User Manual

for S32K3 ICU Driver

Document Number: UM34ICUASRR21-11 Rev0000R3.0.0 Rev. 1.0

Chapter 1

Revision History

Revision	Date	Author	Description
1.0	31.03.2023	NXP RTD Team	S32K3 Real-Time Drivers AUTOSAR 4.4 & R21-11 Version 3.0.0

Chapter 2

Introduction

This User Manual describes NXP Semiconductor AUTOSAR ICU for S32K3. AUTOSAR ICU driver configuration parameters and deviations from the specification are described in ICU Driver chapter of this document. AUTOSAR ICU driver requirements and APIs are described in the AUTOSAR ICU driver software specification document.

2.1 Supported Derivatives

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

- s32k310_mqfp100
- s32k310_lqfp48
- s32k311_mqfp100 / MWCT2015S_mqfp100
- s32k311_lqfp48
- s32k312_mqfp100 / MWCT2016S_mqfp100
- s32k312_mqfp172 / MWCT2016S_mqfp172
- s32k314_mqfp172
- s32k314_mapbga257
- s32k322_mqfp100 / MWCT2D16S_mqfp100
- s32k322_mqfp172 / MWCT2D16S_mqfp172

- s32k324_mqfp172 / MWCT2D17S_mqfp172
- s32k324_mapbga257
- s32k341_mqfp100
- s32k341_mqfp172
- s32k342_mqfp100
- s32k342_mqfp172
- s32k344_mqfp172
- s32k344_mapbga257
- s32k394_mapbga289
- s32k396_mapbga289
- s32k358_mqfp172
- s32k358_mapbga289
- s32k328_mqfp172
- s32k328_mapbga289
- s32k338_mqfp172
- s32k338_mapbga289
- s32k348_mqfp172
- s32k348_mapbga289
- s32m274_lqfp64
- s32m276_lqfp64

All of the above microcontroller devices are collectively named as S32K3.

Note: MWCT part numbers contain NXP confidential IP for Qi Wireless Power.

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** style: Used for important terms, notes and warnings.
- *Italic* style: 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.

Warning

This is a warning

2.4 Acronyms and Definitions

Term	Definition
API	Application Programming Interface
ASM	Assembler
BSMI	Basic Software Make file Interface
CAN	Controller Area Network
C/CPP	C and C++ Source Code
LPCMP	Low Power Comparator
CS	Chip Select
CTU	Cross Trigger Unit
DEM	Diagnostic Event Manager
DET	Development Error Tracer
DMA	Direct Memory Access
ECU	Electronic Control Unit
EMIOS	Enhanced Modular IO Subsystem
FIFO	First In First Out
FTM	Flextimer Module
ICU	Input Capture Unit
ISR	Interrupt Service Routine
LSB	Least Signifigant Bit
MCU	Micro Controller Unit
MIDE	Multi Integrated Development Environment
MSB	Most Significant Bit
N/A	Not Applicable
OS	Operating System
PB Variant	Post Build Variant
PC Variant	Pre Compile Variant
RAM	Random Access Memory
ROM	Read-only Memory
SIUL2	System Integration Unit Lite2
SWS	Software Specification
VLE	Variable Length Encoding
WKPU	Wakeup Unit
XML	Extensible Markup Language

2.5 Reference List

#	Title	Version
1	Specification of ICU Driver	AUTOSAR Release R21-11
2	Specification of Communication Stack Types	AUTOSAR Release R21-11
3	Specification of Compiler Abstraction	AUTOSAR Release R21-11
4	Specification of Platform Types	AUTOSAR Release R21-11

#	Title	Version
5	Specification of Standard Types	AUTOSAR Release R21-11
6	S32K3xx Reference Manual	Rev.6, Draft B, 01/2023
7	S32K39 and S32K37 Reference Manual	Rev. 2 Draft A, 11/2022
8	S32M27x Reference Manual	Rev.2, Draft A, 02/2023
9	S32K3xx Datasheet	Rev. 6, 11/2022
10	S32K396 Datasheet	Rev. 1.1 — 08/2022
11	S32M2xx Datasheet	Rev. 2 RC — 12/2022
11	S32K311 Errata	S32K311_0P98C Mask Set Errata, Rev. 6/March/2023, 3/2023
12	S32K312 Errata	Mask Set Errata for Mask 0P09C, Rev. 25/April/2022
13	S32K342 Errata	Mask Set Errata for Mask 0P97C, Rev. 10, 11/2022
14	S32K3x4 Errata	Mask Set Errata for Mask 0P55A/1P55A, Rev. 14/↩ Oct/2022
15	S32K358 Errata	S32K358_0P14E Mask Set Errata – Rev. 28, 9/2022
16	S32K396 Errata	S32K396_0P40E Mask Set Errata, Rev. DEC2022, 12/2022

Chapter 3

Driver

3.1 Requirements

Requirements for this driver are detailed in the AUTOSAR R21-11 ICU Driver Software Specification document (See Table Reference_list).

Requirements for this driver are detailed in the Autosar Driver Software Specification document (See Table Reference List).

It has vendor-specific requirements and implementation.

3.2 Driver Design Summary

The ICU Driver controls the input capture of the microcontroller. It provides the following features:

- High time / Low time measurement
- Duty Cycle measurement
- Period time measurement

- Edge detection and notification
- Edge counting (with or without hardware gating)
- Edge time stamping
- Wake-up interrupts

For signal edge detection, the edge detector of a capture compare unit or the interrupt controller for external events are used.

For signal measuring a capture timer and at least one capture register are needed. Also, only even channels (2∗n) can be used for signal measurements. This is because the channel after it (2∗n+1) is used internally by the ICU Driver

The EMIOS module of S32K3XX supports period time measurement, edge detection and notification, edge counting and edge time stamping.

The LPCMP, SIUL2 and WKPU module of S32K3XX supports edge detection with notification.

The ICU driver provides an optional API and configuration parameters for changing the base clock of the controlled hardware. A dual clock functionality is offered by switching between two configured values of the clock prescaler.

For each user configured channel, a symbolic name is generated by the Tresos Studio configuration tool. The name shall be consequently used in upper applications.

By default all channels offer interrupt handlers. For each channel not configured by the user in Tresos Studio configuration tool, the code for interrupt handling is removed based on a series of #ifdefs.

The RTD driver assures reentrancy (single core execution) for the APIs based on the following assumptions:

- the "called-again" API is for a different resource (hardware/logic channel);
- common variables/registers accessed with "rmw" are guarded by Exclusive Areas which need to be correctly implemented in RTE on user side;

3.3 Hardware Resources

The hardware resources configured by the Icu driver are **LPCMP**, **SIUL2**, **EMIOS** and **WKPU**.

The LPCMP, SIUL2, EMIOS and WKPU input signal to micro-controller pin mapping can be done by using "IO↩ _Signal_Description_and_Input_multiplexing_tables.xls" from the Reference manual.

Note: WKPU has the first 4 channels, which are the internal wakeup sources channel. Next 60 channels support external wakeup sources, corresponding to 60 GPIO pins (WKPU[0]-WKPU[59] pins). So if you configure pin GPIO WKPU[n], you have to configure corresponding WKPU_CH_(n + 1) in ICU. Ex: Pin WKPU[3] corresponds to WKPU_CH_7.

3.4 Deviations from Requirements

The driver deviates from the AUTOSAR ICU Driver software specification in some places. The table below identifies the AUTOSAR requirements that are not implemented or out of scope for the ICU Driver.

Term	Definition
N/S	Out of scope
N/I	Not implemented
N/F	Not fully implemented

Below table identifies the AUTOSAR requirements that are not fully implemented, implemented differently or out of scope for the ICU driver.

Requirement	Status	Description	Notes
SWS_Icu_00150	N/S	The Icu module shall not check the integrity if several calls for the same ICU channel are used during runtime in different tasks or ISRs.	Rejection Reason: The requirement is violating safety because: The ICU149 is a safety integrity assumption for external environment, which shall be implemented for FTE; For GTE and NTE ICU149 has a role to increase availability because the check will be supported by ICU driver; see also 00149
SWS_Icu_00380	N/S	These requirements are not applicable to this specification.	Not a requirement
SWS_Icu_91002	N/S	This function disables the notification of a channel.Tags: atp.↩ Status=draft	Description specified as draft is not clear. Should be re-assessed on next ASR version
SWS_Icu_91003	N/S	This function enables the notification on the given channel.↩ Tags: atp.Status=draft	Description specified as draft is not clear. Should be re-assessed on next ASR version
SWS_Icu_CONSTR_00003	N/S	If IcuEcucPartitionRef references one or more ECUC partitions, IcuKernelEcuc↩ PartitionRef shall have a multiplicity of one and reference one of these ECUC partitions as well.	Type IV Autosar multicore not implemented for current module (AAI-445), therefore Icu↩ KernelEcucPartitionRef is not supported.

3.5 Driver limitations

- Function Icu_SetClockMode/Emios_Icu_Ip_SetClockMode does not effect to channel which using master bus counter.
- To reduce interrupt complexity and unify IPVs, only one source address and one destination address are used to transmit and store the counter value in DMA mode. so this will be done by enabling eMios SAIC channels profile only when using DMA mode.

3.6 Driver usage and configuration tips

In this chapter, the extra features from our drivers that are not described in the AutoSAR standard are detailed

3.6.1 Icu with DMA feature

Tips for this feature will be added in the next release.

3.6.2 Dual Clock Feature

In order to allow dynamic change of the driver working frequency, the ICU driver has the Dual Clock Feature. The IcuEnableDualClockMode from IcuAutosarExt should be enabled in order to have this feature active. Afterwards, the Prescaler_Alternate parameter allows setting a different pre-scaler for each module. These parameters will be changed when calling the function call Icu_SetClockMode.
Icu_SetClockMode may be called only after Icu_Init is called and when IcuEnableDualClockMode is checked. Our suggested usage of this API is to call it when the driver is in a lower power state but still in active use.

3.6.3 WKPU channel selection

From WKPU peripheral perspective the input channels are counted from 0 to 63 (64 hardware channels are available)

```
0 to 3 are internally routed
4 to 63 are routed to external pins that are named as: WKPU_0 to WKPU_59.
```

From WKPU perspective, the driver have to configure also channels from 0 to 3 and trigger on internal events.

In below picture please see an example of channels allocation as:

- wkpu channel 1 receive input from RTC
- wkpu channel 2 receive input from CMP units
- wkpu channel 3 receive input from RTI
- wkpu channel 59 receive input from external pin WKPU_55

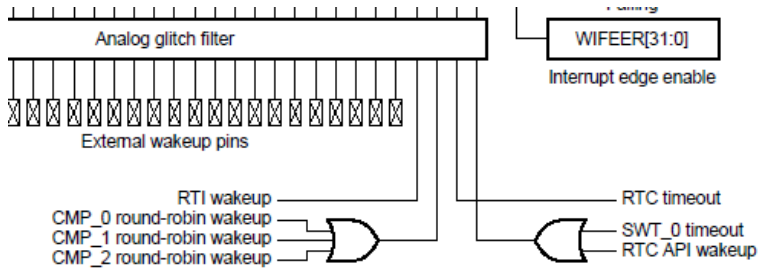

Figure 3.1 Wkpu External and Internal pins diagram

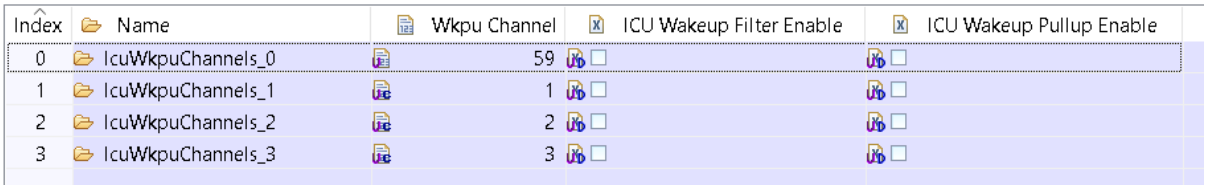

Figure 3.2 Wkpu Peripheral channel selection

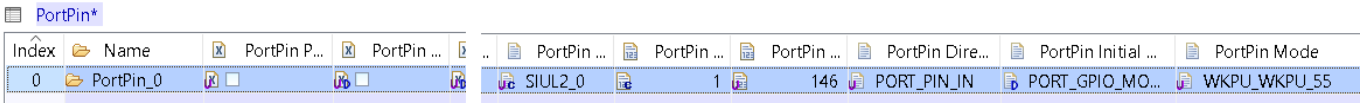

Figure 3.3 External Pin configuration in Port

3.6.4 Configuration when using the EMIOS master bus

The Mcl module allows to enable/disable a EMIOS channel by stopping its clock. Each field controls the clock for the corresponding channel.

- With EB Tresos high layer configuration:

MCL driver should be configured in 3 steps:

step 1: enable Emios common support

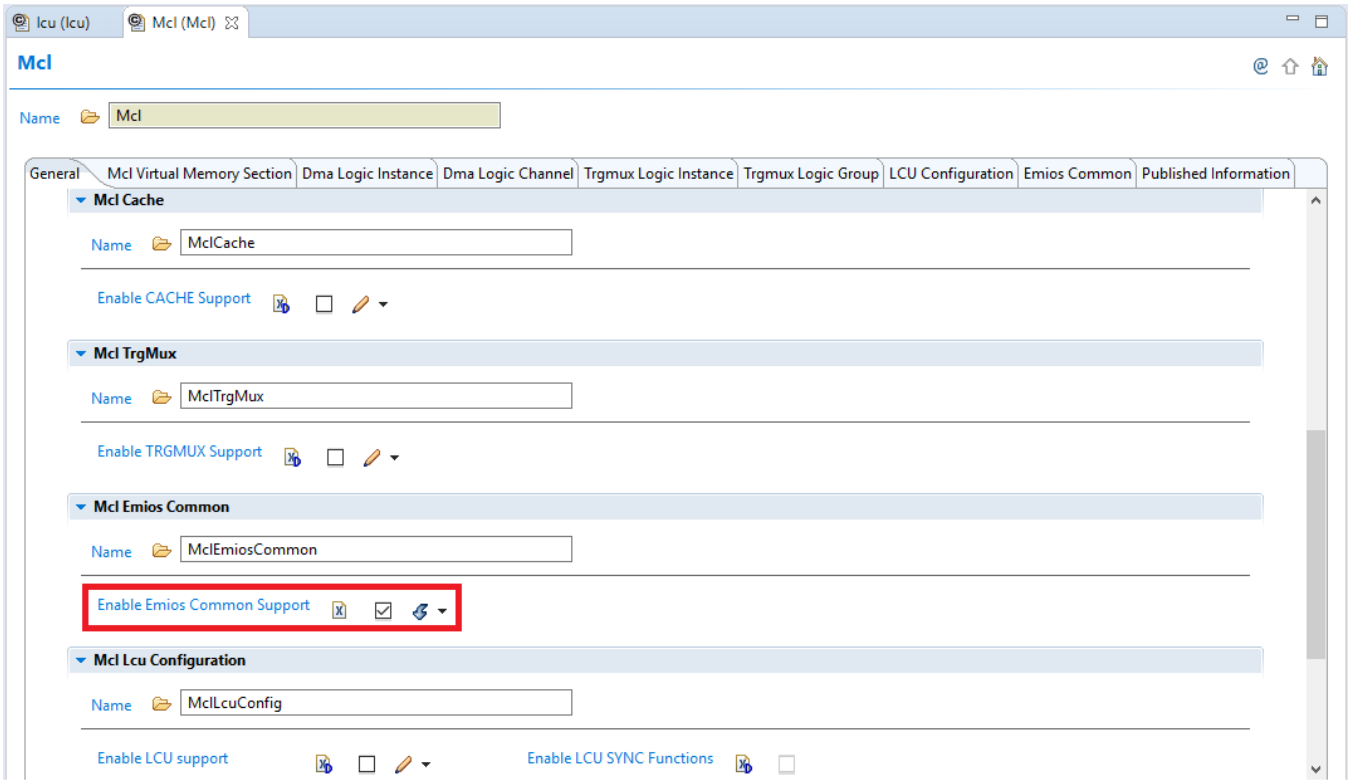

Figure 3.4 Enable Emios common support.

step 2: Configure instance when Emios Common Support is checked

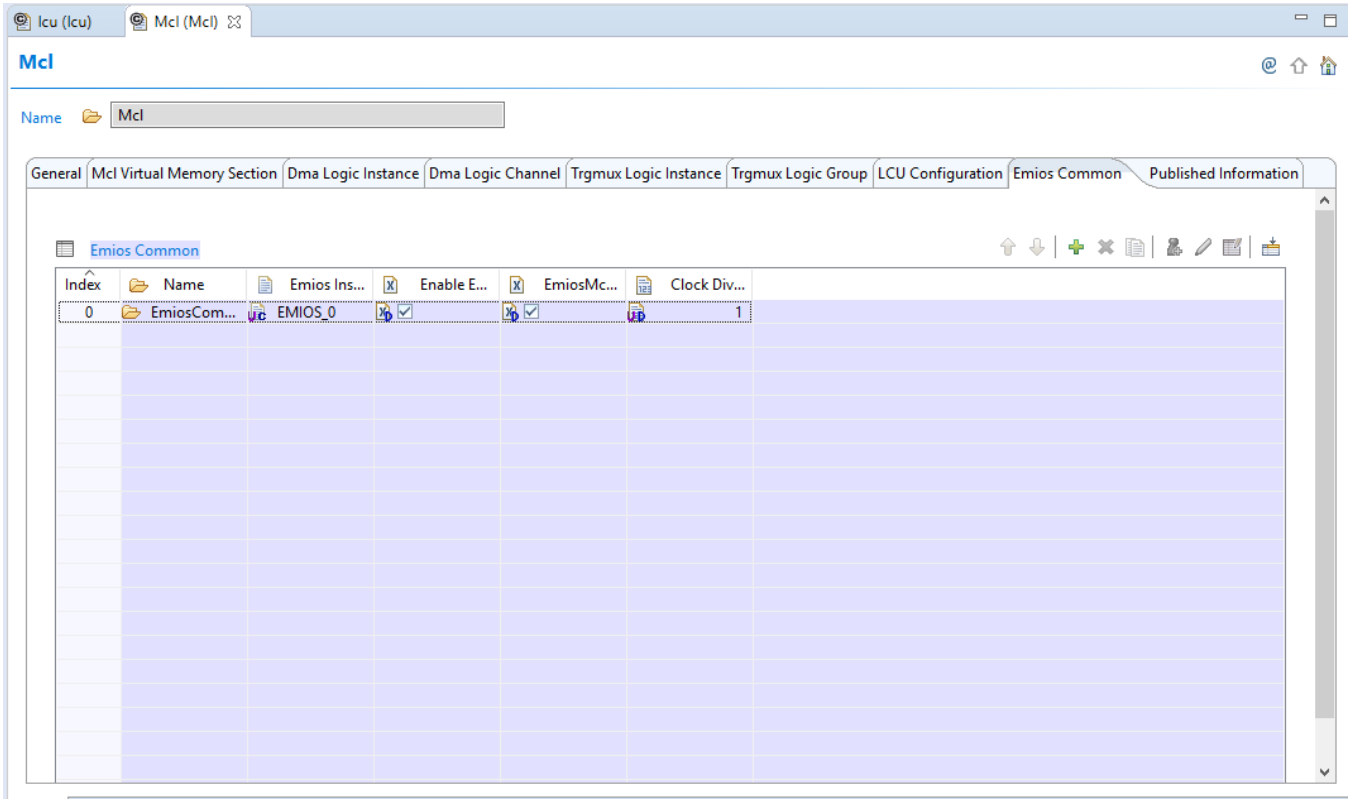

Figure 3.5 Configure instance.

