Integration Manual

for S32K14X MCU Driver

Document Number: IM2MCUASR4.3 Rev0001R1.0.1
Rev. 1.0

Contents

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

Chapter 1
Revision History

Table 1-1. Revision History

Revision	Date	Author	Description
1.0	21/06/2019	NXP MCAL Team	Updated version for ASR 4.3.1S32K14XR1.0.1

Chapter 2
Introduction

This integration manual describes the integration requirements for Mcu 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, s32k142_lqfp48, s32k144_lqfp48, s32k148_lqfp100

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
AUTOSAR	Automotive Open System Architecture
ASM	Assembler
BSMI	Basic Software Make file Interface
CAN	Controller Area Network
DEM	Diagnostic Event Manager
DET	Default Error Tracer
C/CPP	C and C++ Source Code
VLE	Variable Length Encoding
N/A	Not Applicable
MCU	Micro Controller Unit

2.5 Reference List

Table 2-3. Reference List

#	Title	Version
1	Specification of Mcu Driver	AUTOSAR Release 4.3.1
2	S32K14X Reference Manual	Reference Manual, Rev. 9, 9/2018
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)	25/10/2018
7	S32K118 Mask Set Errata for Mask 0N97V (0N97V)	07/01/2019

Chapter 3
Building the Driver

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

3.1 Build Options

The Mcu 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 (Wed Jan 24 16:21:45 CST 2018 build.sh rev=g27a1317 s=L631 Earmv7 -V release_g27a1317_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
-cpu=cortexm0plus	Selects target processor: Arm Cortex M0+
-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
-cpu=cortexm0plus	Selects target processor: Arm Cortex M0+
-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=Cortex-M0+	Selects target processor: Arm Cortex M0+
--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.

Table continues on the next page...

Table 3-4. Compiler Options (continued)

Option	Description
--diag_suppress=Pa050	Suppresses diagnostic messages (warnings) about non-standard line endings.
-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=Cortex-M0+	Selects target processor: Arm Cortex M0+
--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
--cpu=Cortex-M0+	Selects target processor: Arm Cortex M0+
--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.
-Os	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.
-mcpu=cortex-m4	Selects target processor: Arm Cortex M4
-mcpu=cortex-m0plus	Selects target processor: Arm Cortex M0+
-mthumb	Selects generating code that executes in Thumb state.
-ansi	Specifies ANSI C with extensions.
-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.
-msoft-float	Use software 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.
-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..
-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
-DGCC	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the GCC preprocessor symbol.
-std=c99	C programming language standard version c99

Table 3-8. Assembler Options

Option	Description
-mcpu=cortex-m4	Selects target processor: Arm Cortex M4
-mcpu=cortex-m0plus	Selects target processor: Arm Cortex M0+
-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.
--disable-newlib-supplied-syscalls -specs=nosys.specs	These options support for using newlib on core M0+
-u _printf_float -u _scanf_float	These options support generating profile report.
-nostartfiles	Do not use the standard system startup files when linking
-e _start	Specify that the program entry point is _start
-static	The --static flag tells the linker to link a static, not a dynamically linked
-lc	The -lc flag tells the linker to link this binary against the C library, which is newlib in our case.
-lnosys	The -lnosys flag tells the linker to link this binary against the "nosys" library
$(TOOLCHAIN_DIR)/arm-none-eabi/newlib/lib/thumb/v6-m $(TOOLCHAIN_DIR)/lib/gcc/arm-none-eabi/6.3.1/thumb/v6-m	Library for core M0+, added with -L and -B option
$(TOOLCHAIN_DIR)/arm-none-eabi/newlib/lib/thumb $(TOOLCHAIN_DIR)/arm-none-eabi/newlib/lib)	Library for core M4, added with -L and -B option

3.2 Files required for Compilation

This section describes the include files required to compile, assemble (if assembler code) and link the Mcu 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.

