MCAL Configuration Verification Manual for ADC

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.

Intended audience

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

Reference documents

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

- AURIX™ TC3xx MCAL User Manual ADC

Table of contents

Table of contents

Table of contents

Table of contents

1 Adc driver

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

1.1 File: Adc_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: ADC_AR_RELEASE_MAJOR_VERSION

Table 1 ADC_AR_RELEASE_MAJOR_VERSION

Name	ADC_AR_RELEASE_MAJOR_VERSION	
Description	Major version number of AUTOSAR release on which the Adc 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 Adc_Cfg.h file with ArMajorVersion 4	`#define ADC_AR_RELEASE_MAJOR_VERSION (4U)`

1.1.2 Macro: ADC_AR_RELEASE_MINOR_VERSION

Table 2 ADC_AR_RELEASE_MINOR_VERSION

Name	ADC_AR_RELEASE_MINOR_VERSION	
Description	Minor version number of AUTOSAR release on which the Adc 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 Adc_Cfg.h file with ArMinorVersion 2	`#define ADC_AR_RELEASE_MINOR_VERSION (2U)`

1.1.3 Macro: ADC_AR_RELEASE_REVISION_VERSION

Table 3 ADC_AR_RELEASE_REVISION_VERSION

Name	ADC_AR_RELEASE_REVISION_VERSION
Description	Revision version number of AUTOSAR release on which the Adc 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 Adc_Cfg.h file with ArPatchVersion 2	`#define ADC_AR_RELEASE_REVISION_VERSION (2U)`

1.1.4 Macro: ADC_SW_MAJOR_VERSION

Table 4 ADC_SW_MAJOR_VERSION

Name	ADC_SW_MAJOR_VERSION
Description	Major version number of the Adc 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 Adc_Cfg.h file with SwMajorVersion 10	`#define ADC_SW_MAJOR_VERSION (10U)`

1.1.5 Macro: ADC_SW_MINOR_VERSION

Table 5 ADC_SW_MINOR_VERSION

Name	ADC_SW_MINOR_VERSION
Description	Minor version number of the Adc 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 Adc_Cfg.h file with SwMinorVersion 10	`#define ADC_SW_MINOR_VERSION (10U)`

1.1.6 Macro: ADC_SW_PATCH_VERSION

Table 6 ADC_SW_PATCH_VERSION

Name	ADC_SW_PATCH_VERSION
Description	Patch version number of the Adc 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 Adc_Cfg.h file with SwPatchVersion 0	`#define ADC_SW_PATCH_VERSION (0U)`

1.1.7 Macro: ADC_SAFETY_ENABLE

Table 7 ADC_SAFETY_ENABLE

Name	ADC_SAFETY_ENABLE	
Description	Enables/disables the safety features.	
Verification method	The macro is generated as STD_ON if AdcSafetyEnable configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	AdcSafetyEnable = True	`#define ADC_SAFETY_ENABLE (STD_ON)`
	AdcSafetyEnable = False	`#define ADC_SAFETY_ENABLE (STD_OFF)`

1.1.8 Macro: ADC_INIT_CHECK_API

Table 8 ADC_INIT_CHECK_API

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

1.1.9 Macro: ADC_RUN_TIME_API_MODE

Table 9 ADC_RUN_TIME_API_MODE

Name	ADC_RUN_TIME_API_MODE	
Description	Decides the mode of execution of Run Time API's.	
Verification method	The macro is generated as STD_ON if AdcRuntimeApiMode configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	AdcRuntimeApiMode = True	`#define ADC_RUN_TIME_API_MODE (STD_ON)`

	AdcRuntimeApiMode = False	`#define ADC_RUN_TIME_API_MODE (STD_OFF)`

1.1.10 Macro: ADC_INIT_DEINIT_API_MODE

Table 10 ADC_INIT_DEINIT_API_MODE

Name	ADC_INIT_DEINIT_API_MODE	
Description	Determines the mode of execution of Init and DeInit API's.	
Verification method	The macro is generated as STD_ON if AdcInitDeInitApiMode configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	AdcInitDeInitApiMode = True	`#define ADC_INIT_DEINIT_API_MODE (STD_ON)`
	AdcInitDeInitApiMode = False	`#define ADC_INIT_DEINIT_API_MODE (STD_OFF)`

1.1.11 Macro: ADC_DEV_ERROR_DETECT

Table 11 ADC_DEV_ERROR_DETECT

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

1.1.12 Macro: ADC_MULTICORE_ERROR_DETECT

Table 12 ADC_MULTICORE_ERROR_DETECT

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

1.1.13 Macro: ADC_RUNTIME_ERROR_DETECT

Table 13 ADC_RUNTIME_ERROR_DETECT

Name	ADC_RUNTIME_ERROR_DETECT	
Description	Enables/disables the Run-time Error Detection.	
Verification method	The macro is generated as STD_ON if AdcRunTimeErrorDetect configuration parameter is set to 'True' else the macro is generated as STD_OFF. *Note: The macro is applicable only for AUTOSAR version 4.4.0.*	
Example(s)	**Action**	**Generated output**
	AdcRunTimeErrorDetect = True	`#define ADC_RUNTIME_ERROR_DETECT (STD_ON)`
	AdcRunTimeErrorDetect = False	`#define ADC_RUNTIME_ERROR_DETECT (STD_OFF)`

1.1.14 Macro: ADC_ENABLE_START_STOP_GROUP_API

Table 14 ADC_ENABLE_START_STOP_GROUP_API

Name	ADC_ENABLE_START_STOP_GROUP_API	
Description	Enables/disables the Adc_StartGroupConversion and Adc_StopGroupConversion API's.	
Verification method	The macro is generated as STD_ON if AdcEnableStartStopGroupApi configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	AdcEnableStartStopGroupApi = True	`#define ADC_ENABLE_START_STOP_GROUP_API (STD_ON)`
	AdcEnableStartStopGroupApi = False	`#define ADC_ENABLE_START_STOP_GROUP_API (STD_OFF)`

1.1.15 Macro: ADC_DEINIT_API

Table 15 ADC_DEINIT_API

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

1.1.16 Macro: ADC_HW_TRIGGER_API

Table 16 ADC_HW_TRIGGER_API

Name	ADC_HW_TRIGGER_API	
Description	Enables/disables the Adc_EnableHardwareTrigger and Adc_DisableHardwareTrigger API's.	
Verification method	The macro is generated as STD_ON if AdcHwTriggerApi configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	AdcHwTriggerApi = True	`#define ADC_HW_TRIGGER_API (STD_ON)`
	AdcHwTriggerApi = False	`#define ADC_HW_TRIGGER_API (STD_OFF)`

1.1.17 Macro: ADC_READ_GROUP_API

Table 17 ADC_READ_GROUP_API

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

1.1.18 Macro: ADC_STARTUP_CALIB_API

Table 18 ADC_STARTUP_CALIB_API

Name	ADC_STARTUP_CALIB_API	
Description	Enables/disables the Adc_GetStartupCalStatus and Adc_TriggerStartupCal API's.	
Verification method	The macro is generated as STD_ON if AdcStartupCalibApi configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	AdcStartupCalibApi = True	`#define ADC_STARTUP_CALIB_API (STD_ON)`
	AdcStartupCalibApi = False	`#define ADC_STARTUP_CALIB_API (STD_OFF)`

1.1.19 Macro: ADC_TRIGGER_ONE_CONV_ENABLE

Table 19 ADC_TRIGGER_ONE_CONV_ENABLE

Name	ADC_TRIGGER_ONE_CONV_ENABLE
Description	Enables/disables the Dummy Converison before the startup calibration.

Verification method	The macro is generated as STD_ON if AdcTriggerOneConversionEnable configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	AdcTriggerOneConversionEnable = True	`#define ADC_TRIGGER_ONE_CONV_ENABLE (STD_ON)`
	AdcTriggerOneConversionEnable = False	`#define ADC_TRIGGER_ONE_CONV_ENABLE (STD_OFF)`

1.1.20 Macro: ADC_ENABLE_LIMIT_CHECK

Table 20 ADC_ENABLE_LIMIT_CHECK

Name	ADC_ENABLE_LIMIT_CHECK	
Description	Enables/disables the limit checking feature of ADC.	
Verification method	The macro is generated as STD_ON if AdcEnableLimitCheck configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	AdcEnableLimitCheck = True	`#define ADC_ENABLE_LIMIT_CHECK (STD_ON)`
	AdcEnableLimitCheck = False	`#define ADC_ENABLE_LIMIT_CHECK (STD_OFF)`

1.1.21 Macro: ADC_EMUX_ENABLE

Table 21 ADC_EMUX_ENABLE

Name	ADC_EMUX_ENABLE	
Description	Enables/disables the EMUX feature of ADC.	
Verification method	The macro is generated as STD_ON if AdcEmuxEnable configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	AdcEmuxEnable = True	`#define ADC_EMUX_ENABLE (STD_ON)`
	AdcEmuxEnable = False	`#define ADC_EMUX_ENABLE (STD_OFF)`

1.1.22 Macro: ADC_GRP_NOTIF_CAPABILITY

Table 22 ADC_GRP_NOTIF_CAPABILITY

Name	ADC_GRP_NOTIF_CAPABILITY	
Description	Enables/disables the Notification capability of ADC.	
Verification method	The macro is generated as STD_ON if AdcGrpNotifCapability configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	AdcGrpNotifCapability = True	`#define ADC_GRP_NOTIF_CAPABILITY`

		(STD_ON)
	AdcGrpNotifCapability = False	#define ADC_GRP_NOTIF_CAPABILITY (STD_OFF)

1.1.23 Macro: ADC_VERSION_INFO_API

Table 23 ADC_VERSION_INFO_API

Name	ADC_VERSION_INFO_API	
Description	Enables/disables Adc_GetVersionInfo API	
Verification method	The macro is generated as STD_ON if AdcVersionInfoApi configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	AdcVersionInfoApi = True	#define ADC_VERSION_INFO_API (STD_ON)
	AdcVersionInfoApi = False	#define ADC_VERSION_INFO_API (STD_OFF)

1.1.24 Macro: ADC_ENABLE_QUEUING

Table 24 ADC_ENABLE_QUEUING

Name	ADC_ENABLE_QUEUING	
Description	Enables/disables the Queuing mechanism when priority mechanism is disabled.	
Verification method	The macro is generated as STD_ON if AdcEnableQueuing configuration parameter is set to 'True' else the macro is generated as STD_OFF. *Note: This macro generates the configured value of AdcEnableQueuing parameter only when AdcGeneral/AdcEnableStartStopGroupApi = 'true' and AdcGeneral/AdcPriorityImplementation = 'ADC_PRIORITY_NONE' otherwise always generates as STD_OFF.*	
Example(s)	**Action**	**Generated output**
	AdcEnableQueuing = True	#define ADC_ENABLE_QUEUING (STD_ON)
	AdcEnableQueuing = False	#define ADC_ENABLE_QUEUING (STD_OFF)

1.1.25 Macro: ADC_PRIORITY_IMPLEMENTATION

Table 25 ADC_PRIORITY_IMPLEMENTATION

Name	ADC_PRIORITY_IMPLEMENTATION
Description	Determines the type of prioritization mechanism.
Verification method	The macro is generated as a numeric value which corresponds to the number of elements in the list 'AdcGeneral/AdcPriorityImplementation'.

Example(s)	Action	Generated output
	Set AdcPriorityImplementation as ADC_PRIORITY_NONE	`#define ADC_PRIORITY_IMPLEMENTATION (ADC_PRIORITY_NONE)`
	Set AdcPriorityImplementation as ADC_PRIORITY_HW	`#define ADC_PRIORITY_IMPLEMENTATION (ADC_PRIORITY_HW)`
	Set AdcPriorityImplementation as ADC_PRIORITY_HW_SW	`#define ADC_PRIORITY_IMPLEMENTATION (ADC_PRIORITY_HW_SW)`

1.1.26 Macro: ADC_RESULT_HANDLING_IMPLEMENTATION

Table 26 ADC_RESULT_HANDLING_IMPLEMENTATION

Name	ADC_RESULT_HANDLING_IMPLEMENTATION	
Description	Determines the type of result handling mechanism.	
Verification method	The macro is generated as a numeric value which corresponds to the number of elements in the list ‘AdcGeneral/ AdcResultHandlingImplementation’.	
Example(s)	**Action**	**Generated output**
	Set AdcResultHandlingImplementation as ADC_INTERRUPT_MODE_RESULT_HANDLING	`#define ADC_RESULT_HANDLING_IMPLEMENTATION (ADC_INTERRUPT_MODE_RESULT_HANDLING)`
	Set AdcResultHandlingImplementation as ADC_DMA_MODE_RESULT_HANDLING	`#define ADC_RESULT_HANDLING_IMPLEMENTATION (ADC_DMA_MODE_RESULT_HANDLING)`

1.1.27 Macro: ADC_SLEEP_MODE_CFG

Table 27 ICU_17_TIMERIP_EDGE_DETECT_API

Name	ADC_SLEEP_MODE_CFG	
Description	Determines the status of Sleep mode.	
Verification method	The macro is generated as a numeric value which corresponds to the number of elements in the list ‘AdcGeneral/ AdcSleepMode’.	
Example(s)	**Action**	**Generated output**
	Set AdcSleepMode as ADC_SLEEP_MODE_ACCEPT	`#define ADC_SLEEP_MODE_CFG (ADC_SLEEP_MODE_ACCEPT)`
	Set AdcSleepMode as ADC_SLEEP_MODE_REJECT	`#define ADC_SLEEP_MODE_CFG (ADC_SLEEP_MODE_REJECT)`

1.1.28 Macro: ADC_RESULT_ALIGNMENT

Table 28 ADC_RESULT_ALIGNMENT

Name	ADC_RESULT_ALIGNMENT

Description	Determines the type of Result Alignment.	
Verification method	The macro is generated as a numeric value which corresponds to the number of elements in the list 'AdcGeneral/ AdcResultAlignment'.	
Example(s)	**Action**	**Generated output**
	Set AdcResultAlignment as ADC_ALIGN_RIGHT	`#define ADC_RESULT_ALIGNMENT (ADC_ALIGN_RIGHT)`
	Set AdcResultAlignment as ADC_ALIGN_LEFT	`#define ADC_RESULT_ALIGNMENT (ADC_ALIGN_LEFT)`

1.1.29 Macro: ADC_SUPPLY_VOLTAGE

Table 29 ADC_SUPPLY_VOLTAGE

Name	ADC_SUPPLY_VOLTAGE	
Description	Determines the type of Supply Voltage.	
Verification method	The macro is generated as a numeric value which corresponds to the number of elements in the list 'AdcGeneral/ AdcSupplyVoltage'.	
Example(s)	**Action**	**Generated output**
	Set AdcSupplyVoltage as ADC_VOLTAGE_CONTROLLED_BY_SUPPLY	`#define ADC_SUPPLY_VOLTAGE (ADC_VOLTAGE_CONTROLLED_BY_SUPPLY)`
	Set AdcSupplyVoltage as ADC_VOLTAGE_5V	`#define ADC_SUPPLY_VOLTAGE (ADC_VOLTAGE_5V)`
	Set AdcSupplyVoltage as ADC_VOLTAGE_3P3V	`#define ADC_SUPPLY_VOLTAGE (ADC_VOLTAGE_3P3V)`

1.1.30 Macro: ADC_SYNC_CONV_ENABLE

Table 30 ADC_RESULT_ALIGNMENT

Name	ADC_SYNC_CONV_ENABLE	
Description	Enables/disables the synchronous conversions across ADC HW groups.	
Verification method	The macro is generated as STD_ON if AdcSyncConvEnable configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	AdcSyncConvEnable = True	`#define ADC_SYNC_CONV_ENABLE (STD_ON)`
	AdcSyncConvEnable = False	`#define ADC_SYNC_CONV_ENABLE (STD_OFF)`

1.1.31 Macro: ADC_MAX_CH_CONV_TIME

Table 31 ADC_MAX_CH_CONV_TIME

Name	ADC_MAX_CH_CONV_TIME
Description	Determines the maximum channel conversion time in terms of wait loop count.

Verification method	The macro is generated as a numeric value set in the configuration parameter 'AdcGeneral/ AdcMaxChConvTimeCount'.	
Example(s)	**Action**	**Generated output**
	Set AdcMaxChConvTimeCount as 0	`#define ADC_MAX_CH_CONV_TIME (0U)`
	Set AdcMaxChConvTimeCount as 6000	`#define ADC_MAX_CH_CONV_TIME (6000U)`
	Set AdcMaxChConvTimeCount as 16962	`#define ADC_MAX_CH_CONV_TIME (16962U)`

1.1.32 Macro: ADC_LOW_POWER_STATE_SUPPORT

Table 32 ADC_LOW_POWER_STATE_SUPPORT

Name	ADC_LOW_POWER_STATE_SUPPORT	
Description	Enables/disables the low power states support features of ADC.	
Verification method	The macro is generated as STD_ON if AdcLowPowerStatesSupport configuration parameter is set to 'True' else the macro is generated as STD_OFF. *Note: This macro is configurable only when AdcLowPowerStatesSupport parameter exists.*	
Example(s)	**Action**	**Generated output**
	Set AdcLowPowerStatesSupport as True	`#define ADC_LOW_POWER_STATE_SUPPORT (STD_ON)`
	Set AdcLowPowerStatesSupport as False	`#define ADC_LOW_POWER_STATE_SUPPORT (STD_OFF)`

1.1.33 Macro: ADC_POWER_MODES_AVAILABLE

Table 33 ADC_POWER_MODES_AVAILABLE

Name	ADC_POWER_MODES_AVAILABLE	
Description	Determines the bit state for configured power modes with decreasing power consumptions based on the instances of Adc power state configurations i.e. bit position 0 indicates the 1st instances, bit position 1 indicates the 2nd instances so on..	
Verification method	The macro is generated as a numeric value set in the configuration parameter 'AdcGeneral/AdcPowerStateConfig/AdcPowerState'.	
Example(s)	**Action**	**Generated output**
	Set AdcPowerState as 0	`#define ADC_POWER_MODES_AVAILABLE (0x00000001U)`

	Set AdcPowerState as 0 and 2	`#define ADC_POWER_MODES_AVAILABLE (0x00000005U)`
	Set AdcPowerState as 0,1,2 and 3	`#define ADC_POWER_MODES_AVAILABLE (0x0000000FU)`

1.1.34 Macro: ADC_CLC_FAILURE_DEM_NOTIF

Table 34 ADC_CLC_FAILURE_DEM_NOTIF

Name	ADC_CLC_FAILURE_DEM_NOTIF	
Description	Enables/disables the Production error for CLC failure.	
Verification method	The macro is generated as a numeric value when the configuration parameter contains the 'AdcDemEventParameterRefs/AdcClcFailureNotification'.	
Example(s)	**Action**	**Generated output**

	Configure Production error name as AdcClcFailure in DemEventParameter container.	`#define ADC_CLC_FAILURE_DEM_NOTIF (ADC_ENABLE_DEM_REPORT)`
		
		`#define ADC_E_CLC_FAILURE (DemConf_DemEventParameter_AdcClcFailure)`

	AdcClcFailureNotification does not contain Dem event parameter.	`#define ADC_CLC_FAILURE_DEM_NOTIF (ADC_DISABLE_DEM_REPORT).`

1.1.35 Macro: ADC_E_CLC_FAILURE

Table 35 ADC_E_CLC_FAILURE

Name	ADC_E_CLC_FAILURE	
Description	Production error for CLC failure.	
Verification method	The macro is generated as a DemConf_DemEventParameter_<ProductionErrorName> based on Production Error name configured in 'AdcDemEventParameterRefs/AdcClcFailureNotification'. *Note: This macro generates only when configuration parameter contains in 'AdcDemEventParameterRefs/AdcClcFailureNotification'.*	
Example(s)	Action	**Generated output**
	Configure Production error name as AdcClc in DemEventParameter container.	`#define ADC_E_CLC_FAILURE (DemConf_DemEventParameter_AdcClc)`
	Configure Production error name as AdcClcFailure in DemEventParameter container.	`#define ADC_E_CLC_FAILURE (DemConf_DemEventParameter_AdcClcFailure)`

1.1.36 Macro: ADC_CONV_STOP_TIME_DEM_NOTIF

Table 36 ADC_CONV_STOP_TIME_DEM_NOTIF

Name	ADC_CONV_STOP_TIME_DEM_NOTIF	
Description	Enables/disables the Production error for maximum channel conversion time to stop the conversion.	
Verification method	The macro is generated as a numeric value when the configuration parameter contains the 'AdcDemEventParameterRefs/ AdcConvStopTimeNotification'.	
Example(s)	**Action**	**Generated output**

	Configure Production error name as AdcStopConvFailure in DemEventParameter container.	`#define ADC_CONV_STOP_TIME_DEM_NOTIF (ADC_ENABLE_DEM_REPORT)`
		
		`#define ADC_E_CONV_STOP_TIME_FAILURE (DemConf_DemEventParameter_AdcStopConv)`

	AdcConvStopTimeNotification does not contain Dem event parameter.	`#define ADC_CONV_STOP_TIME_DEM_NOTIF (ADC_DISABLE_DEM_REPORT)`

1.1.37 Macro: ADC_E_CONV_STOP_TIME_FAILURE

Table 37 ADC_E_CONV_STOP_TIME_FAILURE

Name	ADC_E_CONV_STOP_TIME_FAILURE	
Description	Production error for maximum channel conversion time to stop the conversion.	
Verification method	The macro is generated as a DemConf_DemEventParameter_<ProductionErrorName> based on Production Error name configured in 'AdcDemEventParameterRefs/ AdcConvStopTimeNotification'. *Note: This macro generates only when configuration parameter contains in 'AdcDemEventParameterRefs/ AdcConvStopTimeNotification'.*	
Example(s)	**Action**	**Generated output**
	Configure Production error name as AdcStopConv in DemEventParameter container.	`#define ADC_E_CONV_STOP_TIME_FAILURE (DemConf_DemEventParameter_AdcStopConv)`
	Configure Production error name as AdcStopConvFailure in DemEventParameter container.	`#define ADC_E_CONV_STOP_TIME_FAILURE (DemConf_DemEventParameter_AdcStopConvFailure)`

1.1.38 Macro: ADC_MAX_GROUPS

Table 38 ADC_MAX_GROUPS

Name	ADC_MAX_GROUPS	
Description	Indicates the maximum number of ADC Channel groups configured across HW units.	
Verification method	The macro is generated as a total number of groups configured across HW units.	
Example(s)	**Action**	**Generated output**
	Configure 1 groups to HW unit1, Configure 3 groups to HW unit2, Configure 5 groups to HW unit3,	`#define ADC_MAX_GROUPS (5)`
	Configure 6 groups to HW unit1, Configure 16 groups to HW unit2, Configure 32 groups to HW unit3,	`#define ADC_MAX_GROUPS (32)`

1.1.39 Macro: ADC[Y]_KERNEL_INDEX_CORE[X]

Table 39 ADC[Y]_KERNEL_INDEX_CORE[X]

