User Manual

for S32K3 OCU Driver

Document Number: UM34OCUASRR21-11 Rev0000R3.0.0 Rev. 1.0

Chapter 1

Revision History

Revision	Date	Author	Description
1.0	31.03.2023	NXP RTD Team	Prepared for release RTD S32K3 3.0.0

Chapter 2

Introduction

This User Manual describes NXP Semiconductor AUTOSAR Output Compare Unit(Ocu) for S32K3XX. AUTOSAR Ocu driver configuration parameters and deviations from the specification are described in Driver chapter of this document. AUTOSAR Ocu driver requirements and APIs are described in the AUTOSAR Ocu 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
AUTOSAR	Automotive Open System Architecture
API	Application Programming Interface
ARTD	Automotive Real Time Drivers
ASR	AUTOSAR
BSW	Basic Software
DEM	Diagnostic Event Manager
DET	Development Error Tracer
DIO	Digital Input Output
DMA	Direct Memory Access
ECU	Electronic Control Unit
ECUC	ECU Configuration
EcuM	ECU state Manager
eMIOS	Enhanced Modular IO Subsystem
FTM	FlexTimer
GUI	Graphical User Interface
HLD	High Level Driver
HW	Hardware
ICU	Input Capture Unit
IP	Intellectual Property, referred as a hardware design block
IPL	IP Layer
IPW	IP Wrapper Layer
ISR	Interrupt Service Routine
MCAL	Microconroller Abstraction Layer
MCU	Micro Controller Unit
N/A	Not Applicable
OCU	Output Compare Unit
OS	Operating System
OSIF	OS Interface
PB Variant	Post Build Variant
PC Variant	Pre Compile Variant
PWM	Pulse Width Modulation
RAM	Random Access Memory
ROM	Read-only Memory
SCI	Serial Communication Interface
SoC	System on Chip
SPI	Serial Peripheral Interface
SWS	Software Specification
VLE	Variable Length Encoding
VSMD	Vendor Specific Module Definition
XML	Extensible Markup Language

2.5 Reference List

#	Title	Version
1	Specification of OCU 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
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 Driver Software Specification document (See Table Reference List).

For CDD: _driver> is a Complex Device Driver (CDD), so there are no AUTOSAR requirements regarding this module.

It has vendor-specific requirements and implementation.

3.2 Driver Design Summary

This driver provides services for initializing and controlling the Output Compare Unit module. Ocu module is used to run a counter until a predefined value is reached. When a compare-match occurs, a hardware interrupt and/or a state change of an output pin will be triggered. Ocu configuration consists of logic channels exclusively bound to one of the available hardware channels. The Ocu logic channel contains definitions for the initial value of the counter, the maximum counter value (threshold), a notification callback to a user defined function and the behavior of the hardware channel designated pin. A clock source and a prescaler value (optionally a second prescaler) must be set for the hardware channel.

The Ocu 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

Derivatives	OCU module	OCU channel available
Ocu_s32k311_lqfp48, Ocu_s32k311_mqfp100	EMIOS0, EMIOS1	Channel 0-23
Ocu_s32k312_mqfp100, Ocu_s32k312_mqfp172	EMIOS0, EMIOS1	Channel 0-23
Ocu_s32k314_mapbga257, Ocu_s32k314_mqfp172	EMIOS0, EMIOS1, EMIOS2	Channel 0-23
Ocu_s32k322_mqfp100, Ocu_s32k322_mqfp172	EMIOS0, EMIOS1	Channel 0-23
Ocu_s32k324_mapbga257, Ocu_s32k324_mqfp172	EMIOS0, EMIOS1, EMIOS2	Channel 0-23
Ocu_s32k328_mapbga289, Ocu_s32k328_mqfp172	EMIOS0, EMIOS1, EMIOS2	Channel 0-23
Ocu_s32k338_mapbga289, Ocu_s32k338_mqfp172	EMIOS0, EMIOS1, EMIOS2	Channel 0-23
Ocu_s32k341_mqfp100, Ocu_s32k341_mqfp172	EMIOS0, EMIOS1	Channel 0-23
Ocu_s32k342_mqfp100, Ocu_s32k342_mqfp172	EMIOS0, EMIOS1	Channel 0-23
Ocu_s32k344_mapbga257, Ocu_s32k344_mqfp172	EMIOS0, EMIOS1, EMIOS2	Channel 0-23
Ocu_s32k348_mapbga289, Ocu_s32k348_mqfp172	EMIOS0, EMIOS1, EMIOS2	Channel 0-23
Ocu_s32k358_mapbga289, Ocu_s32k358_mqfp172	EMIOS0, EMIOS1, EMIOS2	Channel 0-23
Ocu_s32k388_mapbga289, Ocu_s32k388_mqfp172	EMIOS0, EMIOS1, EMIOS2	Channel 0-23
Ocu_s32k396_mapbga289	EMIOS0	Channel 0-23
Ocu_s32m276_lqfp64	EMIOS0, EMIOS1	Channel 0-23

3.4 Deviations from Requirements

The driver deviates from the AUTOSAR Ocu Driver software specification in some places. The table identifies the AUTOSAR requirements that are not fully implemented, implemented differently, not available, not testable or out of scope for the Ocu 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, not available, not testable or out of scope for the driver.

