MCAL Configuration Verification Manual for Fls_17_Dmu

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 Fls_17_Dmu

Table of contents

1 Fls_17_Dmu driver

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

1.1 File: Fls_17_Dmu_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: FLS_17_DMU_AR_RELEASE_MAJOR_VERSION

Table 1 FLS_17_DMU_AR_RELEASE_MAJOR_VERSION

Name	FLS_17_DMU_AR_RELEASE_MAJOR_VERSION	
Description	Major version number of AUTOSAR release on which the FLS_17_DMU 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 FLS_17_DMU_Cfg.h file with ArMajorVersion 4	`#define FLS_17_DMU_AR_RELEASE_MAJOR_VERSION (4U)`

1.1.2 Macro: FLS_17_DMU_AR_RELEASE_MINOR_VERSION

Table 2 FLS_17_DMU_AR_RELEASE_MINOR_VERSION

Name	FLS_17_DMU_AR_RELEASE_MINOR_VERSION	
Description	Minor version number of AUTOSAR release on which the FLS_17_DMU 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 FLS_17_DMU_Cfg.h file with ArMinorVersion 2	`#define FLS_17_DMU_AR_RELEASE_MINOR_VERSION (2U)`

1.1.3 Macro: FLS_17_DMU_AR_RELEASE_REVISION_VERSION

Table 3 FLS_17_DMU_AR_RELEASE_REVISION_VERSION

Name	FLS_17_DMU_AR_RELEASE_REVISION_VERSION

Description	Revision version number of AUTOSAR release on which the FLS_17_DMU 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 FLS_17_DMU_Cfg.h file with ArPatchVersion 2	`#define FLS_17_DMU_AR_RELEASE_REVISION_VERSION (2U)`

1.1.4 Macro: FLS_17_DMU_SW_MAJOR_VERSION

Table 4 FLS_17_DMU_SW_MAJOR_VERSION

Name	FLS_17_DMU_SW_MAJOR_VERSION	
Description	Major version number of the FLS_17_DMU 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 FLS_17_DMU_Cfg.h file with SwMajorVersion 10	`#define FLS_17_DMU_SW_MAJOR_VERSION (10U)`

1.1.5 Macro: FLS_17_DMU_SW_MINOR_VERSION

Table 5 FLS_17_DMU_SW_MINOR_VERSION

Name	FLS_17_DMU_SW_MINOR_VERSION	
Description	Minor version number of the FLS_17_DMU 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 FLS_17_DMU_Cfg.h file with SwMinorVersion 10	`#define FLS_17_DMU_SW_MINOR_VERSION (10U)`

1.1.6 Macro: FLS_17_DMU_SW_PATCH_VERSION

Table 6 FLS_17_DMU_SW_PATCH_VERSION

Name	FLS_17_DMU_SW_PATCH_VERSION	
Description	Patch level version number of the FLS_17_DMU 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 FLS_17_DMU_Cfg.h file with SwPatchVersion 0	`#define FLS_17_DMU_SW_PATCH_VERSION (0U)`

1.1.7 Macro: FLS_17_DMU_INSTANCE_ID

Table 7 FLS_17_DMU_INSTANCE_ID

Name	FLS_17_DMU_INSTANCE_ID	
Description	Provides FLS Instance ID.	
Verification method	The macro is generated as per the value in FlsDriverIndex.	
Example(s)	**Action**	**Generated output**
	FlsDriverIndex = 0	`#define FLS_17_DMU_INSTANCE_ID ((uint8)0)`

1.1.8 Macro: FLS_17_DMU_DEV_ERROR_DETECT

Table 8 FLS_17_DMU_DEV_ERROR_DETECT

Name	FLS_17_DMU_DEV_ERROR_DETECT	
Description	Enables/disables development error check.	
Verification method	The macro is generated as STD_ON if 'FlsGeneral/FlsDevErrorDetect' configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	FlsDevErrorDetect = True	`#define FLS_17_DMU_DEV_ERROR_DETECT (STD_ON)`
	FlsDevErrorDetect = False	`#define FLS_17_DMU_DEV_ERROR_DETECT (STD_OFF)`

1.1.9 Macro: FLS_17_DMU_RUNTIME_ERROR_DETECT

Table 9 FLS_17_DMU_RUNTIME_ERROR_DETECT

Name	FLS_17_DMU_RUNTIME_ERROR_DETECT
Description	Enables/disables runtime error check.
Verification method	The macro is generated as STD_ON if 'FlsGeneral/FlsRunTimeErrorDetect' configuration parameter is set to 'True' else the macro is generated as STD_OFF.

<table>
<tr><th>Example(s)</th><th>Action</th><th>Generated output</th></tr>
<tr><td rowspan="2"></td><td>FlsRunTimeErrorDetect = True</td><td>#define FLS_17_DMU_RUNTIME_ERROR_DETECT (STD_ON)</td></tr>
<tr><td>FlsRunTimeErrorDetect = False</td><td>#define FLS_17_DMU_RUNTIME_ERROR_DETECT (STD_OFF)</td></tr>
</table>

1.1.10 Macro: FLS_17_DMU_SAFETY_ENABLE

Table 10 FLS_17_DMU_SAFETY_ENABLE

<table>
<tr><td><b>Name</b></td><td colspan="2">FLS_17_DMU_SAFETY_ENABLE</td></tr>
<tr><td><b>Description</b></td><td colspan="2">Enables/disables safety error check.</td></tr>
<tr><td><b>Verification method</b></td><td colspan="2">The macro is generated as STD_ON if ‘FlsGeneral/FlsSafetyEnable’ configuration parameter is set to ‘True’ else the macro is generated as STD_OFF.</td></tr>
<tr><td rowspan="3"><b>Example(s)</b></td><td><b>Action</b></td><td><b>Generated output</b></td></tr>
<tr><td>FlsSafetyEnable = True</td><td>#define FLS_17_DMU_SAFETY_ENABLE (STD_ON)</td></tr>
<tr><td>FlsSafetyEnable = False</td><td>#define FLS_17_DMU_SAFETY_ENABLE (STD_OFF)</td></tr>
</table>

1.1.11 Macro: FLS_17_DMU_INITCHECK_API

Table 11 FLS_17_DMU_INITCHECK_API

<table>
<tr><td><b>Name</b></td><td colspan="2">FLS_17_DMU_INITCHECK_API</td></tr>
<tr><td><b>Description</b></td><td colspan="2">Enables/disables the Fls_17_Dmu_InitCheck API.</td></tr>
<tr><td><b>Verification method</b></td><td colspan="2">The macro is generated as STD_ON if ‘FlsGeneral/FlsInitCheckApi’ configuration parameter is set to ‘True’ else the macro is generated as STD_OFF.</td></tr>
<tr><td rowspan="3"><b>Example(s)</b></td><td><b>Action</b></td><td><b>Generated output</b></td></tr>
<tr><td>FlsInitCheckApi = True</td><td>#define FLS_17_DMU_INITCHECK_API (STD_ON)</td></tr>
<tr><td>FlsInitCheckApi = False</td><td>#define FLS_17_DMU_INITCHECK_API (STD_OFF)</td></tr>
</table>

1.1.12 Macro: FLS_17_DMU_USE_INTERRUPTS

Table 12 FLS_17_DMU_USE_INTERRUPTS

Name	FLS_17_DMU_USE_INTERRUPTS
Description	Enables/disables flash job processing triggered by hardware interrupt.
Verification method	The macro has dependency on Ifx Fee.

