Integration Manual

for S32K14X FEE Driver

Document Number: IM2FEEASR4.2 Rev0002R1.0.1
Rev. 1.0

Contents

Section number **Title** **Page**

Chapter 1
Revision History

Table 1-1. Revision History

Revision	Date	Author	Description
1.0	13/07/2018	NXP MCAL Team	Updated version for ASR 4.2.2S32K14X1.0.1 Release

Chapter 2
Introduction

This integration manual describes the integration requirements for FEE Driver for S32K14X microcontrollers.

2.1 Supported Derivatives

The software described in this document is intented to be used with the following microcontroller devices of NXP Semiconductors .

Table 2-1. S32K14X Derivatives

NXP Semiconductors	s32k148_lqfp144, s32k148_lqfp176, s32k148_mapbga100, s32k146_lqfp144, s32k146_lqfp100, s32k146_lqfp64, s32k146_mapbga100, s32k144_lqfp100, s32k144_lqfp64, s32k144_mapbga100, s32k142_lqfp100, s32k142_lqfp64, s32k118_lqfp48, s32k118_lqfp64

All of the above microcontroller devices are collectively named as S32K14X .

2.2 Overview

AUTOSAR (AUTomotive Open System ARchitecture) is an industry partnership working to establish standards for software interfaces and software modules for automobile electronic control systems.

AUTOSAR

- paves the way for innovative electronic systems that further improve performance, safety and environmental friendliness.
- is a strong global partnership that creates one common standard: "Cooperate on standards, compete on implementation".

- is a key enabling technology to manage the growing electrics/electronics complexity. It aims to be prepared for the upcoming technologies and to improve cost-efficiency without making any compromise with respect to quality.
- facilitates the exchange and update of software and hardware over the service life of the vehicle.

2.3 About this Manual

This Technical Reference employs the following typographical conventions:

Boldface type: Bold is used for important terms, notes and warnings.

Italic font: Italic typeface is used for code snippets in the text. Note that C language modifiers such "const" or "volatile" are sometimes omitted to improve readability of the presented code.

Notes and warnings are shown as below:

Note

This is a note.

2.4 Acronyms and Definitions

Table 2-2. Acronyms and Definitions

Term	Definition
API	Application Programming Interface
ASM	Assembler
AUTOSAR	Automotive Open System Architecture
BSMI	Basic Software Make file Interface
BSW	Basic Software
C/CPP	C and C++ Source Code
CAN	Controller Area Network
DEM	Diagnostic Event Manager
DET	Development Error Tracer
DFO	Data Flash Optimized
DW	Double Word
ECC	Error Correcting Code
ECU	Electronic Control Unit
EcuM	ECU Manager Module
EEPROM	Electrically Erasable Programmable Read-Only Memory

Table continues on the next page...

Table 2-2. Acronyms and Definitions (continued)

Term	Definition
FEE	Flash EEPROM Emulation Module/Driver
FLS	Flash memory driver
ISR	Interrupt Service Routine
IVOR	Interrupt Vector Offset Register
job	A FEE block operation passed to the module
MCU	Microcontroller Unit
MemIf	Memory Interface Module
N/A	Not Applicable
NvM	NVRAM Manager
NVRAM	Non-volatile RAM memory
OS	Operating System
RAM	Random Access Memory
SchM	Schedule Manager
VLE	Variable Length Encoding
VSMD	Vendor-Specific Module Definition
XML	Extensible Markup Language

2.5 Reference List

Table 2-3. Reference List

#	Title	Version
1	Specification of FEE Driver	AUTOSAR Release 4.2.2
2	S32K14X Reference Manual	Reference Manual, Rev. 4, 06/2017
3	S32K142 Mask Set Errata for Mask 0N33V (0N33V)	30/11/2017
4	S32K144 Mask Set Errata for Mask 0N57U (0N57U)	30/11/2017
5	S32K146 Mask Set Errata for Mask 0N73V (0N73V)	30/11/2017
6	S32K148 Mask Set Errata for Mask 0N20V (0N20V)	30/11/2017
7	S32K118 Mask Set Errata for Mask 0N97V (0N97V)	26/02/2018

Chapter 3
Building the Driver

This section describes the source files and various compilers, linker options used for building the Autosar FEE driver for NXP SemiconductorsS32K14X . It also explains the EB Tresos Studio plugin setup procedure.