Mcu Files

- Mcu_TS_T40D2M10I1R0\src\Mcu.c
- Mcu_TS_T40D2M10I1R0\src\Mcu.h
- Mcu_TS_T40D2M10I1R0\include\Mcu_EnvCfg.h
- Mcu_TS_T40D2M10I1R0\src\Mcu_IPW.c
- Mcu_TS_T40D2M10I1R0\src\Mcu_IPW_Irq.c
- Mcu_TS_T40D2M10I1R0\include\Mcu_IPW.h
- Mcu_TS_T40D2M10I1R0\include\Mcu_IPW_Irq.h
- Mcu_TS_T40D2M10I1R0\include\Mcu_IPW_Types.h
- Mcu_TS_T40D2M10I1R0\src\Mcu_PCC.c

- Mcu_TS_T40D2M10I1R0\include\Mcu_PCC.h
- Mcu_TS_T40D2M10I1R0\include\Mcu_PCC_Types.h
- Mcu_TS_T40D2M10I1R0\include\Reg_eSys_PCC.h
- Mcu_TS_T40D2M10I1R0\src\Mcu_SCG.c
- Mcu_TS_T40D2M10I1R0\include\Mcu_SCG.h
- Mcu_TS_T40D2M10I1R0\include\Mcu_SCG_Types.h
- Mcu_TS_T40D2M10I1R0\include\Reg_eSys_SCG.h
- Mcu_TS_T40D2M10I1R0\src\Mcu_RCM.c
- Mcu_TS_T40D2M10I1R0\src\Mcu_RCM_Irq.c
- Mcu_TS_T40D2M10I1R0\include\Mcu_RCM.h
- Mcu_TS_T40D2M10I1R0\include\Mcu_RCM_Types.h
- Mcu_TS_T40D2M10I1R0\include\Reg_eSys_RCM.h
- Mcu_TS_T40D2M10I1R0\src\Mcu_CMU.c
- Mcu_TS_T40D2M10I1R0\src\Mcu_CMU_Irq.c
- Mcu_TS_T40D2M10I1R0\include\Mcu_CMU.h
- Mcu_TS_T40D2M10I1R0\include\Mcu_CMU_Types.h
- Mcu_TS_T40D2M10I1R0\include\Reg_eSys_CMU.h
- Mcu_TS_T40D2M10I1R0\include\Mcu_CMU_IPVersion.h
- Mcu_TS_T40D2M10I1R0\src\Mcu_SMC.c
- Mcu_TS_T40D2M10I1R0\include\Mcu_SMC.h
- Mcu_TS_T40D2M10I1R0\include\Mcu_SMC_Types.h
- Mcu_TS_T40D2M10I1R0\include\Reg_eSys_SMC.h
- Mcu_TS_T40D2M10I1R0\src\Mcu_SIM.c
- Mcu_TS_T40D2M10I1R0\include\Mcu_SIM.h
- Mcu_TS_T40D2M10I1R0\include\Mcu_SIM_Types.h
- Mcu_TS_T40D2M10I1R0\include\Reg_eSys_SIM.h
- Mcu_TS_T40D2M10I1R0\src\Mcu_PMC.c
- Mcu_TS_T40D2M10I1R0\src\Mcu_PMC_Irq.c
- Mcu_TS_T40D2M10I1R0\include\Mcu_PMC.h
- Mcu_TS_T40D2M10I1R0\include\Mcu_PMC_Types.h
- Mcu_TS_T40D2M10I1R0\include\Reg_eSys_PMC.h
- Mcu_TS_T40D2M10I1R0\include\Mcu_PMC_IPVersion.h
- Mcu_TS_T40D2M10I1R0\include\Mcu_CortexM4.c
- Mcu_TS_T40D2M10I1R0\include\Reg_eSys_CortexM4.h
- Mcu_TS_T40D2M10I1R0\include\Mcu_CortexM4.h

Mcu Generated Files

- Mcu_TS_T40D2M10I1R0\generate_PC\src\Mcu_Cfg.c - This file should be generated by the user using a configuration tool for compilation

- Mcu_TS_T40D2M10I1R0\generate_PC\include\Mcu_Cfg.h - This file should be generated by the user using a configuration tool for compilation
- Mcu_TS_T40D2M10I1R0\generate_PB\src\Mcu_[VariantName]_PBcfg.c - This file should be generated by the user using a configuration tool for compilation. The file contains the definition of the init pointer for the respective variant. This is used to export the init configuration pointer of Variant [variant] to be used as parameter for Mcu_Init. To have access to the configuration pointer include the file in your project.

Note

As a deviation from standard:

- Mcu_[VariantName]_PBcfg.c - This file will contain the definition for all parameters that are variant aware, independent of the configuration class that will be selected (PC, PB)
- Mcu_Cfg.c - This file will contain the definition for all configuration structures containing only variables that are not variant aware, configured and generated only once. This file alone does not contain the whole structure needed by Mcu_Init function to configure the driver. Based on the number of variants configured in the EcuC, there can be more than one configuration structure for one module even for PreCompile variant.