	1. If 'FlsGeneral/FlsIfxFeeUse' is not enabled and 'FlsGeneral/FlsUseInterrupts' is enabled, then the macro is generated as STD_ON. 2. If 'FlsGeneral/FlsIfxFeeUse' is not enabled and 'FlsGeneral/FlsUseInterrupts' is not enabled, then the macro is generated as STD_OFF. 3. If 'FlsGeneral/FlsIfxFeeUse' is enabled, then the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**

	FlsIfxFeeUse = False
	FlsUseInterrupts = True
		`#define FLS_17_DMU_USE_INTERRUPTS (STD_ON)`

	FlsIfxFeeUse = False
	FlsUseInterrupts = False
		`#define FLS_17_DMU_USE_INTERRUPTS (STD_OFF)`

	FlsIfxFeeUse = True	`#define FLS_17_DMU_USE_INTERRUPTS (STD_OFF)`

1.1.13 Macro: FLS_17_DMU_CANCEL_API

Table 13 FLS_17_DMU_CANCEL_API

Name	FLS_17_DMU_CANCEL_API	
Description	Enables/disables the Fls_17_Dmu_Cancel API.	
Verification method	The macro is generated as STD_ON if 'FlsGeneral/FlsCancelApi' configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	FlsCancelApi = True	`#define FLS_17_DMU_CANCEL_API (STD_ON)`
	FlsCancelApi = False	`#define FLS_17_DMU_CANCEL_API (STD_OFF)`

1.1.14 Macro: FLS_17_DMU_SET_MODE_API

Table 14 FLS_17_DMU_SET_MODE_API

Name	FLS_17_DMU_SET_MODE_API	
Description	Enables/disables Fls_17_Dmu_SetMode API.	
Verification method	The macro is generated as STD_ON if 'FlsGeneral/FlsSetModeApi' configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	FlsSetModeApi = True	`#define FLS_17_DMU_SET_MODE_API (STD_ON)`
	FlsSetModeApi = False	`#define FLS_17_DMU_SET_MODE_API (STD_OFF)`

1.1.15 Macro: FLS_17_DMU_COMPARE_API

Table 15 FLS_17_DMU_COMPARE_API

Name	FLS_17_DMU_COMPARE_API	
Description	Enables/disables Fls_lMainCompare API.	
Verification method	The macro is generated as STD_ON if 'FlsGeneral/FlsCompareApi' configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	FlsCompareApi = True	`#define FLS_17_DMU_COMPARE_API (STD_ON)`
	FlsCompareApi = False	`#define FLS_17_DMU_COMPARE_API (STD_OFF)`

1.1.16 Macro: FLS_17_DMU_BLANK_CHECK_API

Table 16 FLS_17_DMU_BLANK_CHECK_API

Name	FLS_17_DMU_BLANK_CHECK_API	
Description	Enables/disables Fls_lMainBlankCheck API.	
Verification method	The macro is generated as STD_ON if 'FlsGeneral/FlsBlankCheckApi' configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	FlsBlankCheckApi = True	`#define FLS_17_DMU_BLANK_CHECK_API (STD_ON)`
	FlsBlankCheckApi = False	`#define FLS_17_DMU_BLANK_CHECK_API (STD_OFF)`

1.1.17 Macro: FLS_17_DMU_GET_JOB_RESULT_API

Table 17 FLS_17_DMU_GET_JOB_RESULT_API

Name	FLS_17_DMU_GET_JOB_RESULT_API	
Description	Enable/disable Fls_17_Dmu_GetJobResult API.	
Verification method	The macro is generated as STD_ON if 'FlsGeneral/FlsGetJobResultApi' configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	FlsGetJobResultApi= True	`#define FLS_17_DMU_GET_JOB_RESULT_API (STD_ON)`
	FlsGetJobResultApi= False	`#define FLS_17_DMU_GET_JOB_RESULT_API (STD_OFF)`

1.1.18 Macro: FLS_17_DMU_GET_STATUS_API

Table 18 FLS_17_DMU_GET_STATUS_API

Name	FLS_17_DMU_GET_STATUS_API	
Description	Enables/disables Fls_17_Dmu_GetStatus API.	
Verification method	The macro is generated as STD_ON if 'FlsGeneral/FlsGetStatusApi' configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	FlsGetStatusApi = True	`#define FLS_17_DMU_GET_STATUS_API (STD_ON)`
	FlsGetStatusApi = False	`#define FLS_17_DMU_GET_STATUS_API (STD_OFF)`

1.1.19 Macro: FLS_17_DMU_VERSION_INFO_API

Table 19 FLS_17_DMU_VERSION_INFO_API

Name	FLS_17_DMU_VERSION_INFO_API	
Description	Enables/disables Fls_17_Dmu_GetVersionInfo API.	
Verification method	The macro is generated as STD_ON if 'FlsGeneral/FlsVersionInfoApi' configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	FlsVersionInfoApi = True	`#define FLS_17_DMU_VERSION_INFO_API (STD_ON)`
	FlsVersionInfoApi = False	`#define FLS_17_DMU_VERSION_INFO_API (STD_OFF)`

1.1.20 Macro: FLS_17_DMU_USE_ERASESUSPEND

Table 20 FLS_17_DMU_USE_ERASESUSPEND

Name	FLS_17_DMU_USE_ERASESUSPEND	
Description	Enables/disables the suspend/resume functionality of erase operation using the Fls_17_Dmu_SuspendErase and Fls_17_Dmu_ResumeErase API.	
Verification method	The macro is generated as STD_ON if 'FlsIfxSpecificConfig/FlsUseEraseSuspend' configuration parameter is set to 'True' else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	FlsUseEraseSuspend = True	`#define FLS_17_DMU_USE_ERASESUSPEND (STD_ON)`
	FlsUseEraseSuspend = False	`#define FLS_17_DMU_USE_ERASESUSPEND (STD_OFF)`

