Integration Manual

for S32K14X MCL Driver

Document Number: IM2MCLASR4.2 Rev0002R1.0.1
Rev. 1.0

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

Section number Title Page

Chapter 6
Main API Requirements

Chapter 7
Memory Allocation

Chapter 8
Configuration parameters considerations

Chapter 9
Integration Steps

Chapter 10
External Assumptions for MCL driver

Chapter 1
Revision History

Table 1-1. Revision History

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

Chapter 2 Introduction

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

2.1 Supported Derivatives

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

Table 2-1. S32K14X Derivatives

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

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

2.2 Overview

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

AUTOSAR

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

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

2.3 About this Manual

This Technical Reference employs the following typographical conventions:

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

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

Notes and warnings are shown as below:

Note

This is a note.

2.4 Acronyms and Definitions

Table 2-2. Acronyms and Definitions

Term	Definition
API	Application Programming Interface
AUTOSAR	Automotive Open System Architecture
ASM	Assembler
BSMI	Basic Software Make file Interface
CAN	Controller Area Network
DEM	Diagnostic Event Manager
DET	Development Error Tracer
C/CPP	C and C++ Source Code
VLE	Variable Length Encoding
N/A	Not Applicable
MCL	Micro Controller Library
FTM	FlexTimer Module

2.5 Reference List

Table 2-3. Reference List

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

Chapter 3
Building the Driver

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

3.1 Build Options

The MCL driver files are compiled using

- Green Hills Multi 7.1.4 / Compiler 2017.1.4
- (Linaro GCC 6.3-2017.06~dev) 6.3.1 20170509 (Thu Dec 7 13:28:42 CST 2017 build.sh rev=g7fea41d s=L631 Earmv7 -V release_g7fea41d_build_Fed_Earmv7) (from S32-DS-ARM_v2018)
- IAR: V8.11.2

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

Note

The TS_T40D2M10I1R0 plugin name is composed as follow:

TS_T = Target_Id

D = Derivative_Id

M = SW_Version_Major

I = SW_Version_Minor

R = Revision

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

3.1.1 GHS Compiler/Linker/Assembler Options

Table 3-1. Compiler Options

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

Table 3-2. Assembler Options

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

Table 3-3. Linker Options

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

3.1.2 IAR Compiler/Linker/Assembler Options

Table 3-4. Compiler Options

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

Table continues on the next page...

Table 3-4. Compiler Options (continued)

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

Table 3-5. Assembler Options

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

Table 3-6. Linker Options

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

3.1.3 GCC Compiler/Linker/Assembler Options

Table 3-7. Compiler Options

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

Table continues on the next page...

Table 3-7. Compiler Options (continued)

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

Table 3-8. Assembler Options

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

Table 3-9. Linker Options

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

3.2 Files required for Compilation

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

MCL Files

- ..\ MCL_TS_T40D2M10I1R0 \include\CDD_Mcl.h
- ..\ MCL_TS_T40D2M10I1R0 \include\Ftm_Common_Types.h
- ..\ MCL_TS_T40D2M10I1R0 \include\Mcl.h
- ..\ MCL_TS_T40D2M10I1R0 \include\Mcl_Dma.h
- ..\ MCL_TS_T40D2M10I1R0 \include\Mcl_Dma_Types.h
- ..\ MCL_TS_T40D2M10I1R0 \include\Mcl_DmaMux.h
- ..\ MCL_TS_T40D2M10I1R0 \include\Mcl_DmaMux_Types.h
- ..\ MCL_TS_T40D2M10I1R0 \include\Mcl_EnvCfg.h
- ..\ MCL_TS_T40D2M10I1R0 \include\Mcl_IPW.h
- ..\ MCL_TS_T40D2M10I1R0 \include\Mcl_IPW_Notif.h
- ..\ MCL_TS_T40D2M10I1R0 \include\Mcl_IPW_Types.h
- ..\ MCL_TS_T40D2M10I1R0 \include\Mcl_Lmem.h
- ..\ MCL_TS_T40D2M10I1R0 \include\Mcl_Lmem_Types.h
- ..\ MCL_TS_T40D2M10I1R0 \include\Mcl_Notif.h
- ..\ MCL_TS_T40D2M10I1R0 \include\Mcl_TrgMux.h
- ..\ MCL_TS_T40D2M10I1R0 \include\Mcl_TrgMux_Types.h
- ..\ MCL_TS_T40D2M10I1R0 \include\Mcl_Types.h
- ..\ MCL_TS_T40D2M10I1R0 \include\Reg_eSys_Dma.h
- ..\ MCL_TS_T40D2M10I1R0 \include\Reg_eSys_DmaMux.h
- ..\ MCL_TS_T40D2M10I1R0 \include\Reg_eSys_Ftm.h
- ..\ MCL_TS_T40D2M10I1R0 \include\Reg_eSys_Lmem.h
- ..\ MCL_TS_T40D2M10I1R0 \include\Reg_eSys_Lpit.h
- ..\ MCL_TS_T40D2M10I1R0 \include\Reg_eSys_Lptmr.h
- ..\ MCL_TS_T40D2M10I1R0 \include\Reg_eSys_TrgMux.h
- ..\ MCL_TS_T40D2M10I1R0 \src\CDD_Mcl.c
- ..\ MCL_TS_T40D2M10I1R0 \src\Ftm_Common.c
- ..\ MCL_TS_T40D2M10I1R0 \src\LPit_Common.c
- ..\ MCL_TS_T40D2M10I1R0 \src\Lptmr_Common.c
- ..\ MCL_TS_T40D2M10I1R0 \src\Mcl_Dma.c
- ..\ MCL_TS_T40D2M10I1R0 \src\Mcl_Dma_Irq.c
- ..\ MCL_TS_T40D2M10I1R0 \src\Mcl_DmaMux.c
- ..\ MCL_TS_T40D2M10I1R0 \src\Mcl_IPW.c

