to the core specific Can_17_McmCan_kControllerIndexingCore<x>[_<variant>].	
Example(s)	**Action**	**Generated output**
	Configure atleast 1 controller to core 1	`&Can_17_McmCan_kControllerIndexingCore1[0],`

1.2.2.3 Member: CanControllerConfigPtr

Table 64 CanControllerConfigPtr

Name	CanControllerConfigPtr	
Type	Can_17_McmCan_ControllerConfigType*	
Description	Pointer to the base of array which stores the controller configuration for the controllers configured to Core<x>.	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanConfigCore<x>[_<variant>] structure. The structure member is generated as the pointer to the core specific Can_17_McmCan_kControllerConfigCore<x>[_<variant>].	
Example(s)	**Action**	**Generated output**
	Configure atleast 1 controller to core 5	`&Can_17_McmCan_kControllerConfigCore5[0]`

1.2.2.4 Member: CanControllerMsgRAMMapConfigPtr

Table 65 CanControllerMsgRAMMapConfigPtr

Name	CanControllerMsgRAMMapConfigPtr	
Type	Can_17_McmCan_ControllerMsgRAMConfigType*	
Description	Pointer to the base of array which stores the RAM configuration for the controllers configured to Core<x>.	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanConfigCore<x>[_<variant>] structure. The structure member is generated as the pointer to the core specific Can_17_McmCan_kControllerMsgRAMMapConfigCore<x>[_<variant>].	
Example(s)	**Action**	**Generated output**
	Configure atleast 1 controller to core 3	`&Can_17_McmCan_kControllerMsgRAMMapConfigCore3[0],`

1.2.2.5 Member: CanEventHandlingConfigPtr

Table 66 CanEventHandlingConfigPtr

<table>
<tr><td><b>Name</b></td><td colspan="2">CanEventHandlingConfigPtr</td></tr>
<tr><td><b>Type</b></td><td colspan="2">Can_17_McmCan_EventHandlingType *</td></tr>
<tr><td><b>Description</b></td><td colspan="2">Pointer to the base of array which stores the event handling configuration for the controllers configured to Core<x>.</td></tr>
<tr><td><b>Verification method</b></td><td colspan="2">The generated structure member is present in Can_17_McmCan_kMcmCanConfigCore<x>[_<variant>] structure. The structure member is generated as the pointer to the core specific Can_17_McmCan_kEventHandlingConfigCore<x>[_<variant>].</td></tr>
<tr><td><b>Example(s)</b></td><td><b>Action</b></td><td><b>Generated output</b></td></tr>
<tr><td></td><td>Configure atleast 1 controller to core 2</td><td><code>&Can_17_McmCan_kEventHandlingConfigCore2[0],</code></td></tr>
</table>

1.2.2.6 Member: CanBaudrateConfigPtr

Table 67 CanBaudrateConfigPtr

<table>
<tr><td><b>Name</b></td><td colspan="2">CanBaudrateConfigPtr</td></tr>
<tr><td><b>Type</b></td><td colspan="2">Can_17_McmCan_ControllerBaudrateConfigType*</td></tr>
<tr><td><b>Description</b></td><td colspan="2">Pointer to the base of array which stores the baudrate configuration for the controllers configured to Core<x>.</td></tr>
<tr><td><b>Verification method</b></td><td colspan="2">The generated structure member is present in Can_17_McmCan_kMcmCanConfigCore<x>[_<variant>] structure. The structure member is generated as the pointer to the core specific Can_17_McmCan_kBaudrateConfigCore<x>[_<variant>].</td></tr>
<tr><td><b>Example(s)</b></td><td><b>Action</b></td><td><b>Generated output</b></td></tr>
<tr><td></td><td>Configure atleast 1 controller with baudrate configured and allocated to core 0</td><td><code>&Can_17_McmCan_kBaudrateConfigCore0[0],</code></td></tr>
</table>

1.2.2.7 Member: CanFDConfigParamPtr

Table 68 CanFDConfigParamPtr

Name	CanFDConfigParamPtr
Type	Can_17_McmCan_ControllerFDBaudrateConfigType*
Description	Pointer to the base of array which stores the CAN FD baudrate configuration for the controllers configured to Core<x>.
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanConfigCore<x>[_<variant>] structure. The structure member is generated as the pointer to the core specific Can_17_McmCan_kFDBaudrateConfigCore<x>[_<variant>]. This structure element is generated only when atleast 1 CAN FD configuration is present in the CAN driver else this element is not generated.

	If atleast one CAN FD configuration is present in the CAN driver but not in the controllers associated with core<x>, then the element shall be generated with value'NULL_PTR'.	
Example(s)	**Action**	**Generated output**
	Configure atleast 2 controller with baudrate configured and having atleast one CAN FD baudrate configured to core0 and no CAN FD baudrate configured to controllers allocated to core5 Pointer generated for core0	`&Can_17_McmCan_kFDBaudrateConfigCore0[0],`
	Configure atleast 2 controller with baudrate configured and having atleast one CAN FD baudrate configured to core0 and no CAN FD baudrate configured to controllers allocated to core5 Pointer generated for core5	`NULL_PTR,`

1.2.2.8 Member: CanTxHwObjectConfigPtr

Table 69 CanTxHwObjectConfigPtr

Name	CanTxHwObjectConfigPtr	
Type	Can_17_McmCan_TxHwObjectConfigType*	
Description	Pointer to the base of array which stores the transmit hardware object configuration for the controllers configured to Core<x>.	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanConfigCore<x>[_<variant>] structure. The structure member is generated as the pointer to the core specific Can_17_McmCan_kTxHwObjectConfigCore<x>[_<variant>]. This pointer generated only when atleast 1 transmit hardware object is configured for the controllers in the current core<x> else the element is generated as a NULL_PTR.	
Example(s)	**Action**	**Generated output**
	Configure atleast 1 controller with atleast 1 transmit hardware object configured and allocated to core 0	`&Can_17_McmCan_kTxHwObjectConfigCore4[0],`
	Configure atleast 1 controller with no transmit	`NULL_PTR`

	hardware object configured and allocated to core 0	

1.2.2.9 Member: CanSIDFilterConfigPtr

Table 70 CanSIDFilterConfigPtr

Name	CanSIDFilterConfigPtr	
Type	Can_17_McmCan_SIDFilterConfigType*	
Description	Pointer to the base of array which stores the standard/mixed Id receive hardware object configuration for the controllers configured to Core<x>.	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanConfigCore<x>[_<variant>] structure. The structure member is generated as the pointer to the core specific Can_17_McmCan_kSIDFilterConfigCore<x>[_<variant>]. This pointer structure is generated only when atleast 1 standard/Mixed Id type receive hardware object configuration is present in the controllers associated with this core<x> else the element is generated as a NULL_PTR.	
Example(s)	**Action**	**Generated output**
	Configure atleast 1 controller with atleast 1 standard Id type recieve hardware object configured and allocated to core 2	`&Can_17_McmCan_kSIDFilterConfigCore2[0],`
	Configure atleast 1 controller with no standard/mixed Id type recieve hardware object configured and allocated to core 2	`NULL_PTR`

1.2.2.10 Member: CanXIDFilterConfigPtr

Table 71 CanXIDFilterConfigPtr

Name	CanXIDFilterConfigPtr	
Type	Can_17_McmCan_XIDFilterConfigType*	
Description	Pointer to the base of array which stores the extended/mixed Id receive hardware object configuration for the controllers configured to Core<x>.	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanConfigCore<x>[_<variant>] structure. The structure member is generated as the pointer to the core specific Can_17_McmCan_kXIDFilterConfigCore<x>[_<variant>]. This pointer structure is generated only when atleast 1 extended/Mixed Id type receive hardware object configuration is present in the controllers associated with this core<x> else the element is generated as a NULL_PTR.	
Example(s)	**Action**	**Generated output**

	Configure atleast 1 controller with atleast 1 extended Id type recieve hardware object configured and allocated to core 2	`&Can_17_McmCan_kXIDFilterConfigCore2[0],`
	Configure atleast 1 controller with no extended/mixed Id type recieve hardware object configured and allocated to core 2	`NULL_PTR`

1.2.2.11 Member: CanHthPeriodIndexPtr

Table 72 CanHthPeriodIndexPtr

Name	CanHthPeriodIndexPtr	
Type	Can_17_McmCan_HthPeriodIndexType*	
Description	Pointer to the base of array which stores the transmit hardware object configuration indexing for multi-period polling configuration for the controllers configured to Core<x>.	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanConfigCore<x>[_<variant>] structure. The structure member is generated as the pointer to the core specific Can_17_McmCan_kHthPeriodIndexCore<x>[_<variant>]. This structure is generated only when atleast 1 transmit hardware object and having greater than 1 read-write periods configured with the same controller having transmit process handling as polling else the structure element is not generated. In a multi-period configuration if the configuration is present with atleast 1 transmit object present and with transmit processing as polling in one of the controllers associated with this core<x> else the element is generated as a NULL_PTR.	
Example(s)	**Action**	**Generated output**
	Configure atlease 1 transmit hardware object for the controller associated with core 0 with CanTxProcessing as POLLING and 4 periods configured.	`&Can_17_McmCan_kHthPeriodIndexCore0[0],`
	Configure no transmit hardware object for the controller associated with core 0 with CanTxProcessing as POLLING and 4 periods configured.	`NULL_PTR`

1.2.2.12 Member: CanHthMaskObjectConfigPtr

Table 73 CanHthMaskObjectConfigPtr

Name	CanHthMaskObjectConfigPtr	
Type	Can_17_McmCan_HthMaskObjectConfigType*	
Description	Pointer to the base of array which stores the transmit hardware object mask configuration for multi-period polling configuration for the controllers configured to Core<x>.	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanConfigCore<x>[_<variant>] structure. The structure member is generated as the pointer to the core specific Can_17_McmCan_kHthMaskObjectConfigCore<x>[_<variant>]. This structure is generated only when atleast 1 transmit hardware object and having greater than 1 read-write periods configured with the same controller having transmit process handling as polling else the structure element is not generated. In a multi-period configuration if the configuration is present with atleast 1 transmit object present and with transmit processing as polling in one of the controllers associated with this core<x> else the element is generated as a NULL_PTR.	
Example(s)	**Action**	**Generated output**
	Configure atlease 1 transmit hardware object for the controller associated with core 0 with CanTxProcessing as POLLING and 4 periods configured.	`&Can_17_McmCan_kHthMaskObjectConfigCore0[0],`
	Configure no transmit hardware object for the controller associated with core 0 with CanTxProcessing as POLLING and 4 periods configured.	`NULL_PTR`

1.2.2.13 Member: CanPeriodHthMaskConfigPtr

Table 74 CanPeriodHthMaskConfigPtr

Name	CanPeriodHthMaskConfigPtr
Type	Can_17_McmCan_PeriodHthMaskConfigType*
Description	Pointer to the base of array which stores the transmit hardware object period specific mask index configuration for multi-period polling configuration for the controllers configured to Core<x>.
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanConfigCore<x>[_<variant>] structure. The structure member is generated as the pointer to the core specific Can_17_McmCan_kPeriodHthMaskConfigCore<x>[_<variant>].

	This structure is generated only when atleast 1 transmit hardware object and having greater than 1 read-write periods configured with the same controller having transmit process handling as polling else the structure element is not generated. In a multi-period configuration if the configuration is present with atleast 1 transmit object present and with transmit processing as polling in one of the controllers associated with this core<x> else the element is generated as a NULL_PTR.	
Example(s)	**Action**	**Generated output**
	Configure atlease 1 transmit hardware object for the controller associated with core 0 with CanTxProcessing as POLLING and 4 periods configured.	`&Can_17_McmCan_kPeriodHthMaskConfigCore0[0],`
	Configure no transmit hardware object for the controller associated with core 0 with CanTxProcessing as POLLING and 4 periods configured.	`NULL_PTR`

1.2.2.14 Member: CanHrhPeriodIndexPtr

Table 75 CanHrhPeriodIndexPtr

Name	CanHrhPeriodIndexPtr	
Type	Can_17_McmCan_HrhPeriodIndexType*	
Description	Pointer to the base of array which stores the receive hardware object configuration indexing for multi-period polling configuration for the controllers configured to Core<x>.	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanConfigCore<x>[_<variant>] structure. The structure member is generated as the pointer to the core specific Can_17_McmCan_kHrhPeriodIndexCore<x>[_<variant>]. This structure is generated only when atleast 1 receive hardware object and having greater than 1 read-write periods configured with the same controller having transmit process handling as polling else the structure element is not generated. In a multi-period configuration if the configuration is present with atleast 1 receive object present and with transmit processing as polling in one of the controllers associated with this core<x> else the element is generated as a NULL_PTR.	
Example(s)	**Action**	**Generated output**
	Configure atlease 1 Receive hardware object for the controller associated with core 0 with CanRxProcessing	`&Can_17_McmCan_kHrhPeriodIndexCore0[0],`

	as POLLING and 4 periods configured.	
	Configure no receive hardware object for the controller associated with core 0 with CanRxProcessing as POLLING and 4 periods configured.	`NULL_PTR`

1.2.2.15 Member: CanHrhMaskObjectConfigPtr

Table 76 CanHrhMaskObjectConfigPtr

Name	CanHrhMaskObjectConfigPtr	
Type	Can_17_McmCan_HrhMaskObjectConfigType*	
Description	Pointer to the base of array which stores the receive hardware object mask configuration for multi-period polling configuration for the controllers configured to Core<x>.	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanConfigCore<x>[_<variant>] structure. The structure member is generated as the pointer to the core specific Can_17_McmCan_kHrhMaskObjectConfigCore<x>[_<variant>]. This structure is generated only when atleast 1 receive hardware object and having greater than 1 read-write periods configured with the same controller having transmit process handling as polling else the structure element is not generated. In a multi-period configuration if the configuration is present with atleast 1 receive object present and with transmit processing as polling in one of the controllers associated with this core<x> else the element is generated as a NULL_PTR.	
Example(s)	**Action**	**Generated output**
	Configure atlease 1 Receive hardware object for the controller associated with core 0 with CanRxProcessing as POLLING and 4 periods configured.	`&Can_17_McmCan_kHrhMaskObjectConfigCore0[0],`
	Configure no receive hardware object for the controller associated with core 0 with CanRxProcessing as POLLING and 4 periods configured.	`NULL_PTR`

1.2.2.16 Member: CanPeriodHrhMaskConfigPtr

Table 77 CanPeriodHrhMaskConfigPtr

Name	CanPeriodHrhMaskConfigPtr

Type	Can_17_McmCan_PeriodHrhMaskConfigType*	
Description	Pointer to the base of array which stores the receive hardware object period specific mask index configuration for multi-period polling configuration for the controllers configured to Core<x>.	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanConfigCore<x>[_<variant>] structure. The structure member is generated as the pointer to the core specific Can_17_McmCan_kPeriodHrhMaskConfigCore<x>[_<variant>]. This structure is generated only when atleast 1 transmit hardware object and having greater than 1 read-write periods configured with the same controller having transmit process handling as polling else the structure element is not generated. In a multi-period configuration if the configuration is present with atleast 1 transmit object present and with transmit processing as polling in one of the controllers associated with this core<x> else the element is generated as a NULL_PTR.	
Example(s)	**Action**	**Generated output**
	Configure atlease 1 Receive hardware object for the controller associated with core 0 with CanRxProcessing as POLLING and 4 periods configured.	`&Can_17_McmCan_kPeriodHrhMaskConfigCore0[0],`
	Configure no receive hardware object for the controller associated with core 0 with CanRxProcessing as POLLING and 4 periods configured.	`NULL_PTR`

1.2.3 Array: Can_17_McmCan_kControllerIndexingCore<x>[_<variant>][CAN_17_MCMCAN_CORE<x>_NOOF_CONTROLLER]

Table 78 Can_17_McmCan_kControllerIndexingCore<x>[_<variant>][CAN_17_MCMCAN_CORE<x>_NOOF_CONTROLLER]

Name	Can_17_McmCan_kControllerIndexingCore<x>[_<variant>][CAN_17_MCMCAN_CORE<x>_NOOF_CONTROLLER]
Type	Can_17_McmCan_ControllerIndexType
Description	A configuration array to map the core specific controller Id as index to its configured controller Id.
Verification method	The generated file has this array if at least one controller is assigned to Core <x>. <Variant> indicates the name of the post-build variant. For a variant aware configuration the array name is appended with the variant name. For variant unaware configuration <variant> is ignored. This

	array generates the value in the container 'CanControllerId' for the given core specifc controller indexed controller.	
Example(s)	**Action**	**Generated output**

	Configure 3 controllers of Ids 0,1,2. Allocate controller 0 and 1 to core 0 and 2 to core 5.
	The array of core 0
		`static const`
		`Can_17_McmCan_ControllerIndexType \`
		`  Can_17_McmCan_kControllerIndexingCore0[2]`
		`=`
		`{0,2};`

	Configure 3 controllers of Ids 0,1,2. Allocate controller 0 and 1 to core 0 and 2 to core 5.
	The array of core 5
		`static const`
		`Can_17_McmCan_ControllerIndexType \`
		`  Can_17_McmCan_kControllerIndexingCore5[1]`
		`=`
		`{1};`

1.2.4 Structure: Can_17_McmCan_kControllerConfigCore<x>[_<variant>][CAN_17_MCMCAN_CORE<x>_NOOF_CONTROLLER]

Table 79 Can_17_McmCan_kControllerConfigCore<x>[_<variant>][CAN_17_MCMCAN_CORE<x>_NOOF_CONTROLLER]

Name	Can_17_McmCan_kControllerConfigCore<x>[_<variant>][CAN_17_MCMCAN_CORE<x>_NOOF_CONTROLLER]	
Type	Can_17_McmCan_ControllerConfigType	
Description	Configuration structure holding the controller configuration details for all the CAN controllers belonging to Core <x> which will be referenced in core specific configuration structure. (x ranges from 0 to 5)	
Verification method	The generated file has this structure for the different controllers configured for each core. <Variant> indicates the name of the post-build variant. For a variant aware configuration the structure name is appended with the variant name. For variant unaware configuration <variant> is ignored. This structure captures the configuration details of the controller base address, receive pin selection and loop back selection, message object mapping, default baud rate and index to baudrate along with number of baudrates configured for the controller, kernel that the controller belongs to, the logical and core specific controller Id and the CAN FD support (generated only if atleast one CAN FD baudrate is configured in any of the controllers of the CAN driver else structure element is not generated) for the CAN controllers allocated to this core.	
Example(s)	**Action**	**Generated output**

	Configure 2 controllers all allocated to core 0, CAN FD configuration exists (i.e. CAN_17_MCMCAN_FD_ENABLE is STD_ON)	`static const`
		`Can_17_McmCan_ControllerConfigType \`
		`  Can_17_McmCan_kControllerConfigCore0[2] =`
		`{`
		`    {`
		`    /* Can controller Base Node address */`

```
    (volatile Ifx_CAN_N*)0xf0208100U,
    /* combination of Loopback and receive
input pin selection setting */
    0x100U,
    /* The controller Hw object
configuration mapping information */
    {
      /* Tx Message storage start Index */
      0x0U,
      /* Total no of Tx Message configured
*/
      0x5U,
      /* Rx Message SID filter mask start
Index */
      0x0U,
      /* Total no of SID filter mask
configured */
      0x7U,
      /* Rx Message XID filter mask start
Index */
      0x0U,
      /* Total no of XID filter mask
configured */
      0x7U
    },
    /* Default baudrate configuration Index
*/
    0x0U,
    /* Start index value of Baudrate
configuration */
    0x0U,
    /* Total no of Baudrate configuration */
    0x5U,
    /* The controller Associated Kernel
configuration Index */
    0x0U,
    /* The CAN controller Hw Index */
    0x00U,
    /* The CAN controller Logical Hw Index -
Controller ID defined by user */
    0,
    /* FD support status of the controller
*/
    TRUE
    },
    {
    /* Can controller Base Node address */
    (volatile Ifx_CAN_N*)0xf0228100U,
    /* combination of Loopback and receive
input pin selection setting */
    0x0U,
    /* The controller Hw object
configuration mapping information */
    {
      /* Tx Message storage start Index */
      0xaU,
      /* Total no of Tx Message configured
*/
      0x5U,
      /* Rx Message SID filter mask start
Index */
      0xeU,
      /* Total no of SID filter mask
configured */
      0x7U,
      /* Rx Message XID filter mask start
Index */
      0xeU,
      /* Total no of XID filter mask
configured */
      0x7U
    },
    /* Default baudrate configuration Index
*/
    0x8U,
    /* Start index value of Baudrate
configuration */
    0x8U,
    /* Total no of Baudrate configuration */
    0x3U,
    /* The controller Associated Kernel
configuration Index */
    0x2U,
```

	`/* The CAN controller Hw Index */`
	`0x00U,`
	`/* The CAN controller Logical Hw Index - Controller ID defined by user */`
	`2,`
	`/* FD support status of the controller */`
	`FALSE`
	`}`
	`};`

Configure 2 controllers all allocated to core 0 with no CAN FD configuration (i.e. CAN_17_MCMCAN_FD_ENABLE is STD_OFF).	`static const Can_17_McmCan_ControllerConfigType \`
	`Can_17_McmCan_kControllerConfigCore0[2] =`
	`{`
	`{`
	`/* Can controller Base Node address */`
	`(volatile Ifx_CAN_N*)0xf0208100U,`
	`/* combination of Loopback and receive input pin selection setting */`
	`0x100U,`
	`/* The controller Hw object configuration mapping information */`
	`{`
	`/* Tx Message storage start Index */`
	`0x0U,`
	`/* Total no of Tx Message configured */`
	`0x5U,`
	`/* Rx Message SID filter mask start Index */`
	`0x0U,`
	`/* Total no of SID filter mask configured */`
	`0x7U,`
	`/* Rx Message XID filter mask start Index */`
	`0x0U,`
	`/* Total no of XID filter mask configured */`
	`0x7U`
	`},`

```
    /* Default baudrate configuration Index
*/
    0x0U,
    /* Start index value of Baudrate
configuration */
    0x0U,
    /* Total no of Baudrate configuration */
    0x5U,
    /* The controller Associated Kernel
configuration Index */
    0x0U,
    /* The CAN controller Hw Index */
    0x00U,
    /* The CAN controller Logical Hw Index -
Controller ID defined by user */
    0
    },
    {
    /* Can controller Base Node address */
    (volatile Ifx_CAN_N*)0xf0228100U,
    /* combination of Loopback and receive
input pin selection setting */
    0x0U,
    /* The controller Hw object
configuration mapping information */
    {
      /* Tx Message storage start Index */
      0xaU,
      /* Total no of Tx Message configured
*/
      0x5U,
      /* Rx Message SID filter mask start
Index */
      0xeU,
      /* Total no of SID filter mask
configured */
      0x7U,
      /* Rx Message XID filter mask start
Index */
      0xeU,
      /* Total no of XID filter mask
configured */
```

```
          0x7U
        },
        /* Default baudrate configuration Index
*/
        0x8U,
        /* Start index value of Baudrate
configuration */
        0x8U,
        /* Total no of Baudrate configuration */
        0x3U,
        /* The controller Associated Kernel
configuration Index */
        0x2U,
        /* The CAN controller Hw Index */
        0x00U,
        /* The CAN controller Logical Hw Index -
Controller ID defined by user */
        2
        }
};
```

1.2.4.1 Member: CanNodeAddressPtr

Table 80 CanNodeAddressPtr

Name	CanNodeAddressPtr	
Type	Ifx_CAN_N*	
Description	The base address of the configured controller.	
Verification method	The generated structure member is present in Can_17_McmCan_kControllerConfigCore<x>[_<variant>] structure. The structure member is generated based on the container 'CanControllerBaseAddress'.	
Example(s)	**Action**	**Generated output**
	Configure a controller with CanControllerBaseAddress as 4028793088	`(volatile Ifx_CAN_N*)0xf0228100U,`

1.2.4.2 Member: CanNPCRValue

Table 81 CanNPCRValue

Name	CanNPCRValue
Type	uint32
Description	The sfr configuration for receive pin selection configuration.