Files from Base common folder

- Base_TS_T40D2M10I1R0\include\Compiler.h
- Base_TS_T40D2M10I1R0\include\Compiler_Cfg.h
- Base_TS_T40D2M10I1R0\include\ComStack_Cfg.h
- Base_TS_T40D2M10I1R0\include\ComStack_Types.h
- Base_TS_T40D2M10I1R0\include\Mcal.h
- Base_TS_T40D2M10I1R0\include\Mcu_MemMap.h
- Base_TS_T40D2M10I1R0\include\Platform_Types.h
- Base_TS_T40D2M10I1R0\include\Reg_eSys.h
- Base_TS_T40D2M10I1R0\include\StdRegMacros.h
- Base_TS_T40D2M10I1R0\include\Soc_Ips.h
- Base_TS_T40D2M10I1R0\include\Std_Types.h
- Base_TS_T40D2M10I1R0\generate_PC\include\modules.h

Files from Dem folder:

- Dem_TS_T40D2M10I1R0\include\Dem.h
- Dem_TS_T40D2M10I1R0\include\Dem_Types.h

- Dem_TS_T40D2M10I1R0\generate_PC\include\Dem_IntErrId.h
- Dem_TS_T40D2M10I1R0\src\Dem.c

Files from Det folder:

- Det_TS_T40D2M10I1R0\include\Det.h
- Det_TS_T40D2M10I1R0\src\Det.c

Files from Rte folder:

- Rte_TS_T40D2M10I1R0\include\SchM_Mcu.h
- Rte_TS_T40D2M10I1R0\src\SchM_Mcu.c

3.3 Setting up the Plug-ins

The Mcu driver was designed to be configured by using the EB Tresos Studio (version EB tresos Studio 24.0.1 b180321-0610 or later).

Location of various files inside the Mcu module folder:

- VSMD (Vendor Specific Module Definition) file in EB tresos Studio XDM format:
  - ..\Mcu_TS_T40D2M10I1R0 \config\Mcu.xdm
- Pre-configuration file in EB tresos Studio XDM form containing values for reset configuration parameters:
  - ..\Mcu_TS_T40D2M10I1R0 \config_ext\McuPreConfiguration.xdm
- VSMD (Vendor Specific Module Definition) file(s) in AUTOSAR compliant EPD format:
  - ..\ Mcu_TS_T40D2M10I1R0 \autosar\Mcu_<subderivative_name>.epd
- Code Generation Templates for parameters without variation points:
  - ..\Mcu_TS_T40D2M10I1R0\generate_PC\src\Mcu_Cfg.c
  - ..\Mcu_TS_T40D2M10I1R0\generate_PC\include\Mcu_Cfg.h
  - ..\Mcu_TS_T40D2M10I1R0\generate_PC\Mcu_checkvalues.m
  - ..\Mcu_TS_T40D2M10I1R0\generate_PC\Mcu_RegOperations.m
- Code Generation Templates for variant aware parameters:
  - ..\Mcu_TS_T40D2M10I1R0\generate_PB\src\Mcu_PBcfg_[variant].c
  - ..\Mcu_TS_T40D2M10I1R0\generate_PB\include\Mcu_PBcfg_[variant].h
  - ..\Mcu_TS_T40D2M10I1R0\generate_PB\Mcu_checkvalues.m
  - ..\Mcu_TS_T40D2M10I1R0\generate_PB\Mcu_RegOperations.m
- Code Generation Templates for Pre-Compile/Post-Build time configuration parameters:
  - None

Steps to generate the configuration:

1. Copy the module folders Base_ TS_T40D2M10I1R0 , Resource_ TS_T40D2M10I1R0 , Rte_ TS_T40D2M10I1R0 , Mcu_ TS_T40D2M10I1R0 , Dem_ TS_T40D2M10I1R0 , EcuC_ TS_T40D2M10I1R0 , Det_ TS_T40D2M10I1R0 into the Tresos plugins folder.
2. Set the desired Tresos Output location folder for the generated sources and header files.
3. Use the EB tresos Studio 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

The first BSW module to be initialized after "Power-On" shall be MCU. The MCU shall be initialized in the following sequence:

1. Mcu_Init()
2. Mcu_InitClock()
3. Mcu_GetPllStatus() - Till PLL is locked.
4. Mcu_DistributePllClock()
5. Mcu_InitRamSection() - If required

4.2 Function Calls during Shutdown

Mcu_SetMode (sleep mode) API shall be called during GO SLEEP phase of the EcuM's Shutdown state to configure the hardware for Sleep mode. This shall be called after ICU & GPT are set to sleep.

