Integration Manual

for MPC574XG FEE Driver

Document Number: IM35FEEASR4.2 Rev0002R1.0.0
Rev. 4.0.0

Contents

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

Chapter 1
Revision History

Chapter 2
Introduction

Chapter 3
Building the Driver

Chapter 4
Function calls to module

Chapter 5
Module requirements

Chapter 6
Main API Requirements

Chapter 7
Memory Allocation

Chapter 8
Configuration parameters considerations

Chapter 9
Integration Steps

Chapter 10
External Assumptions for FEE driver

Chapter 1
Revision History

Table 1-1. Revision History

Revision	Date	Author	Description
1.0.0	21-Feb-2014	Thai Duong Ba	Initial Release for MPC574XG Beta 0.9.1
1.0.1	01/09/2014	Jan Chochola	Added information related to modified immediate data handling and its configuration.
1.0.2	17/09/2014	Jan Chochola	Updated information related to FeeVirtualPageSize in context of optimized ECC handling in the FLS and FEE module callbacks (IM).
2.0.0	23/04/2015	Livia Firan	Updated for the Calypso RTM 1.0.0
2.0.1	07/07/2015	Quy Nguyen	Updated for the Calypso RTM 1.0.1
2.0.2	08/08/2016	Livia Firan	Updated to add support for the SWAP FOREIGN BLOCKS feature.
3.0.0	08/08/2016	Livia Firan	Updated for the Calypso RTM 1.0.2 release
4.0.0	10/02/2017	Khoa Dang	Updated for Calypso ASR_4.2 RTM_1.0.0 release

Chapter 2
Introduction

This integration manual describes the integration requirements for FEE Driver for MPC574XG microcontrollers.

2.1 Supported Derivatives

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

Table 2-1. MPC574XG Derivatives

NXP Semiconductor	MPC5748G_LQFP176, MPC5748G_MAPBGA256, MPC5748G_MAPBGA324, MPC5747G_LQFP176, MPC5747G_MAPBGA256, MPC5747G_MAPBGA324, MPC5746G_LQFP176, MPC5746G_MAPBGA256, MPC5746G_MAPBGA324, MPC5748C_LQFP176, MPC5748C_MAPBGA256, MPC5748C_MAPBGA324, MPC5747C_LQFP176, MPC5747C_MAPBGA256, MPC5747C_MAPBGA324, MPC5746C_LQFP176, MPC5746C_MAPBGA256, MPC5746C_MAPBGA324, MPC5746C_MAPBGA100, MPC5745C_LQFP176, MPC5745C_MAPBGA256, MPC5745C_MAPBGA100, MPC5744C_LQFP176, MPC5744C_MAPBGA256, MPC5744C_MAPBGA100, MPC5746B_LQFP176, MPC5746B_MAPBGA256, MPC5746B_MAPBGA100, MPC5744B_LQFP176, MPC5744B_MAPBGA256,

Table 2-1. MPC574XG Derivatives

	MPC5744B_MAPBGA100, MPC5745B_LQFP176, MPC5745B_MAPBGA256, MPC5745B_MAPBGA100

All of the above microcontroller devices are collectively named as MPC574XG .

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
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	AUTOSAR 4.2 Rev0002FEE Driver Software Specification Document.	4.2 Rev0002
2	MPC5748G Reference Manual	Rev. 5, 12/2016
3	MPC5746C Reference Manual	Rev. 4, 12/2016
4	MPC5748G_1N81M_Rev.2 (official document) (1N81M)	Jun-16
5	MPC5748G_1N81M_0N78S_Comparison_Summary_v2_0 (internal document) (1N81M, 0N78S)	31.10.2016
6	MPC5746C_1N06M_Rev.4 (official document) (1N06M)	Jul-16
7	MPC5746C_cut1.1_cut2.0_cut2.1_comparison_v0 (internal document) (1N06M, 0N84S, 1N84S)	14-Sep-16
8	C3M_cut2.1_new_errata_20170113 (internal document) (1N84S)	13-Jan-17

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 SemiconductorMPC574XG . It also explains the EB Tresos Studio plugin setup procedure.

3.1 Build Options

The FEE driver files are compiled using

- Windriver DIAB DIAB_5_9_6_2
- Green Hills Multi 7.1.4 / Compiler 2015.1.6

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

Note

The TS_T2D35M10I0R0 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 = 2 identifies PA architecture and Derivative_Id = 35 identifies the MPC574XG)

3.1.1 GHS Compiler/Linker/Assembler Options

Table 3-1. Compiler Options

Option	Description
-cpu=ppc5748gz4204	Selects target processor: ppc5748gz4204
-cpu=ppc5748gz210	Selects target processor: ppc5748gz210
-ansi	Specifies ANSI C with extensions. This mode extends the ANSI X3.159-1989 standard with certain useful and compatible constructs.
-noSPE	Disables the use of SPE and vector floating point instructions by the compiler.
-Ospace	Optimize for size.
-sda=0	Enables the Small Data Area optimization with a threshold of 0.
-vle	Enables VLE code generation
-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
--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
--diag_error 223	Sets the specified compiler diagnostic messages to the level of error
-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
-DUSE_SW_VECTOR_MODE	-D defines a preprocessor symbol and optionally can set it to a value. USE_SW_VECTOR_MODE: By default in the package, drivers are compiled to be used with interrupt controller configured to be in hardware vector mode. In case of AUTOSAR_OS_NOT_USED, the compiler option "-DUSE_SW_VECTOR_MODE" must be added to the list of compiler options to be used with interrupt controller configured to be in software vector mode.
-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.
-c	Produces an object file (called input-file.o) for each source file.

