MCAL Configuration Verification Manual for Fee

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 FEE

Table of contents

1 Fee driver

This chapter describes the details of the configuration data generated by the configuration tool for FEE.

1.1 File: Fee_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 the 'inc' folder.

1.1.1 Macro: FEE_AR_RELEASE_MAJOR_VERSION

Table 1 FEE_AR_RELEASE_MAJOR_VERSION

Name	FEE_AR_RELEASE_MAJOR_VERSION	
Description	Major version number of AUTOSAR release on which the Fee 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 Fee_Cfg.h file with ArMajorVersion 4	`#define FEE_AR_RELEASE_MAJOR_VERSION (4U)`

1.1.2 Macro: FEE_AR_RELEASE_MINOR_VERSION

Table 2 FEE_AR_RELEASE_MINOR_VERSION

Name	FEE_AR_RELEASE_MINOR_VERSION	
Description	Minor version number of AUTOSAR release on which the Fee 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 Fee_Cfg.h file with ArMinorVersion 2	`#define FEE_AR_RELEASE_MINOR_VERSION (2U)`

1.1.3 Macro: FEE_AR_RELEASE_REVISION_VERSION

Table 3 FEE_AR_RELEASE_REVISION_VERSION

Name	FEE_AR_RELEASE_REVISION_VERSION
Description	Revision version number of AUTOSAR release on which the Fee 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 Fee_Cfg.h file with ArPatchVersion 2	`#define FEE_AR_RELEASE_REVISION_VERSION (2U)`

1.1.4 Macro: FEE_SW_MAJOR_VERSION

Table 4 FEE_SW_MAJOR_VERSION

Name	FEE_SW_MAJOR_VERSION	
Description	Major version number of the Fee 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 Fee_Cfg.h file with SwMajorVersion 10	`#define FEE_SW_MAJOR_VERSION (10U)`

1.1.5 Macro: FEE_SW_MINOR_VERSION

Table 5 FEE_SW_MINOR_VERSION

Name	FEE_SW_MINOR_VERSION	
Description	Minor version number of the Fee 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 Fee_Cfg.h file with SwMinorVersion 10	`#define FEE_SW_MINOR_VERSION (10U)`

1.1.6 Macro: FEE_SW_PATCH_VERSION

Table 6 FEE_SW_PATCH_VERSION

Name	FEE_SW_PATCH_VERSION
Description	Patch level version number of the Fee 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 Fee_Cfg.h file with SwPatchVersion 0	`#define FEE_SW_PATCH_VERSION (0U)`

1.1.7 Macro: FEE_SAFETY_ENABLE

Table 7 FEE_SAFETY_ENABLE

Name	FEE_SAFETY_ENABLE	
Description	Enable/disable Safety error detection and reporting.	
Verification method	The macro is generated as STD_ON if FeeSafetyEnable configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	FeeSafetyEnable = True	`#define FEE_SAFETY_ENABLE (STD_ON)`
	FeeSafetyEnable = False	`#define FEE_SAFETY_ENABLE (STD_OFF)`

1.1.8 Macro: FEE_DEV_ERROR_DETECT

Table 8 FEE_DEV_ERROR_DETECT

Name	FEE_DEV_ERROR_DETECT	
Description	Enable/disable development error detection and reporting.	
Verification method	The macro is generated as STD_ON if FeeBlockTypeConfigured is not set to 'FEE_QUASI_STATIC_DATA_ONLY' and FeeDevErrorDetect configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	FeeDevErrorDetect = True FeeBlockTypeConfigured = FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA	`#define FEE_DEV_ERROR_DETECT (STD_ON)`
	FeeDevErrorDetect = False FeeBlockTypeConfigured = FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA	`#define FEE_DEV_ERROR_DETECT (STD_OFF)`
	FeeDevErrorDetect = True FeeBlockTypeConfigured = FEE_QUASI_STATIC_DATA_ONLY	`#define FEE_DEV_ERROR_DETECT (STD_OFF)`

1.1.9 Macro: FEE_INITCHECK_API

Table 9 FEE_INITCHECK_API

Name	FEE_INITCHECK_API
Description	Enable/disable FEE Init Check API.
Verification method	The macro is generated as STD_ON if FeeInitCheckApi configuration parameter is set

	to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	FeeInitCheckApi = True	`#define FEE_INITCHECK_API (STD_ON)`
	FeeInitCheckApi = False	`#define FEE_INITCHECK_API (STD_OFF)`

1.1.10 Macro: FEE_VERSION_INFO_API

Table 10 FEE_VERSION_INFO_API

Name	FEE_VERSION_INFO_API	
Description	Enable/disable Fee_GetVersionInfo API.	
Verification method	The macro is generated as STD_ON if FeeVersionInfoApi configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	FeeVersionInfoApi = True	`#define FEE_VERSION_INFO_API (STD_ON)`
	FeeVersionInfoApi = False	`#define FEE_VERSION_INFO_API (STD_OFF)`

1.1.11 Macro: FEE_GET_CYCLE_COUNT_API

Table 11 FEE_GET_CYCLE_COUNT_API

Name	FEE_GET_CYCLE_COUNT_API	
Description	Enable/disable Fee_17_GetCycleCount API.	
Verification method	The macro is generated as STD_ON if FeeBlockTypeConfigured is not set to 'FEE_QUASI_STATIC_DATA_ONLY' and FeeGetCycleCountApi configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	FeeGetCycleCountApi = True FeeBlockTypeConfigured = FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA	`#define FEE_GET_CYCLE_COUNT_API (STD_ON)`
	FeeGetCycleCountApi = False FeeBlockTypeConfigured = FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA	`#define FEE_GET_CYCLE_COUNT_API (STD_OFF)`
	FeeGetCycleCountApi = True FeeBlockTypeConfigured = FEE_QUASI_STATIC_DATA_ONLY	`#define FEE_GET_CYCLE_COUNT_API (STD_OFF)`

1.1.12 Macro: FEE_SET_MODE_SUPPORTED

Table 12 FEE_SET_MODE_SUPPORTED

Name	FEE_SET_MODE_SUPPORTED
Description	Enable/disable Fee_SetMode API.
Verification method	The macro is generated as STD_ON if FeeSetModeSupported configuration parameter

	is set to 'True' else the macro is generated as STD_OFF. *Note:* *As Fee_SetMode API depends on Fls_SetMode API, the derived parameter expects FLS configuration parameter FlsGeneral/FlsSetModeApi to be set as 'True'.*	
Example(s)	**Action**	**Generated output**
	FeeSetModeSupported = True	`#define FEE_SET_MODE_SUPPORTED (STD_ON)`
	FeeSetModeSupported = False	`#define FEE_SET_MODE_SUPPORTED (STD_OFF)`

1.1.13 Macro: FEE_QSBLOCK_BEGIN_INDEX

Table 13 FEE_QSBLOCK_BEGIN_INDEX

Name	FEE_QSBLOCK_BEGIN_INDEX	
Description	Calculates the first index which indicates beginning of QS data block configuration, in the structure Fee_BlockConfig. All the QS data block configurations are generated consecutively after NVM data block in the structure.	
Verification method	The macro is generated as a numeric value if QS data block count is not 0. The numeric value is the count of NVM data blocks, which is index of the first QS block.	
Example(s)	**Action**	**Generated output**
	• Configure 4 FEE data blocks (FeeBlockConfiguration_0 to FeeBlockConfiguration_3) • Set FeeBlockConfiguration_3/FeeQuasiStaticManager = True, which will result in 3 NVM blocks	`#define FEE_QSBLOCK_BEGIN_INDEX (3U)`
	• Configure 4 FEE data blocks (FeeBlockConfiguration_0 to FeeBlockConfiguration_3) • Set FeeQuasiStaticManager for all the data blocks as false, which will result in 4 NVM data blocks and no QS blocks. Hence the macro is not generated	

1.1.14 Macro: FEE_DATA_BLOCK_SUPPORTED

Table 14 FEE_DATA_BLOCK_SUPPORTED

Name	FEE_DATA_BLOCK_SUPPORTED	
Description	Configure block type supported in FEE.	
Verification method	The macro is generated as a numeric value corresponding to the literal set for FeeBlockTypeConfigured configuration parameter.	
Example(s	**Action**	**Generated output**

)

FeeBlockTypeConfigured = FEE_DOUBLE_SECTOR_DATA_ONLY	`#define FEE_DATA_BLOCK_SUPPORTED FEE_DOUBLE_SECTOR_DATA_ONLY`
FeeBlockTypeConfigured = FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA	`#define FEE_DATA_BLOCK_SUPPORTED FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA`
FeeBlockTypeConfigured = FEE_QUASI_STATIC_DATA_ONLY	`#define FEE_DATA_BLOCK_SUPPORTED FEE_QUASI_STATIC_DATA_ONLY`

1.1.15 Macro: FEE_NUM_QUASI_BLOCKS

Table 15 FEE_NUM_QUASI_BLOCKS

Name	FEE_NUM_QUASI_BLOCKS	
Description	Counts the number of QS data blocks.	
Verification method	The macro is generated as numeric value if FeeBlockTypeConfigured is not set as 'FEE_DOUBLE_SECTOR_DATA_ONLY' and it corresponds to the number of QS data blocks configured.	
Example(s)	**Action**	**Generated output**
	• Configure 4 FEE data blocks (FeeBlockConfiguration_0 to FeeBlockConfiguration_3) • Set FeeBlockConfiguration_3/FeeQuasiStaticManager = True, which will result in 1 QS data block	`#define FEE_NUM_QUASI_BLOCKS (1U)`
	• Configure 4 FEE data blocks (FeeBlockConfiguration_0 to FeeBlockConfiguration_3) • Set FeeQuasiStaticManager = True for all the data blocks, which will result in 4 QS data block	`#define FEE_NUM_QUASI_BLOCKS (4U)`

1.1.16 Macro: FEE_GET_PREV_DATA_API

Table 16 FEE_GET_PREV_DATA_API

Name	FEE_GET_PREV_DATA_API	
Description	Enables/disables Fee_17_GetPrevData API.	
Verification method	The macro is generated as STD_ON if FeeBlockTypeConfigured is not set as 'FEE_QUASI_STATIC_DATA_ONLY' and FeeGetPrevDataApi configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	FeeGetPrevDataApi = True FeeBlockTypeConfigured = FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA	`#define FEE_GET_PREV_DATA_API (STD_ON)`
	FeeGetPrevDataApi = False FeeBlockTypeConfigured =	`#define FEE_GET_PREV_DATA_API (STD_OFF)`

	FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA	
	FeeGetPrevDataApi = True FeeBlockTypeConfigured = FEE_QUASI_STATIC_DATA_ONLY	`#define FEE_GET_PREV_DATA_API (STD_OFF)`

1.1.17 Macro: FEE_CANCELL_ALL_API

Table 17 FEE_CANCELL_ALL_API

Name	FEE_CANCELL_ALL_API	
Description	Enable/disable Fee_17_CancelAll API.	
Verification method	The macro is generated as STD_ON if FeeBlockTypeConfigured is not set as 'FEE_DOUBLE_SECTOR_DATA_ONLY' and FeeCancelAllApi configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	FeeCancelAllApi = True FeeBlockTypeConfigured = FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA	`#define FEE_CANCELL_ALL_API (STD_ON)`
	FeeCancelAllApi = False FeeBlockTypeConfigured = FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA	`#define FEE_CANCELL_ALL_API (STD_OFF)`
	FeeCancelAllApi = True FeeBlockTypeConfigured = FEE_DOUBLE_SECTOR_DATA_ONLY	`#define FEE_CANCELL_ALL_API (STD_OFF)`

1.1.18 Macro: FEE_MAX_BYTES_PER_CYCLE

Table 18 FEE_MAX_BYTES_PER_CYCLE

Name	FEE_MAX_BYTES_PER_CYCLE	
Description	Indicates value of maximum bytes to be processed in one main function cycle (Fee_MainFunction).	
Verification method	The macro is generated as numeric value corresponding to literal selected for FeeMaxBytesPerCycle configuration parameter.	
Example(s)	**Action**	**Generated output**
	FeeMaxBytesPerCycle = FEE_MAX_BYTES_64	`#define FEE_MAX_BYTES_PER_CYCLE (64U)`
	FeeMaxBytesPerCycle = FEE_MAX_BYTES_128	`#define FEE_MAX_BYTES_PER_CYCLE (128U)`
	FeeMaxBytesPerCycle = FEE_MAX_BYTES_256	`#define FEE_MAX_BYTES_PER_CYCLE (256U)`
	FeeMaxBytesPerCycle = FEE_MAX_BYTES_512	`#define FEE_MAX_BYTES_PER_CYCLE (512U)`

1.1.19 Macro: FEE_VIRGIN_FLASH_ILLEGAL_STATE

Table 19 FEE_VIRGIN_FLASH_ILLEGAL_STATE

Name	FEE_VIRGIN_FLASH_ILLEGAL_STATE

Description	Enables/disables the Fee to enter the illegal state when virgin flash is detected.	
Verification method	The macro is generated as STD_ON if FeeVirginFlashIllegalState configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	FeeVirginFlashIllegalState = True	`#define FEE_VIRGIN_FLASH_ILLEGAL_STATE (STD_ON)`
	FeeVirginFlashIllegalState = False	`#define FEE_VIRGIN_FLASH_ILLEGAL_STATE (STD_OFF)`

1.1.20 Macro: FEE_MAIN_FUNCTION_PERIOD

Table 20 FEE_MAIN_FUNCTION_PERIOD

Name	FEE_MAIN_FUNCTION_PERIOD	
Description	Calculates main function period.	
Verification method	This macro is generated to specify the periodicity of the main function in micro seconds. The input parameter FeeMainFunctionPeriod specifies periodicity in seconds.	
Example(s)	**Action**	**Generated output**
	FeeMainFunctionPeriod = 0.01	`#define FEE_MAIN_FUNCTION_PERIOD 10000`
	FeeMainFunctionPeriod = 1	`#define FEE_MAIN_FUNCTION_PERIOD 1000000`

1.1.21 Macro: FEE_FLS_SUPPORTS_ERASE_SUSPEND

Table 21 FEE_FLS_SUPPORTS_ERASE_SUSPEND

Name	FEE_FLS_SUPPORTS_ERASE_SUSPEND	
Description	Enable/disable erase suspend feature.	
Verification method	The macro is generated as STD_ON if FeeUseEraseSuspend configuration parameter is set to 'True' and FlsUseEraseSuspend configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	FeeUseEraseSuspend = True FlsUseEraseSuspend = True	`#define FEE_FLS_SUPPORTS_ERASE_SUSPEND (STD_ON)`
	FeeUseEraseSuspend = False FlsUseEraseSuspend = True	`#define FEE_FLS_SUPPORTS_ERASE_SUSPEND (STD_OFF)`
	FeeUseEraseSuspend = True FlsUseEraseSuspend = False	`#define FEE_FLS_SUPPORTS_ERASE_SUSPEND (STD_OFF)`

1.1.22 Macro: FEE_UNCFG_BLK_OVERFLOW_HANDLE

Table 22 FEE_UNCFG_BLK_OVERFLOW_HANDLE

Name	FEE_UNCFG_BLK_OVERFLOW_HANDLE	
Description	Sets the desired behavior of FEE (garbage collection) when too many un-configured blocks cause overflow.	
Verification method	The macro is generated as a numeric value corresponding to the literal set for FeeUnConfigBlkOverflowHandle configuration parameter.	
Example(s)	**Action**	**Generated output**
	FeeUnConfigBlkOverflowHandle = FEE_CONTINUE	`#define FEE_UNCFG_BLK_OVERFLOW_HANDLE (FEE_CONTINUE)`
	FeeUnConfigBlkOverflowHandle = FEE_STOP_AT_GC	`#define FEE_UNCFG_BLK_OVERFLOW_HANDLE (FEE_STOP_AT_GC)`

1.1.23 Macro: FEE_VIRTUAL_PAGE_SIZE

Table 23 FEE_VIRTUAL_PAGE_SIZE

Name	FEE_VIRTUAL_PAGE_SIZE	
Description	Value of virtual page size in bytes. *Note: This macro is not configurable by the user.*	
Verification method	The macro is generated as numeric value corresponding to value set for FeeVirtualPageSize configuration parameter.	
Example(s)	**Action**	**Generated output**
	FeeVirtualPageSize = 8	`#define FEE_VIRTUAL_PAGE_SIZE (8U)`

1.1.24 Macro: FEE_BLOCK_OVERHEAD

Table 24 FEE_BLOCK_OVERHEAD

Name	FEE_BLOCK_OVERHEAD	
Description	Value of block overhead in bytes. *Note: This macro is not configurable by the user.*	
Verification method	The macro is generated as numeric value corresponding to value set for FeeBlockOverhead configuration parameter.	
Example(s)	**Action**	**Generated output**
	FeeBlockOverhead = 17	`#define FEE_BLOCK_OVERHEAD (17U)`

1.1.25 Macro: FEE_PAGE_OVERHEAD

Table 25 FEE_PAGE_OVERHEAD

Name	FEE_PAGE_OVERHEAD	
Description	Value of page overhead in bytes. *Note: This macro is not configurable by the user.*	
Verification method	The macro is generated as numeric value corresponding to value set for FeePageOverhead configuration parameter.	
Example(s)	**Action**	**Generated output**
	FeeBlockOverhead = 1	#define FEE_PAGE_OVERHEAD (1U)

1.1.26 Macro: FEE_MAX_BLOCK_COUNT

Table 26 FEE_MAX_BLOCK_COUNT

Name	FEE_MAX_BLOCK_COUNT	
Description	Total number of blocks configured.	
Verification method	The macro is generated as numeric value corresponding to value set for FeeMaxBlockCount configuration parameter and number of data block configured. The greater numeric value will be generated.	
Example(s)	**Action**	**Generated output**
	FeeMaxBlockCount = 1 Configured data blocks are 4	#define FEE_E_GC_INIT (DemConf_DemEventParameter_DemEventParameter_0)
	FeeMaxBlockCount = 20 Configured data blocks are 4	#define FEE_E_GC_INIT (DemConf_DemEventParameter_DemEventParameter_0)

1.1.27 Macro: FeeConf_FeeBlockConfiguration_<block name>

Table 27 FeeConf_FeeBlockConfiguration_<block name>

Name	FeeConf_FeeBlockConfiguration_<block name>	
Description	The macro is the symbolic name generated for the configuration parameter 'FeeBlockConfiguration/<block name>/FeeBlockNumber'.	
Verification method	The macro is the symbolic name generated for the configuration parameter 'FeeBlockConfiguration/<block name>/FeeBlockNumber'.	
Example(s)	**Action**	**Generated output**
	• Configure 4 FEE blocks • Container for FeeBlockNumber 1 is named FeeBlockConfiguration_0	#define FeeConf_FeeBlockConfiguration_FeeBlockConfiguration_0 ((uint16)1) #define FeeConf_FeeBlockConfiguration_FeeBlockConfiguration_1 ((uint16)2) #define

	• Container for FeeBlockNumber 2 is named FeeBlockConfiguration_1 • Container for FeeBlockNumber 3 is named FeeBlockConfiguration_2 • Container for FeeBlockNumber 4 is named FeeBlockConfiguration_3	`FeeConf_FeeBlockConfiguration_FeeBlockConfiguration_2 ((uint16)3)` `#define FeeConf_FeeBlockConfiguration_FeeBlockConfiguration_3 ((uint16)4)`

1.1.28 Macro: FEE_GC_INIT_DEM_REPORT

Table 28 FEE_GC_INIT_DEM_REPORT

Name	FEE_GC_INIT_DEM_REPORT	
Description	Enable/disable FEE_E_GC_INIT DEM error code.	
Verification method	The macro is generated as FEE_ENABLE_DEM_REPORT if FeeDemEventParameterRefs configuration container is configured and if FEE_E_GC_INIT container is set to a valid node with a valid value in DEM configuration module else generated as FEE_DISABLE_DEM_REPORT.	
Example(s)	**Action**	**Generated output**
	• Configure a DEM node Dem/DemConfigSet/DemEventParameter/DemEventParameter_0 • Assign the configured DEM node reference to FeeDemEventParameterRefs/FeeDemEventParameterRefs_0/FEE_E_GC_INIT	`#define FEE_GC_INIT_DEM_REPORT (FEE_ENABLE_DEM_REPORT)`
	• FeeDemEventParameterRefs/FeeDemEventParameterRefs_0 is not configured	`#define FEE_GC_INIT_DEM_REPORT (FEE_DISABLE_DEM_REPORT)`
	• Configure FeeDemEventParameterRefs/FeeDemEventParameterRefs_0, no valid DEM node reference given to the FEE_E_GC_INIT container	`#define FEE_GC_INIT_DEM_REPORT (FEE_DISABLE_DEM_REPORT)`