<table>
<tr><td><b>Verification method</b></td><td colspan="2">The generated structure member is present in Can_17_McmCan_kControllerConfigCore<x>[_<variant>] structure. The structure member is generated based on the containers ‘CanControllerLoopbackEnable’ and ‘CanRxInputSelection’.
The structure element has the following bits manipulated according to configuration:
• If ‘CanControllerLoopbackEnable’ is set to ‘True’ then bit 8 of this structure is set to ‘1’ else to ‘0’.
• If ‘CanRxInputSelection’ effects the bits 0 – 2 and the bits are set based on the type selected in this container.
○ CANxx_RXDA value is set to 0
○ CANxx_RXDB value is set to 1
○ CANxx_RXDC value is set to 2
○ CANxx_RXDD value is set to 3
○ CANxx_RXDE value is set to 4
○ CANxx_RXDF value is set to 5
○ CANxx_RXDG value is set to 6
○ CANxx_RXDH value is set to 7</td></tr>
<tr><td><b>Example(s)</b></td><td><b>Action</b></td><td><b>Generated output</b></td></tr>
<tr><td></td><td>Configure a controller with ‘CanControllerLoopbackEnable’ as ‘False’ and using ‘CanRxInputSelection’ as CANxx_RXDC</td><td>0x2U</td></tr>
<tr><td></td><td>Configure a controller with ‘CanControllerLoopbackEnable’ as ‘True’ and using ‘CanRxInputSelection’ as CANxx_RXDA</td><td>0x100U</td></tr>
</table>

1.2.4.3 Member: CanControllerMOMap [CAN_17_MCMCAN_NOOF_MOMAP_PER_CONTROLLER]

Table 82 CanControllerMOMap [CAN_17_MCMCAN_NOOF_MOMAP_PER_CONTROLLER]

Name	CanControllerMOMap [CAN_17_MCMCAN_NOOF_MOMAP_PER_CONTROLLER]
Type	uint16
Description	The array holding the memory mapping to CAN configuration structures and details for the CAN controller.
Verification method	The generated structure member is present in Can_17_McmCan_kControllerConfigCore<x>[_<variant>] structure. The structure member is generated as a array of size 6 filled with the offset in the related structures and the number of elements configured for transmit hardware objects and the standard and extended receive hardware objects for the CAN controller used. • The 1st element in the array gives the start index of transmit hardware objects in the array of structures Can_17_McmCan_kTxHwObjectConfigCore<x>[_<variant>] for the current controller.

	• The 2nd element in the array gives the number of transmit hardware objects that are configured in the array of structures Can_17_McmCan_kTxHwObjectConfigCore<x>[_<variant>] for the current controller. • The 3rd element in the array gives the start index of standard Id hardware objects in the array of structures Can_17_McmCan_kSIDFilterConfigCore<x>[_<variant>] for the current controller. • The 4th element in the array gives the number of standard Id hardware objects that are configured in the array of structures Can_17_McmCan_kSIDFilterConfigCore<x>[_<variant>] for the current controller. • The 5th element in the array gives the start index of extended Id hardware objects in the array of structures Can_17_McmCan_kXIDFilterConfigCore<x>[_<variant>] for the current controller. • The 6th element in the array gives the number of extended Id hardware objects that are configured in the array of structures Can_17_McmCan_kXIDFilterConfigCore<x>[_<variant>] for the current controller.	
Example(s)	**Action**	**Generated output**

	• Configure 2 controller.
	• Configure controller0 with 10 transmit hardware object, 5 standard ID receive hardware objects and 6 extended Id receive hardware objects.
	• Configure controller1 with 6 transmit hardware object, 15 standard ID receive hardware objects and 16 extended Id receive hardware objects.
	The variable for controller0
		`{`
		`/* Tx Message storage start Index */`
		`0x0U,`
		`/* Total no of Tx Message configured */`
		`0xAU,`
		`/* Rx Message SID filter mask start Index */`
		`0x0U,`
		`/* Total no of SID filter mask configured */`
		`0x5U,`
		`/* Rx Message XID filter mask start Index */`
		`0x0U,`
		`/* Total no of XID filter mask configured */`
		`0x6U`
		`}`

	• Configure 2 controller.
	• Configure controller0 with 10 transmit hardware object, 5 standard ID receive hardware objects and 6 extended Id receive hardware objects.
	• Configure controller1 with 6 transmit hardware object, 15 standard ID
		`{`
		`/* Tx Message storage start Index */`
		`0x10U,`
		`/* Total no of Tx Message configured */`
		`0x6U,`

	receive hardware objects and 16 extended Id receive hardware objects.
	The variable for controller1
		`/* Rx Message SID filter mask start Index */`
		`0x5U,`
		`/* Total no of SID filter mask configured */`
		`0xFU,`
		`/* Rx Message XID filter mask start Index */`
		`0x6U,`
		`/* Total no of XID filter mask configured */`
		`0x10U`
		`}`

1.2.4.4 Member: CanDefaultBRCfgIndx

Table 83 CanDefaultBRCfgIndx

Name	CanDefaultBRCfgIndx	
Type	uint16	
Description	The index of the default baudrate configured for this controller.	
Verification method	The generated structure member is present in Can_17_McmCan_kControllerConfigCore<x>[_<variant>] structure. The structure member is generated based on the offset in the Can_17_McmCan_kBaudrateConfigCore<x>[_<variant>] for the baudrate setting configured in container 'CanControllerDefaultBaudrate' for the current controller.	
Example(s)	**Action**	**Generated output**
	Configure a controller with 'CanControllerDefaultBaudrate' of 500kbps which is the 10th configuration in the baudrate configuration structure	`0xAU`

1.2.4.5 Member: CanBaudrateCfgIndx

Table 84 CanBaudrateCfgIndx

Name	CanBaudrateCfgIndx	
Type	uint16	
Description	The offset for the start index of the baudrate configured for this controller.	
Verification method	The generated structure member is present in Can_17_McmCan_kControllerConfigCore<x>[_<variant>] structure. This structure member generates the index in the array Can_17_McmCan_kBaudrateConfigCore<x>[_<variant>] where the first baudrate configurations for the current controller is present.	
Example(s)	**Action**	**Generated output**
	Configure 2 controller with 3 and 4 baudrates each. The element for controller0	`0U`

	Configure 2 controller with 3 and 4 baudrates each. The element for controller1 (the array index for controller 1 is offsetted by 3 as there are 3 baudrates configured in controller 0 whose baudrates are captured in the array of structures Can_17_McmCan_kBaudrateConfigCore<x> from index 0 to 2, as it has 3 elements and baudrate start index is 0).	`3U`

1.2.4.6 Member: CanNoOfBaudrateCfg

Table 85 CanNoOfBaudrateCfg

Name	CanNoOfBaudrateCfg	
Type	uint16	
Description	The total number of baud rates configured for the controller.	
Verification method	The generated structure member is present in Can_17_McmCan_kControllerConfigCore<x>[_<variant>] structure. This structure member generates the number of elements in the structure Can_17_McmCan_kBaudrateConfigCore<x>[_<variant>] configurations for the current controller.	
Example(s)	**Action**	**Generated output**
	Configure 2 controller with 3 and 4 baudrates each. The element for controller0 (3 baudrates configured)	`3U`
	Configure 2 controller with 3 and 4 baudrates each. The element for controller1 (4 baudrates configured)	`4U`

1.2.4.7 Member: CanKernelHwId

Table 86 CanKernelHwId

Name	CanKernelHwId	
Type	uint8	
Description	The kernel Id of the CAN controller.	
Verification method	The generated structure member is present in Can_17_McmCan_kControllerConfigCore<x>[_<variant>] structure. This structure member generates the kernel Id that the current controller belongs to.	
Example(s)	**Action**	**Generated output**
	Configure a controller with CanControllerBaseAddress as 4028793088(of kernel 2)	`0x2U`
	Configure a controller with CanControllerBaseAddress as 4028663040 (of kernel 0)	`0x0U`

1.2.4.8 Member: CanControllerHwId

Table 87 CanControllerHwId

Name	CanControllerHwId	
Type	uint8	
Description	The node Id in the kernel of the CAN controller.	
Verification method	The generated structure member is present in Can_17_McmCan_kControllerConfigCore<x>[_<variant>] structure. This structure member generates the node Id in the kernel that the current controller belongs to.	
Example(s)	**Action**	**Generated output**
	Configure a controller with CanControllerBaseAddress as 4028793088(of kernel 2 node 0)	`0x0U`
	Configure a controller with CanControllerBaseAddress as 4028663040 (of kernel 0 node 1)	`0x1U`

1.2.4.9 Member: CanControllerLogicalId

Table 88 CanControllerLogicalId

Name	CanControllerLogicalId	
Type	uint8	
Description	The configured controller Id of the current controller	
Verification method	The generated structure member is present in Can_17_McmCan_kControllerConfigCore<x>[_<variant>] structure. This structure member is generated based on the ‘CanControllerId’ configured for the current controller.	
Example(s)	**Action**	**Generated output**
	Configure 2 controller with 3 and 4 baudrates each. The element for controller0	`0U`
	Configure 2 controller with 3 and 4 baudrates each. The element for controller1	`1U`

1.2.4.10 Member: CanFDSupport

Table 89 CanFDSupport

Name	CanFDSupport
Type	boolean
Description	Enables/Disables the CAN FD support in the current controller.
Verification method	The generated structure member is present in Can_17_McmCan_kControllerConfigCore<x>[_<variant>] structure. The structure member is generated as a ‘TRUE’ when container ‘CanControllerFdBaudrateConfig/[]’ exists in any of the baudrates of current controller else it is generated as ‘FALSE’.

	Note: The structure element 'CanFDSupport' shall be generated in all Can_17_McmCan_kControllerConfigCore<x>[_<variant>] structures only if atleast one CAN controller baudrate is configured with an element in the list 'CanControllerFdBaudrateConfig' else this element shall not be generated in the structure. This structure member is generated with the value 'FALSE' if the list 'CanControllerFdBaudrateConfig' does not have any elements in theCAN baudrate configuration.	
Example(s)	**Action**	**Generated output**
	Configure a controller with CAN FD baudrate present in the baudaret configurations	`TRUE`
	Configure a controller with no CAN FD baudrate present in the baudrate configurations	`FALSE`

1.2.4.11 Member: CanRxFIFO0ProcessingConfig

Table 90 CanRxFIFO0ProcessingConfig

Name	CanRxFIFO0ProcessingConfig	
Type	Can_17_McmCan_RxFIFOProcessingType	
Description	Indicates the RxFIFO0 processing configuration.	
Verification method	The generated structure member is present in Can_17_McmCan_kControllerConfigCore<x>[_<variant>] structure. The structure member is generated based on the containers CanRxProcessing, CanObjectType and CanHwObjectCount.	
Example(s)	**Action**	**Generated output**
	• Configure 1 controller with CanRxProcessing set as 'INTERRUPT'. • Configure the controller with 2 receive hardware object having CanHwObjectCount as 5 and 1.	`CAN_17_MCMCAN_RX_FIFO_INTERRUPT`
	• Configure 1 controller with CanRxProcessing set as 'POLLING'. • Configure the controller with receive hardware object having CanHwObjectCount as 5.	`CAN_17_MCMCAN_RX_FIFO_POLLING`
	• Configure 1 controller with CanRxProcessing set as 'POLLING'.	`CAN_17_MCMCAN_RX_FIFO_NOT_CONFIGURED`

	• Configure the controller with receive hardware object having CanHwObjectCount as 1.	

1.2.4.12 Member: CanRxFIFO1ProcessingConfig

Table 91 CanRxFIFO1ProcessingConfig

Name	CanRxFIFO1ProcessingConfig	
Type	Can_17_McmCan_RxFIFOProcessingType	
Description	Indicates the RxFIFO1 processing configuration.	
Verification method	The generated structure member is present in Can_17_McmCan_kControllerConfigCore<x>[_<variant>] structure. The structure member is generated based on the containers CanRxProcessing, CanObjectType, CanHwObjectCount and the number of CanHardwareObject.	
Example(s)	**Action**	**Generated output**
	• Configure 1 controller with CanRxProcessing set as 'INTERRUPT'. • Configure the controller with 2 receive hardware object having CanHwObjectCount as 5.	CAN_17_MCMCAN_RX_FIFO_INTERRUPT
	• Configure 1 controller with CanRxProcessing set as 'MIXED'. • Configure the controller with 2 receive hardware object having CanHwObjectCount as 5 and CanHardwareObjectUsesPolling enabled.	CAN_17_MCMCAN_RX_FIFO_POLLING
	• Configure 1 controller with CanRxProcessing set as 'POLLING'. • Configure the controller with 2 receive hardware object having CanHwObjectCount as 1.	CAN_17_MCMCAN_RX_FIFO_NOT_CONFIGURED

1.2.4.13 Member: CanHrhNDAT1PollingMask

Table 92 CanHrhNDAT1PollingMask

Name	CanHrhNDAT1PollingMask
Type	uint32
Description	Specifies the Hrh polling mask for NDAT1
Verification method	The generated structure member is present in Can_17_McmCan_kControllerConfigCore<x>[_<variant>] structure. The structure member is

	generated based on the containers CanObjectType, CanHwObjectCount, CanRxProcessing and the number of CanHardwareObject configured.	
Example(s)	**Action**	**Generated output**
	• Configure 1 controller with CanRxProcessing set as 'POLLING'. • Configure the controller with 2 receive hardware object having CanHwObjectCount as 1.	`0x3U`
	• Configure 1 controller with CanRxProcessing set as 'INTERRUPT'. • Configure the controller with a receive hardware object having CanHwObjectCount as 5.	`0x0U`

1.2.4.14 Member: CanHrhNDAT2PollingMask

Table 93 CanHrhNDAT2PollingMask

Name	CanHrhNDAT2PollingMask	
Type	uint32	
Description	Specifies the Hrh polling mask for NDAT2	
Verification method	The generated structure member is present in Can_17_McmCan_kControllerConfigCore<x>[_<variant>] structure. The structure member is generated based on the containers CanObjectType, CanHwObjectCount, CanRxProcessing and the number of CanHardwareObject configured.	
Example(s)	**Action**	**Generated output**
	• Configure 1 controller with CanRxProcessing set as 'POLLING'. • Configure the controller with a receive hardware object having CanHwObjectCount as 1.	`0x0U`
	• Configure 1 controller with CanRxProcessing set as 'POLLING'. • Configure the controller with 34 receive hardware object having CanHwObjectCount as 1.	`0x3U`

1.2.4.15 Member: CanTxPollingObjectMask

Table 94 CanTxPollingObjectMask

Name	CanTxPollingObjectMask
Type	uint32
Description	Specifies the Hth polling mask for the controller

Verification method	The generated structure member is present in Can_17_McmCan_kControllerConfigCore<x>[_<variant>] structure. The structure member is generated based on the containers CanObjectType, CanTxProcessing and the number of CanHardwareObject configured.	
Example(s)	**Action**	**Generated output**
	• Configure 1 controller with CanTxProcessing set as 'INTERRUPT'. • Configure the controller with 2 receive and 2 transmit hardware object.	`0x0U`
	• Configure 1 controller with CanTxProcessing set as 'POLLLING'. • Configure the controller with 2 receive and 2 transmit hardware object.	`0x3U`

1.2.4.16 Member: CanEnableInterruptMask

Table 95 CanEnableInterruptMask

Name	CanEnableInterruptMask	
Type	uint32	
Description	Specifies the interrupt mask to enable the interrupts for the controller	
Verification method	The generated structure member is present in Can_17_McmCan_kControllerConfigCore<x>[_<variant>] structure. The structure member is generated based on the containers CanTxProcessing, CanRxProcessing and CanBusoffProcessing. The values of the interrupt enable bits depend on the RX FIFO, dedicated, TX EVENT FIFO configured for TX and RX processing.	
Example(s)	**Action**	**Generated output**
	Configure a controller with CanTxProcessing, CanRxProcessing and CanBusoffProcessing set as 'INTERRUPT'. Configure this controller with: 1 receive dedicated hardware object. 1 transmit dedicated hardware object.	`0x2081000U`
	Configure a controller with CanTxProcessing and CanBusoffProcessing set as 'INTERRUPT'. CanRxProcessing is set as 'POLLING'. Configure this controller with: 1 receive dedicated hardware object. 2 transmit hardware object (dedicated and queue).	`0x2001000U`
	Configure a controller with CanTxProcessing, CanRxProcessing and CanBusoffProcessing set as 'INTERRUPT'. Configure this controller with:	`0x2081006U`

	2 receive hardware object (dedicated and FIFO0). 2 transmit hardware object (dedicated and queue).	

1.2.5 Structure: Can_17_McmCan_kControllerMsgRAMMapConfigCore<x>[_<variant>][CAN_17_MCMCAN_CORE<x>_NOOF_CONTROLLER]

Table 96 Can_17_McmCan_kControllerMsgRAMMapConfigCore<x>[_<variant>][CAN_17_MCMCAN_CORE<x>_NOOF_CONTROLLER]

Name	Can_17_McmCan_kControllerMsgRAMMapConfigCore<x>[_<variant>][CAN_17_MCMCAN_CORE<x>_NOOF_CONTROLLER]	
Type	Can_17_McmCan_ControllerMsgRAMConfigType	
Description	Configuration structure holding the RAM allocation details of CAN driver for all the CAN controllers belonging to Core <x> which will be referenced in core specific configuration structure. (x ranges from 0 to 5)	
Verification method	The generated file has this structure for the different RAM memory allocations configured for each controller. <Variant> indicates the name of the post-build variant. For a variant aware configuration the structure name is appended with the variant name. For variant unaware configuration <variant> is ignored. This structure captures the configuration details for RAM memory start addresses for the different memory sections and the count of the different transmit and receive objects configured for CAN controllers allocated to this core.	
Example (s)	**Action**	**Generated output**

	Configure 2 controllers in core 0 with multiplexed transmission enabled in the configuration.	`static const`
		`Can_17_McmCan_ControllerMsgRAMConfigType \`
		`Can_17_McmCan_kControllerMsgRAMMapConfigCore0[2] =`
		`{`
		`  {`
		`  /* Start Address of each section within the Message RAM */`
		`  {`
		`  0xf0200000UL,`
		`  0xf020001cUL,`
		`  0x00000000UL,`
		`  0x00000000UL,`
		`  0xf0200054UL,`
		`  0xf0200104UL,`
		`  0xf020012cUL`
		`  },`
		`  0x5U, 0x5U, 0x0U, 0x0U, 0x0U, 0x0U,0x0U, FALSE`

	},
	
	 {
	 /* Start Address of each section within the Message RAM */
	 {
	 0xf0210000UL,
	 0xf021001cUL,
	 0x00000000UL,
	 0x00000000UL,
	 0xf0210054UL,
	 0xf0210104UL,
	 0xf021012cUL
	 },
	 0x5U, 0x5U, 0x0U, 0x0U, 0x0U, 0x0U,0x0U, FALSE
	 }
	 };

Configure 2 controllers in core 0 with multiplexed transmission disabled in the configuration.	static const Can_17_McmCan_ControllerMsgRAMConfigType \ Can_17_McmCan_kControllerMsgRAMMapConfigCore0[2] = { { /* Start Address of each section within the Message RAM */ { 0xf0200000UL, 0xf020001cUL, 0x00000000UL, 0x00000000UL, 0xf0200054UL, 0xf0200104UL, 0xf020012cUL }, 0x5U, 0x5U, 0x0U, 0x0U, 0x0U, 0x0U,0x0U }, {

```
  /* Start Address of each section within
the Message RAM */
  {
  0xf0210000UL,
  0xf021001cUL,
  0x00000000UL,
  0x00000000UL,
  0xf0210054UL,
  0xf0210104UL,
  0xf021012cUL
  },
  0x5U, 0x5U, 0x0U, 0x0U, 0x0U, 0x0U,0x0U
  }
};
```

