Integration Manual

for MPC5634M ICU Driver

Document Number: IM14ICUASR3.0R2.0.0
Rev. 2.0

freescale™

Contents

Section Number **Title** **Page**

Chapter 1
Revision History

Chapter 2
Introduction

Chapter 3
Building the Driver

Chapter 4
Function calls to module

Chapter 5
Module Requirements

Chapter 6
Main API Requirements

Chapter 7
Memory Allocation

Chapter 8
CONFIGURATION PARAMETER CONSIDERATIONS

Chapter 9
Integration steps

Chapter 1
Revision History

Table 1-1. Revision History

Revision	Date	Author	Description
1.0	27.01.2011	Chandrakanth.P	Initial Version
2.0	27.11.2011	Syed Hussaini	Updated for Monacco RTM2.0.0

Chapter 2
Introduction

This integration manual describes the integration requirements for AUTOSAR microcontroller abstraction layer (MCAL) ICU Driver MPC5634M microcontrollers.

The roadmap for the document is as follows:

Building the Driver

This section gives a brief overview of the build procedure (compiler,linker options and source files) and Plugins setup.

Function calls to module

This section lists the various function calls to modules during Start-up, Shutdown and Wake-up.

Module Requirements

This section specifies the various module requirements related to

- Exclusive areas to be defined in BSW scheduler
- Peripheral Hardware Requirements
- Specific interface to other modules
- ISR to configure within OS
- Dependencies with other AUTOSAR modules

Main API Requirements

This section specifies the requirements related to to the main API and gives a brief overview of the main functions calls within BSW scheduler, API_Name Requirements and calls to notification functions, callbacks, callouts.

Memory Allocation

This section describes the memory allocation requirements namely the sections to be defined in MemMap.h and the linker command file.

CONFIGURATION PARAMETER CONSIDERATIONS

This section covers the various Pre Compile, Link Time and Post Build time configuration parameters.

Integration steps

This section describes in brief the steps for integrating ICU module.

2.1 Supported Derivatives

The software described in this document is intended to be used with the following microcontroller devices of Freescale Semiconductor:

Table 2-1. MPC5634M Supported Derivatives

Freescale Semiconductor	mpc5634m_bga208, mpc5634m_qfp144, mpc5634m_qfp176

All of the above microcontroller devices are collectively named as **MPC5634M**.

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

Abbreviation and Definitions	Description
BSW	Basic Software
DEM	Diagnostic Event Manager
DET	Development Error Tracer
ECU	Electronic Control Unit
ICU	Input Capture Unit
ISR	interrupt Service Routine
OS	Operating System
RAM	Random Access Memory
ROM	Read-only Memory
MCU	Microcontroller Unit
GUI	Graphical User Interface
EcuM	ECU state Manager
API	Application Programming Interface
PB Variant	Post Build Variant
PC Variant	Pre Compile Variant

2.5 Reference List

Table 2-3. Reference List

#	Title	Version
1	AUTOSAR 3.0ICU Driver Software Specification Document.	V2.2.0 R3.0 Rev 0001
2	MPC5634M Reference Manual	Rev. 6, 4 October 2011

Chapter 3
Building the Driver

This section describes the source files and various compiler, linker options used for building the Autosar ICU driver for Freescale SemiconductorMPC5634M microcontrollers. It also explains the Plugins setup procedure.

3.1 Build Options

The ICU driver files are compiled using

- GHS 5.1.7
- DIAB 5_8_0_02 wind00198363 20100511 123238
- CW Version 4.3 build 182
- COSMIC Software PPC C Cross Compiler V4.3.4 - 16 Nov 2011 - Win32-F

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

Note

The TS_T2D14M20I0R0 plugin name is composed as follows:

TS_T = Target_Id

D = Derivative_Id

M = SW_Version_Major

I = SW_Version_Minor

R = Revision

(i.e. Target_Id = 2 identifies PowerPC architecture and
Derivative_Id = 14 identifies the MPC5634M)