1.1.29 Macro: FEE_E_GC_INIT

Table 29 FEE_E_GC_INIT

Name	FEE_E_GC_INIT	
Description	Denotes value referred for the FEE_E_GC_INIT DEM error code.	
Verification method	The macro is generated as DemConf_DemEventParameter_<Dem node name> if FeeDemEventParameterRefs configuration container is configured and if FEE_E_GC_INIT container is set to a valid node with a valid value in DEM configuration module else the macro is not generated.	
Example(s)	**Action**	**Generated output**
	• Configure a DEM node Dem/DemConfigSet/DemEventParameter/DemEventParameter_0 • Assign the configured DEM node reference to FeeDemEventParameterRefs/FeeDemEventParameterRefs_0/ FEE_E_GC_INIT	`#define FEE_E_GC_INIT (DemConf_DemEventParameter_DemEventParameter_0)`
	• Configure FeeDemEventParameterRefs/FeeDemEventParameterRefs_0, no valid DEM node reference given to the FEE_E_GC_INIT container *Note: Macro not generated*	
	• FeeDemEventParameterRefs/FeeDemEventParameterRefs_0 is not configured *Note: Macro not generated*	

1.1.30 Macro: FEE_WRITE_DEM_REPORT

Table 30 FEE_GC_INIT_DEM_REPORT

Name	FEE_WRITE_DEM_REPORT	
Description	Enable/disable FEE_E_WRITE DEM error code.	
Verification method	The macro is generated as FEE_ENABLE_DEM_REPORT if FeeDemEventParameterRefs configuration container is configured and if FEE_E_WRITE container is set to a valid node with a valid value in DEM configuration module else generated as FEE_DISABLE_DEM_REPORT.	
Example(s)	**Action**	**Generated output**
	• Configure a DEM node Dem/DemConfigSet/DemEventParameter/DemEventParameter_0	`#define FEE_WRITE_DEM_REPORT (FEE_ENABLE_DEM_REPORT)`

• Assign the configured DEM node reference to FeeDemEventParameterRefs/FeeDemEventParameterRefs_0/FEE_E_WRITE	
• FeeDemEventParameterRefs/FeeDemEventParameterRefs_0 is not configured	`#define FEE_WRITE_DEM_REPORT (FEE_DISABLE_DEM_REPORT)`
• Configure FeeDemEventParameterRefs/FeeDemEventParameterRefs_0, no valid DEM node reference given to the FEE_E_WRITE container	`#define FEE_WRITE_DEM_REPORT (FEE_DISABLE_DEM_REPORT)`

1.1.31 Macro: FEE_E_WRITE

Table 31 FEE_E_WRITE

Name	FEE_E_WRITE	
Description	Denotes value referred for the FEE_E_WRITE DEM error code.	
Verification method	The macro is generated as DemConf_DemEventParameter_<Dem node name> if FeeDemEventParameterRefs configuration container is configured and if FEE_E_WRITE container is set to a valid node with a valid value in DEM configuration module else the macro is not generated.	
Example(s)	**Action**	**Generated output**
	• Configure a DEM node Dem/DemConfigSet/DemEventParameter/DemEventParameter_0 • Assign the configured DEM node reference to FeeDemEventParameterRefs/FeeDemEventParameterRefs_0/FEE_E_WRITE	`#define FEE_E_WRITE (DemConf_DemEventParameter_DemEventParameter_0)`
	• Configure FeeDemEventParameterRefs/FeeDemEventParameterRefs_0, no valid DEM node reference given to the FEE_E_WRITE container *Note: Macro not generated*	
	• FeeDemEventParameterRefs/FeeDemEventParameterRefs_0 is not configured *Note: Macro not generated*	

1.1.32 Macro: FEE_READ_DEM_REPORT

Table 32 FEE_READ_DEM_REPORT

Name	FEE_READ_DEM_REPORT	
Description	Enable/disable FEE_E_READ DEM error code.	
Verification method	The macro is generated as FEE_ENABLE_DEM_REPORT if FeeDemEventParameterRefs configuration container is configured and if FEE_E_READ container is set to a valid node with a valid value in DEM configuration module else generated as FEE_DISABLE_DEM_REPORT.	
Example(s)	**Action**	**Generated output**
	• Configure a DEM node Dem/DemConfigSet/DemEventParameter/DemEventParameter_0 • Assign the configured DEM node reference to FeeDemEventParameterRefs/FeeDemEventParameterRefs_0/FEE_READ	`#define FEE_READ_DEM_REPORT (FEE_ENABLE_DEM_REPORT)`
	• FeeDemEventParameterRefs/FeeDemEventParameterRefs_0 is not configured	`#define FEE_READ_DEM_REPORT (FEE_DISABLE_DEM_REPORT)`
	• Configure FeeDemEventParameterRefs/FeeDemEventParameterRefs_0, no valid DEM node reference given to the FEE_E_READ container	`#define FEE_READ_DEM_REPORT (FEE_DISABLE_DEM_REPORT)`

1.1.33 Macro: FEE_E_READ

Table 33 FEE_E_READ

Name	FEE_E_READ	
Description	Denotes value referred for the FEE_E_READ DEM error code.	
Verification method	The macro is generated as DemConf_DemEventParameter_<Dem node name> if FeeDemEventParameterRefs configuration container is configured and if FEE_E_READ container is set to a valid node with a valid value in DEM configuration module else the macro is not generated.	
Example(s)	**Action**	**Generated output**
	• Configure a DEM node Dem/DemConfigSet/DemEventParameter/DemEventParameter_0 • Assign the configured DEM node reference to FeeDemEventParameterRefs/FeeDemEventParameterRefs_0/FEE_E_READ	`#define FEE_E_READ (DemConf_DemEventParameter_DemEventParameter_0)`

	• Configure FeeDemEventParameterRefs/FeeDemEventParameterRefs_0, no valid DEM node reference given to the FEE_E_READ container *Note: Macro not generated*	
	• FeeDemEventParameterRefs/FeeDemEventParameterRefs_0 is not configured *Note: Macro not generated*	

1.1.34 Macro: FEE_GC_WRITE_DEM_REPORT

Table 34 FEE_GC_WRITE_DEM_REPORT

Name	FEE_GC_WRITE_DEM_REPORT	
Description	Enable/disable FEE_E_GC_WRITE DEM error code.	
Verification method	The macro is generated as FEE_ENABLE_DEM_REPORT if FeeDemEventParameterRefs configuration container is configured and if FEE_E_GC_WRITE container is set to a valid node with a valid value in DEM configuration module else generated as FEE_DISABLE_DEM_REPORT.	
Example(s)	**Action**	**Generated output**
	• Configure a DEM node Dem/DemConfigSet/DemEventParameter/DemEventParameter_0 • Assign the configured DEM node reference to FeeDemEventParameterRefs/FeeDemEventParameterRefs_0/FEE_E_GC_WRITE	`#define FEE_GC_WRITE_DEM_REPORT (FEE_ENABLE_DEM_REPORT)`
	• FeeDemEventParameterRefs/FeeDemEventParameterRefs_0 is not configured	`#define FEE_GC_WRITE_DEM_REPORT (FEE_DISABLE_DEM_REPORT)`
	• Configure FeeDemEventParameterRefs/FeeDemEventParameterRefs_0, no valid DEM node reference given to the FEE_E_GC_WRITE container	`#define FEE_GC_WRITE_DEM_REPORT (FEE_DISABLE_DEM_REPORT)`

1.1.35 Macro: FEE_E_GC_WRITE

Table 35 FEE_E_GC_WRITE

Name	FEE_E_GC_WRITE

Description	Denotes value referred for the FEE_E_GC_WRITE DEM error code.	
Verification method	The macro is generated as DemConf_DemEventParameter_<Dem node name> if FeeDemEventParameterRefs configuration container is configured and if FEE_E_GC_WRITE container is set to a valid node with a valid value in DEM configuration module else the macro is not generated.	
Example(s)	**Action**	**Generated output**
	• Configure a DEM node Dem/DemConfigSet/DemEventParameter/DemEventParameter_0 • Assign the configured DEM node reference to FeeDemEventParameterRefs/FeeDemEventParameterRefs_0/ FEE_E_GC_WRITE	`#define FEE_E_GC_WRITE (DemConf_DemEventParameter_DemEventParameter_0)`
	• Configure FeeDemEventParameterRefs/FeeDemEventParameterRefs_0, no valid DEM node reference given to the FEE_E_GC_WRITE container *Note: Macro not generated*	
	• FeeDemEventParameterRefs/FeeDemEventParameterRefs_0 is not configured *Note: Macro not generated*	

1.1.36 Macro: FEE_GC_READ_DEM_REPORT

Table 36 FEE_GC_READ_DEM_REPORT

Name	FEE_GC_READ_DEM_REPORT	
Description	Enable/disable FEE_E_GC_READ DEM error code.	
Verification method	The macro is generated as FEE_ENABLE_DEM_REPORT if FeeDemEventParameterRefs configuration container is configured and if FEE_E_GC_READ container is set to a valid node with a valid value in DEM configuration module else generated as FEE_DISABLE_DEM_REPORT.	
Example(s)	**Action**	**Generated output**
	• Configure a DEM node Dem/DemConfigSet/DemEventParameter/DemEventParameter_0 • Assign the configured DEM node reference to FeeDemEventParameterRefs/FeeDemEventParameterRefs_0/FEE_E_GC_READ	`#define FEE_GC_READ_DEM_REPORT (FEE_ENABLE_DEM_REPORT)`

	• FeeDemEventParameterRefs/FeeDemEventParameterRefs_0 is not configured	`#define FEE_GC_READ_DEM_REPORT (FEE_DISABLE_DEM_REPORT)`
	• Configure FeeDemEventParameterRefs/FeeDemEventParameterRefs_0, no valid DEM node reference given to the FEE_E_GC_READ container	`#define FEE_GC_READ_DEM_REPORT (FEE_DISABLE_DEM_REPORT)`