1.1.21 Macro: FLS_17_DMU_IFX_FEE_USED

Table 21 FLS_17_DMU_IFX_FEE_USED

Name	FLS_17_DMU_IFX_FEE_USED	
Description	Enables/disables the usage of IFX FEE module.	
Verification method	The macro is generated as STD_ON if ‘FlsGeneral/FlsIfxFeeUse’ configuration parameter is set to ‘True’ else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	FlsIfxFeeUse = True	#define FLS_17_DMU_IFX_FEE_USED (STD_ON)
	FlsIfxFeeUse = False	#define FLS_17_DMU_IFX_FEE_USED (STD_OFF)

1.1.22 Macro: FLS_17_DMU_BASE_ADDRESS

Table 22 FLS_17_DMU_BASE_ADDRESS

Name	FLS_17_DMU_BASE_ADDRESS	
Description	Specifies DFLASH0 base/start address. *Note: This macro is not configurable by the user. This macro is value is generated as hex converted numeric value specified in Resource manager.*	
Verification method	The macro is generated with the base address of DFLASH0.	
Example(s)	**Action**	**Generated output**
		FLS_17_DMU_BASE_ADDRESS (0xaf000000U)

1.1.23 Macro: FLS_17_DMU_TOTAL_SIZE

Table 23 FLS_17_DMU_TOTAL_SIZE

Name	FLS_17_DMU_TOTAL_SIZE	
Description	Specifies DFLASH0 total size.	
Verification method	This macro is generated as per the total size of DFLASH0 of the derivative device. *Note: The macro has its default value derived from the variant configured in Resource Manager; user can change it with valid values (Between 4kB and DFLASH0 size and Multiple of 4kB).*	
Example(s)	**Action**	**Generated output**
	When device selected is TC38x FlsTotalSize = 0x80000	#define FLS_17_DMU_TOTAL_SIZE (0x80000U)

	When device selected is TC39x FlsTotalSize = 0x100000	`#define FLS_17_DMU_TOTAL_SIZE (0x100000U)`

1.1.24 Macro: FLS_17_DMU_NVMSECTOR0_STARTADDRESS

Table 24 FLS_17_DMU_NVMSECTOR0_STARTADDRESS

Name	FLS_17_DMU_NVMSECTOR0_STARTADDRESS	
Description	Specifies the start address of the first of the two sectors used for double sector algorithm.	
Verification method	This macro will be generated when ‘FlsGeneral/FlsIfxFeeUse’ is True and ‘FlsConfigSet/FlsSectorList/FlsSector/FlsSector_0/FlsNumberOfSectors’ is 2. The macro is generated by performing addition of ‘FlsSector/FlsSector_0/FlsSectorStartaddress’ and ‘FlsGeneral/FlsBaseAddress’. *Note: This macro is generated as hex converted numeric value of the addition operation of the above two parameters.*	
Example(s)	**Action**	**Generated output**
	FlsBaseAddress = 2936012800 Add a container for ‘FlsSector’ and set the configuration parameters as below: FlsSector_0/FlsNumberOfSectors = 2 FlsSector_0/FlsSectorStartaddress = 0	`#define FLS_17_DMU_NVMSECTOR0_STARTADDRESS (0xaf000000U)`
	FlsBaseAddress = 2936012800 Similarly as in the above example, for ‘FlsSector’, if the ‘FlsSectorStartaddress’ is changed keeping the other parameters unchanged: FlsSector_0/FlsNumberOfSectors = 2 FlsSector_0/FlsSectorStartaddress = 131072	`#define FLS_17_DMU_NVMSECTOR0_STARTADDRESS (0xaf020000U)`

1.1.25 Macro: FLS_17_DMU_NVMSECTOR1_STARTADDRESS

Table 25 FLS_17_DMU_NVMSECTOR1_STARTADDRESS

Name	FLS_17_DMU_NVMSECTOR1_STARTADDRESS
Description	Specifies the start address of the second of the two sectors used for double sector algorithm.
Verification method	This macro will be generated when ‘FlsGeneral/FlsIfxFeeUse’ is True and ‘FlsConfigSet/FlsSectorList/FlsSector/FlsSector_0/FlsNumberOfSectors’ is 2. The

	macro is generated by performing addition of 'FlsSector/FlsSector_0/FlsSectorStartaddress', 'FlsSector/FlsSector_0/FlsSectorSize' and 'FlsGeneral/FlsBaseAddress'. *Note: This macro is generated as hex converted numeric value of the addition operation of the above three parameters.*	
Example(s)	**Action**	**Generated output**
	FlsBaseAddress = 2936012800 Add a container for 'FlsSector' and set the configuration parameters as below: FlsSector_0/FlsSectorSize = 131072 FlsSector_0/FlsSectorStartaddress = 0	`#define FLS_17_DMU_NVMSECTOR1_STARTADDRESS (0xaf020000U)`
	FlsBaseAddress = 2936012800 Similarly as in the above example, for 'FlsSector', if the 'FlsSectorStartaddress' and 'FlsSectorSize' is changed keeping the other parameters unchanged: FlsSector_0/FlsSectorSize = 131072 FlsSector_0/FlsSectorStartaddress = 131072	`#define FLS_17_DMU_NVMSECTOR1_STARTADDRESS (0xaf040000U)`

1.1.26 Macro: FLS_17_DMU_NVMSECTOR_SIZE

Table 26 FLS_17_DMU_NVMSECTOR_SIZE

Name	FLS_17_DMU_NVMSECTOR_SIZE	
Description	Specifies the size of the sectors of the double sector algorithm.	
Verification method	This macro will be generated when 'FlsGeneral/FlsIfxFeeUse' is True and 'FlsConfigSet/FlsSectorList/FlsSector/FlsNumberOfSectors' is 2. The macro is generated with the value in 'FlsSector/FlsSectorSize' configured by the user. *Note: This macro is generated as hex converted numeric value set in FlsSectorSize parameter.*	
Example(s)	**Action**	**Generated output**
	Add a container for 'FlsSector' and set the configuration parameters as below: FlsSector_0/FlsSectorSize = 131072	`#define FLS_17_DMU_NVMSECTOR_SIZE (0x20000)`