3.1 Build Options

The FEE driver files are compiled using

- Green Hills Multi 7.1.4 / Compiler 2017.1.4
- (Linaro GCC 6.3-2017.06~dev) 6.3.1 20170509 (Thu Dec 7 13:28:42 CST 2017 build.sh rev=g7fea41d s=L631 Earmv7 -V release_g7fea41d_build_Fed_Earmv7)
- IAR: V8.11.2

The compiler, linker flags used for building the driver are explained below:

Note

The TS_T40D2M10I1R0 plugin name is composed as follow:

TS_T = Target_Id

D = Derivative_Id

M = SW_Version_Major

I = SW_Version_Minor

R = Revision

(i.e. Target_Id = 40 identifies CORTEXM architecture and Derivative_Id = 2 identifies the S32K14X)

3.1.1 GHS Compiler/Linker/Assembler Options

Table 3-1. Compiler Options

Option	Description
-cpu=cortexm4	Selects target processor: Arm Cortex M4
-ansi	Specifies ANSI C with extensions. This mode extends the ANSI X3.159-1989 standard with certain useful and compatible constructs.
-Osize	Optimize for size.
-dual_debug	Enables the generation of DWARF, COFF, or BSD debugging information in the object file
-G	Generates source level debugging information and allows procedure call from debugger's command line.
--no_exceptions	Disables support for exception handling
-Wundef	Generates warnings for undefined symbols in preprocessor expressions
-Wimplicit-int	Issues a warning if the return type of a function is not declared before it is called
-Wshadow	Issues a warning if the declaration of a local variable shadows the declaration of a variable of the same name declared at the global scope, or at an outer scope
-Wtrigraphs	Issues a warning for any use of trigraphs
-Wall	Enables all the warnings about constructions that some users consider questionable, and that are easy to avoid even in conjunction with macros.
--prototype_errors	Generates errors when functions referenced or called have no prototype
--incorrect_pragma_warnings	Valid #pragma directives with wrong syntax are treated as warnings
-noslashcomment	C++ like comments will generate a compilation error
-preprocess_assembly_files	Preprocesses assembly files
-nostartfile	Do not use Start files
--short_enum	Store enumerations in the smallest possible type
-c	Produces an object file (called input-file.o) for each source file.
--no_commons	Allocates uninitialized global variables to a section and initializes them to zero at program startup.
-keeptempfiles	Prevents the deletion of temporary files after they are used. If an assembly language file is created by the compiler, this option will place it in the current directory instead of the temporary directory. Produces an object file (called input-file.o) for each source file.
-list	Creates a listing by using the name of the object file with the .lst extension. Assembler option
-DAUTOSAR_OS_NOT_USED	-D defines a preprocessor symbol and optionally can set it to a value. AUTOSAR_OS_NOT_USED: By default in the package, the drivers are compiled to be used without Autosar OS. If the drivers are used with Autosar OS, the compiler option '-DAUTOSAR_OS_NOT_USED' must be removed from project options
-DDISABLE_MCAL_INTERMODULE_ASR_CHECK	-D defines a preprocessor symbol to disable the inter-module version check for AR_RELEASE versions. DISABLE_MCAL_INTERMODULE_ASR_CHECK: By default in the package, drivers are compiled to perform the inter-module version check as per Autosar BSW004. When the inter-module version check needs to be disabled then the DISABLE_MCAL_INTERMODULE_ASR_CHECK global define must be added to the list of compiler options.
-DGHS	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the GHS preprocessor symbol.