1.2.5.1 Member: CanControllerMsgRAMMap[CAN_17_MCMCAN_NOOF_RAM_SECTIONS_PER_CONTROLLER]

Table 97 CanControllerMsgRAMMap[CAN_17_MCMCAN_NOOF_RAM_SECTIONS_PER_CONTROLLER]

Name	CanControllerMsgRAMMap[CAN_17_MCMCAN_NOOF_RAM_SECTIONS_PER_CONTROLLER]
Type	uint32
Description	The array holding the start addresses for the different sections of memeory in the RAM for the CAN controller.
Verification method	The generated structure member is present in Can_17_McmCan_kControllerMsgRAMMapConfigCore<x>[_<variant>] structure. The structure member is generated as a array of size 7 filled with the start addresses of the different RAM memory sections for the CAN controller used. In the array of 7, The 1st array index holds the start address of the standard Id section, each standard Id of the controller shall need a 4 byte slot of memory, this section starts at the end of the previous controller of the same kernel's last section or if this is the first controller allocated in the kernel then this is the RAM start address of the kernel. The 2nd array index holds the start address of the extended Id section, each extended Id of the controller shall need a 8 byte slot of memory, this section start address starts only after the standard ID section has completed. The 3rd array index holds the start address of the receive FIFO 0 message buffer section, each of this message of the controller shall need a 4 multiplied by 16 (CAN FD enabled in controller) or 4 (CAN FD disabled in controller) byte slot of memory, this section start address starts only after the extended ID section has completed. The 4th array index holds the start address of the receive FIFO 1 message buffer section, each of this message of the controller shall need a 4 multiplied by 16 (CAN FD enabled in

	controller) or 4 (CAN FD disabled in controller) byte slot of memory, this section start address starts only after the receive FIFO 0 message buffer section has completed. The 5th array index holds the start address of the dedicated receive message buffer section, each of this message of the controller shall need a 4 multiplied by 16 (CAN FD enabled in controller) or 4 (CAN FD disabled in controller) byte slot of memory, this section start address starts only after the receive FIFO 1 message buffer section has completed. The 6th array index holds the start address of the transmit event message buffer section, each of the transmit message and length of the transmit queue if available of the controller shall need a 8 slot of memory, this section start address starts only after the dedicated receive message buffer section has completed. The 7th array index holds the start address of the transmit message buffer section, each of the transmit message and length of the transmit queue if available of the controller shall need a 4 multiplied by 16 (CAN FD enabled in controller) or 4 (CAN FD disabled in controller) byte slot of memory, this section start address starts only after the transmit event message buffer section has completed. *Note: If any of the sections do not have elements in it then the start address of that section is set to '0x00000000U'.*	
Example(s)	**Action**	**Generated output**

	Configure a controller with 7 standard Id messages, 7 extended Id messages no receive FIFOs, 11 dedicated receive messages and 5 transmit messages with no transmit queue used.	`{`
		`0xf0200000UL,`
		`0xf020001cUL,`
		`0x00000000UL,`
		`0x00000000UL,`
		`0xf0200054UL,`
		`0xf0200104UL,`
		`0xf020012cUL`
		`}`

1.2.5.2 Member: CanTxDedBuffCount

Table 98 CanTxDedBuffCount

Name	CanTxDedBuffCount	
Type	uint8	
Description	The configured number of dedicated transmit type buffers.	
Verification method	The generated structure member is present in Can_17_McmCan_kControllerMsgRAMMapConfigCore<x>[_<variant>] structure. The structure member is generated based on the number of dedicated transmit messages.	
Example(s)	**Action**	**Generated output**
	Configure 11 transmit messages for the same controller with 1 of the transmit messages having CanHwObjectCount as 4 (>1). The value of CanTxDedBuffCount.	`0xAU`

1.2.5.3 Member: CanTxEvntFIFOSize

Table 99 CanTxEvntFIFOSize

Name	CanTxEvntFIFOSize	
Type	uint8	
Description	The configured number of transmit type buffers.	
Verification method	The generated structure member is present in Can_17_McmCan_kControllerMsgRAMMapConfigCore<x>[_<variant>] structure. The structure member is generated based on the number of transmit messages configured with the value of CanHwObjectCount for transmit queue considered as its size.	
Example(s)	**Action**	**Generated output**
	Configure 11 transmmit messages for the same controller with 1 of the transmit messages having CanHwObjectCount as 4 (>1). The value of CanTxDedBuffCount.	`0xEU`

1.2.5.4 Member: CanRxFIFO0Size

Table 100 CanRxFIFO0Size

Name	CanRxFIFO0Size	
Type	uint8	
Description	The configured number of elements in receive FIFO0 buffer.	
Verification method	The generated structure member is present in Can_17_McmCan_kControllerMsgRAMMapConfigCore<x>[_<variant>] structure. The structure member is generated based on the CanHwObjectCount configured for the first receive hardware object with CanHwObjectCount greater than 1 for the controller.	
Example(s)	**Action**	**Generated output**
	Configure 10 recieve messages for the same controller with 2 of the recieve messages having CanHwObjectCount as 10 and 12 (>1). The value of CanRxFIFO0Size.	`0xAU`

1.2.5.5 Member: CanRxFIFO0Threshold

Table 101 CanRxFIFO0Threshold

Name	CanRxFIFO0Threshold
Type	uint8
Description	The configured number of elements as receive FIFO0 threshold.
Verification method	The generated structure member is present in Can_17_McmCan_kControllerMsgRAMMapConfigCore<x>[_<variant>] structure. The structure member is generated based on the CanHwFIFOThreshold configured for

	the first receive hardware object with CanHwObjectCount greater than 1 for the controller.	
Example(s)	**Action**	**Generated output**
	Configure 10 recieve messages for the same controller with 2 of the recieve messages having CanHwObjectCount as 10 and 12 (>1) and respective CanHwFIFOThreshold as 5 and 6. The value of CanRxFIFO0Size.	`0x5U`

1.2.5.6 Member: CanRxFIFO1Size

Table 102 CanRxFIFO1Size

Name	CanRxFIFO1Size	
Type	uint8	
Description	The configured number of elements in receive FIFO1 buffer.	
Verification method	The generated structure member is present in Can_17_McmCan_kControllerMsgRAMMapConfigCore<x>[_<variant>] structure. The structure member is generated based on the CanHwObjectCount configured for the second receive hardware object with CanHwObjectCount greater than 1 for the controller.	
Example(s)	**Action**	**Generated output**
	Configure 10 recieve messages for the same controller with 2 of the recieve messages having CanHwObjectCount as 10 and 12 (>1). The value of CanRxFIFO1Size.	`0xBU`

1.2.5.7 Member: CanRxFIFO1Threshold

Table 103 CanRxFIFO1Threshold

Name	CanRxFIFO1Threshold	
Type	uint8	
Description	The configured number of elements as receive FIFO1 threshold.	
Verification method	The generated structure member is present in Can_17_McmCan_kControllerMsgRAMMapConfigCore<x>[_<variant>] structure. The structure member is generated based on the CanHwFIFOThreshold configured for the second receive hardware object with CanHwObjectCount greater than 1 for the controller.	
Example(s)	**Action**	**Generated output**

	Configure 10 recieve messages for the same controller with 2 of the recieve messages having CanHwObjectCount as 10 and 12 (>1) and respective CanHwFIFOThreshold as 5 and 6. The value of CanRxFIFO0Size.	`0x6U`

1.2.5.8 Member: CanTxQueueSize

Table 104 CanTxQueueSize

Name	CanTxQueueSize	
Type	uint8	
Description	The configured number of elements in transmit queue buffer.	
Verification method	The generated structure member is present in Can_17_McmCan_kControllerMsgRAMMapConfigCore<x>[_<variant>] structure. The structure member is generated based on the CanHwObjectCount configured for the transmit hardware object with CanHwObjectCount greater than 1 for the controller. *Note: The structure element 'CanTxQueueSize 'shall be generated in all Can_17_McmCan_kControllerMsgRAMMapConfigCore<x>[_<variant>] structures only if atleast one CAN controller has transmit hardware object with CanHwObjectCount value greater than 1 else this element shall not be generated in the structure. This structure member is generated with the value '0' if the controller does not have any transmit hardware object with CanHwObjectCount greater than 1 in the configuration.*	
Example(s)	**Action**	**Generated output**
	Configure 11 transmmit messages for the same controller with 1 of the transmit messages having CanHwObjectCount as 4 (>1). The value of CanTxDedBuffCount.	`0x4U`

1.2.5.9 Member: CanTxQueueStatus

Table 105 CanTxQueueStatus

Name	CanTxQueueStatus
Type	boolean
Description	Enables/Disables transmit queue support for the controller
Verification method	The generated structure member is present in Can_17_McmCan_kControllerMsgRAMMapConfigCore<x>[_<variant>] structure. The

	structure member is generated as a ‘TRUE’ when the controller has a transmit hardware object with CanHwObjectCount greater than 1 else it is generated as ‘FALSE’. *Note: The structure element ‘CanTxQueueStatus’ shall be generated in all Can_17_McmCan_kControllerMsgRAMMapConfigCore<x>[_<variant>] structures only if atleast one CAN controller has transmit hardware object with CanHwObjectCount value greater than 1 else this element shall not be generated in the structure. This structure member is generated with the value ‘FALSE’ if the controller does not have any transmit hardware object with CanHwObjectCount greater than 1 in the configuration*	
Example(s)	**Action**	**Generated output**
	Configure 11 transmmit messages for the same controller with 1 of the transmit messages having CanHwObjectCount as 4 (>1). The value of CanTxDedBuffCount.	`TRUE`
	Configure 11 transmmit messages for the same controller with no of the transmit messages having CanHwObjectCount greater than 1. The value of CanTxDedBuffCount.	`FALSE`

1.2.6 Structure: Can_17_McmCan_kEventHandlingConfigCore<x>[_<variant>][CAN_17_MCMCAN_CORE<x>_NOOF_CONTROLLER]

Table 106 Can_17_McmCan_kEventHandlingConfigCore<x>[_<variant>][CAN_17_MCMCAN_CORE<x>_NOOF_CONTROLLER]

Name	Can_17_McmCan_kEventHandlingConfigCore<x>[_<variant>][CAN_17_MCMCAN_CORE<x>_NOOF_CONTROLLER]	
Type	Can_17_McmCan_EventHandlingType	
Description	Configuration structure of CAN driver detailing the transmit, receive, bus off and wakeup processing strategy for all CAN controllers belonging to Core <x> which will be referenced in core specific configuration structure. (x ranges from 0 to 5)	
Verification method	The generated file has this structure for the different kind of processing configured. <Variant> indicates the name of the post-build variant. For a variant aware configuration the structure name is appended with the variant name. For variant unaware configuration <variant> is ignored. This structure captures the configuration details for transmit, receive, bus off and wakeup processing strategy for CAN controllers allocated to this core.	
Example(s)	**Action**	**Generated output**

	Configure 2 controllers with 1 controller having all processing as INTERRUPT and the other having all processing as POLLING.	`static const Can_17_McmCan_EventHandlingType \`
		`  Can_17_McmCan_kEventHandlingConfigCore0[2] =`
		`{`
		` { {`
		`(CAN_17_MCMCAN_POLLING),(CAN_17_MCMCAN_POLLING),(CAN_17_MCMCAN_POLLING),(CAN_17_MCMCAN_POLLING)} },`
		` { {`
		`(CAN_17_MCMCAN_INTERRUPT),(CAN_17_MCMCAN_INTERRUPT),(CAN_17_MCMCAN_INTERRUPT),(CAN_17_MCMCAN_INTERRUPT)}`
		`}`
		`};`

1.2.6.1 Member: CanTxProcessing

Table 107 CanTxProcessing

Name	CanTxProcessing	
Type	Can_17_McmCan_ProcessingType	
Description	Specifies the way transmission event on the controller is notified.	
Verification method	The generated structure member is present in Can_17_McmCan_kEventHandlingConfigCore<x>[_<variant>] structure. The structure member is generated based on the value of CanTxProcessing of a configured CanController.	
Example(s)	**Action**	**Generated output**
	Configure a controller with CanTxProcessing set as 'INTERRUPT'	`CAN_17_MCMCAN_INTERRUPT`
	Configure a controller with CanTxProcessing set as 'POLLING'	`CAN_17_MCMCAN_POLLING`
	Configure a controller with CanTxProcessing set as 'MIXED'	`CAN_17_MCMCAN_MIXED`

1.2.6.2 Member: CanRxProcessing

Table 108 CanRxProcessing

Name	CanRxProcessing	
Type	Can_17_McmCan_ProcessingType	
Description	Specifies the way reception event on the controller is notified.	
Verification method	The generated structure member is present in Can_17_McmCan_kEventHandlingConfigCore<x>[_<variant>] structure. The structure member is generated based on the value of CanRxProcessing of a configured CanController.	
Example(s)	**Action**	**Generated output**

	Configure a controller with CanRxProcessing set as ‘INTERRUPT’	`CAN_17_MCMCAN_INTERRUPT`
	Configure a controller with CanRxProcessing set as ‘POLLING’	`CAN_17_MCMCAN_POLLING`
	Configure a controller with CanRxProcessing set as ‘MIXED’	`CAN_17_MCMCAN_MIXED`

1.2.6.3 Member: CanBusoffProcessing

Table 109 CanBusoffProcessing

Name	CanBusoffProcessing	
Type	Can_17_McmCan_ProcessingType	
Description	Specifies the way bus off event on the controller is notified.	
Verification method	The generated structure member is present in Can_17_McmCan_kEventHandlingConfigCore<x>[_<variant>] structure. The structure member is generated based on the value of CanBusoffProcessing of a configured CanController.	
Example(s)	**Action**	**Generated output**
	Configure a controller with CanBusoffProcessing set as ‘INTERRUPT’	`CAN_17_MCMCAN_INTERRUPT`
	Configure a controller with CanBusoffProcessing set as ‘POLLING’	`CAN_17_MCMCAN_POLLING`

1.2.6.4 Member: CanWakeupProcessing

Table 110 CanWakeupProcessing

Name	CanWakeupProcessing	
Type	Can_17_McmCan_ProcessingType	
Description	Specifies the way wake up event on the controller is notified.	
Verification method	The generated structure member is present in Can_17_McmCan_kEventHandlingConfigCore<x>[_<variant>] structure. The structure member is generated based on the value of CanWakeupProcessing of a configured CanController.	
Example(s)	**Action**	**Generated output**
	Configure a controller with CanBusoffProcessing set as ‘INTERRUPT’	`CAN_17_MCMCAN_INTERRUPT`

	Configure a controller with CanBusoffProcessing set as ‘POLLING’	`CAN_17_MCMCAN_POLLING`

1.2.7 Structure: Can_17_McmCan_kBaudrateConfigCore<x>[_<variant>][Number Of Baudrates Configured For Core<x>]

Table 111 Can_17_McmCan_kBaudrateConfigCore<x>[_<variant>][Number Of Baudrates Configured For Core<x>]

Name	Can_17_McmCan_kBaudrateConfigCore<x>[_<variant>][Number Of Baudrates Configured For Core<x>]	
Type	Can_17_McmCan_ControllerBaudrateConfigType	
Description	Configuration structure of CAN driver for all baudrates configured allocated to the CAN controllers belonging to Core <x> which will be referenced in core specific configuration structure. (x ranges from 0 to 5)	
Verification method	The generated file has this structure for CAN baud rate configured. <Variant> indicates the name of the post-build variant. For a variant aware configuration the structure name is appended with the variant name. For variant unaware configuration <variant> is ignored. This structure captures the configuration details of all the CAN baudrates configured for CAN controllers allocated to this core.	
Example(s)	**Action**	**Generated output**

	Configure 2 CAN controllers with baudrates of 500kbps for controller0 and 1000kbps for controller1 with no FD baudrates present.	`static const`
		`Can_17_McmCan_ControllerBaudrateConfigType`
		`\`
		`  Can_17_McmCan_kBaudrateConfigCore5[2] =`
		`{`
		
		`  {`
		`    /* Configured Baudrate -> 500 kbps */`
		`    /* Actual Baudrate      -> 500.0 kbps`
		`*/`
		`    /* NBRP   -> 4 */`
		`    /* NSJW   -> 0  */`
		`    /* NTSEG1 -> 9  */`
		`    /* NTSEG2 -> 4  */`
		`    0x40904U,`
		`    500U  },`
		
		`  {`
		`    /* Configured Baudrate -> 1000 kbps */`
		`    /* Actual Baudrate      -> 1000.0 kbps`
		`*/`

	/* NBRP -> 1 */ /* NSJW -> 0 */ /* NTSEG1 -> 11 */ /* NTSEG2 -> 6 */ 0x10b06U, 1000U } };

Configure 2 CAN controllers with baudrates of 500kbps for controller0 and 1000kbps for controller1 with FD baudrate present in controller1.	static const
	Can_17_McmCan_ControllerBaudrateConfigType \
	Can_17_McmCan_kBaudrateConfigCore5[2] =
	{
	{
	/* Configured Baudrate -> 100 kbps */
	/* Actual Baudrate -> 100.0 kbps */
	/* NBRP -> 39 */
	/* NSJW -> 0 */
	/* NTSEG1 -> 4 */
	/* NTSEG2 -> 3 */
	0x270403U,
	100U,
	0x0U,
	FALSE
	},
	{
	/* Configured Baudrate -> 500 kbps */
	/* Actual Baudrate -> 500.0 kbps */
	/* NBRP -> 0 */
	/* NSJW -> 3 */
	/* NTSEG1 -> 62 */
	/* NTSEG2 -> 15 */
	0x6003e0fU,
	500U,
	0x1U,
	TRUE
	}

		};

1.2.7.1 Member: CanControllerBaudrate

Table 112 CanControllerBaudrate

Name	CanControllerBaudrate	
Type	uint32	
Description	The calculated CAN baudrate value.	
Verification method	The generated structure member is present in Can_17_McmCan_kBaudrateConfigCore<x>[_<variant>] structure. The structure member is generated as a numeric value. This value is generated as the value to be written into the CAN<x>_ NBTP sfr for applying the intended baudrate from the baudrate settings. *Note: The sfr value is formed in the structure element with values of NSJW (bits 25-31), NBRP (bits 16 – 24), NTSEG1 (bits 8 - 15) and NTSEG2 (bits 0 – 6). The reserved bits are kept as '0'.* *NSJW is set based on the value configured in container 'CanControllerBaudrateConfig /[]/CanControllerSyncJumpWidth' subtracted with 1.* *NBRP is set based on the calculated time quanta and the Mcu clock being used.* *TSEG1 is the sum of the 'CanControllerBaudrateConfig /[]/CanControllerPropSeg' and the 'CanControllerBaudrateConfig /[]/CanControllerSeg1' configured subtracted by 1.* *TSEG2 is set as the value of container 'CanControllerBaudrateConfig /[]/CanControllerSeg2' subtracted by 1.*	
Example(s)	**Action**	**Generated output**
	Configure a controller with 'CanControllerBaudRate' as 500. With NBRP value 4, NSJW value 0, NTSEG1 value 9 and NTSEG2 value 4.	`0x40904U`

1.2.7.2 Member: CanBaudrateCfg

Table 113 CanBaudrateCfg

Name	CanBaudrateCfg
Type	uint16
Description	The configured CAN baudrate value.
Verification method	The generated structure member is present in Can_17_McmCan_kBaudrateConfigCore<x>[_<variant>] structure. The structure member is

	generated as a numeric value. This value is generated from the CAN baudrate value configured in container ‘CanControllerBaudRate’.	
Example(s)	**Action**	**Generated output**
	Configure a controller with ‘CanControllerBaudRate’ as 500.	`500U`
	Configure a controller with ‘CanControllerBaudRate’ as 1000.	`1000U`

1.2.7.3 Member: CanFDIndex

Table 114 CanFDIndex

Name	CanFDIndex	
Type	uint16	
Description	The CAN FD baudrate index offset associated with current baudrate.	
Verification method	The generated structure member is present in Can_17_McmCan_kBaudrateConfigCore<x>[_<variant>] structure. The structure member is generated as a numeric value. This value is generated as an offset indication to the CAN FD baudrate configured, the index is of the structure Can_17_McmCan_kFDBaudrateConfigCore<x>[_<variant>] of the respective core<x>. *Note: The structure element ‘CanFDIndex’ shall be generated in all Can_17_McmCan_kBaudrateConfigCore<x>[_<variant>] structures only if atleast one CAN controller baudrate is configured with an element in the list ‘CanControllerFdBaudrateConfig’ else this element shall not be generated in the structure. This structure member is generated with the value ‘0’ if the list ‘CanControllerFdBaudrateConfig’ does not have any elements in theCAN baudrate configuration.*	
Example(s)	**Action**	**Generated output**
	Configure 1 controller settings. Configure 3 baudrates for the one controller, with baudrate configuration 0 and 2 having CAN FD baudrate configuration in them. Then the expected CanFDIndex for baudrate0 configuration.	`0U`
	Configure 1 controller settings. Configure 3 baudrates for the one controller, with baudrate configuration 0 and 2 having CAN FD baudrate configuration in them. Then the expected CanFDIndex for baudrate1 configuration.	`0U`
	Configure 1 controller settings. Configure 3 baudrates for the one controller, with baudrate configuration 0	`1U`

	and 2 having CAN FD baudrate configuration in them. Then the expected CanFDIndex for baudrate2 configuration.	