step 3: Configure Emios master bus

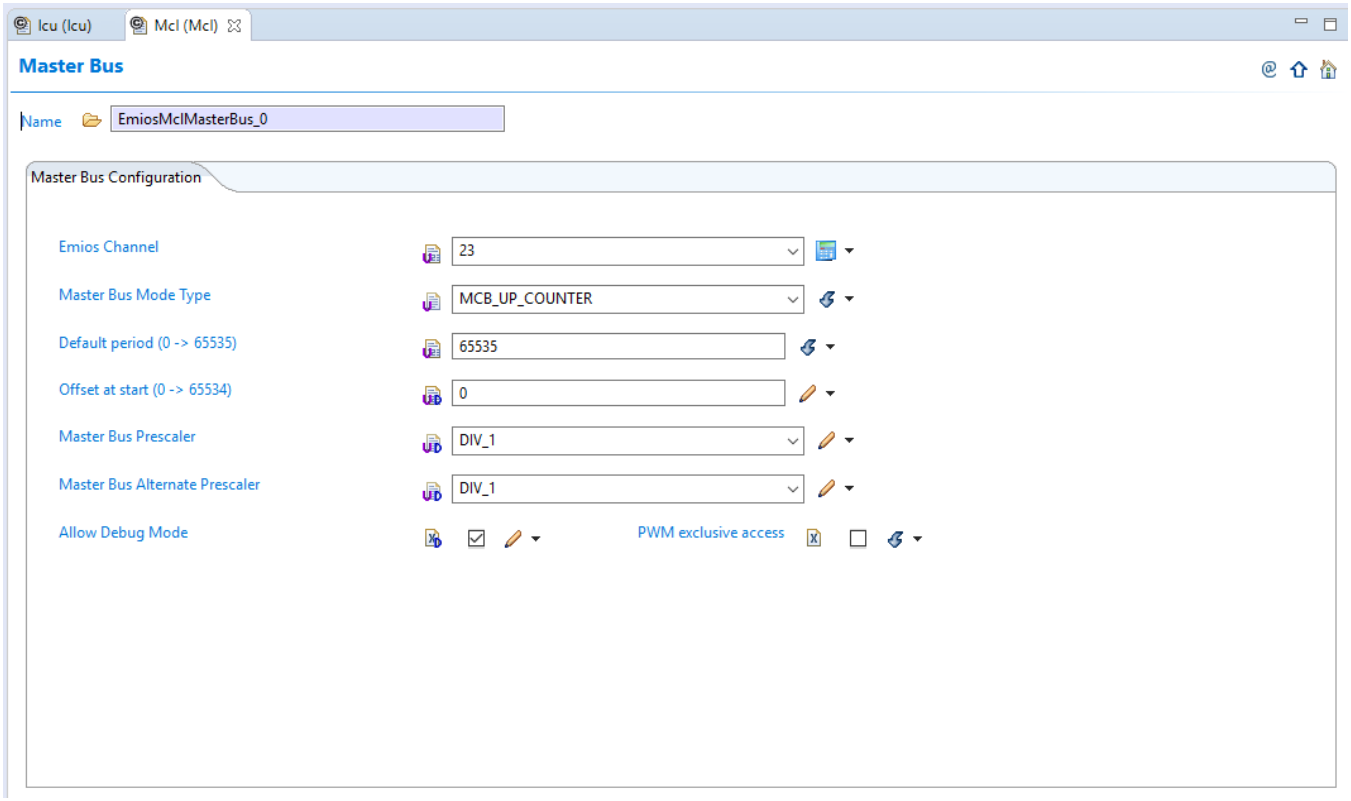

Figure 3.6 Configure Emios master bus.

- With S32DS high layer configuration:
MCL driver should be configured in 2 steps:
step 1: enable Emios common support

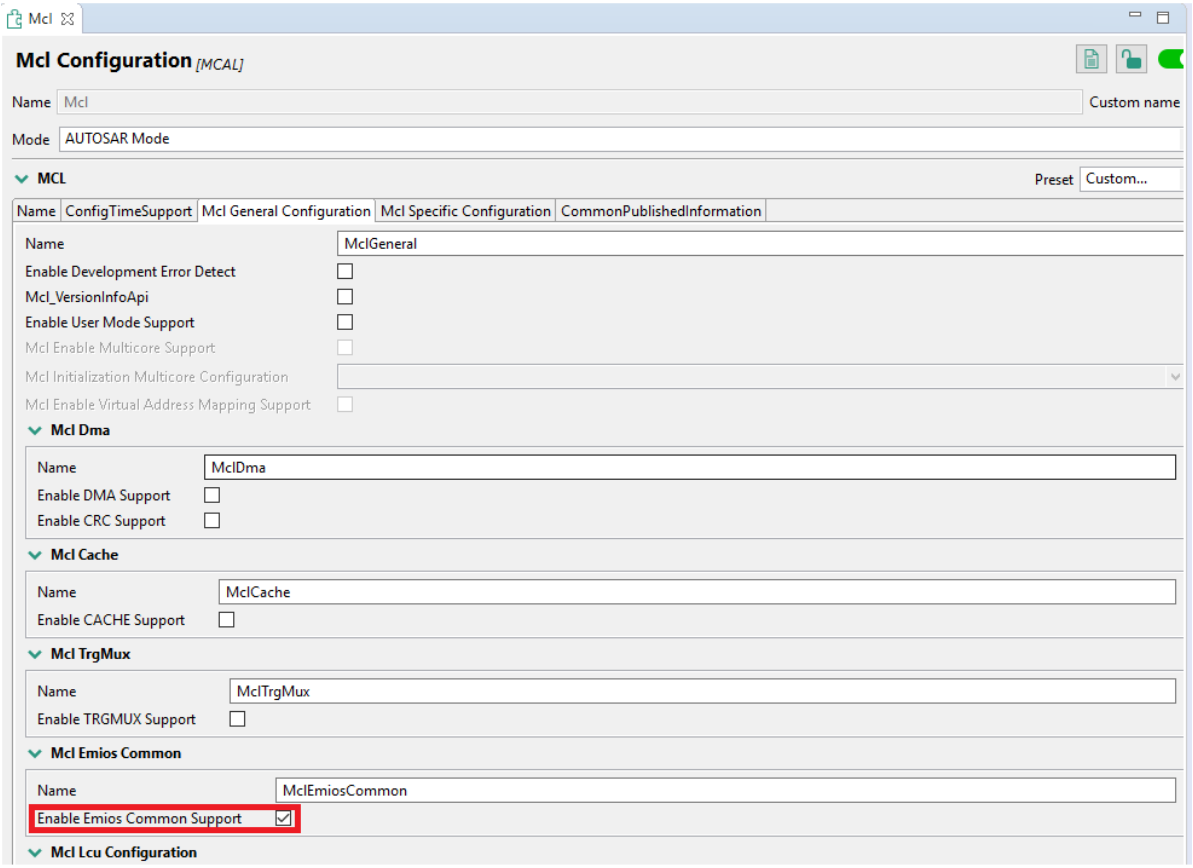

Figure 3.7 Enable Emios common support.

step 2: Configure Emios master bus

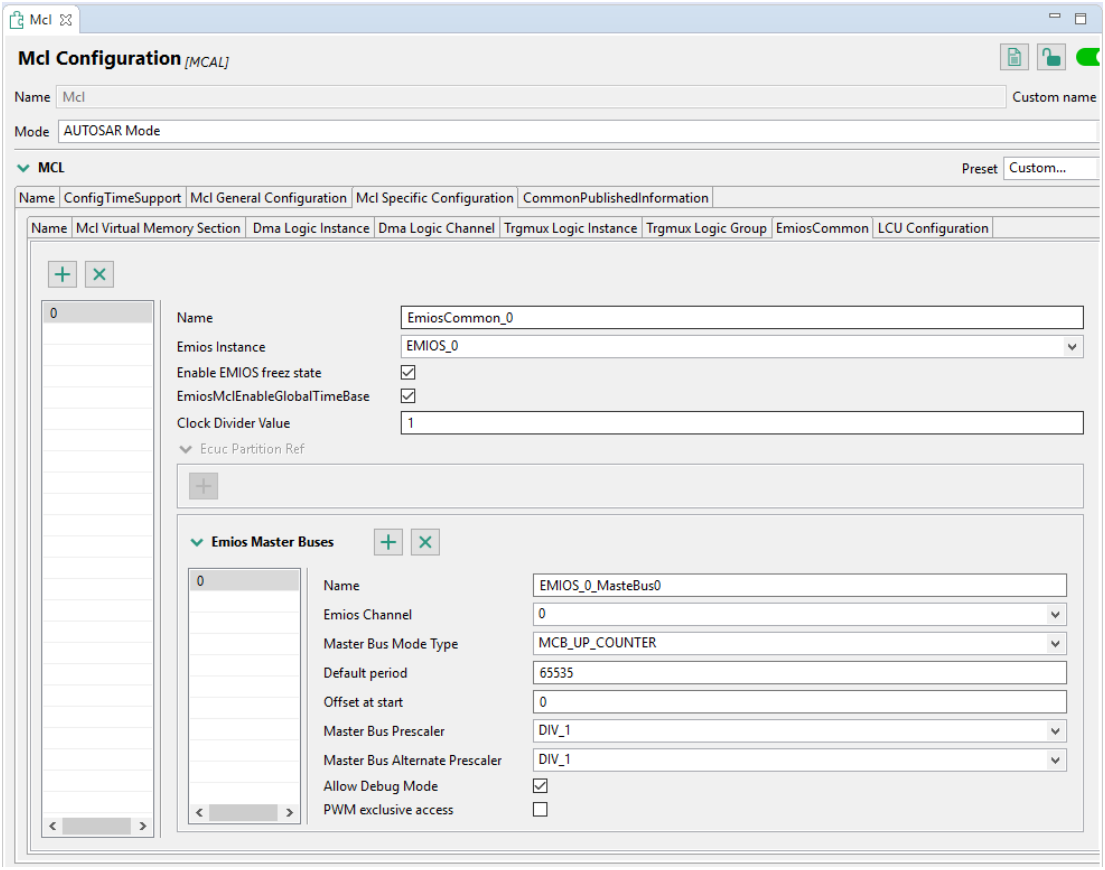

Figure 3.8 Configure Emios master bus.

- With IP layer configuration:
Emios_Mcl_Ip driver should be configured in 2 steps:
step 1: Enable Emios common support

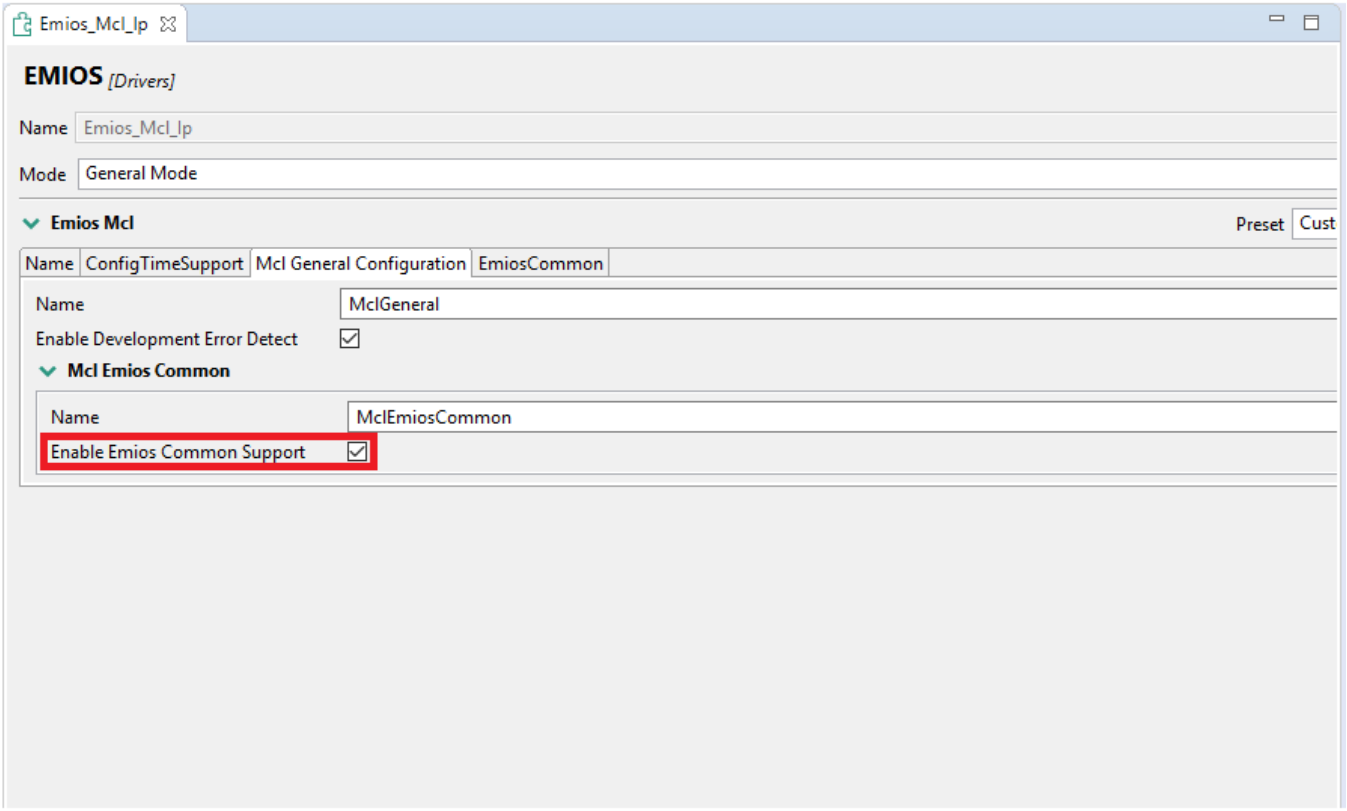

Figure 3.9 Enable Emios common support.

step 2: Configure Emios master bus

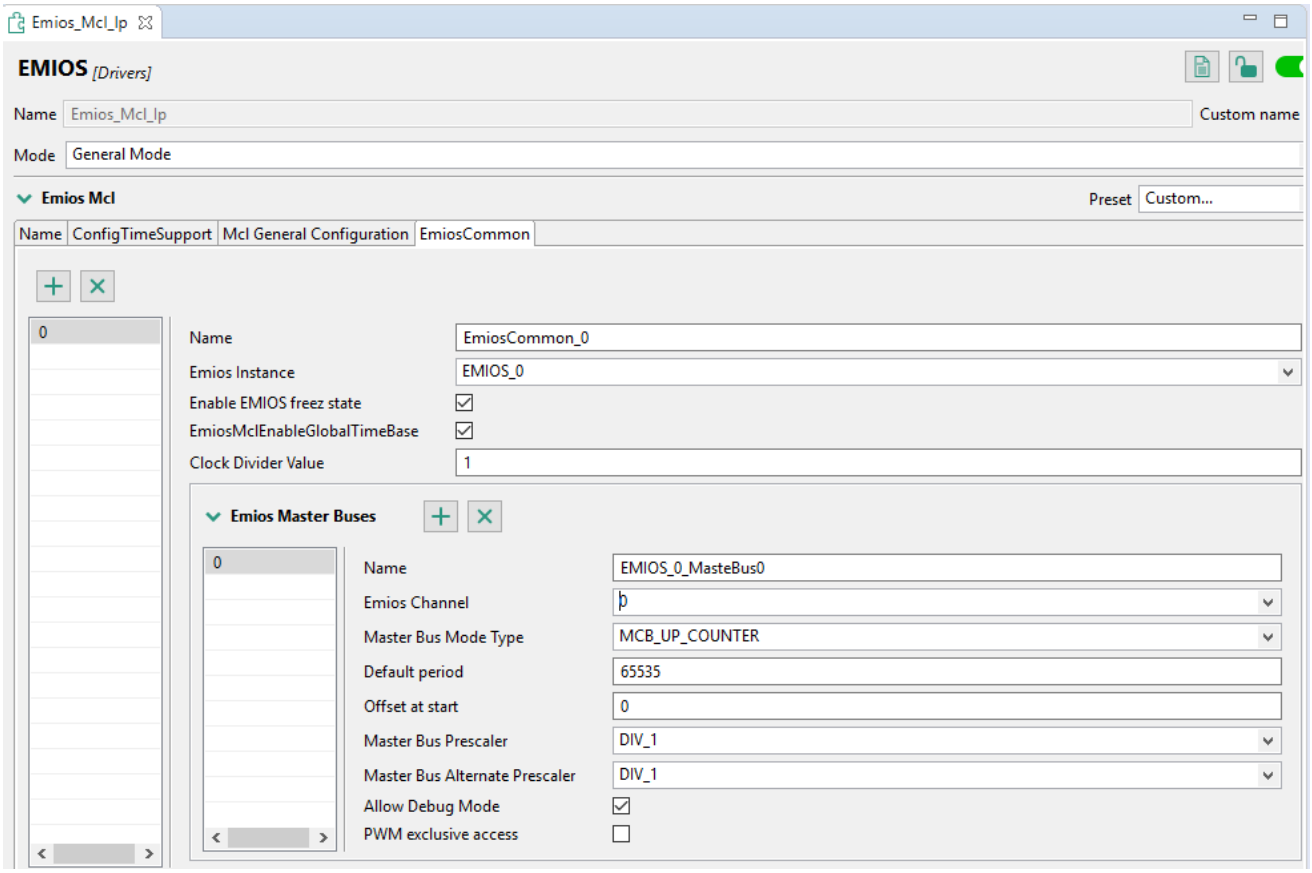

Figure 3.10 Configure Emios master bus.

3.6.5 How to use to change the initial value and max value of counter in ICU_MODE_EDGE_COUNTER mode

Can change initial value and max value of eMios counter in edge counter mode. You can use each function or use both. the functions only support ICU_MODE_EDGE_COUNTER mode.

- With EB Tresos high layer configuration:
Configured and call functions in 2 steps:
step 1: Enable/Disable the service set Initial and Max Counter for eMios.

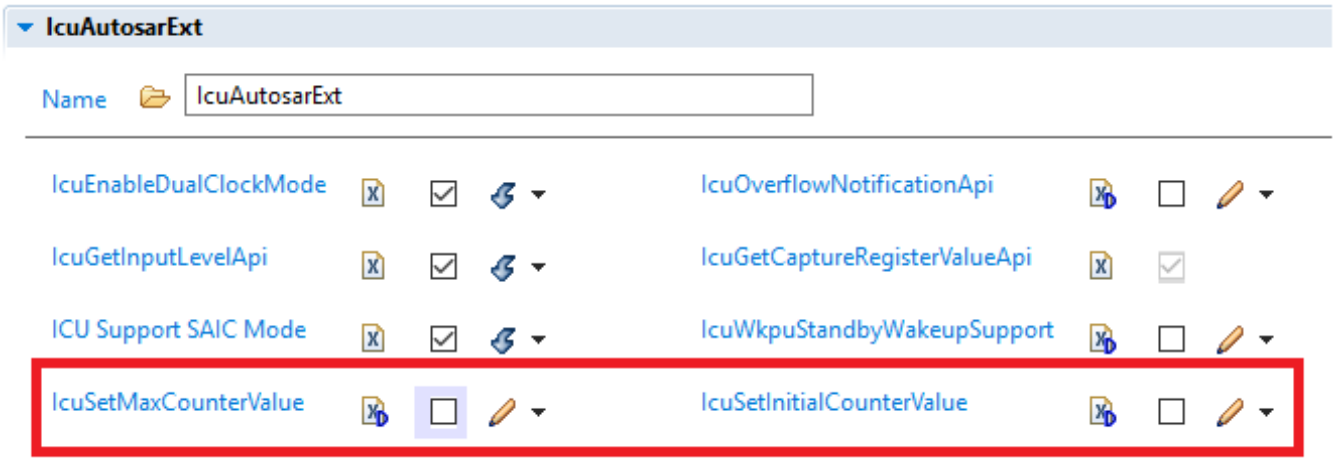

Figure 3.11 Enable/Disable the service.

step 2: Call to functions Call Icu_SetInitialCounterValue(Channel, InitialCounterValue) to change Initial Counter value and call Icu_SetMaxCounterValue(Channel, MaxCounterValue) to change Max Counter value. These functions must be called before Icu_EnableEdgeCount function."