4.3 Function Calls during Wake-up

None.

Chapter 5
Module requirements

5.1 Exclusive areas to be defined in BSW scheduler

None

No exclusive areas are used for the Mcu driver.

5.2 Peripheral Hardware Requirements

None

5.3 ISR to configure within OS – dependencies

Table 5-1. MCU ISRs

ISR Name	Hardware interrupt vector	
	S32K14X	S32K11X
Mcu_PMC_LowVoltage_ISR	20	None
Mcu_RCM_AlternateResetIsr	23	23
Mcu_PMC_SCG_CMU_Isr	None	21

5.4 ISR Macro

MCAL drivers use the ISR macro to define the functions that will process hardware interrupts. Depending on whether the OS is used or not, this macro can have different definitions:

a. OS is not used - AUTOSAR_OS_NOT_USED is defined:

i. If USE_SW_VECTOR_MODE is defined:

```
#define ISR(IsrName) void IsrName(void)
```

In this case, drivers' interrupt handlers are normal C functions and the prolog/epilog handle the context save and restore.

ii. If USE_SW_VECTOR_MODE is not defined:

```
#define ISR(IsrName) INTERRUPT_FUNC void IsrName(void)
```

In this case, drivers' interrupt handlers must save and restore the execution context.

Custom OS is used - AUTOSAR_OS_NOT_USED is not defined

```
#define ISR(IsrName) void OS_isr_##IsrName()
```

In this case, OS is handling the execution context when an interrupt occurs. Drivers' interrupt handlers are normal C functions.

Other vendor's OS is used - AUTOSAR_OS_NOT_USED is not defined. Please refer to the OS documentation for description of the ISR macro.

5.5 Other AUTOSAR modules - dependencies

- **Det:** The DET module is used for enabling Default Error Tracing. The API function used is Det_ReportError(). The activation/deactivation of Default Error Tracing is configurable using the 'McuDevErrorDetect' configuration parameter.
- **Dem:** This module is necessary for enabling reporting of production relevant error status. The API function used is Dem_ReportErrorStatus(). The activation/ deactivation of DEM is configurable using the 'McuDisableDemReportErrorStatus' configuration parameter.
- **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 MCU module. The module is the realization (for a particular ECU) of the interfaces of the AUTOSAR Virtual Function Bus (VFB) and thus provides the infrastructure services for communication between Application Software Components as well as facilitating access to basic software components including the OS

- **Base:** The BASE module contains the common files/definitions needed by the MCAL. This means that it is a dependency for all other MCAL modules.
- **Resource:** Resource module is used to select microcontroller's derivatives.

5.6 Data cache restriction

None

5.7 User Mode support

The **Mcu** can be run from user mode if **McuEnableUserModeSupport** is enabled in the configuration.

All registers of the MCU module can only be written in supervisor mode, so all functions that write to registers will be called as trusted functions.

The following functions of Mcu must be called as trusted functions to access the registers in supervisor mode:

- **Mcu_CM4_EnableDeepSleep**
- **Mcu_CM4_DisableDeepSleep**
- **Mcu_CM4_SystemReset**
- **Mcu_PCC_PeripheralConfig**
- **Mcu_PMC_PowerInit**
- **Mcu_RCM_SystemResetIsrConfig**
- **Mcu_RCM_RestoreSystemResetIsrConfig**
- **Mcu_RCM_ResetInit**
- **Mcu_RCM_GetResetReason**
- **Mcu_RCM_GetResetRawValue**
- **Mcu_SCG_DropSysClkToSircInRunMode**
- **Mcu_SCG_DisableFircClock**
- **Mcu_SCG_DisableSoscClock**
- **Mcu_SCG_DropSystemClockToSirc**
- **Mcu_SCG_SircInit**
- **Mcu_SCG_FircInit**
- **Mcu_SCG_SoscInit**
- **Mcu_SCG_SpllInit**
- **Mcu_SCG_SystemClockInit**
- **Mcu_SCG_DisableClockMonitors**
- **Mcu_SCG_DisableSpllClock**

- **Mcu_SIM_Init**
- **Mcu_SIM_ClockConfig**
- **Mcu_SIM_SRAMRetentionConfig**
- **Mcu_SMC_AllowedModesConfig**
- **Mcu_SMC_ModeConfig**

Chapter 6
Main API Requirements

6.1 Main functions calls within BSW scheduler

None.