Requirement	Status	Description	Notes
SWS_Ocu_00020	N/S	The detection of production errors cannot be switched off.	DEM errors are not used
SWS_Ocu_00023	N/S	Each variable that shall be accessible by AUTOSAR Debugging, shall be defined as global variable.	AUTOSAR Debugging concept is not implemented. In par with PR-MCAL-3330
SWS_Ocu_00024	N/S	All type definitions of variables which shall be debugged shall be accessible by the header file Ocu.h.	AUTOSAR Debugging concept is not implemented. In par with PR-MCAL-3330

Requirement	Status	Description	Notes
SWS_Ocu_00025	N/S	The declaration of variables in the header file shall be such that it is possible to calculate the size of the variables by C-"sizeof".	AUTOSAR Debugging concept is not implemented. In par with PR-MCAL-3330
SWS_Ocu_00026	N/S	Variables available for debugging shall be described in the respective OCU driver Description.	AUTOSAR Debugging concept is not implemented. In par with PR-MCAL-3330
SWS_Ocu_00125	N/S	If source code for caller and callee of Ocu_GetVersionInfo is available; the OCU driver should realize Ocu_Get↩ VersionInfo as a macro, defined in the module's header file.	Ocu_GetVersionInfo is defined as a function as a common approach in all MCAL drivers.
SWS_Ocu_00155	N/S	This parameter is used to allow the OCU channel to trigger an ADC channel upon compare match, if this is supported by hardware. The value of the parameter represents the ADC physical channel to trigger	Trigger for ADC has not supported in OCU driver
SWS_Ocu_00156	N/S	This parameter is used to allow the OCU channel to trigger a DMA channel upon compare match, if this is supported by hardware. The value of the parameter represents the DMA physical channel to trigger	Hardware trigger for DMA has not supported in OCU driver
ECUC_Ocu_00168	N/S	Maps the OCU kernel to zero or one ECUC partitions to assign the driver kernel to a certain core. The EC↩ UC partition referenced is a subset of the ECUC partitions where the O↩ CU driver is mapped to.Tags: atp.↩ Status=draft	Type IV Autosar multicore not implemented for current module (AAI-445), therefore OcuKernelEcucPartitionRef is not supported

3.5 Driver Limitations

- Function Ocu_SetClockMode/Emios_Ocu_Ip_SetClockMode does not effect to channel which using master bus counter.

3.6 Driver usage and configuration tips

The Mcl driver is a dependency for Ocu on the S32K3 platform because it provides common support for eMios IP. To configure any eMios hardware channels for the Ocu driver:

- Include the Mcl module to the configuration project.
- Enable the *EnableEmiosCommonSupport* parameter in *Mcl/General/MclEmiosCommon* container.

- Configure the desired eMios instance and enable EmiosMclEnableGlobalTimeBase parameter in *Mcl/Mcl↩ Config/EmiosCommon* container.

- Optionally, to use one of the local or global available counter busses (A, B, C, D, E) for an eMios channel, it must be previously configured in *Mcl/MclConfig/EmiosCommon/EmiosMasterBuses* container. The *Master↩ BusModeType* must be set to MCB_UP_COUNTER and the default period configured for the master bus must be equal to the default period of the Ocu logical channel.