3.1.1 GHS Compiler/Linker/Assembler Options

Table 3-1. Compiler Options

Option	Description
-cpu=ppc563xm	Selects target processor: ppc563xm
-ansi	Enforces strict ANSI mode (C89 standard)
-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.
--no_commons	Allocates uninitialized global variables to a section and initializes them to zero at program startup. This may improve optimizations by giving the compiler optimizer more information about the location of the variable.
-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
-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
-DGHS	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the GHS preprocessor symbol.
-DEU_DISABLE_ANSILIB_CALLS	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the EU_DISABLE_ANSILIB_CALLS preprocessor symbol.
-DMCAL_CER_VALIDATION	-D defines a preprocessor symbol for CER Report
-DMCAL_VERSION_CHECK	-D defines enable the cross check between the AutoSar component Version Numbers

Table 3-2. Assembler Options

Option	Description
-cpu=ppc563xm	Selects target processor: ppc563xm

Table 3-3. Linker Options

Option	Description
-cpu=ppc563xm	Selects target processor: ppc563xm
-nostartfiles	Do not use Start files.
-vle	Enables VLE code generation
-linker_warnings	Display linker warnings

3.1.2 DIAB Compiler/Linker/Assembler Options

Table 3-4. Compiler Options

Option	Description
-tPPCE200Z3VEG:simple	Sets target processor to PPCE200Z3, generates ELF using EABI conventions, All Single Hardware Floating Point (Single precision uses hardware, double precision is mapped to single precision), 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
-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.
-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
-Xalign-functions=4	Align each function on an address boundary divisible by 4
-g	Generate symbolic debugger information. Do most target-independent optimizations. Also, disable most target-dependent optimizations: option -g2 also disables basic reordering and all peephole optimizations.
-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
-Xforce-prototypes	Generate warnings if a function is used without a previous prototype declaration
-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
-Xlint	Generate warnings when suspicious and non-portable C code is encountered. Enables all warnings

Table continues on the next page...

Table 3-4. Compiler Options (continued)

Option	Description
-ei1604	Suppress the warning messages 1604.
-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
-DDIAB	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the DIAB preprocessor symbol.
-DEU_DISABLE_ANSILIB_CALLS	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the EU_DISABLE_ANSILIB_CALLS preprocessor symbol.
-DMCAL_CER_VALIDATION	-D defines a preprocessor symbol for CER Report

Table 3-5. Assembler Options

Option	Description
-tPPCE200Z3VEN:simple	Selects target processor: PPCE200Z3, generates ELF using EABI conventions, NO floating point support, 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

Table 3-6. Linker Options

Option	Description
-tPPCE200Z3VEN:simple	Selects target processor: PPCE200Z3, generates ELF using EABI conventions, NO floating point support, 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
-lc	Specifies to linker to search for libc.a
-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.
-Xlibc-old	Enables usage of legacy (pre-release 5.6) libraries