1.2.7.4 Member: CanFdConfigEnabled

Table 115 CanFdConfigEnabled

Name	CanFdConfigEnabled	
Type	boolean	
Description	Enables/Disables the CAN FD baudrate configuration.	
Verification method	The generated structure member is present in Can_17_McmCan_kBaudrateConfigCore<x>[_<variant>] structure. The structure member is generated as a 'TRUE' when container 'CanControllerFdBaudrateConfig/[]' exists else it is generated as 'FALSE'. *Note: The structure element 'CanFdConfigEnabled' shall be generated in all Can_17_McmCan_kBaudrateConfigCore<x>[_<variant>] structures only if atleast one CAN controller baudrate is configured with an element in the list 'CanControllerFdBaudrateConfig' else this element shall not be generated in the structure. This structure member is generated with the value 'FALSE' if the list 'CanControllerFdBaudrateConfig' does not have any elements in theCAN baudrate configuration.*	
Example(s)	**Action**	**Generated output**
	Configure 1 controller settings. Configure 3 baudrates for the one controller, with baudrate configuration 0 and 2 having CAN FD baudrate configuration in them. Then the expected CanFDIndex for baudrate0 configuration.	`TRUE`
	Configure 1 controller settings. Configure 3 baudrates for the one controller, with baudrate configuration 0 and 2 having CAN FD baudrate configuration in them. Then the expected CanFDIndex for baudrate1 configuration.	`FALSE`

1.2.8 Structure: Can_17_McmCan_kFDBaudrateConfigCore<x>[_<variant>][Number Of FD Baudrates Configured For Core<x>]

Table 116 Can_17_McmCan_kFDBaudrateConfigCore<x>[_<variant>][Number Of FD Baudrates Configured For Core<x>]

Name	Can_17_McmCan_kFDBaudrateConfigCore<x>[_<variant>][Number Of FD Baudrates Configured For Core<x>]	
Type	Can_17_McmCan_ControllerFDBaudrateConfigType	
Description	Configuration structure of CAN driver for all FD baudrates configured allocated to the CAN controllers belonging to Core <x> which will be referenced in core specific configuration structure. (x ranges from 0 to 5)	
Verification method	The generated file has this structure if at least one of the controller's assigned to Core <x> is having CAN FD baud rate configured. <Variant> indicates the name of the post-build variant. For a variant aware configuration the structure name is appended with the variant name. For variant unaware configuration <variant> is ignored. This structure captures the configuration details of all the CAN FD baudrates configured for CAN controllers allocated to this core. This structure is generated only when atleast 1 CAN FD configuration is present in the controllers associated with this core<x> else it is not generated.	
Example(s)	**Action**	**Generated output**

	Configure 1 CAN controllers with Fd baudrates of 2500kbps each.	`static const`
		`Can_17_McmCan_ControllerFDBaudrateConfigType`
		`\`
		`  Can_17_McmCan_kFDBaudrateConfigCore0[1] =`
		`{`
		`  {`
		`  /* Configured FD Baudrate -> 2500 kbps */`
		`  /* Actual Baudrate     ->`
		`2666.6666666666665 kbps */`
		`  /* DBRP   -> 2 */`
		`  /* DSJW   -> 0  */`
		`  /* DTSEG1 -> 2  */`
		`  /* DTSEG2 -> 0  */`
		`  0x20200U,`
		`  0x0U,`
		`  TRUE`
		`  }`
		`};`

1.2.8.1 Member: CanControllerFDBaudrate

Table 117 **CanControllerFDBaudrate**

Name	CanControllerFDBaudrate
Type	uint32
Description	The calculated CAN FD baudrate value.
Verification method	The generated structure member is present in Can_17_McmCan_kFDBaudrateConfigCore<x>[_<variant>] structure. The structure member

	is generated as a numeric value. This value is generated as the value to be written into the CAN<x>_DBTP sfr for applying the intended FD baudrate from the FD baudrate settings. *Note: The sfr value is formed in the structure element with values of DBRP (bits 16 – 20), DTSEG1 (bits 8 - 12), DTSEG2 (bits 4 – 7) and DSJW (bits 0 – 3). The reserved bits are kept as '0'.* *DBRP is set based on the calculated time quanta and the Mcu clock being used.* *DSEG1 is the sum of the 'CanControllerFdBaudrateConfig/[]/CanControllerPropSeg' and the 'CanControllerFdBaudrateConfig/[]/CanControllerSeg1' configured subtracted by 1.* *DSEG2 is set as the value of container 'CanControllerFdBaudrateConfig/[]/CanControllerSeg2' subtracted by 1.* *DSJW is set based on the value configured in container 'CanControllerFdBaudrateConfig/[]/CanControllerSyncJumpWidth' subtracted with 1.*	
Example(s)	**Action**	**Generated output**
	Configure a controller with 'CanControllerFDBaudRate' as 2500. With DBRP value 2, DSJW value 0, DTSEG1 value 2 and DTSEG2 value 2.	`0x20200U`

1.2.8.2 Member: CanTrcvDelyComp

Table 118 CanTrcvDelyComp

Name	CanTrcvDelyComp	
Type	uint32	
Description	The configuration for transreciever delay compensation offset value for the CAN FD.	
Verification method	The generated structure member is present in Can_17_McmCan_kFDBaudrateConfigCore<x>[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the settings from the container 'CanControllerFdBaudrateConfig/[]/CanControllerTrcvDelayCompensationOffset/[]'. *Note: If the node 'CanControllerFdBaudrateConfig/*[1]/CanControllerTrcvDelayCompensationOffset/[]' does not exist this value is printed as a '0'.*	
Example(s)	**Action**	**Generated output**
	Configure a controller with 'CanControllerFDBaudRate' as 2500, with 'CanControllerTrcvDelayCompensationOffset' as 40	`16U`

	Configure a controller with ‘CanControllerFDBaudRate’ as 2500, with ‘CanControllerTrcvDelayCompensationOffset‘ as 0	0U
	Configure a controller with ‘CanControllerFDBaudRate’ as 2500, with node ‘CanControllerTrcvDelayCompensationOffset‘ not existing	0U

1.2.8.3 Member: CanTxBRSEnable

Table 119 CanTxBRSEnable

Name	CanTxBRSEnable	
Type	boolean	
Description	Enables/Disables the support of bit rate switch during CAN FD mode.	
Verification method	The generated structure member is present in Can_17_McmCan_kFDBaudrateConfigCore<x>[_<variant>] structure. The structure member is generated as a 'TRUE' when container 'CanControllerFdBaudrateConfig/[]/CanControllerTxBitRateSwitch' value is set as 'True' else it is generated as 'FALSE'.	
Example(s)	**Action**	**Generated output**
	Configure a controller with 'CanControllerFDBaudRate' as 2500, with 'CanControllerTxBitRateSwitch' set as True	`TRUE`
	Configure a controller with 'CanControllerFDBaudRate' as 2500, with 'CanControllerTxBitRateSwitch' set as False	`FALSE`

1.2.9 Structure: Can_17_McmCan_kTxHwObjectConfigCore<x>[_<variant>][Total Number Of Transmit Hardware Objects Configured For Controllers allocated to Core<x>]

Table 120 Can_17_McmCan_kTxHwObjectConfigCore<x>[_<variant>][Total Number Of Transmit Hardware Objects Configured For Controllers allocated to Core<x>]

Name	Can_17_McmCan_kTxHwObjectConfigCore<x>[_<variant>][Total Number Of Transmit Hardware Objects Configured For Controllers allocated to Core<x>]
Type	Can_17_McmCan_TxHwObjectConfigType
Description	Configuration structure of CAN driver for all different standard/mixed/ extended ID write hardware objects configured allocated to the CAN controllers belonging to Core <x> which will be referenced in core specific configuration structure. (x ranges from 0 to 5)
Verification method	The generated file has this structure if at least one of the controller's assigned to Core <x> is having associated hardware object with configuration of 'CanObjectType' as 'TRANSMIT'. <Variant> indicates the name of the post-build variant. For a variant aware configuration the structure name is appended with the variant name. For variant unaware configuration <variant> is ignored. This structure captures the configuration details of all the transmit hardware object associated with CAN controllers allocated to this core with respect to read action. The structure element 'CanFdPaddValue' shall be generated only if atleast one element is present in the list 'CanFdPaddingValue' for atleast one transmit hardware object in the CAN driver.

	The structure element 'CanTrigTxStatus' shall be generated only if atleast one element is present in the list 'CanTriggerTransmitEnable' for atleast one transmit hardware object in the CAN driver.	
Example(s)	**Action**	**Generated output**

	Configure 3 transmit hardware objects associated with controller1 associated with core0 with no CanFdPadding configured and hardware object Id 52 with no CanTriggerTransmit configuration.	`static const Can_17_McmCan_TxHwObjectConfigType \`
		`Can_17_McmCan_kTxHwObjectConfigCore0[10] =`
		`{`
		`{50U, 1U, 1U, CAN_17_MCMCAN_ID_EXTENDED, CAN_17_MCMCAN_TX_DED_BUFFER},`
		` {51U, 2U, 1U, CAN_17_MCMCAN_ID_MIXED, CAN_17_MCMCAN_TX_DED_BUFFER},`
		` {52U, 3U, 1U, CAN_17_MCMCAN_ID_STANDARD, CAN_17_MCMCAN_TX_DED_BUFFER}`
		`};`

	Configure 3 transmit hardware objects associated with controller1 associated with core0 with hardware object with Id 50 having CanFdPadding configured as 0xFF and hardware object Id 52 with CanTriggerTransmit enabled.	`static const Can_17_McmCan_TxHwObjectConfigType \`
		`Can_17_McmCan_kTxHwObjectConfigCore0[10] =`
		`{`
		`{50U, 1U, 1U, 255U, CAN_17_MCMCAN_ID_EXTENDED, CAN_17_MCMCAN_TX_DED_BUFFER, FALSE},`
		` {51U, 2U, 1U, 0U, CAN_17_MCMCAN_ID_MIXED, CAN_17_MCMCAN_TX_DED_BUFFER, FALSE},`
		` {52U, 3U, 1U, 0U, CAN_17_MCMCAN_ID_STANDARD, CAN_17_MCMCAN_TX_DED_BUFFER, TRUE},`
		`};`

1.2.9.1 Member: CanTxHwObjId

Table 121 CanTxHwObjId

Name	CanTxHwObjId	
Type	Can_HwHandleType	
Description	The hardware object's Id of the transmit CAN hardware object.	
Verification method	The generated structure member is present in Can_17_McmCan_kTxHwObjectConfigCore<x>[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the value of 'CanObjectId' for this transmit hardware object.	
Example(s)	**Action**	**Generated output**

	Configure a transmit hardware object with 'CanObjectId' set as 28	`28U`

1.2.9.2 Member: CanTxBuffIndx

Table 122 CanTxBuffIndx

Name	CanTxBuffIndx	
Type	uint8	
Description	The index in the transmit buffer where the transmit hardware object shall be placed.	
Verification method	The generated structure member is present in Can_17_McmCan_kTxHwObjectConfigCore<x>[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the buffer index offset for a particular transmit hardware object. For transmit hardware objects configured as a tx queue ('CanHwObjectCount' greater than 1), this value is printed '255'.	
Example(s)	**Action**	**Generated output**
	Configure a transmit hardware object with Id 54 which is the 1st transmit hardware object of the controller and with 'CanHwObjectCount' as 1	`0U`
	Configure a transmit hardware object with Id 59 which is the 6th transmit hardware object of the controller and with 'CanHwObjectCount' as 1	`5U`
	Configure a transmit hardware object with Id 60 which is the 7th transmit hardware object of the controller and with 'CanHwObjectCount' as 10	`255U`

1.2.9.3 Member: HwControllerId

Table 123 HwControllerId

Name	HwControllerId	
Type	uint8	
Description	The CanControllerId associated with the transmit CAN hardware object.	
Verification method	The generated structure member is present in Can_17_McmCan_kTxHwObjectConfigCore<x>[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the value of 'CanControllerId' of the controller which is referenced by this hardware object.	
Example(s)	**Action**	**Generated output**
	Configure a transmit hardware object associated with CanController1	`1U`

1.2.9.4 Member: CanFdPaddValue

Table 124 CanFdPaddValue

Name	CanFdPaddValue	
Type	uint8	
Description	The CAN FD padding value for the transmit hardware object in use.	
Verification method	The generated structure member is present in Can_17_McmCan_kTxHwObjectConfigCore<x>[_<variant>] structure. The structure member is generated as a numeric value. This value is generated from configuration in the container 'CanFdPaddingValue'. *Note: The structure element 'CanFdPaddValue' shall be generated in Can_17_McmCan_kTxHwObjectConfigCore<x>[_<variant>] structures only if atleast one transmit hardware object is configured with 'CanFdPaddingValue' else this element shall not be generated in the structure.* *This structure member is generated with the value '0' if the list CanFdPaddingValue does not have any elements.*	
Example(s)	**Action**	**Generated output**
	Configure a transmit hardware object with 'CanFdPaddingValue' set as 234.	`234U`
	Configure a transmit hardware object with 'CanFdPaddingValue' set as 0.	`0U`

1.2.9.5 Member: CanTxHwObjIdType

Table 125 CanTxHwObjIdType

Name	CanTxHwObjIdType	
Type	uint8	
Description	The type of CAN ID that the transmit CAN hardware object uses.	
Verification method	The generated structure member is present in Can_17_McmCan_kTxHwObjectConfigCore<x>[_<variant>] structure. The structure member is generated as a macro type. This value is generated based on the value of 'CanIdType' for this hardware object.	
Example(s)	**Action**	**Generated output**
	Configure a transmit hardware object with 'CanIdType' as 'MIXED'.	`CAN_17_MCMCAN_ID_MIXED`
	Configure a transmit hardware object with 'CanIdType' as 'STANDARD'.	`CAN_17_MCMCAN_ID_STANDARD`
	Configure a transmit hardware object with 'CanIdType' as 'EXTENDED'.	`CAN_17_MCMCAN_ID_EXTENDED`

1.2.9.6 Member: CanTxBufferType

Table 126 CanTxBufferType

Name	CanTxBufferType
Type	Can_17_McmCan_TxBufferType

Description	The type of buffer that the transmit CAN hardware object uses.	
Verification method	The generated structure member is present in Can_17_McmCan_kTxHwObjectConfigCore<x>[_<variant>] structure. The structure member is generated as a macro type. This value is generated based on the value of ‘CanHwObjectCount’ for this hardware object.	
Example(s)	**Action**	**Generated output**
	Configure a transmit hardware object with ‘CanHwObjectCount’ as 1.	`CAN_17_MCMCAN_TX_DED_BUFFER`
	Configure a transmit hardware object with ‘CanHwObjectCount’ as 10(with ‘CanHwObjectCount’ value greater than 1).	`CAN_17_MCMCAN_TX_QUEUE`

1.2.9.7 Member: CanTrigTxStatus

Table 127 CanTrigTxStatus

Name	CanTrigTxStatus	
Type	boolean	
Description	Enables/Disables the support of trigger transmit for the transmit hardware object.	
Verification method	The generated structure member is present in Can_17_McmCan_kTxHwObjectConfigCore<x>[_<variant>] structure. The structure member is generated as a ‘TRUE’ when the container ‘CanTriggerTransmitEnable’ is set as ‘True’ else it is generated as ‘FALSE’. *Note: The structure element ‘CanTrigTxStatus’ shall be generated in all Can_17_McmCan_kTxHwObjectConfigCore<x>[_<variant>] structures only if atleast one transmit hardware object is configured with an element in the list ‘CanTriggerTransmitEnable’ else this element shall not be generated in the structure.* *This structure member is generated with the value ‘FALSE’ if the list CanTriggerTransmitEnable does not have any elements.*	
Example(s)	**Action**	**Generated output**
	Configure a transmit hardware object with ‘CanTriggerTransmitEnable’ set as True.	`TRUE`
	Configure a transmit hardware object with ‘CanTriggerTransmitEnable’ set as False.	`FALSE`

1.2.10 Structure: Can_17_McmCan_kSIDFilterConfigCore<x>[_<variant>][Total Number

Of Standard/Mixed Type Receive Hardware Objects Configured For Controllers allocated to Core<x>]

Table 128 Can_17_McmCan_kSIDFilterConfigCore<x>[_<variant>][Total Number Of Standard/Mixed Type Receive Hardware Objects Configured For Controllers allocated to Core<x>]

Name	Can_17_McmCan_kSIDFilterConfigCore<x>[_<variant>][Total Number Of Standard/Mixed Type Receive Hardware Objects Configured For Controllers allocated to Core<x>]	
Type	Can_17_McmCan_SIDFilterConfigType	
Description	Configuration structure of CAN driver for all different standard/mixed ID read hardware objects configured allocated to the CAN controllers belonging to Core <x> which will be referenced in core specific configuration structure. (x ranges from 0 to 5)	
Verification method	The generated file has this structure if at least one of the controller's assigned to Core <x> is having associated hardware object with configuration of 'CanIdType' as 'STANDARD' or 'MIXED' and 'CanObjectType' as 'RECEIVE'. <Variant> indicates the name of the post-build variant. For a variant aware configuration the structure name is appended with the variant name. For variant unaware configuration <variant> is ignored. This structure captures the standard Id related receive hardware object details associated with CAN controllers allocated to this core with respect to read action. *Note: When an CAN hardware object has 'CanIdType' type as MIXED, this means it holds both the extended and standard characteristics and hence this hardware object shall have elemenst in both Can_17_McmCan_kSIDFilterConfigCore<x>[_<variant>] and Can_17_McmCan_kXIDFilterConfigCore<x>[_<variant>] filter configuration structures.*	
Example(s)	**Action**	**Generated output**

	- Configure 5 hardware objects of receive type associated with controller 0 allocated to core0.
	- Configure hardware objects 1 and 3 as EXTENDED type, 4 as MIXED type and 0 and 2 as STANDARD type.
	- Configure hardware objects 4 as with hardware object count as 10.
	- Configure Icom with matching receive hardware object 'CanHwFilterCode' and 'CanHwFilterMask' configurations matching hardware objects 0 and 3.
		`static const Can_17_McmCan_SIDFilterConfigType \`
		
		`Can_17_McmCan_kSIDFilterConfigCore0[14] =`
		
		`{`
		`{0xbffe0000U, 0U, 0U, CAN_17_MCMCAN_RX_DED_BUFFER,TRUE },`
		`{0xbc000002U, 2U, 0U, CAN_17_MCMCAN_RX_DED_BUFFER,FALSE },`
		`{0xbff00004U, 4U, 0U, CAN_17_MCMCAN_RX_FIFO0,FALSE }`
		`};`

1.2.10.1 Member: CanSIDFiltEleS0

Table 129 CanSIDFiltEleS0

Name	CanSIDFiltEleS0
Type	Uint32
Description	The standard Id elements receive hardware filters structure 'S0' value.
Verification method	The generated structure member is present in Can_17_McmCan_kSIDFilterConfigCore<x>[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the value to be filled in the filter configuration 'S0' for this standard/mixed id to be set as a compare hardware filter. *Note: The S0 frame is formed by the settings of the SFT(bits 30-31), SFEC(bits 27-29), SFID1(bits 16-26) and SFID2(bits 0-10) values. The reserved bits are kept as '0'.* *SFEC is set to '7' for dedicated Rx buffer, '1' for RXFIFO0 and '2' for RXFIFO1 used.* *SFT is set to a fixed value'2'.* *SFID1 is based on the Hwfilter value set* *SFID2 holds the dedicated buffer index for dedicated rx buffers and the HwFilterMask value for the rx FIFOs configured.*

Example(s)	Action	Generated output
	Configure a Mixed id with filter value 2016 and being 4th dedicated Rx for the controller with number of hardware objects as 1.	`0xBFE00003U`
	Configure a standard id with filter value 2016 and mask value 1024 being the 1st hardware object for the controller with number of hardware objects as 10 (>1).	`0x8FE00400U`

1.2.10.2 Member: CanSidHwObjId

Table 130 CanSidHwObjId

Name	CanSidHwObjId	
Type	Can_HwHandleType	
Description	The hardware object's Id of the standard/mixed type CAN hardware object.	
Verification method	The generated structure member is present in Can_17_McmCan_kSIDFilterConfigCore<x>[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the value of 'CanObjectId' for this standard/mixed id hardware filter.	
Example(s)	**Action**	**Generated output**
	Configure a receive hardware object of standard type with 'CanObjectId' set as 14	`14U`
	Configure a receive hardware object of mixed type with 'CanObjectId' set as 2	`2U`

1.2.10.3 Member: HwControllerId

Table 131 HwControllerId

Name	HwControllerId	
Type	uint8	
Description	The CanControllerId associated with the standard/mixed type CAN hardware object.	
Verification method	The generated structure member is present in Can_17_McmCan_kSIDFilterConfigCore<x>[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the value of 'CanControllerId' of the controller which is referenced by this hardware object.	
Example(s)	**Action**	**Generated output**
	Configure a receive hardware object of mixed Id type associated with CanController1	`1U`
	Configure a receive hardware object of standard Id type associated with CanController3	`3U`

1.2.10.4 Member: CanSidBufferType

Table 132 CanSidBufferType

Name	CanSidBufferType	
Type	Can_17_McmCan_RxBufferType	
Description	The type of receive buffer that the standard/mixed type CAN hardware object uses.	
Verification method	The generated structure member is present in Can_17_McmCan_kSIDFilterConfigCore<x>[_<variant>] structure. The structure member is generated as a macro type. This value is generated based on the value of 'CanHwObjectCount' for this hardware object.	
Example(s)	**Action**	**Generated output**
	Configure a receive hardware object of standard type with 'CanHwObjectCount' as 1.	`CAN_17_MCMCAN_RX_DED_BUFFER`
	Configure the receive hardware object of mixed type with 'CanHwObjectCount' as 13 (with this being the first receive hardware object with 'CanHwObjectCount' value greater than 1 for this associated controller).	`CAN_17_MCMCAN_RX_FIFO0`
	Configure the receive hardware object of standard type with 'CanHwObjectCount' as 22 (with this being the second receive hardware object with 'CanHwObjectCount' value greater than 1 for this associated controller).	`CAN_17_MCMCAN_RX_FIFO1`

1.2.10.5 Member: CanSidPNSupport

Table 133 CanSidPNSupport

Name	CanSidPNSupport	
Type	boolean	
Description	Enables/Disables the support of this standard/mixed Id filter support in pretended network mode.	
Verification method	The generated structure member is present in Can_17_McmCan_kSIDFilterConfigCore<x>[_<variant>] structure. The structure member is generated as a 'TRUE' when the filter configuration for the standard/mixed type of hardware object is a direct match or superset with any of the filter configurations of Icom containers 'CanIcomMessageId' and 'CanIcomMessageIdMask' else it is generated as 'FALSE'. *Note: This structure element is generated only when CanPublicIcomSupport is 'True' else this structure element is not generated.*	
Example(s)	**Action**	**Generated output**
	• Configure a receive hardware object of standard type with	`TRUE`

