Integration Manual

for MPC574XG MCU Driver

Document Number: IM35MCUASR4.2 Rev0002R1.0.0
Rev. 1.0

Contents

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

Chapter 1
Revision History

Table 1-1. Revision History

Revision	Date	Author	Description
1.0	10/02/2017	Dang Cong	Update for MPC574XG/C 4.2 RTM 1.0.0 Release

Chapter 2
Introduction

This integration manual describes the integration requirements for MCU 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
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
PCS	Progressive Clock Switching
CMU	Clock Monitoring Unit
N/A	Not Applicable
MCU	Micro Controller Unit

2.5 Reference List

Table 2-3. Reference List

#	Title	Version
1	AUTOSAR 4.2 Rev0002MCU Driver Software Specification Document.	R4.2.2
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 MCU driver for NXP SemiconductorMPC574XG . It also explains the EB Tresos Studio plugin setup procedure.

3.1 Build Options

The MCU 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 MCU 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.

MCU Files

- MCU_TS_T2D35M10I0R0\src\MCU.c
- MCU_TS_T2D35M10I0R0\src\MCU_CMU.c
- MCU_TS_T2D35M10I0R0\src\MCU_CMU_Irq.c
- MCU_TS_T2D35M10I0R0\src\MCU_eMios.c
- MCU_TS_T2D35M10I0R0\src\MCU_FIRC.c

- MCU_TS_T2D35M10I0R0\src\MCU_FLASH.c
- MCU_TS_T2D35M10I0R0\src\MCU_FXOSC.c
- MCU_TS_T2D35M10I0R0\src\MCU_FXOSC_Irq.c
- MCU_TS_T2D35M10I0R0\src\MCU_IPW.c
- MCU_TS_T2D35M10I0R0\src\MCU_LPU.c
- MCU_TS_T2D35M10I0R0\src\MCU_MC_CGM.c
- MCU_TS_T2D35M10I0R0\src\MCU_MC_Irq.c
- MCU_TS_T2D35M10I0R0\src\MCU_MC_ME.c
- MCU_TS_T2D35M10I0R0\src\MCU_MC_PCU.c
- MCU_TS_T2D35M10I0R0\src\MCU_MC_RGM.c
- MCU_TS_T2D35M10I0R0\src\MCU_PLLDIG.c
- MCU_TS_T2D35M10I0R0\src\MCU_PLLDIG_Irq.c
- MCU_TS_T2D35M10I0R0\src\MCU_PMCDIG.c
- MCU_TS_T2D35M10I0R0\src\MCU_PRAM.c
- MCU_TS_T2D35M10I0R0\src\MCU_SIRC.c
- MCU_TS_T2D35M10I0R0\src\MCU_SIUL2.c
- MCU_TS_T2D35M10I0R0\src\MCU_SSCM.c
- MCU_TS_T2D35M10I0R0\src\MCU_STCU.c
- MCU_TS_T2D35M10I0R0\src\MCU_SXOSC.c
- MCU_TS_T2D35M10I0R0\src\MCU_SXOSC_Irq.c
- MCU_TS_T2D35M10I0R0\include\MCU.h
- MCU_TS_T2D35M10I0R0\include\MCU_EnvCfg.h
- MCU_TS_T2D35M10I0R0\include\MCU_CMU.h
- MCU_TS_T2D35M10I0R0\include\MCU_CMU_Types.h
- MCU_TS_T2D35M10I0R0\include\MCU_eMios.h
- MCU_TS_T2D35M10I0R0\include\MCU_eMios_Types.h
- MCU_TS_T2D35M10I0R0\include\MCU_FIRC.h
- MCU_TS_T2D35M10I0R0\include\MCU_FIRC_Types.h
- MCU_TS_T2D35M10I0R0\include\MCU_FLASH.h
- MCU_TS_T2D35M10I0R0\include\MCU_FLASH_IPVersion.h
- MCU_TS_T2D35M10I0R0\include\MCU_FLASH_Types.h
- MCU_TS_T2D35M10I0R0\include\MCU_FXOSC.h
- MCU_TS_T2D35M10I0R0\include\MCU_FXOSC_Types.h
- MCU_TS_T2D35M10I0R0\include\MCU_IPW.h
- MCU_TS_T2D35M10I0R0\include\MCU_IPW_Types.h
- MCU_TS_T2D35M10I0R0\include\MCU_LPU.h
- MCU_TS_T2D35M10I0R0\include\MCU_LPU_Types.h
- MCU_TS_T2D35M10I0R0\include\MCU_MC_CGM.h
- MCU_TS_T2D35M10I0R0\include\MCU_MC_CGM_Types.h
- MCU_TS_T2D35M10I0R0\include\MCU_MC_MC_IPVersion.h
- MCU_TS_T2D35M10I0R0\include\MCU_MC_ME.h