1.1.37 Macro: FEE_E_GC_READ

Table 37 FEE_E_GC_READ

Name	FEE_E_GC_READ	
Description	Denotes value referred for the FEE_E_GC_READ DEM error code.	
Verification method	The macro is generated as DemConf_DemEventParameter_<Dem node name> if FeeDemEventParameterRefs configuration container is configured and if FEE_E_GC_READ container is set to a valid node with a valid value in DEM configuration module else the macro is not generated.	
Example(s)	**Action**	**Generated output**
	• Configure a DEM node Dem/DemConfigSet/DemEventParameter/DemEventParameter_0 • Assign the configured DEM node reference to FeeDemEventParameterRefs/FeeDemEventParameterRefs_0/ FEE_E_GC_READ	`#define FEE_E_GC_READ (DemConf_DemEventParameter_DemEventParameter_0)`
	• Configure FeeDemEventParameterRefs/FeeDemEventParameterRefs_0, no valid DEM node reference given to the FEE_E_GC_READ container *Note: Macro not generated*	
	• FeeDemEventParameterRefs/FeeDemEventParameterRefs_0 is not configured *Note: Macro not generated*	

1.1.38 Macro: FEE_GC_ERASE_DEM_REPORT

Table 38 FEE_GC_ERASE_DEM_REPORT

Name	FEE_GC_ERASE_DEM_REPORT	
Description	Enable/disable FEE_E_GC_ERASE DEM error code.	
Verification method	The macro is generated as FEE_ENABLE_DEM_REPORT if FeeDemEventParameterRefs configuration container is configured and if FEE_E_GC_ERASE container is set to a valid node with a valid value in DEM configuration module else generated as FEE_DISABLE_DEM_REPORT.	
Example(s)	**Action**	**Generated output**
	• Configure a DEM node Dem/DemConfigSet/DemEventParameter/DemEventParameter_0 • Assign the configured DEM node reference to FeeDemEventParameterRefs/FeeDemEventParameterRefs_0/FEE_E_GC_ERASE	`#define FEE_GC_ERASE_DEM_REPORT (FEE_ENABLE_DEM_REPORT)`
	• FeeDemEventParameterRefs/FeeDemEventParameterRefs_0 is not configured	`#define FEE_GC_ERASE_DEM_REPORT (FEE_DISABLE_DEM_REPORT)`
	• Configure FeeDemEventParameterRefs/FeeDemEventParameterRefs_0, no valid DEM node reference given to the FEE_E_GC_ERASE container	`#define FEE_GC_ERASE_DEM_REPORT (FEE_DISABLE_DEM_REPORT)`

1.1.39 Macro: FEE_E_GC_ERASE

Table 39 FEE_E_GC_ERASE

Name	FEE_E_GC_ERASE	
Description	Denotes value referred for the FEE_E_GC_ERASE DEM error code.	
Verification method	The macro is generated as DemConf_DemEventParameter_<Dem node name> if FeeDemEventParameterRefs configuration container is configured and if FEE_E_GC_ERASE container is set to a valid node with a valid value in DEM configuration module else the macro is not generated.	
Example(s)	**Action**	**Generated output**
	• Configure a DEM node Dem/DemConfigSet/DemEventParameter/DemEventParameter_0 • Assign the configured DEM node reference to FeeDemEventParameterRefs/FeeDemEventParameterRefs_0/ FEE_E_GC_ERASE	`#define FEE_E_GC_ERASE (DemConf_DemEventParameter_DemEventParameter_0)`
	• Configure FeeDemEventParameterRefs/FeeDemEventP	

	arameterRefs_0, no valid DEM node reference given to the FEE_E_GC_ERASE container *Note: Macro not generated*	
	• FeeDemEventParameterRefs/FeeDemEventParameterRefs_0 is not configured *Note: Macro not generated*	

1.1.40 Macro: FEE_INVALIDATE_DEM_REPORT

Table 40 FEE_INVALIDATE_DEM_REPORT

Name	FEE_INVALIDATE_DEM_REPORT	
Description	Enable/disable FEE_E_INVALIDATE DEM error code.	
Verification method	The macro is generated as FEE_ENABLE_DEM_REPORT if FeeDemEventParameterRefs configuration container is configured and if FEE_E_INVALIDATE container is set to a valid node with a valid value in DEM configuration module else generated as FEE_DISABLE_DEM_REPORT.	
Example(s)	**Action**	**Generated output**
	• Configure a DEM node Dem/DemConfigSet/DemEventParameter/DemEventParameter_0 • Assign the configured DEM node reference to FeeDemEventParameterRefs/FeeDemEventParameterRefs_0/FEE_E_INVALIDATE	`#define FEE_INVALIDATE_DEM_REPORT (FEE_ENABLE_DEM_REPORT)`
	• FeeDemEventParameterRefs/FeeDemEventParameterRefs_0 is not configured	`#define FEE_INVALIDATE_DEM_REPORT (FEE_DISABLE_DEM_REPORT)`
	• Configure FeeDemEventParameterRefs/FeeDemEventParameterRefs_0, no valid DEM node reference given to the FEE_E_INVALIDATE container	`#define FEE_INVALIDATE_DEM_REPORT (FEE_DISABLE_DEM_REPORT)`

1.1.41 Macro: FEE_E_INVALIDATE

Table 41 FEE_E_INVALIDATE

Name	FEE_E_INVALIDATE
Description	Denotes value referred for the FEE_E_INVALIDATE DEM error code.

Verification method	The macro is generated as DemConf_DemEventParameter_<Dem node name> if FeeDemEventParameterRefs configuration container is configured and if FEE_E_INVALIDATE container is set to a valid node with a valid value in DEM configuration module else the macro is not generated.	
Example(s)	**Action**	**Generated output**
	• Configure a DEM node Dem/DemConfigSet/DemEventParameter/DemEventParameter_0 • Assign the configured DEM node reference to FeeDemEventParameterRefs/FeeDemEventParameterRefs_0/FEE_E_INVALIDATE	`#define FEE_E_INVALIDATE (DemConf_DemEventParameter_DemEventParameter_0)`
	• Configure FeeDemEventParameterRefs/FeeDemEventParameterRefs_0, no valid DEM node reference given to the FEE_E_INVALIDATE container *Note: Macro not generated*	
	• FeeDemEventParameterRefs/FeeDemEventParameterRefs_0 is not configured *Note: Macro not generated*	

1.1.42 Macro: FEE_WRITE_CYCLES_DEM_REPORT

Table 42 FEE_WRITE_CYCLES_DEM_REPORT

Name	FEE_WRITE_CYCLES_DEM_REPORT	
Description	Enable/disable FEE_E_WRITE_CYCLES_EXHAUSTED DEM error code.	
Verification method	The macro is generated as FEE_ENABLE_DEM_REPORT if FeeDemEventParameterRefs configuration container is configured and if FEE_E_WRITE_CYCLES_EXHAUSTED container is set to a valid node with a valid value in DEM configuration module else generated as FEE_DISABLE_DEM_REPORT.	
Example(s)	**Action**	**Generated output**
	• Configure a DEM node Dem/DemConfigSet/DemEventParameter/DemEventParameter_0 • Assign the configured DEM node reference to FeeDemEventParameterRefs/FeeDemEventParameterRefs_0/ FEE_E_WRITE_CYCLES_EXHAUSTED	`#define FEE_WRITE_CYCLES_DEM_REPORT (FEE_ENABLE_DEM_REPORT)`
	• FeeDemEventParameterRefs/FeeDemEventParameterRefs_0 is not configured	`#define FEE_WRITE_CYCLES_DEM_REPO`

		RT (FEE_DISABLE_DEM_REPORT)
	• Configure FeeDemEventParameterRefs/FeeDemEventParameterRefs_0, no valid DEM node reference given to the FEE_E_WRITE_CYCLES_EXHAUSTED container	#define FEE_WRITE_CYCLES_DEM_REPORT (FEE_DISABLE_DEM_REPORT)

1.1.43 Macro: FEE_E_WRITE_CYCLES_EXHAUSTED

Table 43 FEE_E_WRITE_CYCLES_EXHAUSTED

Name	FEE_E_WRITE_CYCLES_EXHAUSTED	
Description	Denotes value referred for the FEE_E_WRITE_CYCLES_EXHAUSTED DEM error code.	
Verification method	The macro is generated as DemConf_DemEventParameter_<Dem node name> if FeeDemEventParameterRefs configuration container is configured and if FEE_E_WRITE_CYCLES_EXHAUSTED container is set to a valid node with a valid value in DEM configuration module else the macro is not generated.	
Example(s)	**Action**	**Generated output**
	• Configure a DEM node Dem/DemConfigSet/DemEventParameter/DemEventParameter_0 • Assign the configured DEM node reference to FeeDemEventParameterRefs/FeeDemEventParameterRefs_0/FEE_E_WRITE_CYCLES_EXHAUSTED	#define FEE_E_WRITE_CYCLES_EXHAUSTED (DemConf_DemEventParameter_DemEventParameter_0)
	• Configure FeeDemEventParameterRefs/FeeDemEventParameterRefs_0, no valid DEM node reference given to the FEE_E_WRITE_CYCLES_EXHAUSTED container *Note: Macro not generated*	
	• FeeDemEventParameterRefs/FeeDemEventParameterRefs_0 is not configured *Note: Macro not generated*	

1.1.44 Macro: FEE_GC_TRIG_DEM_REPORT

Table 44 FEE_GC_TRIG_DEM_REPORT

Name	FEE_GC_TRIG_DEM_REPORT
Description	Enable/disable FEE_E_GC_TRIG DEM error code.