Table 3-2. Assembler Options

Option	Description
-cpu=ppc5748gz4204	Selects target processor: ppc5748gz4204
-cpu=ppc5748gz210	Selects target processor: ppc5748gz210
-G	Generates source level debugging information and allows procedure call from debugger's command line.
-list	Creates a listing by using the name of the object file with the .lst extension

Table 3-3. Linker Options

Option	Description
-cpu=ppc5748gz4204	Selects target processor: ppc5748gz4204
-cpu=ppc5748gz210	Selects target processor: ppc5748gz210
-nostartfiles	Do not use Start files.
-vle	Enables VLE code generation
--nocpp	Do not Generate Constructors/Destructors
-Mn	sort numerically the MAP file
-delete	The -delete option instructs the linker to remove functions that are not referenced in the final executable.
-ignore_debug_references	Ignores relocations from DWARF debug sections when using -delete. DWARF debug information will contain references to deleted functions that may break some third-party debuggers.
-keepmap	keeps the MAP file in case of link error

3.1.2 DIAB Compiler/Linker/Assembler Options

Table 3-4. Compiler Options

Option	Description
-tPPCE200Z4204N3VEN:simple	Sets target processor to PPCE200Z4204N3VEN, generates ELF using EABI conventions, No floating point support (minimizes the required runtime), selects simple environment settings for Startup Module and Libraries
-tPPCE200Z210N3VEN:simple	Sets target processor to PPCE200Z210N3VEN, generates ELF using EABI conventions, No floating point support (minimizes the required runtime), selects simple environment settings for Startup Module and Libraries
-Xdialect-ansi	Follow the ANSI C standard with some additions
-XO	Enables extra optimizations to produce highly optimized code
-g3	Generate symbolic debugger information and do all optimizations.
-Xsize-opt	Optimize for size rather than speed when there is a choice
-Xsmall-data=0	Set Size Limit for 'small data' Variables to zero.
-Xsmall-const=0	Set Size Limit for “small const” Variables to zero.
-Xaddr-sconst=0x11	Specify addressing for constant static and global variables with size less than or equal to -Xsmall-const to far-absolute.

Table continues on the next page...

Table 3-4. Compiler Options (continued)

Option	Description
-Xaddr-sdata=0x11	Specify addressing for non-constant static and global variables with size less than or equal to -Xsmall-data in size to far-absolute.
-Xno-common	Disable use of the 'COMMON' feature so that the compiler or assembler will allocate each uninitialized public variable in the .bss section for the module defining it, and the linker will require exactly one definition of each public variable
-Xnested-interrupts	Allow nested interrupts
-Xdebug-dwarf2	Generate symbolic debug information in dwarf2 format
-Xdebug-local-all	Force generation of type information for all local variables
-Xdebug-local-cie	Create common information entry per module
-Xdebug-struct-all	Force generation of type information for all typedefs, struct, union and class types
-Xforce-declarations	Generates warnings if a function is used without a previous declaration
-ee1481	Generate an error when the function was used before it has been declared
-Xmacro-undefined-warn	Generates a warning when an undefined macro name occurs in a #if preprocessor directive
-Xlink-time-lint	Enable the checking of object and function declarations across compilation units, as well as the consistency of compiler options used to compile source files
-W:as:,-l	Pass the option '-l' (lower case letter L) to the assembler to get an assembler listing file
-Wa,-Xisa-vle	Instruct the assembler to expect and assemble VLE (Variable Length Encoding) instructions rather than BookE instructions.
-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
-DUSE_SW_VECTOR_MODE	-D defines a preprocessor symbol and optionally can set it to a value. USE_SW_VECTOR_MODE: By default in the package, drivers are compiled to be used with interrupt controller configured to be in hardware vector mode. In case of AUTOSAR_OS_NOT_USED, the compiler option "-DUSE_SW_VECTOR_MODE" must be added to the list of compiler options to be used with interrupt controller configured to be in software vector mode.
-DDIAB	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the DIAB preprocessor symbol.
-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.
-c	Stop after assembly, produce object file.

Table 3-5. Assembler Options

Option	Description
-tPPCE200Z4204N3VEN:simple	Sets target processor to PPCE200Z4204N3VEN, generates ELF using EABI conventions, No floating point support (minimizes the required runtime), selects simple environment settings for Startup Module and Libraries

Table continues on the next page...