- MCU_TS_T2D35M10I0R0\include\MCU_MC_ME_Types.h
- MCU_TS_T2D35M10I0R0\include\MCU_MC_PCU.h
- MCU_TS_T2D35M10I0R0\include\MCU_MC_PCU_Types.h
- MCU_TS_T2D35M10I0R0\include\MCU_MC_RGM.h
- MCU_TS_T2D35M10I0R0\include\MCU_MC_RGM_Types.h
- MCU_TS_T2D35M10I0R0\include\MCU_PLLDIG.h
- MCU_TS_T2D35M10I0R0\include\MCU_PLLDIG_IPVersion.h
- MCU_TS_T2D35M10I0R0\include\MCU_PLLDIG_Types.h
- MCU_TS_T2D35M10I0R0\include\MCU_PMCDIG.h
- MCU_TS_T2D35M10I0R0\include\MCU_PMCDIG_IPVersion.h
- MCU_TS_T2D35M10I0R0\include\MCU_PMCDIG_Types.h
- MCU_TS_T2D35M10I0R0\include\MCU_PRAM.h
- MCU_TS_T2D35M10I0R0\include\MCU_PRAM_IPVersion.h
- MCU_TS_T2D35M10I0R0\include\MCU_PRAM_Types.h
- MCU_TS_T2D35M10I0R0\include\MCU_Reg_eSys_eMios.h
- MCU_TS_T2D35M10I0R0\include\MCU_SIRC.h
- MCU_TS_T2D35M10I0R0\include\MCU_SIRC_Types.h
- MCU_TS_T2D35M10I0R0\include\MCU_SIUL2.h
- MCU_TS_T2D35M10I0R0\include\MCU_SIUL2_Types.h
- MCU_TS_T2D35M10I0R0\include\MCU_SSCM.h
- MCU_TS_T2D35M10I0R0\include\MCU_SSCM_IPVersion.h
- MCU_TS_T2D35M10I0R0\include\MCU_SSCM_Types.h
- MCU_TS_T2D35M10I0R0\include\MCU_STCU.h
- MCU_TS_T2D35M10I0R0\include\MCU_STCU_IPVersion.h
- MCU_TS_T2D35M10I0R0\include\MCU_STCU_Types.h
- MCU_TS_T2D35M10I0R0\include\MCU_SXOSC.h
- MCU_TS_T2D35M10I0R0\include\MCU_SXOSC_Types.h
- MCU_TS_T2D35M10I0R0\include\Mcu_Reg_eSys_FLASHC.h
- MCU_TS_T2D35M10I0R0\include\Mcu_Reg_eSys_SIUL2.h
- MCU_TS_T2D35M10I0R0\include\Reg_eSys_CMU.h
- MCU_TS_T2D35M10I0R0\include\Reg_eSys_FIRC.h
- MCU_TS_T2D35M10I0R0\include\Reg_eSys_FXOSC.h
- MCU_TS_T2D35M10I0R0\include\Reg_eSys_LPU.h
- MCU_TS_T2D35M10I0R0\include\Reg_eSys_MC_CGM.h
- MCU_TS_T2D35M10I0R0\include\Reg_eSys_MC_ME.h
- MCU_TS_T2D35M10I0R0\include\Reg_eSys_MC_PCU.h
- MCU_TS_T2D35M10I0R0\include\Reg_eSys_MC_RGM.h
- MCU_TS_T2D35M10I0R0\include\Reg_eSys_PLLDIG.h
- MCU_TS_T2D35M10I0R0\include\Reg_eSys_PMCDIG.h
- MCU_TS_T2D35M10I0R0\include\Reg_eSys_PRAM.h
- MCU_TS_T2D35M10I0R0\include\Reg_eSys_SIRC.h

- MCU_TS_T2D35M10I0R0\include\Reg_eSys_SSCM.h
- MCU_TS_T2D35M10I0R0\include\Reg_eSys_STCU.h
- MCU_TS_T2D35M10I0R0\include\Reg_eSys_SXOSC.h

MCU Generated Files

- MCU_TS_T2D35M10I0R0\generate_PC\src\MCU_Cfg.c (For PC Variant) - This file should be generated by the user using a configuration tool for compilation
- MCU_TS_T2D35M10I0R0\generate_PC\include\MCU_Cfg.h - This file should be generated by the user using a configuration tool for compilation
- MCU_TS_T2D35M10I0R0\generate_PB\src\MCU_PBcfg_[variant].c (For PB Variant) - This file should be generated by the user using a configuration tool for compilation