1.1.27 Macro: FLS_17_DMU_QSDATA_STARTADDRESS

Table 27 FLS_17_DMU_QSDATA_STARTADDRESS

Name	FLS_17_DMU_QSDATA_STARTADDRESS	
Description	Specifies the start address of the quasi static data.	
Verification method	This macro will be generated when 'FlsGeneral/FlsIfxFeeUse' is True and 'FlsConfigSet/FlsSectorList/FlsSector/FlsNumberOfSectors' is 1. The macro is generated by performing addition of 'FlsSector/FlsSectorStartaddress' and 'FlsGeneral/FlsBaseAddress. *Note:* *This macro is generated as hex converted value of the addition operation of the above two parameters.*	
Example(s)	**Action**	**Generated output**
	FlsBaseAddress = 2936012800 Add a container for 'FlsSector' and set the configuration parameters as below: FlsSector_0/ FlsNumberOfSectors = 1 FlsSector_0/FlsSectorStartaddress = 0	`#define FLS_17_DMU_QSDATA_STARTADDRESS (0xaf000000U)`
	FlsBaseAddress = 2936012800 Similarly as in the above example, for 'FlsSector', if the 'FlsSectorStartaddress' and changed keeping the other parameters unchanged: FlsSector_0/FlsSectorStartaddress = 131072	`#define FLS_17_DMU_QSDATA_STARTADDRESS (0xaf020000U)`

1.1.28 Macro: FLS_17_DMU_QSDATA_SIZE

Table 28 FLS_17_DMU_QSDATA_SIZE

Name	FLS_17_DMU_QSDATA_SIZE	
Description	Specifies the size of the quasi static data sector in bytes.	
Verification method	This macro will be generated when 'FlsGeneral/FlsIfxFeeUse' is True and 'FlsConfigSet/FlsSectorList/FlsSector/FlsSector_0/FlsNumberOfSectors' is 1. The macro is generated with the 'FlsSector/FlsSectorSize' configured by the user. *Note:* *This macro is generated as hex converted numeric value of FlsSectorSize parameter.*	
Example(s)	**Action**	**Generated output**

	Add a container for ‘FlsSector’ and set the configuration parameters as below: FlsSector_0/FlsSectorSize = 262144	#define FLS_17_DMU_QSDATA_SIZE (0x40000U)

1.1.29 Macro: FlsConf_<X>_STARTADDRESS

Table 29 FlsConf_<X>_STARTADDRESS

Name	FlsConf_<X>_STARTADDRESS	
Description	Specifies the start address of each sector configured. <X> indicates the sector number according to the number of sectors configured by the user.	
Verification method	This macro will be generated when ‘FlsGeneral/FlsIfxFeeUse’ is False. The macro is generated by the addition of values in ‘FlsSector/ FlsSectorStartaddress’ and ‘FlsGeneral/FlsBaseAddress’. *Note: This macro is value is generated as hex converted value of the addition operation of the above two parameters.*	
Example(s)	**Action**	**Generated output**
	When number of FlsSector configured is 4 FlsSector_0/FlsSectorStartaddress = 0, FlsSector_1/FlsSectorStartaddress = 262144 FlsSector_2/FlsSectorStartaddress = 327680 FlsSector_3/FlsSectorStartaddress = 393216 FlsBaseAddress = 2936012800	#define FlsConf_FlsSector_0_STARTADDRESS (0xaf000000U) #define FlsConf_FlsSector_1_STARTADDRESS (0xaf040000U) #define FlsConf_FlsSector_2_STARTADDRESS (0xaf050000U) #define FlsConf_FlsSector_3_STARTADDRESS (0xaf060000U)

1.1.30 Macro: FLS_17_DMU_IFX_NVM_PRESENT

Table 30 FLS_17_DMU_IFX_NVM_PRESENT

Name	FLS_17_DMU_IFX_NVM_PRESENT	
Description	Specifies whether the double sector mode in IFX FEE is configured.	
Verification method	The macro is generated as STD_ON when the total number of sectors configured in ‘FlsConfigSet/FlsSectorList/FlsSector/FlsSector_0/FlsNumberOfSectors’ is 2 and ‘FlsConfigSet/FlsSectorList/FlsSector/FlsSector_1/FlsNumberOfSectors’ is 1 else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**
	Configure 2 sectors in ‘FlsSector’ container.	#define FLS_17_DMU_IFX_NVM_PRESENT (STD_ON)

	FlsSector_0/FlsNumberOfSectors = 2 FlsSector_1/FlsNumberOfSectors = 1	
	Configure 2 sectors in 'FlsSector' container. FlsSector_0/FlsNumberOfSectors = 1 FlsSector_1/FlsNumberOfSectors = 1	`#define FLS_17_DMU_IFX_NVM_PRESENT (STD_OFF)`

1.1.31 Macro: FLS_17_DMU_INIT_API_MODE

Table 31 FLS_17_DMU_INIT_API_MODE

Name	FLS_17_DMU_INIT_API_MODE	
Description	Specifies the operating mode for FLS initialization - SUPERVISOR/ USER1.	
Verification method	The macro is generated as FLS_17_DMU_MCAL_SUPERVISOR when 'FlsGeneral/FlsInitApiMode' is FLS_17_DMU_MCAL_SUPERVISOR else FLS_17_DMU_MCAL_USER1.	
Example(s)	**Action**	**Generated output**
	FlsInitApiMode = FLS_17_DMU_MCAL_SUPERVISOR	`#define FLS_17_DMU_INIT_API_MODE (FLS_17_DMU_MCAL_SUPERVISOR)`
	FlsInitApiMode = FLS_17_DMU_MCAL_USER1	`#define FLS_17_DMU_INIT_API_MODE (FLS_17_DMU_MCAL_USER1)`

1.1.32 Macro: FLS_17_DMU_RUNTIME_API_MODE

Table 32 FLS_17_DMU_RUNTIME_API_MODE

Name	FLS_17_DMU_RUNTIME_API_MODE	
Description	Specifies the operating mode for FLS runtime - SUPERVISOR/ USER1.	
Verification method	The macro is generated as FLS_17_DMU_MCAL_SUPERVISOR when 'FlsGeneral/FlsRuntimeApiMode' is FLS_17_DMU_MCAL_SUPERVISOR else FLS_17_DMU_MCAL_USER1	
Example(s)	**Action**	**Generated output**
	FlsRuntimeApiMode = FLS_17_DMU_MCAL_SUPERVISOR	`#define FLS_17_DMU_RUNTIME_API_MODE (FLS_17_DMU_MCAL_SUPERVISOR)`
	FlsRuntimeApiMode = FLS_17_DMU_MCAL_USER1	`#define FLS_17_DMU_RUNTIME_API_MODE (FLS_17_DMU_MCAL_USER1`