- With S32DS high layer configuration:
Configured and call functions in 2 steps:
step 1: Enable/Disable the service set Initial and Max Counter for eMios.

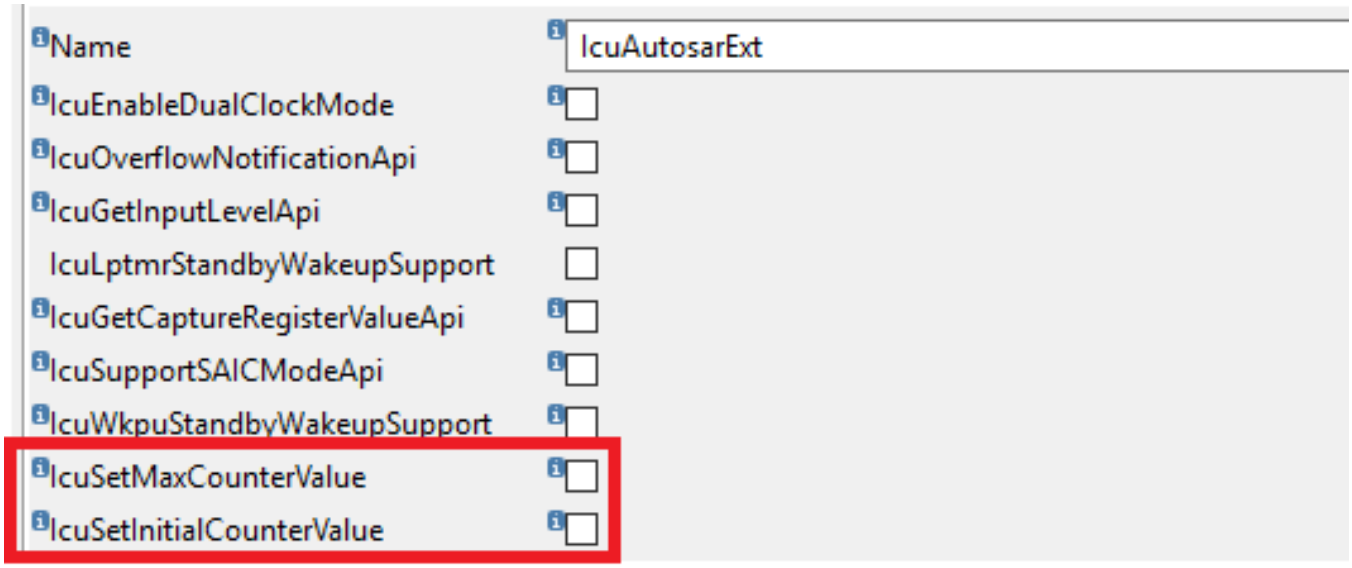

Figure 3.12 Enable/Disable the service.

step 2: Call to functions Call Icu_SetInitialCounterValue(Channel, InitialCounterValue) to change Initial Counter

value and call Icu_SetMaxCounterValue(Channel, MaxCounterValue) to change Max Counter value. These functions must be called before Icu_EnableEdgeCount function."

- With IP layer configuration:
Configured and call functions in 2 steps:
step 1: Enable/Disable the service set Initial and Max Counter for eMios. In eMios IPL you can use these functions without configuration.
step 2: Call to functions Call Emios_Icu_Ip_SetInitialCounterValue(instance, hwChannel, initialCounter) to change Initial Counter value and call Emios_Icu_Ip_SetMaxCounterValue(instance, hwChannel, maxCounter) to change Max Counter value. These functions must be called before Emios_Icu_Ip_EnableEdgeCount function."

3.7 Runtime Errors

The driver does not generate any DEM runtime errors.

3.8 Symbolic Names Disclaimer

All containers having symbolicNameValue set to TRUE in the AUTOSAR schema will generate defines like:

#define <Mip>Conf_<Container_ShortName>_<Container_ID>

For this reason it is forbidden to duplicate the names of such containers across the RTD configurations or to use names that may trigger other compile issues (e.g. match existing `#ifdefs` arguments).

Chapter 4

Tresos Configuration Plug-in

This chapter describes the Tresos configuration plug-in for the driver. All the parameters are described below.

- Module Icu
  - Container IcuConfigSet
    - Parameter IcuMaxChannel
    - Container IcuChannel
      - Parameter IcuChannelId
      - Parameter IcuDMAChannelEnable
      - Parameter IcuDefaultStartEdge
      - Parameter IcuMeasurementMode
      - Parameter IcuOverflowNotification
      - Parameter IcuWakeupCapability
      - Reference IcuChannelEcucPartitionRef
      - Reference IcuChannelRef
      - Reference IcuDMAChannelRef
      - Container IcuSignalEdgeDetection
      - Parameter IcuSignalNotification
      - Container IcuSignalMeasurement
      - Parameter IcuSignalMeasurementProperty
      - Container IcuTimestampMeasurement
      - Parameter IcuTimestampMeasurementProperty
      - Parameter IcuTimestampNotification
      - Container IcuWakeup
      - Reference IcuChannelWakeupInfo
    - Container IcueMios
      - Parameter IcueMiosModule
      - Container IcueMiosChannels
      - Parameter IcueMiosChannel
      - Parameter IcuEmiosFreeze
      - Parameter IcuEmiosPrescaler
      - Parameter IcuEmiosPrescaler_Alternate
      - Parameter IcuEmiosDigitalFilter

- Parameter IcuEmiosBusSelect
- Parameter IcuSubModeforMeasurement
- Parameter IcuSignalMeasureWithoutInterrupt
- Reference IcuEmiosBusRef

* Container IcuSiul2
  - Parameter IcuSiul2Instance
  - Parameter IcuEXT_ISR_InterruptFilterClockPrescaler
  - Parameter IcuEXT_ISR_AlternateInterruptFilterClockPrescaler
  - Container IcuSiul2Channels
  - Parameter IcuSiul2Channel
  - Parameter Icu_EXT_ISR_IFERDigitalFilter
  - Parameter Icu_EXT_ISR_IFMCDigitalFilter
* Container IcuWkpu
  - Container IcuWkpuChannels
  - Parameter IcuWkpuChannel
  - Parameter Icu_EXT_ISR_WIFERDigitalFilter
  - Parameter IcuWKPU_ISR_WIPUER
  - Container IcuWkpuNMIConfiguration
  - Parameter NMICoreSource
  - Parameter DestinationSourceSelect
  - Parameter WakeupRequestEnable
  - Parameter FilterEnable
  - Parameter NMIEdgeEvents
  - Parameter LockRegister
* Container IcuLpCmp
  - Parameter IcuCmpInstanceNumber
  - Container IcuCmp
  - Parameter IcuCmpFunctionalMode
  - Parameter IcuCmpHysteresisLevel
  - Parameter IcuCmpOffsetLevel
  - Parameter IcuCmpEnablePinOutput
  - Parameter IcuCmpEnableInverter
  - Parameter IcuCmpEnableComparatorInvert
  - Parameter IcuCmpEnableHighPowerMode
  - Parameter IcuCmpFilterSamplePeriod
  - Parameter IcuCmpFilterSampleCount
  - Parameter IcuCmpEnableDma
  - Parameter IcuCmpNegativeInputMux
  - Parameter IcuCmpPositiveInputMux
  - Parameter IcuCmpOutputSelect
  - Parameter IcuCmpWindowCloseOutputOverwrite
  - Parameter IcuCmpWindowCloseEvent
  - Parameter IcuCmpEnableInStop
  - Container IcuDac
  - Parameter IcuDacVoltageLevel
  - Parameter IcuDacVoltageRefSource
  - Parameter IcuDacPowerState

        - Container IcuTrigger
        - Parameter IcuTrgRoundRobinEnChannelMask
        - Parameter IcuTrgPrepgmStateChannelMask
        - Parameter IcuTrgInitDelayValue
        - Parameter IcuTrgSampleDelay
        - Parameter IcuTrgFixedChannel
        - Parameter IcuTrgFixedPort
        - Parameter IcuTrgEnableRoundRobinInterrupt
        - Parameter IcuTrgEnableRoundRobin
      * Container IcuHwInterruptConfigList
        - Parameter IcuIsrHwId
        - Parameter IcuIsrEnable
  - Container IcuGeneral
    * Parameter IcuDevErrorDetect
    * Parameter IcuReportWakeupSource
    * Parameter IcuEnableUserModeSupport
    * Parameter IcuMulticoreSupport
    * Reference IcuEcucPartitionRef
    * Reference IcuKernelEcucPartitionRef
  - Container IcuAutosarExt
    * Parameter IcuEnableDualClockMode
    * Parameter IcuOverflowNotificationApi
    * Parameter IcuGetInputLevelApi
    * Parameter IcuGetCaptureRegisterValueApi
    * Parameter IcuSupportSAICModeApi
    * Parameter IcuWkpuStandbyWakeupSupport
    * Parameter IcuSetMaxCounterValue
    * Parameter IcuSetInitialCounterValue
  - Container IcuOptionalApis
    * Parameter IcuDeInitApi
    * Parameter IcuDisableWakeupApi
    * Parameter IcuEdgeCountApi
    * Parameter IcuEnableWakeupApi
    * Parameter IcuGetDutyCycleValuesApi
    * Parameter IcuGetInputStateApi
    * Parameter IcuGetTimeElapsedApi
    * Parameter IcuGetVersionInfoApi
    * Parameter IcuSetModeApi
    * Parameter IcuSignalMeasurementApi
    * Parameter IcuTimestampApi
    * Parameter IcuWakeupFunctionalityApi
    * Parameter IcuEdgeDetectApi

- Container CommonPublishedInformation
  - Parameter ArReleaseMajorVersion
  - Parameter ArReleaseMinorVersion
  - Parameter ArReleaseRevisionVersion
  - Parameter ModuleId
  - Parameter SwMajorVersion
  - Parameter SwMinorVersion
  - Parameter SwPatchVersion
  - Parameter VendorApiInfix
  - Parameter VendorId

4.1 Module Icu

Configuration of the Icu (Input Capture Unit) module

Included containers:

- IcuConfigSet
- IcuGeneral
- IcuAutosarExt
- IcuOptionalApis
- CommonPublishedInformation

Property	Value
type	ECUC-MODULE-DEF
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantSupport	true
supportedConfigVariants	VARIANT-POST-BUILD, VARIANT-PRE-COMPILE

4.2 Container IcuConfigSet

This container is the base for a multiple configuration set

Included subcontainers:

- IcuChannel

- IcueMios
- IcuSiul2
- IcuWkpu
- IcuLpCmp
- IcuHwInterruptConfigList

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.3 Parameter IcuMaxChannel

The value for the IcuMaxChannel must match with the number of IcuChannel configured

For calculating the correct value use the CALC button.

Note: Total number of configured channels should be same across all IcuConfigSets.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	3
max	167
min	1

4.4 Container IcuChannel

Configuration of an individual ICU channel.

Included subcontainers:

- IcuSignalEdgeDetection
- IcuSignalMeasurement
- IcuTimestampMeasurement
- IcuWakeup

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	Infinite
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE

4.5 Parameter IcuChannelId

Channel Id of the ICU channel.

This value will be assigned to the symbolic name derived of the IcuChannel container short name.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	true
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	1
max	166
min	0

4.6 Parameter IcuDMAChannelEnable

IcuDMAChannelEnable indicates if the corresponding channel will use DMA for measurement

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.7 Parameter IcuDefaultStartEdge

Configures the default-activation-edge which shall be used for this channel

if there was no activation-edge configured by the call of service Icu_SetActivationCondition().

In case the Measurement Mode is "IcuSignalMeasurement" and the properties "DutyCycle" or "Period" are set, the edge configured here is used as Default Period Start Edge.

Implementation Type: Icu_ActivationType

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	ICU_RISING_EDGE
literals	['ICU_BOTH_EDGES', 'ICU_FALLING_EDGE', 'ICU_RISING_EDGE']

4.8 Parameter IcuMeasurementMode

Configures the measurement mode of this channel.

User should enable optional parameters with respect to the selected IcuMeasurementMode.

Implementation Type: Icu_MeasurementModeType

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	ICU_MODE_SIGNAL_EDGE_DETECT
literals	['ICU_MODE_EDGE_COUNTER', 'ICU_MODE_SIGNAL_EDGE_DETE↩ CT', 'ICU_MODE_SIGNAL_MEASUREMENT', 'ICU_MODE_TIMESTA↩ MP']

4.9 Parameter IcuOverflowNotification

Icu Overflow Notification Handler

In order to activate this field you have to:

enable IcuOverflowNotificationApi,

choose one of the modes:

ICU_MODE_EDGE_COUNTER,

ICU_MODE_SIGNAL_MEASUREMENT,

ICU_MODE_TIMESTAMP

to enable overflow detection on the internal counter

Note:

Due to hardware implementattion, the Icu Overflow Notification

is not syncronous with the event for ICU_MODE_SIGNAL_MEASUREMENT

and ICU_MODE_TIMESTAMP modes.

The notification will be triggered when measurement completes (for ICU_MODE_SIGNAL_MEASUREMENT)

or the next timestamp event occurs (for ICU_MODE_TIMESTAMP).

Property	Value
type	ECUC-FUNCTION-NAME-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	NULL_PTR

4.10 Parameter IcuWakeupCapability

Information about the wakeup-capability of this channel.

true: Channel is wakeup capable.

false: Channel is not wakeup capable.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.11 Reference IcuChannelEcucPartitionRef

Maps a ICU channel to zero or multiple ECUC partitions to limit the access to this channel group.

The ECUC partitions referenced are a subset of the ECUC partitions where the ICU driver is mapped to.

Property	Value
type	ECUC-REFERENCE-DEF
origin	AUTOSAR_ECUC
lowerMultiplicity	0
upperMultiplicity	Infinite
postBuildVariantMultiplicity	true
multiplicityConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
requiresSymbolicNameValue	False
destination	/AUTOSAR/EcucDefs/EcuC/EcucPartitionCollection/EcucPartition

4.12 Reference IcuChannelRef

Select the ICU hw channel on which the functionality of the current ICU channel will be implemented

Property	Value
type	ECUC-CHOICE-REFERENCE-DEF
origin	NXP
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
requiresSymbolicNameValue	False
destinations	['/TS_T40D34M30I0R0/Icu/IcuConfigSet/IcueMios/IcueMiosChannels', '/↩ TS_T40D34M30I0R0/Icu/IcuConfigSet/IcuWkpu/IcuWkpuChannels', '/T↩ S_T40D34M30I0R0/Icu/IcuConfigSet/IcuWkpu/IcuWkpuNMIConfiguration', '/TS_T40D34M30I0R0/Icu/IcuConfigSet/IcuSiul2/IcuSiul2Channels', '/TS_↩ T40D34M30I0R0/Icu/IcuConfigSet/IcuLpCmp']

4.13 Reference IcuDMAChannelRef

Icu DMA Channel Reference

Reference to the DMA Channel configure for the Request

Property	Value
type	ECUC-CHOICE-REFERENCE-DEF
origin	NXP
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
requiresSymbolicNameValue	False
destinations	['/TS_T40D34M30I0R0/Mcl/MclConfig/dmaLogicChannel_Type']

4.14 Container IcuSignalEdgeDetection

This container contains the configuration (parameters) in case the measurement mode is "IcuSignalEdgeDetection"

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	true
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD

4.15 Parameter IcuSignalNotification

Notification function for signal notification

Property	Value
type	ECUC-FUNCTION-NAME-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	0
upperMultiplicity	1

Property	Value
postBuildVariantMultiplicity	true
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	NULL_PTR

4.16 Container IcuSignalMeasurement

This container contains the configuration (parameters) in case the measurement mode is "IcuSignalMeasurement"

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	true
multiplicityConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE

4.17 Parameter IcuSignalMeasurementProperty

Configures the property that could be measured in case the mode is "IcuSignalMeasurement".

This property can not be changed during runtime.

Followings are measurement mode

ICU_DUTY_CYCLE

ICU_HIGH_TIME

ICU_LOW_TIME

ICU_PERIOD_TIME

Implementation Type: Icu_SignalMeasurementPropertyType

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	ICU_DUTY_CYCLE
literals	['ICU_DUTY_CYCLE', 'ICU_HIGH_TIME', 'ICU_LOW_TIME', 'ICU_P↩ ERIOD_TIME']

4.18 Container IcuTimestampMeasurement

This container contains the configuration (parameters) in case the measurement mode is "IcuTimestamp"

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	true
multiplicityConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE

4.19 Parameter IcuTimestampMeasurementProperty

Configures the handling of the buffer in case the mode is "Timestamp"

Following type of buffer inplemented in current implementation.

ICU_CIRCULAR_BUFFER.

ICU_LINEAR_BUFFER.

Implementation Type: Icu_TimestampBufferType

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	ICU_LINEAR_BUFFER
literals	['ICU_CIRCULAR_BUFFER', 'ICU_LINEAR_BUFFER']

4.20 Parameter IcuTimestampNotification

Notification function if the number of requested timestamps(Notification interval > 0) are acquired

Property	Value
type	ECUC-FUNCTION-NAME-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	true
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	NULL_PTR

4.21 Container IcuWakeup

This container contains the configuration (parameters) needed to configure a wakeup capable channel

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE

4.22 Reference IcuChannelWakeupInfo

If the wakeup-capability is true the wakeup source referenced is transmitted to the ECU State Manager (EcuM) .

Implementation Type: reference to EcuM_WakeupSourceType

Property	Value
type	ECUC-REFERENCE-DEF
origin	AUTOSAR_ECUC
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	true
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
requiresSymbolicNameValue	False
destination	/AUTOSAR/EcucDefs/EcuM/EcuMConfiguration/EcuMCommon↩ Configuration/EcuMWakeupSource

4.23 Container IcueMios

Configuration of a eMios module available on the platfom.

Included subcontainers:

- IcueMiosChannels

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	0
upperMultiplicity	Infinite
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE

4.24 Parameter IcueMiosModule

Select the physical eMios Module.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	0
max	2
min	0

4.25 Container IcueMiosChannels

List of eMios channels available on the platform.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	Infinite
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE

4.26 Parameter IcueMiosChannel

Selects one of the eMios channels available on the platform.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	0
max	23
min	0

4.27 Parameter IcuEmiosFreeze

If selected eMIOS channel registers are freezed in debug mode.

NoteThis is an Implementation Specific Parameter.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.28 Parameter IcuEmiosPrescaler

If an eMIOS channel is being used,

this parameter configures the clock divider value

for the internal prescaler of specific Unified Channel.

Prescaled clock used as clock source for the programmable input filter.