Name	ADC[Y]_KERNEL_INDEX_CORE[X]	
Description	Indicates the array index for the HW unit 'Y' in core 'X' in the Adc_kKernelDataIndex structure. Where 'X' is ranging from 0 to 5 & 'Y' is ranging from 0 to 11 depends on HW Derivative.	
Verification method	The macro is generated as an array index for HW unit 'Y' in core 'X'. *Note: HW units 'Y' not configured to core 'X' is assigned with 0xFFU.*	
Example(s)	**Action**	**Generated output**

	• Set ResourceMMasterCore as core0.
	• Assign HW unit0 to core0,
	• Assign HW unit5 to core0,
	• Assign HW unit8 to core0.
		`#define ADC0_KERNEL_INDEX_CORE0 (0U)`
		`#define ADC1_KERNEL_INDEX_CORE0 (0xFFU)`
		`#define ADC2_KERNEL_INDEX_CORE0 (0xFFU)`
		`#define ADC3_KERNEL_INDEX_CORE0 (0xFFU)`
		`#define ADC4_KERNEL_INDEX_CORE0 (0xFFU)`
		`#define ADC5_KERNEL_INDEX_CORE0 (1U)`
		`#define ADC6_KERNEL_INDEX_CORE0 (0xFFU)`
		`#define ADC7_KERNEL_INDEX_CORE0 (0xFFU)`
		`#define ADC8_KERNEL_INDEX_CORE0 (2U)`
		`#define ADC9_KERNEL_INDEX_CORE0 (0xFFU)`
		`#define ADC10_KERNEL_INDEX_CORE0 (0xFFU)`
		`#define ADC11_KERNEL_INDEX_CORE0 (0xFFU)`

	• Set ResourceMMasterCore as core3.
	• Assign HW unit1 to core3,
	• Assign HW unit6 to core3,
	• Assign HW unit11 to core3.
		`#define ADC0_KERNEL_INDEX_CORE3 (0xFFU)`
		`#define ADC1_KERNEL_INDEX_CORE3 (0U)`
		`#define ADC2_KERNEL_INDEX_CORE3 (0xFFU)`
		`#define ADC3_KERNEL_INDEX_CORE3 (0xFFU)`
		`#define ADC4_KERNEL_INDEX_CORE3 (0xFFU)`
		`#define ADC5_KERNEL_INDEX_CORE3 (0xFFU)`
		`#define ADC6_KERNEL_INDEX_CORE3 (1U)`
		`#define ADC7_KERNEL_INDEX_CORE3 (0xFFU)`
		`#define ADC8_KERNEL_INDEX_CORE3 (0xFFU)`
		`#define ADC9_KERNEL_INDEX_CORE3 (0xFFU)`
		`#define ADC10_KERNEL_INDEX_CORE3 (0xFFU)`
		`#define ADC11_KERNEL_INDEX_CORE3 (2U)`

	• Set ResourceMMasterCore as core5.
	• Assign HW unit11 to core5.
		`#define ADC0_KERNEL_INDEX_CORE5 (0xFFU)`
		`#define ADC1_KERNEL_INDEX_CORE5 (0xFFU)`
		`#define ADC2_KERNEL_INDEX_CORE5 (0xFFU)`
		`#define ADC3_KERNEL_INDEX_CORE5 (0xFFU)`
		`#define ADC4_KERNEL_INDEX_CORE5 (0xFFU)`
		`#define ADC5_KERNEL_INDEX_CORE5 (0xFFU)`
		`#define ADC6_KERNEL_INDEX_CORE5 (0xFFU)`
		`#define ADC7_KERNEL_INDEX_CORE5 (0xFFU)`
		`#define ADC8_KERNEL_INDEX_CORE5 (0xFFU)`
		`#define ADC9_KERNEL_INDEX_CORE5 (0xFFU)`
		`#define ADC10_KERNEL_INDEX_CORE5 (0xFFU)`
		`#define ADC11_KERNEL_INDEX_CORE5 (0U)`

1.1.40 Macro: ADCX_KERNEL_INDEX_CORE[Y]

Table 40 ADCX_KERNEL_INDEX_CORE[Y]

Name	ADCX_KERNEL_INDEX_CORE[Y]	
Description	Indicates the group of all the HW units assigned to the core 'Y' in the Adc_kKernelDataIndex structure. Where ('Y' = Core ID starting from 0 to Max Cores available in the derivative).	
Verification method	The macro is generated as a numeric value of all the groups assigned to the core 'Y'.	
Example(s)	**Action**	**Generated output**
	• Set ResourceMMasterCore as core0. • Assign HW unit0 to core0, • Assign HW unit5 to core0,	`/*** Group of all the indexes used for all the KERNELs on CPU Core0 ***/` `#define ADCX_KERNEL_INDEX_CORE0 ADC0_KERNEL_INDEX_CORE0,\`

	• Assign HW unit8 to core0.	`ADC1_KERNEL_INDEX_CORE0,\`
		`ADC2_KERNEL_INDEX_CORE0,\`
		`ADC3_KERNEL_INDEX_CORE0,\`
		`ADC4_KERNEL_INDEX_CORE0,\`
		`ADC5_KERNEL_INDEX_CORE0,\`
		`ADC6_KERNEL_INDEX_CORE0,\`
		`ADC7_KERNEL_INDEX_CORE0,\`
		`ADC8_KERNEL_INDEX_CORE0,\`
		`ADC9_KERNEL_INDEX_CORE0,\`
		`ADC10_KERNEL_INDEX_CORE0,\`
		`ADC11_KERNEL_INDEX_CORE0`

	• Set ResourceMMasterCore as core0.
	• Assign HW unit0 to core1,
	• Assign HW unit5 to core1,
	• Assign HW unit8 to core1.
		`/*** Group of all the indexes used for all the KERNELs on CPU Core1 ***/`
		`#define ADCX_KERNEL_INDEX_CORE1`
		`ADC0_KERNEL_INDEX_CORE1,\`
		`ADC1_KERNEL_INDEX_CORE1,\`
		`ADC2_KERNEL_INDEX_CORE1,\`
		`ADC3_KERNEL_INDEX_CORE1,\`
		`ADC4_KERNEL_INDEX_CORE1,\`
		`ADC5_KERNEL_INDEX_CORE1,\`
		`ADC6_KERNEL_INDEX_CORE1,\`
		`ADC7_KERNEL_INDEX_CORE1,\`
		`ADC8_KERNEL_INDEX_CORE1,\`
		`ADC9_KERNEL_INDEX_CORE1,\`
		`ADC10_KERNEL_INDEX_CORE1,\`
		`ADC11_KERNEL_INDEX_CORE1`

1.1.41 Macro: ADC_KERNEL_USED_COUNT_CORE[X]

Table 41 ADC_KERNEL_USED_COUNT_CORE[X]

Name	ADC_KERNEL_USED_COUNT_CORE[X]	
Description	Indicates the maximum number of HW units configured for core ‘X’. Where 'X' is ranging from 0 to 5 depends on HW Derivative.	
Verification method	The macro is generated as total number of HW units configured for core ‘X’. *Note: HW units not configured to core ‘X’ is assigned with 0U.*	
Example(s)	**Action**	**Generated output**
	• Set ResourceMMasterCore as core0. • Assign HW unit0 to core0,	`#define ADC_KERNEL_USED_COUNT_CORE0(3U)`

	• Assign HW unit5 to core0, • Assign HW unit8 to core0.	
	• Set ResourceMMasterCore as core3. • Assign HW unit11 to core3.	`#define ADC_KERNEL_USED_COUNT_CORE3(1U)`
	• Set ResourceMMasterCore as core5. • No HW unit configured to core5.	`#define ADC_KERNEL_USED_COUNT_CORE5(0U)`

1.1.42 Macro: ADC_MAX_KERNELS

Table 42 ADC_MAX_KERNELS

Name	ADC_MAX_KERNELS	
Description	Indicates the maximum number of kernels present in the HW.	
Verification method	The macro is generated as a numeric value which corresponds to the number of elements defined in 'Adc.MaxHwUnits' device specific resource properties file.	
Example(s)	**Action**	**Generated output**
	Generate Adc_Cfg.h	`#define ADC_MAX_KERNELS (12U)`

1.1.43 Macro: ADC_MAX_KERNEL_ID

Table 43 ADC_MAX_KERNEL_ID

Name	ADC_MAX_KERNEL_ID	
Description	Indicates the HW unit ID of the last kernel present in the HW.	
Verification method	The macro is generated as a numeric value which corresponds to the number of elements defined in 'Adc.MaxHwUnitId' device specific resource properties file.	
Example(s)	**Action**	**Generated output**
	Generate Adc_Cfg.h	`#define ADC_MAX_KERNEL_ID (12U)`

1.1.44 Macro: ADC_LAST_PRIMARY_KERNELID

Table 44 ADC_LAST_PRIMARY_KERNELID

Name	ADC_LAST_PRIMARY_KERNELID	
Description	Indicates the HW unit ID of the last primary kernel present in the HW.	
Verification method	The macro is generated as a numeric value which corresponds to the number of elements defined in 'Adc.LastPrimaryHwUnit' device specific resource properties file.	
Example(s)	**Action**	**Generated output**
	Generate Adc_Cfg.h	`#define ADC_LAST_PRIMARY_KERNELID (7U)`

1.1.45 Macro: ADC_REQSRC_COUNT

Table 45 ADC_REQSRC_COUNT

Name	ADC_REQSRC_COUNT	
Description	Indicates the request source available per kernel.	
Verification method	The macro is generated as a numeric value which corresponds to the number of elements defined in 'Adc.RSCount' device specific resource properties file.	
Example(s)	**Action**	**Generated output**
	Generate Adc_Cfg.h	`#define ADC_REQSRC_COUNT (3U)`

1.1.46 Macro: ADC_REQSRC_USED_COUNT

Table 46 ADC_REQSRC_USED_COUNT

Name	ADC_REQSRC_USED_COUNT	
Description	Indicates the request source used per kernel.	
Verification method	The macro is generated as a numeric value based on the number of elements defined in 'Adc.RSCount' device specific resource properties file. *Note: Value set in this macro is based on priority implementation in 'AdcGeneral/AdcPriorityImplementation'.*	
Example(s)	**Action**	**Generated output**
	• Configure AdcPriorityImplementation as ADC_PRIORITY_NONE. • Generate Adc_Cfg.h	`#define ADC_REQSRC_USED_COUNT (1U)`
	• Configure AdcPriorityImplementation as ADC_PRIORITY_NONE. • Generate Adc_Cfg.h	`#define ADC_REQSRC_USED_COUNT (3U)`

1.1.47 Macro: ADC_SECONDARY_KERNEL_AVAILABLE

Table 47 ADC_SECONDARY_KERNEL_AVAILABLE

Name	ADC_SECONDARY_KERNEL_AVAILABLE	
Description	Indicates whether secondary HwUnits are available in the hardware or not.	
Verification method	The macro is generated as STD_ON if 'Adc.MaxSecondaryHwUnits' is greater than 0 for the selected device specific resource properties file else it is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	• Adc.MaxSecondaryHwUnits = 4 • Generate Adc_Cfg.h	`#define ADC_SECONDARY_KERNEL_AVAILABLE`

		(STD_ON)
	• Adc.MaxSecondaryHwUnits = 0 • Generate Adc_Cfg.h	#define ADC_SECONDARY_KERNEL_AVAILABLE (STD_OFF)

1.1.48 Macro: ADC_GTM_AVAILABLE

Table 48 ADC_GTM_AVAILABLE

Name	ADC_GTM_AVAILABLE	
Description	Indicates whether GTM is available in the hardware or not.	
Verification method	The macro is generated as STD_ON if 'Gtm.Available' is set to true for the selected device specific resource properties file else it is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	• Gtm.Available = true • Generate Adc_Cfg.h	#define ADC_GTM_AVAILABLE (STD_ON)
	• Gtm.Available = false • Generate Adc_Cfg.h	#define ADC_GTM_AVAILABLE (STD_OFF)

1.1.49 Macro: AdcConf_AdcChannel_<AdcChannelName>

Table 49 AdcConf_AdcChannel_<AdcChannelName>

Name	AdcConf_AdcChannel_<AdcChannelName>	
Description	Indicates the symbolic name with AdcChannleId for each configured AdcChannel.	
Verification method	The macro is generated as a numeric value which is configured in 'AdcConfigSet/AdcHwUnit/AdcChannel. < AdcChannelId> is the name of the ADC channel's container name.	
Example(s)	**Action**	**Generated output**
	• Configure 4 Adc channels. • Container for Adc Channel ID 0 is named as AdcChannel_0. • Container for Adc Channel ID 1 is named as AdcChannel_1. • Container for Adc Channel ID 2 is named as AdcChannel_2 • Container for Adc Channel ID 3 is named as AdcChannel_3	#define AdcConf_AdcChannel_AdcChannel_0 (0U) #define AdcConf_AdcChannel_AdcChannel_1 (1U) #define AdcConf_AdcChannel_AdcChannel_2 (2U) #define AdcConf_AdcChannel_AdcChannel_3 (3U)

1.1.50 Macro: AdcConf_AdcGroup_<AdcGroupName>

Table 50 AdcConf_AdcGroup_<AdcGroupName>

Name	AdcConf_AdcGroup_<AdcGroupName>	
Description	Indicates the symbolic name with AdcGroupId for each configured AdcGroup.	
Verification method	The macro is generated as a numeric value which is configured in 'AdcConfigSet/AdcHwUnit/AdcGroup. < AdcGroupId> is the name of the ADC Group's container name.	
Example(s)	**Action**	**Generated output**

	• Configure 4 Adc Groups.
	• Container for Adc Group ID 0 is named as AdcGroup_0.
	• Container for Adc Group ID 1 is named as AdcGroup_1.
	• Container for Adc Group ID 2 is named as AdcGroup_2
	• Container for Adc Group ID 3 is named as AdcGroup_3
		`#define AdcConf_AdcGroup_AdcGroup_0 (0U)`
		`#define AdcConf_AdcGroup_AdcGroup_1 (1U)`
		`#define AdcConf_AdcGroup_AdcGroup_2 (2U)`
		`#define AdcConf_AdcGroup_AdcGroup_3 (3U)`

1.1.51 Macro: AdcConf_ AdcPowerStateConfig_<AdcPowerStateConfigName>

Table 51 AdcConf_ AdcPowerStateConfig _<AdcPowerStateConfigName>

Name	AdcConf_ AdcPowerStateConfig _<AdcPowerStateConfigName>	
Description	Indicates the symbolic name with AdcPowerState for each configured AdcPowerStateConfig.	
Verification method	The macro is generated as a numeric value which is configured in 'AdcGeneral / AdcPowerStateConfig. <AdcPowerStateConfigName> is the name of the ADC power state config container name.	
Example(s)	**Action**	**Generated output**

	• Configure 3 Adc power state configs.
	• Container for Adc power state 0 is named as AdcPowerStateConfig_0.
	• Container for Adc power state 1 is named as AdcPowerStateConfig_1.
	• Container for Adc power state 3 is named as AdcPowerStateConfig_3.
		`#define AdcConf_AdcPowerStateConfig _AdcPowerStateConfig_0 (0U)`
		`#define AdcConf_AdcPowerStateConfig _AdcPowerStateConfig_1 (1U)`
		`#define AdcConf_AdcPowerStateConfig _AdcPowerStateConfig_3 (3U)`

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

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

1.2.1 Structure: Adc_Config[_<variant>]

Table 52 Adc_Config[_<variant>]

Name	Adc_Config[_<variant>]	
Type	Adc_ConfigType	
Description	Root configuration structure of ADC driver which will be used during initialization.	
Verification method	The generated structure is present in Adc[_<variant>]_PBcfg.c file. <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 HwUnit0 to core0 and HwUnit1 to core1 in ResourceMAllocation of resource manager (variant unaware)	`const Adc_ConfigType Adc_Config=`
		`{`
		`  &Adc_kGlob_Config, /* Global Configuration */`
		`  {`
		`  &Adc_kCore0_Config, /* Core0 Configuration */`
		`  &Adc_kCore1_Config, /* Core1 Configuration */`
		`  (const Adc_CoreConfigType*)0U, /* Core2 Configuration */`
		`  (const Adc_CoreConfigType*)0U /* Core3 Configuration */`
		`  }`
		`};`

	Configure HwUnit0 to core0 and HwUnit1 to core1 in ResourceMAllocation of resource manager (variant Petrol)	`const Adc_ConfigType Adc_Config_Petrol=`
		`{`
		`  &Adc_kGlob_Config_Petrol, /* Global Configuration */`
		`  {`
		`  &Adc_kCore0_Config_Petrol, /* Core0 Configuration */`
		`  &Adc_kCore1_Config_Petrol, /* Core1 Configuration */`
		`  (const Adc_CoreConfigType*)0U, /* Core2 Configuration */`
		`  (const Adc_CoreConfigType*)0U /* Core3 Configuration */`
		`  }`
		`};`

1.2.1.1 Member: GlobalCfgPtr

Table 53 GlobalCfgPtr

Name	GlobalCfgPtr	
Type	Adc_GlobalCfgType*	
Description	Global configuration.	
Verification method	The generated structure member is present in the Adc_Config[_<variant>] structure. For a variant-aware configuration, Member name is appended with the variant name. For variant-unaware configuration <variant> is ignored.	
Example(s)	**Action**	**Generated output**
	Global configuration (variant Petrol)	`&Adc_kGlob_Config_Petrol, /* Global Configuration */`
	Global configuration (variant unaware)	`&Adc_kGlob_Config, /* Global Configuration */`

1.2.1.2 Member: CoreCfgPtr [6]

Table 54 CoreCfgPtr[6]

Name	CoreCfgPtr [6]	
Type	Adc_CoreConfigType *	
Description	Indicates the array of core-specific configuration.	
Verification method	The generated structure member is present in the Adc_Config[_<variant>] structure. If a Core<x> is allocated at least one HW unit, then the element <x> shall be generated as '&Adc_kCore<x>_Config' else 'NULL_PTR' is generated.(x in range 0 to 5).	
Example(s)	**Action**	**Generated output**

	Configure HwUnit0 to core0 and all other HwUnits to core 1 in ResourceMAllocation of resource manager. (variant unaware)	`{`
		`  &Adc_kHwUnit0_Config, /* HW Unit 1 Configuration */`
		`  &Adc_kHwUnit1_Config, /* HW Unit 2 Configuration */`
		`  (Adc_HwUnitCfgType*)0U, /* HW Unit 3 Configuration */`
		`  (Adc_HwUnitCfgType*)0U, /* HW Unit 4 Configuration */`
		`  (Adc_HwUnitCfgType*)0U, /* HW Unit 5 Configuration */`
		`  (Adc_HwUnitCfgType*)0U, /* HW Unit 6 Configuration */`
		`  (Adc_HwUnitCfgType*)0U, /* HW Unit 7 Configuration */`
		`  (Adc_HwUnitCfgType*)0U, /* HW Unit 8 Configuration */`
		`  (Adc_HwUnitCfgType*)0U, /* HW`

	Unit 9 Configuration */ (Adc_HwUnitCfgType*)0U, /* HW Unit 10 Configuration */ (Adc_HwUnitCfgType*)0U, /* HW Unit 11 Configuration */ (Adc_HwUnitCfgType*)0U /* HW Unit 12 Configuration */ }

Configure HwUnit0 to core0 and all other HwUunits to core 1 in ResourceMAllocation of resource manager.
(variant Petrol)
	{
	&Adc_kHwUnit0_Config_Petrol, /* HW Unit 1 Configuration */
	&Adc_kHwUnit1_Config_Petrol, /* HW Unit 2 Configuration */
	(Adc_HwUnitCfgType*)0U, /* HW Unit 3 Configuration */
	(Adc_HwUnitCfgType*)0U, /* HW Unit 4 Configuration */
	(Adc_HwUnitCfgType*)0U, /* HW Unit 5 Configuration */
	(Adc_HwUnitCfgType*)0U, /* HW Unit 6 Configuration */
	(Adc_HwUnitCfgType*)0U, /* HW Unit 7 Configuration */
	(Adc_HwUnitCfgType*)0U, /* HW Unit 8 Configuration */
	(Adc_HwUnitCfgType*)0U, /* HW Unit 9 Configuration */
	(Adc_HwUnitCfgType*)0U, /* HW Unit 10 Configuration */
	(Adc_HwUnitCfgType*)0U, /* HW Unit 11 Configuration */
	(Adc_HwUnitCfgType*)0U /* HW Unit 12 Configuration */
	}

Configure all HwUnits to all cores except core 0 in ResourceMAllocation of resource manager. (variant unaware)	{ (Adc_HwUnitCfgType*)0U, /* HW Unit 1 Configuration */ &Adc_kHwUnit1_Config, /* HW Unit 2 Configuration */ &Adc_kHwUnit2_Config, /* HW Unit 3 Configuration */ &Adc_kHwUnit3_Config, /* HW Unit 4 Configuration */ &Adc_kHwUnit4_Config, /* HW Unit

	5 Configuration */ &Adc_kHwUnit5_Config, /* HW Unit 6 Configuration */ &Adc_kHwUnit6_Config, /* HW Unit 7 Configuration */ &Adc_kHwUnit7_Config, /* HW Unit 8 Configuration */ &Adc_kHwUnit8_Config, /* HW Unit 9 Configuration */ &Adc_kHwUnit9_Config, /* HW Unit 10 Configuration */ &Adc_kHwUnit10_Config, /* HW Unit 11 Configuration */ &Adc_kHwUnit11_Config /* HW Unit 12 Configuration */ }

Configure all HwUnits to all cores except core 0 in ResourceMAllocation of resource manager.
(variant Petrol)
	{
	(Adc_HwUnitCfgType*)0U, /* HW Unit 1 Configuration */
	&Adc_kHwUnit1_Config_Petrol, /* HW Unit 2 Configuration */
	&Adc_kHwUnit2_Config_Petrol, /* HW Unit 3 Configuration */
	&Adc_kHwUnit3_Config_Petrol, /* HW Unit 4 Configuration */
	&Adc_kHwUnit4_Config_Petrol, /* HW Unit 5 Configuration */
	&Adc_kHwUnit5_Config_Petrol, /* HW Unit 6 Configuration */
	&Adc_kHwUnit6_Config_Petrol, /* HW Unit 7 Configuration */
	&Adc_kHwUnit7_Config_Petrol, /* HW Unit 8 Configuration */
	&Adc_kHwUnit8_Config_Petrol, /* HW Unit 9 Configuration */
	&Adc_kHwUnit9_Config_Petrol, /* HW Unit 10 Configuration */
	&Adc_kHwUnit10_Config_Petrol, /* HW Unit 11 Configuration */
	&Adc_kHwUnit11_Config_Petrol /* HW Unit 12 Configuration */
	}

1.2.2 Structure: Adc_kCore<x>_Config[_<variant>]

Table 55 Adc_kCore<x>_Config[_<variant>]