1.1.33 Macro: FLS_17_DMU_PAGE_SIZE

Table 33 FLS_17_DMU_PAGE_SIZE

Name	FLS_17_DMU_PAGE_SIZE	
Description	Specifies the page size of data flash.	
Verification method	The macro is generated with the value in 'FlsConfigSet/FlsSectorList/FlsSector FlsPageSize' configuration parameter. *Note: This macro is not configurable by the user.*	
Example(s)	**Action**	**Generated output**
		`#define FLS_17_DMU_PAGE_SIZE (8U)`

1.1.34 Macro: FLS_17_DMU_ERASE_TIME

Table 34 FLS_17_DMU_ERASE_TIME

Name	FLS_17_DMU_ERASE_TIME	
Description	Specifies the erase time in micro seconds.	
Verification method	The macro is generated with the value in 'FlsPublishedInformation/FlsEraseTime' configuration parameter. *Note: This macro is not configurable by the user.*	
Example(s)	**Action**	**Generated output**
		`#define FLS_17_DMU_ERASE_TIME (1725000U)`

1.1.35 Macro: FLS_17_DMU_WRITE_TIME

Table 35 FLS_17_DMU_WRITE_TIME

Name	FLS_17_DMU_WRITE_TIME	
Description	Specifies the write time in micro seconds.	
Verification method	The macro is generated with the value in 'FlsPublishedInformation/FlsWriteTime' configuration parameter. *Note: This macro is not configurable by the user.*	
Example(s)	**Action**	**Generated output**
		`#define FLS_17_DMU_WRITE_TIME (5140U)`

1.1.36 Macro: FLS_17_DMU_DEFAULT_ERASEVALUE

Table 36 FLS_17_DMU_DEFAULT_ERASEVALUE

Name	FLS_17_DMU_DEFAULT_ERASEVALUE	
Description	Specifies the default erase value.	
Verification method	The macro is generated with the value in 'FlsPublishedInformation/FlsErasedValue' configuration parameter. *Note: This macro is not configurable by the user.*	
Example(s)	**Action**	**Generated output**
		`#define FLS_17_DMU_DEFAULT_ERASEVALUE (0U)`

1.1.37 Macro: FLS_17_DMU_ERASESUSPEND_TIMEOUT

Table 37 FLS_17_DMU_ERASESUSPEND_TIMEOUT

Name	FLS_17_DMU_ERASESUSPEND_TIMEOUT	
Description	Specifies the timeout for erase suspend/resume feature.	
Verification method	The macro is generated when 'FlsIfxSpecificConfig/FlsUseEraseSuspend' is True. The macro is generated with the value in 'FlsIfxSpecificConfig/FlsEraseSuspendTimeout' configuration parameter. *Note: This macro is generated as hex converted numeric value based on the value set in FlsEraseSuspendTimeout parameter.*	
Example(s)	**Action**	**Generated output**

	FlsUseEraseSuspend = True
	FlsEraseSuspendTimeout = 10
		`#define FLS_17_DMU_ERASESUSPEND_TIMEOUT (0xaU)`

1.1.38 Macro: FLS_17_DMU_ECC_ERROR_INFO_API

Table 38 FLS_17_DMU_ECC_ERROR_INFO_API

Name	FLS_17_DMU_ECC_ERROR_INFO_API	
Description	This parameter indicates if Fls_17_Dmu_GetEccErrorPageAddress API should be enabled or disabled.	
Verification method	The macro is generated as STD_ON when 'FlsGeneral/FlsEccErrorInfoApi' and 'FlsGeneral/IfxFeeUsed' is True else the macro is generated as STD_OFF.	
Example(s)	**Action**	**Generated output**

	FlsEccErrorInfoApi = True
	IfxFeeUsed = False
		#define FLS_17_DMU_ECC_ERROR_INFO_API (STD_OFF)

	FlsEccErrorInfoApi = True
	IfxFeeUsed = True
		#define FLS_17_DMU_ECC_ERROR_INFO_API (STD_ON)

1.1.39 Macro: FLS_17_DMU_ERASE_VERIFICATION

Table 39 FLS_17_DMU_ERASE_VERIFICATION

Name	FLS_17_DMU_ERASE_VERIFICATION	
Description	This parameter indicates if the erase verification should be enabled or disabled. Erase verification here means: Comparing the contents of addressed memory area against the value of an erased flash cell whether the block has been completely erased before writing a flash block.	
Verification method	If ASR440 is active, this parameter will be set to STD_ON if 'FlsGeneral/FlsEraseVerificationEnabled' is enabled and 'FlsGeneral/IfxFeeUsed' is disabled. If ASR422 is active, this parameter will be set to STD_ON if 'FlsGeneral/FlsDevErrorDetect' or 'FlsGeneral/FlsSafetyEnable' is enabled and 'FlsGeneral/IfxFeeUsed' is disabled.	
Example(s)	**Action**	**Generated output**

	ArMinorVersion = 4
	FlsEraseVerificationEnabled= True
	IfxFeeUsed = False
		#define FLS_17_DMU_ERASE_VERIFICATION (STD_ON)

	ArMinorVersion = 2
	FlsDevErrorDetect = True
	IfxFeeUsed = True
		#define FLS_17_DMU_ERASE_VERIFICATION (STD_OFF)

1.1.40 Macro: FLS_17_DMU_ERASE_VERIFY_REPORT

Table 40 FLS_17_DMU_ERASE_VERIFY_REPORT

Name	FLS_17_DMU_ERASE_VERIFY_REPORT	
Description	This parameter indicates if the erase verification error reporting should be enabled or disabled.	
Verification method	If ASR440 is active, this parameter will be set to STD_ON if 'FlsGeneral/FlsEraseVerificationEnabled' and 'FlsGeneral/FlsRunTimeErrorDetect' are enabled. If ASR422 is active, this parameter will be set to STD_OFF.	
Example(s)	**Action**	**Generated output**

	ArMinorVersion = 4
	FlsEraseVerificationEnabled= True
		#define FLS_17_DMU_ERASE_VERIFY_REPORT (STD_ON)

	FlsRunTimeErrorDetect = True	
	ArMinorVersion = 2	#define FLS_17_DMU_ERASE_VERIFY_REPORT (STD_OFF)

1.1.41 Macro: FLS_17_DMU_TIMEOUT_SUPERVISION

Table 41 FLS_17_DMU_TIMEOUT_SUPERVISION

Name	FLS_17_DMU_TIMEOUT_SUPERVISION	
Description	This parameter indicates if the timeout monitoring should be enabled or disabled.	
Verification method	If ASR440 is active, this parameter will be set to STD_ON if ‘FlsGeneral/FlsTimeoutSupervisionEnabled’ is enabled. If ASR422 is active, this parameter will be set to STD_ON if ‘FlsGeneral/FlsDevErrorDetect’ or ‘FlsGeneral/FlsSafetyEnable’ is enabled.	
Example(s)	**Action**	**Generated output**