NoteThis is an Implementation Specific Parameter.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	EMIOS_PRESCALER_DIVIDE_1
literals	['EMIOS_PRESCALER_DIVIDE_1', 'EMIOS_PRESCALER_DIVIDE_2', 'EMIOS_PRESCALER_DIVIDE_3', 'EMIOS_PRESCALER_DIVIDE_4', 'EMIOS_PRESCALER_DIVIDE_5', 'EMIOS_PRESCALER_DIVIDE_6', 'EMIOS_PRESCALER_DIVIDE_7', 'EMIOS_PRESCALER_DIVIDE_8', 'EMIOS_PRESCALER_DIVIDE_9', 'EMIOS_PRESCALER_DIVIDE_10', 'EMIOS_PRESCALER_DIVIDE_11', 'EMIOS_PRESCALER_DIVIDE_12', 'EMIOS_PRESCALER_DIVIDE_13', 'EMIOS_PRESCALER_DIVIDE_14', 'EMIOS_PRESCALER_DIVIDE_15', 'EMIOS_PRESCALER_DIVIDE_16']

4.29 Parameter IcuEmiosPrescaler_Alternate

If an eMIOS channel is being used,

prescaler — this parameter configures the alternate clock divider value for the internal prescaler of specific Unified Channel. Prescaled clock used as clock source for the programmable input filter. Note

This is an Implementation Specific Parameter.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD

Property	Value
defaultValue	EMIOS_PRESCALER_DIVIDE_1
literals	['EMIOS_PRESCALER_DIVIDE_1', 'EMIOS_PRESCALER_DIVIDE_2', 'EMIOS_PRESCALER_DIVIDE_3', 'EMIOS_PRESCALER_DIVIDE_4', 'EMIOS_PRESCALER_DIVIDE_5', 'EMIOS_PRESCALER_DIVIDE_6', 'EMIOS_PRESCALER_DIVIDE_7', 'EMIOS_PRESCALER_DIVIDE_8', 'EMIOS_PRESCALER_DIVIDE_9', 'EMIOS_PRESCALER_DIVIDE_10', 'EMIOS_PRESCALER_DIVIDE_11', 'EMIOS_PRESCALER_DIVIDE_12', 'EMIOS_PRESCALER_DIVIDE_13', 'EMIOS_PRESCALER_DIVIDE_14', 'EMIOS_PRESCALER_DIVIDE_15', 'EMIOS_PRESCALER_DIVIDE_16']

4.30 Parameter IcuEmiosDigitalFilter

If a eMIOS channel is being used this option is active,

possible values are: 0 (Bypassed), 2, 4, 8, 16 FLT_Clock periods.

This parameter configures programmable input filter. It selects

the minimum input pulse width [FLT_CLK periods] that can pass through the filter.

NoteThis is an Implementation Specific Parameter.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	EMIOS_DIGITAL_FILTER_BYPASSED
literals	['EMIOS_DIGITAL_FILTER_BYPASSED', 'EMIOS_DIGITAL_FILTER_02', 'EMIOS_DIGITAL_FILTER_04', 'EMIOS_DIGITAL_FILTER_08', 'EMIOS_DIGITAL_FILTER_16']

4.31 Parameter IcuEmiosBusSelect

Selects the counter used with the unified channel

NoteThis is an Implementation Specific Parameter.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	EMIOS_ICU_BUS_A
literals	['EMIOS_ICU_BUS_A', 'EMIOS_ICU_BUS_F', 'EMIOS_ICU_BUS_DIV↩ ERSE', 'EMIOS_ICU_BUS_INTERNAL_COUNTER']

4.32 Parameter IcuSubModeforMeasurement

Selection of the signal measurement mode when IcuSignalMeasurementProperty is ICU_DUTY_CYCLE.

Note

This parameter will be enabled in configuration only when IcuSignalMeasurementPropert
is ICU_DUTY_CYCLE.

The following channels support IPM and IPWM mode.

Advantages of IPM,IPWM mode over SAIC

The size of the driver code to measure duty cycle is less for a channel which
supports IPWM mode

as compared to a channel which uses SAIC mode to capture the duty cycle.

For every period of the measured signal, the interrupt generated is just 1 in
IPWM mode

as compared to a channel which uses SAIC mode where the number of
interrupt generated is 2

Because of these reasons the driver code execution speed is faster in IPWM
mode as compared to SAIC mode.

Limitation of IPM,IPWM mode over SAIC

If we configure the EMIOS channel which uses IPWM mode to capture a
duty cycle of a varying PWM Singal, we will get the interrupt

during only one edge i.e opposite edge of the starting edge. But to capture
the duty cycle of a varying PWM Signal, we have to get the interrupt

during both falling edge and rising edge. Hence in IPWM mode we get the
delayed duty cycle values

Because of this limitation of IPWM mode, we can use EMIOS channel
which uses SAIC mode

to capture the duty cycle of a varying PWM Signal.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	SAIC
literals	['SAIC', 'IPWM', 'IPM']

4.33 Parameter IcuSignalMeasureWithoutInterrupt

Icu Signal measurement without using interrupt. true: Without interruptfalse: With interrupt

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.34 Reference IcuEmiosBusRef

Select the masterbus channel on which the functionality of the current emios channel will be implemented.

Masterbus channel will be referenced from Mcl masterbus configuration

Property	Value
type	ECUC-CHOICE-REFERENCE-DEF
origin	NXP

Property	Value
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
requiresSymbolicNameValue	False
destinations	['/TS_T40D34M30I0R0/Mcl/MclConfig/EmiosCommon/EmiosMclMasterBus']

4.35 Container IcuSiul2

Configuration of a Siul2 module available on the platfom.

Included subcontainers:

- IcuSiul2Channels

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	0
upperMultiplicity	2
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE

4.36 Parameter IcuSiul2Instance

Select the hardware instance of SIUL2.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false

Property	Value
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	0
max	0
min	0

4.37 Parameter IcuEXT_ISR_InterruptFilterClockPrescaler

Configure the clock prescaler which is used to select the clock for all digital filter counters in the SIUL2NoteThis is an Implementation Specific Parameter.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	15
min	0

4.38 Parameter IcuEXT_ISR_AlternateInterruptFilterClockPrescaler

Configure the clock alternate prescaler which is used to select the clock for all digital filter counters in the SIUL2. This is only available when Dual Clock mode is activated.

NoteThis is an Implementation Specific Parameter.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1

Property	Value
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	15
min	0

4.39 Container IcuSiul2Channels

List of Siul2 Channels on flatform.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	Infinite
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE

4.40 Parameter IcuSiul2Channel

Selects one of the Siul2 channels available on the platform.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE

Property	Value
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	0
max	31
min	0

4.41 Parameter Icu_EXT_ISR_IFERDigitalFilter

Enable external digital filter counter on the interrupt pads to filter out glitches on the inputs

Note: This is an Implementation Specific Parameter.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.42 Parameter Icu_EXT_ISR_IFMCDigitalFilter

Maximum Interrupt Filter Counter setting.

Note: This is an Implementation Specific Parameter.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE

Property	Value
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	15
min	0

4.43 Container IcuWkpu

Configuration of a Wkpu module available on the platfom.

Included subcontainers:

- IcuWkpuChannels
- IcuWkpuNMIConfiguration

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	0
upperMultiplicity	Infinite
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE

4.44 Container IcuWkpuChannels

List of Wkpu modules available on the platform.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	0
upperMultiplicity	Infinite
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE

4.45 Parameter IcuWkpuChannel

Selects one of the Wkpu channels available on the platform.

From WKPU peripheral perspective the input channels are counted from 0 to 63 (64 hardware channels are available).

0 to 3 are internally routed.

4 to 63 are routed to external pins that are named as: WKPU_0 to WKPU_59.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	0
max	63
min	0

4.46 Parameter Icu_EXT_ISR_WIFERDigitalFilter

Enable external digital filter counter on the interrupt pads to filter out glitches on the inputs

Note: This is an Implementation Specific Parameter.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.47 Parameter IcuWKPU_ISR_WIPUER

Wakeup/Interrupt Pullup Enable Register is used to enable a pullup

on the corresponding interrupt pads to pull an unconnected

wakeup/interrupt input to a value of '1'.

NoteThis is an Implementation Specific Parameter.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.48 Container IcuWkpuNMIConfiguration

The WKPU NMI configuration for each core.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	0
upperMultiplicity	Infinite
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE

4.49 Parameter NMICoreSource

Selects one of the supported cores for the NMI.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	2
min	0

4.50 Parameter DestinationSourceSelect

NMI Destination Source Select

As wakeup does not support another interrupt than NMI, the destination source select signal bits are reserved and always retain their reset value. This means no other request other than NMI can be generated.

NMI_NON_MASK_INT: Non-maskable interrupt

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	NMI_NON_MASK_INT
literals	['NMI_NON_MASK_INT']

4.51 Parameter WakeupRequestEnable

System wakeup requests from the corresponding NIF0 field

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.52 Parameter FilterEnable

Enable analog glitch filter on the NMI pad input..

NoteThis is an Implementation Specific Parameter.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.53 Parameter NMIEdgeEvents

Configures the NMI Edge Events which shall be used for this NMI

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	RISING_EDGE
literals	['RISING_EDGE', 'FALLING_EDGE', 'BOTH_EDGES']

4.54 Parameter LockRegister

Locks the configuration for the NMI until it is unlocked by a system reset or STANDBY0 mode exit.

NoteThis is an Implementation Specific Parameter.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.55 Container IcuLpCmp

Configuration of a LPCMP module available on the platfom.

Included subcontainers:

- IcuCmp

- IcuDac
- IcuTrigger

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	0
upperMultiplicity	3
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE

4.56 Parameter IcuCmpInstanceNumber

Configure the instance number of IP used.NoteThis is an Implementation Specific Parameter.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	2
min	0

4.57 Container IcuCmp

Configuration of a LPCMP module available on the platfom.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.58 Parameter IcuCmpFunctionalMode

Functional mode of LPCMP

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	CMP_IP_FUNCTIONALMODE_DISABLED
literals	['CMP_IP_FUNCTIONALMODE_DISABLED', 'CMP_IP_FUNCTIONAL↩ MODE_CONTINUOUS', 'CMP_IP_FUNCTIONALMODE_SAMPLED_N↩ ONFILTERED_INT_CLK', 'CMP_IP_FUNCTIONALMODE_SAMPLED↩ _NONFILTERED_EXT_CLK', 'CMP_IP_FUNCTIONALMODE_SAMPL↩ ED_FILTERED_INT_CLK', 'CMP_IP_FUNCTIONALMODE_SAMPLED↩ _FILTERED_EXT_CLK', 'CMP_IP_FUNCTIONALMODE_WINDOWED', 'CMP_IP_FUNCTIONALMODE_WINDOWED_RESAMPLED', 'CMP_IP↩ _FUNCTIONALMODE_WINDOWED_FILTERED']

4.59 Parameter IcuCmpHysteresisLevel

Internal hysteresis mode of LPCMP - see specific implementation

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	CMP_IP_HYSTERESISLEVEL_0
literals	['CMP_IP_HYSTERESISLEVEL_0', 'CMP_IP_HYSTERESISLEVEL_1', 'CMP_IP_HYSTERESISLEVEL_2', 'CMP_IP_HYSTERESISLEVEL_3']

4.60 Parameter IcuCmpOffsetLevel

Comparator offset control - see specific implementation

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	CMP_IP_OFFSETLEVEL_0
literals	['CMP_IP_OFFSETLEVEL_0', 'CMP_IP_OFFSETLEVEL_1']

4.61 Parameter IcuCmpEnablePinOutput

EnablePinOutput.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	false

4.62 Parameter IcuCmpEnableInverter

EnableInverter.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	false

4.63 Parameter IcuCmpEnableComparatorInvert

EnableComparatorInvert.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	false

4.64 Parameter IcuCmpEnableHighPowerMode

EnableHighPowerMode.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1

Property	Value
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	false

4.65 Parameter IcuCmpFilterSamplePeriod

FilterSamplePeriod

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	255
min	0

4.66 Parameter IcuCmpFilterSampleCount

FilterSampleCount

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true

Property	Value
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	255
min	0

4.67 Parameter IcuCmpEnableDma

EnableDma.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	false

4.68 Parameter IcuCmpNegativeInputMux

NegativeInputMux

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	CMP_IP_INPUTMUX_IN0

Property	Value
literals	['CMP_IP_INPUTMUX_IN0', 'CMP_IP_INPUTMUX_IN1', 'CMP_IP_I↩ NPUTMUX_IN2', 'CMP_IP_INPUTMUX_IN3', 'CMP_IP_INPUTMUX_↩ IN4', 'CMP_IP_INPUTMUX_IN5', 'CMP_IP_INPUTMUX_IN6', 'CMP_I↩ P_INPUTMUX_IN7', 'CMP_IP_INPUTMUX_DAC']

4.69 Parameter IcuCmpPositiveInputMux

PositiveInputMux

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	CMP_IP_INPUTMUX_IN0
literals	['CMP_IP_INPUTMUX_IN0', 'CMP_IP_INPUTMUX_IN1', 'CMP_IP_I↩ NPUTMUX_IN2', 'CMP_IP_INPUTMUX_IN3', 'CMP_IP_INPUTMUX_↩ IN4', 'CMP_IP_INPUTMUX_IN5', 'CMP_IP_INPUTMUX_IN6', 'CMP_I↩ P_INPUTMUX_IN7', 'CMP_IP_INPUTMUX_DAC']

4.70 Parameter IcuCmpOutputSelect

OutputSelect

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE

Property	Value
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	CMP_IP_OUTPUTSELECT_COUT
literals	['CMP_IP_OUTPUTSELECT_COUT', 'CMP_IP_OUTPUTSELECT_COUTA']

4.71 Parameter IcuCmpWindowCloseOutputOverwrite

WindowCloseOutputOverwrite

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	CMP_IP_WINDOWCLOSEOUTPUTOVERWRITE_DISABLED
literals	['CMP_IP_WINDOWCLOSEOUTPUTOVERWRITE_DISABLED', 'CMP_↩IP_WINDOWCLOSEOUTPUTOVERWRITE_LOW', 'CMP_IP_WINDOW↩CLOSEOUTPUTOVERWRITE_HIGH']

4.72 Parameter IcuCmpWindowCloseEvent

WindowCloseEvent

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE

Property	Value
defaultValue	CMP_IP_WINDOWCLOSEEVENT_RISING
literals	['CMP_IP_WINDOWCLOSEEVENT_RISING', 'CMP_IP_WINDOWCLO↩ SEEVENT_FALLING', 'CMP_IP_WINDOWCLOSEEVENT_BOTH', 'CM↩ P_IP_WINDOWCLOSEEVENT_NONE']

4.73 Parameter IcuCmpEnableInStop

EnableInStop.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	false

4.74 Container IcuDac

Configuration of a LPCMP module available on the platfom.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.75 Parameter IcuDacVoltageLevel

VoltageLevel

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	255
min	0

4.76 Parameter IcuDacVoltageRefSource

VoltageRefSource

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	CMP_IP_VOLTAGEREFSOURCE_VREF0
literals	['CMP_IP_VOLTAGEREFSOURCE_VREF0', 'CMP_IP_VOLTAGEREFS↩ OURCE_VREF1']

4.77 Parameter IcuDacPowerState

PowerState

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	CMP_IP_DACPOWERSTATE_DISABLED
literals	['CMP_IP_DACPOWERSTATE_DISABLED', 'CMP_IP_DACPOWERST↩ ATE_ENABLED', 'CMP_IP_DACPOWERSTATE_LINKED']

4.78 Container IcuTrigger

Configuration of a LPCMP module available on the platfom.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.79 Parameter IcuTrgRoundRobinEnChannelMask

RoundRobinEnChannelMask

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1

Property	Value
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	255
min	0

4.80 Parameter IcuTrgPrepgmStateChannelMask

PrepgmStateChannelMaks

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	255
min	0

4.81 Parameter IcuTrgInitDelayValue

InitDelayValue

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A

Property	Value
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	255
min	0

4.82 Parameter IcuTrgSampleDelay

SampleDelay

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	CMP_IP_SAMPLEDELAY_0_CYCLES
literals	['CMP_IP_SAMPLEDELAY_0_CYCLES', 'CMP_IP_SAMPLEDELAY_1↩ _CYCLES', 'CMP_IP_SAMPLEDELAY_2_CYCLES', 'CMP_IP_SAMPL↩ EDELAY_3_CYCLES']

4.83 Parameter IcuTrgFixedChannel

FixedChannel

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

Property	Value
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	CMP_IP_FIXEDCHANNEL_0
literals	['CMP_IP_FIXEDCHANNEL_0', 'CMP_IP_FIXEDCHANNEL_1', 'CMP_↩ IP_FIXEDCHANNEL_2', 'CMP_IP_FIXEDCHANNEL_3', 'CMP_IP_FIX↩ EDCHANNEL_4', 'CMP_IP_FIXEDCHANNEL_5', 'CMP_IP_FIXEDCHA↩ NNEL_6', 'CMP_IP_FIXEDCHANNEL_7']

4.84 Parameter IcuTrgFixedPort

FixedPort

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	CMP_IP_FIXEDPORT_PLUS
literals	['CMP_IP_FIXEDPORT_PLUS', 'CMP_IP_FIXEDPORT_MINUS']

4.85 Parameter IcuTrgEnableRoundRobinInterrupt

EnableRoundRobinInterrupt.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

Property	Value
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	false

4.86 Parameter IcuTrgEnableRoundRobin

EnableRoundRobin.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	false

4.87 Container IcuHwInterruptConfigList

List of HW interrupts available for the entire platform.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	0
upperMultiplicity	Infinite
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE

4.88 Parameter IcuIsrHwId

Id of the HW interrupt service routine available platform wide and usable by ICU module.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	true
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	EMIOS_0_CH_1