3.1.3 CW Compiler/Linker/Assembler Options

Table 3-7. Compiler Options

Option	Description
-proc Zen	Generates and links object code for Zen processor. The compiler uses unsigned as the default parameter for the -char switch
-lang c	Expects source code to conform to the language specified by the ISO/IEC 9899-1990 ("C90") standard
-opt all	This option is selected all optimization (the same as -opt speed,level=4,intrinsics,noframe)
-common off	Disables moving uninitialized data into a common section
-sdatathreshold 0	Specifies the threshold size (in bytes) for an item considered by the linker to be small data. (The linker stores small data items in the Small Data address space. The compiler can generate faster code to access this data.)
-sdata2threshold 0	Specifies the threshold size (in bytes) for an item considered by the linker to be small constant data. (The linker stores small constant data items in the Small Constant Data address space.)
-vle	Tells the compiler and linker to generate and lay out Variable Length Encoded (VLE) instructions, available on Zen variants of Power Architecture processors
-use_lmw_stmw on	Enables the use of multiple load and store instructions for function prologues and epilogues
-ir	Include the debug information
-ppc_asm_to_vle	Converts regular Power Architecture assembler mnemonics to equivalent VLE (Variable Length Encoded) assembler mnemonics in the inline assembler
-cpp_exceptions off	When on, generates executable code for C++ exceptions. When off, generates smaller, faster executable code
-func_align 4	Specifies alignment of functions in executable code
-sym dwarf-2,full	Generate DWARF-2-conforming debugging information (Debug With Arbitrary Record Format)
-gdwarf-2	Generate DWARF-2-conforming debugging information (Debug With Arbitrary Record Format). The linker ignores debugging information that is not in the Dwarf 1, Dwarf 2 format
-w on	Turns on most warning messages
-r	Compiler should expect function prototypes
-w undefmacro	Issues warning messages on the use of undefined macros in #if and #elif conditionals
-char unsigned	Controls the default sign of the char data type: char data items are unsigned
-nosyspath	Performs a search of both the user and system paths, treating #include statements of the form #include xyz the same as the form #include "xyz"
-fp none	No floating point code generation
-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
-DEU_DISABLE_ANSILIB_CALLS	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the EU_DISABLE_ANSILIB_CALLS preprocessor symbol.
-DMCAL_CER_VALIDATION	-D defines a preprocessor symbol for CER Report

Table continues on the next page...

Table 3-7. Compiler Options (continued)

Option	Description
-DMCAL_VERSION_CHECK	-D defines enable the cross check between the AutoSar component Version Numbers
-DMWERKS	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the CWpreprocessor symbol.

Table 3-8. Assembler Options

Option	Description
-proc Zen	Generates and links object code for Zen processor. The compiler uses unsigned as the default parameter for the -char switch
-vle	Tells the compiler and linker to generate and lay out Variable Length Encoded (VLE) instructions, available on Zen variants of Power Architecture processors
-sym dwarf-2,full	Generate DWARF-2-conforming debugging information (Debug With Arbitrary Record Format)
-gdwarf-2	Generate DWARF-2-conforming debugging information (Debug With Arbitrary Record Format). The linker ignores debugging information that is not in the Dwarf 1, Dwarf 2 format.

Table 3-9. Linker Options

Option	Description
-proc Zen	Generates and links object code for Zen processor. The compiler uses unsigned as the default parameter for the -char switch
-code_merging all	Removes duplicated functions to reduce object code size
-far_near_addressing	Simplifies address computations to reduce object code size and improve performance
-vle_enhance_merging	Removes duplicated functions that are called by functions that use VLE instructions to reduce object code size
-listdwarf	DWARF debugging information in the linker's map file
-sym dwarf-2,full	Generate DWARF-2-conforming debugging information (Debug With Arbitrary Record Format)
-char unsigned	Controls the default sign of the char data type: char data items are unsigned.

3.1.4 CSMC Compiler/Linker/Assembler Options

Table 3-10. Compiler Options

Option	Description
-l	Create listing file; this option directs the compiler to produce an assembly language file with C source line interspersed in it. Please note that the C source lines are commented in the assembly language file: they start with ';'.
+modvc	Memory model with "medium size" application, in detail: "data" less than 64kb, "constants" less than 64kb, no code size limit

Table continues on the next page...

Table 3-10. Compiler Options (continued)

Option	Description
+rev	Tells the compiler to reverse the order of bits in the bitfields. You need this option in order to use most non-Cosmic header files.
-pc99	authorize the repetition of the const and volatile modifiers in the declaration either directly or indirectly in the typedef.
-odB5	disable the optimization B5.
-pxf	prefix filenames in the debug information with absolute full path name.
+debug	produce debug information to be used by the debug utilities provided with the compiler and by any external debugger.
-DCSMC	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the CSMC preprocessor symbol.
-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
-DEU_DISABLE_ANSILIB_CALLS	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the EU_DISABLE_ANSILIB_CALLS preprocessor symbol.
-DMCAL_CER_VALIDATION	-D defines a preprocessor symbol for CER Report
-DMCAL_VERSION_CHECK	-D defines enable the cross check between the AutoSar component Version Numbers

Table 3-11. Assembler Options

Option	Description
-l	create a listing file. The name of the listing file is derived from the input file name by replacing the suffix by the ".ls" extension

Table 3-12. Linker Options

Option	Description
-p	display symbols with physical address instead of logical address in the map file.