Table 3-5. Assembler Options (continued)

Option	Description
-tPPCE200Z210N3VEN:simple	Sets target processor to PPCE200Z210N3VEN, generates ELF using EABI conventions, No floating point support (minimizes the required runtime), selects simple environment settings for Startup Module and Libraries
-g	Dump the symbols in the global symbol table in each archive file.
-Xisa-vle	Expect and assemble VLE (Variable Length Encoding) instructions rather than Book E instructions. The default code section is named .text_vle instead of .text, and the default code section fill "character" is set to 0x44444444 instead of 0. The .text_vle code section will have ELF section header flags marking it as VLE code, not Book E code.
-Xasm-debug-on	Generate debug line and file information
-Xdebug-dwarf2	Generate symbolic debug information in dwarf2 format
-Xsemi-is-newline	Treat the semicolon (;) as a statement separator instead of a comment character.

Table 3-6. Linker Options

Option	Description
-tPPCE200Z4204N3VEN:simple	Sets target processor to tPPCE200Z4204N3VEN, generates ELF using EABI conventions, No floating point support (minimizes the required runtime), selects simple environment settings for Startup Module and Libraries
-tPPCE200Z210N3VEN:simple	Sets target processor to tPPCE200Z210N3VEN, generates ELF using EABI conventions, No floating point support (minimizes the required runtime), selects simple environment settings for Startup Module and Libraries
-Xelf	Generates ELF object format for output file
-m6	Generates a detailed link map and cross reference table
-Xlink-time-lint	Enable the checking of object and function declarations across compilation units, as well as the consistency of compiler options used to compile source files

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 MPC574XG 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_T2D35M10I0R0\src\Fee.c`
- `..\Fee_TS_T2D35M10I0R0\include\Fee_Cbk.h`
- `..\Fee_TS_T2D35M10I0R0\include\Fee.h`
- `..\Fee_TS_T2D35M10I0R0\include\Fee_InternalTypes.h`

- `..\Fee_TS_T2D35M10I0R0\include\Fee_Types.h`
- `..\Fee_TS_T2D35M10I0R0\include\Fee_Version.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_T2D35M10I0R0\include\MemIf_Types.h`

Files from the Base common folder

- `..\Base_TS_T2D35M10I0R0\include\Compiler.h`
- `..\Base_TS_T2D35M10I0R0\include\Compiler_Cfg.h`
- `..\Base_TS_T2D35M10I0R0\include\Fee_MemMap.h`
- `..\Base_TS_T2D35M10I0R0\include\Platform_Types.h`
- `..\Base_TS_T2D35M10I0R0\include\Mcal.h`
- `..\Base_TS_T2D35M10I0R0\include\Std_Types.h`

Files from the Det folder:

- `..\Det_TS_T2D35M10I0R0\include\Det.h`

Files from the Fls folder:

- `..\Fls_TS_T2D35M10I0R0\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 21.0.0 b160607-0933 or later.)

- VSMD (Vendor Specific Module Definition) file in EB tresos® XDM format:
  - `..\Fee_TS_T2D35M10I0R0\config\Fee.xdm`
- VSMD (Vendor Specific Module Definition) file in AUTOSAR compliant EPD format:
  - `..\Fee_TS_T2D35M10I0R0\autosar\Fee.epd`
- Code generation templates for pre-compile time configuration parameters:
  - `..\Fee_TS_T2D35M10I0R0 \generate\include\Fee_Cfg.h`
  - `..\Fee_TS_T2D35M10I0R0 \generate\src\Fee_Cfg.c`

Steps to generate the configuration:

1. Copy the module folders `Fee_TS_T2D35M10I0R0`, `Fls_TS_T2D35M10I0R0`, `Base_TS_T2D35M10I0R0`, `Resource_TS_T2D35M10I0R0`, `EcuC_TS_T2D35M10I0R0` 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:** This module provides basic data types and auxiliary macros or functions commonly used by other AUTOSAR modules.
- **Det:** This module is necessary for development error tracking. The API function used is Det_ReportError(). The activation/deactivation of the Development Error Tracker is configurable using FeeDevErrorDetect configuration parameter.
- **MemIf:** Memory Interface
- **Resource:** Sub-derivative model is selected from Resource configuration.
- **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:** This module is required for configuring the variant handling in Tresos.

5.6 Data Cache Restriction

If data cache is enabled, even if the flash region is configured as cache-inhibited in the memory protection unit, an ECC event might be reported, if there is an ECC affected location inside the same cache line as the location read by driver. This means that an ECC affected location might falsely trigger and report ECC errors for any other location inside the same cache line. To achieve the same robustness as in the previous platforms, the FeeVirtualPageSize should be updated accordingly, to the same size as the cache line (4 DW, 32 bytes). For more details about ECC and cache coherency issues please see the FLS driver integration manual.

5.7 User Mode support

Fee driver is an independent hardware module, so it can run in user mode without any specific measure.

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.

[FEE047]

<< Values for production code Event Ids are assigned externally by the configuration of the Dem. They are published in the file Dem_IntErrId.h and included via Dem.h. >>

NOTE

FEE doesn't have any production error

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer’s technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.