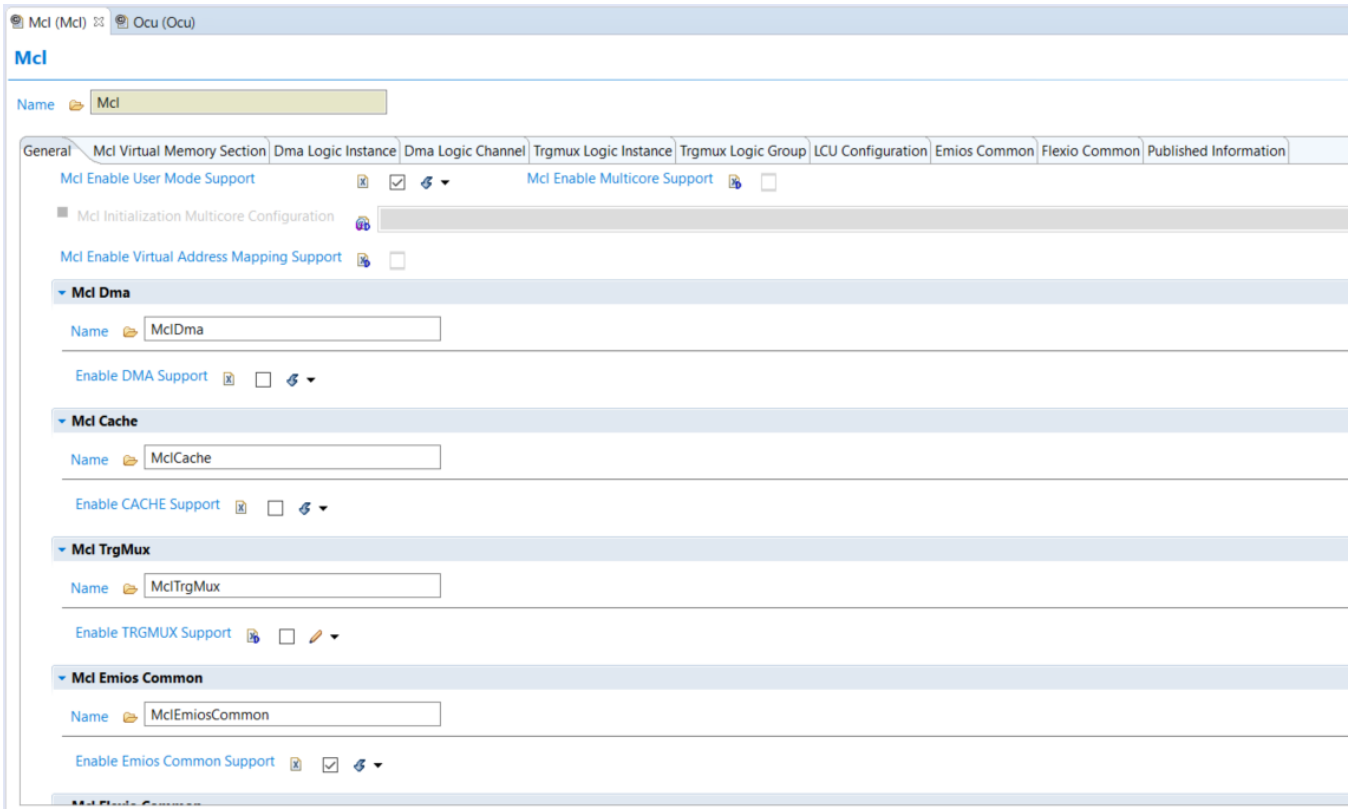

Figure 3.1 eMios common support containers in Mcl

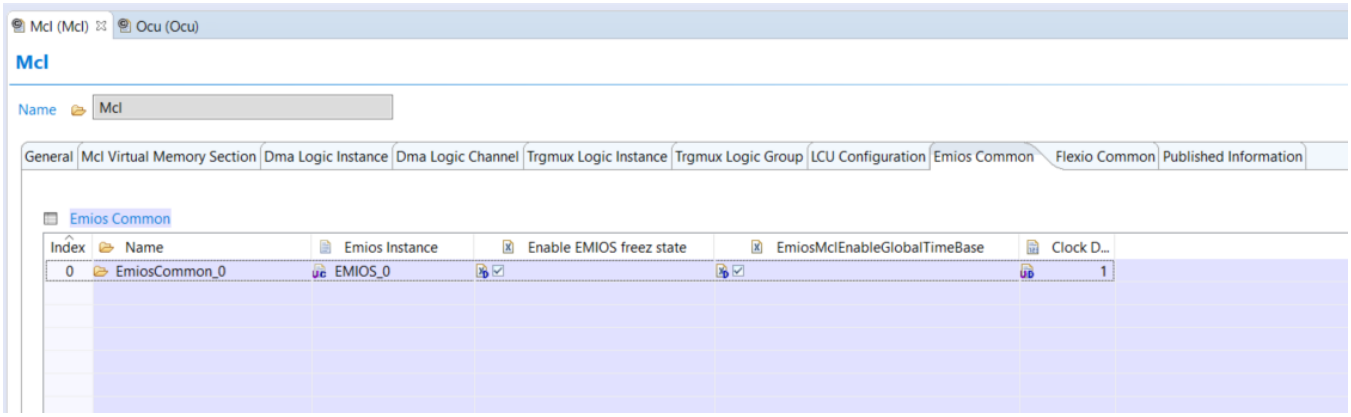

Figure 3.2 eMios common support container in Mcl

3.7 Runtime errors

None.