6.2 API Requirements

None

6.3 Calls to Notification Functions, Callbacks, Callouts

McuResetCallout from **Mcu_PerformReset()** .

McuErrorIsrNotification The callout configured by the user for error ISR notifications.

Chapter 7
Memory Allocation

7.1 Sections to be defined in Mcu_MemMap.h

Table 7-1. Memory Allocation

Section name	Type of section	Description
MCU_START_SEC_CONFIG_DATA_UNSPECIFIED	Configuration Data	Start of Memory Section for Config Data
MCU_STOP_SEC_CONFIG_DATA_UNSPECIFIED	Configuration Data	End of Memory Section for Config Data
MCU_START_SEC_CONST_UNSPECIFIED	Configuration Data	Start of Memory Section for Config Data that is not variant aware
MCU_STOP_SEC_CONST_UNSPECIFIED	Configuration Data	End of Memory Section for Config Data that is not variant aware
MCU_START_SEC_CODE	Code	Start of memory Section for Code
MCU_STOP_SEC_CODE	Code	End of memory Section for Code
MCU_START_SEC_RAMCODE	Code	Start of memory Section for Code to be located in Ram
MCU_STOP_SEC_RAMCODE	Code	End of memory Section for Code to be located in Ram
MCU_START_SEC_VAR_NO_INIT_UNSPECIFIED	Variables	Used for variables, structures, arrays when the SIZE (alignment) does not fit the criteria of 8,16 or 32 bit. These variables are never cleared and never initialized by start-up code.
MCU_STOP_SEC_VAR_NO_INIT_UNSPECIFIED	Variables	End of above section.
MCU_START_SEC_VAR_INIT_32	Variables	Used for variables which have to be aligned to 32 bit. For instance used for variables of size 32 bit or used for composite data types: arrays ,structs containing elements of maximum 32 bits. These variables are initialized with values after every reset.
MCU_STOP_SEC_VAR_INIT_32	Variables	End of above section.
MCU_START_SEC_VAR_INIT_UNSPECIFIED	Variables	Used for variables, structures, arrays, when the SIZE (alignment) does not fit the criteria

Table continues on the next page...

Table 7-1. Memory Allocation (continued)

Section name	Type of section	Description
		of 8,16 or 32 bit. These variables are initialized with values after every reset.
MCU_STOP_SEC_VAR_INIT_UNSPECIFIED	Variables	End of above section.
MCU_START_SEC_CONST_32	Constant Data	Used for constants that have to be aligned to 32 bit.
MCU_STOP_SEC_CONST_32	Constant Data	End of above section.

7.2 Linker command file

Memory shall be allocated for every section defined in Mcu_MemMap.h

Chapter 8
Configuration parameters considerations

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

8.1 Configuration Parameters

Specifies whether the configuration parameter shall be of configuration class Post Build.

Table 8-1. Configuration Parameters

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
Mcu	IMPLEMENTATION_CONFIG_VARIANT		
McuGeneralConfiguration	McuDevErrorDetect		
	McuVersionInfoApi		
	McuGetRamStateApi		
	McuInitClock		
	McuNoPll		
	McuEnterLowPowerMode		
	McuTimeout		
	McuEnableUserModeSupport		
	McuPerformResetApi		
	McuCalloutBeforePerformReset		
	McuPerformResetCallout		
	McuErrorIsrNotification		
McuDebugConfiguration	McuDisableDemReportErrorStatus		
	McuGetPeriphStateApi		
	McuGetMidrStructureApi		
McuPublishedInformation/ McuResetReasonConf	McuResetReason		