3.2 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:

1. OS is not used - AUTOSAR_OS_NOT_USED is defined:
   - 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.

- 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.

2. Freescale Semiconductor 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.
3. 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.

3.3 Files required for Compilation

This section describes the include files required to compile, assemble (if assembler code) and link the Autosar ICU driver for Freescale SemiconductorMPC5634M 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.

ICU Files:

Icu_TS_T2D14M20I0R0

Table 3-13. Include files for compilation

include files	Icu.h
	eMIOS_Icu_LLD_CfgEx.h
	eMIOS_Icu_LLD.h
	Reg_eSys_EMIOS.h
	Reg_eSys_EMIOS_CfgEx.h
	Reg_eSys_SIU.h
	Icu_LLD.h
	SIU_Icu_LLD.h
	SIU_Icu_LLD_CfgEx.h

Table 3-14. Source files for compilation

source files	Icu.c
	eMIOS_Icu_LLD_IRQ.c
	eMIOS_Icu_LLD.c
	Icu_LLD.c
	Reg_eSys_EMIOS_Icu.c
	SIU_Icu_LLD.c
	SIU_Icu_LLD_IRQ.c

Icu_Cfg.c (For PC Variant) - This file should be generated by the user using a configuration tool for compilation

Icu_PBcfg.c (For PB Variant) - This file should be generated by the user using a configuration tool for compilation

Icu_Cfg.h - This file should be generated by the user using a configuration tool for compilation

Other include files:

- **Files from Base folder:**
  - ..\Base_TS_T2D14M20I0R0include\Compiler.h
  - ..\Base_TS_T2D14M20I0R0include\Compiler_Cfg.h
  - ..\Base_TS_T2D14M20I0R0include\MemMap.h
  - ..\Base_TS_T2D14M20I0R0include\Platform_Types.h
  - ..\Base_TS_T2D14M20I0R0include\Std_Types.h
  - ..\Base_TS_T2D14M20I0R0include\Reg_eSys.h
  - ..\Base_TS_T2D14M20I0R0\include\Reg_Macros.h
  - ..\Base_TS_T2D14M20I0R0\include\Mcal.h
  - ..\Base_TS_T2D14M20I0R0\include\ComStack_Types.h
  - ..\Base_TS_T2D14M20I0R0\include\Soc_Ips.h
- **Files from Dem folder:**
  - ..\Dem_TS_T2D14M20I0R0\include\Dem.h
  - ..\Dem_TS_T2D14M20I0R0\include \Dem_IntErrId.h
  - ..\Dem_TS_T2D14M20I0R0\include \Dem_Types.h
- **Files from Det folder:**
  - ..\Det_TS_T2D14M20I0R0\include\Det.h
- **Files from SchM folder:**
  - ..\SchM_TS_T2D14M20I0R0include\SchM_Icu.h
- **Files from Ecum folder:**
  - ..\EcuM_TS_T2D14M20I0R0\include \EcuM_Cbk.h

3.4 Setting up the Plugins

All the Autosar MCAL drivers for MPC5634M were designed to be configured using Tresos® Studio configuration and code generation tool.

Location of various files inside the plugin folder is explained below.

Module Parameter Definition File:

..\Icu_TS_T2D14M20I0R0\config\ Icu.xdm

..\EcuM_TS_T2D14M20I0R0\config\EcuM.xdm

Code Generation Templates for Pre-Compile time configuration parameters:

..\Icu_TS_T2D14M20I0R0\generate_PC\src\Icu_Cfg.c

..\Icu_TS_T2D14M20I0R0\generate_PC\include\Icu_Cfg.h

..\EcuM_TS_T2D14M20I0R0\generate_PC\include\EcuM_Cfg.h

Code Generation Templates for Post-Build time configuration parameters:

..\Icu_TS_T2D14M20I0R0\generate_PB\src\Icu_PBcfg.c

Steps to generate configurations:

1. Copy the module folder (Icu_TS_T2D14M20I0R0) , (EcuM_TS_T2D14M20I0R0) into the Tresos plugins folder.
2. Set the desired Tresos Output location folder for the generated sources and header files.
3. Use the Tresos GUI to modify configuration parameters values.
4. Generate the Pre-Compile and Post-Build files.