Verification method	The macro is generated as FEE_ENABLE_DEM_REPORT if FeeDemEventParameterRefs configuration container is configured and if FEE_E_GC_TRIG container is set to a valid node with a valid value in DEM configuration module else generated as FEE_DISABLE_DEM_REPORT.	
Example(s)	**Action**	**Generated output**
	• Configure a DEM node Dem/DemConfigSet/DemEventParameter/DemEventParameter_0 • Assign the configured DEM node reference to FeeDemEventParameterRefs/FeeDemEventParameterRefs_0/FEE_E_GC_TRIG	`#define FEE_GC_TRIG_DEM_REPORT (FEE_ENABLE_DEM_REPORT)`
	• FeeDemEventParameterRefs/FeeDemEventParameterRefs_0 is not configured	`#define FEE_GC_TRIG_DEM_REPORT (FEE_DISABLE_DEM_REPORT)`
	• Configure FeeDemEventParameterRefs/FeeDemEventParameterRefs_0, no valid DEM node reference given to the FEE_E_GC_TRIG container	`#define FEE_GC_TRIG_DEM_REPORT (FEE_DISABLE_DEM_REPORT)`

1.1.45 Macro: FEE_E_GC_TRIG

Table 45 FEE_E_GC_TRIG

Name	FEE_E_GC_TRIG	
Description	Denotes value referred for the FEE_E_GC_TRIG DEM error code.	
Verification method	The macro is generated as DemConf_DemEventParameter_<Dem node name> if FeeDemEventParameterRefs configuration container is configured and if FEE_E_GC_TRIG container is set to a valid node with a valid value in DEM configuration module else the macro is not generated.	
Example(s)	**Action**	**Generated output**
	• Configure a DEM node Dem/DemConfigSet/DemEventParameter/DemEventParameter_0 • Assign the configured DEM node reference to FeeDemEventParameterRefs/FeeDemEventParameterRefs_0/FEE_E_GC_TRIG	`#define FEE_E_GC_TRIG (DemConf_DemEventParameter_DemEventParameter_0)`
	• Configure FeeDemEventParameterRefs/FeeDemEventParameterRefs_0, no valid DEM node reference given to the FEE_E_GC_TRIG container *Note: Macro not generated*	

	• FeeDemEventParameterRefs/FeeDemEventParameterRefs_0 is not configured *Note: Macro not generated*	

1.1.46 Macro: FEE_UNCFG_BLK_DEM_REPORT

Table 46 FEE_UNCFG_BLK_DEM_REPORT

Name	FEE_UNCFG_BLK_DEM_REPORT	
Description	Enable/disable FEE_E_UNCONFIG_BLK_EXCEEDED DEM error code.	
Verification method	The macro is generated as FEE_ENABLE_DEM_REPORT if FeeDemEventParameterRefs configuration container is configured and if FEE_E_UNCONFIG_BLK_EXCEEDED container is set to a valid node with a valid value in DEM configuration module else generated as FEE_DISABLE_DEM_REPORT.	
Example(s)	**Action**	**Generated output**
	• Configure a DEM node Dem/DemConfigSet/DemEventParameter/DemEventParameter_0 • Assign the configured DEM node reference to FeeDemEventParameterRefs/FeeDemEventParameterRefs_0/FEE_E_UNCONFIG_BLK_EXCEEDED	`#define FEE_UNCFG_BLK_DEM_REPORT (FEE_ENABLE_DEM_REPORT)`
	• FeeDemEventParameterRefs/FeeDemEventParameterRefs_0 is not configured	`#define FEE_UNCFG_BLK_DEM_REPORT (FEE_DISABLE_DEM_REPORT)`
	• Configure FeeDemEventParameterRefs/FeeDemEventParameterRefs_0, no valid DEM node reference given to the FEE_E_UNCONFIG_BLK_EXCEEDED container	`#define FEE_UNCFG_BLK_DEM_REPORT (FEE_DISABLE_DEM_REPORT)`

1.1.47 Macro: FEE_E_UNCONFIG_BLK_EXCEEDED

Table 47 FEE_E_UNCONFIG_BLK_EXCEEDED

Name	FEE_E_UNCONFIG_BLK_EXCEEDED	
Description	Denotes value referred for the FEE_E_UNCONFIG_BLK_EXCEEDED DEM error code.	
Verification method	The macro is generated as DemConf_DemEventParameter_<Dem node name> if FeeDemEventParameterRefs configuration container is configured and if FEE_E_UNCONFIG_BLK_EXCEEDED container is set to a valid node with a valid value in DEM configuration module else the macro is not generated.	
Examp	**Action**	**Generated output**

le(s)	• Configure a DEM node Dem/DemConfigSet/DemEventParameter/DemEventParameter_0 • Assign the configured DEM node reference to FeeDemEventParameterRefs/FeeDemEventParameterRefs_0/FEE_E_UNCONFIG_BLK_EXCEEDED	`#define FEE_E_UNCONFIG_BLK_EXCEEDED (DemConf_DemEventParameter_DemEventParameter_0)`
	• Configure FeeDemEventParameterRefs/FeeDemEventParameterRefs_0, no valid DEM node reference given to the FEE_E_UNCONFIG_BLK_EXCEEDED container *Note: Macro not generated*	
	• FeeDemEventParameterRefs/FeeDemEventParameterRefs_0 is not configured *Note: Macro not generated*	

1.1.48 Macro: FEE_DEM_ENABLED

Table 48 FEE_DEM_ENABLED

Name	FEE_DEM_ENABLED	
Description	Enable/disable DEM error reporting.	
Verification method	The macro is generated as STD_ON if FeeBlockTypeConfigured is not set to 'FEE_QUASI_STATIC_DATA_ONLY' and if FeeDemEventParameterRefs configuration container is configured and if any one of the DEM error code container is set to a valid node with a valid value in DEM configuration module else generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	• Configure a DEM node Dem/DemConfigSet/DemEventParameter/DemEventParameter_0 • Assign the configured DEM node reference to FeeDemEventParameterRefs/FeeDemEventParameterRefs_0/FEE_E_GC_WRITE	`#define FEE_DEM_ENABLED (STD_ON)`
	• Configure a DEM node Dem/DemConfigSet/DemEventParameter/DemEventParameter_0 • Don't assign the configured DEM node reference to FeeDemEventParameterRefs/FeeDemEventParameterRefs_0/FEE_E_GC_WRITE	`#define FEE_DEM_ENABLED (STD_OFF)`

1.1.49 Macro: FEE_GET_ECC_ERROR_INFO_API

Table 49 FEE_GET_ECC_ERROR_INFO_API

Name	FEE_GET_ECC_ERROR_INFO_API	
Description	Enable/disable Fee_17_GetEccErrorInfo API.	
Verification method	The macro is generated as STD_ON if FeeEccErrorInfoApi configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	FeeEccErrorInfoApi = True	`#define FEE_GET_ECC_ERROR_INFO_API (STD_ON)`
	FeeEccErrorInfoApi = False	`#define FEE_GET_ECC_ERROR_INFO_API (STD_OFF)`

1.1.50 Macro: FEE_RUNTIME_ERROR_DETECT

Table 50 FEE_RUNTIME_ERROR_DETECT

Name	FEE_RUNTIME_ERROR_DETECT	
Description	Enable/disable runtime error reporting.	
Verification method	The macro is generated as STD_ON if AUTOSAR version is 440 and if FeeRunTimeErrorDetect configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	• AUTOSAR version is 440 • FeeRunTimeErrorDetect = True	`#define FEE_RUNTIME_ERROR_DETECT (STD_ON)`
	• AUTOSAR version is 440 • FeeRunTimeErrorDetect = False	`#define FEE_RUNTIME_ERROR_DETECT (STD_OFF)`

1.1.51 Macro: FEE_PAGES_PER_FEEMAIN

Table 51 FEE_PAGES_PER_FEEMAIN

Name	FEE_PAGES_PER_FEEMAIN	
Description	Number of pages to be processed in one main cylcle for cache table initialization.	
Verification method	This macro is generated to specify the number of pages to be processed in each main cycle for cache table initialization. If 0 is configured then all the blocks shall to be processed in 1 main cycle.	
Example(s)	**Action**	**Generated output**
	• Configure 2 or more data block. • Configure FeeBlocksScannedPerCycle to 2.	`#define FEE_PAGES_PER_FEEMAIN (4U)`
	• Configue 2 data block data block. • Configure FeeBlocksScannedPerCycle to 0.	`#define FEE_PAGES_PER_FEEMAIN (65535U)`

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

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

1.2.1 Structure: Fee_Config[_<variant>]

Table 52 Fee_Config[_<variant>]

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

	Configure 3 FEE data block with no variant support	`const Fee_ConfigType Fee_Config =`
		`{`
		`  /* Fee State Data Structure */`
		`  &Fee_StateVar,`
		`  /* Pointer to logical block configurations */`
		`  &Fee_BlockConfig[0], #endif`
		`  /* Fee Job end notification API */`
		`  (Fee_NotifFunctionPtrType)NULL_PTR,`
		`  /* Fee Job error notification API */`
		`  (Fee_NotifFunctionPtrType)NULL_PTR,`
		`  /* Fee QS Job end notification API */`
		`  (Fee_NotifFunctionPtrType)NULL_PTR,`
		`  /* Fee QS Job error notification API */`
		`  (Fee_NotifFunctionPtrType)NULL_PTR,`
		`  /* Fee threshold value */`
		`  200U,`
		`  /* Number of blocks configured */`
		`  3U,`
		`  {`
		`   /* Keep the unconfigured blocks */`
		`   FEE_UNCONFIG_BLOCK_KEEP,`
		`   /* Restart Garbage Collection when`

	user job is requested */ FEE_GC_RESTART_WRITE, /* Erase Suspend feature is enabled */ FEE_ERASE_SUSPEND_ENABLED, /* Reserved */ 0U }, /* Fee NVM Illegal State notification */ (Fee_NotifFunctionPtrType)NULL_PTR, /* Fee NVM Illegal State notification */ (Fee_NotifFunctionPtrType)NULL_PTR, /* QS Hardening Error notification */ (Fee_NotifFunctionPtrType)NULL_PTR, /* Erase All feature is disabled */ (boolean)FALSE };
Configure 3 FEE data blocks with variant support. Set the name of the variant 'Petrol'	const Fee_ConfigType Fee_Config_Petrol = { /* Fee State Data Structure */ &Fee_StateVar, /* Pointer to logical block configurations */ &Fee_BlockConfig_Petrol[0], #endif /* Fee Job end notification API */ (Fee_NotifFunctionPtrType)NULL_PTR, /* Fee Job error notification API */ (Fee_NotifFunctionPtrType)NULL_PTR, /* Fee QS Job end notification API */ (Fee_NotifFunctionPtrType)NULL_PTR, /* Fee QS Job error notification API */ (Fee_NotifFunctionPtrType)NULL_PTR, /* Fee threshold value */ 200U,

```
  /* Number of blocks configured */
  3U,
  {
   /* Keep the unconfigured blocks */
   FEE_UNCONFIG_BLOCK_KEEP,
   /* Restart Garbage Collection when
user job is requested */
   FEE_GC_RESTART_WRITE,

    /* Erase Suspend feature is
enabled */
    FEE_ERASE_SUSPEND_ENABLED,
    /* Reserved */
    0U
   },
  /* Fee NVM Illegal State
notification */
  (Fee_NotifFunctionPtrType)NULL_PTR,
  /* Fee NVM Illegal State
notification */
  (Fee_NotifFunctionPtrType)NULL_PTR,
  /* QS Hardening Error notification
*/
  (Fee_NotifFunctionPtrType)NULL_PTR,
  /* Erase All feature is disabled */
  (boolean)FALSE
};
```