Table 3-2. Assembler Options

Option	Description
-cpu=cortexm4	Selects target processor: Arm Cortex M4
-c	Produces an object file (called input-file.o) for each source file.
-preprocess_assembly_files	Preprocesses assembly files
-asm=list	Creates a listing by using the name of the object file with the .lst extension. Assembler option

Table 3-3. Linker Options

Option	Description
-Mn	Map file numeric ordering
-delete	Removal from the executable of functions that are unused and unreferenced
-v	Display removed unused functions
-ignore_debug_references	Ignores relocations from DWARF debug sections when using -delete.
-map	Creates a detailed map file
-keepmap	Keep the map file in the event of a link error
-lstartup	Link libstartup library -Run-time environment startup routines
-lsys	Link libsys library -Run-time environment system routines
-larch	Link libarch library -Target-specific run-time support. Any file produced by the Green Hills Compiler may depend on symbols in this library.
-lansi	Link libansi library -the standard C library
-L(/lib/thumb2)	Link thumb2 library
-lutf8_s32	Include utf8_s32.a to use the Wide Character Functions

3.1.2 IAR Compiler/Linker/Assembler Options

Table 3-4. Compiler Options

Option	Description
--cpu=Cortex-M4	Selects target processor: Arm Cortex M4
--cpu_mode=thumb	Selects generating code that executes in Thumb state.
--endian=little	Specifies the endianess of core: little endian.
-Ohz	Sets the optimization level to High, favoring size.
-c	Produces an object file (called input-file.o) for each source file.
--no_clustering	Disables static clustering optimizations.
--no_mem_idioms	Makes the compiler to not optimize code sequences that clear, set, or copy a memory region.
--no_explicit_zero_opt	Places the zero initialized variables in data section instead of bss.
--debug	Makes the compiler include information in the object modules.
--diag_suppress=Pa050	Suppresses diagnostic messages (warnings) about non-standard line endings.

Table continues on the next page...

Table 3-4. Compiler Options (continued)

Option	Description
-DAUTOSAR_OS_NOT_USED	-D defines a preprocessor symbol and optionally can set it to a value. AUTOSAR_OS_NOT_USED: By default in the package, the drivers are compiled to be used without Autosar OS. If the drivers are used with Autosar OS, the compiler option '-DAUTOSAR_OS_NOT_USED' must be removed from project options
-DIAR	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the IAR preprocessor symbol.
--require_prototypes	Forces the compiler to verify that all functions have proper prototypes.
--no_wrap_diagnostics	Disables line wrapping of diagnostic messages issued by compiler.
--no_system_include	Disables the automatic search for system include files.
-e	Enables language extensions. This option is needed by FLS driver which uses _packed structures.

Table 3-5. Assembler Options

Option	Description
--cpu=Cortex-M4	Selects target processor: Arm Cortex M4
--cpu_mode=thumb	Selects generating code that executes in Thumb state.
-g	Use this option to disable the automatic search for system include files.

Table 3-6. Linker Options

Option	Description
--cpu=Cortex-M4	Selects target processor: Arm Cortex M4
--map filename	Produces a map file.
--no_library_search	Disables automatic runtime library search.
--entry _start	Treats the symbol _start as a root symbol and as the start of the application.
--enable_stack_usage	Enables stack usage analysis.
--skip_dynamic_initialization	Suppress dynamic initialization during system startup.
--no_wrap_diagnostics	Disables line wrapping of diagnostic messages issued by linker.
--config	Specifies the configuration file to be used by the linker.

3.1.3 GCC Compiler/Linker/Assembler Options

Table 3-7. Compiler Options

Option	Description
-c	Produces an object file (called input-file.o) for each source file.
-O1	Use optimization for size.
-ggdb3	Produce debugging information for use by GDB. Level 3 includes extra information, such as all the macro definitions present in the program.

Table continues on the next page...