Name	Adc_kCore<x>_Config[_<variant>]	
Type	Adc_CoreConfigType	
Description	Configuration structure of ADC driver for Core which will be referenced in root configuration structure. ('x' = Core ID starting from 0 to Max Cores available in the derivative).	
Verification method	The generated file has this structure when HW unit 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 HwUnit0, HwUnit1 and HwUnit2 to Core0 in ResourceMAllocation of resource manager. (variant unaware)	`static const Adc_CoreConfigType Adc_kCore0_Config=`
		`{`
		`  {`
		`  &Adc_kHwUnit0_Config, /* HW Unit 1 Configuration */`
		`  &Adc_kHwUnit1_Config, /* HW Unit 2 Configuration */`
		`  &Adc_kHwUnit2_Config,, /* HW Unit 3 Configuration */`
		`  (Adc_HwUnitCfgType*)0U, /* HW Unit 4 Configuration */`
		`  (Adc_HwUnitCfgType*)0U, /* HW Unit 5 Configuration */`
		`  (Adc_HwUnitCfgType*)0U, /* HW Unit 6 Configuration */`
		`  (Adc_HwUnitCfgType*)0U, /* HW Unit 7 Configuration */`
		`  (Adc_HwUnitCfgType*)0U, /* HW Unit 8 Configuration */`
		`  (Adc_HwUnitCfgType*)0U, /* HW Unit 9 Configuration */`
		`  (Adc_HwUnitCfgType*)0U, /* HW Unit 10 Configuration */`
		`  (Adc_HwUnitCfgType*)0U, /* HW Unit 11 Configuration */`
		`  (Adc_HwUnitCfgType*)0U /* HW Unit 12 Configuration */`
		`  }`
		`};`

	Configure HwUnit3, HwUnit4	`static const Adc_CoreConfigType`

	and HwUnit5 to Core5 in ResourceMAllocation of resource manager. (variant Petrol)	Adc_kCore0_Config_Petrol=
		{
		  {
		  (Adc_HwUnitCfgType*)0U, /* HW Unit 1 Configuration */
		  (Adc_HwUnitCfgType*)0U, /* HW Unit 2 Configuration */
		  (Adc_HwUnitCfgType*)0U, /* HW Unit 3 Configuration */
		  &Adc_kHwUnit3_Config_Petrol, /* HW Unit 4 Configuration */
		  &Adc_kHwUnit4_Config_Petrol, /* HW Unit 5 Configuration */
		  &Adc_kHwUnit5_Config_Petrol, /* HW Unit 6 Configuration */
		  (Adc_HwUnitCfgType*)0U, /* HW Unit 7 Configuration */
		  (Adc_HwUnitCfgType*)0U, /* HW Unit 8 Configuration */
		  (Adc_HwUnitCfgType*)0U, /* HW Unit 9 Configuration */
		  (Adc_HwUnitCfgType*)0U, /* HW Unit 10 Configuration */
		  (Adc_HwUnitCfgType*)0U, /* HW Unit 11 Configuration */
		  (Adc_HwUnitCfgType*)0U /* HW Unit 12 Configuration */
		  }
		};

1.2.2.1 Member: HwUnitCfgPtr

Table 56 HwUnitCfgPtr

Name	HwUnitCfgPtr	
Type	Adc_HwUnitCfgType *	
Description	Indicates the array of HW unit specific configuration.	
Verification method	The generated structure member is present in the Adc_kCore<x>_Config[_<variant>] structure. If a Core is configured with at least one HW unit, then the HW unit will be generated as ‘Adc_kHwUnit<x>_Config[_<variant>]’ else ‘NULL_PTR’ is generated.('x' = Hw Unit ID starting from 0 to Max HW Units available in the derivative).	
Example(s)	**Action**	**Generated output**
	Configure HwUnit0, HwUnit1 and HwUnit2 to Core0 in ResourceMAllocation of resource manager.	{ &Adc_kHwUnit0_Config, /* HW Unit 1 Configuration */

	(variant unaware)	`&Adc_kHwUnit1_Config, /* HW Unit 2 Configuration */`
		`&Adc_kHwUnit2_Config,, /* HW Unit 3 Configuration */`
		`(Adc_HwUnitCfgType*)0U, /* HW Unit 4 Configuration */`
		`(Adc_HwUnitCfgType*)0U, /* HW Unit 5 Configuration */`
		`(Adc_HwUnitCfgType*)0U, /* HW Unit 6 Configuration */`
		`(Adc_HwUnitCfgType*)0U, /* HW Unit 7 Configuration */`
		`(Adc_HwUnitCfgType*)0U, /* HW Unit 8 Configuration */`
		`(Adc_HwUnitCfgType*)0U, /* HW Unit 9 Configuration */`
		`(Adc_HwUnitCfgType*)0U, /* HW Unit 10 Configuration */`
		`(Adc_HwUnitCfgType*)0U, /* HW Unit 11 Configuration */`
		`(Adc_HwUnitCfgType*)0U /* HW Unit 12 Configuration */`
		`}`

	Configure HwUnit3, HwUnit4 and HwUnit5 to Core5 in ResourceMAllocation of resource manager.
	(variant Petrol)
		`{`
		`(Adc_HwUnitCfgType*)0U, /* HW Unit 1 Configuration */`
		`(Adc_HwUnitCfgType*)0U, /* HW Unit 2 Configuration */`
		`(Adc_HwUnitCfgType*)0U, /* HW Unit 3 Configuration */`
		`&Adc_kHwUnit3_Config_Petrol, /* HW Unit 4 Configuration */`
		`&Adc_kHwUnit4_Config_Petrol, /* HW Unit 5 Configuration */`
		`&Adc_kHwUnit5_Config_Petrol, /* HW Unit 6 Configuration */`
		`(Adc_HwUnitCfgType*)0U, /* HW Unit 7 Configuration */`
		`(Adc_HwUnitCfgType*)0U, /* HW Unit 8 Configuration */`
		`(Adc_HwUnitCfgType*)0U, /* HW Unit 9 Configuration */`
		`(Adc_HwUnitCfgType*)0U, /* HW Unit 10 Configuration */`
		`(Adc_HwUnitCfgType*)0U, /* HW Unit 11`

		Configuration */ (Adc_HwUnitCfgType*)0U /* HW Unit 12 Configuration */ }

1.2.3 Structure: Adc_kGlob_Config[_<variant>]

Table 57 Adc_kGlob_Config[_<variant>]

Name	Adc_kGlob_Config[_<variant>]	
Type	Adc_GlobalCfgType	
Description	Global configuration structure for all Hw Units of ADC driver which will be referenced in root configuration structure.	
Verification method	The generated structure member is present in the Adc_Config[_<variant>] structure. For a variant-aware configuration, Member name is appended with the <variant> name. For variant-unaware configuration <variant> is ignored.	
Example(s)	**Action**	**Generated output**

	Global configuration (variant Petrol)	static const Adc_GlobalCfgType Adc_kGlob_Config_Petrol=
		{
		0x00000000U, /*Configuration value for GLOBCFG register */
		0x00000080U, /*Configuration value for GLOBICLASS0 register */
		0x00000040U /*Configuration value for GLOBICLASS1 register */
		0x00000080U, /*Configuration value for EMUXSEL register */
		};

	Global configuration (variant unaware)	static const Adc_GlobalCfgType Adc_kGlob_Config=
		{
		0x00000000U, /*Configuration value for GLOBCFG register */
		0x00000080U, /*Configuration value for GLOBICLASS0 register */
		0x00000040U /*Configuration value for GLOBICLASS1 register */
		0x00000080U, /*Configuration value for EMUXSEL register */
		};

1.2.3.1 Member: GlobalCfg

Table 58 GlobalCfg

Name	GlobalCfg	
Type	uint32	
Description	Indicates the global configuration value of all HW units.	
Verification method	The structure member is generated as a value of global configuration for GLOBCFG register. Bit 12 stores value configured in AdcSyncClockDisable. Bits 13-14 store value configured in AdcSupplyVoltage. Other bits are always generated as 0.	
Example(s)	**Action**	**Generated output**
	• Configure AdcSyncClockDisable with 0. • Configure AdcSupplyVoltage with ADC_VOLTAGE_3P3V.	`0x00004000U, /*Configuration value for GLOBCFG register */`
	• Configure AdcSyncClockDisable with 1. • Configure AdcSupplyVoltage with ADC_VOLTAGE_5V.	`0x00003000U, /*Configuration value for GLOBCFG register */`

1.2.3.2 Member: GlobInputClass0Cfg

Table 59 GlobInputClass0Cfg

Name	GlobInputClass0Cfg	
Type	uint32	
Description	Indicates the global input class0 configuration value of all HW units.	
Verification method	The structure member is generated as a value of global input class0 configuration for GLOBICLASS0 register. Bits 0-4 store value configured in AdcChSampleTime. Bits 6-7 store value configured in AdcChPreChargeClkCycles. Bits 8-9 store value configured in AdcChConvMode. Bit 10 stores value configured in AdcChSESPSEnable. Bits 16-20 store value configured in AdcEmuxChSampleTime when AdcEmuxEnable parameter is enabled. Bits 22-23 store value configured in AdcEmuxChPreChargeClkCycles when AdcEmuxEnable parameter is enabled. Bits 24-25 store value configured in AdcEmuxChConvMode when AdcEmuxEnable parameter is enabled. Bit 26 stores value configured in AdcEmuxChSESPSEnable when AdcEmuxEnable parameter is enabled. Other bits are always generated as 0.	
Example(s)	**Action**	**Generated output**

• Configure AdcChSampleTime with 0. • Configure AdcChPreChargeClkCycles with ADC_INPUT_PRECHARGE_CYCLES_16. • Configure AdcChConvMode with ADC_NOISE_REDUCTION_STEPS_0. • Configure AdcChSESPSEnable with 0. • Disable parameter AdcEmuxEnable	`0x00000080U, /*Configuration value for GLOBICLASS0 register */`
• Configure AdcChSampleTime with 10. • Configure AdcChPreChargeClkCycles with ADC_INPUT_PRECHARGE_CYCLES_8. • Configure AdcChConvMode with ADC_NOISE_REDUCTION_STEPS_1. • Configure AdcChSESPSEnable with 1. • Enable parameter AdcEmuxEnable • Configure AdcEmuxChSampleTime with 0. • Configure AdcEmuxChPreChargeClkCycles with ADC_INPUT_PRECHARGE_CYCLES_16. • Configure AdcEmuxChConvMode with ADC_NOISE_REDUCTION_STEPS_0. • Configure AdcEmuxChSESPSEnable with 0.	`0x0080054aU, /*Configuration value for GLOBICLASS0 register */`

1.2.3.3 Member: GlobInputClass1Cfg

Table 60 GlobInputClass1Cfg

Name	GlobInputClass0Cfg
Type	uint32

Description	Indicates the global input class1 configuration value of all HW units.	
Verification method	The structure member is generated as a value of global input class1 configuration for GLOBICLASS1 register. Bits 0-4 store value configured in AdcChSampleTime. Bits 6-7 store value configured in AdcChPreChargeClkCycles. Bits 8-9 store value configured in AdcChConvMode. Bit 10 stores value configured in AdcChSESPSEnable. Bits 16-20 store value configured in AdcEmuxChSampleTime when AdcEmuxEnable parameter is enabled. Bits 22-23 store value configured in AdcEmuxChPreChargeClkCycles when AdcEmuxEnable parameter is enabled. Bits 24-25 store value configured in AdcEmuxChConvMode when AdcEmuxEnable parameter is enabled. Bit 26 stores value configured in AdcEmuxChSESPSEnable when AdcEmuxEnable parameter is enabled. Other bits are always generated as 0.	
Example(s)	**Action**	**Generated output**
	• Configure AdcChSampleTime with 0. • Configure AdcChPreChargeClkCycles with ADC_INPUT_PRECHARGE_CYCLES_16. • Configure AdcChConvMode with ADC_NOISE_REDUCTION_STEPS_0. • Configure AdcChSESPSEnable with 0. • Configure EmuxEnable with false	`0x00000080U, /*Configuration value for GLOBICLASS1 register */`
	• Configure AdcChSampleTime with 10. • Configure AdcChPreChargeClkCycles with ADC_INPUT_PRECHARGE_CYCLES_8. • Configure AdcChConvMode with ADC_NOISE_REDUCTION_STEPS_1. • Configure AdcChSESPSEnable with 1. • Configure EmuxEnable with true • Configure AdcEmuxChSampleTime with 0. • Configure AdcEmuxChPreChargeClkCycles with	`0x0080054aU, /*Configuration value for GLOBICLASS1 register */`

	ADC_INPUT_PRECHARGE_CYCLES_16. • Configure AdcEmuxChConvMode with ADC_NOISE_REDUCTION_STEPS_0. • Configure AdcEmuxChSESPSEnable with 0.	

1.2.3.4 Member: GlobEmuxGrpInterfaceCfg

Table 61 GlobEmuxGrpInterfaceCfg

Name	GlobEmuxGrpInterfaceCfg	
Type	uint32	
Description	Indicates the interface selection for external multiplexer.	
Verification method	The structure member is generated as a value of external multiplexer interface configuration for EMUXSEL register. Bits 0-3 store value configured in AdcEmuxGroupInterface0. Bits 4-7 store value configured in AdcEmuxGroupInterface1. Other bits are always generated as 0.	
Example(s)	**Action**	**Generated output**
	• Configure AdcEmuxGroupInterface0 with HWUNIT_ADC0. • Configure AdcEmuxGroupInterface1 with HWUNIT_ADC8.	`0x00000080U, /*Configuration value for EMUXSEL register */`
	• Configure AdcEmuxGroupInterface0 with HWUNIT_ADC2. • Configure AdcEmuxGroupInterface1 with HWUNIT_ADC4.	`0x00000042U, /*Configuration value for EMUXSEL register */`

1.2.4 Structure: Adc_kHwUnit<x>_Config[_<variant>]

Table 62 Adc_kHwUnit<x>_Config[_<variant>]

Name	Adc_kHwUnit<x>_Config[_<variant>]
Type	Adc_HwUnitCfgType
Description	Configuration structure of ADC driver for HW unit which will be referenced in core specific configuration structure. ('x' = Hw Unit ID starting from 0 to Max HW Units available in the derivative).
Verification method	The generated structure member is present in the Adc_kCore<x>_Config[_<variant>] structure. For a variant-aware configuration, Member name is appended with the <variant> name. For variant-unaware configuration <variant> is ignored.

Example(s)	Action	Generated output

	HwUnit 0 configuration (variant Petrol)	`static const Adc_HwUnitCfgType Adc_kHwUnit0_Config_Petrol=`
		`{`
		`  &Adc_kHwUnit0Hw_Config_Petrol, /*Analog Converter Configuration*/`
		`  &Adc_kHwUnit0Ch_Config_Petrol[0U], /*Channel Configuration structure*/`
		`  &Adc_kHwUnit0Grp_Config_Petrol[0U], /*Group Configuration structure*/`
		`  0x00007f77U, /* Mask for SW triggered groups*/`
		`  0x00000088U,/* Mask for HW triggered groups*/`
		`  ADC_SYNC_CONV_MODE_MASTER, /* Synchronous conversion mode */`
		`  { 0x01U, 0x02U, 0xffU }, /* Slave Kernels */`
		`  15U, /* Group Count for HW Unit 0*/`
		`  7U /* Bit Mask for SRNs used for HW Unit 0*/`
		`};`

	HwUnit 0 configuration (variant unaware)	`static const Adc_HwUnitCfgType Adc_kHwUnit0_Config=`
		`{`
		`  &Adc_kHwUnit0Hw_Config, /*Analog Converter Configuration*/`
		`  &Adc_kHwUnit0Ch_Config[0U], /*Channel Configuration structure*/`
		`  &Adc_kHwUnit0Grp_Config[0U], /*Group Configuration structure*/`
		`  0x00007f77U, /* Mask for SW triggered groups*/`
		`  0x00000088U,/* Mask for HW triggered groups*/`
		`  ADC_SYNC_CONV_MODE_MASTER, /* Synchronous conversion mode */`
		`  { 0x01U, 0x02U, 0xffU }, /* Slave Kernels */`
		`  15U, /* Group Count for HW Unit 0*/`
		`  7U /* Bit Mask for SRNs used for HW Unit 0*/`
		`};`

1.2.4.1 Member: HwCfgPtr

Table 63 HwCfgPtr

Name	HwCfgPtr	
Type	Adc_HwCfgType*	
Description	Indicates the analog converter configuration structure.	
Verification method	The generated structure member is present in the Adc_kHwUnit<x>_Config[_<variant>] structure. For a variant-aware configuration, Member name is appended with the variant name. For variant-unaware configuration <variant> is ignored.	
Example(s)	**Action**	**Generated output**
	Configure HwUnit0 in AdcHwUnit container. (variant unaware)	`&Adc_kHwUnit0Hw_Config /*Analog Converter Configuration*/`
	Configure HwUnit1 in AdcHwUnit container. (variant Petrol)	`&Adc_kHwUnit1Hw_Config_Petrol /*Analog Converter Configuration*/`

1.2.4.2 Member: ChCfgPtr

Table 64 ChCfgPtr

Name	ChCfgPtr	
Type	Adc_ChannelCfgType*	
Description	Indicates the channel configuration structure.	
Verification method	The generated structure member is present in the Adc_kHwUnit<x>_Config[_<variant>] structure. For a variant-aware configuration, Member name is appended with the variant name. For variant-unaware configuration <variant> is ignored.	
Example(s)	**Action**	**Generated output**
	Configure HwUnit0 in AdcHwUnit container. (variant unaware)	`&Adc_kHwUnit0Ch_Config[0U] /*Channel Configuration structure*/`
	Configure HwUnit1 in AdcHwUnit container. (variant Petrol)	`&Adc_kHwUnit1Ch_Config_Petrol[0U] /*Channel Configuration structure*/`

1.2.4.3 Member: GrpCfgPtr

Table 65 GrpCfgPtr

Name	GrpCfgPtr	
Type	Adc_GroupCfgType*	
Description	Indicates the group configuration structure.	
Verification method	The generated structure member is present in the Adc_kHwUnit<x>_Config[_<variant>] structure. For a variant-aware configuration, Member name is appended with the variant name. For variant-unaware configuration <variant> is ignored.	
Example(s)	**Action**	**Generated output**

Configure HwUnit0 in AdcHwUnit container. (variant unaware)	`&Adc_kHwUnit0Grp_Config[0U] /*Group Configuration structure*/`
Configure HwUnit1 in AdcHwUnit container. (variant Petrol)	`&Adc_kHwUnit1Grp_Config[0U] /*Group Configuration structure*/`

1.2.4.4 Member: SwTrigGrpMask

Table 66 SwTrigGrpMask

Name	SwTrigGrpMask	
Type	uint32	
Description	Indicates the mask values of SW triggered groups configured for the hardware unit.	
Verification method	The structure member is generated as mask values of configured SW triggered groups.	
Example(s)	**Action**	**Generated output**
	Configure AdcGroup0 and AdcGroup1 to HwUnit0.	`0x00000003U /* Mask for SW triggered groups*/`
	Configure AdcGroup0, AdcGroup1, AdcGroup2 and AdcGroup3 to HwUnit0.	`0x0000000FU /* Mask for SW triggered groups*/`
	Configure AdcGroup0, AdcGroup1, AdcGroup2, AdcGroup3, AdcGroup4, AdcGroup5, AdcGroup6 and AdcGroup7 to HwUnit0.	`0x000000FFU /* Mask for SW triggered groups*/`

1.2.4.5 Member: HwTrigGrpMask

Table 67 HwTrigGrpMask

Name	HwTrigGrpMask	
Type	uint32	
Description	Indicates the mask values of HW triggered groups configured for the hardware unit.	
Verification method	The structure member is generated as a mask values of configured HW triggered groups.	
Example(s)	**Action**	**Generated output**
	Configure AdcGroup0 and AdcGroup1 to HwUnit0.	`0x00000003U /* Mask for HW triggered groups*/`
	Configure AdcGroup0, AdcGroup1, AdcGroup2 and AdcGroup3 to HwUnit0.	`0x0000000FU /* Mask for HW triggered groups*/`
	Configure AdcGroup0, AdcGroup1, AdcGroup2, AdcGroup3,	`0x000000FFU /* Mask for SW triggered`

	AdcGroup4, AdcGroup5, AdcGroup6 and AdcGroup7 to HwUnit0.	`groups*/`

1.2.4.6 Member: SyncConvMode

Table 68 SyncConvMode

Name	SyncConvMode	
Type	Adc_SyncConvModeType	
Description	Indicates the sync conversion mode of the hardware unit.	
Verification method	The structure member is generated as a sync conversion mode configured for the hardware unit. *Note: This parameter is user configurable only when 'AdcGeneral/AdcSyncConvEnable' is enabled.*	
Example(s)	**Action**	**Generated output**
	Configure AdcSyncConvMode as ADC_STAND_ALONE to HwUnit0.	`ADC_SYNC_CONV_MODE_NONE /* Synchronous conversion mode */`
	Configure AdcSyncConvMode as ADC_SYNC_MASTER to HwUnit0.	`ADC_SYNC_CONV_MODE_MASTER /* Synchronous conversion mode */`
	Configure AdcSyncConvMode as ADC_SYNC_SLAVE to HwUnit0.	`ADC_SYNC_CONV_MODE_SLAVE /* Synchronous conversion mode */`

1.2.4.7 Member: SlaveKernels[ADC_KERNELS_PER_SYNGRP - 1U]

Table 69 SlaveKernels[ADC_KERNELS_PER_SYNGRP - 1U]

Name	SlaveKernels[ADC_KERNELS_PER_SYNGRP - 1U]	
Type	uint8	
Description	Indicates the array of slave kernels configured for the master kernel of synchronization group.	
Verification method	The structure member is generated with an array of base address of slave kernels configured for the master kernel of synchronization group. The value of ADC_KERNELS_PER_SYNGRP is 4. *Note: This parameter is user configurable only when 'AdcGeneral/AdcSyncConvEnable' is enabled.*	
Example(s)	**Action**	**Generated output**
	Configure	`{ 0xffU, 0xffU, 0xffU } /* Slave Kernels`

	AdcSyncConvMode as ADC_STAND_ALONE for all HwUnits of synchronization group.	*/
	• Configure AdcSyncConvMode as ADC_STAND_MASTER for HwUnit0. • Configure AdcSyncConvMode as ADC_STAND_SLAVE for HwUnit1 and HwUnit2. • Configure AdcSyncConvMode as ADC_STAND_ALONE for HwUnit3.	{ 0x01U, 0x02U, 0xffU } /* Slave Kernels */
	• Configure AdcSyncConvMode as ADC_STAND_MASTER for HwUnit0. • Configure AdcSyncConvMode as ADC_STAND_SLAVE for HwUnit1,HwUnit2 and HwUnit3.	{ 0x01U, 0x02U, 0x03U } /* Slave Kernels */

1.2.4.8 Member: NoOfGroups

Table 70 NoOfGroups

Name	NoOfGroups	
Type	uint8	
Description	Indicates the values of number of groups configured for the hardware unit.	
Verification method	The structure member is generated as a value of number of groups configured for the hardware unit.	
Example(s)	**Action**	**Generated output**
	Configure AdcGroup0, AdcGroup1 and AdcGroup 2 to HwUnit0.	3U /* Group Count for HW Unit 0*/
	Configure AdcGroups from 0 to 14 to the HwUnit0.	15U /* Group Count for HW Unit 0*/