Files from Base common folder

- ..\Base_ TS_T2D35M10I0R0 \include\Compiler.h
- ..\Base_ TS_T2D35M10I0R0 \include\Compiler_Cfg.h
- ..\Base_ TS_T2D35M10I0R0 \include\ComStack_Types.h
- ..\Base_ TS_T2D35M10I0R0 \include\Mcu_MemMap.h
- ..\Base_ TS_T2D35M10I0R0 \include\Mcal.h
- ..\Base_ TS_T2D35M10I0R0 \include\Platform_Types.h
- ..\Base_ TS_T2D35M10I0R0 \include\Std_Types.h
- ..\Base_ TS_T2D35M10I0R0 \include\Reg_eSys.h
- ..\Base_ TS_T2D35M10I0R0 \include\Soc_Ips.h
- ..\Base_ TS_T2D35M10I0R0 \include\Reg_Macros.h

Files from Dem folder:

- ..\Dem_ TS_T2D35M10I0R0 \include\Dem.h
- ..\Dem_ TS_T2D35M10I0R0 \include\Dem_IntErrId.h
- ..\Dem_ TS_T2D35M10I0R0 \include\Dem_Types.h

Files from Det folder:

- ..\Det_ TS_T2D35M10I0R0 \include\Det.h

Files from Rte folder:

- Rte_TS_T2D35M10I0R0\include\SchM_Mcu.h

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 21.0.0 b160607-0933 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_T2D35M10I0R0 \config\Mcu.xdm
- Pre-configuration file in EB tresos Studio XDM form containing values for reset configuration parameters:
  - ..\MCU_TS_T2D35M10I0R0 \config_ext\McuPreConfiguration.xdm
- VSMD (Vendor Specific Module Definition) file(s) in AUTOSAR compliant EPD format:
  - ..\ MCU_TS_T2D35M10I0R0 \autosar\Mcu.epd
- Code Generation Templates for parameters without variation points:
  - ..\MCU_TS_T2D35M10I0R0\generate_PC\src\MCU_Cfg.c
  - ..\MCU_TS_T2D35M10I0R0\generate_PC\include\MCU_Cfg.h
  - ..\MCU_TS_T2D35M10I0R0\generate_PC\MCU_checkvalues.m
  - ..\MCU_TS_T2D35M10I0R0\generate_PC\MCU_RegOperations.m
- Code Generation Templates for variant aware parameters:
  - ..\MCU_TS_T2D35M10I0R0\generate_PB\src\MCU_PBcfg_[variant].c
  - ..\MCU_TS_T2D35M10I0R0\generate_PB\MCU_checkvalues.m
  - ..\MCU_TS_T2D35M10I0R0\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_T2D35M10I0R0 , Resource_ TS_T2D35M10I0R0, Rte_ TS_T2D35M10I0R0 , Mcu_ TS_T2D35M10I0R0, Dem_TS_T2D35M10I0R0, EcuC_TS_T2D35M10I0R0, Det_TS_T2D35M10I0R0 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

MCU_EXCLUSIVE_AREA_00 is used in Mcu_eMios_Init function.

MCU_EXCLUSIVE_AREA_01 is used in Mcu_eMios_ConfigureGpren function.

Critical Region Exclusive Matrix

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

The critical regions from interrupts are grouped in “Interrupt Service Routines Critical Regions (composed diagram)”. If an exclusive area is “exclusive” with the composed “Interrupt Service Routines Critical Regions (composed diagram)” group, it means that it is exclusive with each one of the ISR critical regions.

Table 5-1. Exclusive Areas

	MCU_EXCLUSIVE_AREA_00	MCU_EXCLUSIVE_AREA_01	Interrupt Service Routines Critical Regions(composed diagram)
MCU_EXCLUSIVE_AREA_00		X	
MCU_EXCLUSIVE_AREA_01	X	X	

5.2 Peripheral Hardware Requirements

None

5.3 ISR to configure within OS – dependencies

The following ISR's are used by the MCU driver:

Table 5-2. MCU ISRs

ISR Name	Hardware interrupt vector
Mcu_LPU_InvalidMode_ISR	250
Mcu_Mcv4_ModeEnterSafe_ISR	251
Mcu_Mcv4_ModeTransitionComplete_ISR	252
Mcu_Mcv4_InvalidMode_ISR	253
Mcu_Mcv4_InvalidConfiguration_ISR	254
Mcu_Mcv4_AlternateReset_ISR	255
Mcu_Fxosc_ISR	257
Mcu_Sxosc_ISR	258
Mcu_PllDig_Pll0_LossOfLock_ISR	480

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** This module is necessary for enabling Development error detection. The API function used is Det_ReportError(). The activation/deactivation of Development error detection is configurable using 'CanDevErrorDetect' configuration parameter.
- **Dem:** This module is necessary for enabling reporting of production relevant error status. The API function used is Dem_ReportErrorStatus().
- **EcuC:** This module is required for configuring the variant handling in Tresos.
- **Rte:** This 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:** Sub-Derivative model is selected from Resource configuration.