Table 3-7. Compiler Options (continued)

Option	Description
-mcpu=cortex-m4	Selects target processor: Arm Cortex M4
-mthumb	Selects generating code that executes in Thumb state.
-mlittle-endian	Generate code for a processor running in little-endian mode.
-fomit-frame-pointer	Removes the frame pointer for all functions, which might make debugging harder.
-mhard-float	Use hardware floating-point instructions.
-fno-common	Specifies that the compiler should place uninitialized global variables in the data section of the object file, rather than generating them as common blocks.
-Wall	Enables all the warnings about constructions that some users consider questionable, and that are easy to avoid even in conjunction with macros.
-Wextra	Enables some extra warning flags that are not enabled by '-Wall'.
-Wstrict-prototypes	Warn if a function is declared or defined without specifying the argument types.
-Wno-sign-compare	Do not warn when a comparison between signed and unsigned values could produce an incorrect result when the signed value is converted to unsigned.
-DAUTOSAR_OS_NOT_USED	-D defines a preprocessor symbol and optionally can set it to a value. AUTOSAR_OS_NOT_USED: By default in the package, the drivers are compiled to be used without Autosar OS. If the drivers are used with Autosar OS, the compiler option '-DAUTOSAR_OS_NOT_USED' must be removed from project options
-DLINARO	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the LINARO preprocessor symbol.
-fstack-usage	Geneates an extra file that specifies the maximum amount of stack used, on a per-function basis.
-fdump-ipa-all	Enables all inter-procedural analysis dumps.
-Werror=implicit-function-declaration	Generates an error when the prototype of the function is not defined..

Table 3-8. Assembler Options

Option	Description
-mcpu=cortex-m4	Selects target processor: Arm Cortex M4
-c	Produces an object file (called input-file.o) for each source file.
-mthumb	This option specifies that the assembler should start assembling Thumb instructions.
-x assembler-with-cpp	Indicates that the assembly code contains C directives and the C preprocessor must be run.

Table 3-9. Linker Options

Option	Description
-Map=filename	Print a link map to the file mapfile.
-T scriptfile	Use scriptfile as the linker script. This script replaces ld's default linker script(rather than adding to it), so commandfile must specify everything necessary to describe the output file.

3.2 Files Required for Compilation

This section describes the include files required to compile, assemble (if assembler code) and link the FEE driver for S32K14X microcontrollers.

To avoid integration of incompatible files, all the include files from other modules shall have the same AR_MAJOR_VERSION and AR_MINOR_VERSION, i.e. only files with the same AUTOSAR major and minor versions can be compiled.

FEE Files

- ..\Fee_TS_T40D2M10I1R0\src\Fee.c
- ..\Fee_TS_T40D2M10I1R0\include\Fee_Cbk.h
- ..\Fee_TS_T40D2M10I1R0\include\Fee.h
- ..\Fee_TS_T40D2M10I1R0\include\Fee_InternalTypes.h
- ..\Fee_TS_T40D2M10I1R0\include\Fee_Types.h
- Fee_Cfg.h - this file should be generated by the user using a configuration/generation tool
- Fee_Cfg.c - this file should be generated by the user using a configuration/generation tool

Other includes files:

Files from the MemIf folder:

- ..\MemIf_TS_T40D2M10I1R0\include\MemIf_Types.h

Files from the Base common folder

- ..\Base_TS_T40D2M10I1R0\include\Compiler.h
- ..\Base_TS_T40D2M10I1R0\include\Compiler_Cfg.h
- ..\Base_TS_T40D2M10I1R0\include\Fee_MemMap.h
- ..\Base_TS_T40D2M10I1R0\include\Platform_Types.h
- ..\Base_TS_T40D2M10I1R0\include\Mcal.h
- ..\Base_TS_T40D2M10I1R0\include\Std_Types.h

Files from the Det folder:

- ..\Det_TS_T40D2M10I1R0\include\Det.h

Files from the Fls folder:

- ..\Fls_TS_T40D2M10I1R0\include\Fls.h

3.3 Setting up the Plug-ins

The FEE driver was designed to be configured by using the EB tresos® (version EB tresos Studio 23.0.0 b170330-0431 or later.)