‘CanHwObjectCount’ as 1 and with filter value matching the Icom filter configuration. • Configure CanPublicIcomSupport as True.	
• Configure a receive hardware object of mixed type ‘CanHwObjectCount’ as 15. (with this being the first receive hardware object with ‘CanHwObjectCount’ value greater than 1 for this associated controller) and with filter value a superset the Icom filter configuration. • Configure CanPublicIcomSupport as True.	`TRUE`
• Configure a receive hardware object of standard type with ‘CanHwObjectCount’ as 36 (with this being the second receive hardware object with ‘CanHwObjectCount’ value greater than 1 for this associated controller) and with filter value a subset the Icom filter configuration. • Configure CanPublicIcomSupport as True.	`FALSE`

1.2.11 Structure: Can_17_McmCan_kXIDFilterConfigCore<x>[_<variant>][Total Number Of Extended/Mixed Type Receive Hardware Objects Configured For Controllers allocated to Core<x>]

Table 134 Can_17_McmCan_kXIDFilterConfigCore<x>[_<variant>][Total Number Of Extended/Mixed Type Receive Hardware Objects Configured For Controllers allocated to Core<x>]

Name	Can_17_McmCan_kXIDFilterConfigCore<x>[_<variant>][Total Number Of Extended/Mixed Type Receive Hardware Objects Configured For Controllers allocated to Core<x>]
Type	Can_17_McmCan_XIDFilterConfigType
Description	Configuration structure of CAN driver for all different extended/mixed ID read hardware objects configured allocated to the CAN controllers belonging to Core <x> which will be referenced in core specific configuration structure. (x ranges from 0 to 5)
Verification method	The generated file has this structure if at least one of the controller’s assigned to Core <x> is having associated hardware object with configuration of ‘CanIdType’ as ‘EXTENDED’ or ’MIXED’ and ‘CanObjectType’ as ‘RECEIVE’. <Variant> indicates the name of the post-build variant. For a variant aware configuration the structure name is appended with the variant name. For variant unaware configuration <variant> is ignored. This structure captures the

	extended Id related receive hardware object details associated with CAN controllers allocated to this core with respect to read action. *Note: When an CAN hardware object has 'CanIdType' type as MIXED, this means it holds both the extended and standard characteristics and hence this hardware object shall have elemenst in both Can_17_McmCan_kSIDFilterConfigCore<x>[_<variant>] and Can_17_McmCan_kXIDFilterConfigCore<x>[_<variant>] filter configuration structures.*	
Example(s)	**Action**	**Generated output**

	• Configure 5 hardware objects of receive type associated with controller 0 allocated to core0.
	• Configure hardware objects 1 and 3 as EXTENDED type, 4 as MIXED type and 0 and 2 as STANDARD type.
	• Configure hardware objects 4 as with hardware object count as 10.
	• Configure Icom with matching receive hardware object 'CanHwFilterCode' and 'CanHwFilterMask' configurations matching hardware objects 0 and 3.
		`static const Can_17_McmCan_XIDFilterConfigType \`
		`Can_17_McmCan_kXIDFilterConfigCore0[3] =`
		`{`
		` {0xffffffffU, 0x80000001U, 1U, 0U, CAN_17_MCMCAN_RX_DED_BUFFER, FALSE},`
		` {0xf0000000U, 0x80000003U, 3U, 0U, CAN_17_MCMCAN_RX_DED_BUFFER, TRUE},`
		` {0xffc00000U, 0x80000004U, 4U, 0U, CAN_17_MCMCAN_RX_FIFO0, FALSE}`
		`};`

1.2.11.1 Member: CanXIDFiltEleF0

Table 135 CanXIDFiltEleF0

Name	CanXIDFiltEleF0	
Type	Uint32	
Description	The extended Id elements receive hardware filters structure 'F0' value.	
Verification method	The generated structure member is present in Can_17_McmCan_kXIDFilterConfigCore<x>[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the value to be filled in the filter configuration 'F0' for this extended/mixed id to be set as a compare hardware filter. *Note: The F0 frame is formed by the settings of the EFEC(bits 29-31) and EFID1(bits 0-28) values.* *EFEC is set to '7' for dedicated Rx buffer, '1' for RXFIFO0 and '2' for RXFIFO1 used.* *EFID1 is based on the Hwfilter value set*	
Example(s)	**Action**	**Generated output**

	Configure a Mixed id with filter value 2016 and being 4th dedicated Rx for the controller with number of hardware objects as 1.	`0xE00007E0U`
	Configure a extended id with filter value 2016 and mask value 1024 being the 1st hardware object for the controller with number of hardware objects as 10 (>1).	`0x200007E0U`

1.2.11.2 Member: CanXIDFiltEleF1

Table 136 CanXIDFiltEleF1

Name	CanXIDFiltEleF1	
Type	Uint32	
Description	The extended Id elements receive hardware filters structure ‘F1’ value.	
Verification method	The generated structure member is present in Can_17_McmCan_kXIDFilterConfigCore<x>[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the value to be filled in the filter configuration ‘F0’ for this extended/mixed id to be set as a compare hardware filter. *Note: The F0 frame is formed by the settings of the EFT(bits 30-31) and EFID2(bits 0-28) values. The reserved bits are kept as ‘0’.* *SFT is set to a fixed value’2’.* *EFID2 holds the dedicated buffer index for dedicated rx buffers and the HwFilterMask value for the rx FIFOs configured.*	
Example(s)	**Action**	**Generated output**
	Configure a Mixed id with filter value 2016 and being 4th dedicated Rx for the controller with number of hardware objects as 1.	`0x80000003U`
	Configure a extended id with filter value 2016 and mask value 1024 being the 1st hardware object for the controller with number of hardware objects as 10 (>1).	`0x80000400U`

1.2.11.3 Member: CanXidHwObjId

Table 137 CanXidHwObjId

Name	CanXidHwObjId
Type	Can_HwHandleType
Description	The hardware object Id of the extended/mixed type CAN hardware object.
Verification method	The generated structure member is present in Can_17_McmCan_kXIDFilterConfigCore<x>[_<variant>] structure. The structure member is

	generated as a numeric value. This value is generated based on the value of 'CanObjectId' for this extended/mixed id hardware filter.	
Example(s)	**Action**	**Generated output**
	Configure a receive hardware object of extended type with 'CanObjectId' set as 10	`10U`
	Configure a receive hardware object of mixed type with 'CanObjectId' set as 20	`20U`

1.2.11.4 Member: HwControllerId

Table 138 HwControllerId

Name	HwControllerId	
Type	Uint8	
Description	The CanControllerId associated with the extended/mixed type CAN hardware object.	
Verification method	The generated structure member is present in Can_17_McmCan_kXIDFilterConfigCore<x>[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the value of 'CanControllerId' of the controller which is referenced by this hardware object.	
Example(s)	**Action**	**Generated output**
	Configure a receive hardware object of extended Id type associated with CanController0	`0U`
	Configure a receive hardware object of mixed Id type associated with CanController5	`5U`

1.2.11.5 Member: CanXidBufferType

Table 139 CanXidBufferType

Name	CanXidBufferType	
Type	Can_17_McmCan_RxBufferType	
Description	The type of receive buffer that the extended/mixed type CAN hardware object uses.	
Verification method	The generated structure member is present in Can_17_McmCan_kXIDFilterConfigCore<x>[_<variant>] structure. The structure member is generated as a macro type. This value is generated based on the value of 'CanHwObjectCount' for this hardware object.	
Example(s)	**Action**	**Generated output**
	Configure a receive hardware object of extended type with 'CanHwObjectCount' as 1.	`CAN_17_MCMCAN_RX_DED_BUFFER`
	Configure the receive hardware object of extended type with 'CanHwObjectCount' as 15 (with this being the first receive hardware object with	`CAN_17_MCMCAN_RX_FIFO0`

	'CanHwObjectCount' value greater than 1 for this associated controller).	
	Configure the receive hardware object of mixed type with 'CanHwObjectCount' as 33 (with this being the second receive hardware object with 'CanHwObjectCount' value greater than 1 for this associated controller).	`CAN_17_MCMCAN_RX_FIFO1`

1.2.11.6 Member: CanXidPNSupport

Table 140 CanXidPNSupport

Name	CanXidPNSupport	
Type	boolean	
Description	Enables/Disables the support of this extended/mixed Id filter support in pretended network mode.	
Verification method	The generated structure member is present in Can_17_McmCan_kXIDFilterConfigCore<x>[_<variant>] structure. The structure member is generated as a 'TRUE' when the filter configuration for the extended/mixed type of hardware object is a direct match or superset with any of the filter configurations of Icom containers 'CanIcomMessageId' and 'CanIcomMessageIdMask' else it is generated as 'FALSE'. *Note: This structure element is generated only when CanPublicIcomSupport is 'True' else this structure element is not generated.*	
Example(s)	**Action**	**Generated output**
	• Configure a receive hardware object of extended type with 'CanHwObjectCount' as 1 and with filter value matching the Icom filter configuration. • Configure CanPublicIcomSupport as True.	`TRUE`
	• Configure a receive hardware object of extended type 'CanHwObjectCount' as 12 (with this being the first receive hardware object with 'CanHwObjectCount' value greater than 1 for this associated controller) and with filter value not matching/superset the Icom filter configuration. • Configure CanPublicIcomSupport as True.	`FALSE`
	• Configure a receive hardware object of mixed type with	`FALSE`

	'CanHwObjectCount' as 10 (with this being the second receive hardware object with 'CanHwObjectCount' value greater than 1 for this associated controller) and with filter value subset of the Icom filter configuration. • Configure CanPublicIcomSupport as True.	

1.2.12 Array: Can_17_McmCan_kHthPeriodIndexCore<x>[_<variant>][CAN_17_MCMCAN_NOOF_RX_TX_PERIODS_CONFIG]

Table 141 Can_17_McmCan_kHthPeriodIndexCore<x>[_<variant>][CAN_17_MCMCAN_NOOF_RX_TX_PERIODS_CONFIG]

<table>
<tr><td><b>Name</b></td><td colspan="2">Can_17_McmCan_kHthPeriodIndexCore<x>[_<variant>][CAN_17_MCMCAN_NOOF_RX_TX_PERIODS_CONFIG]</td></tr>
<tr><td><b>Type</b></td><td colspan="2">Can_17_McmCan_HthPeriodIndexType</td></tr>
<tr><td><b>Description</b></td><td colspan="2">Configuration structure of CAN driver for all different periods configured allocated to the CAN controllers belonging to Core <x> which will be referenced in core specific configuration structure. (x ranges from 0 to 5)</td></tr>
<tr><td><b>Verification method</b></td><td colspan="2">The generated file has this structure if at least one of the controller's assigned to Core <x> is having 'CanTxProcessing' as 'POLLING' and elements in 'CanMainFunctionRWPeriods' is greater than 1. <Variant> indicates the name of the post-build variant. For a variant aware configuration the structure name is appended with the variant name. For variant unaware configuration <variant> is ignored. This structure captures at the 'CanMainFunctionRWPeriods' indexes if they are used by this core for write operation actions.

<i>Note: The 'CanMainFunctionRWPeriods' index that is not used by the Core<x> shall print a deafult value as 255.</i></td></tr>
<tr><td rowspan="2"><b>Example(s)</b></td><td><b>Action</b></td><td><b>Generated output</b></td></tr>
<tr><td>• Configure 2 read-write periods in 'CanMainFunctionRWPeriods'
• Configure a Controller1 associated with core 0 with a transmit type hardware object reffering to CanMainFunctionRWPeriods index 1 as the period of polling.</td><td><code>static const
Can_17_McmCan_HthPeriodIndexType \
  Can_17_McmCan_kHthPeriodIndexCore0[2]
=
{255,0};</code></td></tr>
</table>

1.2.13 Structure: Can_17_McmCan_kHthMaskObjectConfigCore<x>[_<variant>][Total Number Of Transmit Hardware Objects Configured For Controllers with Tranmission Event 'POLLING' allocated to Core<x>]

Table 142 Can_17_McmCan_kHthMaskObjectConfigCore<x>[_<variant>][Total Number Of Transmit Hardware Objects Configured For Controllers with Tranmission Event 'POLLING' allocated to Core<x>]

Name	Can_17_McmCan_kHthMaskObjectConfigCore<x>[_<variant>][Total Number Of Transmit Hardware Objects Configured For Controllers with Tranmission Event 'POLLING' allocated to Core<x>]	
Type	Can_17_McmCan_HthMaskObjectConfigType	
Description	Configuration structure of CAN driver for all different write hardware objects configured allocated to the CAN controllers belonging to Core <x> which will be referenced in core specific configuration structure. (x ranges from 0 to 5)	
Verification method	The generated file has this structure if at least one of the controller's assigned to Core <x> is having 'CanTxProcessing' as 'POLLING' and elements in 'CanMainFunctionRWPeriods' is greater than 1. <Variant> indicates the name of the post-build variant. For a variant aware configuration the structure name is appended with the variant name. For variant unaware configuration <variant> is ignored. This structure captures the transmit objects details associated with CAN controllers allocated to this core with respect to read action.	
Example(s)	**Action**	**Generated output**

	• Configure 4 CAN controllers.
	• Configure CAN conytroller 2 and 3 to core 5 and controller 0 and 1 to core 0.
	• Configure 5 different read- write operation periods.
	• Configure 54-58 transmit hardware objects in CanControllerId 2
	• Configure 59-63 transmit hardware objects in CanControllerId 3
		`static const`
		`Can_17_McmCan_HthMaskObjectConfigType \`
		
		`Can_17_McmCan_kHthMaskObjectConfigCore5[10]`
		`=`
		`{`
		`  {0x1U, 54U, 2U},`
		`  {0x2U, 55U, 2U},`
		`  {0x4U, 56U, 2U},`
		`  {0x8U, 57U, 2U},`
		`  {0x10U, 58U, 2U},`
		`  {0x1U, 59U, 3U},`
		`  {0x2U, 60U, 3U},`
		`  {0x4U, 61U, 3U},`
		`  {0x8U, 62U, 3U},`
		`  {0x1f0U, 63U, 3U}`
		`};`

1.2.13.1 Member: CanTxBufferMaskvalue

Table 143 CanTxBufferMaskvalue

Name	CanTxBufferMaskvalue	
Type	Uint32	
Description	The calculated buffer value to locate the Tx messages associated with the given transmit hardware object.	
Verification method	The generated structure member is present in Can_17_McmCan_kHthMaskObjectConfigCore<x>[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the expected value of the write operation successful buffer when the message is transmitted by using the particular transmit hardware object. For transmit hardware objects configured as a tx queue ('CanHwObjectCount' greater than 1), this value is printed as the last index after all the dedicated tx buffers for the controller has been exhausted.	
Example(s)	**Action**	**Generated output**
	Configure a transmit hardware object with Id 54 which is the 1st transmit hardware object of the controller	0x1U
	Configure a transmit hardware object with Id 56 which is the 3rd transmit hardware object of the controller	0x4U
	Configure a transmit hardware object with Id 58 which is of queue type wiyj queue size of 2 and the maximum number of dedicated transmits are 4 for the controller	0x10U

1.2.13.2 Member: CanPerHthHwObjId

Table 144 CanPerHthHwObjId

Name	CanPerHthHwObjId	
Type	Can_HwHandleType	
Description	The CanObjectId for this CAN hardware object.	
Verification method	The generated structure member is present in Can_17_McmCan_kHthMaskObjectConfigCore<x>[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the value of 'CanObjectId' for this hardware object.	
Example(s)	**Action**	**Generated output**
	Configure a transmit hardware object with Id 54 which is the 1st transmit hardware object of the controller	54U
	Configure a transmit hardware object with Id 63 which is the 10th transmit hardware object of the controller	63U

1.2.13.3 Member: HwControllerId

Table 145 HwControllerId

Name	HwControllerId	
Type	uint8	
Description	The CanControllerId associated with this CAN hardware object.	
Verification method	The generated structure member is present in Can_17_McmCan_kHthMaskObjectConfigCore<x>[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the value of 'CanControllerId' of the controller which is referenced by this hardware object.	
Example(s)	**Action**	**Generated output**
	Configure a transmit hardware object with Id 54 which is the 1st transmit hardware object of the CanController0	`0U`
	Configure a transmit hardware object with Id 54 which is the 1st transmit hardware object of the CanController10	`10`

1.2.14 Structure: Can_17_McmCan_kPeriodHthMaskConfigCore<x>[_<variant>]>][Total Number Of different CanMainFunctionRWPeriod Reffered By Transmit Hardware Objects Configured For Controllers with Tranmission Event 'POLLING' allocated to Core<x>]

Table 146 Can_17_McmCan_kPeriodHthMaskConfigCore<x>[_<variant>][Total Number Of different CanMainFunctionRWPeriod Reffered By Transmit Hardware Objects Configured For Controllers with Tranmission Event 'POLLING' allocated to Core<x>]

Name	Can_17_McmCan_kPeriodHthMaskConfigCore<x>[_<variant>][Total Number Of different CanMainFunctionRWPeriod Reffered By Transmit Hardware Objects Configured For Controllers with Tranmission Event 'POLLING' allocated to Core<x>]	
Type	Can_17_McmCan_PeriodHthMaskConfigType	
Description	Configuration structure of CAN driver for all different write periods configured allocated to the CAN controllers belonging to Core <x> which will be referenced in core specific configuration structure. (x ranges from 0 to 5)	
Verification method	The generated file has this structure if at least one of the controller's assigned to Core <x> is having 'CanTxProcessing' as 'POLLING' and elements in 'CanMainFunctionRWPeriods' is greater than 1. <Variant> indicates the name of the post-build variant. For a variant aware configuration the structure name is appended with the variant name. For variant unaware configuration <variant> is ignored. This structure captures the start and end indexes for the detail period related hardware object configurations for of each read-write periods associated with the controllers of this core.	
Example(s)	**Action**	**Generated output**

	Configure all 23 transmit hardware objects of controller 0 associated with core0 to the same period 0.	`static const`
		`Can_17_McmCan_PeriodHthMaskConfigType \`
		
		`Can_17_McmCan_kPeriodHthMaskConfigCore0[1]`
		`=`
		`{`
		`{0U, 10U}`
		`};`

1.2.14.1 Member: CanPerHthStartIndx

Table 147 CanPerHthStartIndx

Name	CanPerHthStartIndx	
Type	Can_HwHandleType	
Description	The start index offset for the current core specific multi- period write configuration.	
Verification method	The generated structure member is present in Can_17_McmCan_kPeriodHthMaskConfigCore<x>[_<variant>] structure. The structure member is generated as a numeric value. This value the array offset in structure Can_17_McmCan_kHthMaskObjectConfigCore<x>[_<variant>] holding the information for the transmit hardware objects associated with the current core specific period index.	
Example(s)	**Action**	**Generated output**
	• Configure 20 hardware objects with transmit type and associated with same controller. • Configure 15 hardware objects associated with period0 configured to core 5. • Configure 5 hardware objects associated with period2 configured to core 5. The generated CanPerHthStartIndx at array index 0 (for period0 0th period for core0) for core 0	`0U`
	• Configure 20 hardware objects with transmit type and associated with same controller. • Configure 15 hardware objects associated with period0 configured to core 5. • Configure 5 hardware objects associated with period2 configured to core 5. The generated CanPerHthStartIndx at array index 1 (for period2 1st period for core0) for core 0	`15U`

1.2.14.2 Member: CanPerHthEndIndx

Table 148 CanPerHthEndIndx

Name	CanPerHthEndIndx	
Type	Can_HwHandleType	
Description	The number of hardware object elements associated with the current core specific multi-period write	
Verification method	The generated structure member is present in Can_17_McmCan_kPeriodHthMaskConfigCore<x>[_<variant>] structure. The structure member is generated as a numeric value. This value the number of hardware object elements associated with the current core specific multi- period write in structure Can_17_McmCan_kHthMaskObjectConfigCore<x>[_<variant>].	
Example(s)	**Action**	**Generated output**
	• Configure 20 hardware objects with transmit type and associated with same controller. • Configure 15 hardware objects associated with period0 configured to core 5. • Configure 5 hardware objects associated with period2 configured to core 5. The generated CanPerHthEndIndx at array index 0 (for period0 0th period for core0) for core 0	`15U`
	• Configure 20 hardware objects with transmit type and associated with same controller. • Configure 15 hardware objects associated with period0 configured to core 5. • Configure 5 hardware objects associated with period2 configured to core 5. The generated CanPerHthEndIndx at array index 1 (for period2 1st period for core0) for core 0	`5U`

1.2.15 Array: Can_17_McmCan_kHrhPeriodIndexCore<x>[_<variant>][CAN_17_MCMCAN_NOOF_RX_TX_PERIODS_CONFIG]

Table 149 Can_17_McmCan_kHrhPeriodIndexCore<x>[_<variant>][CAN_17_MCMCAN_NOOF_RX_TX_PERIODS_CONFIG]

Name	Can_17_McmCan_kHrhPeriodIndexCore<x>[_<variant>][CAN_17_MCMCAN_NOOF_RX_TX_PERIODS_CONFIG]	
Type	Can_17_McmCan_HrhPeriodIndexType	
Description	Configuration structure of CAN driver for all different periods configured allocated to the CAN controllers belonging to Core <x> which will be referenced in core specific configuration structure. (x ranges from 0 to 5)	
Verification method	The generated file has this structure if at least one of the controller's assigned to Core <x> is having 'CanRxProcessing' as 'POLLING' and elements in 'CanMainFunctionRWPeriods' is greater than 1. <Variant> indicates the name of the post-build variant. For a variant aware configuration the structure name is appended with the variant name. For variant unaware configuration <variant> is ignored. This structure captures at the 'CanMainFunctionRWPeriods' indexes if they are used by this core for read operation actions. *Note: The 'CanMainFunctionRWPeriods' index that is not used by the Core<x> shall print a deafult value as 255.*	
Example(s)	**Action**	**Generated output**