- ..\ MCL_TS_T40D2M10I1R0 \src\Mcl_Lmem.c
- ..\ MCL_TS_T40D2M10I1R0 \src\Mcl_TrgMux.c

MCL Generated Files

- CDD_Mcl_Cfg.c (For PC Variant) - For driver compilation, this file should be generated by the user using a configuration tool
- CDD_Mcl_PBcfg_[VariantName].c (For PB Variant) - For driver compilation, this file should be generated by the user using a configuration tool.The file contains the definition of the init pointer for the respective variant.
- CDD_Mcl_PBcfg_[VariantName].h (For PB Variant) - For driver compilation, this file should be generated by the user using a configuration tool.The file contains the export of the init pointer for the respective variant.In case the user needs to call the Mcl_Init with the pointer for variant [VariantName], then he needs to include the file CDD_Mcl_PBcfg_[VariantName].h to have the export of the init pointer of Variant X.
- CDD_Mcl_Cfg.h - For driver compilation, this file should be generated by the user using a configuration tool

Files from Base common folder

- ..\Base_TS_T40D2M10I1R0 \include\Compiler.h
- ..\Base_TS_T40D2M10I1R0 \include\Compiler_Cfg.h
- ..\Base_TS_T40D2M10I1R0 \include\CompilerDefinition.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\MemMap.h
- ..\Base_TS_T40D2M10I1R0 \include\Platform_Types.h
- ..\Base_TS_T40D2M10I1R0 \include\Reg_eSys.h
- ..\Base_TS_T40D2M10I1R0 \include\RegLockMacros.h
- ..\Base_TS_T40D2M10I1R0 \include\SilRegMacros.h
- ..\Base_TS_T40D2M10I1R0 \include\Soc_Ips.h
- ..\Base_TS_T40D2M10I1R0 \include\Std_Types.h
- ..\Base_TS_T40D2M10I1R0 \include\StdRegMacros.h

Files from Dem folder:

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

3.3 Setting up the Plug-ins

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

Location of various files inside the MCL module folder:

- VSMD (Vendor Specific Module Definition) file in EB tresos Studio XDM format:
  - ..\ MCL _ TS_T40D2M10I1R0 \config\Mcl.xdm
  - ..\ Dem _ TS_T40D2M10I1R0 \config\Dem.xdm
  - ..\ Resource _ TS_T40D2M10I1R0 \config\Resource.xdm

- VSMD (Vendor Specific Module Definition) file(s) in AUTOSAR compliant EPD format:
  - ..\ MCL _ TS_T40D2M10I1R0 \autosar\Mcl_<subderivative_name>.epd
  - ..\ Dem _ TS_T40D2M10I1R0 \autosar\Dem.epd
  - ..\ Resource _ TS_T40D2M10I1R0 \autosar \Resource_<subderivative_name>.epd

- Code Generation Templates for Pre-Compile time configuration parameters:
  - ..\ MCL _ TS_T40D2M10I1R0 \output\src\CDD_Mcl_Cfg.c
  - ..\ MCL _ TS_T40D2M10I1R0 \output\include\CDD_Mcl_Cfg.h

- Code Generation Templates for Post-Build time configuration parameters:
  - ..\ MCL _ TS_T40D2M10I1R0 \output\src\CDD_Mcl_PBCfg.c
  - ..\ MCL _ TS_T40D2M10I1R0 \output\include\CDD_Mcl_Cfg.h

Steps to generate the configuration:

1. Copy the module folders Mcl _ TS_T40D2M10I1R0 , Dem_ TS_T40D2M10I1R0 , Base_ TS_T40D2M10I1R0 , Resource_ 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.