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
CommonPublishedInformation	ArReleaseMajorVersion		
	ArReleaseMinorVersion		
	ArReleaseRevisionVersion		
	ModuleId		
	SwMajorVersion		
	SwMinorVersion		
	SwPatchVersion		
	VendorApiInfix		
	VendorId		
McuModuleConfiguration	McuNumberOfMcuModes		
	McuRamSectors		
	McuResetSetting		
	McuRTCCLKINFrequencyHz		
	McuRtcClkSelect		
	McuLPOClockSelect		
	McuLPO32KClockEnable		
	McuLPO1KClockEnable		
	McuClockSrcFailureNotification		
McuModuleConfiguration/ McuAllowedModes	McuAllowHighSpeedRunMode		
	McuAllowVeryLowPowerModes		
McuModuleConfiguration/ McuSIMConfig/ McuChipControlConfiguration	McuEnableAdcSupplyMonitoring		
	McuAdcSupply		
	McuPDBBackToBackSelect		
	McuPTB14InterleaveChannelSelect		
	McuPTB13InterleaveChannelSelect		
	McuPTB1InterleaveChannelSelect		
	McuPTB0InterleaveChannelSelect		
McuModuleConfiguration/ McuSIMConfig/ McuFlexTimerConfiguration	McuFTM3ExternalClockPinSelect		
	McuFTM2ExternalClockPinSelect		
	McuFTM1ExternalClockPinSelect		

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
	McuFTM0ExternalClockPinSelect		
	McuFTM7ExternalClockPinSelect		
	McuFTM6ExternalClockPinSelect		
	McuFTM5ExternalClockPinSelect		
	McuFTM4ExternalClockPinSelect		
	McuFTM3Fault0Select		
	McuFTM3Fault1Select		
	McuFTM3Fault2Select		
	McuFTM2Fault0Select		
	McuFTM2Fault1Select		
	McuFTM2Fault2Select		
	McuFTM1Fault0Select		
	McuFTM1Fault1Select		
	McuFTM1Fault2Select		
	McuFTM0Fault0Select		
	McuFTM0Fault1Select		
	McuFTM0Fault2Select		
	McuFTM3Ch0ModulationSelect		
	McuFTM3Ch1ModulationSelect		
	McuFTM3Ch2ModulationSelect		
	McuFTM3Ch3ModulationSelect		
	McuFTM3Ch4ModulationSelect		
	McuFTM3Ch5ModulationSelect		
	McuFTM3Ch6ModulationSelect		
	McuFTM3Ch7ModulationSelect		
	McuFTM0Ch0ModulationSelect		
	McuFTM0Ch1ModulationSelect		
	McuFTM0Ch2ModulationSelect		

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
	McuFTM0Ch3ModulationSelect		
	McuFTM0Ch4ModulationSelect		
	McuFTM0Ch5ModulationSelect		
	McuFTM0Ch6ModulationSelect		
	McuFTM0Ch7ModulationSelect		
	McuFTM2Ch1InputSelect		
	McuFTM2Ch0InputSelect		
	McuFTM1Ch0InputSelect		
	McuFTMGlobalLoadEnable		
	McuFTM7SyncBit		
	McuFTM6SyncBit		
	McuFTM5SyncBit		
	McuFTM4SyncBit		
	McuFTM3SyncBit		
	McuFTM2SyncBit		
	McuFTM1SyncBit		
	McuFTM0SyncBit		
	McuQspiClkSelect		
	McuRMII_ClkSelect		
	McuRMII_Clk_OBE		
	McuFTM7OBEControl		
	McuFTM6OBEControl		
	McuFTM5OBEControl		
	McuFTM4OBEControl		
	McuFTM3OBEControl		
	McuFTM2OBEControl		
	McuFTM1OBEControl		
	McuFTM0OBEControl		
	McuFTM_GTBControl		
McuModuleConfiguration/ McuSIMConfig/ McuAdcOptionsConfiguration	McuADC1PreTrigeerSourceSelect		
	McuADC1SoftwarePreTrigeerSourceSelect		
	McuADC1TrigeerSourceSelect		
	McuADC0PreTrigeerSourceSelect		