1.2.4.9 Member: SRNUsed

Table 71 SRNUsed

Name	SRNUsed

Type	uint8	
Description	Indicates the values of number of SRNs used for the hardware unit.	
Verification method	The structure member is generated as a value of number of SRNs used for the hardware unit. *Note: SRN number is derived based on the configuration parameter of Priority Implementation and limit checking feature.*	
Example(s)	**Action**	**Generated output**
	• Configure AdcPriorityImplementation as ADC_PRIORITY_NONE. • Configure AdcChannelLimitCheck as Enabled.	`9U /* Bit Mask for SRNs used for HW Unit 0*/`
	• Configure AdcPriorityImplementation as ADC_PRIORITY_HW_SW. • Configure AdcChannelLimitCheck as Disabled.	`7U /* Bit Mask for SRNs used for HW Unit 0*/`

1.2.5 Structure: Adc_kHwUnit[x]Hw_Config[_<variant>]

Table 72 Adc_kHwUnit[x]Hw_Config[_<variant>]

Name	Adc_kHwUnit[x]Hw_Config[_<variant>]	
Type	Adc_HwCfgType	
Description	Configuration structure of ADC driver for an analog converter specific configuration values for HW unit which will be referenced in HW unit specific configuration structure. ('x' = Hw Unit ID starting from 0 to Max HW Units available in the derivative).	
Verification method	The generated structure member is present in the Adc_kHwUnit<x>_Config[_<variant>] structure. For a variant-aware configuration, Member name is appended with the <variant> name. For variant-unaware configuration <variant> is ignored.	
Example(s)	**Action**	**Generated output**

	HwUnit0 configuration (variant Petrol)	`static const Adc_HwCfgType Adc_kHwUnit0Hw_Config_Petrol=`
		`{`
		`  0x02180005U, /*Configuration value for G0ANCFG register*/`
		`  0x00000003U, /*Configuration value for G0ARBCFG register*/`
		`  0x01000000U, /*Configuration value for G0ARBPR register*/`

		0x000003dfU, /*Configuration value for G0ICLASS0 register*/ 0x00000682U, /*Configuration value for G0ICLASS1 register*/ 0x00000040U /*Configuration value for G0SYNCTR register*/ };

	Hw unit 0 configuration (variant unaware)	static const Adc_HwCfgType Adc_kHwUnit0Hw_Config=
		{
		0x02180005U, /*Configuration value for G0ANCFG register*/
		0x00000003U, /*Configuration value for G0ARBCFG register*/
		0x01000000U, /*Configuration value for G0ARBPR register*/
		0x000003dfU, /*Configuration value for G0ICLASS0 register*/
		0x00000682U, /*Configuration value for G0ICLASS1 register*/
		0x00000040U /*Configuration value for G0SYNCTR register*/
		};

1.2.5.1 Member: GrpAnalogFuncCfg

Table 73 GrpAnalogFuncCfg

Name	GrpAnalogFuncCfg	
Type	uint32	
Description	Indicates the analog configuration value of HW unit <x>.('x' = Hw Unit ID starting from 0 to Max HW Units available in the derivative).	
Verification method	The structure member is generated as a value of analog configuration for GxANCFG register. Bit 0 stores the value configured in AdcIdlePrechargeEnable. Bit 1 stores the value configured in AdcInputBufferEnable. Bit 2 stores the value configured in AdcPrechargeReference. Bit 3 stores the value configured in AdcReferencePrechargePhases. Bits 4-5 store the value configured in AdcCalibrationSampleTime. Bit 6 stores the value configured in AdcPostCalibrationDisable. Bits 16-18 store the value configured in AdcAnalogClockSyncDelay. Bit 19 stores the value configured in AdcSampleSyncEnable. Bits 20-24 store the value configured in AdcPrescale. Bits 25 stores the value configured in AdcMSBDoubleClkEnable. Other bits are always generated as 0.	
Example(s)	**Action**	**Generated output**
	• Configure	0x02180005U, /*Configuration value for

AdcIdlePrechargeEnable as Enabled to HwUnit0. • Configure AdcInputBufferEnable as Disabled to HwUnit0. • Configure AdcPrechargeReference as ADC_VDD_VSM_USED to HwUnit0. • Configure AdcReferencePrechargePhases as ADC_PRECHARGE_PHASE_1 to HwUnit0. • Configure AdcCalibrationSampleTime as ADC_CAL_TIME_2_TIMES_TADCI to HwUnit0. • Configure AdcPostCalibrationDisable as Disabled to HwUnit0. • Configure AdcAnalogClockSyncDelay as 0 to HwUnit0. • Configure AdcSampleSyncEnable as Enabled to HwUnit0. • Configure value in AdcPrescale as 2 to HwUnit0. • Configure AdcMSBDoubleClkEnable as Enabled to HwUnit0.	G0ANCFG register*/
• Configure AdcIdlePrechargeEnable as Enabled to HwUnit1. • Configure AdcInputBufferEnable as Enabled to HwUnit1. • Configure AdcPrechargeReference as ADC_VDD_VSM_NOT_USED to HwUnit1. • Configure AdcReferencePrechargePhases as	0x0019005bU, /*Configuration value for G1ANCFG register*/

	ADC_PRECHARGE_PHASE_2 to HwUnit1. • Configure AdcCalibrationSampleTime as ADC_CAL_TIME_4_TIMES_TADCI to HwUnit1. • Configure AdcPostCalibrationDisable as Enabled to HwUnit1. • Configure AdcAnalogClockSyncDelay as 1 to HwUnit1. • Configure AdcSampleSyncEnable as Enabled to HwUnit1. • Configure value in AdcPrescale as 2 to HwUnit1. • Configure AdcMSBDoubleClkEnable as Disabled to HwUnit1.	

1.2.5.2 Member: GrpArbitCfg

Table 74 GrpArbitCfg

Name	GrpArbitCfg	
Type	uint32	
Description	Indicates the arbitration configuration value of HW unit <x>.('x' = Hw Unit ID starting from 0 to Max HW Units available in the derivative)	
Verification method	The structure member is generated as a value of arbitration configuration for GxARBCFG register. Bits 0-1 generate 0 when AdcSyncConvEnable is enabled and AdcSyncConvMode is configured with ADC_SYNC_SLAVE and generates 3 when AdcSyncConvMode is ADC_SYNC_MASTER or ADC_STAND_ALONE. Other bits are always generated as 0.	
Example(s)	**Action**	**Generated output**
	Configure AdcSyncConvMode as ADC_STAND_ALONE to HwUnit0.	`0x00000003U /*Configuration value for G0ARBCFG register*/`
	Configure AdcSyncConvMode as ADC_SYNC_MASTER to HwUnit1.	`0x00000003U /*Configuration value for G1ARBCFG register*/`
	Configure AdcSyncConvMode as ADC_SYNC_SLAVE to	`0x00000000U /*Configuration value for G2ARBCFG register*/`

	HwUnit2.	

1.2.5.3 Member: GrpArbitPrioCfg

Table 75 GrpArbitPrioCfg

Name	GrpArbitPrioCfg	
Type	uint32	
Description	Indicates the arbitration priority configuration value of HW unit <x>.('x' = Hw Unit ID starting from 0 to Max HW Units available in the derivative)	
Verification method	The structure member is generated as a value of arbitration priority configuration for GxARBPR register. AdcPriorityImplementation is configured as ADC_PRIORITY_NONE: • Bit 24 always generates 1. AdcPriorityImplementation is configured as ADC_PRIORITY_HW or ADC_PRIORITY_HW_SW: • Bits 0-1 always generate 0. • Bit 3 generates the value configured in AdcRequestSource0ConvMode. • Bits 4-5 always generate 1. • Bit 7 generates the value configured in AdcRequestSource1ConvMode. • Bits 8-9 always generate 2. • Bit 11 generates the value configured in AdcRequestSource2ConvMode. • Bits 24-26 always generate 7. Other bits are always generated as 0.	
Example(s)	**Action**	**Generated output**
	• Configure AdcPriorityImplementation as ADC_PRIORITY_NONE.	`0x01000000U /*Configuration value for G0ARBPR register*/`
	• Configure AdcPriorityImplementation as ADC_PRIORITY_HW_SW. • Configure AdcRequestSource0ConvMode, AdcRequestSource1ConvMode and AdcRequestSource2ConvMode with ADC_WAIT_FOR_START_MODE.	`0x07000210U /*Configuration value for G0ARBPR register*/`
	• Configure AdcPriorityImplementation as ADC_PRIORITY_HW. • Configure AdcRequestSource0ConvMode, AdcRequestSource1ConvMode and AdcRequestSource2ConvMode with ADC_CANCEL_INJECT_REPEAT_M	`0x07000a98U /*Configuration value for G0ARBPR register*/`

	ODE.	

1.2.5.4 Member: KernelInputClass0Cfg

Table 76 KernelInputClass0Cfg

Name	KernelInputClass0Cfg	
Type	uint32	
Description	Indicates the kernel input class 0 configuration value of HW unit <x>.('x' = Hw Unit ID starting from 0 to Max HW Units available in the derivative)	
Verification method	The structure member is generated as a value of kernel input class 0 configurations for GxICLASS0 register. Bits 0-4 store value configured in AdcChSampleTime. Bits 6-7 store value configured in AdcChPreChargeClkCycles. Bits 8-9 store value configured in AdcChConvMode. Bit 10 stores value configured in AdcChSESPSEnable. Bits 16-20 store value configured in AdcEmuxChSampleTime when AdcEmuxEnable parameter is enabled. Bits 22-23 store value configured in AdcEmuxChPreChargeClkCycles when AdcEmuxEnable parameter is enabled. Bits 24-25 store value configured in AdcEmuxChConvMode when AdcEmuxEnable parameter is enabled. Bit 26 stores value configured in AdcEmuxChSESPSEnable when AdcEmuxEnable parameter is enabled. Other bits are always generated as 0.	
Example(s)	**Action**	**Generated output**
	• Configure AdcChSampleTime with 0. • Configure AdcChPreChargeClkCycles with ADC_INPUT_PRECHARGE_CYCLES_16. • Configure AdcChConvMode with ADC_NOISE_REDUCTION_STEPS_0. • Configure AdcChSESPSEnable with 0. • Disable parameter AdcEmuxEnable	`0x00000080U /*Configuration value for GxICLASS0 register */`
	• Configure AdcChSampleTime with 10. • Configure AdcChPreChargeClkCycles with ADC_INPUT_PRECHARGE_CYCLES_8. • Configure AdcChConvMode with ADC_NOISE_REDUCTION_STEPS_	`0x0080054aU, /*Configuration value for GxICLASS0 register */`

	1. • Configure AdcChSESPSEnable with 1. • Enable parameter AdcEmuxEnable • Configure AdcEmuxChSampleTime with 0. • Configure AdcEmuxChPreChargeClkCycles with ADC_INPUT_PRECHARGE_CYCLES_16. • Configure AdcEmuxChConvMode with ADC_NOISE_REDUCTION_STEPS_0. • Configure AdcEmuxChSESPSEnable with 0.	

1.2.5.5 Member: KernelInputClass1Cfg

Table 77 KernelInputClass1Cfg

Name	KernelInputClass1Cfg	
Type	uint32	
Description	Indicates the kernel input class 1 configuration value of HW unit <x>.('x' = Hw Unit ID starting from 0 to Max HW Units available in the derivative)	
Verification method	The structure member is generated as a value of kernel input class 1 configurations for GxICLASS1 register. Bits 0-4 store value configured in AdcChSampleTime. Bits 6-7 store value configured in AdcChPreChargeClkCycles. Bits 8-9 store value configured in AdcChConvMode. Bit 10 stores value configured in AdcChSESPSEnable. Bits 16-20 store value configured in AdcEmuxChSampleTime when AdcEmuxEnable parameter is enabled. Bits 22-23 store value configured in AdcEmuxChPreChargeClkCycles when AdcEmuxEnable parameter is enabled. Bits 24-25 store value configured in AdcEmuxChConvMode when AdcEmuxEnable parameter is enabled. Bit 26 stores value configured in AdcEmuxChSESPSEnable when AdcEmuxEnable parameter is enabled. Other bits are always generated as 0.	
Example(s)	**Action**	**Generated output**
	• Configure AdcChSampleTime with 0. • Configure AdcChPreChargeClkCycles with ADC_INPUT_PRECHARGE_CYCLES	`0x00000080U, /*Configuration value for GxICLASS1 register */`

_16. • Configure AdcChConvMode with ADC_NOISE_REDUCTION_STEPS_0. • Configure AdcChSESPSEnable with 0. • Disable parameter AdcEmuxEnable	
• Configure AdcChSampleTime with 10. • Configure AdcChPreChargeClkCycles with ADC_INPUT_PRECHARGE_CYCLES_8. • Configure AdcChConvMode with ADC_NOISE_REDUCTION_STEPS_1. • Configure AdcChSESPSEnable with 1. • Enable parameter AdcEmuxEnable • Configure AdcEmuxChSampleTime with 0. • Configure AdcEmuxChPreChargeClkCycles with ADC_INPUT_PRECHARGE_CYCLES_16. • Configure AdcEmuxChConvMode with ADC_NOISE_REDUCTION_STEPS_0. • Configure AdcEmuxChSESPSEnable with 0.	`0x0080054aU, /*Configuration value for GxICLASS1 register */`

1.2.5.6 Member: GrpSyncCtrlCfg

Table 78 GrpSyncCtrlCfg

Name	GrpSyncCtrlCfg
Type	uint32
Description	Indicates the synchronization control configuration value of HW unit <x>.('x' = Hw Unit ID starting from 0 to Max HW Units available in the derivative)
Verification method	The structure member is generated as a value of synchronization control configurations for GxSYNCTR register. AdcSyncConvMode is ADC_STAND_ALONE: • Bits 0-1 and 4-6 always generate 0.

	AdcSyncConvMode is ADC_SYNC_MASTER or ADC_SYNC_SLAVE: • Bits 0-1 and 4-6 store value based on configured value in AdcSyncConvMode as Master or Slave. Other bits are always generated as 0.	
Example(s)	**Action**	**Generated output**
	Configure AdcSyncConvMode as ADC_SYNC_MASTER to HwUnit0.	`0x00000030U /*Configuration value for G0SYNCTR register*/`
	Configure AdcSyncConvMode as ADC_SYNC_SLAVE to HwUnit1.	`0x00000031U /*Configuration value for G1SYNCTR register*/`
	Configure AdcSyncConvMode as ADC_STAND_ALONE to HwUnit2.	`0x00000000U /*Configuration value for G2SYNCTR register*/`

1.2.6 Structure: Adc_kHwUnit[x]Ch_Config[_<variant>][y]

Table 79 Adc_kHwUnit[x]Hw_Config[_<variant>]

Name	Adc_kHwUnit[x]Ch_Config[_<variant>][y]	
Type	Adc_ChannelCfgType	
Description	Configuration structure of ADC driver for an array of channel specific configuration parameter which will be referenced in HW unit specific configuration structure. ('x' = Hw Unit ID starting from 0 to Max HW Units available in the derivative. 'y'= Channel count ranging from 0 to Max Channels available in the Hw derivative).	
Verification method	The generated structure member is present in the Adc_kHwUnit<x>_Config[_<variant>] structure. For a variant-aware configuration, Member name is appended with the <variant> name. For variant-unaware configuration <variant> is ignored.	
Example(s)	**Action**	**Generated output**

	Configure 2 channels to HwUnit 1.
	(variant Petrol)
		`static const Adc_ChannelCfgType Adc_kHwUnit1Ch_Config_Petrol[2]=`
		`{`
		`  {`
		`  0x00000000U, /*Configuration value for the G1CHCTR0 register*/`
		`  0x00000000U, /*Configuration value for the G1BOUND register*/`
		`  0U, /*Analog Channel number for the corresponding Logical Channel*/`
		`  0U /*Limit Check channel or not */`
		`  },`

		{
		0x00000000U, /*Configuration value for the G1CHCTR0 register*/
		0x00000000U, /*Configuration value for the G1BOUND register*/
		0U, /*Analog Channel number for the corresponding Logical Channel*/
		0U /*Limit Check channel or not */
		}
		};

	Configure 3 channels to HwUnit 2.
	(variant unaware)
		static const Adc_ChannelCfgType Adc_kHwUnit2Ch_Config[3]=
		{
		{
		0x00000400U, /*Configuration value for the G2CHCTR0 register*/
		0x00000000U, /*Configuration value for the G2BOUND register*/
		0U, /*Analog Channel number for the corresponding Logical Channel*/
		0U /*Limit Check channel or not */
		},
		{
		0x00000000U, /*Configuration value for the G2CHCTR0 register*/
		0x00000000U, /*Configuration value for the G2BOUND register*/
		0U, /*Analog Channel number for the corresponding Logical Channel*/
		0U /*Limit Check channel or not */
		},
		{
		0x00000000U, /*Configuration value for the G2CHCTR0 register*/
		0x00000000U, /*Configuration value for the G2BOUND register*/
		0U, /*Analog Channel number for the corresponding Logical Channel*/
		0U /*Limit Check channel or not */
		}
		};

1.2.6.1 Member: ChannelChctrCfg

Table 80 ChannelChctrCfg

Name	ChannelChctrCfg	
Type	uint32	
Description	Indicates the channel control configuration value of channel <y> of HW unit <x> .('x' = Hw Unit ID starting from 0 to Max HW Units available in the derivative. 'y'= Channel count ranging from 0 to Max Channels available in the Hw derivative).	
Verification method	The structure member is generated as a value of channel control configuration for GxCHCTR register. Bits 0-1 store the value configured in AdcInputClassSelection. Bits 4-5 always generate 0. Bits 6-7 store the value configured in AdcChannelLimitCheck. Bits 8-9 generate the value configured based on ChannelRangeSelect. Bit 10 stores the value configured in AdcSyncConvChannelEnable. Bit 11 stores the value configured in AdcChannelRefVoltsrcHigh. Bit 21 stores the value configured in AdcResultAlignment. Bits 28-29 store the value configured in AdcBWDPrechargeLevel. Bit 30 stores the value configured in AdcBWDEnable. Other bits are always generated as 0.	
Example(s)	**Action**	**Generated output**
	• Configure AdcInputClassSelection as ADC_HWUNIT_CLASS_0. • Configure AdcChannelLimitCheck as Disabled. • Configure AdcSyncConvChannelEnable as Disabled. • Configure AdcChannelRefVoltsrcHigh as ADC_USES_VREF. • Configure AdcResultAlignment as ADC_ALIGN_RIGHT. • Configure AdcBWDEnable as Disabled.	`0x00000000U, /*Configuration value for the G0CHCTR0 register*/`
	• Configure AdcInputClassSelection as ADC_GLOBAL_CLASS_1. • Configure AdcChannelLimitCheck as Disabled. • Configure AdcSyncConvChannelEnable	`0x50200403U, /*Configuration value for the G0CHCTR0 register*/`

	as Enabled. • Configure AdcChannelRefVoltsrcHigh as ADC_USES_VREF. • Configure AdcResultAlignment as ADC_ALIGN_LEFT. • Configure AdcBWDEnable as Enabled. • Configure AdcBWDPrechargeLevel as ADC_BWD_PRECH_VAGND.	
	• Configure AdcInputClassSelection as ADC_HWUNIT_CLASS_1. • Configure AdcChannelLimitCheck as Enabled. • Configure AdcSyncConvChannelEnable as Disabled. • Configure AdcChannelRangeSelect as ADC_RANGE_ALWAYS. • Configure AdcChannelRefVoltsrcHigh as ADC_USES_VREF. • Configure AdcResultAlignment as ADC_ALIGN_LEFT. • Configure AdcBWDEnable as Enabled. • Configure AdcBWDPrechargeLevel as ADC_BWD_PRECH_VAREF.	`0x40200341U, /*Configuration value for the G1CHCTR0 register*/`

1.2.6.2 Member: BoundaryValues

Table 81 BoundaryValues

Name	BoundaryValues
Type	uint32
Description	Indicates the boundary configuration value of channel <y> of HW unit <x> .('x' = Hw Unit ID starting from 0 to Max HW Units available in the derivative. 'y'= Channel count ranging from 0 to Max Channels available in the Hw derivative).
Verification	The structure member is generated as a value of boundary configuration for GxBOUND register.

method	Bits 0-11 and 16-27 always generate 0 when AdcChannelLimitCheck is Disabled. Bits 0-11 store the value configured in AdcChannelLowLimit when AdcChannelLimitCheck is Enabled. Bits 16-27 store the value configured in AdcChannelHighLimit when AdcChannelLimitCheck is Enabled. Other bits are always generated as 0.	
Example(s)	**Action**	**Generated output**
	• Configure AdcChannelLimitCheck as Disabled.	`0x00000000U /*Configuration value for the G0BOUND register*/`
	• Configure AdcChannelLimitCheck as Enabled. • Configure AdcChannelRangeSelect as ADC_RANGE_ALWAYS. • Configure AdcChannelLowLimit as 0. • Configure AdcChannelHighLimit as 4095.	`0x00000000U /*Configuration value for the G1BOUND register*/`
	• Configure AdcChannelLimitCheck as Enabled. • Configure AdcChannelRangeSelect as ADC_RANGE_BETWEEN. • Configure AdcChannelLowLimit as 1000. • Configure AdcChannelHighLimit as 4000.	`0x0fa003e9U /*Configuration value for the G1BOUND register*/`

1.2.6.3 Member: AnChannelNo

Table 82 AnChannelNo

Name	AnChannelNo	
Type	Adc_ChannelType	
Description	Indicates the analog channel number configuration value of channel <y> of HW unit <x> .('x' = Hw Unit ID starting from 0 to Max HW Units available in the derivative. 'y'= Channel count ranging from 0 to Max Channels available in the Hw derivative).	
Verification method	The structure member is generated as a value of analog channel number for the corresponding logical channel.	
Example(s)	**Action**	**Generated output**

	Configure AdcAnChannelNum as G1CH0.	`0U, /*Analog Channel number for the corresponding Logical Channel*/`
	Configure AdcAnChannelNum as G1CH3.	`3U, /*Analog Channel number for the corresponding Logical Channel*/`

1.2.6.4 Member: LimitCheckEnabled

Table 83 LimitCheckEnabled

Name	LimitCheckEnabled	
Type	uint8	
Description	Indicates the limit check configuration value of channel <y> of HW unit <x> .('x' = Hw Unit ID starting from 0 to Max HW Units available in the derivative. 'y'= Channel count ranging from 0 to Max Channels available in the Hw derivative).	
Verification method	The structure member is generated as a value of limit check for the corresponding logical channel. *Note: This parameter is user configurable only when 'AdcGeneral/ AdcEnableLimitCheck' is enabled.*	
Example(s)	**Action**	**Generated output**
	Configure AdcChannelLimitCheck as Enabled.	`1U /*Limit Check channel or not */`
	Configure AdcChannelLimitCheck as Disabled.	`0U /*Limit Check channel or not */`

1.2.7 Structure: Adc_kHwUnit[x]Grp_Config[_<variant>][y]

Table 84 Adc_kHwUnit[x]Grp_Config[_<variant>]][y]