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.

Mcu module will set the UAA bit in REG_PROT_GCR register for MC_ME, MC_RGM, LPU and only for MC_CGM (because FXOSC, SXOSC, PLLDIG and CMU are in the same address space as MC_CGM).

Note: Writing the UAA bit in REG_PROT_GCR for MC_CGM will allow user mode access to MC_CGM registers, all PLLDIG modules, FIRC, SXOSC and FXOSC.

The following functions of Mcu must be called as trusted functions:

- **Mcu_FXOSC_Config** to write FXOSC_CTL register which can only be written in supervisor mode.
- **Mcu_PMCDIG_Config** to write PMCDIG registers which can only be written in supervisor mode.
- **Mcu_FLASH_Init** to write PFLASH registers which can only be written in supervisor mode.
- **Mcu_FLASH_SetWS** to write PFLASH registers which can only be written in supervisor mode.
- **Mcu_MC_RGM_ResetInit** to write MC_RGM registers which can only be written in supervisor mode.
- **Mcu_MC_RGM_GetResetReason** to write MC_RGM registers which can only be written in supervisor mode.
- **Mcu_MC_RGM_GetResetRawValue** to write MC_RGM registers which can only be written in supervisor mode.
- **Mcu_PRAM_SetRamWS** to write PFLASH registers which can only be written in supervisor mode.
- **Mcu_MC_CGM_AuxClockConfig** to write MC_CGM_AC6_DC0 register which can only be written in supervisor mode

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.

McuCmuNotification The callout configured by the user for CMU notifications.

McuPrepareMemoryConfig The callout configured by the user for preparing the RAM/ FLASH configuration.

Chapter 7
Memory Allocation

7.1 Sections to be defined in 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		
	McuCmuNotification		
	McuErrorIsrNotification		
	McuDisableFlashConfigFromRam		
	McuUseLpuModes		
McuPrepareMemoryConfig			

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
McuDebugConfiguration	McuDisableDemReportErrorStatus		
	McuGetPeriphStateApi		
	McuGetSystemStatetApi		
	McuGetPowerModeStatetApi		
	McuSscmGetMemConfigApi		
	McuSscmGetStatusApi		
	McuSscmGetUopsApi		
	McuGetMidrStructureApi		
	McuGetPowerDomainApi		
	McuEmiosConfigureGprenApi		
	McuDisableCmuApi		
McuCoreControlConfiguration	McuCADDRnControl		
McuEMIOSConfiguration	McuEMIOSControl		
McuPublishedInformation/ McuResetReasonConf	McuResetReason		
CommonPublishedInformation	ArReleaseMajorVersion		
	ArReleaseMinorVersion		
	ArReleaseRevisionVersion		
	ModuleId		
	SwMajorVersion		
	SwMinorVersion		
	SwPatchVersion		
	VendorApiInfix		
	VendorId		
McuModuleConfiguration	McuNumberOfMcuModes		
	McuRamSectors		
	McuResetSetting		
	McuFastCrystalFrequencyHz		
	McuSlowCrystalFrequencyHz		
	McuFastIrcFrequencyHz		
	McuSlowIrcFrequencyHz		
	McuExternalClockFrequencyHz		
	McuClockSrcFailureNotification		
McuModuleConfiguration/ McuClockSettingConfig	McuClockSettingId		
	McuClksysClockMcuControl		

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
	McuSystemClockSwitch		
	McuSystemClockF160Freq		
	McuSystemClkDiv0_En		
	McuSystemClkDiv_Divider0		
	McuSystemClockS160Freq		
	McuSystemClkDiv1_En		
	McuSystemClkDiv_Divider1		
	McuSystemClockS80Freq		
	McuSystemClkDiv2_En		
	McuSystemClkDiv_Divider2		
	McuSystemClockS40Freq		
	McuSystemClkDiv3_En		
	McuSystemClkDiv_Divider3		
	McuSystemClockF40Freq		
	McuSystemClkDiv4_En		
	McuSystemClkDiv_Divider4		
	McuSystemClockF80Freq		
	McuSystemClkDiv5_En		
	McuSystemClkDiv_Divider5		
	McuSystemClockFS80Freq		
	McuSystemClkDiv6_En		
	McuSystemClkDiv_Divider6		
	McuSystemClockF20Freq		
	McuClkSetFXOSC_En		
	McuClkSetSXOSC_En		
	McuClkSetFIRCOSC_En		
	McuClkSetSIRCOSC_En		
	McuClkSetPLL0_En		
McuModuleConfiguration/ McuClockSettingConfig/ McuFIRC	McuFIRCUnderMcuControl		
	McuFIRC_Div		
	McuFIRC_DivOutputValue		
McuModuleConfiguration/ McuClockSettingConfig/ McuSIRC	McuSIRCUnderMcuControl		
	McuSIRC_Div		
	McuSIRC_DivOutputValue		
McuModuleConfiguration/ McuClockSettingConfig/ McuSXOSC	McuSXOSCUnderMcuControl		
	McuSXOSC_OSCBYP		
	McuSXOSC_ALC		
	McuSXOSC_EOCV		