Chapter 4
Function calls to module

4.1 Function Calls during Startup

This driver does not need OS Support except for ISRs. Hence can be initialized either in STARTUP1 or STARTUP2 phase of EcuM initialization. This depends on the implementation, desired duration for STARTUP1 & Target hardware design. The API to be called is Icu_Init(ConfigPtr).

NOTE

For proper driver usage, prior MCU and PORT modules initialization should be done.

4.2 Function Calls during Shutdown

Icu_SetMode(ICU_MODE_SLEEP) API shall be called during GO SLEEP phase of EcuM to configure the hardware for Sleep mode.

4.3 Function Calls during Wakeup

The ICU shall report the wakeup event to EcuM through EcuM_CheckWakeupEvent (event) upon a wakeup event.

Chapter 5
Module Requirements

5.1 Exclusive areas to be defined in BSW scheduler

In the current implementation, ICU is using the services of Schedule Manager (SchM) for entering and exiting the critical regions. SchM implementation is done by the integrators of the MCAL using OS or non-OS services. For testing the ICU, stubs are used for SchM.

All ICU notification functions are called outside any critical region. Global variables updates are performed by ISRs before calling the user notification functions. So the ICU internal state is consistent at the moment of the notification call.

The ISR critical regions must not block the other critical regions to avoid deadlocks. This is ensured by exiting the ISR critical region before calling the user notification functions.

The following critical regions are used in the ICU driver:

5.1.1 ICU_EXCLUSIVE_AREA_00

Used in function Icu_GetTimeElapsed, protects against concurrent access in

- write to Icu_Period
- write to Icu_ActivePulseWidth

5.1.2 ICU_EXCLUSIVE_AREA_01

Used in function Icu_GetDutyCycleValues, protects against concurrent access in

- write to Icu_ActivePulseWidth
- write to Icu_Period

5.1.3 ICU_EXCLUSIVE_AREA_02

Used in function Icu_LLD_GetBitChState, protects against concurrent access in

- read from Icu_ChannelState

5.1.4 ICU_EXCLUSIVE_AREA_03

Used in function Icu_LLD_SetBitChState, protects against concurrent access in

- write to Icu_ChannelState

5.1.5 ICU_EXCLUSIVE_AREA_04

Used in function Icu_LLD_ClearBitChState, protects against concurrent access in

- write to Icu_ChannelState

5.1.6 ICU_EXCLUSIVE_AREA_05

Used in function Icu_LLD_StartTimestamp, protects against concurrent access in

- write to Icu_Buffer
- write to Icu_BufferIndex
- write to Icu_BufferNotify
- write to Icu_BufferSize
- write to Icu_NotifyCount

5.1.7 ICU_EXCLUSIVE_AREA_06

Used in function Icu_LLD_GetTimestampIndex, protects against concurrent access in

- read from Icu_Buffer
- read from Icu_BufferIndex

5.1.8 ICU_EXCLUSIVE_AREA_07

Used in function Icu_LLD_ResetEdgeCount, protects against concurrent access in

- write to Icu_EdgeCount

5.1.9 ICU_EXCLUSIVE_AREA_08

Used in function Icu_LLD_DisableEdgeCount, protects against concurrent access in

- write to Icu_EdgeCount

5.1.10 ICU_EXCLUSIVE_AREA_09

Used in function Icu_LLD_GetEdgeNumbers, protects against concurrent access in

- read from Icu_EdgeCount

5.1.11 ICU_EXCLUSIVE_AREA_10

Used in function Icu_LLD_StartSignalMeasurement, protects against concurrent access in