Name	Adc_kHwUnit[x]Grp_Config[_<variant>][y]	
Type	Adc_GroupCfgType	
Description	Configuration structure of ADC driver for an array of group specific configuration parameter. Group specific configuration is common for all the channels belonging to the group. ('x' = Hw Unit ID starting from 0 to Max HW Units available in the derivative. 'y'= Group count ranging from 0 to 31).	
Verification method	The generated structure member is present in the Adc_kHwUnit<x>_Config[_<variant>] structure. For a variant-aware configuration, Member name is appended with the <variant> name. For variant-unaware configuration <variant> is ignored.	
Example(s)	**Action**	**Generated output**

	Configure 2 Groups to HwUnit 1.
	(variant Petrol)
		`static const Adc_GroupCfgType Adc_kHwUnit1Grp_Config_Petrol[2]=`
		`{`
		`  {/*Group Configuration structure for`

```
Adc1Group_0 - ID32*/
    /*
      Group Properties:
      Trigger Source: ADC_TRIGG_SRC_SW
      Trigger Edge:
      HW Trigger Source: ADC_TRIG_NONE
      HW Gate Source: ADC_GATE_NONE
      Gate Level: ADC_GATE_LVL_HIGH
    */
    /* Notification Function Address */
    (Adc_NotifyFnPtrType)0U,
    /*Address for Group Definition
Structure*/

&Adc_kHwUnit1GrpAdc1Group_0_Config[0U],
    /*Address for the GTM trigger
configuration structure*/
    (const
Mcu_17_Gtm_TomAtomChConfigType *)0U,
    /*Address for the GTM gate
configuration structure*/
    (const
Mcu_17_Gtm_TomAtomChConfigType *)0U,
    /*Address for the ERU trigger
configuration structure*/
    (const Adc_EruChannelCfgType *)0U,
    /*Address for the ERU gate
configuration structure*/
    (const Adc_EruChannelCfgType *)0U,
    /*Configuration value for the
G1QCTRL register*/
    0x00000000U,
    /*Configuration value for the G1QMR
register*/
    0x00000001U,
    /*Configuration value for the
G1ALIAS register*/
    0x00000100U,
    /* Configuration value for G1REQTM
register*/
    0x00000000U,
    /*Bit Mask for all the analog
channels configured for the group*/
```

```
    0x0001U,
    /*Bit Mask for all the result
registers configured for the group*/
    0x0001U,
    /*Bit Mask for all the analog
channels configured for synchronous
conversion*/
    0x0000U,
    /*Bit Mask for all the result
registers configured for synchronous
conversion*/
    0x0000U,
    ADC_TRIGG_SRC_SW,
    ADC_CONV_MODE_ONESHOT,
    ADC_ACCESS_MODE_SINGLE,
    ADC_STREAM_BUFFER_LINEAR,
    1U, /*Number of streaming samples
for the group*/
    ADC_OTHER_HW_USED, /*HW peripheral
used for Trigger*/
    ADC_OTHER_HW_USED, /*HW peripheral
used for Gate*/
    55U, /*Priority Level for the
group*/
    1U, /*Channel Count for the group*/
    0U, /*Limit Check enabled for the
group*/
    7U, /* EMUX configuration of the
Group */
    1U /* Diagnostic channels configured
for the Group */
  },


  {/*Group Configuration structure for
AdcGroup_32 - ID33*/
    /*
      Group Properties:
      Trigger Source: ADC_TRIGG_SRC_SW
      Trigger Edge:
      HW Trigger Source: ADC_TRIG_NONE
      HW Gate Source: ADC_GATE_NONE
      Gate Level: ADC_GATE_LVL_HIGH
    */
```

```
    /* Notification Function Address */
    (Adc_NotifyFnPtrType)0U,
    /*Address for Group Definition
Structure*/

&Adc_kHwUnit1GrpAdcGroup_32_Config[0U],
    /*Address for the GTM trigger
configuration structure*/
    (const
Mcu_17_Gtm_TomAtomChConfigType *)0U,
    /*Address for the GTM gate
configuration structure*/
    (const
Mcu_17_Gtm_TomAtomChConfigType *)0U,
    /*Address for the ERU trigger
configuration structure*/
    (const Adc_EruChannelCfgType *)0U,
    /*Address for the ERU gate
configuration structure*/
    (const Adc_EruChannelCfgType *)0U,
    /*Configuration value for the
G1QCTRL register*/
    0x00000000U,
    /*Configuration value for the G1QMR
register*/
    0x00000001U,
    /*Configuration value for the
G1ALIAS register*/
    0x00000100U,
    /* Configuration value for G1REQTM
register*/
    0x00000000U,
    /*Bit Mask for all the analog
channels configured for the group*/
    0x0001U,
    /*Bit Mask for all the result
registers configured for the group*/
    0x0001U,
    /*Bit Mask for all the analog
channels configured for synchronous
conversion*/
    0x0000U,
    /*Bit Mask for all the result
registers configured for synchronous
```

		conversion*/
		0x0000U,
		ADC_TRIGG_SRC_SW,
		ADC_CONV_MODE_ONESHOT,
		ADC_ACCESS_MODE_SINGLE,
		ADC_STREAM_BUFFER_LINEAR,
		1U, /*Number of streaming samples for the group*/
		ADC_OTHER_HW_USED, /*HW peripheral used for Trigger*/
		ADC_OTHER_HW_USED, /*HW peripheral used for Gate*/
		0U, /*Priority Level for the group*/
		1U, /*Channel Count for the group*/
		0U, /*Limit Check enabled for the group*/
		3U, /* EMUX configuration of the Group */
		0U /* Diagnostic channels configured for the Group */
		}
		};

	Configure 1 Group to HwUnit 2. (variant unaware)	static const Adc_GroupCfgType Adc_kHwUnit2Grp_Config[1]= { {/*Group Configuration structure for Adc2Group_0 - ID64*/ /* Group Properties: Trigger Source: ADC_TRIGG_SRC_SW Trigger Edge: HW Trigger Source: ADC_TRIG_NONE HW Gate Source: ADC_GATE_NONE Gate Level: ADC_GATE_LVL_HIGH */ /* Notification Function Address */ (Adc_NotifyFnPtrType)0U, /*Address for Group Definition Structure*/ &Adc_kHwUnit2GrpAdc2Group_0_Config[0U], /*Address for the GTM trigger

```
configuration structure*/
    (const
Mcu_17_Gtm_TomAtomChConfigType *)0U,
    /*Address for the GTM gate
configuration structure*/
    (const
Mcu_17_Gtm_TomAtomChConfigType *)0U,
    /*Address for the ERU trigger
configuration structure*/
    (const Adc_EruChannelCfgType *)0U,
    /*Address for the ERU gate
configuration structure*/
    (const Adc_EruChannelCfgType *)0U,
    /*Configuration value for the
G2QCTRL register*/
    0x00000000U,
    /*Configuration value for the G2QMR
register*/
    0x00000001U,
    /*Configuration value for the
G2ALIAS register*/
    0x00000100U,
    /* Configuration value for G2REQTM
register*/
    0x00000000U,
    /*Bit Mask for all the analog
channels configured for the group*/
    0x0001U,
    /*Bit Mask for all the result
registers configured for the group*/
    0x0001U,
    /*Bit Mask for all the analog
channels configured for synchronous
conversion*/
    0x0000U,
    /*Bit Mask for all the result
registers configured for synchronous
conversion*/
    0x0000U,
    ADC_TRIGG_SRC_SW,
    ADC_CONV_MODE_ONESHOT,
    ADC_ACCESS_MODE_SINGLE,
    ADC_STREAM_BUFFER_LINEAR,
    1U, /*Number of streaming samples
```

		for the group*/
		ADC_OTHER_HW_USED, /*HW peripheral used for Trigger*/
		ADC_OTHER_HW_USED, /*HW peripheral used for Gate*/
		0U, /*Priority Level for the group*/
		1U, /*Channel Count for the group*/
		0U, /*Limit Check enabled for the group*/
		7U, /* EMUX configuration of the Group */
		1U /* Diagnostic channels configured for the Group */
		}
		};

1.2.7.1 Member: NotifyPtr

Table 85 NotifyPtr

Name	NotifyPtr	
Type	Adc_NotifyFnPtrType	
Description	Indicates the address of application notification call back for the group <y> of HW unit <x> . ('x' = Hw Unit ID starting from 0 to Max HW Units available in the derivative. 'y'= Group count ranging from 0 to 31).	
Verification method	The structure member is generated as an address of application notification call back for the group. *Note: This parameter is user configurable only when 'AdcGeneral/ AdcGrpNotifCapability' is enabled.*	
Example(s)	**Action**	**Generated output**
	Configure AdcGrpNotifCapability as Disabled	`/* Notification Function Address */ (Adc_NotifyFnPtrType)0U,`
	Configure AdcNotification as IoHwAb_AdcNotification64	`/* Notification Function Address */ IoHwAb_AdcNotification64,`
	Configure AdcNotification as IoHwAb_AdcNotification100	`/* Notification Function Address */ IoHwAb_AdcNotification100,`

1.2.7.2 Member: GroupDefinition

Table 86 GroupDefinition

Name	GroupDefinition

Type	Adc_GroupDefType*	
Description	Indicates the array of structure containing the group definition. Each element of the structures array defines the analog channel and result register configuration.	
Verification method	The generated structure member is present in the Adc_kHwUnit[x]Grp_Config[_<variant>][y] structure. For a variant-aware configuration, Member name is appended with the variant name. For variant-unaware configuration <variant> is ignored.	
Example(s)	**Action**	**Generated output**
	Configure Adc0Group_0 to HwUnit0 (variant unaware)	`/*Address for Group Definition Structure*/` `&Adc_kHwUnit0GrpAdc0Group_0_Config[0U]`
	Configure Adc1Group_0 to HwUnit1 (variant Petrol)	`/*Address for Group Definition Structure*/` `&Adc_kHwUnit1GrpAdc1Group_0_Config_Petrol[0U],`

1.2.7.3 Member: GtmTrigCfg

Table 87 GtmTrigCfg

Name	GtmTrigCfg	
Type	Mcu_17_Gtm_TomAtomChConfigType*	
Description	Indicates the GTM (ATOM / TOM) trigger configuration structure.	
Verification method	The generated structure member is present in the Adc_kHwUnit[x]Grp_Config[_<variant>][y] structure. For a variant-aware configuration, Member name is appended with the variant name. For variant-unaware configuration <variant> is ignored.	
Example(s)	**Action**	**Generated output**
	Configure Adc0Group_5 to HwUnit0 with GTM as trigger source. (variant unaware)	`/*Address for the GTM trigger configuration structure*/` `(const Mcu_17_Gtm_TomAtomChConfigType *)&Adc_kHwUnit0GrpAdc0Group_5GtmTrig_Config`
	Configure Adc11Group_2 to HwUnit11 with GTM as trigger source. (variant Petrol)	`/*Address for the GTM trigger configuration structure*/` `(const Mcu_17_Gtm_TomAtomChConfigType *)&Adc_kHwUnit11GrpAdc11Group_2GtmTrig_Config_Petrol`

1.2.7.4 Member: GtmGateCfg

Table 88 GtmGateCfg

Name	GtmGateCfg
Type	Mcu_17_Gtm_TomAtomChConfigType*
Description	Indicates the GTM (ATOM / TOM) gating configuration structure.
Verification method	The generated structure member is present in the Adc_kHwUnit[x]Grp_Config[_<variant>][y] structure. For a variant-aware configuration, Member name is appended with the variant name. For variant-unaware configuration <variant> is ignored.

Example(s)	Action	Generated output
	Configure Adc2Group_0 to HwUnit2 with GTM as gate source. (variant unaware)	`/*Address for the GTM gate configuration structure*/` `(const Mcu_17_Gtm_TomAtomChConfigType *)&Adc_kHwUnit2GrpAdc2Group_0GtmGate_Config`
	Configure Adc8Group_2 to HwUnit8 with GTM as gate source. (variant Petrol)	`/*Address for the GTM gate configuration structure*/` `(const Mcu_17_Gtm_TomAtomChConfigType *)&Adc_kHwUnit8GrpAdc8Group_2GtmGate_Config_Petrol`

1.2.7.5 Member: EruTrigCfg

Table 89 EruTrigCfg

Name	EruTrigCfg	
Type	Adc_EruChannelCfgType *	
Description	Indicates the ERU trigger configuration structure.	
Verification method	The generated structure member is present in the Adc_kHwUnit[x]Grp_Config[_<variant>][y] structure. For a variant-aware configuration, Member name is appended with the variant name. For variant-unaware configuration <variant> is ignored.	
Example(s)	**Action**	**Generated output**
	Configure Adc0Group_9 to HwUnit0 with ERU as trigger source. (variant unaware)	`/*Address for the ERU trigger configuration structure*/` `(const Adc_EruChannelCfgType *)&Adc_kHwUnit0GrpAdc0Group_9EruTrig_Config`
	Configure Adc8Group_2 to HwUnit8 with ERU as trigger source. (variant Petrol)	`/*Address for the ERU trigger configuration structure*/` `(const Adc_EruChannelCfgType *)&Adc_kHwUnit8GrpAdc8Group_2EruTrig_Config_Petrol`

1.2.7.6 Member: EruGateCfg

Table 90 EruGateCfg

Name	EruGateCfg	
Type	Adc_EruChannelCfgType *	
Description	Indicates the ERU gating configuration structure.	
Verification method	The generated structure member is present in the Adc_kHwUnit[x]Grp_Config[_<variant>][y] structure. For a variant-aware configuration, Member name is appended with the variant name. For variant-unaware configuration <variant> is ignored.	
Example(s)	**Action**	**Generated output**
	Configure Adc0Group_13 to	`*Address for the ERU gate configuration`

	HwUnit0 with ERU as gate source. (variant unaware)	`structure*/` `(const Adc_EruChannelCfgType *)&Adc_kHwUnit0GrpAdc0Group_13EruGate_Config`
	Configure Adc8Group_2 to HwUnit8 with ERU as gate source. (variant Petrol)	`*Address for the ERU gate configuration structure*/` `(const Adc_EruChannelCfgType *)&Adc_kHwUnit8GrpAdc8Group_2EruGate_Config_Petrol`

1.2.7.7 Member: GroupQCtrlCfg

Table 91 GroupQCtrlCfg

Name	GroupQCtrlCfg	
Type	uint32	
Description	Indicates the value of queue source control register that selects the external gate and /or trigger signal.	
Verification method	The structure member is generated as a value of queue source control configuration for GxQCTRLy register. Bits 8-11 generate the value based on HW configured in AdcHwExtTrigSelect. Bits 13-14 generate the value based on HW and HW signal configured in AdcHwExtTrigSelect and AdcHwTrigSignal. Bit 28 generates 1 when value configured in AdcHwTrigTimer is not equal to 0. Other bits are always generated as 0.	
Example(s)	**Action**	**Generated output**
	• Configure AdcGroupTriggSrc as ADC_TRIGG_SRC_HW. • Configure AdcHwExtTrigSelect as ADC_TRIG_8_GxREQTRI_GTM_ADCx_TRIG0. • Configure AdcHwExtGateSelect as ADC_GATE_12_GxREQGTM_ERUPDOUTx. • Configure AdcHwTrigTimer as 0. • Configure AdcHwTrigSignal as ADC_HW_TRIG_RISING_EDGE.	`/*Configuration value for the G0QCTRL register*/` `0x000c4800U`
	• Configure AdcGroupTriggSrc as ADC_TRIGG_SRC_HW. • Configure AdcHwExtTrigSelect as ADC_TRIG_NONE. • Configure AdcHwExtGateSelect as ADC_GATE_NONE. • Configure AdcHwTrigTimer as 1000. • Configure AdcHwTrigSignal as ADC_HW_TRIG_RISING_EDGE.	`/*Configuration value for the G0QCTRL register*/` `0x10000000U`

1.2.7.8 Member: GroupQModeCfg

Table 92 GroupQModeCfg

Name	GroupQModeCfg	
Type	uint32	
Description	Indicates the value of queue mode register that selects the operating mode of a queued request source.	
Verification method	The structure member is generated as a value of queue mode configuration for GxQMRy register. Bits 0-1 store the value 1 when AdcGroupTriggSrc is ADC_TRIGG_SRC_SW otherwise generate the values from 0 to 3 based on value configured in AdcHwExtTrigSelect, AdcHwExtGateSelect and AdcHwGateSignal. Bit 2 stores the value 1 when AdcHwExtTrigSelect is not equal to ADC_TRIG_NONE or AdcHwTrigTimer is not equal to 0 otherwise stores the value 0. Other bits are always generated as 0.	
Example(s)	**Action**	**Generated output**
	• Configure AdcGroupTriggSrc as ADC_TRIGG_SRC_SW.	`/*Configuration value for the G0QMR register*/` `0x00000001U`
	• Configure AdcGroupTriggSrc as ADC_TRIGG_SRC_HW. • Configure AdcHwExtTrigSelect as ADC_TRIG_NONE or AdcHwTrigTimer as 100. • Configure AdcHwExtGateSelect as ADC_GATE_NONE.	`/*Configuration value for the G0QMR register*/` `0x00000005U`
	• Configure AdcGroupTriggSrc as ADC_TRIGG_SRC_HW. • Configure AdcHwExtTrigSelect as ADC_TRIG_8_GxREQTRI_GTM_ADCx_TRIG0 or AdcHwTrigTimer as 100 • Configure AdcHwExtGateSelect as ADC_GATE_12_GxREQGTM_ERUPDOUTx.	`/*Configuration value for the G0QMR register*/` `0x00000006U`
	• Configure AdcGroupTriggSrc as ADC_TRIGG_SRC_HW. • Configure AdcHwExtTrigSelect as ADC_TRIG_15_GxREQTRP_GxREQGTySEL. • Configure AdcHwExtGateSelect as ADC_GATE_2_GxREQGTC_CCU6061_TRIG0. • Configure AdcHwGateSignal as ADC_GATE_LVL_HIGH.	`/*Configuration value for the G0QMR register*/` `0x00000005U`
	• Configure AdcGroupTriggSrc as ADC_TRIGG_SRC_HW.	`/*Configuration value for the G0QMR register*/`

• Configure AdcHwExtTrigSelect as ADC_TRIG_1_GxREQTRB_CCU61_SR3 or AdcHwTrigTimer as 0 • Configure AdcHwExtGateSelect as ADC_GATE_0_GxREQGTA_GTM_ADCx_TRIG0. • Configure AdcHwGateSignal as ADC_GATE_LVL_HIGH.	`0x00000006U`
• Configure AdcGroupTriggSrc as ADC_TRIGG_SRC_HW. • Configure AdcHwExtTrigSelect as ADC_TRIG_1_GxREQTRB_CCU61_SR3 or AdcHwTrigTimer as 0 • Configure AdcHwExtGateSelect as ADC_GATE_0_GxREQGTA_GTM_ADCx_TRIG0. • Configure AdcHwGateSignal as ADC_GATE_LVL_LOW.	`/*Configuration value for the G0QMR register*/` `0x00000007U`

1.2.7.9 Member: AliasChCfg

Table 93 AliasChCfg

Name	AliasChCfg	
Type	uint32	
Description	Indicates the value of alias register that replaces the channel numbers of channels CH0 and CH1 with another channel number.	
Verification method	The structure member is generated as a value of alias configuration for GxALIAS register. Bits 0-4 store the alias channel configured in AdcChannel0Alias. Bits 8-12 store the alias channel configured in AdcChannel1Alias. Other bits are always generated as 0.	
Example(s)	**Action**	**Generated output**
	• Configure AdcChannel0Alias as 0. • Configure AdcChannel1Alias as 1.	`/*Configuration value for the G0ALIAS register*/` `0x00000100U`
	• Configure AdcChannel0Alias as 4. • Configure AdcChannel1Alias as 5.	`/*Configuration value for the G0ALIAS register*/` `0x00000504U`

1.2.7.10 Member: GrpReqTmCfg

Table 94 GrpReqTmCfg

Name	GrpReqTmCfg

Type	uint32	
Description	Indicates the value of request timer register that configures the operating mode of a source-specific request timer.	
Verification method	The structure member is generated as a value of request timer configuration for GxREQTMi register. Bits 0-1 always generate 3 when AdcGroupTriggSrc is ADC_TRIGG_SRC_HW and AdcHwTrigTimer is not equal to 0. Bits 6-15 store the value configured in AdcHwTrigTimer. Bits 22-31 store the value configured in AdcHwTrigTimer. Other bits are always generated as 0.	
Example(s)	**Action**	**Generated output**
	• Configure AdcGroupTriggSrc as ADC_TRIGG_SRC_HW. • Configure AdcHwTrigTimer as 100.	`/* Configuration value for G0REQTM register*/` `0x19001903U`
	• Configure AdcGroupTriggSrc as ADC_TRIGG_SRC_HW. • Configure AdcHwTrigTimer as 1000.	`/* Configuration value for G0REQTM register*/` `0xfa00fa03U`

1.2.7.11 Member: ChannelMask

Table 95 ChannelMask

Name	ChannelMask	
Type	uint16	
Description	Indicates the mask value for channels configured for the group. Each bit represents the corresponding analog channel.	
Verification method	The structure member is generated as a mask value for the analog channels configured for the group.	
Example(s)	**Action**	**Generated output**
	Configure AdcGroupDefinition with 7 channels from channel 0 to channel 6.	`/*Bit Mask for all the analog channels configured for the group*/` `0x007fU`
	Configure AdcGroupDefinition with 4 channels from channel 4 to channel 7.	`/*Bit Mask for all the analog channels configured for the group*/` `0x00f0U`

1.2.7.12 Member: ResultRegMask

Table 96 ResultRegMask

Name	ResultRegMask
Type	uint16
Description	Indicates the mask value for result register configured for the group. Each bit represents the corresponding analog channel.