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
	McuSXOSC_M_OSC		
	McuSXOSC_Div		
	McuSXOSC_DivOutputValue		
McuModuleConfiguration/ McuClockSettingConfig/ McuFXOSC	McuFXOSCUnderMcuControl		
	McuFXOSC_OSCBYP		
	McuFXOSC_OSCM		
	McuFXOSC_EOCV		
	McuFXOSC_M_OSC		
	McuFXOSC_Div		
	McuFXOSC_DivOutputValue		
McuModuleConfiguration/ McuClockSettingConfig/ McuProgClkSwitch	McuProgresSwitchDuration		
McuModuleConfiguration/ McuClockSettingConfig/ McuProgClkSwitch/ McuPcsConfig	McuPCS_Name		
	McuPCS_DivInit		
	McuPCS_DivRate		
	McuPCS_DivStart		
	McuPCS_DivEnd		
McuModuleConfiguration/ McuClockSettingConfig/ McuCLKOUT1	McuCLKOUT1UnderMcuControl		
	McuCLKOUT1_Source		
	McuCLKOUT1Div_En		
	McuCLKOUT1_Divisor		
	McuCLKOUT1Freq		
McuModuleConfiguration/ McuClockSettingConfig/ McuAuxClock2	McuAuxClockUnderMcuControl		
	McuAuxClk_Source		
	McuAuxClkFreq		
McuModuleConfiguration/ McuClockSettingConfig/ McuAuxClock3	McuAuxClockUnderMcuControl		
	McuAuxClk_Source		
	McuAuxClkFreq		
McuModuleConfiguration/ McuClockSettingConfig/ McuAuxClock4	McuAuxClockUnderMcuControl		
	McuAuxClk_Source		
	McuAuxClkFreq		
McuModuleConfiguration/ McuClockSettingConfig/ McuAuxClock5	McuAuxClockUnderMcuControl		
	McuAuxClk_Source		

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
	McuAuxClkFreq		
McuModuleConfiguration/ McuClockSettingConfig/ McuAuxClock6	McuAuxClockUnderMcuControl		
	McuAuxClk_Source		
	McuAuxClkDiv0_En		
	McuAuxClkDiv0_Divisor		
	McuAuxClkDiv0Freq		
McuModuleConfiguration/ McuClockSettingConfig/ McuAuxClock8	McuAuxClockUnderMcuControl		
	McuAuxClk_Source		
	McuAuxClkFreq		
McuModuleConfiguration/ McuClockSettingConfig/ McuAuxClock9	McuAuxClockUnderMcuControl		
	McuAuxClk_Source		
	McuAuxClkFreq		
McuModuleConfiguration/ McuClockSettingConfig/ McuPll_0	McuPLLUnderMcuControl		
McuModuleConfiguration/ McuClockSettingConfig/ McuPll_0/ McuPll_Configuration	McuPllCal3Mfden		
	McuPllDvRfdphi		
	McuPllDvRfdphi1		
	McuPllDvPrediv		
	McuPllDvMfd		
	McuPllFmStepsize		
	McuPllFmStepno		
	McuPllFdMfn		
	McuPllCalBypcal		
	McuPllFmSscgbyp		
	McuPllFdSmden		
	McuPllFdSdm2		
	McuPllFdSdm3		
	McuPllFdDthDis		
	McuPllCrLolie		
	McuPllCrLolre		
	McuExternalPowerDownCycleInt		
McuModuleConfiguration/ McuClockSettingConfig/ McuPll_0/McuPll_Parameter	PLL_PHI_Frequency		
	PLL_PHI1_Frequency		
	PLL_VCO_Frequency		