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
	McuADC0SoftwarePreTrigeer SourceSelect		
	McuADC0TrigeerSourceSelec t		
	McuSoftwareTriggerToTRGM UX		
McuModuleConfiguration/ McuClockSettingConfig	McuClockSettingId		
	McuSysClockUnderMcuContr ol		
	McuScgClkOutSelect		
McuModuleConfiguration/ McuClockSettingConfig/ McuRunClockConfig	McuPreDivSystemClockFrequ ency		
	McuCoreClockFrequency		
	McuSystemClockFrequency		
	McuBusClockFrequency		
	McuFlashClockFrequency		
	McuSystemClockSwitch		
	McuCoreClockDivider		
	McuBusClockDivider		
	McuSlowClockDivider		
	McuScgClkOutFrequency		
McuModuleConfiguration/ McuClockSettingConfig/ McuVlprClockConfig	McuPreDivSystemClockFrequ ency		
	McuCoreClockFrequency		
	McuSystemClockFrequency		
	McuBusClockFrequency		
	McuFlashClockFrequency		
	McuSystemClockSwitch		
	McuCoreClockDivider		
	McuBusClockDivider		
	McuSlowClockDivider		
	McuScgClkOutFrequency		
McuModuleConfiguration/ McuClockSettingConfig/ McuHsrunClockConfig	McuPreDivSystemClockFrequ ency		
	McuCoreClockFrequency		
	McuSystemClockFrequency		
	McuBusClockFrequency		
	McuFlashClockFrequency		
	McuSystemClockSwitch		
	McuCoreClockDivider		

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
	McuBusClockDivider		
	McuSlowClockDivider		
	McuScgClkOutFrequency		
McuModuleConfiguration/ McuClockSettingConfig/ McuSystemOSCClockConfig	McuSOSCUnderMcuControl		
	McuSOSCFrequency		
	McuSOSCDiv2Frequency		
	McuSOSCDiv1Frequency		
	McuSOSCEnable		
	McuSOSCClockMonitorReset Enable		
	McuSOSCClockMonitorEnabl e		
	McuSOSCDiv2		
	McuSOSCDiv1		
	McuSOSCRangeSelect		
	McuSOSCHighGainOscillator Select		
	McuSOSCExternalReference Select		
McuModuleConfiguration/ McuClockSettingConfig/ McuSIRCClockConfig	McuSIRCUnderMcuControl		
	McuSIRCFrequency		
	McuSIRCDiv2Frequency		
	McuSIRCDiv1Frequency		
	McuSIRCEnable		
	McuSIRCLowPowerEnable		
	McuSIRCStopEnable		
	McuSIRCDiv2		
	McuSIRCDiv1		
	McuSIRCRangeSelect		
McuModuleConfiguration/ McuClockSettingConfig/ McuFIRCClockConfig	McuFIRCUnderMcuControl		
	McuFIRCFrequency		
	McuFIRCDiv2Frequency		
	McuFIRCDiv1Frequency		
	McuFIRCEnable		
	McuFIRCRegulatorEnable		
	McuFIRCDiv2		
	McuFIRCDiv1		
	McuFIRCRangeSelect		

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
McuModuleConfiguration/ McuClockSettingConfig/ McuSystemPll	McuSystemPllUnderMcuControl		
	McuSPLLFrequency		
	McuSPLLDiv2Frequency		
	McuSPLLDiv1Frequency		
	McuSPLLEnable		
	McuSPLLClockMonitorResetEnable		
	McuSPLLClockMonitorEnable		
	McuSPLLDiv2		
	McuSPLLDiv1		
	McuSPLLInputClkPreDivider		
	McuSPLLInputFrequency		
	McuSPLLMultiplier		
McuModuleConfiguration/ McuClockSettingConfig/ McuSIMClockConfig	McuDebugTraceDividerEnable		
	McuTraceClockDivider		
	McuTraceClockFraction		
	McuTraceClockSelect		
	McuClockOutEnable		
	McuClockOutDivider		
	McuClockOutSelect		
	McuEIMClockGatingEnable		
	McuERMClockGatingEnable		
	McuDMAClockGatingEnable		
	McuMPUClockGatingEnable		
	McuMSCMClockGatingEnable		
McuModuleConfiguration/ McuClockSettingConfig/ McuPeripheralClockConfig	McuPerName		
	McuPeripheralClockEnable		
	McuPeripheralClockSelect		
	McuPeripheralClockDivider		
	McuPeripheralFractionalDivider		
	McuPeripheralClockFrequency		
McuModuleConfiguration/ McuClockSettingConfig/ McuClockReferencePoint	McuClockReferencePointFrequency		
	McuClockFrequencySelect		