Dependencies

- **RESOURCE** is required to select processor derivative. Current Mcl driver has support for the following derivatives, each one having attached a Resource file: 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 .

- **DET** is required for signaling the development error detection (parameters out of range, null pointers, etc).
- **DEM** is required for signaling the production error detection (hardware failure, etc).

Chapter 4
Function calls to module

4.1 Function Calls during Start-up

The API to be called for this is Mcl _Init(). The MCU module should be initialized before MCL. All modules which use MCL features (example: DMA, Crossbar, Lmem, Ftm, Gtm, TrgMux...) should be initialized after MCL.

4.2 Function Calls during Shutdown

The API to be called for this is Mcl_DeInit(). The MCU module should be deinitialized after MCL. All modules which use MCL features (example: DMA, Crossbar, Gtm, TrgMux) should be deinitialized before MCL.

4.3 Function Calls during Wake-up

NA.

Chapter 5
Module requirements

5.1 Exclusive areas to be defined in BSW scheduler

In the current implementation, MCL 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 MCL, stubs are used for RTE. The following critical regions are used in the MCL driver:

5.1.1 MCL_EXCLUSIVE_AREA_12 With DMA, Used in function Dma_Init, protects the DMA_CR and DMA_CPR registers.

5.1.2 MCL_EXCLUSIVE_AREA_13 With DMA, used in function Dma_DeInit, protects the DMA_CR and DMA_CPR registers.

5.1.3 MCL_EXCLUSIVE_AREA_14 With DMA, Used in function Dma_SetChannelPriority, protects the DMA_CR and DMA_CPR registers.

5.1.4 MCL_EXCLUSIVE_AREA_15 With DMA, Used in function Dma_ConfigTcd, protects the TCDx.2ND_WORD, TCDx.6TH_WORD and TCDx.8TH_WORD registers.

5.1.5 MCL_EXCLUSIVE_AREA_16 With DMA, Used in function Mcl_IPW_DmaSetSModSize, protects the TCDx.2ND_WORD, TCDx.6TH_WORD and TCDx.8TH_WORD registers.

5.1.6 MCL_EXCLUSIVE_AREA_17 With DMA, Used in function Mcl_IPW_DmaSetDModSize, protects the TCDx.2ND_WORD, TCDx.6TH_WORD and TCDx.8TH_WORD registers.

5.1.7 MCL_EXCLUSIVE_AREA_18 With DMA, Used in function Mcl_IPW_DmaSetSoff, protects the TCDx.2ND_WORD, TCDx.6TH_WORD and TCDx.8TH_WORD registers.

5.1.8 MCL_EXCLUSIVE_AREA_19 With DMA, Used in function Mcl_IPW_DmaSetCiter, protects the TCDx.2ND_WORD, TCDx.6TH_WORD and TCDx.8TH_WORD registers.

5.1.9 MCL_EXCLUSIVE_AREA_20 With DMA, Used in function Mcl_IPW_DmaSetLinkAndIterCount, protects the TCDx.2ND_WORD, TCDx. 6TH_WORD and TCDx.8TH_WORD registers.

5.1.10 MCL_EXCLUSIVE_AREA_21 With DMA, Used in function Mcl_IPW_DmaSetDoff, protects the TCDx.2ND_WORD, TCDx.6TH_WORD and TCDx.8TH_WORD registers.

5.1.11 MCL_EXCLUSIVE_AREA_22 With DMA, Used in function Mcl_IPW_DmaSetIntMaj, protects the TCDx.2ND_WORD, TCDx.6TH_WORD and TCDx.8TH_WORD registers.

5.1.12 MCL_EXCLUSIVE_AREA_23 With DMA, Used in function Mcl_IPW_DmaClearIntMaj, protects the TCDx.2ND_WORD, TCDx.6TH_WORD and TCDx.8TH_WORD registers.

5.1.13 MCL_EXCLUSIVE_AREA_24 With DMA, Used in function Mcl_IPW_DmaSetFlags, protects the TCDx.2ND_WORD, TCDx.6TH_WORD and TCDx.8TH_WORD registers.

5.1.14 MCL_EXCLUSIVE_AREA_25 With DMA, Used in function Mcl_IPW_DmaSetBiter, protects the TCDx.2ND_WORD, TCDx.6TH_WORD and TCDx.8TH_WORD registers.