Property	Value
literals	['EMIOS_0_CH_0', 'EMIOS_0_CH_1', 'EMIOS_0_CH_2', 'EMIOS_0_CH_3', 'EMIOS_0_CH_4', 'EMIOS_0_CH_5', 'EMIOS_0_CH_6', 'EMIOS_0_CH_7', 'EMIOS_0_CH_8', 'EMIOS_0_CH_9', 'EMIOS_0_CH_10', 'EMIOS_0_CH_11', 'EMIOS_0_CH_12', 'EMIOS_0_CH_13', 'EMIOS_0_CH_14', 'EMIOS_0_CH_15', 'EMIOS_0_CH_16', 'EMIOS_0_CH_17', 'EMIOS_0_CH_18', 'EMIOS_0_CH_19', 'EMIOS_0_CH_20', 'EMIOS_0_CH_21', 'EMIOS_0_CH_22', 'EMIOS_0_CH_23', 'EMIOS_1_CH_0', 'EMIOS_1_CH_1', 'EMIOS_1_CH_2', 'EMIOS_1_CH_3', 'EMIOS_1_CH_4', 'EMIOS_1_CH_5', 'EMIOS_1_CH_6', 'EMIOS_1_CH_7', 'EMIOS_1_CH_8', 'EMIOS_1_CH_9', 'EMIOS_1_CH_10', 'EMIOS_1_CH_11', 'EMIOS_1_CH_12', 'EMIOS_1_CH_13', 'EMIOS_1_CH_14', 'EMIOS_1_CH_15', 'EMIOS_1_CH_16', 'EMIOS_1_CH_17', 'EMIOS_1_CH_18', 'EMIOS_1_CH_19', 'EMIOS_1_CH_20', 'EMIOS_1_CH_21', 'EMIOS_1_CH_22', 'EMIOS_1_CH_23', 'EMIOS_2_CH_0', 'EMIOS_2_CH_1', 'EMIOS_2_CH_2', 'EMIOS_2_CH_3', 'EMIOS_2_CH_4', 'EMIOS_2_CH_5', 'EMIOS_2_CH_6', 'EMIOS_2_CH_7', 'EMIOS_2_CH_8', 'EMIOS_2_CH_9', 'EMIOS_2_CH_10', 'EMIOS_2_CH_11', 'EMIOS_2_CH_12', 'EMIOS_2_CH_13', 'EMIOS_2_CH_14', 'EMIOS_2_CH_15', 'EMIOS_2_CH_16', 'EMIOS_2_CH_17', 'EMIOS_2_CH_18', 'EMIOS_2_CH_19', 'EMIOS_2_CH_20', 'EMIOS_2_CH_21', 'EMIOS_2_CH_22', 'EMIOS_2_CH_23', 'WKPU_CH_0', 'WKPU_CH_1', 'WKPU_CH_2', 'WKPU_CH_3', 'WKPU_CH_4', 'WKPU_CH_5', 'WKPU_CH_6', 'WKPU_CH_7', 'WKPU_CH_8', 'WKPU_CH_9', 'WKPU_CH_10', 'WKPU_CH_11', 'WKPU_CH_12', 'WKPU_CH_13', 'WKPU_CH_14', 'WKPU_CH_15', 'WKPU_CH_16', 'WKPU_CH_17', 'WKPU_CH_18', 'WKPU_CH_19', 'WKPU_CH_20', 'WKPU_CH_21', 'WKPU_CH_22', 'WKPU_CH_23', 'WKPU_CH_24', 'WKPU_CH_25', 'WKPU_CH_26', 'WKPU_CH_27', 'WKPU_CH_28', 'WKPU_CH_29', 'WKPU_CH_30', 'WKPU_CH_31', 'WKPU_CH_32', 'WKPU_CH_33', 'WKPU_CH_34', 'WKPU_CH_35', 'WKPU_CH_36', 'WKPU_CH_37', 'WKPU_CH_38', 'WKPU_CH_39', 'WKPU_CH_40', 'WKPU_CH_41', 'WKPU_CH_42', 'WKPU_CH_43', 'WKPU_CH_44', 'WKPU_CH_45', 'WKPU_CH_46', 'WKPU_CH_47', 'WKPU_CH_48', 'WKPU_CH_49', 'WKPU_CH_50', 'WKPU_CH_51', 'WKPU_CH_52', 'WKPU_CH_53', 'WKPU_CH_54', 'WKPU_CH_55', 'WKPU_CH_56', 'WKPU_CH_57', 'WKPU_CH_58', 'WKPU_CH_59', 'CMP_0', 'CMP_1', 'CMP_2', 'SIUL2_0_IRQ_CH_0', 'SIUL2_0_IRQ_CH_1', 'SIUL2_0_IRQ_CH_2', 'SIUL2_0_IRQ_CH_3', 'SIUL2_0_IRQ_CH_4', 'SIUL2_0_IRQ_CH_5', 'SIUL2_0_IRQ_CH_6', 'SIUL2_0_IRQ_CH_7', 'SIUL2_0_IRQ_CH_8', 'SIUL2_0_IRQ_CH_9', 'SIUL2_0_IRQ_CH_10', 'SIUL2_0_IRQ_CH_11', 'SIUL2_0_IRQ_CH_12', 'SIUL2_0_IRQ_CH_13', 'SIUL2_0_IRQ_CH_14', 'SIUL2_0_IRQ_CH_15', 'SIUL2_0_IRQ_CH_16', 'SIUL2_0_IRQ_CH_17', 'SIUL2_0_IRQ_CH_18', 'SIUL2_0_IRQ_CH_19', 'SIUL2_0_IRQ_CH_20', 'SIUL2_0_IRQ_CH_21', 'SIUL2_0_IRQ_CH_22', 'SIUL2_0_IRQ_CH_23', 'SIUL2_0_IRQ_CH_24', 'SIUL2_0_IRQ_CH_25', 'SIUL2_0_IRQ_CH_26', 'SIUL2_0_IRQ_CH_27', 'SIUL2_0_IRQ_CH_28', 'SIUL2_0_IRQ_CH_29', 'SIUL2_0_IRQ_CH_30', 'SIUL2_0_IRQ_CH_31']

4.89 Parameter IcuIsrEnable

Status of the HW Interrupt (true - Interrupt shall be enable platform wide; false - Interrupt shall be disabled

platform wide.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	false

4.90 Container IcuGeneral

Configuration of general ICU parameters.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.91 Parameter IcuDevErrorDetect

Switches the Development Error Detection and Notification on or off.

true: Enabled.

false: Disabled.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF

Property	Value
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	true

4.92 Parameter IcuReportWakeupSource

Switch for enabling Wakeup source reporting.

true: Report Wakeup source.

false: Do not report Wakeup source.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	true

4.93 Parameter IcuEnableUserModeSupport

When this parameter is enabled, the Icu module will adapt to run from User Mode, with the following measures:

a) configuring REG_PROT for SIUL2 IP so that the registers under protection can be accessed from user mode by setting UAA bit in REG_PROT_GCR to 1

b) using 'call trusted function' stubs for all internal function calls that access registers requiring supervisor mode.

for more information, please see chapter 5.7 User Mode Support in IM

Note: Implementation Specific Parameter.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.94 Parameter IcuMulticoreSupport

When this parameter is enabled, the ICU module will adapt to run with Multicore:

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.95 Reference IcuEcucPartitionRef

Maps the ICU driver to zero or multiple ECUC partitions to make the driver API available in the according partition.

Depending on the addressed timer resource the interfaces operate as follows.

Property	Value
type	ECUC-REFERENCE-DEF
origin	AUTOSAR_ECUC
lowerMultiplicity	0

Property	Value
upperMultiplicity	Infinite
postBuildVariantMultiplicity	true
multiplicityConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
requiresSymbolicNameValue	False
destination	/AUTOSAR/EcucDefs/EcuC/EcucPartitionCollection/EcucPartition

4.96 Reference IcuKernelEcucPartitionRef

Maps the ICU kernel to zero or one ECUC partitions to assign the driver kernel to a certain core.

The ECUC partition referenced is a subset of the ECUC partitions where the ICU driver is mapped to.

Property	Value
type	ECUC-REFERENCE-DEF
origin	AUTOSAR_ECUC
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	true
multiplicityConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
requiresSymbolicNameValue	False
destination	/AUTOSAR/EcucDefs/EcuC/EcucPartitionCollection/EcucPartition

4.97 Container IcuAutosarExt

Enabling the settings of this section will configure the driver in a mode not compliant with AUTOSAR requirements.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF

Property	Value
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.98 Parameter IcuEnableDualClockMode

Enables prescaler settings at mode transition.

true: Enabled.

false: Disabled.

Note: This feature is not required by Autosar.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.99 Parameter IcuOverflowNotificationApi

Add / removes Overflow Notification functionality.

Enabling IcuOverflowNotificationApi overflow events will not be treated as errors and a Notification Handler can be provided.

If this optional API is not enabled, overflow events will trigger DET Report Error.

Note:

Due to hardware implementattion, the Icu Overflow Notification is not syncronous with

the event for ICU_MODE_SIGNAL_MEASUREMENT and ICU_MODE_TIMESTAMP modes. The notification

will be triggered when measurement completes (for ICU_MODE_SIGNAL_MEASUREMENT) or the

next timestamp event occurs (for ICU_MODE_TIMESTAMP).

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.100 Parameter IcuGetInputLevelApi

Add / removes Icu_GetInputLevel API from the code.

This function returns Input pin state.

Note: This feature is not required by Autosar.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.101 Parameter IcuGetCaptureRegisterValueApi

Adds / removes service Icu_GetCaptureRegisterValue from the code.

This function returns value of Capture register for the mesurement channel or timestamp mode channel which is called by the user.

It's enabled when IcuTimestampApi or IcuSignalMeasurementApi is true.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.102 Parameter IcuSupportSAICModeApi

Enable/Disable SAIC mode for eMios IP.

Note: This feature is not required by Autosar.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.103 Parameter IcuWkpuStandbyWakeupSupport

Icu_Init() will not clear the wakeup flags (WISR register) if it is already set during init.

Note: This feature is not required by Autosar.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP

Property	Value
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.104 Parameter IcuSetMaxCounterValue

Enable/Disable support changer Max Counter value.

Note: This feature is not required by Autosar and only supports eMios IPV in ICU_MODE_EDGE_COUNTER mode.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.105 Parameter IcuSetInitialCounterValue

Enable/Disable support changer Initial Counter value.

Note: This feature is not required by Autosar and only supports eMios IPV in ICU_MODE_EDGE_COUNTER mode.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP

Property	Value
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.106 Container IcuOptionalApis

This container contains all configuration switches for configuring optional API services of the ICU driver.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.107 Parameter IcuDeInitApi

Adds / removes the service Icu_DeInit() from the code.

true: Icu_DeInit() can be used.

false: Icu_DeInit() can not be used.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1

Property	Value
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	true

4.108 Parameter IcuDisableWakeupApi

Adds / removes the service Icu_DisableWakeup() from the code.

true: Icu_DisableWakeup() can be used.

false: Icu_DisableWakeup() can not be used.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	true

4.109 Parameter IcuEdgeCountApi

Adds / removes all services related to the edge counting

functionality as listed below, from the code: Icu_ResetEdgeCount(),

Icu_EnableEdgeCount(), Icu_DisableEdgeCount(), Icu_GetEdgeNumbers().

true: The services listed above can be used.

false: The services listed above can not be used.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	true

4.110 Parameter IcuEnableWakeupApi

Adds / removes the service Icu_EnableWakeup() from the code.

true: Icu_EnableWakeup() can be used.

false: Icu_EnableWakeup() can not be used.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	true

4.111 Parameter IcuGetDutyCycleValuesApi

Adds / removes the service Icu_GetDutyCycleValues() from the code.

true: Icu_GetDutyCycleValues() can be used.

false: Icu_GetDutyCycleValues() can not be used.

Note: If IcuSignalMeasurementApi == OFF this switch is shall also be set to OFF.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	true

4.112 Parameter IcuGetInputStateApi

Adds / removes the service Icu_GetInputState() from the code.

true: Icu_GetInputState() can be used.

false: Icu_GetInputState() can not be used.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	true

4.113 Parameter IcuGetTimeElapsedApi

Adds / removes the service Icu_GetTimeElapsed() from the code.

true: Icu_GetTimeElapsed() can be used.

false: Icu_GetTimeElapsed() can not be used.

Note: If IcuSignalMeasurementApi == OFF this switch is shall also be set to OFF.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	true

4.114 Parameter IcuGetVersionInfoApi

Adds / removes the service Icu_GetVersionInfo() from the code.

true: Icu_GetVersionInfo() can be used.

false: Icu_GetVersionInfo() can not be used.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	true

4.115 Parameter IcuSetModeApi

Adds / removes the service Icu_SetMode() from the code.

true: Icu_SetMode() can be used.

false: Icu_SetMode() can not be used.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	true

4.116 Parameter IcuSignalMeasurementApi

Adds / removes the services Icu_StartSignalMeasurement() and Icu_StopSignalMeasurement() from the code.

true: Icu_StartSignalMeasurement() and Icu_StopSignalMeasurement() can be used.

false: Icu_StartSignalMeasurement() and Icu_StopSignalMeasurement() can not be used.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	true

4.117 Parameter IcuTimestampApi

Adds / removes all services related to the timestamping functionality as listed below from the code:
Icu_StartTimestamp(), Icu_StopTimestamp(), Icu_GetTimestampIndex().

true: The services listed above can be used.

false: The services listed above can not be used.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	true

4.118 Parameter IcuWakeupFunctionalityApi

Adds / removes the service Icu_CheckWakeup() from the code.

true: Icu_CheckWakeup() can be used.

false: Icu_CheckWakeup() can not be used.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	true

4.119 Parameter IcuEdgeDetectApi

Adds / removes the services Icu_EnableEdgeDetection() and Icu_DisableEdgeDetection() from the code.

true: Icu_EnableEdgeDetection() and Icu_DisableEdgeDetection() can be used.

false: Icu_EnableEdgeDetection() and Icu_DisableEdgeDetection() can not be used.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	true

4.120 Container CommonPublishedInformation

Common container, aggregated by all modules. It contains published information about vendor and versions.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.121 Parameter ArReleaseMajorVersion

Major version number of AUTOSAR specification on which the appropriate implementation is based on.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A

Property	Value
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
defaultValue	4
max	4
min	4

4.122 Parameter ArReleaseMinorVersion

Minor version number of AUTOSAR specification on which the appropriate implementation is based on.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
defaultValue	7
max	7
min	7

4.123 Parameter ArReleaseRevisionVersion

Revision version number of AUTOSAR specification on which the appropriate implementation is based on.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

Property	Value
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
defaultValue	0
max	0
min	0

4.124 Parameter ModuleId

Module ID of this module from Module List.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
defaultValue	122
max	122
min	122

4.125 Parameter SwMajorVersion

Major version number of the vendor specific implementation of the module. The numbering is vendor specific.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION

Property	Value
	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
defaultValue	3
max	3
min	3

4.126 Parameter SwMinorVersion

Minor version number of the vendor specific implementation of the module. The numbering is vendor specific.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
defaultValue	0
max	0
min	0

4.127 Parameter SwPatchVersion

Patch level version number of the vendor specific implementation of the module. The numbering is vendor specific.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
defaultValue	0

Property	Value
max	0
min	0

4.128 Parameter VendorApiInfix

In driver modules which can be instantiated several times on a single ECU, BSW00347 requires

that the name of APIs is extended by the VendorId and a vendor specific name.

This parameter is used to specify the vendor specific name. In total, the

implementation specific name is generated as follows: <ModuleName>_>VendorId>_<VendorApiInfix>.

E.g. assuming that the VendorId of the implementor is 123 and the implementer chose a VendorApiInfix of "v11r456" a api name Can_Write defined in the SWS will translate to Can_123_v11r456Write.

This parameter is mandatory for all modules with upper multiplicity > 1. It shall not be used for modules with upper multiplicity =1.

Property	Value
type	ECUC-STRING-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
defaultValue	

4.129 Parameter VendorId

Vendor ID of the dedicated implementation of this module according to the AUTOSAR vendor list.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false

Property	Value
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
defaultValue	43
max	43
min	43

None.

Chapter 5

Module Index

5.1 Software Specification

Here is a list of all modules:

Chapter 6

Module Documentation

6.1 WKPU IPL

6.1.1 Detailed Description WKPU HW module.

The Wakeup Unit (WKPU) supports external sources that can generate interrupts or wakeup events and external source(s) that can cause non-maskable interrupt request(s) or wakeup event(s).
In addition, it combines its wakeup events with those generated from other wakeup sources to supply a single wakeup to the system.

Wakeup Unit Configurations
The 'Wakeup Unit (WKPU)' provides a configurable low-power wakeup capability to the device from multiple configurable asynchronous wakeup events. The wakeup unit on the device supports 4 internal sources and 60 external sources that can generate interrupts or wakeup events. It also supports a non-maskable interrupt input.

External signal description

- The WKPU has signal inputs that can be used as external interrupt sources in normal run mode or as system wakeup sources in certain power down modes.

NMI configuration

The Wakeup Unit (WKPU) supports one external source that can cause non-maskable interrupts to on-chip cores and wakeup events to the system. Herein there are two application cores indicated, CM7_0 and CM7_1. In the event of any wakeup (internal or external), the WKPU initiates the recovery of the device and feeds this interrupt to the core(s) depending on the configurations.

Data Structures

- struct Wkpu_Ip_ChannelConfigType
  WKPU interrupt configuration structure. *More...*
- struct Wkpu_Ip_IrqConfigType
  Wkpu IP specific configuration structure type. *More...*
- struct Wkpu_Ip_State
  WKPU IP state structure. *More...*

Types Reference

- typedef void(* Wkpu_Ip_NotifyType) (void)
  The notification functions shall have no parameters and no return value.

Enum Reference

- enum Wkpu_Ip_EdgeType
  Edge event.
- enum Wkpu_Ip_StatusType
  Wkpu_Ip_StatusType.

Function Reference

- void Wkpu_Ip_EnableInterrupt (uint8 instance, uint8 hwChannel)
  Enable the interrupt request and wakeup generation.
- void Wkpu_Ip_DisableInterrupt (uint8 instance, uint8 hwChannel)
  ICU driver function that disables the interrupt of a WKPU channel.
- Wkpu_Ip_StatusType Wkpu_Ip_Init (uint8 instance, const Wkpu_Ip_IrqConfigType *userConfig)
  Icu driver function that initializes WKPU channels.
- Wkpu_Ip_StatusType Wkpu_Ip_DeInit (uint8 instance)
  ICU driver function that resets WKPU configuration.
- void Wkpu_Ip_SetActivationCondition (uint8 instance, uint8 hwChannel, Wkpu_Ip_EdgeType edge)
  ICU driver function that sets activation condition of WKPU channel.
- boolean Wkpu_Ip_GetInputState (uint8 instance, uint8 hwChannel)
  ICU driver function that gets the input state of WKPU channel.
- void Wkpu_Ip_EnableNotification (uint8 hwChannel)
  Driver function Enable Notification for timestamp.
- void Wkpu_Ip_DisableNotification (uint8 hwChannel)
  Driver function Disable Notification for timestamp.

6.1.2 Data Structure Documentation

6.1.2.1 struct Wkpu_Ip_ChannelConfigType

WKPU interrupt configuration structure.

Definition at line 174 of file Wkpu_Ip_Types.h.

Data Fields

Type	Name	Description
uint8	hwChannel	The WKPU hardware channel.
boolean	filterEn	WKPU/interrupt filter enable.
boolean	pullEn	WKPU/interrupt pull enable.
Wkpu_Ip_EdgeType	edgeEvent	WKPU/interrupt edge events.
Wkpu_Ip_CallbackType	callback	Pointer to the callback function.
Wkpu_Ip_NotifyType	WkpuChannelNotification	The notification functions shall have no parameters and no return value.
uint16	callbackParam	The logic channel for which callback is set.

6.1.2.2 struct Wkpu_Ip_IrqConfigType

Wkpu IP specific configuration structure type.

Definition at line 192 of file Wkpu_Ip_Types.h.

Data Fields

Type	Name	Description
uint8	numChannels	Number of channels in configuration.
const Wkpu_Ip_ChannelConfigType(*	pChannelsConfig)[]	Pointer to channels configration.

6.1.2.3 struct Wkpu_Ip_State

WKPU IP state structure.

This structure is used by the IPL driver for internal logic. The content is populated at initialization time.

Definition at line 211 of file Wkpu_Ip_Types.h.

Data Fields

Type	Name	Description
boolean	chInit	Initialization state.
Wkpu_Ip_CallbackType	callback	Pointer to the callback function.
Wkpu_Ip_NotifyType	WkpuChannelNotification	The notification functions for SIGNAL_EDGE_DETECT mode.
uint16	callbackParam	The logic channel for which callback is set.
boolean	notificationEnable	Store the initialization state that determines whether Notifications are enabled.

6.1.3 Types Reference

6.1.3.1 Wkpu_Ip_NotifyType

```
typedef void(* Wkpu_Ip_NotifyType) (void)
```

The notification functions shall have no parameters and no return value.

Definition at line 143 of file Wkpu_Ip_Types.h.

6.1.4 Enum Reference

6.1.4.1 Wkpu_Ip_EdgeType

```
enum Wkpu_Ip_EdgeType
```

Edge event.

Enumerator

WKPU_IP_NONE_EDGE	None event.
WKPU_IP_RISING_EDGE	Rising edge event.
WKPU_IP_FALLING_EDGE	Falling edge event.
WKPU_IP_BOTH_EDGES	Both rising and falling edge event.

Definition at line 122 of file Wkpu_Ip_Types.h.

6.1.4.2 Wkpu_Ip_StatusType

```
enum Wkpu_Ip_StatusType
```

Wkpu_Ip_StatusType.

This indicates the operation success or fail

Enumerator

WKPU_IP_SUCCESS	Status for success operation return.
WKPU_IP_ERROR	General error return status.

Definition at line 134 of file Wkpu_Ip_Types.h.

6.1.5 Function Reference

6.1.5.1 Wkpu_Ip_EnableInterrupt()

```
void Wkpu_Ip_EnableInterrupt (
             uint8 instance,
             uint8 hwChannel )
```

Enable the interrupt request and wakeup generation.