- VSMD (Vendor Specific Module Definition) file in EB tresos® XDM format:
  - `..\Fee_TS_T40D2M10I1R0\config\Fee.xdm`
- VSMD (Vendor Specific Module Definition) file in AUTOSAR compliant EPD format:
  - `..\Fee_TS_T40D2M10I1R0\autosar\Fee.epd`
- Code generation templates for pre-compile time configuration parameters:
  - `..\Fee_TS_T40D2M10I1R0 \generate\include\Fee_Cfg.h`
  - `..\Fee_TS_T40D2M10I1R0 \generate\src\Fee_Cfg.c`

Steps to generate the configuration:

1. Copy the module folders `Fee_TS_T40D2M10I1R0`, `Fls_TS_T40D2M10I1R0`, `Base_TS_T40D2M10I1R0`, `Resource_TS_T40D2M10I1R0`, `EcuC_TS_T40D2M10I1R0` into the EB tresos® `plugins` folder.
2. Set the desired output location folder for the generated sources and header files.
3. Use the EB tresos® GUI to modify ECU configuration parameters values.
4. Generate the configuration files.

Chapter 4
Function calls to module

4.1 Function Calls During Start-up

FEE shall be initialized during STARTUP2 phase of EcuM initialization. The API member to be called to accomplish this is Fee_Init.

Notes:

- Fee module is the upper layer module which works on FLS module.
- Fls_Init function must be called before calling the Fee_Init.
- Fee_MainFunction and Fls_MainFunction routines must be called repeatedly for the FEE module initialization and its operation. When an operation (initialization or standard one) finishes, the Fee_GetStatus returns MEMIF_IDLE.

4.2 Function Calls During Shutdown

None.

4.3 Function Calls During Wake-up

None.

Chapter 5
Module requirements

5.1 Exclusive areas to be defined in BSW scheduler

In the current implementation, FEE is using the services of Run-Time Environment (RTE) for entering and exiting the critical regions. RTE implementation is done by the integrators of the MCAL using OS or non-OS services. For testing the FEE, stubs are used for RTE.

FEE driver has four exclusive areas (EA) FEE_EXCLUSIVE_AREA_00, FEE_EXCLUSIVE_AREA_01, FEE_EXCLUSIVE_AREA_02, and FEE_EXCLUSIVE_AREA_03. The purpose of these exclusive areas is to make the functions Fee_Read, Fee_Write, Fee_InvalidateBlock and Fee_EraseImmediateBlock thread safe and thus protect FEE internal job variables.

Critical Region Exclusive Matrix

Below is the table depicting the exclusivity between different critical region IDs from the FEE driver. If there is an “X” in a table, it means that those 2 critical regions cannot interrupt each other.

FEE_EXCLUSIVE_AREA_00 Used in Fee_Read.

FEE_EXCLUSIVE_AREA_01 Used in Fee_Write.

FEE_EXCLUSIVE_AREA_02 Used in Fee_InvalidateBlock.

FEE_EXCLUSIVE_AREA_03 Used in Fee_EraseImmediateBlock.

Table 5-1. Exclusive Areas

	FEE_EXCLUSIVE_AREA_00	FEE_EXCLUSIVE_AREA_01	FEE_EXCLUSIVE_AREA_02	FEE_EXCLUSIVE_AREA_03
FEE_EXCLUSIVE_AREA_00		x	x	x

Table continues on the next page...

Table 5-1. Exclusive Areas (continued)

	FEE_EXCLUSIVE_AREA_00	FEE_EXCLUSIVE_AREA_01	FEE_EXCLUSIVE_AREA_02	FEE_EXCLUSIVE_AREA_03
FEE_EXCLUSIVE_AREA_01	x		X	X
FEE_EXCLUSIVE_AREA_02	x	x		X
FEE_EXCLUSIVE_AREA_03	x	x	X	

5.2 Peripheral Hardware Requirements

The FEE module is hardware independent module and depends on the underlying FLS module and its configuration only.

5.3 ISR to Configure within OS – Dependencies

None.

5.4 ISR Macro

None.

5.5 Other AUTOSAR Modules Dependencies

- **Base:** The BASE module contains the common files/definitions needed by all MCAL modules.
- **Det:** The DET module is used for enabling Development error detection. The API function used is Det_ReportError(). The activation / deactivation of Development error detection is configurable using the ‘FeeDevErrorDetect’ configuration parameter.
- **MemIf:** This module allows the NVRAM manager to access several memory abstraction modules.
- **Resource:** Resource module is used to select microcontroller’s derivatives.