5.1.15 MCL_EXCLUSIVE_AREA_26 With DMA, Used in function Dma_ConfigLinkedTcd, protects the TCDx.2ND_WORD, TCDx.6TH_WORD and TCDx.8TH_WORD registers.

5.1.16 MCL_EXCLUSIVE_AREA_27 With DMA, Used in function Dma_ConfigScatterGatherTcd, protects the TCDx.2ND_WORD, TCDx.6TH_WORD and TCDx.8TH_WORD registers.

5.1.17 MCL_EXCLUSIVE_AREA_28 With DMA, Used in function Dma_DisableNotification, protects the TCDx.2ND_WORD, TCDx.6TH_WORD and TCDx.8TH_WORD registers.

5.1.18 MCL_EXCLUSIVE_AREA_29 With DMA, Used in function Dma_EnableNotification, protects the TCDx.2ND_WORD, TCDx.6TH_WORD and TCDx.8TH_WORD registers.

5.1.19 MCL_EXCLUSIVE_AREA_30 With DMA, Used in function Dma_ConfigScatterGatherChannel, protects the TCDx.2ND_WORD, TCDx. 6TH_WORD and TCDx.8TH_WORD registers.

5.1.20 MCL_EXCLUSIVE_AREA_31 With DMA, Used in function Dma_ConfigLinkedChannel protects the TCDx.2ND_WORD, TCDx.6TH_WORD and TCDx.8TH_WORD registers.

5.1.21 MCL_EXCLUSIVE_AREA_32 With LMem, Used in function Lmem_CacheDisablePc, protects the protects LMEM_PCCCR and LMEM_PSCCR registers (only on ARM platforms).

5.1.22 MCL_EXCLUSIVE_AREA_33 With LMem, Used in function Lmem_CacheDisablePs, protects the protects the protects LMEM_PCCCR and LMEM_PSCCR registers (only on ARM platforms).

5.1.23 MCL_EXCLUSIVE_AREA_34 With LMem, Used in function Lmem_CacheLaunchCommand, protects the protects LMEM_PCCCR and LMEM_PSCCR registers (only on ARM platforms).

Critical Region Exclusive Matrix

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

Table 5-1. Exclusive Areas

	MCL_EA_12	MCL_EA_13	MCL_EA_14	MCL_EA_15	MCL_EA_16	MCL_EA_17	MCL_EA_18	MCL_EA_19	MCL_EA_20	MCL_EA_21	MCL_EA_22	MCL_EA_23	MCL_EA_24	MCL_EA_25	MCL_EA_26	MCL_EA_27	MCL_EA_28	MCL_EA_29	MCL_EA_30	MCL_EA_31	MCL_EA_32	MCL_EA_33	MCL_EA_34
MCL_EA_12			x																				
MCL_EA_13			x																				

Table continues on the next page...

Table 5-1. Exclusive Areas (continued)

	MCL_EA_12	MCL_EA_13	MCL_EA_14	MCL_EA_15	MCL_EA_16	MCL_EA_17	MCL_EA_18	MCL_EA_19	MCL_EA_20	MCL_EA_21	MCL_EA_22	MCL_EA_23	MCL_EA_24	MCL_EA_25	MCL_EA_26	MCL_EA_27	MCL_EA_28	MCL_EA_29	MCL_EA_30	MCL_EA_31	MCL_EA_32	MCL_EA_33	MCL_EA_34
MCL_EA_14	x	x	x																				
MCL_EA_15					x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x			
MCL_EA_16				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x			
MCL_EA_17				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x			
MCL_EA_18				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x			

Table continues on the next page...

Table 5-1. Exclusive Areas (continued)

	MCL_EA_12	MCL_EA_13	MCL_EA_14	MCL_EA_15	MCL_EA_16	MCL_EA_17	MCL_EA_18	MCL_EA_19	MCL_EA_20	MCL_EA_21	MCL_EA_22	MCL_EA_23	MCL_EA_24	MCL_EA_25	MCL_EA_26	MCL_EA_27	MCL_EA_28	MCL_EA_29	MCL_EA_30	MCL_EA_31	MCL_EA_32	MCL_EA_33	MCL_EA_34
MCL_EA_19				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x			
MCL_EA_20				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x			
MCL_EA_21				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x			
MCL_EA_22				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x			
MCL_EA_23				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x			

Table continues on the next page...