	ArMinorVersion = 4
	FlsTimeoutSupervisionEnabled = True
		#define
		FLS_17_DMU_TIMEOUT_SUPERVISION
		(STD_ON)

	ArMinorVersion = 2
	FlsDevErrorDetect = True
		#define
		FLS_17_DMU_TIMEOUT_SUPERVISION
		(STD_ON)

1.1.42 Macro: FLS_17_DMU_WRITE_VERIFICATION

Table 42 FLS_17_DMU_WRITE_VERIFICATION

Name	FLS_17_DMU_WRITE_VERIFICATION	
Description	This parameter indicates if the write verification error reporting should be enabled or disabled.	
Verification method	If ASR440 is active, this parameter will be set to STD_ON if ‘FlsGeneral/FlsWriteVerificationEnabled’ and ‘FlsGeneral/FlsRunTimeErrorDetect’ are enabled. If ASR422 is active, this parameter will be set to STD_OFF.	
Example(s)	**Action**	**Generated output**

	ArMinorVersion = 4
	FlsWriteVerificationEnabled = True
	FlsRunTimeErrorDetect = True
		#define
		FLS_17_DMU_WRITE_VERIFICATION
		(STD_ON)

	ArMinorVersion = 2	#define FLS_17_DMU_WRITE_VERIFICATION (STD_OFF)

1.1.43 Macro: Fls_17_DmuConf_FlsGeneral_FlsDriverIndex

Table 43 Fls_17_DmuConf_FlsGeneral_FlsDriverIndex

Name	Fls_17_DmuConf_FlsGeneral_FlsDriverIndex	
Description	Index of the driver, used by IFX FEE.	
Verification method	This macro in turn refers to the ‘FLS_17_DMU_INSTANCE_ID’ macro.	
Example(s)	**Action**	**Generated output**

		`#define`
		`Fls_17_DmuConf_FlsGeneral_FlsDriverIndex`
		`(FLS_17_DMU_INSTANCE_ID)`

1.2 File: Fls_17_Dmu[_<variant>]_PBcfg.c

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

1.2.1 Structure: Fls_17_Dmu_Config[_<variant>]

Table 44 Fls_17_Dmu_Config[_<variant>]

Name	Fls_17_Dmu_Config[_<variant>]	
Type	Fls_17_Dmu_ConfigType	
Description	Root configuration structure of FLS driver which will be used during initialization.	
Verification method	The generated structure is present in Fls_17_Dmu[_<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 FLS with FEE (variant unaware)	`const Fls_17_Dmu_ConfigType Fls_17_Dmu_Config =`
		`{`
		`    /* Fls state variable structure */`
		`    &FlsStateVar,`
		`    /* Maximum number of bytes to Read in one cycle */`
		`    /* Fast Mode */`
		`    64U,`

	/* Normal Mode */ 32U, /* Job End Notification */ &Fee_JobEndNotification, /* Job Error Notification */ &Fee_JobErrorNotification, /* EVER Notification */ &Fee_17_JobEraseErrorNotification, /* PVER Notification */ NULL_PTR, /* Illegal State Notification */ NULL_PTR, /*Wait state configuration for Read access and error correction */ (((uint32)FLS_17_DMU_WAIT_STATE_READACCESS9) | ((uint32)FLS_17_DMU_WAIT_STATE_ERRORCORRECTION1 << 16U)) , /* FlsCallCycle for timeout monitoring, convert to us by multiplying by 1000 * 1000 */ 10000U, /* Default mode of FLS driver */ MEMIF_MODE_SLOW };
Configure FLS with FEE. Variant name is ‘Petrol’)	const Fls_17_Dmu_ConfigType Fls_17_Dmu_Config_Petrol = { /* Fls state variable structure */ &FlsStateVar,

```
    /* Maximum number of bytes to Read in one
cycle */
    /* Fast Mode */
    64U,

    /* Normal Mode */
    32U,

    /* Job End Notification */
    &Fee_JobEndNotification,

    /* Job Error Notification */
    &Fee_JobErrorNotification,

 /* EVER Notification */
    &Fee_17_JobEraseErrorNotification,

/* PVER Notification */
    NULL_PTR,

    /* Illegal State Notification */
    NULL_PTR,

    /*Wait state configuration for Read access
and error correction */

(((uint32)FLS_17_DMU_WAIT_STATE_READACCESS9) |

((uint32)FLS_17_DMU_WAIT_STATE_ERRORCORRECTION1
<< 16U)) ,

    /* FlsCallCycle for timeout monitoring,
convert to us by multiplying by
       1000 * 1000 */
    10000U,

    /* Default mode of FLS driver */
    MEMIF_MODE_SLOW
};
```

1.2.1.1 Member: FlsStateVarPtr

Table 45 FlsStateVarPtr

<table>
<tr><td><b>Name</b></td><td colspan="2">FlsStateVarPtr</td></tr>
<tr><td><b>Type</b></td><td colspan="2">Fls_17_Dmu_StateType *</td></tr>
<tr><td><b>Description</b></td><td colspan="2">Pointer to the structure of FLS state variables.</td></tr>
<tr><td><b>Verification method</b></td><td colspan="2">The generated structure member is present in the Fls_17_Dmu_Config [_<variant>] structure. The member is generated with the value in ‘FlsIfxSpecificConfig/FlsStateVarStruct’ configuration parameter.
<i>Note: It is assumed that user will configure the member with correct name. When the field is filled incorrectly or left blank, the generation will not throw any error, however the compilation will fail.</i></td></tr>
<tr><td><b>Example(s)</b></td><td><b>Action</b></td><td><b>Generated output</b></td></tr>
<tr><td></td><td>FlsStateVarStruct = FlsStateVar</td><td><code>&FlsStateVar</code></td></tr>
</table>

1.2.1.2 Member: FlsFastRead

Table 46 FlsFastRead

<table>
<tr><td><b>Name</b></td><td colspan="2">FlsFastRead</td></tr>
<tr><td><b>Type</b></td><td colspan="2">Fls_17_Dmu_LengthType</td></tr>
<tr><td><b>Description</b></td><td colspan="2">Maximum number of bytes to read or compare in one cycle of the FLS driver's job processing function in fast mode.
<i>Note: The value configured for FlsMaxReadFastMode should be more than the value configured for FlsMaxReadNormalMode.</i></td></tr>
<tr><td><b>Verification method</b></td><td colspan="2">The generated structure member contains value configured in ‘FlsConfigSet /FlsMaxReadFastMode’ configuration parameter.</td></tr>
<tr><td><b>Example(s)</b></td><td><b>Action</b></td><td><b>Generated output</b></td></tr>
<tr><td></td><td>FlsMaxReadFastMode = 64</td><td><code>64U</code></td></tr>
</table>

1.2.1.3 Member: FlsSlowRead

Table 47 FlsSlowRead

Name	FlsSlowRead
Type	Fls_17_Dmu_LengthType
Description	Maximum number of bytes to read or compare in one cycle of the flash driver's job processing function in normal mode.