This function setup generation of interrupt and wakeup generation.

Parameters

in	*instance*	Hardware instance of WKPU used.
in	*hwChannel*	Hardware channel of WKPU used.

Returns

void

6.1.5.2 Wkpu_Ip_DisableInterrupt()

```
void Wkpu_Ip_DisableInterrupt (
             uint8 instance,
             uint8 hwChannel )
```

ICU driver function that disables the interrupt of a WKPU channel.

This function disables WKPU Channel Interrupt.

Parameters

in	*instance*	Hardware instance of WKPU used.
in	*hwChannel*	Hardware channel of WKPU used.

Returns

void

6.1.5.3 Wkpu_Ip_Init()

```
Wkpu_Ip_StatusType Wkpu_Ip_Init (
            uint8 instance,
            const Wkpu_Ip_IrqConfigType * userConfig )
```

Icu driver function that initializes WKPU channels.

This function:

- Sets Interrupt Filter Enable Register
- Sets Wakeup/Interrupt Pull-up Enable Register
- Sets Activation Condition

Parameters

in	*instance*	Hardware instance of WKPU used.
in	*userConfig*	- Pointer to array of with channels configuration.

Returns

void

6.1.5.4 Wkpu_Ip_DeInit()

```
Wkpu_Ip_StatusType Wkpu_Ip_DeInit (
            uint8 instance )
```

ICU driver function that resets WKPU configuration.

This function:

- Disables IRQ Interrupt
- Clears Wakeup/Interrupt Filter Enable Register
- Clears Wakeup/Interrupt Pull-up Enable Register
- Clears edge event enable registers
- Clear Interrupt Filter Enable Register

Parameters

in	*instance*	Hardware instance of WKPU used.

Returns

void

6.1.5.5 Wkpu_Ip_SetActivationCondition()

```
void Wkpu_Ip_SetActivationCondition (
             uint8 instance,
             uint8 hwChannel,
             Wkpu_Ip_EdgeType edge )
```

ICU driver function that sets activation condition of WKPU channel.

This function enables the requested activation condition(rising, falling or both edges) for corresponding WKPU channels.

Parameters

in	*instance*	Hardware instance of WKPU used.
in	*hwChannel*	Hardware channel of WKPU used.
in	*edge*	Edge type for activation.

Returns

void

6.1.5.6 Wkpu_Ip_GetInputState()

```
boolean Wkpu_Ip_GetInputState (
             uint8 instance,
             uint8 hwChannel )
```

ICU driver function that gets the input state of WKPU channel.

This function:

- Checks if interrupt flags for corresponding WKPU channel is set then it clears the interrupt flag and returns the value as TRUE.

Parameters

in	*instance*	Hardware instance of WKPU used.
in	*hwChannel*	Hardware channel of WKPU used.

Returns

boolean

- TRUE - if channel is active
- FALSE - If channel is in idle

6.1.5.7 Wkpu_Ip_EnableNotification()

```
void Wkpu_Ip_EnableNotification (
             uint8 hwChannel )
```

Driver function Enable Notification for timestamp.

6.1.5.8 Wkpu_Ip_DisableNotification()

```
void Wkpu_Ip_DisableNotification (
             uint8 hwChannel )
```

Driver function Disable Notification for timestamp.

6.2 EMIOS IPL

6.2.1 Detailed Description Enhanced Modular IO Subsystem (eMIOS)

Driver consideration The S32K3XX has up to 3 instances of eMIOS that are each configured as in RM.

Input capture mode Initialize Input Measurement Mode or Single Action Input Capture mode. Have 2 options:

- Input Measurement Mode This mode allows the measurement of the width of a positive or negative pulse or period of an input signal by capturing two consecutive rising edges or two consecutive falling edges. In period measurement mode: Successive input captures are done on consecutive edges of the same polarity. The input signal must have at least four system clock cycles period in order to be properly captured by the synchronization logic at the channel input even if the input filter is in by-pass mode.
- Single Action Input Capture mode In SAIC mode, when a triggering event occurs on the input pin, the value on the selected time base is captured.

Data Structures

- struct eMios_Icu_Ip_DutyCycleType

  Structure that contains ICU Duty cycle parameters. It contains the values needed for calculating duty cycles i.e Period time value and active time value. More...
- struct eMios_Icu_Ip_ConfigType

  eTimer IP specific configuration structure type More...
- struct eMios_Icu_Ip_ChStateType

  This structure is used by the IPL driver for internal logic. More...

Macros

- #define EMIOS_ICU_IP_CB_NONE

  EMIOS Channels defines.

Types Reference

- typedef void(* eMios_Icu_Ip_NotifyType) (void)

  The notification functions shall have no parameters and no return value.
- typedef void(* eMios_Icu_Ip_CallbackType) (uint16 callbackParam1, boolean callbackParam2)

  HLD Callback type for each channel reporting events or events and overflow .
- typedef void(* eMios_Icu_Ip_LogicChStateCbType) (uint16 logicChannel, uint8 mask, boolean set)

  Callback type for each channel.

Enum Reference

- enum eMios_Icu_Ip_EdgeType
  eMIOS_Activation EDGE
- enum eMios_Icu_Ip_ModeType
  Operation mode for ICU driver.
- enum eMios_Icu_Ip_SubModeType
  Enable/disable DMA support for timestamp.
- enum eMios_Icu_Ip_MeasStatusType
  Stores the state in which a signal measurement is.
- enum eMios_Icu_Ip_MeasType
  Type of operation for signal measurement.
- enum eMios_Icu_Ip_LevelType
  Enumeration used for returning the level of input pin.
- enum eMios_Icu_Ip_ClockModeType
  Definition of prescaler type (Normal or Alternate)
- enum eMios_Icu_Ip_BusType
  Definition of master bus type.
- enum eMios_Icu_Ip_StatusType
  Generic error codes.
- enum eMios_Icu_Ip_UCModeType
  Selection of the signal measurement mode when IcuSignalMeasurementProperty is ICU_DUTY_CYCLE.
- enum eMios_Icu_Ip_PrescalerType
  Selects the clock divider value for the UC internal prescaler.
- enum eMios_Icu_Ip_FilterType
  Selects the the input filter.

Function Reference

- eMios_Icu_Ip_StatusType Emios_Icu_Ip_Init (uint8 instance, const eMios_Icu_Ip_ConfigType *user↩ Config)
  Emios_Icu_Ip_Init.
- eMios_Icu_Ip_StatusType Emios_Icu_Ip_Deinit (uint8 instance)
  Emios_Icu_Ip_Deinit.
- void Emios_Icu_Ip_SetActivation (uint8 instance, uint8 hwChannel, eMios_Icu_Ip_EdgeType edge)
  Icu driver function that sets activation condition of eMIOS channel.
- void Emios_Icu_Ip_EnableEdgeDetection (uint8 instance, uint8 hwChannel)
  Emios_Icu_Ip_EnableEdgeDetection.
- void Emios_Icu_Ip_DisableEdgeDetection (uint8 instance, uint8 hwChannel)
  Emios_Icu_Ip_DisableEdgeDetection.
- void Emios_Icu_Ip_EnableNotification (uint8 instance, uint8 hwChannel)
  Driver function Enable Notification.
- void Emios_Icu_Ip_DisableNotification (uint8 instance, uint8 hwChannel)
  Driver function Disable Notification.
- boolean Emios_Icu_Ip_GetInputState (uint8 instance, uint8 hwChannel)
  Icu driver function that gets the input state of eMIOS channel.

- eMios_Icu_Ip_LevelType Emios_Icu_Ip_GetInputLevel (uint8 instance, uint8 hwChannel)
  This function returns the actual status of PIN.
- boolean Emios_Icu_Ip_GetOverflow (uint8 instance, uint8 hwChannel)
  Emios_Icu_Ip_GetOverflow.
- void Emios_Icu_Ip_EnableInterrupt (uint8 instance, uint8 hwChannel)
  Emios_Icu_Ip_EnableInterrupt.
- void Emios_Icu_Ip_DisableInterrupt (uint8 instance, uint8 hwChannel)
  Emios_Icu_Ip_DisableInterrupt.
- void Emios_Icu_Ip_IrqHandler (uint8 instance, uint8 channel)
  Icu driver function that handles the interrupt of eMIOS channel.
- Icu_MemMap h void Emios_Icu_Ip_SetUserAccessAllowed (uint32 EmiosBaseAddr)
  Emios_Icu_Ip_SetUserAccessAllowed.

6.2.2 Data Structure Documentation

6.2.2.1 struct eMios_Icu_Ip_DutyCycleType

Structure that contains ICU Duty cycle parameters. It contains the values needed for calculating duty cycles i.e Period time value and active time value.

Definition at line 330 of file Emios_Icu_Ip_Types.h.

Data Fields

- eMios_Icu_ValueType ActiveTime
  Low or High time value.
- eMios_Icu_ValueType PeriodTime
  Period time value.

6.2.2.1.1 Field Documentation

6.2.2.1.1.1 ActiveTime `eMios_Icu_ValueType ActiveTime`

Low or High time value.

Definition at line 332 of file Emios_Icu_Ip_Types.h.

6.2.2.1.1.2 PeriodTime `eMios_Icu_ValueType PeriodTime`

Period time value.

Definition at line 333 of file Emios_Icu_Ip_Types.h.

6.2.2.2 struct eMios_Icu_Ip_ConfigType

eTimer IP specific configuration structure type

Definition at line 373 of file Emios_Icu_Ip_Types.h.

Data Fields

Type	Name	Description
uint8	nNumChannels	Number of eMios channels in the Icu configuration.
const eMios_Icu_Ip_ChannelConfigType(∗	pChannelsConfig)[]	Pointer to the configured channels for eMios.

6.2.2.3 struct eMios_Icu_Ip_ChStateType

This structure is used by the IPL driver for internal logic.

Definition at line 383 of file Emios_Icu_Ip_Types.h.

Data Fields

Type	Name	Description
eMios_Icu_Ip_UCModeType	operationMode	eMios UC mode of operation.
eMios_Icu_Ip_BusType	BusSelected	Master bus selection.
eMios_Icu_Ip_ModeType	channelMode	EMIOS channel mode.
eMios_Icu_Ip_SubModeType	dmaMode	Support DMA or not.
eMios_Icu_Ip_EdgeType	edgeTrigger	Type of edge used for activation.
eMios_Icu_Ip_CallbackType	callback	Calback for other types of measurement.
eMios_Icu_Ip_LogicChStateCbType	logicChStateCallback	Calback for HLD logic channel status changes.
uint16	callbackParam	Logic channel for which callback is executed.
boolean	msWithoutInterrupt	Measurement of ICU signal property without using interrupt.
eMios_Icu_Ip_NotifyType	eMiosChannelNotification	The notification functions for TIME_STAMP or SIGNAL_EDGE_DETECT mode.
eMios_Icu_Ip_NotifyType	eMiosOverflowNotification	The notification functions for TIME_STAMP or SIGNAL_EDGE_DETECT mode.
boolean	notificationEnable	Enables or disables the user notification call.
boolean	channelsInitState	

6.2.3 Macro Definition Documentation

6.2.3.1 EMIOS_ICU_IP_CB_NONE

```
#define EMIOS_ICU_IP_CB_NONE
```

EMIOS Channels defines.

Definition at line 93 of file Emios_Icu_Ip_Types.h.

6.2.4 Types Reference

6.2.4.1 eMios_Icu_Ip_NotifyType

```
typedef void(* eMios_Icu_Ip_NotifyType) (void)
```

The notification functions shall have no parameters and no return value.

Definition at line 323 of file Emios_Icu_Ip_Types.h.

6.2.4.2 eMios_Icu_Ip_CallbackType

```
typedef void(* eMios_Icu_Ip_CallbackType) (uint16 callbackParam1, boolean callbackParam2)
```

HLD Callback type for each channel reporting events or events and overflow .

Definition at line 337 of file Emios_Icu_Ip_Types.h.

6.2.4.3 eMios_Icu_Ip_LogicChStateCbType

```
typedef void(* eMios_Icu_Ip_LogicChStateCbType) (uint16 logicChannel, uint8 mask, boolean set)
```

Callback type for each channel.

Definition at line 340 of file Emios_Icu_Ip_Types.h.

6.2.5 Enum Reference

6.2.5.1 eMios_Icu_Ip_EdgeType

```
enum eMios_Icu_Ip_EdgeType
```

eMIOS_Activation EDGE

Indicates the channel activation type(Rising, Falling, Both Edges or Opposite Edges).

Enumerator

EMIOS_ICU_NO_PIN_CONTROL	No trigger.
EMIOS_ICU_RISING_EDGE	Rising edge trigger.
EMIOS_ICU_FALLING_EDGE	Rising edge trigger.
EMIOS_ICU_BOTH_EDGES	Rising and falling edge trigger.

Definition at line 137 of file Emios_Icu_Ip_Types.h.

6.2.5.2 eMios_Icu_Ip_ModeType

enum eMios_Icu_Ip_ModeType

Operation mode for ICU driver.

Enumerator

EMIOS_ICU_MODE_NO_MEASUREMENT	No measurement mode.
EMIOS_ICU_MODE_SIGNAL_EDGE_DETECT	Signal edge detect measurement mode.
EMIOS_ICU_MODE_SIGNAL_MEASUREMENT	Signal measurement mode.
EMIOS_ICU_MODE_TIMESTAMP	Timestamp measurement mode.
EMIOS_ICU_MODE_EDGE_COUNTER	Edge counter measurement mode.

Definition at line 154 of file Emios_Icu_Ip_Types.h.

6.2.5.3 eMios_Icu_Ip_SubModeType

enum eMios_Icu_Ip_SubModeType

Enable/disable DMA support for timestamp.

Definition at line 169 of file Emios_Icu_Ip_Types.h.

6.2.5.4 eMios_Icu_Ip_MeasStatusType

enum eMios_Icu_Ip_MeasStatusType

Stores the state in which a signal measurement is.

Definition at line 178 of file Emios_Icu_Ip_Types.h.

6.2.5.5 eMios_Icu_Ip_MeasType

enum eMios_Icu_Ip_MeasType

Type of operation for signal measurement.

Enumerator

EMIOS_ICU_NO_MEASUREMENT	No measurement.
EMIOS_ICU_LOW_TIME	The time measurement for OFF period.
EMIOS_ICU_HIGH_TIME	The time measurement for ON period.
EMIOS_ICU_PERIOD_TIME	Period measurement between two consecutive falling/raising edges.
EMIOS_ICU_DUTY_CYCLE	The fraction of active period.

Definition at line 189 of file Emios_Icu_Ip_Types.h.

6.2.5.6 eMios_Icu_Ip_LevelType

enum eMios_Icu_Ip_LevelType

Enumeration used for returning the level of input pin.

Enumerator

EMIOS_ICU_LEVEL_LOW	Low level state.
EMIOS_ICU_LEVEL_HIGH	High level state.

Definition at line 218 of file Emios_Icu_Ip_Types.h.

6.2.5.7 eMios_Icu_Ip_ClockModeType

enum eMios_Icu_Ip_ClockModeType

Definition of prescaler type (Normal or Alternate)

Enumerator

EMIOS_ICU_NORMAL_CLK	Normal prescaler
EMIOS_ICU_ALTERNATE_CLK	Alternate prescaler

Definition at line 228 of file Emios_Icu_Ip_Types.h.

6.2.5.8 eMios_Icu_Ip_BusType

enum eMios_Icu_Ip_BusType

Definition of master bus type.

Enumerator

EMIOS_ICU_BUS_A	Bus A
EMIOS_ICU_BUS_DIVERSE	Bus diverse
EMIOS_ICU_BUS_F	Bus F
EMIOS_ICU_BUS_INTERNAL_COUNTER	Internal counter.

Definition at line 236 of file Emios_Icu_Ip_Types.h.

6.2.5.9 eMios_Icu_Ip_StatusType

enum eMios_Icu_Ip_StatusType

Generic error codes.

Enumerator

EMIOS_IP_STATUS_SUCCESS	Generic operation success status.
EMIOS_IP_STATUS_ERROR	Generic operation failure status.

Definition at line 245 of file Emios_Icu_Ip_Types.h.

6.2.5.10 eMios_Icu_Ip_UCModeType

enum eMios_Icu_Ip_UCModeType

Selection of the signal measurement mode when IcuSignalMeasurementProperty is ICU_DUTY_CYCLE.

Enumerator

EMIOS_ICU_UNINIT	un-initialized.
EMIOS_ICU_SAIC	SAIC mode.
EMIOS_ICU_IPM	IPWM mode.
EMIOS_ICU_IPWM	IPWM mode.

Definition at line 254 of file Emios_Icu_Ip_Types.h.

6.2.5.11 eMios_Icu_Ip_PrescalerType

`enum eMios_Icu_Ip_PrescalerType`

Selects the clock divider value for the UC internal prescaler.

Enumerator

EMIOS_PRESCALER_DIVIDE_1	EMIOS_PRESCALER_DIVIDE_1.
EMIOS_PRESCALER_DIVIDE_2	EMIOS_PRESCALER_DIVIDE_2.
EMIOS_PRESCALER_DIVIDE_3	EMIOS_PRESCALER_DIVIDE_3.
EMIOS_PRESCALER_DIVIDE_4	EMIOS_PRESCALER_DIVIDE_4.
EMIOS_PRESCALER_DIVIDE_5	EMIOS_PRESCALER_DIVIDE_5.
EMIOS_PRESCALER_DIVIDE_6	EMIOS_PRESCALER_DIVIDE_6.
EMIOS_PRESCALER_DIVIDE_7	EMIOS_PRESCALER_DIVIDE_7.
EMIOS_PRESCALER_DIVIDE_8	EMIOS_PRESCALER_DIVIDE_8.
EMIOS_PRESCALER_DIVIDE_9	EMIOS_PRESCALER_DIVIDE_9.
EMIOS_PRESCALER_DIVIDE_10	EMIOS_PRESCALER_DIVIDE_10.
EMIOS_PRESCALER_DIVIDE_11	EMIOS_PRESCALER_DIVIDE_11.
EMIOS_PRESCALER_DIVIDE_12	EMIOS_PRESCALER_DIVIDE_12.
EMIOS_PRESCALER_DIVIDE_13	EMIOS_PRESCALER_DIVIDE_13.
EMIOS_PRESCALER_DIVIDE_14	EMIOS_PRESCALER_DIVIDE_14.
EMIOS_PRESCALER_DIVIDE_15	EMIOS_PRESCALER_DIVIDE_15.
EMIOS_PRESCALER_DIVIDE_16	EMIOS_PRESCALER_DIVIDE_16.

Definition at line 268 of file Emios_Icu_Ip_Types.h.

6.2.5.12 eMios_Icu_Ip_FilterType

`enum eMios_Icu_Ip_FilterType`

Selects the the input filter.

Enumerator

EMIOS_DIGITAL_FILTER_BYPASSED	EMIOS_DIGITAL_FILTER_BYPASSED.
EMIOS_DIGITAL_FILTER_02	EMIOS_DIGITAL_FILTER_02.
EMIOS_DIGITAL_FILTER_04	EMIOS_DIGITAL_FILTER_04.
EMIOS_DIGITAL_FILTER_08	EMIOS_DIGITAL_FILTER_08.
EMIOS_DIGITAL_FILTER_16	EMIOS_DIGITAL_FILTER_16.

Definition at line 305 of file Emios_Icu_Ip_Types.h.

6.2.6 Function Reference

6.2.6.1 Emios_Icu_Ip_Init()

```
eMios_Icu_Ip_StatusType Emios_Icu_Ip_Init (
            uint8 instance,
            const eMios_Icu_Ip_ConfigType * userConfig )
```

Emios_Icu_Ip_Init.