- **Fls:** The flash driver provides services for reading, writing and erasing flash memory and a configuration interface for modifying the write/erase protection if supported by the underlying hardware.
- **EcuC:** The ECUC module is used for ECU configuration. MCAL modules need ECUC to retrieve the variant information
- **Rte:** The RTE module is needed for implementing data consistency of exclusive areas that are used by FEE module.

5.6 Data Cache Restriction

To achieve the same robustness as in the previous platforms, due to ECC page restriction and CACHE coherency issues, it is recommended to set the FeeVirtualPageSize to 16 bytes. For more details about ECC and cache coherency issues please see the FLS driver integration manual.

5.7 User Mode support

The Fee driver can be run in user mode

Chapter 6
Main API Requirements

6.1 Main Function Calls Within BSW Scheduler

- Fee_MainFunction()
- Fls_MainFunction()

Call rate depends on target application, i.e. how fast the data must be read/written/ compared.

6.2 API Requirements

Before calling the Fee_Write() function for immediate data, the function Fee_EraseImmediateBlock() must be called to pre-erase the flash area.

6.3 Calls to Notification Functions, Callbacks, Callouts

The FEE module provides user-configurable notifications

- FeeNvMJobEndNotification,
- FeeNvMJobErrorNotification,
- FeeClusterFormatNotification.

FeeNvMJobEndNotification and FeeNvMJobErrorNotification are usually routed to the NvM. FeeClusterFormatNotification is called by Fee to inform the user in case a cluster format is triggered during the Fee initialization.

Additionally, the FEE module publishes two APIs

- Fee_JobEndNotification: This callback notification is used by the underlying FLS module to report the successful end of an FLS operation.
- Fee_JobErrorNotification: This callback notification is used by the underlying FLS module to report the failure of an FLS operation.

Both callbacks must be configured in the FLS module (notifications) regardless of its operation mode (synchronous or asynchronous).

Chapter 7
Memory Allocation

7.1 Sections to Be Defined in Fee_MemMap.h

For precompile:

```
#ifdef FEE_START_SEC_CONST_UNSPECIFIED
#undef FEE_START_SEC_CONST_UNSPECIFIED
#undef MEMMAP_ERROR
/*no definition -> default compiler settings are used */
#endif
#ifdef FEE_STOP_SEC_CONST_UNSPECIFIED
#undef FEE_STOP_SEC_CONST_UNSPECIFIED
#undef MEMMAP_ERROR
/*no definition -> default compiler settings are used */
#endif
```

For code:

```
#ifdef FEE_START_SEC_CODE
#undef FEE_START_SEC_CODE
#undef MEMMAP_ERROR
/*no definition -> default compiler settings are used */
#endif
#ifdef FEE_STOP_SEC_CODE
#undef FEE_STOP_SEC_CODE
#undef MEMMAP_ERROR
/*no definition -> default compiler settings are used */
#endif
```

For variables:

```
#ifdef FEE_START_SEC_VAR_INIT_8
#undef FEE_START_SEC_VAR_INIT_8
#undef MEMMAP_ERROR
/*no definition -> default compiler settings are used */
#endif
#ifdef FEE_STOP_SEC_VAR_INIT_8
#undef FEE_STOP_SEC_VAR_INIT_8
#undef MEMMAP_ERROR
/*no definition -> default compiler settings are used */
```

```
#endif



#ifdef FEE_START_SEC_VAR_INIT_16
#undef FEE_START_SEC_VAR_INIT_16
#undef MEMMAP_ERROR
/*no definition -> default compiler settings are used */
#endif
#ifdef FEE_STOP_SEC_VAR_INIT_16
#undef FEE_STOP_SEC_VAR_INIT_16
#undef MEMMAP_ERROR
/*no definition -> default compiler settings are used */
#endif



#ifdef FEE_START_SEC_VAR_INIT_UNSPECIFIED
#undef FEE_START_SEC_VAR_INIT_UNSPECIFIED
#undef MEMMAP_ERROR
/*no definition -> default compiler settings are used */
#endif
#ifdef FEE_STOP_SEC_VAR_INIT_UNSPECIFIED
#undef FEE_STOP_SEC_VAR_INIT_UNSPECIFIED
#undef MEMMAP_ERROR
/*no definition -> default compiler settings are used */
#endif



#ifdef FEE_START_SEC_VAR_NO_INIT_UNSPECIFIED
#undef FEE_START_SEC_VAR_NO_INIT_UNSPECIFIED
#undef MEMMAP_ERROR
/*no definition -> default compiler settings are used */
#endif
#ifdef FEE_STOP_SEC_VAR_NO_INIT_UNSPECIFIED
#undef FEE_STOP_SEC_VAR_NO_INIT_UNSPECIFIED
#undef MEMMAP_ERROR
/*no definition -> default compiler settings are used */
#endif
```