1.2.1.1 Member: FeeStatePtr

Table 53 FeeStatePtr

Name	FeeStatePtr	
Type	Fee_StateDataType *	
Description	Pointer to the state variable data structure.	
Verification method	The generated structure member is present in the Fee_Config[_<variant>] structure. The name set for FeeStateVarStructure configuration parameter is generated.	
Example(s)	**Action**	**Generated output**
	Set FeeStateVarStructure = Fee_StateVar1	`&Fee_StateVar1,`
	Set FeeStateVarStructure =	`&Fee_StateVar2,`

	Fee_StateVar2	
	Set FeeStateVarStructure = PetrolVersion	`&PetrolVersion,`

1.2.1.2 Member: FeeBlockConfigPtr

Table 54 FeeBlockConfigPtr

Name	FeeBlockConfigPtr	
Type	Fee_BlockType *	
Description	Pointer to logical block configurations.	
Verification method	The generated structure member is present in the Fee_Config[_<variant>] structure. The member is generated as variable address pointing to Fee_Block[_<variant>] structure.	
Example(s)	**Action**	**Generated output**
	Configure 3 FEE data block with no variant support	`/* Pointer to logical block configurations */` `&Fee_BlockConfig[0],`
	Configure 3 FEE data blocks with variant support. Set the name of the variant 'Petrol'	`/* Pointer to logical block configurations */` `&Fee_BlockConfig_Petrol[0],`

1.2.1.3 Member: FeeNvmJobEndNotification

Table 55 FeeNvmJobEndNotification

Name	FeeNvmJobEndNotification	
Type	Fee_NotifFunctionPtrType	
Description	Job end notification for call back routine.	
Verification method	The generated structure member is present in the Fee_Config[_<variant>] structure. The member is generated as <call back routine name> if name is set in the container FeeNvmJobEndNotification else generated as NULL_PTR. If FeeBlockTypeConfigured = 'FEE_QUASI_STATIC_DATA_ONLY' then the member is not generated.	
Example(s)	**Action**	**Generated output**
	• Set FeeNvmJobEndNotification container node name as 'Nvm_EndNotif' • Set FeeBlockTypeConfigured = FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA	`(Fee_NotifFunctionPtrType)Nvm_EndNotif,`
	• Set FeeNvmJobEndNotification container node name as 'NULL_PTR' • Set FeeBlockTypeConfigured = FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA	`(Fee_NotifFunctionPtrType) NULL_PTR,`

	• Set FeeNvmJobEndNotification container node name as ‘’(empty) • Set FeeBlockTypeConfigured = FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA	`(Fee_NotifFunctionPtrType) NULL_PTR,`

1.2.1.4 Member: FeeNvmJobErrorNotification

Table 56 FeeNvmJobErrorNotification

Name	FeeNvmJobErrorNotification	
Type	Fee_NotifFunctionPtrType	
Description	Job error notification call back routine.	
Verification method	The generated structure member is present in the Fee_Config[_<variant>] structure. The member is generated as <call back routine name> if name is set in the container FeeNvmJobErrorNotification else generated as NULL_PTR. If FeeBlockTypeConfigured = ‘FEE_QUASI_STATIC_DATA_ONLY’ then the member is not generated.	
Example(s)	**Action**	**Generated output**
	• Set FeeNvmJobErrorNotification container node name as ‘Nvm_ErrorNotif’ • Set FeeBlockTypeConfigured = FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA	`(Fee_NotifFunctionPtrType)Nvm_ErrorNotif,`
	• Set FeeNvmJobErrorNotification container node name as ‘NULL_PTR’ • Set FeeBlockTypeConfigured = FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA	`(Fee_NotifFunctionPtrType) NULL_PTR,`
	• Set FeeNvmJobErrorNotification container node name as ‘’(empty) • Set FeeBlockTypeConfigured = FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA	`(Fee_NotifFunctionPtrType) NULL_PTR,`

1.2.1.5 Member: FeeQsJobEndNotification

Table 57 FeeQsJobEndNotification

Name	FeeQsJobEndNotification
Type	Fee_NotifFunctionPtrType
Description	Qs job end notification call back routine.
Verification method	The generated structure member is present in the Fee_Config[_<variant>] structure. The member is generated as <call back routine name> if name is set in the container FeeQsJobEndNotification else generated as NULL_PTR. If FeeBlockTypeConfigured = ‘FEE_DOUBLE_SECTOR_DATA_ONLY’ then the member is not generated.

Example(s)	Action	Generated output
	• Set FeeQsJobEndNotification container node name as 'Qs_EndNotif' • Set FeeBlockTypeConfigured = FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA	`(Fee_NotifFunctionPtrType)Qs_EndNotif,`
	• Set FeeQsJobEndNotification container node name as 'NULL_PTR' • Set FeeBlockTypeConfigured = FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA	`(Fee_NotifFunctionPtrType) NULL_PTR,`
	• Set FeeQsJobEndNotification container node name as ''(empty) • Set FeeBlockTypeConfigured = FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA	`(Fee_NotifFunctionPtrType) NULL_PTR,`

1.2.1.6 Member: FeeQsJobErrorNotification

Table 58 FeeQsJobErrorNotification

Name	FeeQsJobErrorNotification
Type	Fee_NotifFunctionPtrType
Description	Qs job error notification call back routine.
Verification method	The generated structure member is present in the Fee_Config[_<variant>] structure. The member is generated as <call back routine name> if name is set in the container FeeQsJobErrorNotification else generated as NULL_PTR. If FeeBlockTypeConfigured = 'FEE_DOUBLE_SECTOR_DATA_ONLY' then the member is not generated.

Example(s)	Action	Generated output
	• Set FeeQsJobErrorNotification container node name as 'Qs_ErrorNotif' • Set FeeBlockTypeConfigured = FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA	`(Fee_NotifFunctionPtrType)Qs_ErrorNotif,`
	• Set FeeQsJobErrorNotification container node name as 'NULL_PTR' • Set FeeBlockTypeConfigured = FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA	`(Fee_NotifFunctionPtrType) NULL_PTR,`
	• Set FeeQsJobErrorNotification container node name as ''(empty) • Set FeeBlockTypeConfigured = FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA	`(Fee_NotifFunctionPtrType) NULL_PTR,`

1.2.1.7 Member: FeeThresholdLimit

Table 59 FeeThresholdLimit

Name	FeeThresholdLimit	
Type	uint32	
Description	Indicates Fee Threshold Limit.	
Verification method	The generated structure member is present in the Fee_Config[_<variant>] structure. The member is generated as numeric value if FeeBlockTypeConfigured is not set to 'FEE_QUASI_STATIC_DATA_ONLY' and it corresponds to the value set in FeeThresholdValue configuration parameter else the member is not generated.	
Example(s)	**Action**	**Generated output**
	• Set FeeBlockTypeConfigured = 'FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA' • Set Threshold Limit = 300	`300U,`

1.2.1.8 Member: FeeBlkCnt

Table 60 FeeBlkCnt

Name	FeeBlkCnt	
Type	uint16	
Description	Counts total number of data blocks.	
Verification method	The generated structure member is present in the Fee_Config[_<variant>] structure. The member is generated as numeric value it corresponds to total number of data block configured.	
Example(s)	**Action**	**Generated output**
	Configure 4 FEE data block with no variant support.	`4U,`
	Configure 3 FEE data block	`3U,`

1.2.1.9 Member: FeeGCConfigSetting

Table 61 FeeGCConfigSetting

Name	FeeGCConfigSetting
Type	Fee_GCConfigType
Description	Setting for unconfigured blocks handling and GC restart point.
Verification method	The generated structure member is present in the Fee_Config[_<variant>] structure. The member is generated as structure with following members. - FeeUnconfigBlock – Generate as numeric value corresponding to literal set for FeeUnConfigBlock configuration parameter if FeeBlockTypeConfigured is not set to 'FEE_QUASI_STATIC_DATA_ONLY' else do not get generated. - FeeGcRestartPoint – Generate as numeric value corresponding to literal set for FeeGcRestart configuration parameter.

	- FeeUseEraseSuspend – Generates as FEE_ERASE_SUSPEND_ENABLED if FeeUseEraseSuspend configuration parameter is set to True else FEE_ERASE_SUSPEND_DISABLED. - Unused – Always generates as 0 (not user configurable).	
Example(s)	**Action**	**Generated output**

	• Set FeeBlockTypeConfigured = 'FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA'
	• Set FeeUnConfigBlock = FEE_UNCONFIG_BLOCK_IGNORE
	• Set FeeGcRestart = FEE_GC_RESTART_INIT
	• Set FeeUseEraseSuspend = False
		`{`
		`/* Ignore the unconfigured blocks */`
		`FEE_UNCONFIG_BLOCK_IGNORE,`
		`/* Restart Garbage Collection during initialization */`
		`FEE_GC_RESTART_INIT,`
		`/* Erase Suspend feature is disabled */`
		`FEE_ERASE_SUSPEND_DISABLED,`
		`/* Reserved */`
		`0U`
		`}`

	• Set FeeBlockTypeConfigured = 'FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA'
	• Set FeeUnConfigBlock = FEE_UNCONFIG_BLOCK_KEEP
	• Set FeeGcRestart = FEE_GC_RESTART_WRITE
	• Set FeeUseEraseSuspend = True
		`{`
		`/* Keep the unconfigured blocks */`
		`FEE_UNCONFIG_BLOCK_KEEP,`
		`/* Restart Garbage Collection when user job is requested */`
		`FEE_GC_RESTART_INIT,`
		`/* Erase Suspend feature is enabled */`
		`FEE_ERASE_SUSPEND_ENABLED,`
		`/* Reserved */`
		`0U`
		`}`