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 Ocu
  - Container OcuConfigurationOfOptionalApis
    - Parameter OcuDeInitApi
    - Parameter OcuGetCounterApi
    - Parameter OcuNotificationSupported
    - Parameter OcuSetAbsoluteThresholdApi
    - Parameter OcuSetPinActionApi
    - Parameter OcuSetPinStateApi
    - Parameter OcuSetRelativeThresholdApi
    - Parameter OcuVersionInfoApi
  - Container OcuConfigSet
    - Parameter OcuCountdirection
    - Container OcuChannel
      - Parameter OcuChannelId
      - Parameter OcuAssignedHardwareChannel
      - Parameter OcuDefaultThreshold
      - Parameter OcuHardwareTriggeredAdc
      - Parameter OcuHardwareTriggeredDMA
      - Parameter OcuMaxCounterValue
      - Parameter OcuNotification
      - Parameter OcuOutputPinUsed
      - Parameter OcuOutputPinDefaultState
      - Parameter OcuOutputPinAction
      - Parameter OcuUseMcuReferencePoint
      - Parameter OcuChannelTickDuration
      - Reference OcuHWSpecificSettingsRef
      - Reference OcuChannelEcucPartitionRef
      - Reference OcuMcuClockReferencePoint
    - Container OcuHWSpecificSettings
      - Parameter OcuHardwareElements

- Parameter OcuEmiosBusSelect
- Parameter OcuClockSource
- Parameter OcuPrescale
- Parameter OcuPrescale_Alternate
- Parameter OcuEmiosFreeze

– Container OcuGeneral

* Parameter OcuMulticoreEnabled
* Parameter OcuDevErrorDetect
* Parameter OcuEnableDualClockMode
* Parameter OcuEnableUserModeSupport
* Parameter OcuDownCountingSupport
* Reference OcuEcucPartitionRef
* Reference OcuKernelEcucPartitionRef
* Container OcuHwResourceConfig
  - Parameter OcuHwResourceId
  - Parameter OcuIsrEnable
  - Parameter OcuChannelIsUsed

– Container CommonPublishedInformation

* Parameter ArReleaseMajorVersion
* Parameter ArReleaseMinorVersion
* Parameter ArReleaseRevisionVersion
* Parameter ModuleId
* Parameter SwMajorVersion
* Parameter SwMinorVersion
* Parameter SwPatchVersion
* Parameter VendorApiInfix
* Parameter VendorId

4.1 Module Ocu

Configuration of Ocu (Output Compare Unit) module.

Included containers:

- OcuConfigurationOfOptionalApis
- OcuConfigSet
- OcuGeneral
- CommonPublishedInformation

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

4.2 Container OcuConfigurationOfOptionalApis

Configuration of optional APIs.

Included subcontainers:

- None

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

4.3 Parameter OcuDeInitApi

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

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

4.4 Parameter OcuGetCounterApi

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

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

Property	Value
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

4.5 Parameter OcuNotificationSupported

Adds / removes the services Ocu_EnableNotification() and Ocu_DisableNotification() from the code.

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

4.6 Parameter OcuSetAbsoluteThresholdApi

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

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	AUTOSAR_ECUC
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	true

4.7 Parameter OcuSetPinActionApi

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

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

4.8 Parameter OcuSetPinStateApi

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

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

4.9 Parameter OcuSetRelativeThresholdApi

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

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

4.10 Parameter OcuVersionInfoApi

Switch to indicate that the Ocu_GetVersionInfo() is supported.

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	false

4.11 Container OcuConfigSet

This container is the base of a Configuration Set, which contains the configured OCU channels. This way, different configuration sets can be defined for post-build process.

Included subcontainers:

- OcuChannel
- OcuHWSpecificSettings

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

4.12 Parameter OcuCountdirection

This parameter indicates the count direction for the whole OCU driver.

NoteeMios IP does not support down counting. Up-counting is always used as default

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	true
multiplicityConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	OCU_UPCOUNTING
literals	['OCU_DOWNCOUNTING', 'OCU_UPCOUNTING']

4.13 Container OcuChannel

Configuration of an individual OCU channel.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	72
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE

4.14 Parameter OcuChannelId

Channel Id of the OCU channel. This value will be assigned to the symbolic name derived from the OcuChannel container short name. It defines the assignment of the channel to the physical OCU hardware channel.

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-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	1
max	65535
min	0

4.15 Parameter OcuAssignedHardwareChannel

The physical hardware channel that is assigned to this logical channel.

Note Please use OcuHWSpecificSettingsRef parameter to assign a hardware channel.

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

Property	Value
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	1
max	255
min	0

4.16 Parameter OcuDefaultThreshold

Value of comparison threshold used for Initialization.

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-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	10000
max	16777215
min	0

4.17 Parameter OcuHardwareTriggeredAdc

This parameter is used to allow the OCU channel to trigger an ADC channel upon compare match, if this is supported by hardware. The value of the parameter represents the ADC physical channel to trigger..

NoteTrigger for ADC should be routed via MCL and has not supported yet in OCU driver. Hardware trigger for ADC could be used by PWM

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	0

Property	Value
upperMultiplicity	1
postBuildVariantMultiplicity	true
multiplicityConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	0
max	255
min	0

4.18 Parameter OcuHardwareTriggeredDMA

This parameter is used to allow the OCU channel to trigger a DMA channel upon compare match, if this is supported by hardware. The value of the parameter represents the DMA physical channel to trigger.

NoteHardware trigger for DMA has not supported yet in OCU driver.