Table 5-1. Exclusive Areas (continued)

	MCL_EA_12	MCL_EA_13	MCL_EA_14	MCL_EA_15	MCL_EA_16	MCL_EA_17	MCL_EA_18	MCL_EA_19	MCL_EA_20	MCL_EA_21	MCL_EA_22	MCL_EA_23	MCL_EA_24	MCL_EA_25	MCL_EA_26	MCL_EA_27	MCL_EA_28	MCL_EA_29	MCL_EA_30	MCL_EA_31	MCL_EA_32	MCL_EA_33	MCL_EA_34
MCL_EA_24				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x			
MCL_EA_25				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x			
MCL_EA_26				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x			
MCL_EA_27				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x			
MCL_EA_28				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x			

Table continues on the next page...

Table 5-1. Exclusive Areas (continued)

	MCL_EA_12	MCL_EA_13	MCL_EA_14	MCL_EA_15	MCL_EA_16	MCL_EA_17	MCL_EA_18	MCL_EA_19	MCL_EA_20	MCL_EA_21	MCL_EA_22	MCL_EA_23	MCL_EA_24	MCL_EA_25	MCL_EA_26	MCL_EA_27	MCL_EA_28	MCL_EA_29	MCL_EA_30	MCL_EA_31	MCL_EA_32	MCL_EA_33	MCL_EA_34
MCL_EA_29				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x			
MCL_EA_30				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x			
MCL_EA_31				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x			
MCL_EA_32																					x	x	x
MCL_EA_33																					x	x	x

Table continues on the next page...

Table 5-1. Exclusive Areas (continued)

	MCL_EA_12	MCL_EA_13	MCL_EA_14	MCL_EA_15	MCL_EA_16	MCL_EA_17	MCL_EA_18	MCL_EA_19	MCL_EA_20	MCL_EA_21	MCL_EA_22	MCL_EA_23	MCL_EA_24	MCL_EA_25	MCL_EA_26	MCL_EA_27	MCL_EA_28	MCL_EA_29	MCL_EA_30	MCL_EA_31	MCL_EA_32	MCL_EA_33	MCL_EA_34
MCL_EA_34																					x	x	x

Note

- **MCL_EA_xx** means **MCL_EXCLUSIVE_AREA_xx**

5.2 Peripheral Hardware Requirements

None.

5.3 ISR to configure within OS – dependencies

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

The ISR table is presented below. Depending on the derivative used, some of the ISRs may not be available. For complete details please consult the Reference Manual:

Table 5-2. eDMA 0 interrupts

eDMA 0 Interrupts	Hardware interrupt vector
MCL_DMA_CH_0_ISR	0
MCL_DMA_CH_1_ISR	1
MCL_DMA_CH_2_ISR	2
MCL_DMA_CH_3_ISR	3
MCL_DMA_CH_4_ISR	4
MCL_DMA_CH_5_ISR	5
MCL_DMA_CH_6_ISR	6

Table continues on the next page...

Table 5-2. eDMA 0 interrupts (continued)

eDMA 0 Interrupts	Hardware interrupt vector
MCL_DMA_CH_7_ISR	7
MCL_DMA_CH_8_ISR	8
MCL_DMA_CH_9_ISR	9
MCL_DMA_CH_10_ISR	10
MCL_DMA_CH_11_ISR	11
MCL_DMA_CH_12_ISR	12
MCL_DMA_CH_13_ISR	13
MCL_DMA_CH_14_ISR	14
MCL_DMA_CH_15_ISR	15
MCL_DMA0_ERROR_ISR	16

NOTE

In case of AUTOSAR_OS_NOT_USED, the compiler option "-DUSE_HW_VECTOR_MODE" must be added to the list of compiler options to be used with interrupt controller configured to be in hardware vector mode.

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

- **BASE:** The BASE module contains the common files/definitions needed by all MCAL modules.
- **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 'MclDevErrorDetect' configuration parameter.
- **Dem:** This module is necessary for enabling reporting of production relevant error status. The API function used is Dem_ReportErrorStatus().
- **Resource:** Sub-Derivative model is selected from Resource configuration.
- **RTE:** The RTE module is needed for implementing data consistency of exclusive areas that are used by Mcl module.

5.6 Data Cache Restriction

DMA transfers may issue cache coherency problems when D-CACHE is enabled and the buffers used as TCD source and destination are allocated in cacheable areas. To avoid possible coherency issues, the integrator has to ensure one of the following:

- The D-CACHE is disabled, or
- The D-CACHE is enabled and the buffers used as TCD source and destination are allocated in NON-CACHEABLE areas, or
- The D-CACHE is enabled, the buffers used as TCD source and destination are allocated in CACHEABLE areas and **MclSynchronizeCache** configuration parameter is enabled and the destination buffers used by DMA are start and end aligned to the cache line size(fill cache lines entirely).

If **MclSynchronizeCache** is enabled, cache clear and flush functions will be called at the beginning and end of MCL driver jobs, in order to maintain consistency between cache and the memory region modified by DMA.