	• Set FeeBlockTypeConfigured = 'FEE_QUASI_STATIC_DATA_ONLY' • Set FeeUseEraseSuspend = True	`{` `/* Restart Garbage Collection when user job is`

		requested */ FEE_GC_RESTART_INIT, /* Erase Suspend feature is enabled */ FEE_ERASE_SUSPEND_ENABLED, /* Reserved */ 0U }

1.2.1.10 Member: FeeNvmIllegalStateNotification

Table 62 FeeNvmIllegalStateNotification

Name	FeeNvmIllegalStateNotification	
Type	Fee_NotifFunctionPtrType	
Description	Notification pointer in case of Nvm Illegal State.	
Verification method	The generated structure member is present in the Fee_Config[_<variant>] structure. The member is generated as <call back routine name> if name is set in FeeNvmIllegalStateNotification configuration parameter else generated as NULL_PTR. *Note: FeeNvmIllegalStateNotification configuration parameter is editable if FeeBlockTypeConfigured is not set to 'FEE_QUASI_STATIC_DATA_ONLY'*	
Example(s)	**Action**	**Generated output**
	Set FeeNvmIllegalStateNotification configuration parameter name as 'Nvm_illegalNotif'	`(Fee_NotifFunctionPtrType) Nvm_illegalNotif,`
	Set FeeNvmIllegalStateNotification configuration parameter name as 'NULL_PTR'	`(Fee_NotifFunctionPtrType) NULL_PTR,`
	Set FeeNvmIllegalStateNotification configuration parameter name as ''(empty)	`(Fee_NotifFunctionPtrType) NULL_PTR,`

1.2.1.11 Member: FeeQsIllegalStateNotification

Table 63 FeeQsIllegalStateNotification

Name	FeeQsIllegalStateNotification
Type	Fee_NotifFunctionPtrType
Description	Notification pointer in case of QS Illegal State.
Verification	The generated structure member is present in the Fee_Config[_<variant>] structure. The member is generated as <call back routine name> if name is set in

method	FeeQsIllegalStateNotification configuration parameter else generated as NULL_PTR. *Note: FeeQsIllegalStateNotification configuration parameter is editable if FeeBlockTypeConfigured is not set to 'FEE_DOUBLE_SECTOR_DATA_ONLY'*	
Example(s)	**Action**	**Generated output**
	Set FeeQsIllegalStateNotification configuration parameter name as 'Qs_illegalNotif'	`(Fee_NotifFunctionPtrType) Qs_illegalNotif,`
	Set FeeQsIllegalStateNotification configuration parameter name as 'NULL_PTR'	`(Fee_NotifFunctionPtrType) NULL_PTR,`
	Set FeeQsIllegalStateNotification configuration parameter name as ''(empty)	`(Fee_NotifFunctionPtrType) NULL_PTR,`

1.2.1.12 Member: FeeQsHardenErrorNotification

Table 64 FeeQsHardenErrorNotification

Name	FeeQsHardenErrorNotification	
Type	Fee_NotifFunctionPtrType	
Description	Notification pointer in case of QS hardening error.	
Verification method	The generated structure member is present in the Fee_Config[_<variant>] structure. The member is generated as <call back routine name> if name is set in FeeQsHardenErrorNotification configuration parameter else generated as NULL_PTR. *Note: FeeQsHardenErrorNotification configuration parameter is editable if FeeBlockTypeConfigured is set to 'FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA'*	
Example(s)	**Action**	**Generated output**
	Set FeeQsHardenErrorNotification configuration parameter name as 'Qs_HardeningErrNotif'	`(Fee_NotifFunctionPtrType) Qs_HardeningErrNotif,`
	Set FeeQsHardenErrorNotification configuration parameter name as 'NULL_PTR'	`(Fee_NotifFunctionPtrType) NULL_PTR,`
	Set FeeQsHardenErrorNotification configuration parameter name as ''(empty)	`(Fee_NotifFunctionPtrType) NULL_PTR,`

1.2.2 Structure: Fee_ BlockConfig[_<variant>]

Table 65 Fee_ BlockConfig[_<variant>]

Name	Fee_ BlockConfig[_<variant>]	
Type	Fee_BlockType	
Description	Configuration structure of FEE driver for all data blocks.	
Verification method	The generated file has this structure containing configuration information for each data block. <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 3 FEE data block	`static const Fee_BlockType Fee_BlockConfig [ ] =`
		`{`
		`{`
		`10000U, /* Block Cycle Count */`
		`(uint8)FEE_NORMAL_DATA, /* Block type is Normal */`
		`1U, /* Block number */`
		`8192U, /* Fee Block Size */`
		`0x00U, /* Fee Block address */`
		`0U, /* Fee Block instance */`
		`FEE_NVM_USER /* Fee quasi/NVM manager */`
		`},`
		`{`
		`20000U, /* Block Cycle Count */`
		`(uint8)FEE_IMMEDIATE_DATA, /* Block type is Immediate */`
		`2U, /* Block number */`
		`12288U, /* Fee Block Size */`
		`0x00U, /* Fee Block address */`
		`0U, /* Fee Block instance */`
		`FEE_NVM_USER /* Fee quasi/NVM manager */`
		`},`
		`{`
		`50U, /* Block Cycle Count */`
		`(uint8)FEE_NORMAL_DATA, /* Block type is Normal */`
		`400U, /* Block number */`
		`4096U, /* Fee Block Size */`

	258048U, /* Fee Block address */ 1U, /* Fee Block instance */ FEE_QUASI_STATIC_USER /* Fee quasi manager */ }, }
Configure 3 FEE data block with variant support. Set the name of the variant as 'Petrol'	static const Fee_BlockType Fee_BlockConfig_Petrol[ ] = { { 10000U, /* Block Cycle Count */ (uint8)FEE_NORMAL_DATA, /* Block type is Normal */ 1U, /* Block number */ 8192U, /* Fee Block Size */ 0x00U, /* Fee Block address */ 0U, /* Fee Block instance */ FEE_NVM_USER /* Fee quasi/NVM manager */ }, { 20000U, /* Block Cycle Count */ (uint8)FEE_IMMEDIATE_DATA, /* Block type is Immediate */ 2U, /* Block number */ 12288U, /* Fee Block Size */ 0x00U, /* Fee Block address */ 0U, /* Fee Block instance */ FEE_NVM_USER /* Fee quasi/NVM manager */ }, { 50U, /* Block Cycle Count */ (uint8)FEE_NORMAL_DATA, /* Block type is Normal */ 400U, /* Block number */ 4096U, /* Fee Block Size */ 258048U, /* Fee Block address */ 1U, /* Fee Block instance */ FEE_QUASI_STATIC_USER /* Fee quasi manager */ },

	}

1.2.2.1 Member: CycleCountLimit

Table 66 CycleCountLimit

Name	CycleCountLimit	
Type	unsigned_int : 24	
Description	Indicates block cycle count configured.	
Verification method	The generated structure member is present in the Fee_ BlockConfig[_<variant>] structure. The member is generated as numeric value corresponding to the value set in FeeNumberOfWriteCycles configuration parameter.	
Example(s)	**Action**	**Generated output**
	• Configure 2 data blocks • Set FeeBlockConfiguration_0/FeeNumberOfWriteCycles = 10000	`10000U, /* block cycle count */`
	Set FeeBlockConfiguration_1/FeeNumberOfWriteCycles = 1000	`1000U, /* block cycle count */`

1.2.2.2 Member: FeeImmediateData

Table 67 FeeImmediateData

Name	FeeImmediateData	
Type	unsigned_int : 8	
Description	Denotes configured block data is immediate data or normal data.	
Verification method	The generated structure member is present in the Fee_ BlockConfig[_<variant>] structure. The member is generated as FEE_IMMEDIATE_DATA if FeeImmediateData configuration parameter is set to 'True' else generated as FEE_NORMAL_DATA.	
Example(s)	**Action**	**Generated output**
	• Configure 2 data blocks • Set FeeBlockConfiguration_0/FeeImmediateData = False	`(uint8)FEE_NORMAL_DATA,` `/* Block type is Normal */`
	Set FeeBlockConfiguration_1/ FeeImmediateData = True	`(uint8)FEE_IMMEDIATE_DATA,` `/* Block type is Normal */`

1.2.2.3 Member: BlockNumber

Table 68 BlockNumber

Name	BlockNumber
Type	unsigned_int : 16
Description	Indicates logical block number.
Verification method	The generated structure member is present in the Fee_ BlockConfig[_<variant>] structure. The member is generated as numeric value corresponds to value set for FeeBlockNumber configuration parameter.

Example(s)	Action	Generated output
	• Configure 2 data blocks • Set FeeBlockConfiguration_0/ FeeBlockNumber = 21	`21U, /* Block number */`
	Set FeeBlockConfiguration_1/ FeeBlockNumber = 2	`1U, /* Block number */`

1.2.2.4 Member: Size

Table 69 Size

Name	Size	
Type	unsigned_int : 16	
Description	Size of the data block configured.	
Verification method	The generated structure member is present in the Fee_BlockConfig[_<variant>] structure. The member is generated as numeric value corresponds to value set for FeeBlockSize configuration parameter.	
Example(s)	Action	Generated output
	• Configure 2 data blocks • Set FeeBlockConfiguration_0/ FeeBlockNumber = 21	`21U, /* Block number */`
	Set FeeBlockConfiguration_1/ FeeBlockNumber = 2	`1U, /* Block number */`

1.2.2.5 Member: Address

Table 70 Address

Name	Address	
Type	unsigned_int : 32	
Description	Block address for Qs data block in DFlash.	
Verification method	The generated structure member is present in the Fee_BlockConfig[_<variant>] structure. The member is generated as numerical value corresponds to value set for FeeQsBlockAddress configuration parameter if FeeBlockTypeConfigured is not set to 'FEE_DOUBLE_SECTOR_DATA_ONLY' else the structure member is not generated.	
Example(s)	Action	Generated output
	• Set FeeBlockTypeConfigured = 'FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA' • Configure 2 data blocks FeeBlockConfiguration_0 as Nvm and FeeBlockConfiguration_1 as Qs • Set FeeBlockConfiguration_1/FeeQsBlockAddress = 258048	`258048U, /* Fee Block address */`
	• Set FeeBlockTypeConfigured = 'FEE_DOUBLE_SECTOR_DATA_ONLY'	