7.2 Linker Command File

Memory shall be allocated for every section defined in `Fee_MemMap.h`.

Chapter 8
Configuration parameters considerations

Configuration parameter class for Autosar FEE driver fall into the following variants as defined below:

8.1 Configuration Parameters

Configuration parameter class for AUTOSAR FEE driver fall into the following variants as defined below:

Table 8-1. Configuration Parameters

Configuration List-Container	Configuration Parameters	Configuration Variant	Current Implementation
FeeGeneral			
	FeeDevErrorDetect	VariantPreCompile	PreCompile
	FeeMainFunctionPeriod	VariantPreCompile	PreCompile
	FeeNvmJobEndNotification	VariantPreCompile	PreCompile
	FeeNvmJobErrorNotification	VariantPreCompile	PreCompile
	FeeClusterFormatNotification	VariantPreCompile	PreCompile
	FeePollingMode	VariantPreCompile	PreCompile
	FeeSetModeSupported	VariantPreCompile	PreCompile
	FeeVersionInfoApi	VariantPreCompile	PreCompile
	FeeVirtualPageSize	VariantPreCompile	PreCompile
	FeeDataBufferSize	VariantPreCompile	PreCompile
	FeeBlockAlwaysAvailable	VariantPreCompile	PreCompile
	FeeLegacyMode	VariantPreCompile	PreCompile
	FeeLegacyEraseMode	VariantPreCompile	PreCompile
	FeeSwapForeignBlocksEnabled	VariantPreCompile	PreCompile
	FeeMarkEmptyBlocksInvalid	VariantPreCompile	PreCompile
	FeeConfigAssignment	VariantPreCompile	PreCompile
	FeeMaximumNumberBlocks	VariantPreCompile	PreCompile
FeeBlockConfiguration			

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration List-Container	Configuration Parameters	Configuration Variant	Current Implementation
	FeeClusterGroupRef	VariantPreCompile	PreCompile
	FeeBlockNumber	VariantPreCompile	PreCompile
	FeeBlockSize	VariantPreCompile	PreCompile
	FeeImmediateData	VariantPreCompile	PreCompile
	FeeBlockAssignment	VariantPreCompile	PreCompile
	FeeNumberOfWriteCycles	VariantPreCompile	PreCompile
	FeeDeviceIndex	VariantPreCompile	PreCompile
FeeClusterGroup			
	FeeCluster	VariantPreCompile	PreCompile
FeeCluster			
	FeeSector	VariantPreCompile	PreCompile
FeeSector			
	FeeSectorRef	VariantPreCompile	PreCompile
	FeeSectorIndex	VariantPreCompile	PreCompile

Chapter 9
Integration Steps

This section gives a brief overview of the steps needed for integrating Flash EEPROM Emulation :

- Generate the required FEE configurations. For more details refer to section Files Required for Compilation
- Allocate proper memory sections in FEE_MemMap.h and linker command file. For more details refer to section Sections to Be Defined in MemMap.h
- Compile & build the FEE with all the dependent modules. For more details refer to section Building the Driver

9.1 ISR Reference

None.

Chapter 10
External Assumptions for FEE driver

The section presents requirements that must be complied with when integrating FEE driver into the application.

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Document Number IM2FEEASR4.2 Rev0002R1.0.1
Revision 1.0

AUTOSAR