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
McuModuleConfiguration/ McuDemEventParameterRefs	MCU_E_TIMEOUT_FAILURE		
	MCU_E_CLOCK_FAILURE		
McuModuleConfiguration/ McuModeSettingConf	McuMode		
	McuPowerMode		
McuModuleConfiguration/ McuRamSectorSettingConf	McuRamSectorId		
	McuRamDefaultValue		
	McuRamSectionBaseAddress		
	McuRamSectionSize		
	McuRamSectionBaseAddrLinkerSym		
	McuRamSectionSizeLinkerSym		
	McuRamSectionWriteSize		
McuModuleConfiguration/ McuInterruptEvents	McuVoltageErrorEvent		
	McuAlternateResetEvent		
McuModuleConfiguration/ McuResetConfig	McuResetPinFilterBusClockSelect		
	McuResetPinFilterInStopMode		
	McuResetPinFilterInRunAndWait		
McuModuleConfiguration/ McuResetConfig/ McuSystemInterruptEnable	McuResetDelayTime		
	McuStopAcknowledgeErrorInterrupt		
	McuMDMAPSystemResetInterrupt		
	McuSoftwareInterrupt		
	McuCoreLockupInterrupt		
	McuJTAGResetInterrupt		
	McuGlobalInterrupt		
	McuExternalResetPinInterrupt		
	McuWatchdogInterrupt		
	McuLossOfLockInterrupt		
	McuLossOfClockInterrupt		
McuModuleConfiguration/ McuPowerControl	McuLowVoltageDetectInterruptEnable		
	McuLowVoltageDetectResetEnable		
	McuLowVoltageWarningInterruptEnable		

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
	McuLPODisable		
	McuClockBiasDisable		
	McuLowPowerBiasEnable		
	McuLpoTrimming		

Chapter 9
Integration Steps

This section gives a brief overview of the steps needed for integrating Micro Control Unit :

- Generate the required Mcu configurations. For more details refer to section Files required for Compilation
- Allocate proper memory sections in Mcu_MemMap.h and linker command file. For more details refer to section Sections to be defined in Mcu_MemMap.h
- Compile & build the Mcu with all the dependent modules. For more details refer to section Building the Driver

Chapter 10
ISR Reference

ISR functions exported by the Mcu driver.

Chapter 11
External Assumptions for MCU driver

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

[SMCAL_CPR_EXT163]

<< If interrupts are locked a centralized function pair to lock and unlock interrupts shall be used. >>

[SMCAL_CPR_EXT175]

<< The integrator shall assure the execution of code from system RAM when flash memory configurations need to be change (i.e. PFCR control fields of PFLASH memory need to be change) >>

[SMCAL_CPR_EXT182]

<< The integrator shall assure that the following Mcu functions (Mcu_InitClock, Mcu_DistributePllClock, Mcu_InitRamSection) are not interrupted during their execution. >>

[SWS_Mcu_00244]

<< If the register can affect several hardware modules and if it is an I/O register, it shall be initialised by the PORT driver. >>

NOTE

These registers are not unde MCU's coverage

[SWS_Mcu_00246]

<< One-time writable registers that require initialisation directly after reset shall be initialised by the startup code.(BSW12125, BSW12461) >>

NOTE

This requierement refers to the start-up code

[SWS_Mcu_00247]

<< All other registers not mentioned before shall be initialised by the start-up code. (BSW12125, BSW12461) >>

NOTE

This requierement refers to the start-up code

[SWS_Mcu_00136]

<< The MCU module's environment shall call the function Mcu_InitRamSection only after the MCU module has been initialized using the function Mcu_Init. >>

[SWS_Mcu_00139]

<< The MCU module's environment shall only call the function Mcu_InitClock after the MCU module has been initialized using the function Mcu_Init. >>

[SWS_Mcu_00141]

<< The function Mcu_DistributePllClock shall remove the current clock source (for example internal oscillator clock) from MCU clock distribution. (BSW12336) >>

[SWS_Mcu_00142]

<< If the function Mcu_DistributePllClock is called before PLL has locked, this function shall return E_NOT_OK immediately, without any further action. >>

[SWS_Mcu_00145]

<< The MCU module's environment shall only call the function Mcu_PerformReset after the MCU module has been initialized by the function Mcu_Init. >>

[SWS_Mcu_00148]

<< The MCU module's environment shall only call the function Mcu_SetMode after the MCU module has been initialized by the function Mcu_Init. >>

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.

While NXP has implemented advanced security features, all products may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on customer's applications and products, and NXP accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.

Document Number IM2MCUASR4.3 Rev0001R1.0.1
Revision 1.0