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
McuModuleConfiguration/ McuClockSettingConfig/ McuEMIOS/McuEMIOS_0	McuEMIOS_Clk		
	McuMdisBit		
	McuFreezeBit		
	McuGlobalTimeBaseEnable		
	McuExternalTimeBase		
	McuGlobalPrescalerEnable		
	McuGlobalPrescaler		
McuModuleConfiguration/ McuClockSettingConfig/ McuEMIOS/McuEMIOS_1	McuEMIOS_Clk		
	McuMdisBit		
	McuFreezeBit		
	McuGlobalTimeBaseEnable		
	McuExternalTimeBase		
	McuGlobalPrescalerEnable		
	McuGlobalPrescaler		
McuModuleConfiguration/ McuClockSettingConfig/ McuEMIOS/McuEMIOS_2	McuEMIOS_Clk		
	McuMdisBit		
	McuFreezeBit		
	McuGlobalTimeBaseEnable		
	McuExternalTimeBase		
	McuGlobalPrescalerEnable		
	McuGlobalPrescaler		
McuModuleConfiguration/ McuClockSettingConfig/ McuClkMonitor_0	McuClkMonitorEn		
	McuRCDivisorFactor		
	McuHighFrequencyRef		
	McuLowFrequencyRef		
McuModuleConfiguration/ McuClockSettingConfig/ McuFlash	McuFlashAddrPipelineControl		
	McuFlashReadWaitStateControl		
McuModuleConfiguration/ McuClockSettingConfig/ McuRam/ McuRamPRAMC_0_PRCR1	McuFT_DIS		
	McuPRI		
	McuP1BODIS		
	McuP0BODIS		
McuModuleConfiguration/ McuClockSettingConfig/ McuRam/ McuRamPRAMC_1_PRCR1	McuFT_DIS		
	McuP0BODIS		
McuModuleConfiguration/ McuClockSettingConfig/	McuFT_DIS		
	McuP0BODIS		

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
McuRam/ McuRamPRAMC_2_PRCR1			
McuModuleConfiguration/ McuClockSettingConfig/ McuClockReferencePoint	McuClockFrequencySelect		
	McuClockReferencePointFrequency		
McuModuleConfiguration/ McuDemEventParameterRefs	MCU_E_TIMEOUT_FAILURE		
	MCU_E_INVALIDMODE_CONFIG		
	MCU_E_SSCM_CER_FAILURE		
	MCU_E_CLOCK_FAILURE		
McuModuleConfiguration/ McuModeSettingConf	McuMode		
	McuPowerMode		
	McuSystemClockSwitch		
	McuFircControl		
	McuFXoscControl		
	McuSIrcOscControl		
	McuSXOscControl		
	McuPll0Control		
	McuFlashPowerControl		
	McuOutputPowerDownControl		
	McuMainVoltageRegulatorControl		
	McuPowerLevelControl		
	McuFastTransition		
McuModuleConfiguration/ McuModeSettingConf/ McuCoreConfiguration	McuCoreCaddr1ResetEnable		
	McuCoreZ4ADefaultBootAddrEnable		
	McuCoreCaddr1BootAddr		
	McuCoreCaddr2ResetEnable		
	McuCoreZ4BDefaultBootAddrEnable		
	McuCoreCaddr2BootAddr		
	McuCoreCaddr3ResetEnable		
	McuCoreZ2DefaultBootAddrEnable		
	McuCoreCaddr3BootAddr		
McuModuleConfiguration/ McuModeSettingConf/ McuModeSettingConfAfterWkp	McuSystemClockSwitch		
	McuFircControl		
	McuFXoscControl		

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
	McuSIrcOscControl		
	McuSXOscControl		
	McuPll0Control		
	McuFlashPowerControl		
	McuOutputPowerDownControl		
	McuPowerLevelControl		
McuModuleConfiguration/ McuModeSettingConf/ McuLpuConfiguration	McuCAN0ContinuousOperation		
	McuDirectDrun		
	McuSystemClockSelect		
	McuLpuSleep		
	McuFircControl		
	McuFXoscControl		
	McuSIrcOscControl		
	McuSXOscControl		
	McuLpuMRIGInterruptEnable		
	McuLpuNEMInterruptEnable		
	McuDisableEmios2		
	McuDisableEmios0		
	McuDisableCmp2		
	McuDisableCmp1		
	McuDisableCmp0		
	McuDisableMemu1		
	McuDisableBctu		
	McuDisableCan0		
	McuDisableLin0		
	McuDisableSpi0		
	McuDisableAdc0		
McuModuleConfiguration/ McuRamSectorSettingConf	McuRamSectorId		
	McuRamDefaultValue		
	McuRamSectionBaseAddress		
	McuRamSectionSize		
	McuRamSectionBaseAddrLinkerSym		
	McuRamSectionSizeLinkerSym		
McuModuleConfiguration/ McuEnableMode	McuModeStop0		
	McuModeStandBy0		
	McuModeRun3		