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

4.19 Parameter OcuMaxCounterValue

Maximum value in ticks, the counter of the OCU channel is able to count.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	AUTOSAR_ECUC

Property	Value
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	65535
max	16777215
min	1

4.20 Parameter OcuNotification

User callback function

NOTE: please use NULL or NULL_PTR w/o any quotes. If the used string is different from NULL or NULL_PTR it will be used as the configured function name.

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

4.21 Parameter OcuOutputPinUsed

Information about the usage of an output pin on this channel.

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

Property	Value
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

4.22 Parameter OcuOutputPinDefaultState

The parameter OcuOutputPinDefaultState represents the state that a pin associated with a channel shall be set to after initialisation.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	true
multiplicityConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	OCU_LOW
literals	['OCU_HIGH', 'OCU_LOW']

4.23 Parameter OcuOutputPinAction

The parameter OcuOutputPinAction represents the action that a pin associated with a channel shall be set imediatly after a compare match event.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	0

Property	Value
upperMultiplicity	1
postBuildVariantMultiplicity	true
multiplicityConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	OCU_DISABLE
literals	['OCU_SET_LOW', 'OCU_SET_HIGH', 'OCU_TOGGLE', 'OCU_DISABLE']

4.24 Parameter OcuUseMcuReferencePoint

This parameter allows for automatic calculation of the Channel Tick duration based on the

configured MCU Referenced Clock.

If OcuUseMcuReferencePoint is set to true than selectig a Clock source from MCU is possible and based on that source clock OcuChannelTickDuration can be auto-calculated.

Note This paramenter is not supported in current implementation.

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-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	false

4.25 Parameter OcuChannelTickDuration

Specifies the tick duration of the counter of the channel. This parameter is the number of the input clock edges (rising edges or falling edges exclusively) counted each time to increase the counter by one unit.

Note. The default value is calculated using a default prescaler of 1. For a different prescaler value the calculated value should be devided by the said prescaler value.

Note Ocu Channel Tick Duration has not supported yet in OCU driver.

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-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	1
max	32768
min	1

4.26 Reference OcuHWSpecificSettingsRef

Reference to the OcuHWSpecificSettings used by the OcuChannel.

Property	Value
type	ECUC-REFERENCE-DEF
origin	AUTOSAR_ECUC
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
destination	/AUTOSAR/EcucDefs/Ocu/OcuConfigSet/OcuHWSpecificSettings

4.27 Reference OcuChannelEcucPartitionRef

Maps a OCU 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 OCU driver is mapped to.

Property	Value
type	ECUC-REFERENCE-DEF

Property	Value
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

4.28 Reference OcuMcuClockReferencePoint

Reference to the eMios clock source configuration,

which is set in the MCU driver configuration.

Note This parameter is not supported in current implementation.

Property	Value
type	ECUC-REFERENCE-DEF
origin	NXP
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/Mcu/McuModuleConfiguration/McuClockSetting↩ Config/McuClockReferencePoint

4.29 Container OcuHWSpecificSettings

Configuration of an Ocu eMios module available on the platfom.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	0
upperMultiplicity	72
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE

4.30 Parameter OcuHardwareElements

Selects one of the eMios modules available on the platform.

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_0_CH_0
literals	['EMIOS_0_CH_0', 'EMIOS_0_CH_1', 'EMIOS_0_CH_2', 'EMIOS_0_C↩ H_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', 'EM↩ IOS_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', 'EMIO↩ S_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', 'EMIO↩ S_1_CH_5', 'EMIOS_1_CH_6', 'EMIOS_1_CH_7', 'EMIOS_1_CH_8', 'E↩ MIOS_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', 'EMIO↩ S_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', 'EMIO↩ S_2_CH_3', 'EMIOS_2_CH_4', 'EMIOS_2_CH_5', 'EMIOS_2_CH_6', 'E↩ MIOS_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', 'EMIO↩ S_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']

4.31 Parameter OcuEmiosBusSelect

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_OCU_BUS_A
literals	['EMIOS_OCU_BUS_A', 'EMIOS_OCU_BUS_B', 'EMIOS_OCU_BUS_C', 'EMIOS_OCU_BUS_D', 'EMIOS_OCU_BUS_F', 'EMIOS_OCU_BUS_IN↩ TERNAL_COUNTER']

4.32 Parameter OcuClockSource

Optional OCU driver specific clock prescale factor, if supported by hardware. Implementation is defined vendor specific.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	true
multiplicityConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	OCU_SYSTEM_CLOCK
literals	['OCU_SYSTEM_CLOCK', 'OCU_FIXED_FREQ_CLOCK', 'OCU_EXTE↩ RNAL_CLOCK']

4.33 Parameter OcuPrescale

Optional OCU driver specific clock prescale factor, if supported by hardware. Implementation is defined vendor specific.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	true
multiplicityConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	DIV_1
literals	['DIV_1', 'DIV_2', 'DIV_3', 'DIV_4', 'DIV_5', 'DIV_6', 'DIV_7', 'DIV_8', 'DIV_9', 'DIV_10', 'DIV_11', 'DIV_12', 'DIV_13', 'DIV_14', 'DIV_15', 'DI↩ V_16']

4.34 Parameter OcuPrescale_Alternate

Optional specific clock prescale factor used for this eMios channel selected internal counterbus. This parameter will be used by the Ocu_SetClockMode function.