Verification method	The structure member is generated as a mask value for result register configured for the group.	
Example(s)	**Action**	**Generated output**
	Configure AdcResRegDefinition with 7 channels from channel 0 to channel 6.	`/*Bit Mask for all the result registers configured for the group*/` `0x007fU`
	Configure AdcResRegDefinition with 4 channels from channel 4 to channel 7.	`/*Bit Mask for all the result registers configured for the group*/` `0x00f0U`

1.2.7.13 Member: SyncChannelMask

Table 97 SyncChannelMask

Name	SyncChannelMask	
Type	uint16	
Description	Indicates the mask value for sync channels configured for the group. Each bit represents the corresponding analog channel.	
Verification method	The structure member is generated as a mask value for sync channels configured for the group.	
Example(s)	**Action**	**Generated output**
	• Configure AdcSyncConvChannelEnable as Enabled for 4 channels from channel 0 to channel 3 when AdcSyncConvEnable is enabled and AdcSyncConvMode is ADC_SYNC_MASTER. • Configure AdcGroupDefinition with 4 channels from channel 0 to channel 3.	`/*Bit Mask for all the analog channels configured for synchronous conversion*/` `0x000FU`
	• Configure AdcSyncConvChannelEnable as Enabled for 4 channels from channel 0 to channel 3 when AdcSyncConvEnable is enabled and AdcSyncConvMode is ADC_SYNC_MASTER. • Configure AdcGroupDefinition with 2 channels from channel 0 to channel 1.	`/*Bit Mask for all the analog channels configured for synchronous conversion*/` `0x0003U`

1.2.7.14 Member: SyncResultRegMask

Table 98 SyncResultRegMask

Name	SyncResultRegMask	
Type	uint16	
Description	Indicates the mask value for sync result register configured for the group. Each bit represents the corresponding analog channel.	
Verification method	The structure member is generated as a mask value for sync result register configured for the group.	
Example(s)	**Action**	**Generated output**
	• Configure AdcSyncConvChannelEnable as Enabled for 4 channels from channel 0 to channel 3 when AdcSyncConvEnable is enabled and AdcSyncConvMode is ADC_SYNC_MASTER. • Configure AdcResRegDefinition with 4 channels from channel 0 to channel 3.	`/*Bit Mask for all the result registers configured for synchronous conversion*/` `0x000FU`
	• Configure AdcSyncConvChannelEnable as Enabled for 4 channels from channel 0 to channel 3 when AdcSyncConvEnable is enabled and AdcSyncConvMode is ADC_SYNC_MASTER. • Configure AdcResRegDefinition with 2 channels from channel 0 to channel 1.	`/*Bit Mask for all the result registers configured for synchronous conversion*/` `0x0003U`

1.2.7.15 Member: TriggerSource

Table 99 TriggerSource

Name	TriggerSource	
Type	Adc_TriggerSourceType	
Description	Indicates the trigger source (HW / SW) configured for the group.	
Verification method	The structure member is generated as a value of trigger source configured for the group.	
Example(s)	**Action**	**Generated output**
	Configure AdcGroupTriggSrc as	`ADC_TRIGG_SRC_SW`

	ADC_TRIGG_SRC_SW.	
	Configure AdcGroupTriggSrc as ADC_TRIGG_SRC_HW.	ADC_TRIGG_SRC_HW

1.2.7.16 Member: ConvMode

Table 100 ConvMode

Name	ConvMode	
Type	Adc_GroupConvModeType	
Description	Indicates the conversion mode (Continuous / One-Shot) configured for the group.	
Verification method	The structure member is generated as a value of conversion mode configured for the group.	
Example(s)	**Action**	**Generated output**
	Configure AdcGroupConversionMode as ADC_CONV_MODE_CONTINUOUS.	ADC_CONV_MODE_CONTINUOUS
	Configure AdcGroupConversionMode as ADC_CONV_MODE_ONESHOT.	ADC_CONV_MODE_ONESHOT

1.2.7.17 Member: AccessMode

Table 101 AccessMode

Name	AccessMode	
Type	Adc_GroupAccessModeType	
Description	Indicates the access mode (steaming / single) configured for the group.	
Verification method	The structure member is generated as a value of access mode configured for the group.	
Example(s)	**Action**	**Generated output**
	Configure AdcGroupAccessMode as ADC_ACCESS_MODE_STREAMING.	ADC_ACCESS_MODE_STREAMING
	Configure AdcGroupAccessMode as ADC_ACCESS_MODE_SINGLE.	ADC_ACCESS_MODE_SINGLE

1.2.7.18 Member: StreamMode

Table 102 StreamMode

Name	StreamMode
Type	Adc_StreamBufferModeType
Description	Indicates the streaming mode (linear /circular) configured for the group.
Verification method	The structure member is generated as a value of streaming mode configured for the group.

Example(s)	Action	Generated output
	Configure AdcStreamingBufferMode as ADC_STREAM_BUFFER_LINEAR.	ADC_STREAM_BUFFER_LINEAR
	Configure AdcStreamingBufferMode as ADC_STREAM_BUFFER_CIRCULAR.	ADC_STREAM_BUFFER_CIRCULAR

1.2.7.19 Member: NumOfSamples

Table 103 NumOfSamples

Name	NumOfSamples	
Type	Adc_StreamNumSampleType	
Description	Indicates the number of samples for streaming groups.	
Verification method	The structure member is generated as a value of number of samples for streaming groups.	
Example(s)	**Action**	**Generated output**
	Configure AdcStreamingNumSamples as 2.	2U
	Configure AdcStreamingNumSamples as 10.	10U

1.2.7.20 Member: HwTrigType

Table 104 HwTrigType

Name	HwTrigType	
Type	Adc_HwTrigGateType	
Description	Indicates the HW trigger source (GTM / ERU / OTHER) configured for the group.	
Verification method	The structure member is generated as a value of HW trigger source configured for the group.	
Example(s)	**Action**	**Generated output**
	Configure AdcHwExtTrigSelect as ADC_TRIG_8_GxREQTRI_GTM_ADCx_TRIG0 when AdcGroupTriggSrc as ADC_TRIGG_SRC_HW.	ADC_GTM_HW_USED
	Configure AdcHwExtTrigSelect as ADC_TRIG_7_GxREQTRH_ERUIOUTx when AdcGroupTriggSrc as ADC_TRIGG_SRC_HW.	ADC_ERU_HW_USED
	Configure AdcGroupTriggSrc as ADC_TRIGG_SRC_SW.	ADC_OTHER_HW_USED

	Configure AdcHwExtTrigSelect as ADC_TRIG_1_GxREQTRB_CCU61_SR3 when AdcGroupTriggSrc as ADC_TRIGG_SRC_HW.	`ADC_OTHER_HW_USED`

1.2.7.21 Member: HwGateType

Table 105 HwGateType

Name	HwGateType	
Type	Adc_HwTrigGateType	
Description	Indicates the HW gate source (GTM / ERU / OTHER) configured for the group.	
Verification method	The structure member is generated as a value of HW gate source configured for the group.	
Example(s)	**Action**	**Generated output**
	Configure AdcHwExtGateSelect as ADC_TRIG_8_GxREQTRI_GTM_ADCx_TRIG0 when AdcGroupTriggSrc as ADC_TRIGG_SRC_HW.	`ADC_GTM_HW_USED`
	Configure AdcHwExtGateSelect as ADC_TRIG_7_GxREQTRH_ERUIOUTx when AdcGroupTriggSrc as ADC_TRIGG_SRC_HW.	`ADC_ERU_HW_USED`
	Configure AdcGroupTriggSrc as ADC_TRIGG_SRC_SW.	`ADC_OTHER_HW_USED`
	Configure AdcHwExtGateSelect as ADC_TRIG_1_GxREQTRB_CCU61_SR3 when AdcGroupTriggSrc as ADC_TRIGG_SRC_HW.	`ADC_OTHER_HW_USED`

1.2.7.22 Member: GrpPriority

Table 106 GrpPriority

Name	GrpPriority	
Type	uint8	
Description	Indicates the priority level configured for the group.	
Verification method	The structure member is generated as a value of priority level configured for the group. *Note: The member is user configurable only when the configuration parameter AdcPriorityImplementation is not equal to ADC_PRIORITY_NONE.*	
Example(s)	**Action**	**Generated output**
	Configure AdcPriorityImplementation as ADC_PRIORITY_NONE.	`0U, /*Priority Level for the group*/`

	Configure AdcGroupPriority as 20 when AdcPriorityImplementation is configured as ADC_PRIORITY_HW.	`0U, /*Priority Level for the group*/`
	Configure AdcGroupPriority as 254 when AdcPriorityImplementation is configured as ADC_PRIORITY_HW.	`1U /*Priority Level for the group*/`
	Configure AdcGroupPriority as 255 when AdcPriorityImplementation is configured as ADC_PRIORITY_HW.	`2U /*Priority Level for the group*/`
	Configure AdcGroupPriority as 200 when AdcPriorityImplementation is configured as ADC_PRIORITY_HW_SW.	`200U /*Priority Level for the group*/`
	Configure AdcGroupPriority as 55 when AdcPriorityImplementation is configured as ADC_PRIORITY_HW_SW.	`55U /*Priority Level for the group*/`

1.2.7.23 Member: NoOfChannels

Table 107 NoOfChannels

Name	NoOfChannels	
Type	uint8	
Description	Indicates the number of channels configured for the group.	
Verification method	The structure member is generated as a value of number of channels configured for the group.	
Example(s)	**Action**	**Generated output**
	Configure AdcGroupDefinition with 4 channels.	`4U /*Channel Count for the group*/`
	Configure AdcGroupDefinition with 2 channels.	`2U /*Channel Count for the group*/`

1.2.7.24 Member: LimitCheckGroup

Table 108 LimitCheckGroup

Name	LimitCheckGroup

Type	uint8	
Description	Indicates the limit check configured for the group.	
Verification method	The structure member is generated as a value of limit check configured for the group.	
Example(s)	**Action**	**Generated output**
	Configure AdcGroupDefinition with a channel which is enabled with limit check.	`1U /*Limit Check enabled for the group*/`
	Configure AdcGroupDefinition with a channel which is disabled with limit check.	`0U /*Limit Check disabled for the group*/`

1.2.7.25 Member: GrpEmuxCfg

Table 109 GrpEmuxCfg

Name	GrpEmuxCfg	
Type	uint8	
Description	Indicates the EMUX configuration of the Group	
Verification method	The structure member is generated as a value of AdcEmuxStartSelection configured for the group. Bit 0 stores the value configured in the parameter AdcEmuxChGroup. Bits 1-3 store the value configured in the parameter AdcEmuxStartSelection. Other bits are always generated as 0.	
Example(s)	**Action**	**Generated output**
	• Enable AdcEmuxChGroup parameter. • Configure AdcEmuxStartSelection as 3.	`7U /* EMUX configuration of the Group */`
	Disable AdcEmuxChGroup parameter	`0U /* EMUX configuration of the Group */`

1.2.7.26 Member: DiagnosticChGrp

Table 110 DiagnosticChGrp

Name	DiagnosticChGrp	
Type	uint8	
Description	Indicates whether a diagnostic channel is configured for the group.	
Verification method	The structure member is generated as 1 for the group if group is configured with channel, which is enabled with any one of the parameter AdcPullDownDiagnosticEnable or AdcMultiplexerDiagnosticEnable or AdcConverterDiagnosticEnable.	
Example(s)	**Action**	**Generated output**
	Configure AdcGroupDefinition with channels which are enabled with	`1U /* Diagnostic channels configured for the Group */`

any one of the below parameters: • AdcPullDownDiagnosticEnable • AdcMultiplexerDiagnosticEnable • AdcConverterDiagnosticEnable	
Configure AdcGroupDefinition with channels which are disabled with all the below parameters: • AdcPullDownDiagnosticEnable • AdcMultiplexerDiagnosticEnable • AdcConverterDiagnosticEnable	`0U /* Diagnostic channels configured for the Group */`

1.2.8 Structure: Adc_kHwUnit[x]Grp[name]_Config[_<variant>][y]

Table 111 Adc_kHwUnit[x]Grp[name]_Config[_<variant>] [y]

Name	Adc_kHwUnit[x]Grp[name]_Config[_<variant>][y]	
Type	Adc_GroupDefType	
Description	Configuration structure of ADC driver for an array of configured analog channels and result registers. ('x' = Hw Unit ID starting from 0 to Max HW Units available in the derivative. 'y'= Channel count ranging from 0 to Max Channels available in the Hw derivative and 'name'= Name of the group configured to the HW unit).	
Verification method	The generated structure member is present in the Adc_kHwUnit[x]Grp_Config[_<variant>][y] structure. For a variant-aware configuration, Member name is appended with the <variant> name. For variant-unaware configuration <variant> is ignored.	
Example(s)	**Action**	**Generated output**

	Configure AdcGroupDefinition and AdcResRegDefinition with 8 channels from channel 0 to channel 7 to the Group 'Adc0Group_5' of HwUnit0.
	(variant Petrol)
		`/**Group Definition of Adc0Group_5- ID5 of HW Unit 0 */`
		`static const Adc_GroupDefType Adc_kHwUnit0GrpAdc0Group_5_Config_Petrol[8]=`
		`{`
		`/*AS Logical Channel*/ /*Analog Channel*/ /*Result Register*/ /*Channel Diagnostic Data*/`
		`{ 0U, 0U, 0U, 0x00000200U },`
		`{ 1U, 1U, 1U, 0x00000400U },`
		`{ 2U, 2U, 2U, 0x00000800U },`
		`{ 3U, 3U, 3U, 0x00007000U },`
		`{ 4U, 4U, 4U, 0x00000000U },`
		`{ 5U, 5U, 5U, 0x00000000U },`
		`{ 6U, 6U, 6U, 0x00000000U },`
		`{ 7U, 7U, 7U, 0x00000000U }`
		`};`

	Configure AdcGroupDefinition and AdcResRegDefinition with 6 channels from channel 0 to channel 5 to the Group 'Adc1Group_0' of HwUnit1.
	(variant unaware)
		/**Group Definition of Adc1Group_0- ID32 of HW Unit 1 */
		static const Adc_GroupDefType Adc_kHwUnit1GrpAdc1Group_0_Config[6]=
		{
		/*AS Logical Channel*/ /*Analog Channel*/ /*Result Register*/ /*Channel Diagnostic Data*/
		{ 0U, 0U, 0U, 0x00000200U },
		{ 1U, 1U, 1U, 0x00000400U },
		{ 2U, 2U, 2U, 0x00000800U },
		{ 3U, 3U, 3U, 0x00007000U },
		{ 4U, 4U, 4U, 0x00000000U },
		{ 5U, 5U, 5U, 0x00000000U }
		};

1.2.8.1 Member: ASChannelId

Table 112 ASChannelId

Name	ASChannelId	
Type	Adc_ChannelType	
Description	Indicates the index of channel in the Adc Channel array configured to the group.	
Verification method	The structure member is generated as a value of index of the Adc Channel array configured to the group.	
Example(s)	**Action**	**Generated output**
	Configure AdcGroupDefinition with 2 channels from channel 0 to channel 1.	/*AS Logical Channel*/ /*Analog Channel*/ /*Result Register*/ /*Channel Diagnostic Data*/ { 0U, 0U, 0U, 0x00000200U }, { 1U, 1U, 1U, 0x00000400U }
	Configure AdcGroupDefinition with 5 channels from channel 0 to channel 4.	/*AS Logical Channel*/ /*Analog Channel*/ /*Result Register*/ /*Channel Diagnostic Data*/ { 0U, 0U, 0U, 0x00000200U }, { 1U, 1U, 1U, 0x00000400U }, { 2U, 2U, 2U, 0x00000800U }, { 3U, 3U, 3U, 0x00007000U }, { 4U, 4U, 4U, 0x00000000U }

1.2.8.2 Member: AnalogChannelNo

Table 113 AnalogChannelNo

Name	AnalogChannelNo	
Type	Adc_ChannelType	
Description	Indicates the channel number of channels in the Adc Channel array configured to the group.	
Verification method	The structure member is generated as a value of channel number of the Adc Channel array configured to the group.	
Example(s)	**Action**	**Generated output**

	Configure AdcGroupDefinition with 2 channels from channel 0 to channel 1.	`/*AS Logical Channel*/ /*Analog Channel*/ /*Result Register*/ /*Channel Diagnostic Data*/`
		`{ 0U, 0U, 0U, 0x00000200U },`
		`{ 1U, 1U, 1U, 0x00000400U }`

	Configure AdcGroupDefinition with 4 channels from channel 0 to channel 3.	`/*AS Logical Channel*/ /*Analog Channel*/ /*Result Register*/ /*Channel Diagnostic Data*/`
		`{ 0U, 0U, 0U, 0x00000200U },`
		`{ 1U, 1U, 1U, 0x00000400U },`
		`{ 2U, 2U, 2U, 0x00000800U },`
		`{ 3U, 3U, 3U, 0x00007000U }`

1.2.8.3 Member: ResultReg

Table 114 ResultReg

Name	ResultReg	
Type	Adc_ResultRegType	
Description	Indicates the result register for storing the result of channels used in the Adc Channel array configured to the group.	
Verification method	The structure member is generated as a value of result register for storing the result of channels used in the Adc Channel array configured to the group.	
Example(s)	**Action**	**Generated output**

	Configure AdcResRegDefinition with 2 channels from channel 0 to channel 1.	`/*AS Logical Channel*/ /*Analog Channel*/ /*Result Register*/ /*Channel Diagnostic Data*/`
		`{ 0U, 0U, 0U, 0x00000200U },`
		`{ 1U, 1U, 1U, 0x00000400U }`

	Configure AdcResRegDefinition with 4 channels from channel 0 to channel 3.	`/*AS Logical Channel*/ /*Analog Channel*/ /*Result Register*/ /*Channel Diagnostic Data*/`
		`{ 0U, 0U, 0U, 0x00000200U },`
		`{ 1U, 1U, 1U, 0x00000400U },`
		`{ 2U, 2U, 2U, 0x00000800U },`
		`{ 3U, 3U, 3U, 0x00007000U }`

1.2.8.4 Member: AnChDiagnosticsCfg

Table 115 AnChDiagnosticsCfg

Name	AnChDiagnosticsCfg	
Type	uint32	
Description	Indicates the diagnostic value to be stored in the QINR register of channels used in the Adc Channel array configured to the group.	
Verification method	The structure member is generated as a value of diagnostics of the Adc Channel array configured to the group. Bit 9 stores the value configured in the AdcPullDownDiagnosticEnable parameter. Bits 10-11 store the value configured in the AdcMultiplexerDiagnosticLevel parameter, if AdcMultiplexerDiagnosticEnable parameter is true. Bit 12 stores the value configured in the AdcConverterDiagnosticEnable parameter. Bits 13-14 store the value configured in the AdcConverterDiagnosticsLevel parameter, if AdcConverterDiagnosticEnable parameter is true. Other bits are always generated as 0.	
Example(s)	**Action**	**Generated output**

	Configure AdcGroupDefinition with 2 channels from channel 0 to channel 1 with following configurations:
	• Configure channel0 with AdcPullDownDiagnosticEnable as true.
	• Configure channel1 with AdcMultiplexerDiagnosticLevel as ADC_MD_PULL_UP
		`/*AS Logical Channel*/ /*Analog Channel*/ /*Result Register*/ /*Channel Diagnostic Data*/`
		`{ 0U, 0U, 0U, 0x00000200U },`
		`{ 1U, 1U, 1U, 0x00000800U }`

	Configure AdcGroupDefinition with 4 channels from channel 0 to channel 3 with following configurations:
	• Configure channel0 with AdcConverterDiagnosticsLevel as ADC_CD_PULL_DEVICE_VDDM.
	• Configure channel1 with AdcMultiplexerDiagnosticLevel as ADC_MD_PULL_DOWN
	• Don't configure any diagnostics to channel2 and channel3
		`/*AS Logical Channel*/ /*Analog Channel*/ /*Result Register*/ /*Channel Diagnostic Data*/`
		`{ 0U, 0U, 0U, 0x00001000U },`
		`{ 1U, 1U, 1U, 0x00000400U },`
		`{ 2U, 2U, 2U, 0x00000000U },`
		`{ 3U, 3U, 3U, 0x00000000U }`

1.2.9 Structure: Adc_kHwUnit[x]Grp[name]EruTrig_Config[_<variant>]

Table 116 Adc_kHwUnit[x]Grp[name]EruTrig_Config[_<variant>]

Name	Adc_kHwUnit[x]Grp[name]EruTrig_Config[_<variant>]	
Type	Adc_EruChannelCfgType	
Description	Configuration structure of ADC driver for configured ERU trigger. ('x' = HwUnit ID starting from 0 to Max HwUnits available in the derivative and 'name'= Name of the group configured to the HW unit).	
Verification method	The generated structure member is present in the Adc_kHwUnit[x]Grp_Config[_<variant>][y] structure. For a variant-aware configuration, Member name is appended with the <variant> name. For variant-unaware configuration <variant> is ignored.	
Example(s)	**Action**	**Generated output**
	Configure ERU trigger to group 'Adc0Group_5' of HwUnit0. (variant Petrol)	`static const Adc_EruChannelCfgType Adc_kHwUnit0GrpAdc0Group_5EruTrig_Config_Petrol= { 0x3b20U, /*EICR register configuration*/ 0x4000U, /*IGCR register configuration*/ 7U, /*ERS channel*/ 3U /*OGU channel*/ };`
	Configure ERU trigger to group 'Adc1Group_0' of HwUnit1. (variant unaware)	`static const Adc_EruChannelCfgType Adc_kHwUnit1GrpAdc1Group_0EruTrig_Config= { 0x3b20U, /*EICR register configuration*/ 0x4000U, /*IGCR register configuration*/ 7U, /*ERS channel*/ 3U /*OGU channel*/ };`

1.2.9.1 Member: EruEicrCfg

Table 117 EruEicrCfg

Name	EruEicrCfg
Type	uint16
Description	Indicates the value of external input channel configured to the group.
Verification method	The structure member is generated as a value of external input channel configured to the group for EICR register. Bits 4-6 store the suffixed value after _SEL of 'AdcEruErsInputPin'. Bits 8-9 store the value configured in AdcHwTrigSignal. Bits 12-14 store the suffixed value of '/Mcu/Mcu/McuHardwareResourceAllocationConf_0/ McuEruAllocationConf_0/McuEruChannelOutputUnitConf_0' after McuEruChannelOutputUnitConf_. *Note: This parameter is user configurable only when AdcGroupTriggSrc is 'ADC_TRIGG_SRC_HW ' and AdcHwExtTrigSelect is 'ERUIOUT'.*

	Note: ERS channel configured in 'AdcEruErsInputPin' is used to determine the bit position of EICR SFR i.e. Odd Channel starts from bit position is 16, and even channel starts from bit position is 0.