This function is called separately for each EMIOS hw channel corresponding to the configured Icu channels, and:

- Disables the interrupt corresponding to eMIOS channel
- Initializes prescaler value, channel filter, freeze enable, and bus select fields
- Defines on which edge the period starts
- Clears the (pending) interrupt flag corresponding to eMIOS channel
- Resets the UC A register.
- Enables the SAIC mode for eMIOS channels.

Parameters

in	*instance*	- EMIOS instance used.
in	*userConfig*	- pointer to eMios configuration structure

6.2.6.2 Emios_Icu_Ip_Deinit()

```
eMios_Icu_Ip_StatusType Emios_Icu_Ip_Deinit (
            uint8 instance )
```

Emios_Icu_Ip_Deinit.

This function is called separately for each EMIOS hw channel corresponding to the configured Icu channels, and:

- Resets the eMIOS channel control register
- Resets the UC A register.
- Clears the (pending) interrupt flag corresponding to eMIOS channel

Parameters

in	*peMiosIpConfig*	- pointer to eMios configuration structure

6.2.6.3 Emios_Icu_Ip_SetActivation()

```
void Emios_Icu_Ip_SetActivation (
            uint8 instance,
            uint8 hwChannel,
            eMios_Icu_Ip_EdgeType edge )
```

Icu driver function that sets activation condition of eMIOS channel.

This function enables the requested activation condition(rising, falling or both edges) for corresponding eMIOS channels.

Parameters

in	*instance*	- eMIOS module index
in	*hwChannel*	- eMIOS channel index
in	*edge*	- type of edge to be used

6.2.6.4 Emios_Icu_Ip_EnableEdgeDetection()

```
void Emios_Icu_Ip_EnableEdgeDetection (
            uint8 instance,
            uint8 hwChannel )
```

Emios_Icu_Ip_EnableEdgeDetection.

eMIOS IP function that starts the edge detection service for an eMIOS channel

Parameters

in	*instance*	- eMIOS module index
in	*hwChannel*	- eMIOS encoded hardware channel

Returns

void

6.2.6.5 Emios_Icu_Ip_DisableEdgeDetection()

```
void Emios_Icu_Ip_DisableEdgeDetection (
            uint8 instance,
            uint8 hwChannel )
```

Emios_Icu_Ip_DisableEdgeDetection.

eMIOS IP function that stops the edge detection service for an eMIOS channel

Parameters

in	*instance*	- eMIOS module index
in	*hwChannel*	- eMIOS encoded hardware channel

Returns

void

6.2.6.6 Emios_Icu_Ip_EnableNotification()

```
void Emios_Icu_Ip_EnableNotification (
            uint8 instance,
            uint8 hwChannel )
```

Driver function Enable Notification.

Parameters

in	*instance*	Hardware instance used.
in	*hwChannel*	Hardware channel used.

Returns

void

6.2.6.7 Emios_Icu_Ip_DisableNotification()

```
void Emios_Icu_Ip_DisableNotification (
             uint8 instance,
             uint8 hwChannel )
```

Driver function Disable Notification.

Parameters

in	*instance*	Hardware instance used.
in	*hwChannel*	Hardware channel used.

Returns

void

6.2.6.8 Emios_Icu_Ip_GetInputState()

```
boolean Emios_Icu_Ip_GetInputState (
             uint8 instance,
             uint8 hwChannel )
```

Icu driver function that gets the input state of eMIOS channel.

This function:

- Checks if interrupt flags for corresponding eMIOS channel is set then it clears the interrupt flag and returns the value as true.

Parameters

in	*instance*	- eMIOS module index
in	*hwChannel*	- eMIOS encoded hardware channel

Returns

boolean

Return values

true	- if channel is active
false	- if channel is idle

6.2.6.9 Emios_Icu_Ip_GetInputLevel()

```
eMios_Icu_Ip_LevelType Emios_Icu_Ip_GetInputLevel (
            uint8 instance,
            uint8 hwChannel )
```

This function returns the actual status of PIN.

This function returns the actual status o PIN

Parameters

in	*instance*	- eMIOS module index
in	*hwChannel*	- eMIOS encoded hardware channel

Returns

void

6.2.6.10 Emios_Icu_Ip_GetOverflow()

```
boolean Emios_Icu_Ip_GetOverflow (
            uint8 instance,
            uint8 hwChannel )
```

Emios_Icu_Ip_GetOverflow.

eMIOS IP function that get the state of the overflow flag

Parameters

in	*instance*	- eMIOS module index
in	*hwChannel*	- eMIOS encoded hardware channel

Returns

boolean the state of the overflow flag

Return values

true	the overflow flag is set
false	the overflow flag is not set

Returns

void

6.2.6.11 Emios_Icu_Ip_EnableInterrupt()

```
void Emios_Icu_Ip_EnableInterrupt (
            uint8 instance,
            uint8 hwChannel )
```

Emios_Icu_Ip_EnableInterrupt.

This function Clears the pending interrupts of eMIOS channels and enables eMIOS Channel interrupt

Parameters

in	*instance*	- eMIOS module index
in	*hwChannel*	- eMIOS Channel index

6.2.6.12 Emios_Icu_Ip_DisableInterrupt()

```
void Emios_Icu_Ip_DisableInterrupt (
            uint8 instance,
            uint8 hwChannel )
```

Emios_Icu_Ip_DisableInterrupt.

This function disables eMIOS Channel interrupt

Parameters

in	*instance*	- eMIOS module index
in	*hwChannel*	- eMIOS Channel index

6.2.6.13 Emios_Icu_Ip_IrqHandler()

```
void Emios_Icu_Ip_IrqHandler (
            uint8 instance,
            uint8 channel )
```

Icu driver function that handles the interrupt of eMIOS channel.

This function:

- Reads the status register
- Clears the pending interrupt
- Processes interrupt for corresponding eMIOS channel

Parameters

in	*channel*	- eMIOS hardware channel

6.2.6.14 Emios_Icu_Ip_SetUserAccessAllowed()

```
Icu_MemMap h void Emios_Icu_Ip_SetUserAccessAllowed (
             uint32 EmiosBaseAddr )
```

Emios_Icu_Ip_SetUserAccessAllowed.

This function is called externally by OS Application

Parameters

in	*EmiosBaseAddr*	- The base address of Emios.

6.3 SIUL2 IPL

6.3.1 Detailed Description SIUL2 HW module.

SIUL2 IP layer hardware module.

SIUL2 provides control over all electrical pin controls and ports with 16 bits of bidirectional, general-purpose input and output signals. SIUL2 enables you to select the functions and electrical characteristics that appear on external chip pins. It also controls the multiplexing of internal signals from one module to another and controls chip I/O. It supports as many as 32 external interrupts with trigger event configuration.

SIUL2 provides dedicated pad control to general-purpose pads that can be configured as either inputs or outputs. It provides registers for you to read values from GPIO pads configured as inputs and to write values to GPIO pads configured as outputs:

- When configured as output, you can write to an internal register to control the state driven on the associated output pad.
- When configured as **input**, you can detect the state of the associated pad by reading the value from an internal register.
- When configured as input and output, the pad value can be read back to check if the written value appeared on the pad.

Data Structures

- struct Siul2_Icu_Ip_ChannelConfigType
  SIUL2 IP layer channel configuration structure. *More...*
- struct Siul2_Icu_Ip_InstanceConfigType
  SIUL2 IP layer instance configuration structure. *More...*
- struct Siul2_Icu_Ip_State
  SIUL2 IP state structure. *More...*
- struct Siul2_Icu_Ip_ConfigType
  SIUL2 IP layer configuration structure. *More...*

Macros

- #define ICU_START_SEC_CODE
  SIUL2 External Interrupt Channels defines.
- #define ICU_STOP_SEC_CODE
  Siul2_Icu_SetUserAccessAllowed.

Types Reference

- typedef void(* Siul2_Icu_Ip_NotifyType) (void)
  The notification functions shall have no parameters and no return value.
- typedef void(* Siul2_Icu_Ip_CallbackType) (uint16 callbackParam1, boolean callbackParam2)
  Callback signature used in each channel with an active interrupt.

Enum Reference

- enum Siul2_Icu_Ip_ClockModeType
  Definition of prescaler type.
- enum Siul2_Icu_Ip_EdgeType
  Siul2_Icu_ActivationType.
- enum Siul2_Icu_Ip_IrqDmaSelectType
  Siul2_Icu_IrqDmaSelectType.
- enum Siul2_Icu_Ip_StatusType
  SIUL2 IP layer operation status.

Function Reference

- Siul2_Icu_Ip_StatusType Siul2_Icu_Ip_DeInit (uint8 instance)
  Driver function that de-initializes SIUL hardware channel.
- Siul2_Icu_Ip_StatusType Siul2_Icu_Ip_Init (uint8 instance, const Siul2_Icu_Ip_ConfigType *userConfig)
  Driver function that initializes SIUL hardware channel.
- void Siul2_Icu_Ip_SetActivationCondition (uint8 instance, uint8 hwChannel, Siul2_Icu_Ip_EdgeType edge)
- boolean Siul2_Icu_Ip_GetInputState (uint8 instance, uint8 hwChannel)
  ICU driver function that sets activation condition of SIUL2 channel.
- void Siul2_Icu_Ip_EnableInterrupt (uint8 instance, uint8 hwChannel)
  ICU driver function that enables the interrupt of SIUL2 channel.
- void Siul2_Icu_Ip_DisableInterrupt (uint8 instance, uint8 hwChannel)
  ICU driver function that disables the interrupt of SIUL2 channel.
- void Siul2_Icu_Ip_SetClockMode (uint8 instance, Siul2_Icu_Ip_ClockModeType mode)
  Icu driver function used to set the global prescaler of a SIUL2 module.
- void Siul2_Icu_Ip_EnableNotification (uint8 instance, uint8 hwChannel)
  Driver function Enable Notification for timestamp.
- void Siul2_Icu_Ip_DisableNotification (uint8 instance, uint8 hwChannel)
  Driver function Disable Notification for timestamp.

6.3.2 Data Structure Documentation

6.3.2.1 struct Siul2_Icu_Ip_ChannelConfigType

SIUL2 IP layer channel configuration structure.

Definition at line 151 of file Siul2_Icu_Ip_Types.h.

Data Fields

Type	Name	Description
uint8	hwChannel	The interrupt pin index

Data Fields

Type	Name	Description
boolean	digFilterEn	Enables digital filter
uint8	maxFilterCnt	Maximum interrupt filter value
Siul2_Icu_Ip_IrqDmaSelectType	intSel	Switch between DMA and interrupt request
Siul2_Icu_Ip_EdgeType	intEdgeSel	The type of edge event
Siul2_Icu_Ip_CallbackType	callback	Pointer to the callback function.
Siul2_Icu_Ip_NotifyType	Siul2ChannelNotification	The notification functions shall have no parameters and no return value.
uint8	callbackParam	The logic channel for which callback is set.

6.3.2.2 struct Siul2_Icu_Ip_InstanceConfigType

SIUL2 IP layer instance configuration structure.

Definition at line 164 of file Siul2_Icu_Ip_Types.h.

Data Fields

Type	Name	Description
uint8	intFilterClk	Siul2 interrupt clock prescaller digital filter.
uint8	altIntFilterClk	Siul2 interrupt clock prescaller digital filter.

6.3.2.3 struct Siul2_Icu_Ip_State

SIUL2 IP state structure.

This structure is used by the IPL driver for internal logic. The content is populated at initialization time.

Definition at line 175 of file Siul2_Icu_Ip_Types.h.

Data Fields

Type	Name	Description
boolean	chInit	Initialization state.
Siul2_Icu_Ip_CallbackType	callback	Pointer to the callback function.
Siul2_Icu_Ip_NotifyType	Siul2ChannelNotification	The notification functions for SIGNAL_EDGE_DETECT mode.
uint16	callbackParam	The logic channel for which callback is set.
boolean	notificationEnable	State of the notification.

6.3.2.4 struct Siul2_Icu_Ip_ConfigType

SIUL2 IP layer configuration structure.

Definition at line 186 of file Siul2_Icu_Ip_Types.h.

Data Fields

Type	Name	Description
uint8	numChannels	Number of channels in the configuration.
const Siul2_Icu_Ip_InstanceConfigType *	pInstanceConfig	Pointer to the instance configuration.
const Siul2_Icu_Ip_ChannelConfigType(*	pChannelsConfig)[]	Pointer to the channels configuration.

6.3.3 Macro Definition Documentation

6.3.3.1 ICU_START_SEC_CODE

```
#define ICU_START_SEC_CODE
```

SIUL2 External Interrupt Channels defines.

Definition at line 173 of file Siul2_Icu_Ip_Irq.h.

6.3.3.2 ICU_STOP_SEC_CODE

```
#define ICU_STOP_SEC_CODE
```

Siul2_Icu_SetUserAccessAllowed.

This function is called externally by OS Application

Parameters

in	*siul2BaseAddr*	- The base address of Siul2 module.

Definition at line 113 of file Siul2_Icu_Ip_TrustedFunctions.h.

6.3.4 Types Reference

6.3.4.1 Siul2_Icu_Ip_NotifyType

```
typedef void(* Siul2_Icu_Ip_NotifyType) (void)
```

The notification functions shall have no parameters and no return value.

Definition at line 146 of file Siul2_Icu_Ip_Types.h.

6.3.4.2 Siul2_Icu_Ip_CallbackType

```
typedef void(* Siul2_Icu_Ip_CallbackType) (uint16 callbackParam1, boolean callbackParam2)
```

Callback signature used in each channel with an active interrupt.

Definition at line 148 of file Siul2_Icu_Ip_Types.h.

6.3.5 Enum Reference

6.3.5.1 Siul2_Icu_Ip_ClockModeType

```
enum Siul2_Icu_Ip_ClockModeType
```

Definition of prescaler type.

Definition of prescaler type (Normal or Alternate)

Enumerator

SIUL2_ICU_NORMAL_CLK	Normal prescaler
SIUL2_ICU_ALTERNATE_CLK	Alternate prescaler

Definition at line 106 of file Siul2_Icu_Ip_Types.h.

6.3.5.2 Siul2_Icu_Ip_EdgeType

```
enum Siul2_Icu_Ip_EdgeType
```

Siul2_Icu_ActivationType.

This indicates the activation type SIUL2 channel (Rising, Falling or Both)

Enumerator

SIUL2_ICU_DISABLE	Interrupt disable.
SIUL2_ICU_RISING_EDGE	Interrupt on rising edge.
SIUL2_ICU_FALLING_EDGE	Interrupt on falling edge.
SIUL2_ICU_BOTH_EDGES	Interrupt on either edge.

Definition at line 117 of file Siul2_Icu_Ip_Types.h.

6.3.5.3 Siul2_Icu_Ip_IrqDmaSelectType

enum Siul2_Icu_Ip_IrqDmaSelectType

Siul2_Icu_IrqDmaSelectType.

This indicates the type of request DMA or IRQ when activation edge is detected

Enumerator

SIUL2_ICU_IRQ	Generate an interrupt request.
SIUL2_ICU_DMA	Generate an DMA request

Definition at line 129 of file Siul2_Icu_Ip_Types.h.

6.3.5.4 Siul2_Icu_Ip_StatusType

enum Siul2_Icu_Ip_StatusType

SIUL2 IP layer operation status.

Enumerator

SIUL2_ICU_IP_SUCCESS	Status for success operation return.
SIUL2_ICU_IP_ERROR	General error return status.

Definition at line 136 of file Siul2_Icu_Ip_Types.h.

6.3.6 Function Reference

6.3.6.1 Siul2_Icu_Ip_DeInit()

```
Siul2_Icu_Ip_StatusType Siul2_Icu_Ip_DeInit (
            uint8 instance )
```

Driver function that de-initializes SIUL hardware channel.

This function:

- Restore to reset values SIUL2 registers used on init.

Parameters

in	*instance*	- Instance number used

Returns

Siul2_Icu_Ip_StatusType - The status of DeInit

6.3.6.2 Siul2_Icu_Ip_Init()

```
Siul2_Icu_Ip_StatusType Siul2_Icu_Ip_Init (
            uint8 instance,
            const Siul2_Icu_Ip_ConfigType * userConfig )
```

Driver function that initializes SIUL hardware channel.

This function:

- Disables interrupt.
- Sets Interrupt filter enable register
- Sets Interrupt Filter Clock Prescaler Register
- Sets Activation Condition

Parameters

in	*instance*	Hardware instance of SIUL2 used.
in	*userConfig*	Instance configuration.

Returns

Siul2_Icu_Ip_StatusType - The status of Init

6.3.6.3 Siul2_Icu_Ip_SetActivationCondition()

```
void Siul2_Icu_Ip_SetActivationCondition (
            uint8 instance,
            uint8 hwChannel,
            Siul2_Icu_Ip_EdgeType edge )
```

This function enables the requested activation condition(rising, falling or both edges) for corresponding SIUL2 channels.

Parameters

in	*instance*	Hardware instance of SIUL2 used.
in	*hwChannel*	Hardware channel of SIUL2 used.
in	*edge*	Edge activation type used.

6.3.6.4 Siul2_Icu_Ip_GetInputState()

```
boolean Siul2_Icu_Ip_GetInputState (
            uint8 instance,
            uint8 hwChannel )
```

ICU driver function that sets activation condition of SIUL2 channel.

Parameters

in	*instance*	Hardware instance of SIUL2 used.
in	*hwChannel*	Hardware channel of SIUL2 used.

Returns

boolean Input state.

6.3.6.5 Siul2_Icu_Ip_EnableInterrupt()

```
void Siul2_Icu_Ip_EnableInterrupt (
            uint8 instance,
            uint8 hwChannel )
```

ICU driver function that enables the interrupt of SIUL2 channel.

This function enables SIUL2 Channel Interrupt.

Parameters

in	*instance*	Hardware instance of SIUL2 used.
in	*hwChannel*	Hardware channel of SIUL2 used.

Returns

void

6.3.6.6 Siul2_Icu_Ip_DisableInterrupt()

```
void Siul2_Icu_Ip_DisableInterrupt (
            uint8 instance,
            uint8 hwChannel )
```

ICU driver function that disables the interrupt of SIUL2 channel.

This function disables SIUL2 Channel Interrupt.

Parameters

in	*instance*	Hardware instance of SIUL2 used.
in	*hwChannel*	Hardware channel of SIUL2 used.

Returns

void

6.3.6.7 Siul2_Icu_Ip_SetClockMode()

```
void Siul2_Icu_Ip_SetClockMode (
            uint8 instance,
            Siul2_Icu_Ip_ClockModeType mode )
```

Icu driver function used to set the global prescaler of a SIUL2 module.

This function:

- Sets IFCPR register with a prescaler value

Parameters

in	*instance*	Hardware instance of SIUL2 used.
in	*mode*	Global prescaler for the SIUL2 module.

Returns

void

6.3.6.8 Siul2_Icu_Ip_EnableNotification()

```
void Siul2_Icu_Ip_EnableNotification (
            uint8 instance,
            uint8 hwChannel )
```

Driver function Enable Notification for timestamp.

Parameters

in	*instance*	Hardware instance of FTM used.
in	*hwChannel*	Hardware channel of FTM used.

Returns

void

6.3.6.9 Siul2_Icu_Ip_DisableNotification()

```
void Siul2_Icu_Ip_DisableNotification (
            uint8 instance,
            uint8 hwChannel )
```

Driver function Disable Notification for timestamp.

Parameters

in	*instance*	Hardware instance of FTM used.
in	*hwChannel*	Hardware channel of FTM used.