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
	McuModeRun2		
	McuModeRun1		
McuModuleConfiguration/ McuCoreConfiguration/ McuCoreZ4AConfiguration	McuCoreReset		
	McuUseDefaultBootAddrEnable		
	McuBootAddress		
	McuCCTL1Run3		
	McuCCTL1Run2		
	McuCCTL1Run1		
	McuCCTL1Run0		
	McuCCTL1Drun		
	McuCCTL1Safe		
McuModuleConfiguration/ McuCoreConfiguration/ McuCoreZ4BConfiguration	McuCoreReset		
	McuUseDefaultBootAddrEnable		
	McuBootAddress		
	McuCCTL2Run3		
	McuCCTL2Run2		
	McuCCTL2Run1		
	McuCCTL2Run0		
	McuCCTL2Drun		
	McuCCTL2Safe		
McuModuleConfiguration/ McuCoreConfiguration/ McuCoreZ2Configuration	McuCoreReset		
	McuUseDefaultBootAddrEnable		
	McuBootAddress		
	McuCCTL3Run3		
	McuCCTL3Run2		
	McuCCTL3Run1		
	McuCCTL3Run0		
	McuCCTL3Drun		
	McuCCTL3Safe		
McuModuleConfiguration/ McuInterruptTransition	McuTransitionComplete		
	McuSafeMode		
	McuInvalidMode		
	McuInvalidConfiguration		
	McuInvalidCoreConfiguration		
McuModuleConfiguration/ McuInterruptEvents	McuAlternateResetEvent		

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
McuModuleConfiguration/ McuPlatformFlashConfig	McuPort0Master15Prefetch		
	McuPort0Master14Prefetch		
	McuPort0Master13Prefetch		
	McuPort0Master12Prefetch		
	McuPort0Master11Prefetch		
	McuPort0Master10Prefetch		
	McuPort0Master9Prefetch		
	McuPort0Master8Prefetch		
	McuPort0Master7Prefetch		
	McuPort0Master6Prefetch		
	McuPort0Master5Prefetch		
	McuPort0Master4Prefetch		
	McuPort0Master3Prefetch		
	McuPort0Master2Prefetch		
	McuPort0Master1Prefetch		
	McuPort0Master0Prefetch		
	McuPort0DataPrefetch		
	McuPort0InstructionPrefetch		
	McuPort0PrefetchLimit		
	McuPort0ReadBufferEnable		
	McuPort1Master15Prefetch		
	McuPort1Master14Prefetch		
	McuPort1Master13Prefetch		
	McuPort1Master12Prefetch		
	McuPort1Master11Prefetch		
	McuPort1Master10Prefetch		
	McuPort1Master9Prefetch		
	McuPort1Master8Prefetch		
	McuPort1Master7Prefetch		
	McuPort1Master6Prefetch		
	McuPort1Master5Prefetch		
	McuPort1Master4Prefetch		
	McuPort1Master3Prefetch		
	McuPort1Master2Prefetch		
	McuPort1Master1Prefetch		
	McuPort1Master0Prefetch		
	McuPort1DataPrefetch		
	McuPort1InstructionPrefetch		
	McuPort1PrefetchLimit		

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
	McuPort1ReadBufferEnable		
	McuPort2Master15Prefetch		
	McuPort2Master14Prefetch		
	McuPort2Master13Prefetch		
	McuPort2Master12Prefetch		
	McuPort2Master11Prefetch		
	McuPort2Master10Prefetch		
	McuPort2Master9Prefetch		
	McuPort2Master8Prefetch		
	McuPort2Master7Prefetch		
	McuPort2Master6Prefetch		
	McuPort2Master5Prefetch		
	McuPort2Master4Prefetch		
	McuPort2Master3Prefetch		
	McuPort2Master2Prefetch		
	McuPort2Master1Prefetch		
	McuPort2Master0Prefetch		
	McuPort2DataPrefetch		
	McuPort2InstructionPrefetch		
	McuPort2PrefetchLimit		
	McuPort2ReadBufferEnable		
	McuPort0PageBufferConfiguration		
	McuPort1PageBufferConfiguration		
	McuPort2PageBufferConfiguration		
	McuBAFDisable		
	McuFlashArbitrationMode		
	McuMaster0AccessProt		
	McuMaster1AccessProt		
	McuMaster2AccessProt		
	McuMaster3AccessProt		
	McuMaster4AccessProt		
	McuMaster5AccessProt		
	McuMaster6AccessProt		
	McuMaster7AccessProt		
	McuMaster8AccessProt		
	McuMaster9AccessProt		
	McuMaster10AccessProt		