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	DIV_1
literals	['DIV_1', 'DIV_2', 'DIV_3', 'DIV_4', 'DIV_5', 'DIV_6', 'DIV_7', 'DIV_8', 'DIV_9', 'DIV_10', 'DIV_11', 'DIV_12', 'DIV_13', 'DIV_14', 'DIV_15', 'DI↩ V_16']

4.35 Parameter OcuEmiosFreeze

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

4.36 Container OcuGeneral

This container contains the module-wide configuration parameters of the OCU Driver.

Included subcontainers:

- OcuHwResourceConfig

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

4.37 Parameter OcuMulticoreEnabled

Switch to enable/disable multi-core feature.

User can choose ENABLE multi-core feature by checking this option, this will force to configure at least 1 ECUC partition in

OcuChannelEcucPartitionRef, and each OCU channel in OcuChannel to configure at least 1 ECUC partition reference in OcuChannelEcucPartitionRef container to fulfill generating code condition; OR unchecked this option to DISABLE multi-core feature, performing this action will force user to remove all ECUC partition reference in every OCU channels contained in OcuChannel and in OcuChannelEcucPartitionRef.

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

4.38 Parameter OcuDevErrorDetect

Switch for enabling the development error detection.

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	false

4.39 Parameter OcuEnableDualClockMode

Adds / removes the services Ocu_SetClockMode() from the code. This function is called when the prescaler value needs to be change to maintain same period at different frequency.

Note:

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	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.40 Parameter OcuEnableUserModeSupport

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

Configuring REG_PROT for eMios IP so that the registers under protection can be accessed from user mode by setting UAA bit in REG_PROT_GCR to 1

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

Note This parameter is not supported in current implementation.

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

4.41 Parameter OcuDownCountingSupport

When this parameter is enabled, it is possible to configure OcuCountdirection with OCU_DOWNCOUNTING.

NoteeMios IP does not support down counting

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

4.42 Reference OcuEcucPartitionRef

Maps the OCU 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
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

4.43 Reference OcuKernelEcucPartitionRef

Maps the OCU 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 OCU 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

4.44 Container OcuHwResourceConfig

List of HW interrupts available for the entire platform.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	72
upperMultiplicity	72
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.45 Parameter OcuHwResourceId

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

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	true
lowerMultiplicity	1

Property	Value
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
literals	['EMIOS_0_CH_0', 'EMIOS_0_CH_1', 'EMIOS_0_CH_2', 'EMIOS_0_C↩H_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', 'EM↩IOS_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', 'EMIO↩S_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', 'EMIO↩S_1_CH_5', 'EMIOS_1_CH_6', 'EMIOS_1_CH_7', 'EMIOS_1_CH_8', 'E↩MIOS_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', 'EMIO↩S_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', 'EMIO↩S_2_CH_3', 'EMIOS_2_CH_4', 'EMIOS_2_CH_5', 'EMIOS_2_CH_6', 'E↩MIOS_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', 'EMIO↩S_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']

4.46 Parameter OcuIsrEnable

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

4.47 Parameter OcuChannelIsUsed

This column configures HW channels which are going to be used.

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

4.48 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

4.49 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

Property	Value
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
defaultValue	4
max	4
min	4

4.50 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-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
defaultValue	7
max	7
min	7

4.51 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

Property	Value
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
defaultValue	0
max	0
min	0

4.52 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-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
defaultValue	125
max	125
min	125

4.53 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

Property	Value
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
defaultValue	3
max	3
min	3

4.54 Parameter SwMinorVersion

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

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-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
defaultValue	0
max	0
min	0

4.55 Parameter SwPatchVersion

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

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-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
defaultValue	0
max	0
min	0

4.56 Parameter VendorApiInfix

In driver modules which can be instantiated several times on a single ECU, BSW00347 requires that he 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-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
defaultValue	

4.57 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
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: 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 Ocu IPL

6.1.1 Detailed Description

Ocu eMIOS driver header file.

Ocu eMIOS driver header file specific to OCU driver.

Ocu eMIOS - header file support for OCU driver.

eMios driver interface.

eMios header file, containing the variables and functions that are exported by the IP driver.

eMios specific defines which need to be exported to external application

Data Structures

- struct Emios_Ocu_Ip_ChStateType

  internal context structure More...
- struct Emios_Ocu_Ip_ChannelConfigType

  eMios IP channel specific configuration structure for the Ocu functionality More...
- struct Emios_Ocu_Ip_ConfigType

  eMios IP specific configuration structure type More...