Example(s)	Action	Generated output
	• Configure AdcHwTrigSignal as ADC_HW_TRIG_BOTH_EDGES. • Configure AdcEruOguRef as McuEruChannelOutputUnitConf_0. • Configure AdcEruErsInputPin as ERS_REQ0A_PORTS_P15_4_SEL0.	`0x0b00U /*EICR register configuration*/`
	• Configure AdcHwTrigSignal as ADC_HW_TRIG_RISING_EDGE. • Configure AdcEruOguRef as McuEruChannelOutputUnitConf_0. • Configure AdcEruErsInputPin as ERS_REQ7C_PORTS_P15_1_SEL2.	`0x0a20U /*EICR register configuration*/`

1.2.9.2 Member: EruIgcrCfg

Table 118 EruIgcrCfg

Name	EruIgcrCfg
Type	uint16
Description	Indicates the value of gating control configured to the group.
Verification method	The structure member is generated as a value of gating control configured to the group for IGCR register. Bits 0-13 always generate 0. Bits 14-15 always generate 1. *Note: This parameter is generated only when AdcGroupTriggSrc is 'ADC_TRIGG_SRC_HW ' and AdcHwExtTrigSelect is 'ERUIOUT'.* *Note: ERS channel configured in 'AdcEruErsInputPin' is used to determine the bit position of EICR SFR i.e. Odd Channel starts from bit position is 16, and even channel starts from bit position is 0.*

Example(s)	Action	Generated output
	Generate Adc[_<variant>]_PBcfg.c	`0x4000U /*IGCR register configuration*/`

1.2.9.3 Member: ErsChannel

Table 119 ErsChannel

Name	ErsChannel	
Type	uint8	
Description	Indicates the value of ERS channel configured to the group.	
Verification method	The structure member is generated with a suffixed value specified in the configuration parameter '/Mcu/Mcu/McuHardwareResourceAllocationConf_0/ McuEruAllocationConf_0/McuEruChannelInputLineConf_3' after McuEruChannelInputLineConf_. *Note: This parameter is user configurable only when AdcGroupTriggSrc is 'ADC_TRIGG_SRC_HW ' and AdcHwExtTrigSelect is 'ERUIOUT'.*	
Example(s)	**Action**	**Generated output**
	Configure AdcEruErsRef as McuEruAllocationConf_0/McuEruChannelInputLineConf_3.	`3U /*ERS channel*/`
	Configure AdcEruErsRef as McuEruAllocationConf_0/McuEruChannelInputLineConf_7.	`7U /*ERS channel*/`

1.2.9.4 Member: OguChannel

Table 120 OguChannel

Name	OguChannel	
Type	uint8	
Description	Indicates the value of OGU channel configured to the group.	
Verification method	The structure member is generated with a suffixed value specified in the configuration parameter '/Mcu/Mcu/McuHardwareResourceAllocationConf_0/ McuEruAllocationConf_0/McuEruChannelOutputUnitConf_0' after McuEruChannelOutputUnitConf_. *Note: This parameter is user configurable only when AdcGroupTriggSrc is 'ADC_TRIGG_SRC_HW ' and AdcHwExtTrigSelect is 'ERUIOUT'.*	
Example(s)	**Action**	**Generated output**
	Configure AdcEruErsRef as McuEruAllocationConf_0/McuEruChannelOutputUnitConf_2.	`2U /*OGU channel*/`
	Configure AdcEruErsRef as McuEruAllocationConf_0/McuEruChannelOutputUnitConf_6.	`6U /*OGU channel*/`

1.2.10 Structure: Adc_kHwUnit[x]Grp[name]EruGate_Config[_<variant>]

Table 121 Adc_kHwUnit[x]Grp[name]EruGate_Config[_<variant>]

Name	Adc_kHwUnit[x]Grp[name]EruGate_Config[_<variant>]
Type	Adc_EruChannelCfgType

Description	Configuration structure of ADC driver for configured ERU gate. ('x' = HwUnit ID starting from 0 to Max HwUnits available in the derivative and 'name'= Name of the group configured to the HW unit).	
Verification method	The generated structure member is present in the Adc_kHwUnit[x]Grp_Config[_<variant>][y] structure. For a variant-aware configuration, Member name is appended with the <variant> name. For variant-unaware configuration <variant> is ignored.	
Example(s)	**Action**	**Generated output**

	Configure ERU gate to group ‘Adc0Group_5’ of HwUnit0.
	(variant Petrol)
		`static const Adc_EruChannelCfgType Adc_kHwUnit0GrpAdc0Group_5EruGate_Config_Petrol=`
		`{`
		`  0x0520U, /*EICR register configuration*/`
		`  0x0001U, /*IGCR register configuration*/`
		`  0U, /*ERS channel*/`
		`  0U /*OGU channel*/`
		`};`

	Configure ERU gate to group ‘Adc1Group_0’ of HwUnit1.
	(variant unaware)
		`static const Adc_EruChannelCfgType Adc_kHwUnit1GrpAdc1Group_0EruGate_Config=`
		`{`
		`  0x0600U, /*EICR register configuration*/`
		`  0x0020U, /*IGCR register configuration*/`
		`  5U, /*ERS channel*/`
		`  3U /*OGU channel*/`
		`};`

1.2.10.1 Member: EruEicrCfg

Table 122 EruEicrCfg

Name	EruEicrCfg
Type	uint16
Description	Indicates the value of external input channel configured to the group.
Verification method	The structure member is generated as a value of external input channel configured to the group for EICR register. Bits 4-6 store the suffixed value after _SEL of ‘AdcEruErsInputPin’. Bits 8-9 store the value configured in AdcHwTrigSignal. Bit 10 always generates 1. *Note: This parameter is user configurable only when AdcGroupTriggSrc is 'ADC_TRIGG_SRC_HW ' and AdcHwExtTrigSelect is 'ERUIOUT'.* *Note: ERS channel configured in ‘AdcEruErsInputPin’ is used to determine the bit position of EICR SFR i.e. Odd Channel starts from bit position is 16, and even channel starts from bit position is 0.*

Example(s)	Action	Generated output
	• Configure AdcHwGateSignal as ADC_GATE_LVL_HIGH. • Configure AdcEruErsInputPin as ERS_REQ5A_PORTS_P15_8_SEL0 .	`0x0600U /*EICR register configuration*/`
	• Configure AdcHwGateSignal as ADC_GATE_LVL_LOW. • Configure AdcEruErsInputPin as ERS_REQ5B_GTM_TOM1_12_SEL 1.	`0x0510U /*EICR register configuration*/`

1.2.10.2 Member: EruIgcrCfg

Table 123 EruIgcrCfg

Name	EruIgcrCfg
Type	uint16
Description	Indicates the value of gating control configured to the group.
Verification method	The structure member is generated as a value of gating control configured to the group for IGCR register. Bits 0-7 select the channel based on the suffix value specified in the configuration parameter '/Mcu/Mcu/McuHardwareResourceAllocationConf_0/ McuEruAllocationConf_0/McuEruChannelInputLineConf_3' after McuEruChannelInputLineConf_ *Note: This parameter is generated only when AdcGroupTriggSrc is 'ADC_TRIGG_SRC_HW ' and AdcHwExtTrigSelect is 'ERUIOUT'.* *Note: ERS channel configured in 'AdcEruErsInputPin' is used to determine the bit position of EICR SFR i.e. Odd Channel starts from bit position is 16, and even channel starts from bit position is 0.*

Example(s)	Action	Generated output
	Configure AdcEruErsRef as McuEruChannelInputLineConf_0	`0x0001U /*IGCR register configuration*/`
	Configure AdcEruErsRef as McuEruChannelInputLineConf_5	`0x0020U /*IGCR register configuration*/`

1.2.10.3 Member: ErsChannel

Table 124 ErsChannel

Name	ErsChannel
Type	uint8
Description	Indicates the value of ERS channel configured to the group.
Verification	The structure member is generated with a suffixed value specified in the configuration parameter

method	'/Mcu/Mcu/McuHardwareResourceAllocationConf_0/ McuEruAllocationConf_0/McuEruChannelInputLineConf_3' after McuEruChannelInputLineConf_. *Note: This parameter is user configurable only when AdcGroupTriggSrc is 'ADC_TRIGG_SRC_HW ' and AdcHwExtTrigSelect is 'ERUIOUT'.*

Example(s)	**Action**	**Generated output**
	Configure AdcEruErsRef as McuEruChannelInputLineConf_3	`3U /*ERS channel*/`
	Configure AdcEruErsRef as McuEruChannelInputLineConf_7	`7U /*ERS channel*/`

1.2.10.4 Member: OguChannel

Table 125 OguChannel

Name	OguChannel
Type	uint8
Description	Indicates the value of OGU channel configured to the group.
Verification method	The structure member is generated with a suffixed value specified in the configuration parameter '/Mcu/Mcu/McuHardwareResourceAllocationConf_0/ McuEruAllocationConf_0/McuEruChannelOutputUnitConf_0' after McuEruChannelOutputUnitConf_. *Note: This parameter is user configurable only when AdcGroupTriggSrc is 'ADC_TRIGG_SRC_HW ' and AdcHwExtTrigSelect is 'ERUIOUT'.*

Example(s)	**Action**	**Generated output**
	Configure AdcEruErsRef as McuEruChannelOutputUnitConf_2	`2U /*OGU channel*/`
	Configure AdcEruErsRef as McuEruChannelOutputUnitConf_6	`6U /*OGU channel*/`

1.2.11 Structure: Adc_kHwUnit[x]Grp[name]GtmTrig_Config[_<variant>]

Table 126 Adc_kHwUnit[x]Grp[name]GtmTrig_Config[_<variant>]

Name	Adc_kHwUnit[x]Grp[name]GtmTrig_Config[_<variant>]	
Type	Mcu_17_Gtm_TomAtomChConfigType	
Description	Configuration structure of ADC driver for configured GTM trigger. ('x' = HwUnit ID starting from 0 to Max HwUnits available in the derivative and 'name'= Name of the group configured to the HW unit).	
Verification method	The generated structure member is present in the Adc_kHwUnit[x]Grp_Config[_<variant>][y] structure. For a variant-aware configuration, Member name is appended with the <variant> name. For variant-unaware configuration <variant> is ignored.	
Example(s)	**Action**	**Generated output**

	Configure GTM trigger to group ‘Adc0Group_5’ of HwUnit0.
	(variant Petrol)
		`static const Mcu_17_Gtm_TomAtomChConfigType Adc_kHwUnit0GrpAdc0Group_5GtmTrig_Config_Petrol=`
		`{`
		`  MCU_GTM_TIMER_TOM, /*GTM_TOM Timer Type Used*/`
		`  0x00000006U, /* Timer ID */`
		`  0x00002800U, /*Control Register Value for GTM_TOM_0 */`
		`  0x00000000U, /*CN0 Register value*/`
		`  0x000003d0U, /*CM0 register value*/`
		`  0x000001e8U, /*CM1 register value*/`
		`  0x000003d0U, /*SR0 register value*/`
		`  0x000001e8U, /*SR1 register value*/`
		`  0x00U /*Interrupt Enable and Interrupt Mode values*/`
		`};`

	Configure GTM trigger to group ‘Adc0Group_8’ of HwUnit0.
	(variant unaware)
		`static const Mcu_17_Gtm_TomAtomChConfigType Adc_kHwUnit0GrpAdc0Group_8GtmTrig_Config=`
		`{`
		`  MCU_GTM_TIMER_ATOM, /*GTM_ATOM Timer Type Used*/`
		`  0x00000305U, /* Timer ID */`
		`  0x00005802U, /*Control Register Value for GTM_ATOM_3 */`
		`  0x00000000U, /*CN0 Register value*/`
		`  0x00001388U, /*CM0 register value*/`
		`  0x000009c4U, /*CM1 register value*/`
		`  0x00001388U, /*SR0 register value*/`
		`  0x000009c4U, /*SR1 register value*/`
		`  0x00U /*Interrupt Enable and Interrupt Mode values*/`
		`};`

1.2.11.1 Member: TimerType

Table 127 TimerType

Name	TimerType	
Type	Mcu_17_Gtm_TimerOutType	
Description	TOM/ATOM channel used to service the ADC driver.	
Verification method	The structure member is generated with TOM/ATOM timer type used to service the ADC driver.	
Example(s)	**Action**	**Generated output**

	Configure GtmTimerUsed = McuGtmTomAllocationConf_0 /McuGtmTomChannelAllocation Conf_0 in GtmTimerConfiguration_0	`MCU_GTM_TIMER_TOM /*GTM_TOM Timer Type Used*/`
	Configure GtmTimerUsed = McuGtmAtomAllocationConf_0 /McuGtmAtomChannelAllocation Conf_0 in GtmTimerConfiguration_0	`MCU_GTM_TIMER_ATOM /*GTM_ATOM Timer Type Used*/`

1.2.11.2 Member: TimerId

Table 128 TimerId

Name	TimerId	
Type	Mcu_17_Gtm_TimerChIdentifierType	
Description	TOM/ATOM channel identifier.	
Verification method	The structure member is generated as numeric value used to represent timer module number and channel number.	
Example(s)	**Action**	**Generated output**
	Configure GtmTimerUsed = McuGtmAtomAllocationConf_3 /McuGtmAtomChannelAllocationConf_3 in GtmTimerConfiguration_0	`0x00000303U /* Timer ID */`
	Configure GtmTimerUsed = McuGtmTomAllocationConf_0 /McuGtmTomChannelAllocation Conf_6 in GtmTimerConfiguration_0	`0x00000006U /* Timer ID */`

1.2.11.3 Member: TimerChCtrlReg

Table 129 TimerChCtrlReg

Name	TimerChCtrlReg
Type	uint32
Description	TOM/ATOM channel control registers value.
Verification method	The structure member is generated as value of the control register for TOM/ATOM channel. Steps to calculate TimerChCtrlReg: • Fixed value for TimerChCtrlReg is 0x00000802 for ATOM and 0x00000800 for TOM • Based on the GtmTimerClockSelect, value of clock select is left shifted by 12 and OR'ed with TimerChCtrlReg.

	TImerChCtrlReg = (TImerChCtrlReg | (ClockSelect<<12)) • Left shift 1 by 11 and OR'ed with TimerChCtrlReg. TImerChCtrlReg = (TImerChCtrlReg | (1<<11)) • If GTM Timer Type is 'ATOM' then TimerChCtrlReg OR'ed with 2. TImerChCtrlReg = (TImerChCtrlReg | 2)	
Example(s)	**Action**	**Generated output**
	• Configure GtmTimerUsed = McuGtmTomAllocationConf_1 /McuGtmTomChannelAllocationConf_6 in GtmTimerConfiguration_0. • Configure GtmTimerClockSelect = GTM_FIXED_CLOCK_2 in GtmTimerConfiguration_0.	`0x00002800U /*Control Register Value for GTM_TOM_1 */`
	• Configure GtmTimerUsed = McuGtmAtomAllocationConf_3/ McuGtmAtomChannelAllocationConf_5 GtmTimerConfiguration_0. • Configure GtmTimerClockSelect = GTM_CONFIGURABLE_CLOCK_5 in GtmTimerConfiguration_0	`0x00005802U /*Control Register Value for GTM_ATOM_3 */`

1.2.11.4 Member: TimerChCN0Reg

Table 130 TimerChCN0Reg

Name	TimerChCN0Reg	
Type	uint32	
Description	TOM/ATOM channel CN0 register value.	
Verification method	The structure member is generated as value of the CN0 register for TOM/ATOM channel. *Note: This member is not configurable by the user and always generated as 0.*	
Example(s)	**Action**	**Generated output**
	Generate Adc[_<variant>]_PBcfg.c	`0x00000000U /*CN0 Register value*/`

1.2.11.5 Member: TimerChCM0Reg

Table 131 TimerChCM0Reg

Name	TimerChCM0Reg

Type	uint32	
Description	TOM/ATOM channel CM0 register value.	
Verification method	The structure member is generated as value of the CM0 register for TOM/ATOM channel. Steps to calculate TimerChCM0Reg: • GTM frequency calculation fGtm=((McuGTMFrequency * GtmDenominator)/ GtmNumerator). • Derive the TOM and ATOM timer from the configure parameter GtmTimerClockSelect. • Calculate the fGTM: fGtm=(fGtm / GtmClusterDivVal) fGtm= fGtm/ ClockDivider • Calculate TomChCM0Reg: TomChCM0Reg: = ((GtmTimerTimePeriod * fGtm)/100000)	
Example(s)	**Action**	**Generated output**
	• Configure GtmTimerUsed = McuGtmAtomAllocationConf_1/ McuGtmAtomChannelAllocationConf_6 in GtmTimerConfiguration_0. • Configure GtmTimerClockSelect = GTM_CONFIGURABLE_CLOCK_4 in GtmTimerConfiguration_0 • Configure GtmTimerTimePeriod= 6000 in GtmTimerConfiguration_0. • Configure CLS_CLK_CFG_ENABLED_WITH_DIV_SEL2 in GtmGlobalConfiguration_0/ GtmClusterConf/ GtmClusterConf_0/ GtmCmuClusterInputClockDividerEnable. • Configure GtmClusterConfClock4Src= CMU_CONF_CLOCK4_SEL0 in GtmGlobalConfiguration/*[1]/GtmCluster Conf/ GtmClusterConf_1/ GtmClusterConfClockSetting. • Configre GTM frequency = 50MHZ.	`0x00000bb8U /*CM0 register value*/`
	• Configure GtmTimerUsed = McuGtmAtomAllocationConf_1/ McuGtmAtomChannelAllocationConf_6 in	`0x00001770U /*CM0 register value*/ value*/`

GtmTimerConfiguration_0. • Configure GtmTimerClockSelect = GTM_CONFIGURABLE_CLOCK_4 in GtmTimerConfiguration_0. • Configure GtmTimerTimePeriod = 6000 in GtmTimerConfiguration_0. • Configure CLS_CLK_CFG_ENABLED_WITHOUT_DIV_SEL1 in GtmGlobalConfiguration_0/ GtmClusterConf/ GtmClusterConf_0/ GtmCmuClusterInputClockDividerEnable. • Configure GtmClusterConfClock4Src= CMU_CONF_CLOCK8_SEL1 in GtmGlobalConfiguration/*[1]/GtmClusterConf/ GtmClusterConf_0/ GtmClusterConfClockSetting. • Configre GTM frequency = 50MHZ.	
• Configure GtmTimerUsed = McuGtmTomAllocationConf_3/ McuGtmTomChannelAllocationConf_3 in GtmTimerConfiguration_0. • Configure GtmTimerClockSelect = GTM_FIXED_CLOCK_3 in GtmTimerConfiguration_0. • Configure GtmTimerTimePeriod = 100000 in GtmTimerConfiguration_0. • Configure CLS_CLK_CFG_ENABLED_WITH_DIV_SEL2 in GtmGlobalConfiguration_0/ GtmClusterConf/ GtmClusterConf_0/ GtmCmuClusterInputClockDividerEnable. • Configure GtmClusterConfClock4Src= CMU_CONF_CLOCK4_SEL0 in	`0x00000262U /*CM0 register value*/`

	GtmGlobalConfiguration/*[1]/GtmCluster Conf/ GtmClusterConf_0/ GtmClusterConfClockSetting. • Configre GTM frequency = 50MHZ.	

1.2.11.6 Member: TimerChCM1Reg

Table 132 TimerChCM1Reg

Name	TimerChCM1Reg	
Type	uint32	
Description	TOM/ATOM channel CM1 register value.	
Verification method	The structure member is generated as value of the CM1 register for TOM/ATOM channel. Calculate TimerChCM1Reg: • TimerChCM1Reg = (TimerChCM0Reg/2) (TimerChCM0Reg value is derived as mentioned in the Table 131 verification method) *Note: This member is not configurable by the user*	
Example(s)	**Action**	**Generated output**
	Generate Adc[_<variant>]_PBcfg.c for below configurations used for generation of TimerChCM0Reg • Configure GtmTimerUsed = McuGtmAtomAllocationConf_1/ McuGtmAtomChannelAllocationConf_6 in GtmTimerConfiguration_0. • Configure GtmTimerClockSelect = GTM_CONFIGURABLE_CLOCK_4 in GtmTimerConfiguration_0. • GtmTimerTimePeriod = 6000 in GtmTimerConfiguration_0. • Configure CLS_CLK_CFG_ENABLED_WITH_ DIV_SEL2 in GtmGlobalConfiguration_0/ GtmClusterConf/ GtmClusterConf_0/ GtmCmuClusterInputClockDividerEnable. • Configure GtmClusterConfClock4Src= CMU_CONF_CLOCK4_SEL0 in	`0x00000131U /*CM1 register value*/`

GtmGlobalConfiguration/*[1]/GtmCluster Conf/ GtmClusterConf_0/ GtmClusterConfClockSetting. • Configre GTM frequency = 50MHZ.	
Generate Adc[_<variant>]_PBcfg.c for below configurations used for generation of TimerChCM0Reg • Configure GtmTimerUsed = McuGtmAtomAllocationConf_1/ McuGtmAtomChannelAllocationConf_6 in GtmTimerConfiguration_0. • Configure GtmTimerClockSelect = GTM_CONFIGURABLE_CLOCK_4 in GtmTimerConfiguration_0. • Configure GtmTimerTimePeriod=6000 in GtmTimerConfiguration_0. • Configure CLS_CLK_CFG_ENABLED_WITHOUT_DIV_SEL1 in GtmGlobalConfiguration_0/ GtmClusterConf/ GtmClusterConf_0/ GtmCmuClusterInputClockDividerEnable. • Configure GtmClusterConfClock4Src= CMU_CONF_CLOCK8_SEL1 in GtmGlobalConfiguration/*[1]/GtmCluster Conf/ GtmClusterConf_0/ GtmClusterConfClockSetting. • Configre GTM frequency = 50MHZ.	`0x00000bb8U /*CM1 register value*/`

1.2.11.7 Member: TimerChSR0Reg

Table 133 TimerChSR0Reg

Name	TimerChSR0Reg	
Type	uint32	
Description	TOM/ATOM channel SR0 register value.	
Verification method	The structure member is generated as value of the SR0 register for TOM/ATOM channel. Calculate TimerChSR0Reg: • TimerChSR0Reg = (TimerChCM0Reg) (TimerChCM0Reg value is derived as mentioned in the Table 131 verification method) *Note: This member is not configurable by the user*	
Example(s)	**Action**	**Generated output**
	Generate Adc[_<variant>]_PBcfg.c for below configurations used for generation of TimerChCM0Reg • Configure GtmTimerUsed = McuGtmAtomAllocationConf_1/ McuGtmAtomChannelAllocationConf_6 in GtmTimerConfiguration_0. • Configure GtmTimerClockSelect = GTM_CONFIGURABLE_CLOCK_4 in GtmTimerConfiguration_0. • Configure GtmTimerTimePeriod= 6000 in GtmTimerConfiguration_0. • Configure CLS_CLK_CFG_ENABLED_WITH_ DIV_SEL2 in GtmGlobalConfiguration_0/ GtmClusterConf/ GtmClusterConf_0/ GtmCmuClusterInputClockDividerEnable. • Configure GtmClusterConfClock4Src= CMU_CONF_CLOCK4_SEL0 in GtmGlobalConfiguration/*[1]/GtmCluster Conf/ GtmClusterConf_0/ GtmClusterConfClockSetting. • Configre GTM frequency = 50MHZ.	`0x00000bb8U /*SR0 register value*/`

	Generate Adc[_<variant>]_PBcfg.c for below configurations used for generation of TimerChCM0Reg • Configure GtmTimerUsed = McuGtmAtomAllocationConf_1/ McuGtmAtomChannelAllocationConf_6 in GtmTimerConfiguration_0. • Configure GtmTimerClockSelect = GTM_CONFIGURABLE_CLOCK_4 in GtmTimerConfiguration_0. • Configure GtmTimerTimePeriod = 6000 in GtmTimerConfiguration_0. • Configure CLS_CLK_CFG_ENABLED_WITHOUT_DIV_SEL1 in GtmGlobalConfiguration_0/ GtmClusterConf/ GtmClusterConf_0/ GtmCmuClusterInputClockDividerEnable. • Configure GtmClusterConfClock4Src= CMU_CONF_CLOCK8_SEL1 in GtmGlobalConfiguration/*[1]/GtmClusterConf/ GtmClusterConf_0/ GtmClusterConfClockSetting. • Configre GTM frequency = 50MHZ.	`0x00001770U /*SR0 register value*/ value*/`

1.2.11.8 Member: TimerChSR1Reg

Table 134 TimerChSR1Reg

Name	TimerChSR1Reg
Type	uint32
Description	TOM/ATOM channel SR1 register value.
Verification method	The structure member is generated as value of the SR1 register for TOM/ATOM channel. Calculate TimerChSR1Reg: • TimerChSR1Reg = (TimerChCM0Reg/2) (TimerChCM0Reg value is derived as mentioned in the Table 131 verification method) *Note: This member is not configurable by the user*

Example(s)	Action	Generated output
	Generate Adc[_<variant>]_PBcfg.c for below configurations used for generation of TimerChCM0Reg • Configure GtmTimerUsed = McuGtmAtomAllocationConf_1/ McuGtmAtomChannelAllocationConf_6 in GtmTimerConfiguration_0. • Configure GtmTimerClockSelect = GTM_CONFIGURABLE_CLOCK_4 in GtmTimerConfiguration_0 • Configure GtmTimerTimePeriod= 6000 in GtmTimerConfiguration_0. • Configure CLS_CLK_CFG_ENABLED_WITH_DIV_SEL2 in GtmGlobalConfiguration_0/ GtmClusterConf/ GtmClusterConf_0/ GtmCmuClusterInputClockDividerEnable. • Configure GtmClusterConfClock4Src= CMU_CONF_CLOCK4_SEL0 in GtmGlobalConfiguration/*[1]/GtmCluster Conf/ GtmClusterConf_1/ GtmClusterConfClockSetting. • Configre GTM frequency = 50MHZ.	`0x000005dcU /*SR1 register value*/`
	Generate Adc[_<variant>]_PBcfg.c for below configurations used for generation of TimerChCM0Reg • Configure GtmTimerUsed = McuGtmAtomAllocationConf_1/ McuGtmAtomChannelAllocationConf_6 in GtmTimerConfiguration_0. • Configure GtmTimerClockSelect = GTM_CONFIGURABLE_CLOCK_4 in GtmTimerConfiguration_0.	`0x00000bb8U /*SR1 register value*/`

	• Configure GtmTimerTimePeriod = 6000 in GtmTimerConfiguration_0. • Configure CLS_CLK_CFG_ENABLED_WITHOUT_DIV_SEL1 in GtmGlobalConfiguration_0/ GtmClusterConf/ GtmClusterConf_0/ GtmCmuClusterInputClockDividerEnable. • Configure GtmClusterConfClock4Src= CMU_CONF_CLOCK8_SEL1 in GtmGlobalConfiguration/*[1]/GtmClusterConf/ GtmClusterConf_0/ GtmClusterConfClockSetting. • Configre GTM frequency = 50MHZ.	