	• Configure 2 read-write periods in 'CanMainFunctionRWPeriods'
	• Configure a Controller0 associated with core 0 with a rercieve type hardware object reffering to CanMainFunctionRWPeriods index 0 as the period of polling.
		`static const`
		`Can_17_McmCan_HrhPeriodIndexType \`
		`  Can_17_McmCan_kHrhPeriodIndexCore0[2] =`
		`{0,255};`

1.2.16 Structure: Can_17_McmCan_kHrhMaskObjectConfigCore<x>[_<variant>][Total Number Of Receive Hardware Objects Configured For Controllers with Reception Event 'POLLING' allocated to Core<x>]

Table 150 Can_17_McmCan_kHrhMaskObjectConfigCore<x>[_<variant>][Total Number Of Receive Hardware Objects Configured For Controllers with Reception Event 'POLLING' allocated to Core<x>]

Name	Can_17_McmCan_kHrhMaskObjectConfigCore<x>[_<variant>][Total Number Of Receive Hardware Objects Configured For Controllers with Reception Event 'POLLING' allocated to Core<x>]
Type	Can_17_McmCan_HrhMaskObjectConfigType
Description	Configuration structure of CAN driver for all different read hardware objects configured allocated to the CAN controllers belonging to Core <x> which will be referenced in core specific configuration structure. (x ranges from 0 to 5)

Verification method	The generated file has this structure if at least one of the controller's assigned to Core <x> is having 'CanRxProcessing' as 'POLLING' and elements in 'CanMainFunctionRWPeriods' is greater than 1. <Variant> indicates the name of the post-build variant. For a variant aware configuration the structure name is appended with the variant name. For variant unaware configuration <variant> is ignored. This structure captures the recive objects details associated with the CAN controllers allocated to this core with respect to read action.	
Example(s)	**Action**	**Generated output**

	• Configure 2 CAN controllers to core 0.
	• Configure 5 different read- write operation periods.
	• Configure 0-10 receive hardware objects in CanControllerId 0
	• Configure 11-21 receive hardware objects in CanControllerId 1
		`static const`
		`Can_17_McmCan_HrhMaskObjectConfigType \`
		
		`Can_17_McmCan_kHrhMaskObjectConfigCore0[22] =`
		
		`{`
		`  {0x1U, 0x0U, 0U, 0U, CAN_17_MCMCAN_RX_DED_BUFFER},`
		`  {0x2U, 0x0U, 1U, 0U, CAN_17_MCMCAN_RX_DED_BUFFER},`
		`  {0x4U, 0x0U, 2U, 0U, CAN_17_MCMCAN_RX_DED_BUFFER},`
		`  {0x8U, 0x0U, 3U, 0U, CAN_17_MCMCAN_RX_DED_BUFFER},`
		`  {0x10U, 0x0U, 4U, 0U, CAN_17_MCMCAN_RX_DED_BUFFER},`
		`  {0x20U, 0x0U, 5U, 0U, CAN_17_MCMCAN_RX_DED_BUFFER},`
		`  {0x40U, 0x0U, 6U, 0U, CAN_17_MCMCAN_RX_DED_BUFFER},`
		`  {0x80U, 0x0U, 7U, 0U, CAN_17_MCMCAN_RX_DED_BUFFER},`
		`  {0x100U, 0x0U, 8U, 0U, CAN_17_MCMCAN_RX_DED_BUFFER},`
		`  {0x200U, 0x0U, 9U, 0U, CAN_17_MCMCAN_RX_DED_BUFFER},`
		`  {0x400U, 0x0U, 10U, 0U, CAN_17_MCMCAN_RX_DED_BUFFER},`
		`  {0x1U, 0x0U, 11U, 1U, CAN_17_MCMCAN_RX_DED_BUFFER},`
		`  {0x2U, 0x0U, 12U, 1U, CAN_17_MCMCAN_RX_DED_BUFFER},`
		`  {0x4U, 0x0U, 13U, 1U, CAN_17_MCMCAN_RX_DED_BUFFER},`
		`  {0x8U, 0x0U, 14U, 1U, CAN_17_MCMCAN_RX_DED_BUFFER},`

		`{0x10U, 0x0U, 15U, 1U, CAN_17_MCMCAN_RX_DED_BUFFER},`
		`{0x20U, 0x0U, 16U, 1U, CAN_17_MCMCAN_RX_DED_BUFFER},`
		`{0x40U, 0x0U, 17U, 1U, CAN_17_MCMCAN_RX_DED_BUFFER},`
		`{0x80U, 0x0U, 18U, 1U, CAN_17_MCMCAN_RX_DED_BUFFER},`
		`{0x100U, 0x0U, 19U, 1U, CAN_17_MCMCAN_RX_DED_BUFFER},`
		`{0x200U, 0x0U, 20U, 1U, CAN_17_MCMCAN_RX_DED_BUFFER},`
		`{0x400U, 0x0U, 21U, 1U, CAN_17_MCMCAN_RX_DED_BUFFER}`
		`};`

1.2.16.1 Member: CanPerRxbufferMaskvalue0

Table 151 CanPerRxbufferMaskvalue0

Name	CanPerRxbufferMaskvalue0	
Type	Uint32	
Description	The calculated buffer value to locate the Rx messages (whose buffer location is between 0-31st location) associated with the given recieve hardware object.	
Verification method	The generated structure member is present in Can_17_McmCan_kHrhMaskObjectConfigCore<x>[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the expected value of the read operation successful buffer when the message is recieved by using the particular recieve hardware object. For recieve hardware objects configured as a Rx FIFO ('CanHwObjectCount' greater than 1), this value is printed as the '0'.	
Example(s)	**Action**	**Generated output**
	Configure a receive hardware object with Id 5 which is the 1st recieve hardware object of the controller	`0x1U`
	Configure a receive hardware object with Id 33 which is the 28th recieve hardware object of the controller	`0x10000000U`
	Configure a receive hardware object with Id 43 which is the 38th recieve hardware object of the controller	`0x0U`

1.2.16.2 Member: CanPerRxbufferMaskvalue1

Table 152 CanPerRxbufferMaskvalue1

Name	CanPerRxbufferMaskvalue1

Type	Uint32	
Description	The calculated buffer value to locate the Rx messages (whose buffer location is between 0-63rd location) associated with the given recieve hardware object.	
Verification method	The generated structure member is present in Can_17_McmCan_kHrhMaskObjectConfigCore<x>[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the expected value of the read operation successful buffer when the message is recieved by using the particular recieve hardware object. For recieve hardware objects configured as a Rx FIFO ('CanHwObjectCount' greater than 1), this value is printed as the '0'.	
Example(s)	**Action**	**Generated output**
	Configure a receive hardware object with Id 5 which is the 1st recieve hardware object of the controller	`0U`
	Configure a receive hardware object with Id 43 which is the 38th recieve hardware object of the controller	`0x00000040U`
	Configure a receive hardware object with Id 53 which is the 48th receive hardware object of the controller	`0x00020000U`

1.2.16.3 Member: CanPerHrhHwObjId

Table 153 CanPerHrhHwObjId

Name	CanPerHrhHwObjId	
Type	Can_HwHandleType	
Description	The CanObjectId for this CAN hardware object.	
Verification method	The generated structure member is present in Can_17_McmCan_kHrhMaskObjectConfigCore<x>[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the value of 'CanObjectId' for this hardware object.	
Example(s)	**Action**	**Generated output**
	Configure a transmit hardware object with Id 5 which is the 1st recieve hardware object of the controller	`5U`
	Configure a transmit hardware object with Id 25 which is the 10th recieve hardware object of the controller	`25`

1.2.16.4 Member: HwControllerId

Table 154 HwControllerId

Name	HwControllerId
Type	uint8
Description	The CanControllerId associated with this CAN hardware object.

Verification method	The generated structure member is present in Can_17_McmCan_kHrhMaskObjectConfigCore<x>[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the value of 'CanControllerId' of the controller which is referenced by this hardware object.	
Example(s)	**Action**	**Generated output**
	Configure a transmit hardware object with Id 10 which is the 10th recieve hardware object of the CanController0	`0U`
	Configure a transmit hardware object with Id 10 which is the 19th recieve hardware object of the CanController5	`5U`

1.2.16.5 Member: CanPerHrhBufferType

Table 155 CanPerHrhBufferType

Name	CanPerHrhBufferType	
Type	Can_17_McmCan_RxBufferType	
Description	The type of receive buffer that the CAN hardware object uses.	
Verification method	The generated structure member is present in Can_17_McmCan_kHrhMaskObjectConfigCore<x>[_<variant>] structure. The structure member is generated as a macro type. This value is generated based on the value of 'CanHwObjectCount' for this hardware object.	
Example(s)	**Action**	**Generated output**
	Configure a receive hardware object with 'CanHwObjectCount' as 1.	`CAN_17_MCMCAN_RX_DED_BUFFER`
	Configure the receive hardware object with 'CanHwObjectCount' as 10. (with this being the first receive hardware object with 'CanHwObjectCount' value greater than 1 for this associated controller).	`CAN_17_MCMCAN_RX_FIFO0`
	Configure the receive hardware object with 'CanHwObjectCount' as 20. (with this being the second receive hardware object with 'CanHwObjectCount' value greater than 1 for this associated controller).	`CAN_17_MCMCAN_RX_FIFO1`

1.2.17 Structure: Can_17_McmCan_kPeriodHrhMaskConfigCore<x>[_<variant>][Total Number Of different CanMainFunctionRWPeriod Reffered By Receive

Hardware Objects Configured For Controllers with Tranmission Event ‘POLLING’ allocated to Core<x>]

Table 156 Can_17_McmCan_kPeriodHrhMaskConfigCore<x>[_<variant>][Total Number Of different CanMainFunctionRWPeriod Reffered By Receive Hardware Objects Configured For Controllers with Tranmission Event ‘POLLING’ allocated to Core<x>]

Name	Can_17_McmCan_kPeriodHrhMaskConfigCore<x>[_<variant>][Total Number Of different CanMainFunctionRWPeriod Reffered By Receive Hardware Objects Configured For Controllers with Tranmission Event ‘POLLING’ allocated to Core<x>]	
Type	Can_17_McmCan_PeriodHrhMaskConfigType	
Description	Configuration structure of CAN driver for all different read periods configured allocated to the CAN controllers belonging to Core <x> which will be referenced in core specific configuration structure. (x ranges from 0 to 5)	
Verification method	The generated file has this structure if at least one of the controller’s assigned to Core <x> is having ‘CanRxProcessing’ as ‘POLLING’ and elements in ‘CanMainFunctionRWPeriods’ is greater than 1. <Variant> indicates the name of the post-build variant. For a variant aware configuration the structure name is appended with the variant name. For variant unaware configuration <variant> is ignored. This structure captures the start and end indexes for the detail period related hardware object configurations for of each read-write periods associated with the controllers of this core.	
Example(s)	**Action**	**Generated output**

	Configure all 23 receive hardware objects of controller 0 associated with core0 to the same period 0.	`static const Can_17_McmCan_PeriodHrhMaskConfigType \`
		`Can_17_McmCan_kPeriodHrhMaskConfigCore0[1] =`
		`{`
		`{0U, 22U}`
		`};`

1.2.17.1 Member: CanPerHrhStartIndx

Table 157 CanPerHrhStartIndx

Name	CanPerHrhStartIndx	
Type	Can_HwHandleType	
Description	The start index offset for the current core specific multi- period read configuration.	
Verification method	The generated structure member is present in Can_17_McmCan_kPeriodHrhMaskConfigCore<x>[_<variant>] structure. The structure member is generated as a numeric value. This value the array offset in structure Can_17_McmCan_kHrhMaskObjectConfigCore<x>[_<variant>] holding the information for the recieve hardware objects associated with the current core specific period index.	
Example(s)	**Action**	**Generated output**

	Configure 20 hardware objects with receive type and associated with same controller. Configure 15 hardware objects associated with period0 configured to core 5. Configure 5 hardware objects associated with period2 configured to core 5. The generated CanPerHrhStartIndx at array index 0 (for period0 0th period for core0) for core 0	`0U`
	Configure 20 hardware objects with receive type and associated with same controller. Configure 15 hardware objects associated with period0 configured to core 5. Configure 5 hardware objects associated with period2 configured to core 5. The generated CanPerHrhStartIndx at array index 1 (for period2 1st period for core0) for core 0	`15U`

1.2.17.2 Member: CanPerHrhEndIndx

Table 158 CanPerHrhEndIndx

Name	CanPerHrhEndIndx	
Type	Can_HwHandleType	
Description	The number of hardware object elements associated with the current core specific multi-period read	
Verification method	The generated structure member is present in Can_17_McmCan_kPeriodHrhMaskConfigCore<x>[_<variant>] structure. The structure member is generated as a numeric value. This value the number of hardware object elements associated with the current core specific multi- period read in structure Can_17_McmCan_kHrhMaskObjectConfigCore<x>[_<variant>].	
Example(s)	**Action**	**Generated output**
	Configure 20 hardware objects with receive type and associated with same controller. Configure 15 hardware objects associated with period0 configured to core 5. Configure 5 hardware objects associated with period2 configured to core 5. The generated CanPerHrhEndIndx at array index 0 (for period0 0th period for core0) for core 0	`15U`

	Configure 20 hardware objects with receive type and associated with same controller. Configure 15 hardware objects associated with period0 configured to core 5. Configure 5 hardware objects associated with period2 configured to core 5. The generated CanPerHrhEndIndx at array index 1 (for period2 1st period for core0) for core 0	`5U`

1.2.18 Structure: Can_17_McmCan_kMcmCanModuleConfig[_<variant>][Total Number Of Different Kernels the Configured Controllers belong to in the CAN driver]

Table 159 Can_17_McmCan_kMcmCanModuleConfig[_<variant>][Total Number Of Different Kernels the Configured Controllers belong to in the CAN driver]

Name	Can_17_McmCan_kMcmCanModuleConfig[_<variant>][Total Number Of Different Kernels the Configured Controllers belong to in the CAN driver]	
Type	Can_17_McmCan_McmModuleConfigType	
Description	Configuration structure of CAN driver for general kernel level configuration.	
Verification method	<Variant> indicates the name of the post-build variant. For a variant aware configuration the structure name is appended with the variant name. For variant unaware configuration <variant> is ignored. The array size of this structure type is dependenent on the total number of CAN kernels used by the configuration. This value is derived from the value in the container 'CanBaseAddressPtr'. The structure captures the base address of the kernels and the details on the nodes in the kernels used.	
Example(s)	**Action**	**Generated output**

- Configure 3 CAN controllers.
- Configure CAN controller0 of kernel0 node 3
- Configure controller1 of kernel2 node 0
- Configure controller2 of kernel 1 node 2

```
static const
Can_17_McmCan_McmModuleConfigType \

Can_17_McmCan_kMcmCanModuleConfig[3] =
{
  {
    /* The Global Base address of Kernel
module */
    (volatile Ifx_CAN*) 0xf0200000U,
    /* The CAN node is enabled or not
within the kernel*/
    {
      /* Node 0 of kernel enable state
*/
      FALSE,
      /* Node 1 of kernel enable state
*/
      FALSE,
      /* Node 2 of kernel enable state
*/
      FALSE,
      /* Node 3 of kernel enable state
*/
      TRUE
    }
  },


  {
    /* The Global Base address of Kernel
module */
    (volatile Ifx_CAN*) 0xf0210000U,
    /* The CAN node is enabled or not
within the kernel*/
    {
      /* Node 0 of kernel enable state
*/
      FALSE,
      /* Node 1 of kernel enable state
*/
      FALSE,
      /* Node 2 of kernel enable state
*/
```

```
   TRUE,
   /* Node 3 of kernel enable state
*/
   FALSE
  }
 },


 {
  /* The Global Base address of Kernel
module */
  (volatile Ifx_CAN*) 0xf0220000U,
  /* The CAN node is enabled or not
within the kernel*/
  {
   /* Node 0 of kernel enable state
*/
   TRUE,
   /* Node 1 of kernel enable state
*/
   FALSE,
   /* Node 2 of kernel enable state
*/
   FALSE,
   /* Node 3 of kernel enable state
*/
   FALSE
  }
 }
};
```

1.2.18.1 Member: CanBaseAddressPtr

Table 160 CanBaseAddressPtr

Name	CanBaseAddressPtr	
Type	Ifx_CAN*	
Description	The Kernel start address for the CAN controllers configured.	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanModuleConfig[_<variant>] structure. The structure member is generated as a memory address pointe value. This value is generated based on the start address of the different kernels associated with the CAN controllers configured.	
Example(s)	**Action**	**Generated output**

	• Configure 2 CAN controllers. • Configure CAN controller0 of kernel0 node 3 • Configure controller1 of kernel2 node 0 The generated CanBaseAddressPtr at array index 0	`(volatile Ifx_CAN*) 0xf0200000U`
	• Configure 2 CAN controllers. • Configure CAN controller0 of kernel0 node 3 • Configure controller1 of kernel2 node 0 The generated CanBaseAddressPtr at array index 1	`(volatile Ifx_CAN*) 0xf0220000U`

1.2.18.2 Member: CanUsedHwCfgIndx[CAN_17_MCMCAN_NOOF_NODES_PER_KERNEL]

Table 161 CanUsedHwCfgIndx[CAN_17_MCMCAN_NOOF_NODES_PER_KERNEL]

Name	CanUsedHwCfgIndx[CAN_17_MCMCAN_NOOF_NODES_PER_KERNEL]	
Type	boolean	
Description	Enables/Disables each node in the kernel	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanModuleConfig[_<variant>] structure. The structure member is generated as a 'TRUE' when the node Id in the kernel is configured else it is generated as 'FALSE'. *Note: The node Id is is identified by the array index.*	
Example(s)	**Action**	**Generated output**

	• Configure 3 CAN controllers.
	• Configure CAN controller0 of kernel0 node 3
	• Configure controller1 of kernel2 node 0
	• Configure controller2 of kernel2 node 2
	The generated CanUsedHwCfgIndx at array index 0
		`{`
		`/* Node 0 of kernel enable state */`
		`FALSE,`
		`/* Node 1 of kernel enable state */`
		`FALSE,`
		`/* Node 2 of kernel enable state */`
		`FALSE,`
		`/* Node 3 of kernel enable state */`
		`TRUE`
		`}`

	• Configure 3 CAN controllers.
	• Configure CAN controller0 of kernel0 node 3
	• Configure controller1 of kernel2 node 0
	• Configure controller2 of kernel2 node 2
	The generated CanUsedHwCfgIndx at array index 1
		`{`
		`/* Node 0 of kernel enable state */`
		`TRUE,`
		`/* Node 1 of kernel enable state */`
		`FALSE,`
		`/* Node 2 of kernel enable state */`
		`TRUE,`
		`/* Node 3 of kernel enable state */`
		`FALSE`
		`}`

1.2.19 Structure: Can_17_McmCan_kMcmCanPhyContIndexConfig[_<variant>][Total Number Of Controllers That Can be Configured in the CAN driver]

Table 162 Can_17_McmCan_kMcmCanPhyContIndexConfig[_<variant>][Total Number Of Controllers That Can be Configured in the CAN driver]

Name	Can_17_McmCan_kMcmCanPhyContIndexConfig[_<variant>][Total Number Of Controllers That Can be Configured in the CAN driver]
Type	Can_17_McmCan_PhyControllerIndexType
Description	Configuration structure capturing the mapping details of the configured CanControllerId, core specifc controller offset and core assigned info for all the CAN controllers present in the CAN.

Verification method	<Variant> indicates the name of the post-build variant. For a variant aware configuration the structure name is appended with the variant name. For variant unaware configuration <variant> is ignored. This generated configuration array of structure maps to all the CAN controllers available in CAN.The size of the array is fixed to the maximum number of controllers available in the CAN. The generated structure contains the details of the configured CanControllerId, core specifc controller offset and core assigned for each of the controllers available in the CAN. The array of 12 is arranged with the 4 nodes (0 to 3) of the 3 kernels(0 to 2) mapped one after the other. *Note: At the index of the CAN physical controllers that are not used the structure elements shall be set to a default value of 255.*	
Example(s)	**Action**	**Generated output**

	- Configure 3 CAN controllers.
	- Configure CAN controller0 of kernel0 node 3 allocated to core2
	- Configure controller1 of kernel2 node 0 allocated to core2
	- Configure controller2 of kernel 1 node 2 allocated to core1
		`static const`
		`Can_17_McmCan_PhyControllerIndexType \`
		
		`Can_17_McmCan_kMcmCanPhyContIndexConfig[12]`
		`=`
		`{`
		`  {255,255,255},`
		`  {255,255,255},`
		`  {255,255,255},`
		`  {0,0,2},`
		`  {255,255,255},`
		`  {255,255,255},`
		`  {2,0,1},`
		`  {255,255,255},`
		`  {1,1,2},`
		`  {255,255,255},`
		`  {255,255,255},`
		`  {255,255,255}`
		`};`

1.2.19.1 Member: CanPLogicContIndex

Table 163 CanPLogicContIndex

Name	CanPLogicContIndex
Type	uint8
Description	The CanControllerId for this physical CAN controller.

Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanPhyContIndexConfig[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the value of the container 'CanControllerId' for a given physical CAN controller (derived from the 'CanBaseAddressPtr' value of a controller) configured. *Note: This structure element for a physical CAN controller that is not configured shall be printed with the value 255.*	
Example(s)	**Action**	**Generated output**
	• Configure 3 CAN controllers. • Configure CAN controller0 of kernel0 node 3 allocated to core2 • Configure controller1 of kernel2 node 0 allocated to core2 • Configure controller2 of kernel 1 node 2 allocated to core1 The generated CanPLogicContIndex at array index (kernelID(0)*4 + nodeId(3)) = 3	`0U`
	• Configure 3 CAN controllers. • Configure CAN controller0 of kernel0 node 3 allocated to core2 • Configure controller1 of kernel2 node 0 allocated to core2 • Configure controller2 of kernel 1 node 2 allocated to core1 The generated CanPLogicContIndex at array index (kernelID(1)*4 + nodeId(2)) = 6	`2U`
	• Configure 3 CAN controllers. • Configure CAN controller0 of kernel0 node 3 allocated to core2 • Configure controller1 of kernel2 node 0 allocated to core2 • Configure controller2 of kernel 1 node 2 allocated to core1 The generated CanPLogicContIndex at array index (kernelID(1)*4 + nodeId(1)) = 5 (Non configured physical index).	`255U`

1.2.19.2 Member: CanPCoreSpecContIndex

Table 164 CanPCoreSpecContIndex

Name	CanPCoreSpecContIndex

Type	uint8	
Description	The core specific controller offset for this physical CAN controller.	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanPhyContIndexConfig[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the core specific offset of the physical CAN controller. *Note: This structure element for a physical CAN controller that is not configured shall be printed with the value 255.*	
Example(s)	**Action**	**Generated output**
	• Configure 3 CAN controllers. • Configure CAN controller0 of kernel0 node 3 allocated to core2 • Configure controller1 of kernel2 node 0 allocated to core2 • Configure controller2 of kernel 1 node 2 allocated to core1 The generated CanPCoreSpecContIndex at array index (kernelID(0)*4 + nodeId(3)) = 3	`0U`
	• Configure 3 CAN controllers. • Configure CAN controller0 of kernel0 node 3 allocated to core2 • Configure controller1 of kernel2 node 0 allocated to core2 • Configure controller2 of kernel 1 node 2 allocated to core1 The generated CanPLogicContIndex at array index (kernelID(1)*4 + nodeId(2)) = 6	`0U`
	• Configure 3 CAN controllers. • Configure CAN controller0 of kernel0 node 3 allocated to core2 • Configure controller1 of kernel2 node 0 allocated to core2 • Configure controller2 of kernel 1 node 2 allocated to core1 The generated CanPCoreSpecContIndex at array index (kernelID(1)*4 + nodeId(1)) = 5 (Non configured physical index).	`255U`

1.2.19.3 Member: CanPCoreAssigned

Table 165 CanPCoreAssigned

Name	CanPCoreAssigned	
Type	uint8	
Description	The core to which this Physical CAN controller is assigned to.	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanPhyContIndexConfig[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the core to which the physical CAN controller is assigned to. *Note: This structure element for a physical CAN controller that is not configured shall be printed with the value 255.*	
Example(s)	**Action**	**Generated output**
	• Configure 3 CAN controllers. • Configure CAN controller0 of kernel0 node 3 allocated to core2 • Configure controller1 of kernel2 node 0 allocated to core2 • Configure controller2 of kernel 1 node 2 allocated to core1 The generated CanPCoreSpecContIndex at array index (kernelID(0)*4 + nodeId(3)) = 3	`2U`
	• Configure 3 CAN controllers. • Configure CAN controller0 of kernel0 node 3 allocated to core2 • Configure controller1 of kernel2 node 0 allocated to core2 • Configure controller2 of kernel 1 node 2 allocated to core1 The generated CanPLogicContIndex at array index (kernelID(1)*4 + nodeId(2)) = 6	`1U`
	• Configure 3 CAN controllers. • Configure CAN controller0 of kernel0 node 3 allocated to core2 • Configure controller1 of kernel2 node 0 allocated to core2 • Configure controller2 of kernel 1 node 2 allocated to core1 The generated CanPCoreSpecContIndex at array index (kernelID(1)*4 + nodeId(1)) = 5 (Non configured physical index).	`255U`

1.2.20 Structure: Can_17_McmCan_kMcmCanLogicContIndexConfig[_<variant>][CAN_17_MCMCAN_NOOF_CONTROLLER]

Table 166 Can_17_McmCan_kMcmCanLogicContIndexConfig[_<variant>][CAN_17_MCMCAN_NOOF_CONTROLLER]

<table>
<tr><td><b>Name</b></td><td colspan="2">Can_17_McmCan_kMcmCanLogicContIndexConfig[_<variant>][CAN_17_MCMCAN_NOOF_CONTROLLER]</td></tr>
<tr><td><b>Type</b></td><td colspan="2">Can_17_McmCan_LogicalControllerIndexType</td></tr>
<tr><td><b>Description</b></td><td colspan="2">Configuration structure capturing the mapping details of the core allocation, core specifc controller offset and kernel and node index that the specific configured controller Id.</td></tr>
<tr><td><b>Verification method</b></td><td colspan="2">&lt;Variant> indicates the name of the post-build variant. For a variant aware configuration the structure name is appended with the variant name. For variant unaware configuration <variant> is ignored.
This generated configuration array of structure maps only to the CAN controllers configured and indexing is based on the CAN controller Id.The size of the array generated is based on the controllers configured.
The generated structure contains the details of the core allocation, core specifc controller offset and kernel and node index for the logical(configured) controller Id.</td></tr>
<tr><td><b>Example(s)</b></td><td><b>Action</b></td><td><b>Generated output</b></td></tr>
<tr><td></td><td>• Configure 3 CAN controllers.
• Configure CAN controller0 of kernel0 node 3 allocated to core2
• Configure controller1 of kernel2 node 0 allocated to core2
• Configure controller2 of kernel 1 node 2 allocated to core1</td><td><code>static const Can_17_McmCan_LogicalControllerIndexType \</code>
<code>Can_17_McmCan_kMcmCanLogicContIndexConfig[3]</code>
<code>=</code>
<code>{</code>
<code> {2,0,3,0},</code>
<code> {2,1,0,2},</code>
<code> {1,0,2,1}</code>
<code>};</code></td></tr>
</table>

1.2.20.1 Member: CanLCoreAssigned

Table 167 CanLCoreAssigned

<table>
<tr><td><b>Name</b></td><td colspan="2">CanLCoreAssigned</td></tr>
<tr><td><b>Type</b></td><td colspan="2">uint8</td></tr>
<tr><td><b>Description</b></td><td colspan="2">The core to which this CanControllerId is assigned to.</td></tr>
<tr><td><b>Verification method</b></td><td colspan="2">The generated structure member is present in Can_17_McmCan_kMcmCanLogicContIndexConfig[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the core to which the CanControllerId is allocated.</td></tr>
<tr><td><b>Example(s)</b></td><td><b>Action</b></td><td><b>Generated output</b></td></tr>
<tr><td></td><td>• Configure 3 CAN controllers.</td><td><code>4U</code></td></tr>
</table>

	• Configure CAN controller0 allocated to core4 • Configure CAN controller1 allocated to core5 • Configure CAN controller2 allocated to core4 The generated CanLCoreAssigned at array index 0	
	• Configure 3 CAN controllers. • Configure CAN controller0 allocated to core4 • Configure CAN controller1 allocated to core5 • Configure CAN controller2 allocated to core4 The generated CanLCoreAssigned at array index 1	`5U`
	• Configure 3 CAN controllers. • Configure CAN controller0 allocated to core4 • Configure CAN controller1 allocated to core5 • Configure CAN controller2 allocated to core4 The generated CanLCoreAssigned at array index 2	`4U`

1.2.20.2 Member: CanLCoreSpecContIndex

Table 168 CanLCoreSpecContIndex

Name	CanLCoreSpecContIndex	
Type	uint8	
Description	The core specific controller offset for this CanControllerId.	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanLogicContIndexConfig[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the core specific offset of the controller to which the CanControllerId is allocated.	
Example(s)	**Action**	**Generated output**
	• Configure 3 CAN controllers. • Configure CAN controller0 allocated to core4 • Configure CAN controller1 allocated to core5	`0U`

	• Configure CAN controller2 allocated to core4 The generated CanLCoreSpecContIndex at array index 0	
	• Configure 3 CAN controllers. • Configure CAN controller0 allocated to core4 • Configure CAN controller1 allocated to core5 • Configure CAN controller2 allocated to core4 The generated CanLCoreSpecContIndex at array index 1	`1U`
	• Configure 3 CAN controllers. • Configure CAN controller0 allocated to core4 • Configure CAN controller1 allocated to core5 • Configure CAN controller2 allocated to core4 The generated CanLCoreSpecContIndex at array index 2	`0U`

1.2.20.3 Member: CanLContPhyIndex

Table 169 CanLContPhyIndex

Name	CanLContPhyIndex	
Type	uint8	
Description	The physical node Id in the kernel to which this CanControllerId is assigned to.	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanLogicContIndexConfig[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the index of the node in the kernel that the CanControllerId is allocated.	
Example(s)	**Action**	**Generated output**
	• Configure 3 CAN controllers. • Configure CAN controller0 of kernel0 node 3 allocated to core2 • Configure controller1 of kernel2 node 0 allocated to core2 • Configure controller2 of kernel 1 node 2 allocated to core1 The generated CanLContPhyIndex at array index 0	`3U`
	• Configure 3 CAN controllers.	`0U`

	• Configure CAN controller0 of kernel0 node 3 allocated to core2 • Configure controller1 of kernel2 node 0 allocated to core2 • Configure controller2 of kernel 1 node 2 allocated to core1 The generated CanLContPhyIndex at array index 1	
	• Configure 3 CAN controllers. • Configure CAN controller0 of kernel0 node 3 allocated to core2 • Configure controller1 of kernel2 node 0 allocated to core2 • Configure controller2 of kernel 1 node 2 allocated to core1 The generated CanLContPhyIndex at array index 2	`2U`

1.2.20.4 Member: CanLKerPhyIndex

Table 170 CanLKerPhyIndex

Name	CanLKerPhyIndex	
Type	uint8	
Description	The physical kernel Id to which this CanControllerId is assigned to.	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanLogicContIndexConfig[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the index of the kernel that the CanControllerId is allocated.	
Example(s)	**Action**	**Generated output**
	• Configure 3 CAN controllers. • Configure CAN controller0 of kernel0 node 3 allocated to core2 • Configure controller1 of kernel2 node 0 allocated to core2 • Configure controller2 of kernel 1 node 2 allocated to core1 The generated CanLKerPhyIndex at array index 0	`0U`
	• Configure 3 CAN controllers. • Configure CAN controller0 of kernel0 node 3 allocated to core2 • Configure controller1 of kernel2 node 0 allocated to core2	`2U`

	- Configure controller2 of kernel 1 node 2 allocated to core1 The generated CanLKerPhyIndex at array index 1	
	- Configure 3 CAN controllers. - Configure CAN controller0 of kernel0 node 3 allocated to core2 - Configure controller1 of kernel2 node 0 allocated to core2 - Configure controller2 of kernel 1 node 2 allocated to core1 The generated CanLKerPhyIndex at array index 2	`1U`

1.2.21 Structure: Can_17_McmCan_kMcmCanHthIndexConfig[_<variant>][Total Number Of Transmit Hardware Objects Configured For Controllers In the CAN Driver]

Table 171 Can_17_McmCan_kMcmCanHthIndexConfig[_<variant>][Total Number Of Transmit Hardware Objects Configured For Controllers In the CAN Driver]

Name	Can_17_McmCan_kMcmCanHthIndexConfig[_<variant>][Total Number Of Transmit Hardware Objects Configured For Controllers In the CAN Driver]	
Type	Can_17_McmCan_HthIndexType	
Description	Configuration structure capturing the core allocation, controller allocation and core specific Hth indexing for the count of the transmit hardware object offset.	
Verification method	The generated file has this structure if at least one hardware object of 'CanObjectType' set as 'TRANSMIT' is configured. <Variant> indicates the name of the post-build variant. For a variant aware configuration the structure name is appended with the variant name. For variant unaware configuration <variant> is ignored. This generated configuration array of structure maps at the array element index (which is the hardware object Id of the transmit object type subtracted by the highest hardware object Id of receive object type). The generated structure contains the details of the core and controller associated to that hardware object Id along with the offset of the transmit hardware object with respect to core it is assigned to. *Note: This generated configuration structure does not change with the number of hardware objects configured as RECEIVE type.*	
Example(s)	**Action**	**Generated output**

	• Configure 3 CAN controllers
	• Allocate controller 0 and 1 to Core 2 and controller 2 to core 0.
	• Configure 20 hardware objects.
	• Configure 3 hardware objects as RECEIVE and 4 hardware objects as TRANSMIT for controller 0.
	• Configure 5 hardware objects as RECEIVE for controller 1.
	• Configure 5 hardware objects as RECEIVE and 3 hardware objects as TRANSMIT for controller 2.
	The array Can_17_McmCan_kMcmCanHthIndexConfig will be generated with 7 (4 TRANSMIT hardware objects in controller0 + 3 TRANSMIT hardware objects in controller2) as the array size.
		`static const`
		`Can_17_McmCan_HthIndexType \`
		`Can_17_McmCan_kMcmCanHthIndexConfig[7] =`
		`{`
		`  {2,0,0},`
		`  {2,0,1},`
		`  {2,0,2},`
		`  {2,0,3},`
		`  {0,1,0},`
		`  {0,1,1},`
		`  {0,1,2}`
		`};`

1.2.21.1 Member: CanHthCoreAssigned

Table 172 CanHthCoreAssigned

Name	CanHthCoreAssigned	
Type	uint8	
Description	The core to which this Hth is assigned to.	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanHthIndexConfig[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the core to which the specific transmit hardware objects referenced controller is allocated.	
Example(s)	**Action**	**Generated output**
	• Configure 2 controllers • Allocate controller0 to core 2 and controller1 to core 0. • Configure 5 hardware objects with 3 hardware objects of type TRANSMIT	`2U`

	• Configure the transmit hardware object 2 and 3 referencing controller0, transmit hardware object 4 referencing controller1 and the receive hardware objects 0 and 1 to controller 0. The generated CanHthCoreAssigned at array index 0	
	• Configure 2 controllers • Allocate controller0 to core 2 and controller1 to core 0. • Configure 5 hardware objects with 3 hardware objects of type TRANSMIT • Configure the transmit hardware object 2 and 3 referencing controller0, transmit hardware object 4 referencing controller1 and the receive hardware objects 0 and 1 to controller 0. The generated CanHthCoreAssigned at array index 1	`2U`
	• Configure 2 controllers • Allocate controller0 to core 2 and controller1 to core 0. • Configure 5 hardware objects with 3 hardware objects of type TRANSMIT • Configure the transmit hardware object 2 and 3 referencing controller0, transmit hardware object 4 referencing controller1 and the receive hardware objects 0 and 1 to controller 0. The generated CanHthCoreAssigned at array index 2	`0U`

1.2.21.2 Member: CanHthLogicContIndex

Table 173 CanHthLogicContIndex

Name	CanHthLogicContIndex	
Type	uint8	
Description	The controller to which this Hth is assigned to.	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanHthIndexConfig[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the controller to which the specific transmit hardware objects referenced.	
Example(s)	**Action**	**Generated output**

	• Configure 2 controllers • Allocate controller0 to core 2 and controller1 to core 0. • Configure 5 hardware objects with 3 hardware objects of type TRANSMIT • Configure the transmit hardware object 2 and 3 referencing controller0, transmit hardware object 4 referencing controller1 and the receive hardware objects 0 and 1 to controller 0. The generated CanHthLogicContIndex at array index 0	`0U`
	• Configure 2 controllers • Allocate controller0 to core 2 and controller1 to core 0. • Configure 5 hardware objects with 3 hardware objects of type TRANSMIT • Configure the transmit hardware object 2 and 3 referencing controller0, transmit hardware object 4 referencing controller1 and the receive hardware objects 0 and 1 to controller 0. The generated CanHthLogicContIndex at array index 1	`0U`
	• Configure 2 controllers • Allocate controller0 to core 2 and controller1 to core 0. • Configure 5 hardware objects with 3 hardware objects of type TRANSMIT • Configure the transmit hardware object 2 and 3 referencing controller0, transmit hardware object 4 referencing controller1 and the receive hardware objects 0 and 1 to controller 0. The generated CanHthLogicContIndex at array index 2	`1U`

1.2.21.3 Member: CanHthCoreSpecIndex

Table 174 CanHthCoreSpecIndex

Name	CanHthCoreSpecIndex
Type	Uint16

Description	The core specific Hth offset.	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanHthIndexConfig[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the transmit hardware object offset in the core specific transmit hardware objects configured.	
Example(s)	**Action**	**Generated output**
	• Configure 2 controllers • Allocate controller0 to core 2 and controller1 to core 0. • Configure 5 hardware objects with 3 hardware objects of type TRANSMIT • Configure the transmit hardware object 2 and 3 referencing controller0, transmit hardware object 4 referencing controller1 and the receive hardware objects 0 and 1 to controller 0. The generated CanHthCoreSpecIndex at array index 0	`0U`
	• Configure 2 controllers • Allocate controller0 to core 2 and controller1 to core 0. • Configure 5 hardware objects with 3 hardware objects of type TRANSMIT • Configure the transmit hardware object 2 and 3 referencing controller0, transmit hardware object 4 referencing controller1 and the receive hardware objects 0 and 1 to controller 0. The generated CanHthCoreSpecIndex at array index 1	`1U`
	• Configure 2 controllers • Allocate controller0 to core 2 and controller1 to core 0. • Configure 5 hardware objects with 3 hardware objects of type TRANSMIT • Configure the transmit hardware object 2 and 3 referencing controller0, transmit hardware object 4 referencing controller1 and the receive hardware objects 0 and 1 to controller 0. The generated CanHthCoreSpecIndex at array index 2	`0U`

1.2.22 Structure: Can_17_McmCan_kMcmCanIcomConfig[_<variant>][CAN_17_MCMCAN_NOOF_ICOM_CONFIGURATIONS]

Table 175 Can_17_McmCan_kMcmCanIcomConfig[_<variant>][CAN_17_MCMCAN_NOOF_ICOM_CONFIGURATIONS]

<table>
<tr><td><b>Name</b></td><td colspan="2">Can_17_McmCan_kMcmCanIcomConfig[_<variant>][CAN_17_MCMCAN_NOOF_ICOM_CONFIGURATIONS]</td></tr>
<tr><td><b>Type</b></td><td colspan="2">Can_17_McmCan_IcomConfigType</td></tr>
<tr><td><b>Description</b></td><td colspan="2">Configuration structure of CAN driver for general Icom related configuration.</td></tr>
<tr><td><b>Verification method</b></td><td colspan="2">The generated file has this structure if at least one Icom message is configured for any of the messages of Icom. <Variant> indicates the name of the post-build variant. For a variant aware configuration the structure name is appended with the variant name. For variant unaware configuration <variant> is ignored. The array size of this structure type is dependenent on the number of ‘CanIcomConfig’ configured in the Icom configuration set.
<i>Note: This array is generated only when CanPublicIcomSupport is ‘True’ else this array of structures is not generated.</i></td></tr>
<tr><td><b>Example(s)</b></td><td><b>Action</b></td><td><b>Generated output</b></td></tr>
<tr><td></td><td>• Configure 2 elements in CanIcomConfig
• Configure CanIcomConfig 1 with 5 messages in it and CanIcomWakeOnBusOff disabled
• Configure CanIcomConfig 2 with 3 messages in it and CanIcomWakeOnBusOff enabled.</td><td><code>static const Can_17_McmCan_IcomConfigType \</code>
<code>Can_17_McmCan_kMcmCanIcomConfig[CAN_17_MCMCAN_NOOF_ICOM_CONFIGURATIONS] =</code>
<code>{</code>
<code> {</code>
<code> 0U, 5U, FALSE</code>
<code> },</code>
<code> {</code>
<code> 5U, 3U, TRUE</code>
<code> }</code>
<code>};</code></td></tr>
</table>

1.2.22.1 Member: CanIcomFirstMsgIndx

Table 176 CanIcomFirstMsgIndx

Name	CanIcomFirstMsgIndx
Type	uint16
Description	The index in the array (that contains all the Icom message configurations in the CAN driver) form which the first message associated with this Icom configuration is present.

Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanIcomConfig[_<variant>] structure. The structure member is generated as a numeric value. This value is generates the index in the array Can_17_McmCan_kMcmCanIcomRxMsgConfig[_<variant>] where the first message of this 'CanIcomConfig' is available amongst all the messages configured in the Icom. *Note: This structure member is generated with the value '0' if the first message of the current CanIcomConfig is the first message configured overall for the Icom.*	
Example(s)	**Action**	**Generated output**
	• Configure 2 CanIcomConfig • Configure CanIcomConfig 1 with 4 message • Configure CanIcomConfig 2 with 10 message The CanIcomFirstMsgIndx for the message 1.	`0U`
	• Configure 2 CanIcomConfig • Configure CanIcomConfig 1 with 4 message • Configure CanIcomConfig 2 with 10 message The CanIcomFirstMsgIndx for the message 2.	`4U`

1.2.22.2 Member: CanIcomNoOfMsgIndx

Table 177 CanIcomNoOfMsgIndx

Name	CanIcomNoOfMsgIndx	
Type	uint16	
Description	The number of messages configured in CanIcomConfig_x/CanIcomRxMessage.	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanIcomConfig[_<variant>] structure. The structure member is generated as a numeric value. This value is the total number of messages configured for the CanIcomConfig. It is the count in the list 'CanIcomRxMessage'.	
Example(s)	**Action**	**Generated output**
	• Configure 2 CanIcomConfig • Configure CanIcomConfig 1 with 4 message • Configure CanIcomConfig 2 with 10 message The CanIcomNoOfMsgIndx for the message 1.	`4U`
	• Configure 2 CanIcomConfig	`10U`

	• Configure CanIcomConfig 1 with 4 message • Configure CanIcomConfig 2 with 10 message The CanIcomNoOfMsgIndx for the message 2.	