Macros

- #define OCU_EMIOS_INVALID_CHANNEL_NUM

  invalid eMios channel number
- #define MASTER_MODE_UP_COUNTER_U32

  Bus Select: MC up counter.
- #define MASTER_MODE_UP_DOWN_COUNTER_U32

  Bus Select: MC up/down counter.
- #define MASTER_MODE_UP_BUFFERED_COUNTER_U32

  Bus Select: MCB up.
- #define MASTER_MODE_UP_DOWN_BUFFERED_COUNTER_U32

  Bus Select: MCB up/down.
- #define MCL_MASTER_BUS_PRESCALER_1_U32

  Initialization option to set the channel prescaler value to 1.
- #define MCL_MASTER_BUS_PRESCALER_2_U32

  Initialization option to set the channel prescaler value to 2.
- #define MCL_MASTER_BUS_PRESCALER_3_U32

  Initialization option to set the channel prescaler value to 3.
- #define MCL_MASTER_BUS_PRESCALER_4_U32

  Initialization option to set the channel prescaler value to 4.
- #define OCU_EMIOS_SET_LOW

  Edge Pin Action values.
- #define OCU_EMIOS_SET_HIGH

  The channel pin will be set LOW upon compare match.
- #define OCU_EMIOS_SET_TOGGLE

  The channel pin will be set to the opposite of its current level HIGH upon compare match.
- #define OCU_EMIOS_SET_DISABLE

  The channel pin will remain at its current level upon compare match.
- #define OCU_EMIOS_LOW

  Pin State levels.
- #define OCU_EMIOS_HIGH

  Ocu Pin level is logic high.
- #define OCU_EMIOS_PRIMARY_PRESCALER

  Prescaler values.
- #define OCU_EMIOS_ALTERNATIVE_PRESCALER

  Selected value is the alternative configured prescaler.
- #define OCU_EMIOS_CM_IN_REF_INTERVAL

  Ocu Return values.
- #define OCU_EMIOS_CM_OUT_REF_INTERVAL

  The compare match will not occur inside the current Reference Interval.

Types Reference

- typedef uint8 Emios_Ocu_Ip_PinActionType
  Edge Pin Action type.
- typedef uint8 Emios_Ocu_Ip_PinStateType
  Pin State level.
- typedef uint8 Emios_Ocu_Ip_SelectPrescalerType
  Prescaler type.
- typedef uint8 Emios_Ocu_Ip_ReturnType
  Ocu Return Type.
- typedef eMIOS_Type Emios_Ocu_Ip_xRegLayoutType
- typedef void(* Emios_Ocu_Ip_CallbackType) (uint16 CallbackParam)
  Channel notification typedef.

Enum Reference

- enum Emios_Ocu_Ip_OperationModeType
  UC modes of operation - Output compare mode.
- enum Emios_Ocu_Ip_CountDirectionType
  Ocu Count direction.
- enum Emios_Ocu_Ip_BusSelectType
  Counter bus select.

Function Reference

- void Emios_Ocu_Ip_Init (const Emios_Ocu_Ip_ConfigType *const pEmiosIpConfig)
  Ocu driver initialization function for eMios module.
- void Emios_Ocu_Ip_DeInit (uint8 InstNum)
  Ocu driver de-initialization function for eMios module.
- void Emios_Ocu_Ip_StartChannel (uint8 InstNum, uint8 ChNum)
  Ocu driver function for starting the eMios timer channel.
- void Emios_Ocu_Ip_StopChannel (uint8 InstNum, uint8 ChNum)
  Ocu driver function for stopping the eMios timer channel.
- Emios_Ocu_Ip_ValueType Emios_Ocu_Ip_GetCounter (uint8 InstNum, uint8 ChNum)
  Ocu driver function for getting the current counter value for a eMIOS timer channel.
- void Emios_Ocu_Ip_SetPinState (uint8 InstNum, uint8 ChNum, Emios_Ocu_Ip_PinStateType PinState)
  Ocu driver function for setting the Pin State for the current eMIOS channel.
- void Emios_Ocu_Ip_SetPinAction (uint8 InstNum, uint8 ChNum, Emios_Ocu_Ip_PinActionType Pin↩ Action)
  Ocu driver function for setting the Pin Action for the current eMIOS channel.
- Emios_Ocu_Ip_ReturnType Emios_Ocu_Ip_SetAbsoluteThreshold (uint8 InstNum, uint8 ChNum, Emios↩ _Ocu_Ip_ValueType ReferenceVal, Emios_Ocu_Ip_ValueType AbsoluteVal)
  The function changes the eMios compare register value.
- Emios_Ocu_Ip_ReturnType Emios_Ocu_Ip_SetRelativeThreshold (uint8 InstNum, uint8 ChNum, Emios↩ _Ocu_Ip_ValueType RelativeValue)

The function changes the eMios compare register value.