- write to Icu_ActivePulseWidth
- write to Icu_Int_Counter
- write to Icu_Period

5.1.12 ICU_EXCLUSIVE_AREA_11

Used in function Icu_LLD_SignalMeasurement, protects against concurrent access in

- write to Icu_ActivePulseWidth
- write to Icu_Int_Counter
- write to Icu_Start

5.1.13 ICU_EXCLUSIVE_AREA_12

Used in function Icu_LLD_Timestamp, protects against concurrent access in

- write to Icu_Buffer
- write to Icu_BufferIndex
- read from Icu_BufferNotify

- read from Icu_Buffer_Size
- write to Icu_NotifyCount

5.1.14 ICU_EXCLUSIVE_AREA_13

Used in function eMIOS_LLD_EnableInterrupt, protects against concurrent access in

- write to EMIOS Reg. CSR
- write to EMIOS Reg. CCR

5.1.15 ICU_EXCLUSIVE_AREA_14

Used in function eMIOS_LLD_DisableInterrupt, protects against concurrent access in

- write to EMIOS Reg. CCR

5.1.16 ICU_EXCLUSIVE_AREA_15

Used in function eMIOS_LLD_UCSetMode, protects against concurrent access in

- write to EMIOS Reg. CCR

5.1.17 ICU_EXCLUSIVE_AREA_16

Used in function eMIOS_LLD_GetInputState, protects against concurrent access in

- write to EMIOS Reg. CSR

5.1.18 ICU_EXCLUSIVE_AREA_17

Used in function SIU_LLD_EnableInterrupt, protects against concurrent access in

- write to SIU Reg. EISR
- write to SIU Reg. DIRER

5.1.19 ICU_EXCLUSIVE_AREA_18

Used in function SIU_LLD_DisableInterrupt, protects against concurrent access in

- write to SIU Reg. EISR
- write to SIU Reg. DIRER

5.2 Critical region exclusivity matrix

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

	ICU_EXCLUSIVE_AREA_00	ICU_EXCLUSIVE_AREA_01	ICU_EXCLUSIVE_AREA_02	ICU_EXCLUSIVE_AREA_03	ICU_EXCLUSIVE_AREA_04	ICU_EXCLUSIVE_AREA_05	ICU_EXCLUSIVE_AREA_06	ICU_EXCLUSIVE_AREA_07	ICU_EXCLUSIVE_AREA_08	ICU_EXCLUSIVE_AREA_09	ICU_EXCLUSIVE_AREA_10	ICU_EXCLUSIVE_AREA_11	ICU_EXCLUSIVE_AREA_12	ICU_EXCLUSIVE_AREA_13	ICU_EXCLUSIVE_AREA_14	ICU_EXCLUSIVE_AREA_15
ICU_EXCLUSIVE_AREA_00	x	x														
ICU_EXCLUSIVE_AREA_01	x	x														
ICU_EXCLUSIVE_AREA_02			x													
ICU_EXCLUSIVE_AREA_03				x												
ICU_EXCLUSIVE_AREA_04					x											
ICU_EXCLUSIVE_AREA_05						x										
ICU_EXCLUSIVE_AREA_06							x									
ICU_EXCLUSIVE_AREA_07								x								
ICU_EXCLUSIVE_AREA_08									x							
ICU_EXCLUSIVE_AREA_09										x						
ICU_EXCLUSIVE_AREA_10											x					
ICU_EXCLUSIVE_AREA_11												x				
ICU_EXCLUSIVE_AREA_12													x			
ICU_EXCLUSIVE_AREA_13														x		
ICU_EXCLUSIVE_AREA_14															x	
ICU_EXCLUSIVE_AREA_15																x

Figure 5-1. Critical region table

5.3 Peripheral Hardware Requirements

EMIOS channels 0-23 depending on MPC5634M derivative and external interrupts IRQ0–IRQ15 are available for the ICU driver..