	Note: *Member is not generated*	

1.2.2.6 Member: Instances

Table 71 Instances

Name	Instances	
Type	unsigned_int : 16	
Description	Number of Qs Block instances.	
Verification method	The generated structure member is present in the Fee_BlockConfig[_<variant>] structure. The member is generated as numerical value corresponds to value set for FeeQsBlockInstances configuration parameter if FeeBlockTypeConfigured is not set to 'FEE_DOUBLE_SECTOR_DATA_ONLY' else the structure member is not generated.	
Example(s)	**Action**	**Generated output**
	• Set FeeBlockTypeConfigured = 'FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA' • Configure 2 data blocks FeeBlockConfiguration_0 as Nvm and FeeBlockConfiguration_1 as Qs • Set FeeBlockConfiguration_1/FeeQsBlockInstances = 10	`10U, /* Fee Block instance */`
	• Set FeeBlockTypeConfigured = 'FEE_DOUBLE_SECTOR_DATA_ONLY' *Note:* *Member is not generated*	

1.2.2.7 Member: FeeUser

Table 72 FeeUser

Name	FeeUser	
Type	unsigned_int : 8	
Description	Indicates user type of the data block, Nvm or Quasi.	
Verification method	The generated structure member is present in the Fee_BlockConfig[_<variant>] structure. The member is generated as FEE_QUASI_STATIC_USER if FeeQuasiStaticManager configuration parameter is set to 'True' else generated as FEE_NVM_USER. If the FeeBlockTypeConfigured configuration parameter is set to 'FEE_DOUBLE_SECTOR_DATA_ONLY' then the member is not generated.	
Example(s)	**Action**	**Generated output**
	• Set FeeBlockTypeConfigured = 'FEE_DOUBLE_SECTOR_AND_QUASI_STATIC_DATA' • Configure 2 data blocks and set FeeBlockConfiguration_1/FeeQuasiStaticManag	`FEE_QUASI_STATIC_USER /* Fee quasi manager */`

	er = True (1 QS block)	
	• Set FeeBlockConfiguration_0/FeeQuasiStaticManager = False (1 Nvm block)	`FEE_NORMAL_DATA /* Fee quasi manager */`
	• Set FeeBlockTypeConfigured = 'FEE_DOUBLE_SECTOR_DATA_ONLY' *Note: Member is not generated*	

1.2.3 Function Declaration: FeeNvmJobEndNotification

Table 73 FeeNvmJobEndNotification

Name	FeeNvmJobEndNotification	
Type	void <Notification name as per configured>(void)	
Description	The function declaration for Nvm job end notification function.	
Verification method	The function declaration is generated as extern void <Notification name as per configured>(void) if the name set for FeeNvmJobEndNotification configuration container node is not a number, is not 'NvM_JobEndNotification', is not blank, not undefined and not 'NULL_PTR' else it is not generated.	
Example(s)	**Action**	**Generated output**
	Set FeeNvmJobEndNotification container node name as 'Nvm_EndNotif'	`extern void Nvm_EndNotif(void);`
	Set FeeNvmJobEndNotification container node name as 'NULL_PTR' *Note: Declaration is not generated*	
	FeeNvmJobEndNotification container node id not present *Note: Declaration is not generated*	
	Set FeeNvmJobEndNotification container node name as 'NvM_JobEndNotification' *Note: Declaration is not generated*	
	Set FeeNvmJobEndNotification container node name as '' (empty) *Note: Declaration is not generated*	

1.2.4 Function Declaration: FeeNvmJobErrorNotification

Table 74 FeeNvmJobErrorNotification

Name	FeeNvmJobErrorNotification	
Type	void <Notification name as per configured>(void)	
Description	The function declaration for Nvm job error notification function.	
Verification method	The function declaration is generated as extern void <Notification name as per configured>(void) if the name set for FeeNvmJobErrorNotification configuration container node is not a number, is not 'NvM_JobErrorNotification', is not blank, not undefined and not 'NULL_PTR' else it is not generated.	
Example(s)	**Action**	**Generated output**
	Set FeeNvmJobErrorNotification container node name as 'Nvm_ErrNotif'	`extern void Nvm_ErrNotif(void);`
	Set FeeNvmJobErrorNotification container node name as 'NULL_PTR' *Note: Declaration is not generated*	
	FeeNvmJobErrorNotification container node id not present *Note: Declaration is not generated*	
	Set FeeNvmJobErrorNotification container node name as 'NvM_JobEndNotification' *Note: Declaration is not generated*	
	Set FeeNvmJobErrorNotification container node name as '' (empty) *Note: Declaration is not generated*	

1.2.5 Function Declaration: FeeQsJobEndNotification

Table 75 FeeQsJobEndNotification

Name	FeeQsJobEndNotification
Type	void <Notification name as per configured>(void)
Description	The function declaration for Qs job end notification function.
Verification method	The function declaration is generated as extern void <Notification name as per configured>(void) if the name set for FeeQsJobEndNotification configuration container

	node is not a number, is not blank, not undefined and not 'NULL_PTR' else it is not generated.	
Example(s)	**Action**	**Generated output**
	Set FeeQsJobEndNotification container node name as 'Qs_EndNotif'	`extern void Qs_EndNotif(void);`
	Set FeeQsJobEndNotification container node name as 'NULL_PTR' *Note: Declaration is not generated*	
	FeeQsJobEndNotification container node id not present *Note: Declaration is not generated*	
	Set FeeQsJobEndNotification container node name as '' (empty) *Note: Declaration is not generated*	

1.2.6 Function Declaration: FeeQsJobErrorNotification

Table 76 FeeQsJobErrorNotification

Name	FeeQsJobErrorNotification	
Type	void <Notification name as per configured>(void)	
Description	The function declaration for Qs job error notification function.	
Verification method	The function declaration is generated as extern void <Notification name as per configured>(void) if the name set for FeeQsJobErrorNotification configuration container node is not a number, is not blank, not undefined and not 'NULL_PTR' else it is not generated.	
Example(s)	**Action**	**Generated output**
	Set FeeQsJobErrorNotification container node name as 'Qs_ErrNotif'	`extern void Qs_ErrNotif(void);`
	Set FeeQsJobErrorNotification container node name as 'NULL_PTR' *Note: Declaration is not generated*	
	FeeQsJobErrorNotification container node id not present	

	Note: *Declaration is not generated*	
	Set FeeQsJobErrorNotification container node name as ‘’ (empty) *Note:* *Declaration is not generated*	

1.2.7 Function Declaration: FeeNvmIllegalStateNotification

Table 77 FeeNvmIllegalStateNotification

Name	FeeNvmIllegalStateNotification	
Type	void <Notification name as per configured>(void)	
Description	The function declaration for Nvm job illegal state notification function.	
Verification method	The function declaration is generated as extern void <Notification name as per configured>(void) if the name set for FeeNvmIllegalStateNotification configuration is not blank, not undefined and not ‘NULL_PTR’ else it is not generated.	
Example(s)	**Action**	**Generated output**
	Set FeeNvmIllegalStateNotification container node name as ‘Nvm_illStateNotif’	`extern void Nvm_illStateNotif (void);`
	Set FeeNvmIllegalStateNotification container node name as ‘NULL_PTR’ *Note:* *Declaration is not generated*	
	FeeNvmIllegalStateNotification container node id not present *Note:* *Declaration is not generated*	
	Set FeeNvmIllegalStateNotification container node name as ‘’ (empty) *Note:* *Declaration is not generated*	

1.2.8 Function Declaration: FeeQsIllegalStateNotification

Table 78 FeeQsIllegalStateNotification

Name	FeeQsIllegalStateNotification
Type	void <Notification name as per configured>(void)

Description	The function declaration for Nvm job illegal state notification function.	
Verification method	The function declaration is generated as extern void <Notification name as per configured>(void) if the name set for FeeQsIllegalStateNotification configuration container node is not blank not undefined and not 'NULL_PTR' else it is not generated.	
Example(s)	**Action**	**Generated output**
	Set FeeQsIllegalStateNotification container node name as 'Qs_illStateNotif'	`extern void Qs_illStateNotif (void);`
	Set FeeQsIllegalStateNotification container node name as 'NULL_PTR' *Note: Declaration is not generated*	
	FeeQsIllegalStateNotification container node id not present *Note: Declaration is not generated*	
	Set FeeQsIllegalStateNotification container node name as '' (empty) *Note: Declaration is not generated*	

1.2.9 Function Declaration: FeeQsHardenErrorNotification

Table 79 FeeQsHardenErrorNotification

Name	FeeQsHardenErrorNotification	
Type	void <Notification name as per configured>(void)	
Description	The function declaration for Nvm job illegal state notification function.	
Verification method	The function declaration is generated as extern void <Notification name as per configured>(void) if the name set for FeeQsHardenErrorNotification configuration container node is not blank, not undefined and not 'NULL_PTR' else it is not generated.	
Example(s)	**Action**	**Generated output**
	Set FeeQsHardenErrorNotification container node name as 'Qs_hardenErrorNotif'	`extern void Qs_hardenErrorNotif (void);`
	Set FeeQsHardenErrorNotification container node name as 'NULL_PTR' *Note: Declaration is not generated*	

	FeeQsHardenErrorNotification container node id not present *Note:* *Declaration is not generated*	
	Set FeeQsHardenErrorNotification container node name as ‘’ (empty) *Note:* *Declaration is not generated*	

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

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

1.3.1 Structure: Fee_Config[_<variant>]

Table 80 Fee_Config[_<varaint>]

Name	Fee_Config[_<variant>]	
Type	Fee_ConfigType	
Description	Declaration of root configuration structure of FEE driver which will be used during initialization.	
Verification method	The generated structure is present in Fee[_<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 FEE and generate (variant-unaware)	`extern const Fee_ConfigType Fee_Config;`
	Configure FEE and generate (variant-aware. Variant name is 'Petrol')	`extern const Fee_ConfigType Fee_Config_Petrol;`

Revision history

Major changes since the last revision

Date	Version	Description
24-Nov-20	1.0	Released.

12-Nov-20	0.1	• Added following derived parameter
		  - FEE_GET_ECC_ERROR_INFO_API
		  - FEE_RUNTIME_ERROR_DETECT
		  - FEE_PAGES_PER_FEEMAIN
		• FEE_DFLASH_WORDLINE_SIZE removed
		• FEE driver chapter moved from MC-ISAR_TC3xx_Config_Verification_Manual_BASIC.pdf to this document.

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

Edition 2020-11-24
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.