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

Call-back Notifications:

None.

User Notification:

The MCL Driver provides a notification per channel. The ISRs shall be responsible for resetting the interrupt's flags (if needed by hardware) and calling the corresponding notification function. The notifications can be configured as pointers to user defined functions. If notification is not desired, NULL_PTR shall be configured.

Mcl_Notification_<Channel>

The syntax of this function is as follows:

void NotificationName

(

void

)

According to the last call of Mcl_EnableNotification, this notification function shall be called when the major iteration count completes or when the major iteration is half complete.

Chapter 7
Memory Allocation

7.1 Sections to be defined in MemMap.h

Tables descibe Sections to be defined in MemMap.h:

Table 7-1. Sectionto be define

<Section name>	Tyep of section	Description
MCL_START_SEC_CONFIG_DATA_UNSPECIFIED	Configuration Data	Start of Memory Section for Config Data.
MCL_STOP_SEC_CONFIG_DATA_UNSPECIFIED	Configuration Data	End of Memory Section for Config Data.
MCL_START_SEC_CODE	Code	Start of memory Section for Code in flash.
MCL_STOP_SEC_CODE	Code	Stop of memory Section for Code in flash.
MCL_START_SEC_RAMCODE	Code	Start of memory Section for Code in ram.
MCL_STOP_SEC_RAMCODE	Code	Stop of memory Section for Code in ram.
MCL_START_SEC_VAR_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 initialized with values after every reset.
MCL_STOP_SEC_VAR_INIT_UNSPECIFIED	Variables	End of above section.
MCL_START_SEC_VAR_INIT_16	Variables	Used for variables which have to be aligned to 16 bit. For instance used for variables of size 16 bit or used for composite data types: arrays,

Table continues on the next page...

Table 7-1. Sectionto be define (continued)

		structs containing elements of maximum 16 bits. These variables are initialized with values after every reset
MCL_STOP_SEC_VAR_INIT_16	Variables	End of above section.
MCL_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 (BBS).
MCL_STOP_SEC_VAR_NO_INIT_UNSPECIFIED	Variables	End of above section.

7.2 Linker command file

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

Chapter 8
Configuration parameters considerations

Configuration parameter class for MCL 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
Mcl	IMPLEMENTATION_CONFIG_VARIANT	Pre Compile parameter for all Variants of Configuration	Pre Compile
McIGeneral	MclDisableDemReportErrorStatus	Pre Compile parameter for all Variants of Configuration	Pre Compile
	MclDevErrorDetect	VariantPC or VariantPB	Post Build
	MclDmaNotificationSupported	VariantPC or VariantPB	Post Build
	MclErrorChecking	VariantPC or VariantPB	Post Build
	Mcl_VersionInfoApi	VariantPC or VariantPB	Post Build
	Mcl_DmaGetChannelErrorStatusApi	VariantPC or VariantPB	Post Build
	Mcl_DmaGetGlobalErrorStatusApi	VariantPC or VariantPB	Post Build
	Mcl_DeInitApi	VariantPC or VariantPB	Post Build
	EnableDMA	VariantPC or VariantPB	Post Build
	MclEnableTrgMux	VariantPC or VariantPB	Post Build
	MclEnableUserModeSupport	VariantPC or VariantPB	Post Build
	MclErrorNotificationDma0	VariantPC or VariantPB	Post Build
	MclLmemEnableCacheApi	VariantPC or VariantPB	Post Build
	MclSynchronizeCache	VariantPC or VariantPB	Post Build
	MclLmemEnableWriteBuffer	VariantPC or VariantPB	Post Build
	MclLmemCacheTimeout	VariantPC or VariantPB	Post Build