Refer Table ICU Hardware Channel availability for MPC5634M family in User Manual

5.4 ISR to configure within OS – dependencies

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

The ISR table is presented below; depending on the derivative used some of the ISR may not be available, for complete details please consult the Reference Manual:

Table 5-1. eMIOS 0 Interrupt Service Routine

EMIOS 0 Interrupts	Hardware interrupt vector
EMIOS_0_CH_0_ISR	51
EMIOS_0_CH_1_ISR	52
EMIOS_0_CH_2_ISR	53
EMIOS_0_CH_4_ISR	55
EMIOS_0_CH_8_ISR	59
EMIOS_0_CH_9_ISR	60
EMIOS_0_CH_10_ISR	61
EMIOS_0_CH_11_ISR	62
EMIOS_0_CH_12_ISR	63
EMIOS_0_CH_13_ISR	64
EMIOS_0_CH_14_ISR	65
EMIOS_0_CH_15_ISR	66
EMIOS_0_CH_23_ISR	209

Table 5-2. SIU Interrupt Service Routine

SIU IRQ Interrupts	Hardware interrupt vector
SIU_EXT_IRQ0	46
SIU_EXT_IRQ1	47
SIU_EXT_IRQ3	49
SIU_EXT_IRQ4_15	50

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.

5.5 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:

1. OS is not used - AUTOSAR_OS_NOT_USED is defined:
   - 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.
   - 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.
2. Freescale Semiconductor 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.
3. 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.

   Please refer to the OS documentation for description of the ISR macro.

5.6 Other AUTOSAR modules - dependencies

Development Error Tracer:

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 the 'IcuDevErrorDetect' configuration parameter.

Diagnostic Event Manager:

This module is necessary for enabling reporting of production relevant error status. Since there are no production relevant error codes in ICU this is not used.

ECU State Manager:

This module is used for processing the Wakeup notifications of ICU. Whenever the module is in ‘Sleep’ mode and a wakeup event occurs on a wakeup capable channel, it is reported to EcuM through the EcuM_CheckWakeupEvent () API. This is configurable using the ‘IcuChannelWakeupInfo’ configuration parameter.

Configuration dependency to other module:

For generating configuration files of ICU, EcuM also is required as ICU refers to EcuM parameter. EcuM need to be configure first before generating configuration files of ICU.

Hence template files for EcuM are provided at

..\EcuM_<plugin_name>\autosar\EcuM.epd (Module Parameter Definition File – AUTOSAR Format)

..\EcuM_<plugin_name>\config\EcuM.xdm (Module Parameter Definition File – Tresos Format)

Chapter 6
Main API Requirements

6.1 Main functions calls within BSW scheduler

None

6.2 Calls to notification functions, callbacks, callouts

Call-back Notifications:

None.

User Notification:

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

Icu_SignalNotification_<Channel>

The syntax of this function is as follows:

void NotificationName

(

void

)

According to the last call of Icu_EnableNotification, this notification function shall be called if the requested signal edge (rising / falling / both edges) occurs (once per edge).

Icu_TimestampNotification_<Channel>

The syntax of this function is as follows:

void TimestampNotificationName

(

void

)

This notification shall be called if the number of requested timestamps (Notification interval > 0) are acquired and if the notification has been enabled by the callof Icu_EnableNotification(). After a call of Icu_DisableNotification() this function must not be called.

An extern declaration of these functions is available in Icu_PBcfg.c. The functions shall be implemented by the user.

Chapter 7
Memory Allocation

7.1 Sections to be defined in MemMap.h

For Post Build data:

```
#ifdef ICU_START_CONFIG_DATA_UNSPECIFIED
    #undef ICU_START_CONFIG_DATA_UNSPECIFIED
    #undef MEMMAP_ERROR
    #pragma ghs section const=".pbicu_cfg"
#endif

#ifdef ICU_STOP_CONFIG_DATA_UNSPECIFIED
    #undef ICU_STOP_CONFIG_DATA_UNSPECIFIED
    #undef MEMMAP_ERROR
    #pragma ghs section
#endif
```

For Code:

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

For Variables:

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

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

For Constant data:

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

7.2 Linker command file

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

Chapter 8
CONFIGURATION PARAMETER CONSIDERATIONS

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

Table 8-1. Configuration Parameters

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
IcuGeneral			
	IcuDevErrorDetect	Pre Compile parameter for all Variants of Configuration	Pre Compile
	IcuIndex	Pre Compile parameter for all Variants of Configuration	Pre Compile
	IcuReportWakeupSource	Pre Compile parameter for all Variants of Configuration	Pre Compile
IcuOptionalApis			
	IcuDeInitApi	Pre Compile parameter for all Variants of Configuration	Pre Compile
	IcuDisableWakeupApi	Pre Compile parameter for all Variants of Configuration	Pre Compile
	IcuEdgeCountApi	Pre Compile parameter for all Variants of Configuration	Pre Compile
	IcuEnableWakeupApi	Pre Compile parameter for all Variants of Configuration	Pre Compile
	IcuGetDutyCycleValuesApi	Pre Compile parameter for all Variants of Configuration	Pre Compile
	IcuGetInputStateApi	Pre Compile parameter for all Variants of Configuration	Pre Compile
	IcuGetTimeElapsedApi	Pre Compile parameter for all Variants of Configuration	Pre Compile
	IcuGetVersionInfoApi	Pre Compile parameter for all Variants of Configuration	Pre Compile
	IcuSetModeApi	Pre Compile parameter for all Variants of Configuration	Pre Compile
	IcuTimestampApi	Pre Compile parameter for all Variants of Configuration	Pre Compile

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
	IcuSignalMeasurementApi	Pre Compile parameter for all Variants of Configuration	Pre Compile
	IcuOverflowNotificationApi	Pre Compile parameter for all Variants of Configuration	Pre Compile
IcuConfigSet			
	IcuMaxChannel	VariantPC or VariantPB	Post Build
	IcuChannelId	VariantPC or VariantPB	Post Build
IcuChannel			
	IcuHwChannel	VariantPC or VariantPB	Post Build
	IcuOverflowNotification	VariantPC or VariantPB	Post Build
	IcuEmiosFreeze	VariantPC or VariantPB	Post Build
	IcuEmiosPrescaler	VariantPC or VariantPB	Post Build
	IcuEmiosDigitalFilter	VariantPC or VariantPB	Post Build
	IcuEmiosBusSelect	VariantPC or VariantPB	Post Build
	IcuDefaultStartEdge	VariantPC or VariantPB	Post Build
	IcuMeasurementMode	VariantPC or VariantPB	Post Build
	IcuUserModeForDutycycle	VariantPC or VariantPB	Post Build
	IcuWakeupCapability	VariantPC or VariantPB	Post Build
	IcuEmiosPrescaler_Alternate	VariantPC or VariantPB	Post Build
IcuEdgeCounterMeasurement			
IcuSignalEdgeDetection			
	IcuSignalNotification	VariantPC or VariantPB	Post Build
IcuSignalMeasurement			
	IcuSignalMeasurementProperty	VariantPC or VariantPB	Post Build
cuTimestampMeasurement			
	IcuTimestampMeasurementProperty	VariantPC or VariantPB	Post Build
	IcuTimestampNotification	VariantPC or VariantPB	Post Build
IcuWakeup			
	IcuChannelWakeupInfo	VariantPC or VariantPB	Post Build
IcuNonAUTOSAR			
	IcuEnableDualClockMode	VariantPC or VariantPB	Pre Compile

Chapter 9
Integration steps

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

- Generate the required ICU configurations. For more details refer to section **Files required for Compilation**
- Allocate proper memory sections in MemMap.h and linker command file. For more details refer to section **Memory Allocation**.
- Make sure all include files for compilation are as per section **ISR macro**
- Map the ISRs to their vector locations. For more details refer to section peripheral_hw_requirements
- Compile & build the ICU with all the dependent modules. For more details refer to sections **Building the Driver** and ISR to configure withing OS

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
1-800-441-2447 or +1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Document Number: IM14ICUASR3.0R2.0.0
Rev. 2.0

freescale™