Returns

void

6.4 CMP IPL

6.4.1 Detailed Description CMP HW module.

The low power comparator (LPCMP) module provides a circuit for comparing two analog input voltages. It comprises a comparator (CMP), a DAC and an analog mux (ANMUX). The CMP circuit is designed to operate across the full range of the supply voltage, known as rail-to-rail operation. The DAC is a 256-tap resistor ladder network that provides a selectable voltage reference for applications requiring a voltage reference. The 256-tap resistor ladder network divides the supply reference Vin into 256 voltage levels. A 8-bit digital signal input selects the output voltage level, which varies from Vin to Vin/256. Vin can be selected from two voltage sources, vrefh0 and vrefh1. See the chip-specific LPCMP information for the source of vrefh0 and vrefh1.

The Analog MUX (ANMUX) provides a circuit for selecting an analog input signal from eight channels. One channel is provided by the DAC. Refer to the chip-specific LPCMP information section for details on which device resources are connected to other channels. The mux circuit is designed to operate across the full range of the supply voltage.

6.5 Icu Driver

6.5.1 Detailed Description

Data Structures

- struct Icu_ChannelConfigType

  Structure that contains ICU channel configuration. More...
- struct Icu_ConfigType

  This type contains initialization data. More...

Types Reference

- typedef uint8 Icu_ChannelStateType

  ICU Channel state type.
- typedef uint16 Icu_ChannelType

  This gives the numeric ID (hardware channel number) of an ICU channel.
- typedef Icu_TimerRegisterWidthType Icu_ValueType

  Type for saving the timer register width value.
- typedef uint16 Icu_MeasurementSubModeType

  Type for saving the ICU measurement submode type.
- typedef void(∗ Icu_NotifyType) (void)

  The notification functions shall have no parameters and no return value.

Enum Reference

- enum Icu_ModeType

  Allow enabling or disabling of all interrupts which are not required for the ECU wakeup.
- enum Icu_InputStateType

  Input state of an ICU channel.
- enum Icu_MeasurementModeType

  Definition of the measurement mode type.
- enum Icu_ActivationType

  Definition of the type of activation of an ICU channel.
- enum Icu_LevelType

  Return the status of the pin.
- enum Icu_SelectPrescalerType

  Definition of prescaler type.

Function Reference

- void Icu_Init (const Icu_ConfigType *ConfigPtr)
  This function initializes the driver.
- void Icu_DeInit (void)
  This function de-initializes the ICU module.
- void Icu_SetActivationCondition (Icu_ChannelType Channel, Icu_ActivationType Activation)
  This function sets the activation-edge for the given channel.
- void Icu_DisableNotification (Icu_ChannelType Channel)
  This function disables the notification of a channel.
- void Icu_EnableNotification (Icu_ChannelType Channel)
  This function enables the notification on the given channel.
- Icu_InputStateType Icu_GetInputState (Icu_ChannelType Channel)
  This function returns the status of the ICU input.
- void Icu_EnableEdgeDetection (Icu_ChannelType Channel)
  This function enables or re-enables the detection of edges of the given channel.
- void Icu_DisableEdgeDetection (Icu_ChannelType Channel)
  This function disables the detection of edges of the given channel.
- void Icu_SetClockMode (Icu_SelectPrescalerType selectPrescaler)
  This function sets all channels prescalers based on the input mode.
- Icu_LevelType Icu_GetInputLevel (Icu_ChannelType Channel)
  This function returns the actual status of PIN.
- void Icu_ReportWakeupAndOverflow (uint16 Channel, boolean bOverflow)
  This function reports the wakeup and overflow events, if available.
- void Icu_ReportEvents (uint16 Channel, boolean bOverflow)
  This function reports the wakeup event, overflow event and notification, if available.
- void Icu_LogicChStateCallback (uint16 logicChannel, uint8 mask, boolean set)
  Signature of change logic channel state callback function.

Variables

- const Icu_ConfigType * Icu_pCfgPtr [(1U)]
  Pointer initialized during init with the address of the received configuration structure.
- Icu_ModeType Icu_CurrentMode
  Saves the current Icu mode.
- volatile Icu_ChannelStateType Icu_aChannelState [((Icu_ChannelType) 3U)]
  Stores actual state and configuration of ICU Channels.

6.5.2 Data Structure Documentation

6.5.2.1 struct Icu_ChannelConfigType

Structure that contains ICU channel configuration.

It contains the information like Icu Channel Mode, Channel Notification function, overflow Notification function.

Definition at line 557 of file Icu.h.

Data Fields

- boolean Icu_WakeupCapabile

  Channel wakeup capability enable.
- Icu_ActivationType Icu_ActivEdge

  RISING_EDGE, FALLING_EDGE or BOTH_EDGES for EDGE_COUNTER.
- Icu_MeasurementModeType Icu_ChannelMode

  EDGE_DETECT, TIME_STAMP, SIGNAL_MEASUREMENT or EDGE_COUNTER.
- Icu_MeasurementSubModeType Icu_ChannelProperty

  CIRCULAR_BUFFER or LINEAR_BUFFER for TIME_STAMP, DUTY_CYCLE, HIGH_TIME, LOW_TIME or PERIOD_TIME for SIGNAL_MEASUREMENT and RISING_EDGE, FALLING_EDGE or BOTH_EDGES for EDGE_COUNTER.
- Icu_NotifyType Icu_ChannelNotification

  Icu Channel Notification function for TIME_STAMP or EDGE_COUNTER mode.
- const Icu_Ipw_ChannelConfigType * Icu_IpwChannelConfigPtr

  Pointer to the ipw channel pointer configuration.

6.5.2.1.1 Field Documentation

6.5.2.1.1.1 Icu_WakeupCapabile `boolean Icu_WakeupCapabile`

Channel wakeup capability enable.

Definition at line 560 of file Icu.h.

6.5.2.1.1.2 Icu_ActivEdge `Icu_ActivationType Icu_ActivEdge`

RISING_EDGE, FALLING_EDGE or BOTH_EDGES for EDGE_COUNTER.

Definition at line 562 of file Icu.h.

6.5.2.1.1.3 Icu_ChannelMode `Icu_MeasurementModeType Icu_ChannelMode`

EDGE_DETECT, TIME_STAMP, SIGNAL_MEASUREMENT or EDGE_COUNTER.

Definition at line 564 of file Icu.h.

6.5.2.1.1.4 Icu_ChannelProperty Icu_MeasurementSubModeType Icu_ChannelProperty

CIRCULAR_BUFFER or LINEAR_BUFFER for TIME_STAMP, DUTY_CYCLE, HIGH_TIME, LOW_TIME or PERIOD_TIME for SIGNAL_MEASUREMENT and RISING_EDGE, FALLING_EDGE or BOTH_EDGES for EDGE_COUNTER.

Definition at line 568 of file Icu.h.

6.5.2.1.1.5 Icu_ChannelNotification Icu_NotifyType Icu_ChannelNotification

Icu Channel Notification function for TIME_STAMP or EDGE_COUNTER mode.

Definition at line 570 of file Icu.h.

6.5.2.1.1.6 Icu_IpwChannelConfigPtr const Icu_Ipw_ChannelConfigType* Icu_IpwChannelConfigPtr

Pointer to the ipw channel pointer configuration.

Definition at line 583 of file Icu.h.

6.5.2.2 struct Icu_ConfigType

This type contains initialization data.

he notification functions shall be configurable as function pointers within the initialization data structure (Icu_ConfigType). This type of the external data structure shall contain the initialization data for the ICU driver. It shall contain:

- Wakeup Module Info (in case the wakeup-capability is true)
- ICU dependent properties for used HW units
- Clock source with optional prescaler (if provided by HW)

Definition at line 597 of file Icu.h.

Data Fields

- uint8 nNumChannels
  The number of configured logical channels.
- const Icu_ChannelConfigType(* Icu_ChannelConfigPtr)[]
  Pointer to the list of Icu configured channels.
- uint8 nNumInstances
  The number of IP instances configured.
- const Icu_Ipw_IpConfigType(* Icu_IpConfigPtr)[]
  Pointer to the list of Icu configured channels.
- const uint8(* Icu_IndexChannelMap)[]
  channel index in each partition map table
- uint8 u32CoreId
  Core index.

6.5.2.2.1 Field Documentation

6.5.2.2.1.1 nNumChannels `uint8 nNumChannels`

The number of configured logical channels.

Definition at line 600 of file Icu.h.

6.5.2.2.1.2 Icu_ChannelConfigPtr `const Icu_ChannelConfigType(* Icu_ChannelConfigPtr)[]`

Pointer to the list of Icu configured channels.

Definition at line 603 of file Icu.h.

6.5.2.2.1.3 nNumInstances `uint8 nNumInstances`

The number of IP instances configured.

Definition at line 606 of file Icu.h.

6.5.2.2.1.4 Icu_IpConfigPtr `const Icu_Ipw_IpConfigType(* Icu_IpConfigPtr)[]`

Pointer to the list of Icu configured channels.

Definition at line 609 of file Icu.h.

6.5.2.2.1.5 Icu_IndexChannelMap `const uint8(* Icu_IndexChannelMap)[]`

channel index in each partition map table

Definition at line 612 of file Icu.h.

6.5.2.2.1.6 u32CoreId `uint8 u32CoreId`

Core index.

Definition at line 615 of file Icu.h.

6.5.3 Types Reference

6.5.3.1 Icu_ChannelStateType

```
typedef uint8 Icu_ChannelStateType
```

ICU Channel state type.

Definition at line 219 of file Icu_Types.h.

6.5.3.2 Icu_ChannelType

```
typedef uint16 Icu_ChannelType
```

This gives the numeric ID (hardware channel number) of an ICU channel.

Definition at line 224 of file Icu_Types.h.

6.5.3.3 Icu_ValueType

```
typedef Icu_TimerRegisterWidthType Icu_ValueType
```

Type for saving the timer register width value.

Definition at line 229 of file Icu_Types.h.

6.5.3.4 Icu_MeasurementSubModeType

```
typedef uint16 Icu_MeasurementSubModeType
```

Type for saving the ICU measurement submode type.

Definition at line 257 of file Icu_Types.h.

6.5.3.5 Icu_NotifyType

```
typedef void(* Icu_NotifyType) (void)
```

The notification functions shall have no parameters and no return value.

Definition at line 262 of file Icu_Types.h.

6.5.4 Enum Reference

6.5.4.1 Icu_ModeType

```
enum Icu_ModeType
```

Allow enabling or disabling of all interrupts which are not required for the ECU wakeup.

Enumerator

ICU_MODE_NORMAL	Normal operation, all used interrupts are enabled according to the notification requests.
ICU_MODE_SLEEP	Reduced power operation. In sleep mode only those notifications are available which are configured as wakeup capable.

Definition at line 102 of file Icu_Types.h.

6.5.4.2 Icu_InputStateType

`enum Icu_InputStateType`

Input state of an ICU channel.

Enumerator

ICU_ACTIVE	An activation edge has been detected.
ICU_IDLE	No activation edge has been detected since the last call of Icu_GetInputState() or Icu_Init().

Definition at line 115 of file Icu_Types.h.

6.5.4.3 Icu_MeasurementModeType

`enum Icu_MeasurementModeType`

Definition of the measurement mode type.

Enumerator

ICU_MODE_SIGNAL_EDGE_DETECT	Mode for detecting edges.
ICU_MODE_SIGNAL_MEASUREMENT	Mode for measuring different times between various configurable edges.
ICU_MODE_TIMESTAMP	Mode for capturing timer values on configurable edges.
ICU_MODE_EDGE_COUNTER	Mode for counting edges on configurable edges.

Definition at line 127 of file Icu_Types.h.

6.5.4.4 Icu_ActivationType

enum Icu_ActivationType

Definition of the type of activation of an ICU channel.

Enumerator

ICU_RISING_EDGE	An appropriate action shall be executed when a rising edge occurs on the ICU input signal.
ICU_FALLING_EDGE	An appropriate action shall be executed when a falling edge occurs on the ICU input signal.
ICU_BOTH_EDGES	An appropriate action shall be executed when either a rising or falling edge occur on the ICU input signal.

Definition at line 174 of file Icu_Types.h.

6.5.4.5 Icu_LevelType

enum Icu_LevelType

Return the status of the pin.

Enumeration of to check the status of pin.

Enumerator

ICU_LEVEL_LOW	Default Input PIN Status.
ICU_LEVEL_HIGH	As Icu_GetInputState do not give the Actual PIN status user can call the Non Autosar API Icu_GetInputLevel to get the Actual status of PIN.

Definition at line 190 of file Icu_Types.h.

6.5.4.6 Icu_SelectPrescalerType

enum Icu_SelectPrescalerType

Definition of prescaler type.

Enumerator

ICU_NORMAL_CLOCK_MODE	Default channel prescaler.
ICU_ALTERNATE_CLOCK_MODE	Alternate channel prescaler mode.

Definition at line 207 of file Icu_Types.h.

6.5.5 Function Reference

6.5.5.1 Icu_Init()

```
void Icu_Init (
            const Icu_ConfigType * ConfigPtr )
```

This function initializes the driver.

This service is a non reentrant function used for driver initialization. The Initialization function shall initialize all relevant registers of the configured hardware with the values of the structure referenced by the parameter Config↩ Ptr. If the hardware allows for only one usage of the register, the driver module implementing that functionality is responsible for initializing the register. The initialization function of this module shall always have a pointer as a parameter, even though for Variant PC no configuration set shall be given. Instead a NULL pointer shall be passed to the initialization function. The Icu module environment shall not call Icu_Init during a running operation (e. g. timestamp measurement or edge counting).

Parameters

in	*ConfigPtr*	Pointer to a selected configuration structure.

Returns

void

6.5.5.2 Icu_DeInit()

```
void Icu_DeInit (
            void  )
```

This function de-initializes the ICU module.

This service is a Non reentrant function used for ICU De-Initialization After the call of this service, the state of the peripherals used by configuration shall be the same as after power on reset. Values of registers which are not writable are excluded. This service shall disable all used interrupts and notifications. The Icu module environment shall not call Icu_DeInit during a running operation (e. g. timestamp measurement or edge counting)

Returns

void

Precondition

Icu_Init must be called before.

6.5.5.3 Icu_SetActivationCondition()

```
void Icu_SetActivationCondition (
            Icu_ChannelType Channel,
            Icu_ActivationType Activation )
```

This function sets the activation-edge for the given channel.

This service is reentrant and shall set the activation-edge according to Activation parameter for the given channel. This service shall support channels which are configured for the following Icu_MeasurementMode:

- ICU_MODE_SIGNAL_EDGE_DETECT
- ICU_MODE_TIMESTAMP
- ICU_MODE_EDGE_COUNTER

Parameters

in	*Channel*	Logical number of the ICU channel
in	*Activation*	Type of activation.

Returns

void

Precondition

Icu_Init must be called before. The channel must be properly configured (ICU_MODE_SIGNAL_EDGE↩ _DETECT, ICU_MODE_TIMESTAMP, ICU_MODE_EDGE_COUNTER).

6.5.5.4 Icu_DisableNotification()

```
void Icu_DisableNotification (
            Icu_ChannelType Channel )
```

This function disables the notification of a channel.

This function is reentrant and disables the notification of a channel.

Parameters

in	*Channel*	Logical number of the ICU channel

Returns

void

Precondition

Icu_Init must be called before.

6.5.5.5 Icu_EnableNotification()

```
void Icu_EnableNotification (
             Icu_ChannelType Channel )
```

This function enables the notification on the given channel.

This function is reentrant and enables the notification on the given channel. The notification will be reported only when the channel measurement property is enabled or started

Parameters

in	*Channel*	Logical number of the ICU channel

Returns

void

Precondition

Icu_Init must be called before.

6.5.5.6 Icu_GetInputState()

```
Icu_InputStateType Icu_GetInputState (
             Icu_ChannelType Channel )
```

This function returns the status of the ICU input.

This service is reentrant shall return the status of the ICU input. Only channels which are configured for the following Icu_MeasurementMode shall be supported:

- ICU_MODE_SIGNAL_EDGE_DETECT,
- ICU_MODE_SIGNAL_MEASUREMENT.

Parameters

in	*Channel*	Logical number of the ICU channel

Returns

Icu_InputStateType

Return values

ICU_ACTIVE	An activation edge has been detected
ICU_IDLE	No activation edge has been detected since the last call of Icu_GetInputState() or Icu_Init().

Precondition

Icu_Init must be called before.

6.5.5.7 Icu_EnableEdgeDetection()

```
void Icu_EnableEdgeDetection (
            Icu_ChannelType Channel )
```

This function enables or re-enables the detection of edges of the given channel.

This function is reentrant enables or re-enables the detection of edges of the given channel.

Parameters

in	*Channel*	Logical number of the ICU channel

Returns

void

Precondition

Icu_Init must be called before. The channel must be configured in Measurement Mode Edge Counter

6.5.5.8 Icu_DisableEdgeDetection()

```
void Icu_DisableEdgeDetection (
            Icu_ChannelType Channel )
```

This function disables the detection of edges of the given channel.

This function is reentrant and disables the detection of edges of the given channel.

Parameters

in	*Channel*	Logical number of the ICU channel

Returns

void

Precondition

Icu_Init must be called before. The channel must be configured in Measurement Mode Edge Detection.

6.5.5.9 Icu_SetClockMode()

```
void Icu_SetClockMode (
            Icu_SelectPrescalerType selectPrescaler )
```

This function sets all channels prescalers based on the input mode.

Parameters

selectPrescaler	Select the used prescaler: prescaler/alternatePresclaer.

Returns

void

Precondition

Icu_Init must be called before.

6.5.5.10 Icu_GetInputLevel()

```
Icu_LevelType Icu_GetInputLevel (
            Icu_ChannelType Channel )
```

This function returns the actual status of PIN.

This function returns the actual status of PIN.

Parameters

in	*Channel*	Logical number of the ICU channel

Returns

Icu_LevelType

Precondition

Icu_Init must be called before.

6.5.5.11 Icu_ReportWakeupAndOverflow()

```
void Icu_ReportWakeupAndOverflow (
            uint16 Channel,
            boolean bOverflow )
```

This function reports the wakeup and overflow events, if available.

This function reports the wakeup and overflow events, if available. Called from hardware interrupt routine and route to user overflow handler

Parameters

in	*Channel*	Hardware number identifier of the ICU channel
in	*bOverflow*	Parameter that indicates the source of report is an overflow

Returns

void

Precondition

Icu_Init must be called before.

6.5.5.12 Icu_ReportEvents()

```
void Icu_ReportEvents (
            uint16 Channel,
            boolean bOverflow )
```

This function reports the wakeup event, overflow event and notification, if available.

This function reports the wakeup event, overflow event and notification, if available

Parameters

in	*Channel*	Harware number identifier of the ICU channel
in	*overflow*	Parameter that indicates the source of report is an overflow

Returns

void

Precondition

Icu_Init must be called before.

6.5.5.13 Icu_LogicChStateCallback()

```
void Icu_LogicChStateCallback (
            uint16 logicChannel,
            uint8 mask,
            boolean set )
```

Signature of change logic channel state callback function.

Parameters

logicChannel	Logical number of the ICU channel
mask	Bit mark
set	Set value

6.5.6 Variable Documentation

6.5.6.1 Icu_pCfgPtr

```
const Icu_ConfigType* Icu_pCfgPtr[(1U)]  [extern]
```

Pointer initialized during init with the address of the received configuration structure.

Will be used by all functions to access the configuration data.

6.5.6.2 Icu_CurrentMode

```
Icu_ModeType Icu_CurrentMode  [extern]
```

Saves the current Icu mode.

6.5.6.3 Icu_aChannelState

```
volatile Icu_ChannelStateType Icu_aChannelState[((Icu_ChannelType) 3U)]  [extern]
```

Stores actual state and configuration of ICU Channels.

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

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

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