Verification method	The generated structure member contains value configured in 'FlsConfigSet/FlsMaxReadNormalMode' configuration parameter.	
Example(s)	**Action**	**Generated output**
	FlsMaxReadNormalMode = 32	`32U`

1.2.1.4 Member: FlsJobEndNotificationPtr

Table 48 FlsJobEndNotificationPtr

Name	FlsJobEndNotificationPtr	
Type	Fls_17_Dmu_NotifFunctionPtrType	
Description	Function pointer to the job end notification routine provided by upper layer module, typically the Fee module.	
Verification method	The structure member is generated with function name or address configured in the configuration container 'FlsJobEndNotification'. This will also generate a function declaration for valid value in 'FlsJobEndNotification' (Refer 1.2.2).	
Example(s)	**Action**	**Generated output**
	FlsJobEndNotification = 23245	Structure member: (Fls_17_Dmu_NotifFunctionPtrType)23245 Function declaration: Not generated
	FlsJobEndNotification = 0	Structure member: (Fls_17_Dmu_NotifFunctionPtrType)0U Function declaration: Not generated
	FlsJobEndNotification = NULL_PTR	Structure member: NULL_PTR Function declaration: Not generated
	FlsJobEndNotification = NULL	Structure member: NULL_PTR Function declaration: Not generated
	FlsJobEndNotification = <not configured>	Structure member: NULL_PTR Function declaration: Not generated
	FlsJobEndNotification = Fee_JobEndNotification	Structure member: &Fee_JobEndNotification Function declaration:

		`extern void Fee_JobEndNotification(void);`

1.2.1.5 Member: FlsJobErrorNotificationPtr

Table 49 FlsJobErrorNotificationPtr

Name	FlsJobErrorNotificationPtr	
Type	Fls_17_Dmu_NotifFunctionPtrType	
Description	Function pointer to the job error notification routine provided by upper layer module, typically the Fee module.	
Verification method	The structure member is generated with function name or address configured in the configuration container 'FlsJobErrorNotification'. This will also generate a function declaration for valid value in 'FlsJobErrorNotification' (Refer 1.2.2).	
Example(s)	**Action**	**Generated output**

	FlsJobErrorNotification = 23245	`Structure member:`
		`(Fls_17_Dmu_NotifFunctionPtrType)23245`
		`Function declaration:`
		`Not generated`

	FlsJobErrorNotification = 0	`Structure member:`
		`(Fls_17_Dmu_NotifFunctionPtrType)0U`
		`Function declaration:`
		`Not generated`

	FlsJobErrorNotification = NULL	`Structure member:`
		`NULL_PTR`
		`Function declaration:`
		`Not generated`

	FlsJobErrorNotification = <not configured>	`Structure member:`
		`NULL_PTR`
		`Function declaration:`
		`Not generated`

	FlsJobErrorNotification = NULL_PTR	`Structure member:`
		`NULL_PTR`
		`Function declaration:`
		`Not generated`

	FlsJobErrorNotification = Fee_JobErrorNotification	`Structure member:`
		`&Fee_JobErrorNotification`
		`Function declaration:`

		extern void Fee_JobErrorNotification(void);

1.2.1.6 Member: FlsEraseVerifyErrNotifPtr

Table 50 FlsEraseVerifyErrNotifPtr

Name	FlsEraseVerifyErrNotifPtr	
Type	Fls_17_Dmu_NotifFunctionPtrType	
Description	Function pointer to the EVER notification routine provided by upper layer module, typically the Fee module.	
Verification method	The structure member is generated with function name configured in the configuration container 'FlsEraseVerifyErrNotif'. This will also generate a function declaration for valid value in 'FlsEraseVerifyErrNotif' (Refer 1.2.2). *Note: This structure member is not editable by the user.*	
Example(s)	**Action**	**Generated output**

	When FlsEraseVerifyErrNotif is not configured	`Structure member:`
		`NULL_PTR`
		`Function declaration:`
		`Not generated`

	When FlsEraseVerifyErrNotif is configured	`Structure member:`
		`&Fee_17_JobEraseErrorNotification`
		`Function declaration:`
		`extern void Fee_17_JobEraseErrorNotification(void);`

1.2.1.7 Member: FlsProgVerifyErrNotifPtr

Table 51 FlsProgVerifyErrNotifPtr

Name	FlsProgVerifyErrNotifPtr	
Type	Fls_17_Dmu_NotifFunctionPtrType	
Description	Function pointer to the PVER notification routine provided by upper layer module, typically the Fee module.	
Verification method	The structure member is generated with function name configured in the configuration container 'FlsProgVerifyErrNotif'. This will also generate a function declaration for valid value in 'FlsProgVerifyErrNotif' (Refer 1.2.2). *Note: This structure member is not editable by the user.*	
Example(s)	**Action**	**Generated output**

	When FlsProgVerifyErrNotif is not configured	`Structure member:`
		`NULL_PTR`
		`Function declaration:`

		Not generated
	When FlsProgVerifyErrNotif is configured	Structure member: &Fee_17_JobProgErrorNotification Function declaration: extern void Fee_17_JobProgErrorNotification(void);

1.2.1.8 Member: FlsIllegalStateNotificationPtr

Table 52 FlsIllegalStateNotificationPtr

Name	FlsIllegalStateNotificationPtr	
Type	Fls_17_Dmu_NotifFunctionPtrType	
Description	Function pointer FLS Illegal state notification routine. Illegal state here means that FLS is not able to proceed. No more FLS request shall be triggered; Reset of the system is recommended.	
Verification method	The structure member is generated with function name or address configured in the configuration parameter 'FlsIfxSpecificConfig/FlsIllegalStateNotification'. This will also generate a function declaration for valid value in 'FlsIfxSpecificConfig/FlsIllegalStateNotification'. (Refer 1.2.2).	
Example(s)	**Action**	**Generated output**
	FlsIllegalStateNotification = 23245	Structure member: (Fls_17_Dmu_NotifFunctionPtrType)23245 Function declaration: Not generated
	FlsIllegalStateNotification = NULL	Structure member: NULL_PTR Function declaration: Not generated
	FlsIllegalStateNotification = <not configured>	Structure member: NULL_PTR Function declaration: Not generated
	FlsIllegalStateNotification = 0	Structure member: (Fls_17_Dmu_NotifFunctionPtrType)0U Function declaration: Not generated
	FlsIllegalStateNotification = FlsIllegalStateNotificationfun	Structure member: &FlsIllegalStateNotificationfun Function declaration:

		`extern void FlsIllegalStateNotificationfun(void);`

1.2.1.9 Member: FlsWaitStates

Table 53 FlsWaitStates

Name	FlsWaitStates	
Type	uint32	
Description	Wait state configuration for read access and error correction.	
Verification method	The structure member is generated as OR operation of 'FlsIfxSpecificConfig/FlsWaitStateRead' and 'FlsIfxSpecificConfig/FlsWaitStateErrorCorrection' configuration parameters.	
Example(s)	**Action**	**Generated output**

	FlsWaitStateRead = FLS_17_DMU_WAITSTATE_READ_9
	FlsWaitStateErrorCorrection = FLS_17_DMU_WAITSTATE_ERRCOREC_1
		`(((uint32)FLS_17_DMU_WAITSTATE_READ_9) | ((uint32)FLS_17_DMU_WAITSTATE_ERRCOREC_1 << 16U))`

1.2.1.10 Member: FlsCallCycle

Table 54 FlsCallCycle

Name	FlsCallCycle	
Type	uint32	
Description	Cycle time for call to the job processing main function (Fls_17_Dmu_MainFunction(void)).	
Verification method	The structure member is generated only when FLS_17_DMU_DEV_ERROR_DETECT is ON or FLS_17_DMU_SAFETY_ENABLE is ON. The value generated is derived by converting the value of 'FlsConfigSet/FlsCallCycle' (given in seconds) to micro seconds (multiplying by 10^6).	
Example(s)	**Action**	**Generated output**
	FlsCallCycle = 0.01	`10000U`
	FlsCallCycle = 0.0001	`100U`

1.2.1.11 Member: FlsDefaultMode

Table 55 FlsDefaultMode

Name	FlsDefaultMode	
Type	MemIf_ModeType	
Description	Default read mode of the data flash (DFLASH0) on the device after initialization.	
Verification method	The structure member is generated with the value provided in 'FlsConfigSet/FlsDefaultMode' configuration parameter.	
Example(s)	**Action**	**Generated output**

	FlsDefaultMode = MEMIF_MODE_SLOW	`MEMIF_MODE_SLOW`
	FlsDefaultMode = MEMIF_MODE_FAST	`MEMIF_MODE_FAST`

1.2.2 Function pointer: Fls_17_Dmu_NotifFunctionPtrType

Table 56 Fls_17_Dmu_NotifFunctionPtrType

Name	Fls_17_Dmu_NotifFunctionPtrType	
Type	Fls_17_Dmu_NotifFunctionPtrType*	
Description	The user defined notification function which would be invoked different instances of FLS job.	
Verification method	The extern declaration for the corresponding notification function(s) would be generated when a function name is provided in the configuration parameter. Below are the list of parameters against which it will generate an extern declaration: • ‘FlsJobEndNotification’ • ‘FlsJobErrorNotification’ • ‘FlsEraseVerifyErrNotif’ • ‘FlsProgVerifyErrNotif’ • ‘FlsIfxSpecificConfig/FlsIllegalStateNotification’. If address is provided as a numerical value or ‘null pointer’ to any of the above function parameters, then extern declaration will not be generated for the corresponding parameter.	
Example(s)	**Action**	**Generated output**
	FlsJobEndNotification = Fee_JobEndNotification	`extern void Fee_JobEndNotification(void);`
	FlsJobErrorNotification = Fee_JobErrorNotification	`extern void Fee_JobErrorNotification(void);`
	When FlsEraseVerifyErrNotif is configured	`extern void Fee_17_JobEraseErrorNotification(void);`
	When FlsProgVerifyErrNotif is configured	`extern void Fee_17_JobProgErrorNotification(void);`
	FlsIllegalStateNotification = FlsIllegalStateNotificationfun	`extern void FlsIllegalStateNotificationfun(void);`

1.3 File: Fls_17_Dmu[_<variant>]_PBcfg.h

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

1.3.1 Structure: Fls_17_Dmu_Config[_<variant>]

Table 57 Fls_17_Dmu_Config[_<varaint>]

<table>
<tr><td><b>Name</b></td><td colspan="2">Fls_17_Dmu_Config[_<variant>]</td></tr>
<tr><td><b>Type</b></td><td colspan="2">Fls_17_Dmu_ConfigType</td></tr>
<tr><td><b>Description</b></td><td colspan="2">Declaration of root configuration structure of FLS driver which will be used during initialization.</td></tr>
<tr><td><b>Verification method</b></td><td colspan="2">The generated structure is present in Fls_17_Dmu[_<variant>]_PBcfg.h file. The <variant> indicates the name of the post-build variant. For a variant-aware configuration the structure name is appended with the variant name. For variant-unaware configuration <variant> is ignored.</td></tr>
<tr><td rowspan="3"><b>Example(s)</b></td><td><b>Action</b></td><td><b>Generated output</b></td></tr>
<tr><td>Configure FLS and generate (variant-unaware)</td><td><code>extern const Fls_17_Dmu_ConfigType Fls_17_Dmu_Config;</code></td></tr>
<tr><td>Configure FLS and generate (variant-aware. Variant name is ‘Petrol’)</td><td><code>extern const Fls_17_Dmu_ConfigType Fls_17_Dmu_Config_Petrol;</code></td></tr>
</table>

Revision history

Major changes since the last revision

Date	Version	Description
2020-11-25	1.0	• Released.
2020-11-25	0.1	• Added following macros: FLS_17_DMU_ERASE_VERIFICATION, FLS_17_DMU_ERASE_VERIFY_REPORT, FLS_17_DMU_WRITE_VERIFICATION, FLS_17_DMU_TIMEOUT_SUPERVISION, FLS_17_DMU_ECC_ERROR_INFO_API and Fls_17_DmuConf_FlsGeneral_FlsDriverIndex • FLS 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 2020-11-25

Published by

Infineon Technologies AG

81726 Munich, Germany

© 2020 Infineon Technologies AG.
All Rights Reserved.

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.