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
	McuMaster11AccessProt		
	McuMaster12AccessProt		
	McuMaster13AccessProt		
	McuMaster14AccessProt		
	McuMaster15AccessProt		
McuModuleConfiguration/ McuRunConfig	McuModeRun3		
	McuModeRun2		
	McuModeRun1		
	McuModeRun0		
	McuModeDRun		
	McuModeSafe		
McuModuleConfiguration/ McuLowPowerConfig	McuStop0		
	McuStandby0		
McuModuleConfiguration/ McuPeripheral	McuPerName		
	McuPerDBGConfig		
	McuPerRunConfig		
	McuPerLowPwrConfig		
McuModuleConfiguration/ McuResetConfig	McuResetType		
	McuFunctResetEscThreshold		
	McuDestResetEscThreshold		
McuModuleConfiguration/ McuResetConfig/ Mcu_D_LVD_LV_PD2_COLD_ResetSource	McuDisableReset		
	McuEnableInterruptSafe		
	McuResetPhase		
	McuResetPin		
McuModuleConfiguration/ McuResetConfig/ Mcu_D_HVD_LV_COLD_ResetSource	McuDisableReset		
	McuEnableInterruptSafe		
	McuResetPhase		
McuModuleConfiguration/ McuResetConfig/ Mcu_D_LVD_IO_A_HI_ResetSource	McuDisableReset		
	McuEnableInterruptSafe		
	McuResetPhase		
McuModuleConfiguration/ McuResetConfig/ Mcu_D_Z2_DBG_ResetSource	McuDisableReset		
	McuEnableInterruptSafe		
	McuResetPhase		
	McuResetPin		
McuModuleConfiguration/ McuResetConfig/	McuDisableReset		
	McuEnableInterruptSafe		

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
Mcu_D_Z4B_DBG_ResetSource	McuResetPhase		
	McuResetPin		
McuModuleConfiguration/ McuResetConfig/ Mcu_D_Z4A_DBG_ResetSource	McuDisableReset		
	McuEnableInterruptSafe		
	McuResetPhase		
	McuResetPin		
McuModuleConfiguration/ McuResetConfig/ Mcu_BE_FCCU_SHORT_ResetSource	McuResetPin		
McuModuleConfiguration/ McuResetConfig/ Mcu_BE_FCCU_LONG_ResetSource	McuResetPin		
McuModuleConfiguration/ McuResetConfig/ Mcu_D_CMU_OLR_ResetSource	McuDisableReset		
	McuEnableInterruptSafe		
	McuResetPhase		
	McuResetPin		
McuModuleConfiguration/ McuResetConfig/ Mcu_D_JTAG_FUNC_ResetSource	McuDisableReset		
	McuEnableInterruptSafe		
	McuResetPhase		
	McuResetPin		
McuModuleConfiguration/ McuResetConfig/ Mcu_D_NMI_WKPU_ResetSource	McuDisableReset		
	McuEnableInterruptSafe		
	McuResetPhase		
	McuResetPin		
McuModuleConfiguration/ McuResetConfig/ Mcu_SS_SOFT_FUNC_ResetSource	McuResetPhase		
	McuResetPin		
McuModuleConfiguration/ McuResetConfig/ Mcu_BE_HSM_FUNC_Reset Source	McuResetPin		
McuModuleConfiguration/ McuResetConfig/ Mcu_SS_EXR_ResetSource	McuResetPhase		
McuModuleConfiguration/ McuPowerControl/ McuPmcRamDomain_Config	McuPmcRdCr_MemSleep		
	McuPmcRdCr_PadKeep		
	McuPmcRdCr_Rd64Ret		

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
	McuPmcRdCr_Rd128Ret		
	McuPmcRdCr_Rd256Ret		
McuModuleConfiguration/ McuPowerControl/ McuPmcMisc_Config	McuPmcMcr_FlashLpFastExistDis		
	McuPmcMcr_LvdPd2ColdRee		
	McuPmcMcr_HvdRee		
	McuPmcMcr_LvdIoHiRee		

Chapter 9
Integration Steps

This section gives a brief overview of the steps needed for integrating Micro Controller 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 MemMap.h
- Compile & build the MCU with all the dependent modules. For more details refer to section Building the Driver

Chapter 10
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. >>

[SWS_Mcu_00208]

<< The MCU module's environment shall call this function only if the MCU module has been already initialized using the function MCU_Init.(BSW13701) >>

NOTE

This feature is not available on S32K14X hardware

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Document Number IM35MCUASR4.2 Rev0002R1.0.0
Revision 1.0

AUTOSAR