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
MclDemEventParameterRefs	MCL_DMA_E_DESCRIPTOR	Pre Compile parameter for all Variants of Configuration	Pre Compile
	MCL_DMA_E_ECC	Pre Compile parameter for all Variants of Configuration	Pre Compile
	MCL_DMA_E_BUS	Pre Compile parameter for all Variants of Configuration	Pre Compile
	MCL_DMA_E_PRIORITY	Pre Compile parameter for all Variants of Configuration	Pre Compile
	MCL_DMA_E_INCONSISTENCY	Pre Compile parameter for all Variants of Configuration	Pre Compile
	MCL_DMA_E_UNRECOGNIZED	Pre Compile parameter for all Variants of Configuration	Pre Compile
CommonPublishedInformation	ArReleaseMajorVersion	Pre Compile parameter for all Variants of Configuration	Pre Compile
	ArReleaseMinorVersion	Pre Compile parameter for all Variants of Configuration	Pre Compile
	ArReleaseRevisionVersion	Pre Compile parameter for all Variants of Configuration	Pre Compile
	ModuleId	Pre Compile parameter for all Variants of Configuration	Pre Compile
	SwMajorVersion	Pre Compile parameter for all Variants of Configuration	Pre Compile
	SwMinorVersion	Pre Compile parameter for all Variants of Configuration	Pre Compile
	SwPatchVersion	Pre Compile parameter for all Variants of Configuration	Pre Compile
	VendorApiInfix	Pre Compile parameter for all Variants of Configuration	Pre Compile
	VendorId	Pre Compile parameter for all Variants of Configuration	Pre Compile
MclIsrAvailable	MclIsrName	VariantPC or VariantPB	Post Build
	MclIsrEnabled	VariantPC or VariantPB	Post Build
MclConfigSet/DmaInstance	MclEDMA_CX	VariantPC or VariantPB	Post Build
	MclEDMA_ECX	VariantPC or VariantPB	Post Build
	MclEDMA_HALT	VariantPC or VariantPB	Post Build
	MclEDMA_HOE	VariantPC or VariantPB	Post Build
	MclEDMA_ERGA	VariantPC or VariantPB	Post Build
	MclEDMA_ERCA	VariantPC or VariantPB	Post Build
	MclEDMA_EDBG	VariantPC or VariantPB	Post Build
MclConfigSet/DMAChannel	MclDMAChannelId	VariantPC or VariantPB	Post Build
	DmaHwChannel	VariantPC or VariantPB	Post Build
	DMAChannelPriority	VariantPC or VariantPB	Post Build
	ECP	VariantPC or VariantPB	Post Build

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
	DPA	VariantPC or VariantPB	Post Build
	EMI	VariantPC or VariantPB	Post Build
	MclDmaTransferCompletionNotif	VariantPC or VariantPB	Post Build
	MclDMAChannelEnable	VariantPC or VariantPB	Post Build
	MclDMAChannelTriggerEnable	VariantPC or VariantPB	Post Build
	DmaSource0	VariantPC or VariantPB	Post Build
MclConfigSet/TriggerMux	TrgMuxDmaMux0Input0	VariantPC or VariantPB	Post Build
	TrgMuxDmaMux0Input1	VariantPC or VariantPB	Post Build
	TrgMuxDmaMux0Input2	VariantPC or VariantPB	Post Build
	TrgMuxDmaMux0Input3	VariantPC or VariantPB	Post Build
	TrgMuxDmaMux0LockEn	VariantPC or VariantPB	Post Build
	TrgMuxXOut0Input0	VariantPC or VariantPB	Post Build
	TrgMuxXOut0Input1	VariantPC or VariantPB	Post Build
	TrgMuxXOut0Input2	VariantPC or VariantPB	Post Build
	TrgMuxXOut0Input3	VariantPC or VariantPB	Post Build
	TrgMuxXOut0LockEn	VariantPC or VariantPB	Post Build
	TrgMuxXOut1Input0	VariantPC or VariantPB	Post Build
	TrgMuxXOut1Input1	VariantPC or VariantPB	Post Build
	TrgMuxXOut1Input2	VariantPC or VariantPB	Post Build
	TrgMuxXOut1Input3	VariantPC or VariantPB	Post Build
	TrgMuxXOut1LockEn	VariantPC or VariantPB	Post Build
	TrgMuxAdc0Input0	VariantPC or VariantPB	Post Build
	TrgMuxAdc0Input1	VariantPC or VariantPB	Post Build
	TrgMuxAdc0Input2	VariantPC or VariantPB	Post Build
	TrgMuxAdc0Input3	VariantPC or VariantPB	Post Build
	TrgMuxAdc0LockEn	VariantPC or VariantPB	Post Build
	TrgMuxCmp0Input0	VariantPC or VariantPB	Post Build
	TrgMuxCmp0LockEn	VariantPC or VariantPB	Post Build
	TrgMuxFtm0Input0	VariantPC or VariantPB	Post Build
	TrgMuxFtm0Input1	VariantPC or VariantPB	Post Build
	TrgMuxFtm0Input2	VariantPC or VariantPB	Post Build
	TrgMuxFtm0Input3	VariantPC or VariantPB	Post Build
	TrgMuxFtm0LockEn	VariantPC or VariantPB	Post Build
	TrgMuxFtm1Input0	VariantPC or VariantPB	Post Build
	TrgMuxFtm1Input1	VariantPC or VariantPB	Post Build
	TrgMuxFtm1Input2	VariantPC or VariantPB	Post Build
	TrgMuxFtm1Input3	VariantPC or VariantPB	Post Build
	TrgMuxFtm1LockEn	VariantPC or VariantPB	Post Build