1.2.22.3 Member: CanIcomWakeOnBusOff

Table 178 CanIcomWakeOnBusOff

Name	CanIcomWakeOnBusOff	
Type	boolean	
Description	Enables/Disables wake up detection in Icom at bus off	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanIcomConfig[_<variant>] structure. The structure member is generated as a 'TRUE' when the container 'CanIcomWakeOnBusOff' is set as 'True' else it is generated as 'FALSE'.	
Example(s)	**Action**	**Generated output**
	Configure Icom message with CanIcomWakeOnBusOff as 'True'	`TRUE`
	onfigure Icom message with CanIcomWakeOnBusOff as 'False'	`FALSE`

1.2.23 Structure: Can_17_McmCan_kMcmCanIcomRxMsgConfig[_<variant>][CAN_17_MCMCAN_NOOF_ICOM_MSGCONFIGURATIONS]

Table 179 Can_17_McmCan_kMcmCanIcomRxMsgConfig[_<variant>][CAN_17_MCMCAN_NOOF_ICOM_MSGCONFIGURATIONS]

Name	Can_17_McmCan_kMcmCanIcomRxMsgConfig[_<variant>][CAN_17_MCMCAN_NOOF_ICOM_MSGCONFIGURATIONS]	
Type	Can_17_McmCan_IcomRxMsgConfigType	
Description	Configuration structure of CAN driver for all Icom messages related configurations.	
Verification method	The generated file has this structure if at least one Icom message is configured for any of the messages of Icom. <Variant> indicates the name of the post-build variant. For a variant aware configuration the structure name is appended with the variant name. For variant unaware configuration <variant> is ignored. The array size of this structure type is dependenent on the total number of messages configured in the Icom configuration set. *Note: This array is generated only when CanPublicIcomSupport is 'True' else this array of structures is not generated.*	
Example(s)	**Action**	**Generated output**

	• Configure 3 messages in Icom.
	• Message 1 with message Id type being extended with no message Id mask, with DLC 5, Icom length error disabled, no message counter and 10 signals configured in it.
	• Message 2 with message Id type being standard, with message Id mask of 1024, with DLC 8, Icom length error enabled message counter value set to 10 and no signal configured in it.
	• Message 3 with message Id type being standard with no message Id mask, with DLC 8, Icom length error enabled, message counter as 5 and 1 signal configured in it.
		`static const`
		`Can_17_McmCan_IcomRxMsgConfigType \`
		
		`Can_17_McmCan_kMcmCanIcomRxMsgConfig[CAN_17_MCMCAN_NOOF_ICOM_MSGCONFIGURATIONS] =`
		`{`
		`{268435456U, 0U, 0U, 5U, 0U,  10U, FALSE},`
		`{2046U, 1024U, 10U, 8U, 0U,  0U, TRUE},`
		`{1028U, 0U, 5U, 8U, 10U,  1U, TRUE}`
		`};`

1.2.23.1 Member: CanIcomMsgId

Table 180 CanIcomMsgId

Name	CanIcomMsgId
Type	Can_IdType
Description	The CAN message Id for the Icom message
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanIcomRxMsgConfig[_<variant>] structure. The structure member is generated as a numeric value. This value is generated from the Can message Id value configured in parameter 'CanIcomMessageId'.

Example(s)	**Action**	**Generated output**
	Configure CanIcomMessageId as 268435456U	`268435456U`
	Configure CanIcomMessageId as 2046U	`2046U`

1.2.23.2 Member: CanIcomMaskRef

Table 181 CanIcomMaskRef

Name	CanIcomMaskRef	
Type	Can_IdType	
Description	The configured message Id mask for the Icom message configured.	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanIcomRxMsgConfig[_<variant>] structure. The structure member is generated as a numeric value. This value is generated from the Can message Id mask value configured in container 'CanIcomMessageIdMask'. *Note: This structure member is generated with the value '0' if the list CanIcomMessageIdMask does not have any elements.*	
Example(s)	**Action**	**Generated output**
	Configure Icom message with CanIcomMessageIdMask as 268435456U	`268435456U`
	Configure Icom message with CanIcomMessageIdMask without any elements.	`0U`

1.2.23.3 Member: CanIcomCntrVal

Table 182 CanIcomCntrVal

Name	CanIcomCntrVal	
Type	uint16	
Description	The number of times the Icom message needs to be received before the CAN driver considers a wakeup by the current Icom message.	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanIcomRxMsgConfig[_<variant>] structure. The structure member is generated as a numeric value. This value is generated from the Can message counter value configured in container 'CanIcomCounterValue'. *Note: This structure member is generated with the value '0' if the list CanIcomCounterValue does not have any elements.*	
Example(s)	**Action**	**Generated output**
	Configure Icom message with CanIcomCounterValue as 10	`10U`

	Configure Icom message with CanIcomCounterValue without any elements.	`0U`

1.2.23.4 Member: CanIcomDLC

Table 183 CanIcomDLC

Name	CanIcomDLC	
Type	uint8	
Description	The CAN message data length for the Icom message	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanIcomRxMsgConfig[_<variant>] structure. The structure member is generated as a numeric value. This value is generated from the CAN message data length value configured in container 'CanIcomPayloadLength'.	
Example(s)	**Action**	**Generated output**
	Configure Icom message with CanIcomPayloadLength as 1	`1U`
	Configure Icom message with CanIcomPayloadLength as 8	`8U`

1.2.23.5 Member: CanIcomFirstSignalIndx

Table 184 CanIcomFirstSignalIndx

Name	CanIcomFirstSignalIndx	
Type	uint8	
Description	The index in the array of Icom message signal configuration for the first signal associated with this message in Icom.	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanIcomRxMsgConfig[_<variant>] structure. The structure member is generated as a numeric value. This value is generated based on the array offset at which the first signal for this message is amongst all the signals in the structure Can_17_McmCan_kMcmCanIcomRxMsgSignalConfig[_<variant>]. *Note: This structure member is generated with the value '0' if the message does not have any signals configured. This structure member is generated with the value '0' if the messages first signal is the first signal configured overall for the Icom.*	
Example(s)	**Action**	**Generated output**
	• Configure 3 Icom message • Configure message 1 with 3 signals • Configure message 2 with 4 signals • Configure message 3 with no signal • The CanIcomFirstSignalIndx for the message 1.	`0U`
	• Configure 3 Icom message	`3U`

	• Configure message 1 with 3 signals • Configure message 2 with 4 signals • Configure message 3 with no signal • The CanIcomFirstSignalIndx for the message 2.	
	• Configure 3 Icom message • Configure message 1 with 3 signals • Configure message 2 with 4 signals • Configure message 3 with no signal • The CanIcomFirstSignalIndx for the message 3.	`0U`

1.2.23.6 Member: CanIcomNoOfSignalIndx

Table 185 CanIcomNoOfSignalIndx

Name	CanIcomNoOfSignalIndx	
Type	uint8	
Description	The number of signals configured for this message in Icom	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanIcomRxMsgConfig[_<variant>] structure. The structure member is generated as a numeric value. This value is the total number of signals configured for the message. It is the count in the list 'CanIcomRxMessageSignalConfig'. *Note: This structure member is generated with the value '0' if the message does not have any signals configured.*	
Example(s)	**Action**	**Generated output**
	• Configure 3 Icom message • Configure message 1 with 3 signals • Configure message 2 with 4 signals • Configure message 3 with no signal • The CanIcomNoOfSignalIndx for the message 1.	`3U`
	• Configure 3 Icom message • Configure message 1 with 3 signals • Configure message 2 with 4 signals • Configure message 3 with no signal • The CanIcomNoOfSignalIndx for the message 2.	`4U`
	• Configure 3 Icom message • Configure message 1 with 3 signals • Configure message 2 with 4 signals	`0U`

	• Configure message 3 with no signal • The CanIcomNoOfSignalIndx for the message 3.	

1.2.23.7 Member: CanIcomLengthErr

Table 186 CanIcomLengthErr

Name	CanIcomLengthErr	
Type	boolean	
Description	Enables/ Disables wake up detection in Icom at detection of data length error	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanIcomRxMsgConfig[_<variant>] structure. The structure member is generated as a 'TRUE' when the container 'CanIcomPayloadLengthError' is set as 'True' else it is generated as 'FALSE'.	
Example(s)	**Action**	**Generated output**
	Configure Icom message with CanIcomPayloadLengthError as 'True'	`TRUE`
	Configure Icom message with CanIcomPayloadLengthError as 'False'	`FALSE`

1.2.24 Structure: Can_17_McmCan_kMcmCanIcomRxMsgSignalConfig[_<variant>] [CAN_17_MCMCAN_NOOF_ICOM_SIGNALCONFIGURATIONS]

Table 187 Can_17_McmCan_kMcmCanIcomRxMsgSignalConfig[_<variant>][CAN_17_MCMCAN_NOOF_ICOM_SIGNALCONFIGURATIONS]

Name	Can_17_McmCan_kMcmCanIcomRxMsgSignalConfig[_<variant>][CAN_17_MCMCAN_NOOF_ICOM_SIGNALCONFIGURATIONS]	
Type	Can_17_McmCan_IcomRxMsgSignalConfigType	
Description	Configuration structure of CAN driver for all Icom message signals related configurations.	
Verification method	The generated file has this structure if at least one Icom receive message signal in container container 'CanIcomRxMessageSignalConfig' is configured for any of the messages of Icom. <Variant> indicates the name of the post-build variant. For a variant aware configuration the structure name is appended with the variant name. For variant unaware configuration <variant> is ignored. The array size of this structure type is dependenent on the total number of signals configured in all of the Icom messages. *Note: This array is generated only when atleast one Icom signal is defined in the configuration and CanPublicIcomSupport is 'True' else this array of structures is not generated.*	
Example(s)	**Action**	**Generated output**

	- Configure 2 Icom messages.
	- Configure 1st Icom message with 1 Rx signal configuration with EQUAL operation and valid compare values configured.
	- Configure 2nd Icom message with 2 signals with valid compare values and EQUAL and GREATER operations configured.
		`static const Can_17_McmCan_IcomRxMsgSignalConfigType \`
		`Can_17_McmCan_kMcmCanIcomRxMsgSignalConfig[CAN_17_MCMCAN_NOOF_ICOM_SIGNALCONFIGURATIONS] =`
		`{`
		`{`
		`{0xffU, 0xffU, 0xffU, 0xffU, 0xffU, 0xffU, 0xffU, 0xffU },`
		`{0xddU, 0xccU, 0xbbU, 0xaaU, 0xddU, 0xccU, 0xbbU, 0xaaU },`
		`CAN_17_MCMCAN_ICOM_OPER_EQUAL`
		`},`
		`{`
		`{0xffU, 0xffU, 0xffU, 0xffU, 0xffU, 0xffU, 0xffU, 0xffU },`
		`{0xddU, 0xccU, 0xbbU, 0xaaU, 0xddU, 0xccU, 0xbbU, 0xaaU },`
		`CAN_17_MCMCAN_ICOM_OPER_EQUAL`
		`},`
		`{`
		`{0xffU, 0xffU, 0xffU, 0xffU, 0xffU, 0xffU, 0xffU, 0xffU },`
		`{0xddU, 0xccU, 0xbbU, 0xaaU, 0xddU, 0xccU, 0xbbU, 0xaaU },`
		`CAN_17_MCMCAN_ICOM_OPER_GREATER`
		`}`
		`};`

1.2.24.1 Member: CanIcomSignalMask[CAN_17_MCMCAN_ICOM_DATA_SIGNAL_MATCH_SIZE]

Table 188 CanIcomSignalMask[CAN_17_MCMCAN_ICOM_DATA_SIGNAL_MATCH_SIZE]

Name	CanIcomSignalMask[CAN_17_MCMCAN_ICOM_DATA_SIGNAL_MATCH_SIZE]
Type	uint8
Description	The mask value for the Icom signal
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanIcomRxMsgSignalConfig[_<variant>] structure. The structure member is generated as a numeric array of size 8 with its range between 0x00 to 0xff. This value is generated from the values configured in containers

	‘CanIcomSignalMaskUpper32bits’ and ‘CanIcomSignalMaskLower32bits’ of the respective signal. *Note: The first 4 elements of the array is filled with the 2 byte each starting with MSB considering a 32 bit value from container ‘CanIcomSignalMaskLower32bits’ and the last 4 elements of the array is filled with the 2 byte each starting with MSB considering a 32 bit value from container ‘CanIcomSignalMaskUpper32bits’.*	
Example(s)	**Action**	**Generated output**
	Configure a Icom signal with CanIcomSignalMaskLower32bits value as 0 and CanIcomSignalMaskUpper32bits as 65200 (0xfeb0)	{0x00U, 0x00U, 0x00U, 0x00U, 0x00U, 0x00U, 0xfeU, 0xb0U }
	Configure a Icom signal with CanIcomSignalMaskLower32bits value as 2835(0xb13) and CanIcomSignalMaskUpper32bits as 0	{0x00U, 0x00U, 0x0bU, 0x13U, 0x00U, 0x00U, 0x00U, 0x00U}
	Configure a Icom signal with CanIcomSignalMaskLower32bits value as 2720279315 (0xa2242b13) and CanIcomSignalMaskUpper32bits as 1696745540 (0x65224844)	{0xa2U, 0x24U, 0x2bU, 0x13U, 0x65U, 0x22U, 0x48U, 0x44U}

1.2.24.2 Member: CanIcomSignalValue[CAN_17_MCMCAN_ICOM_DATA_SIGNAL_MATCH_SIZE]

Table 189 CanIcomSignalValue[CAN_17_MCMCAN_ICOM_DATA_SIGNAL_MATCH_SIZE]

Name	CanIcomSignalValue[CAN_17_MCMCAN_ICOM_DATA_SIGNAL_MATCH_SIZE]	
Type	uint8	
Description	The signal value for the Icom signal	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanIcomRxMsgSignalConfig[_<variant>] structure. The structure member is generated as a numeric array of size 8 with its range between 0x00 to 0xff. This value is generated from the values configured in containers ‘CanIcomSignalValueUpper32bits’ and ‘CanIcomSignalValueLower32bits’ of the respective signal. *Note: The first 4 elements of the array is filled with the 2 byte each starting with MSB considering a 32 bit value from container ‘CanIcomSignalValueLower32bits’ and the last 4 elements of the array is filled with the 2 byte each starting with MSB considering a 32 bit value from container ‘CanIcomSignalValueUpper32bits’.*	
Example(s)	**Action**	**Generated output**
	Configure a Icom signal with CanIcomSignalValueLower32bits value as 0 and	{0x00U, 0x00U, 0x00U, 0x00U, 0x00U, 0x00U, 0xfeU, 0xb0U }

	CanIcomSignalValueUpper32bits as 65200 (0xfeb0)	
	Configure a Icom signal with CanIcomSignalValueLower32bits value as 2835(0xb13) and CanIcomSignalValueUpper32bits as 0	`{0x00U, 0x00U, 0x0bU, 0x13U, 0x00U, 0x00U, 0x00U, 0x00U}`
	Configure a Icom signal with CanIcomSignalValueLower32bits value as 2720279315 (0xa2242b13) and CanIcomSignalValueUpper32bits as 1696745540 (0x65224844)	`{0xa2U, 0x24U, 0x2bU, 0x13U, 0x65U, 0x22U, 0x48U, 0x44U}`

1.2.24.3 Member: CanIcomSignalOper

Table 190 CanIcomSignalOper

Name	CanIcomSignalOper	
Type	Can_17_McmCan_IcomSignalOperType	
Description	The comparison operation to be performed using the signal during Icom wakeup check	
Verification method	The generated structure member is present in Can_17_McmCan_kMcmCanIcomRxMsgSignalConfig[_<variant>] structure. The structure member is generated based on the configuration present in the drop down list in container 'CanIcomSignalOperation' for the particular signal.	
Example(s)	**Action**	**Generated output**
	Configure a Icom signal with CanIcomSignalOperation set as 'AND'	`CAN_17_MCMCAN_ICOM_OPER_AND`
	Configure a Icom signal with CanIcomSignalOperation set as 'SMALLER'	`CAN_17_MCMCAN_ICOM_OPER_SMALLER`
	Configure a Icom signal with CanIcomSignalOperation set as 'XOR'	`CAN_17_MCMCAN_ICOM_OPER_XOR`

1.2.25 Function Definition: Can_17_McmCan_MainFunction_Write_<Period Index>

Table 191 Can_17_McmCan_MainFunction_Write_<Period Index>

Name	Can_17_McmCan_MainFunction_Write_<Period Index>
Type	void Can_17_McmCan_MainFunction_Write_<Period Index> (void)
Description	The function definition for the multi-period write in polling mode generated based on the configuration parameter 'CanMainFunctionRWPeriods'
Verification method	The number of function definition generated is based on the number of elements in the list configured in 'CanMainFunctionRWPeriods'. < Period Index > is the element number of the 'CanMainFunctionRWPeriods'. This function generated shall internally call the function Can_17_McmCan_lTxPeriodHandler with passed parameter same as < Period Index >. *Note: This function definition is generated only when atleast one CAN controller container 'CanTxProcessing' is set to 'Polling' and number of elements*

	configured in list 'CanMainFunctionRWPeriods' is greater than 1 else this function definition is not generated.	
Example(s)	**Action**	**Generated output**

	• Configure 5 elements in 'CanMainFunctionRWPeriods' list.
	• Configure 3 CAN controllers.
	• Configure 1 Can controller with 'CanTxProcessing' set as 'Polling'.
		`void Can_17_McmCan_MainFunction_Write_0(void)`
		`{`
		`    Can_17_McmCan_lTxPeriodHandler(0);`
		`}`
		`void Can_17_McmCan_MainFunction_Write_1(void)`
		`{`
		`    Can_17_McmCan_lTxPeriodHandler(1);`
		`}`
		`void Can_17_McmCan_MainFunction_Write_2(void)`
		`{`
		`    Can_17_McmCan_lTxPeriodHandler(2);`
		`}`
		`void Can_17_McmCan_MainFunction_Write_3(void)`
		`{`
		`    Can_17_McmCan_lTxPeriodHandler(3);`
		`}`
		`void Can_17_McmCan_MainFunction_Write_4(void)`
		`{`
		`    Can_17_McmCan_lTxPeriodHandler(4);`
		`}`

1.2.26 Function Definition: Can_17_McmCan_MainFunction_Read_<Period Index>

Table 192 Can_17_McmCan_MainFunction_Read_<Period Index>

Name	Can_17_McmCan_MainFunction_Read_<Period Index>
Type	void Can_17_McmCan_MainFunction_Read_<Period Index> (void)
Description	The function definition for the multi-period read in polling mode generated based on the configuration parameter 'CanMainFunctionRWPeriods'
Verification method	The number of function definition generated is based on the number of elements in the list configured in 'CanMainFunctionRWPeriods'. < Period Index > is the element number of the 'CanMainFunctionRWPeriods'. This function generated shall internally call the

	function Can_17_McmCan_lRxPeriodHandler with passed parameter same as < Period Index >. *Note: This function definition is generated only when atleast one CAN controller container 'CanRxProcessing' is set to 'Polling' and number of elements configured in list 'CanMainFunctionRWPeriods' is greater than 1 else this function definition is not generated.*	
Example(s)	**Action**	**Generated output**

	• Configure 5 elements in 'CanMainFunctionRWPeriods' list.
	• Configure 4 CAN controllers.
	• Configure 3 Can controllers with 'CanRxProcessing' set as 'Polling'.
		`void Can_17_McmCan_MainFunction_Read_0(void)`
		`{`
		`    Can_17_McmCan_lRxPeriodHandler(0);`
		`}`
		`void Can_17_McmCan_MainFunction_Read_1(void)`
		`{`
		`    Can_17_McmCan_lRxPeriodHandler(1);`
		`}`
		`void Can_17_McmCan_MainFunction_Read_2(void)`
		`{`
		`    Can_17_McmCan_lRxPeriodHandler(2);`
		`}`
		`void Can_17_McmCan_MainFunction_Read_3(void)`
		`{`
		`    Can_17_McmCan_lRxPeriodHandler(3);`
		`}`
		`void Can_17_McmCan_MainFunction_Read_4(void)`
		`{`
		`    Can_17_McmCan_lRxPeriodHandler(4);`
		`}`

1.3 File: Can_17_McmCan[_<variant>]_PBcfg.h

The generated header file contains the declaration of the root configuration structure. Post-build time configuration mechanism allows configurable functionality of CAN driver that is deployed as object code. The file is generated in ‘inc’ folder.

1.3.1 Structure: Can_17_McmCan_Config[_<variant>]

Table 193 Can_17_McmCan_Config[_<varaint>]

<table>
<tr><td><b>Name</b></td><td colspan="2">Can_17_McmCan_Config[_<variant>]</td></tr>
<tr><td><b>Type</b></td><td colspan="2">Can_17_McmCan_ConfigType</td></tr>
<tr><td><b>Description</b></td><td colspan="2">Declaration of root configuration structure of CAN driver which will be used during initialization.</td></tr>
<tr><td><b>Verification method</b></td><td colspan="2">The generated structure is present in Can_17_McmCan[_<variant>]_PBcfg.h file. The <variant> indicates the name of the post-build variant. For a variant-aware configuration the structure name is appended with the variant name. For variant-unaware configuration <variant> is ignored.</td></tr>
<tr><td rowspan="3"><b>Example(s)</b></td><td><b>Action</b></td><td><b>Generated output</b></td></tr>
<tr><td>Configure CAN and generate (variant-unaware)</td><td><code>extern const
Can_17_McmCan_ConfigType
Can_17_McmCan_Config;</code></td></tr>
<tr><td>Configure CAN and generate (variant-aware. Variant name is ‘Petrol’)</td><td><code>extern const
Can_17_McmCan_ConfigType
Can_17_McmCan_Config_Petrol;</code></td></tr>
</table>

Revision history

Major changes since the last revision

Date	Version	Description
03-12-2020	v3.0	Released Version
30-11-2020	v2.1	Changes made for 2.0.0-rc
27-02-2019	v1.10.0_2.0	Adde Pbcfg.h
26-02-2019	v1.10.0_1.0	Released Version
26-02-2019	v1.10.0_0.1	Initial Version

Edition 2020-12-03
Published by

Infineon Technologies AG

81726 Munich, Germany

Do you have a question about this document?

Email: erratum@infineon.com

Document reference
Doc_Number

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

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

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

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

For further information on the product, technology, delivery terms and conditions and prices please contact your nearest Infineon Technologies office (**www.infineon.com**).

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

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