1.2.11.9 Member: TimerChIntEnMode

Table 135 TimerChIntEnMode

Name	TimerChIntEnMode	
Type	uint8	
Description	TOM/ATOM channel interrupt enable and interrupt mode values.	
Verification method	The structure member is generated as value of the interrupt enable and interrupt mode for TOM/ATOM. *Note: This member is not configurable by the user*	
Example(s)	**Action**	**Generated output**
	Generate Adc[_<variant>]_PBcfg.c	`0x00U /*Interrupt Enable and Interrupt Mode values*/`

1.2.12 Structure: Adc_kHwUnit[x]Grp[name]GtmGate_Config[_<variant>]

Table 136 Adc_kHwUnit[x]Grp[name]GtmGate_Config[_<variant>]

Name	Adc_kHwUnit[x]Grp[name]GtmGate_Config[_<variant>]
Type	Mcu_17_Gtm_TomAtomChConfigType
Description	Configuration structure of ADC driver for configured GTM gate. ('x' = HwUnit ID starting from 0 to Max HwUnits available in the derivative and 'name'= Name of the group configured to the HW unit).
Verification method	The generated structure member is present in the Adc_kHwUnit[x]Grp_Config[_<variant>][y] structure. For a variant-aware configuration, Member name is appended with the <variant> name. For variant-unaware configuration <variant> is ignored.

Example(s)	Action	Generated output

	Configure GTM gate to group ‘Adc0Group_5’ of HwUnit0.
	(variant Petrol)
		`static const Mcu_17_Gtm_TomAtomChConfigType Adc_kHwUnit0GrpAdc0Group_5GtmGate_Config_Petrol=`
		`{`
		`MCU_GTM_TIMER_TOM, /*GTM_TOM Timer Type Used*/`
		`0x00000006U, /* Timer ID */`
		`0x00002800U, /*Control Register Value for GTM_TOM_0 */`
		`0x00000000U, /*CN0 Register value*/`
		`0x000003d0U, /*CM0 register value*/`
		`0x000001e8U, /*CM1 register value*/`
		`0x000003d0U, /*SR0 register value*/`
		`0x000001e8U, /*SR1 register value*/`
		`0x00U /*Interrupt Enable and Interrupt Mode values*/`
		`};`

	Configure GTM gate to group ‘Adc0Group_8’ of HwUnit0.
	(variant unaware)
		`static const Mcu_17_Gtm_TomAtomChConfigType Adc_kHwUnit0GrpAdc0Group_8GtmGate_Config=`
		`{`
		`MCU_GTM_TIMER_ATOM, /*GTM_ATOM Timer Type Used*/`
		`0x00000305U, /* Timer ID */`
		`0x00005802U, /*Control Register Value for GTM_ATOM_3 */`
		`0x00000000U, /*CN0 Register value*/`
		`0x00001388U, /*CM0 register value*/`
		`0x000009c4U, /*CM1 register value*/`
		`0x00001388U, /*SR0 register value*/`
		`0x000009c4U, /*SR1 register value*/`
		`0x00U /*Interrupt Enable and Interrupt Mode values*/`
		`};`

1.2.12.1 Member: TimerType

Table 137 TimerType

Name	TimerType
Type	Mcu_17_Gtm_TimerOutType
Description	TOM/ATOM channel used to service the ADC driver.
Verification method	The structure member is generated with TOM/ATOM timer type used to service the ADC driver.

Example(s)	Action	Generated output
	Configure GtmTimerUsed = McuGtmTomAllocationConf_0 /McuGtmTomChannelAllocation Conf_0 in GtmTimerConfiguration_0	`MCU_GTM_TIMER_TOM /*GTM_TOM Timer Type Used*/`
	Configure GtmTimerUsed = McuGtmAtomAllocationConf_0 /McuGtmAtomChannelAllocation Conf_0 in GtmTimerConfiguration_0	`MCU_GTM_TIMER_ATOM /*GTM_ATOM Timer Type Used*/`

1.2.12.2 Member: TimerId

Table 138 TimerId

Name	TimerId	
Type	Mcu_17_Gtm_TimerChIdentifierType	
Description	TOM/ATOM channel identifier.	
Verification method	The structure member is generated as numeric value used to represent timer module number and channel number.	
Example(s)	**Action**	**Generated output**
	Configure GtmTimerUsed = McuGtmAtomAllocationConf_3 /McuGtmAtomChannelAllocationConf_3 in GtmTimerConfiguration_0	`0x00000303U /* Timer ID */`
	Configure GtmTimerUsed = McuGtmTomAllocationConf_0 /McuGtmTomChannelAllocation Conf_6 in GtmTimerConfiguration_0	`0x00000006U /* Timer ID */`

1.2.12.3 Member: TimerChCtrlReg

Table 139 TimerChCtrlReg

Name	TimerChCtrlReg
Type	uint32
Description	TOM/ATOM channel control registers value.
Verification method	The structure member is generated as value of the control register for TOM/ATOM channel. Steps to calculate TimerChCtrlReg: • Fixed value for TimerChCtrlReg is 0x00000802 for ATOM and 0x00000800 for TOM • Based on the GtmTimerClockSelect, value of clock select is left shifted by 12 and OR'ed

	with TimerChCtrlReg. TImerChCtrlReg = (TImerChCtrlReg | (ClockSelect<<12)) • Left shift 1 by 11 and OR'ed with TimerChCtrlReg. TImerChCtrlReg = (TImerChCtrlReg | (1<<11)) • If GTM Timer Type is 'ATOM' then TimerChCtrlReg OR'ed with 2. • TImerChCtrlReg = (TImerChCtrlReg | 2)	
Example(s)	**Action**	**Generated output**
	• Configure GtmTimerUsed = McuGtmTomAllocationConf_1 /McuGtmTomChannelAllocation Conf_6 in GtmTimerConfiguration_0. • Configure GtmTimerClockSelect = GTM_FIXED_CLOCK_2 in GtmTimerConfiguration_0.	`0x00002800U /*Control Register Value for GTM_TOM_1 */`
	• Configure GtmTimerUsed = McuGtmAtomAllocationConf_3/ McuGtmAtomChannelAllocationConf_5 GtmTimerConfiguration_0. • Configure GtmTimerClockSelect = GTM_CONFIGURABLE_CLOCK_5 in GtmTimerConfiguration_0.	`0x00005802U /*Control Register Value for GTM_ATOM_3 */`

1.2.12.4 Member: TimerChCN0Reg

Table 140 TimerChCN0Reg

Name	TimerChCN0Reg	
Type	uint32	
Description	TOM/ATOM channel CN0 register value.	
Verification method	The structure member is generated as value of the CN0 register for TOM/ATOM channel. *Note: This member is not configurable by the user*	
Example(s)	**Action**	**Generated output**
	Generate Adc[_<variant>]_PBcfg.c	`0x00000000U /*CN0 Register value*/`

1.2.12.5 Member: TimerChCM0Reg

Table 141 TimerChCM0Reg

Name	TimerChCM0Reg	
Type	uint32	
Description	TOM/ATOM channel CM0 register value.	
Verification method	The structure member is generated as value of the CM0 register for TOM/ATOM channel. Steps to calculate TimerChCM0Reg: • GTM frequency calculation fGtm=((McuGTMFrequency * GtmDenominator)/ GtmNumerator). • Derive the TOM and ATOM timer from the configure parameter GtmTimerClockSelect. • Calculate the fGTM: fGtm=(fGtm / GtmClusterDivVal) fGtm= fGtm/ ClockDivider • Calculate TomChCM0Reg: TomChCM0Reg: = ((GtmTimerTimePeriod * fGtm)/100000)	
Example(s)	**Action**	**Generated output**
	• Configure GtmTimerUsed = McuGtmAtomAllocationConf_1/ McuGtmAtomChannelAllocationConf_6 in GtmTimerConfiguration_0. • Configure GtmTimerClockSelect = GTM_CONFIGURABLE_CLOCK_4 in GtmTimerConfiguration_0 • Configure GtmTimerTimePeriod= 6000 in GtmTimerConfiguration_0. • Configure CLS_CLK_CFG_ENABLED_WITH_DIV_SEL2 in GtmGlobalConfiguration_0/ GtmClusterConf/ GtmClusterConf_0/ GtmCmuClusterInputClockDividerEnable. • Configure GtmClusterConfClock4Src= CMU_CONF_CLOCK4_SEL0 in GtmGlobalConfiguration/*[1]/GtmCluster Conf/ GtmClusterConf_1/ GtmClusterConfClockSetting.	`0x00000bb8U /*CM0 register value*/`

• Configre GTM frequency = 50MHZ.	
• Configure GtmTimerUsed = McuGtmAtomAllocationConf_1/ McuGtmAtomChannelAllocationConf_6 in GtmTimerConfiguration_0. • Configure GtmTimerClockSelect = GTM_CONFIGURABLE_CLOCK_4 in GtmTimerConfiguration_0. • Configure GtmTimerTimePeriod = 6000 in GtmTimerConfiguration_0. • Configure CLS_CLK_CFG_ENABLED_WITHOUT_DIV_SEL1 in GtmGlobalConfiguration_0/ GtmClusterConf/ GtmClusterConf_0/ GtmCmuClusterInputClockDividerEnable. • Configure GtmClusterConfClock4Src= CMU_CONF_CLOCK8_SEL1 in GtmGlobalConfiguration/*[1]/GtmClusterConf/ GtmClusterConf_0/ GtmClusterConfClockSetting. • Configre GTM frequency = 50MHZ.	`0x00001770U /*CM0 register value*/ value*/`
• Configure GtmTimerUsed = McuGtmTomAllocationConf_3/ McuGtmTomChannelAllocationConf_3 in GtmTimerConfiguration_0. • Configure GtmTimerClockSelect = GTM_FIXED_CLOCK_3 in GtmTimerConfiguration_0. • Configure GtmTimerTimePeriod = 100000 in GtmTimerConfiguration_0. • Configure CLS_CLK_CFG_ENABLED_WITH_DIV_SEL2 in GtmGlobalConfiguration_0/ GtmClusterConf/ GtmClusterConf_0/	`0x00000262U /*CM0 register value*/`

	GtmCmuClusterInputClockDividerEnable. • Configure GtmClusterConfClock4Src= CMU_CONF_CLOCK4_SEL0 in GtmGlobalConfiguration/*[1]/GtmCluster Conf/ GtmClusterConf_0/ GtmClusterConfClockSetting. • Configre GTM frequency = 50MHZ.	

1.2.12.6 Member: TimerChCM1Reg

Table 142 TimerChCM1Reg

Name	TimerChCM1Reg	
Type	uint32	
Description	TOM/ATOM channel CM1 register value.	
Verification method	The structure member is generated as value of the CM1 register for TOM/ATOM channel. Steps to calculate TimerChCM1Reg • TimerChCM1Reg = (TimerChCM0Reg/2) (TimerChCM0Reg value is derived as mentioned in the Table 141verification method) *Note: This member is not configurable by the user*	
Example(s)	**Action**	**Generated output**
	Generate Adc[_<variant>]_PBcfg.c for below configurations used for generation of TimerChCM0Reg • Configure GtmTimerUsed = McuGtmAtomAllocationConf_1/ McuGtmAtomChannelAllocationConf_6 in GtmTimerConfiguration_0. • Configure GtmTimerClockSelect = GTM_CONFIGURABLE_CLOCK_4 in GtmTimerConfiguration_0. • GtmTimerTimePeriod = 6000 in GtmTimerConfiguration_0. • Configure CLS_CLK_CFG_ENABLED_WITH_ DIV_SEL2 in GtmGlobalConfiguration_0/ GtmClusterConf/ GtmClusterConf_0/	`0x000005dcU /*CM1 register value*/`

GtmCmuClusterInputClockDividerEnable. • Configure GtmClusterConfClock4Src= CMU_CONF_CLOCK4_SEL0 in GtmGlobalConfiguration/*[1]/GtmCluster Conf/ GtmClusterConf_0/ GtmClusterConfClockSetting. • Configre GTM frequency = 50MHZ.	
Generate Adc[_<variant>]_PBcfg.c for below configurations used for generation of TimerChCM0Reg • Configure GtmTimerUsed = McuGtmAtomAllocationConf_1/ McuGtmAtomChannelAllocationConf_6 in GtmTimerConfiguration_0. • Configure GtmTimerClockSelect = GTM_CONFIGURABLE_CLOCK_4 in GtmTimerConfiguration_0. • Configure GtmTimerTimePeriod=6000 in GtmTimerConfiguration_0. • Configure CLS_CLK_CFG_ENABLED_WITHOUT_DIV_SEL1 in GtmGlobalConfiguration_0/ GtmClusterConf/ GtmClusterConf_0/ GtmCmuClusterInputClockDividerEnable. • Configure GtmClusterConfClock4Src= CMU_CONF_CLOCK8_SEL1 in GtmGlobalConfiguration/*[1]/GtmCluster Conf/ GtmClusterConf_0/ GtmClusterConfClockSetting. • Configre GTM frequency = 50MHZ.	`0x00000bb8U /*CM1 register value*/`

1.2.12.7 Member: TimerChSR0Reg

Table 143 TimerChSR0Reg

Name	TimerChSR0Reg

Type	uint32	
Description	TOM/ATOM channel SR0 register value.	
Verification method	The structure member is generated as value of the SR0 register for TOM/ATOM channel. Steps to calculate TimerChSR0Reg • TimerChSR0Reg = (TimerChCM0Reg) (TimerChCM0Reg value is derived as mentioned in the Table 141verification method) *Note: This member is not configurable by the user*	
Example(s)	**Action**	**Generated output**
	Generate Adc[_<variant>]_PBcfg.c for below configurations used for generation of TimerChCM0Reg • Configure GtmTimerUsed = McuGtmAtomAllocationConf_1/ McuGtmAtomChannelAllocationConf_6 in GtmTimerConfiguration_0. • Configure GtmTimerClockSelect = GTM_CONFIGURABLE_CLOCK_4 in GtmTimerConfiguration_0. • Configure GtmTimerTimePeriod= 6000 in GtmTimerConfiguration_0. • Configure CLS_CLK_CFG_ENABLED_WITH_ DIV_SEL2 in GtmGlobalConfiguration_0/ GtmClusterConf/ GtmClusterConf_0/ GtmCmuClusterInputClockDividerEnable. • Configure GtmClusterConfClock4Src= CMU_CONF_CLOCK4_SEL0 in GtmGlobalConfiguration/*[1]/GtmCluster Conf/ GtmClusterConf_0/ GtmClusterConfClockSetting. • Configre GTM frequency = 50MHZ.	`0x00000bb8U /*SR0 register value*/`
	Generate Adc[_<variant>]_PBcfg.c for below configurations used for generation of TimerChCM0Reg • Configure GtmTimerUsed =	`0x00001770U /*SR0 register value*/ value*/`

	McuGtmAtomAllocationConf_1/ McuGtmAtomChannelAllocationConf_6 in GtmTimerConfiguration_0. • Configure GtmTimerClockSelect = GTM_CONFIGURABLE_CLOCK_4 in GtmTimerConfiguration_0. • Configure GtmTimerTimePeriod = 6000 in GtmTimerConfiguration_0. • Configure CLS_CLK_CFG_ENABLED_WITHOUT_DIV_SEL1 in GtmGlobalConfiguration_0/ GtmClusterConf/ GtmClusterConf_0/ GtmCmuClusterInputClockDividerEnable. • Configure GtmClusterConfClock4Src= CMU_CONF_CLOCK8_SEL1 in GtmGlobalConfiguration/*[1]/GtmClusterConf/ GtmClusterConf_0/ GtmClusterConfClockSetting. • Configre GTM frequency = 50MHZ.	

1.2.12.8 Member: TimerChSR1Reg

Table 144 TimerChSR1Reg

Name	TimerChSR1Reg	
Type	uint32	
Description	TOM/ATOM channel SR1 register value.	
Verification method	The structure member is generated as value of the SR1 register for TOM/ATOM channel. Steps to calculate TimerChSR1Reg • TimerChSR1Reg = (TimerChCM0Reg/2) (TimerChCM0Reg value is derived as mentioned in the Table 141verification method) *Note: This member is not configurable by the user*	
Example(s)	**Action**	**Generated output**
	Generate Adc[_<variant>]_PBcfg.c for below configurations used for generation of TimerChCM0Reg • Configure GtmTimerUsed = McuGtmAtomAllocationConf_1/ McuGtmAtomChannelAllocationConf_6 in GtmTimerConfiguration_0. • Configure GtmTimerClockSelect = GTM_CONFIGURABLE_CLOCK_4 in GtmTimerConfiguration_0 • Configure GtmTimerTimePeriod= 6000 in GtmTimerConfiguration_0. • Configure CLS_CLK_CFG_ENABLED_WITH_DIV_SEL2 in GtmGlobalConfiguration_0/ GtmClusterConf/ GtmClusterConf_0/ GtmCmuClusterInputClockDividerEnable. • Configure GtmClusterConfClock4Src= CMU_CONF_CLOCK4_SEL0 in GtmGlobalConfiguration/*[1]/GtmCluster Conf/ GtmClusterConf_1/ GtmClusterConfClockSetting. • Configre GTM frequency = 50MHZ.	`0x000005dcU /*SR1 register value*/`

Generate Adc[_<variant>]_PBcfg.c for below configurations used for generation of TimerChCM0Reg • Configure GtmTimerUsed = McuGtmAtomAllocationConf_1/ McuGtmAtomChannelAllocationConf_6 in GtmTimerConfiguration_0. • Configure GtmTimerClockSelect = GTM_CONFIGURABLE_CLOCK_4 in GtmTimerConfiguration_0. • Configure GtmTimerTimePeriod = 6000 in GtmTimerConfiguration_0. • Configure CLS_CLK_CFG_ENABLED_WITHOUT_DIV_SEL1 in GtmGlobalConfiguration_0/ GtmClusterConf/ GtmClusterConf_0/ GtmCmuClusterInputClockDividerEnable. • Configure GtmClusterConfClock4Src= CMU_CONF_CLOCK8_SEL1 in GtmGlobalConfiguration/*[1]/GtmCluster Conf/ GtmClusterConf_0/ GtmClusterConfClockSetting. • Configre GTM frequency = 50MHZ.	`0x00000bb8U /*SR1 register value*/`

1.2.12.9 Member: TimerChIntEnMode

Table 145 TimerChIntEnMode

Name	TimerChIntEnMode	
Type	uint8	
Description	TOM/ATOM channel interrupt enable and interrupt mode values.	
Verification method	The structure member is generated as value of the interrupt enable and interrupt mode for TOM/ATOM. *Note: This member is not configurable by the user.*	
Example(s)	**Action**	**Generated output**
	Generate Adc[_<variant>]_PBcfg.c	`0x00U /*Interrupt Enable and Interrupt Mode values*/`

1.2.13 Function declaration: Adc_NotifyFnPtrType

Table 146 Adc_NotifyFnPtrType

Name	Adc_NotifyFnPtrType	
Type	Adc_NotifyFnPtrType *	
Description	The extern declaration of the user defined notification function which would be invoked on completion of Adc group conversion.	
Verification method	The function configured in 'AdcNotification' would be populated as a prototype with extern qualifier. *Note: This parameter is user configurable only when 'AdcGeneral/ AdcGrpNotifCapability' is enabled.* *Note: This prototype would not be generated if the function is configured as NULL_PTR in 'AdcNotification'.*	
Example(s)	**Action**	**Generated output**
	Configure 'IoHwAb_AdcNotification1' Notify function in 'AdcNotification' container.	`extern void IoHwAb_AdcNotification1(void);`
	Configure 'IoHwAb_AdcNotification5' Notify function in 'AdcNotification' container.	`extern void IoHwAb_AdcNotification5(void);`

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

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

1.3.1 Structure: Adc_Config[_<variant>]

Table 147 Adc_Config[_<varaint>]

Name	Adc_Config[_<variant>]	
Type	Adc_ConfigType	
Description	Extern declaration of root configuration structure of ADC driver which will be used during initialization.	
Verification method	The generated structure is present in Adc[_<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.	
Example(s)	**Action**	**Generated output**
	Configure HW unit0 to core0, and HW unit1 to core1 in ResourceMAllocation of resource manager. (variant unaware)	`/* Extern declaration of Adc Config Root */` `extern const Adc_ConfigType Adc_Config;`

	Configure HW unit0 to core0, and HW unit1 to core1 in ResourceMAllocation of resource manager and (variant Petrol)	`/* Extern declaration of Adc Config Root for Petrol */` `extern const Adc_ConfigType Adc_Config_Petrol;`

Revision history

Major changes since the last revision

Date	Version	Description
2023-05-23	2.0	Document Released.
2023-05-19	1.1	Documentation updated to change DEM to Productions error where applicable in sections 1.1.34, 1.1.35, 1.1.36, 1.1.37
2020-12-02	1.0	Document Released.
2020-12-01	0.1	- ADC driver chapter moved from MC-ISAR_TC3xx_Config_Verification_Manual_BASIC.pdf to this document. - Added derived configuration parameter and configuration structure member for Runtime Error Detection, EMUX and diagnostic features.

Trademarks
All referenced product or service names and trademarks are the property of their respective owners.

Edition 2023-05-23
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.