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
	TrgMuxFtm2Input0	VariantPC or VariantPB	Post Build
	TrgMuxFtm2Input1	VariantPC or VariantPB	Post Build
	TrgMuxFtm2Input2	VariantPC or VariantPB	Post Build
	TrgMuxFtm2Input3	VariantPC or VariantPB	Post Build
	TrgMuxFtm2LockEn	VariantPC or VariantPB	Post Build
	TrgMuxFtm3Input0	VariantPC or VariantPB	Post Build
	TrgMuxFtm3Input1	VariantPC or VariantPB	Post Build
	TrgMuxFtm3Input2	VariantPC or VariantPB	Post Build
	TrgMuxFtm3Input3	VariantPC or VariantPB	Post Build
	TrgMuxFtm3LockEn	VariantPC or VariantPB	Post Build
	TrgMuxPdb0Input0	VariantPC or VariantPB	Post Build
	TrgMuxPdb0LockEn	VariantPC or VariantPB	Post Build
	TrgMuxPdb1Input0	VariantPC or VariantPB	Post Build
	TrgMuxPdb1LockEn	VariantPC or VariantPB	Post Build
	TrgMuxFlexIoInput0	VariantPC or VariantPB	Post Build
	TrgMuxFlexIoInput1	VariantPC or VariantPB	Post Build
	TrgMuxFlexIoInput2	VariantPC or VariantPB	Post Build
	TrgMuxFlexIoInput3	VariantPC or VariantPB	Post Build
	TrgMuxFlexIoLockEn	VariantPC or VariantPB	Post Build
	TrgMuxLpitInput0	VariantPC or VariantPB	Post Build
	TrgMuxLpitInput1	VariantPC or VariantPB	Post Build
	TrgMuxLpitInput2	VariantPC or VariantPB	Post Build
	TrgMuxLpitInput3	VariantPC or VariantPB	Post Build
	TrgMuxLpitLockEn	VariantPC or VariantPB	Post Build
	TrgMuxLpuart0Input0	VariantPC or VariantPB	Post Build
	TrgMuxLpuart0LockEn	VariantPC or VariantPB	Post Build
	TrgMuxLpuart1Input0	VariantPC or VariantPB	Post Build
	TrgMuxLpuart1LockEn	VariantPC or VariantPB	Post Build
	TrgMuxLpi2c0Input0	VariantPC or VariantPB	Post Build
	TrgMuxLpi2c0LockEn	VariantPC or VariantPB	Post Build
	TrgMuxLpspi0Input0	VariantPC or VariantPB	Post Build
	TrgMuxLpspi0LockEn	VariantPC or VariantPB	Post Build
	TrgMuxLpspi1Input0	VariantPC or VariantPB	Post Build
	TrgMuxLpspi1LockEn	VariantPC or VariantPB	Post Build
	TrgMuxLptmr0Input0	VariantPC or VariantPB	Post Build
	TrgMuxLptmr0LockEn	VariantPC or VariantPB	Post Build
	TrgMuxLpi2c1Input0	VariantPC or VariantPB	Post Build
	TrgMuxLpi2c1LockEn	VariantPC or VariantPB	Post Build
	TrgMuxFtm4Input0	VariantPC or VariantPB	Post Build

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
	TrgMuxFtm4LockEn	VariantPC or VariantPB	Post Build
	TrgMuxFtm5Input0	VariantPC or VariantPB	Post Build
	TrgMuxFtm5LockEn	VariantPC or VariantPB	Post Build
	TrgMuxFtm6Input0	VariantPC or VariantPB	Post Build
	TrgMuxFtm6LockEn	VariantPC or VariantPB	Post Build
	TrgMuxFtm7Input0	VariantPC or VariantPB	Post Build
	TrgMuxFtm7LockEn	VariantPC or VariantPB	Post Build

Chapter 9
Integration Steps

This section gives a brief overview of the steps needed for integrating MicroController Library :

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

Chapter 10
External Assumptions for MCL driver

The section presents requirements that must be complied with when integrating MCL 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_EXT176]

<< The integrator shall assure that (MSN)_Init() and (MSN)_DeInit() functions do not interrupt each other. >>

[SMCAL_CPR_EXT177]

<< When caches are enabled and data buffers are allocated in cachable memory regions the buffers involved in DMA transfer shall be aligned with both start and end to cache line size.

>>

NOTE

Rationale: This ensures that no other buffers/variables to compete for the same cache lines.

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Document Number IM2MCLASR4.2 Rev0002R1.0.1
Revision 1.0