- void Emios_Ocu_Ip_SetClockMode (uint8 InstNum, Emios_Ocu_Ip_SelectPrescalerType Prescaler)

  The function changes the prescaler of eMIOS channel.
- uint32 Emios_Ocu_Ip_GetMaxCounterValue (uint8 InstNum, uint8 ChNum)

  Emios_Ocu_Ip_GetMaxCounterValue.
- void Emios_Ocu_Ip_DisableNotification (uint8 InstNum, uint8 ChNum)

  This function is used to disable the OCU compare match notification.
- void Emios_Ocu_Ip_EnableNotification (uint8 InstNum, uint8 ChNum)

  This function is used to enable the OCU compare match notification of the indexed channel.
- void Emios_Ocu_Ip_IrqHandler (uint8 InstNum, uint8 ChIdx)

  Driver routine to process all the interrupts of eMios.

- #define OCU_EMIOS_OUTPUT_DISABLED

  eMios register define used to configure counter clock source.

- #define EMIOS_PRESCALER_DIVIDE1

6.1.2 Data Structure Documentation

6.1.2.1 struct Emios_Ocu_Ip_ChStateType

internal context structure

This structure is used by the IPL driver for internal logic. The content is populated on Init

Definition at line 373 of file Emios_Ocu_Ip_Types.h.

6.1.2.2 struct Emios_Ocu_Ip_ChannelConfigType

eMios IP channel specific configuration structure for the Ocu functionality

Definition at line 383 of file Emios_Ocu_Ip_Types.h.

Data Fields

Type	Name	Description
const uint8	HwChannel	Assigned eMios Hw Channel Id.
const Emios_Ocu_Ip_ValueType	DefaultThreshold	Compare match threshold for the current channel.
const Emios_Ocu_Ip_ValueType	MaxCounterValue	Maximum counter value for the current channel.
const uint32	AltControlValue	eMios channel parameters bits 4 .. 7 --> Prescaler configuration

Data Fields

Type	Name	Description
const uint32	ControlValue	eMios configuration parameter bit 24 --> Set Edge Polarity bit bit 23 --> Edge action on match bit bits 21 .. 22 --> Set Counter Bus bits bit 10 --> Set pin used bits 5 .. 8 --> Prescaler configuration bit 1 --> Set or Disable output bit 0 --> Set or Disable timer functionality in freeze mode
const uint8	HwChannelCounterBus	Assigned eMios Hw Channel Id of counter bus.
const Emios_Ocu_Ip_CallbackType	mCallbackFunc	Channel notification function.
const uint16	mCallbackParam	Channel notification parameter.

6.1.2.3 struct Emios_Ocu_Ip_ConfigType

eMios IP specific configuration structure type

Definition at line 416 of file Emios_Ocu_Ip_Types.h.

Data Fields

Type	Name	Description
const uint8	NumChannels	Number of configured channels within this eMios instance.
const uint8	InstanceNo	Number of this eMios instance.
const Emios_Ocu_Ip_ChannelConfigType(*	pcxChannelsConfig)[]	Pointer to the channels configured for this eMios instnace.

6.1.3 Macro Definition Documentation

6.1.3.1 OCU_EMIOS_OUTPUT_DISABLED

```
#define OCU_EMIOS_OUTPUT_DISABLED
```

eMios register define used to configure counter clock source.

Definition at line 110 of file Emios_Ocu_Ip_Types.h.

6.1.3.2 EMIOS_PRESCALER_DIVIDE1

```
#define EMIOS_PRESCALER_DIVIDE1
```

eMios unified channel prescaller control

Definition at line 155 of file Emios_Ocu_Ip_Types.h.

6.1.3.3 OCU_EMIOS_INVALID_CHANNEL_NUM

```
#define OCU_EMIOS_INVALID_CHANNEL_NUM
```

invalid eMios channel number

Definition at line 191 of file Emios_Ocu_Ip_Types.h.

6.1.3.4 MASTER_MODE_UP_COUNTER_U32

```
#define MASTER_MODE_UP_COUNTER_U32
```

Bus Select: MC up counter.

Definition at line 196 of file Emios_Ocu_Ip_Types.h.

6.1.3.5 MASTER_MODE_UP_DOWN_COUNTER_U32

```
#define MASTER_MODE_UP_DOWN_COUNTER_U32
```

Bus Select: MC up/down counter.

Definition at line 201 of file Emios_Ocu_Ip_Types.h.

6.1.3.6 MASTER_MODE_UP_BUFFERED_COUNTER_U32

```
#define MASTER_MODE_UP_BUFFERED_COUNTER_U32
```

Bus Select: MCB up.

Definition at line 206 of file Emios_Ocu_Ip_Types.h.

6.1.3.7 MASTER_MODE_UP_DOWN_BUFFERED_COUNTER_U32

```
#define MASTER_MODE_UP_DOWN_BUFFERED_COUNTER_U32
```

Bus Select: MCB up/down.

Definition at line 211 of file Emios_Ocu_Ip_Types.h.

6.1.3.8 MCL_MASTER_BUS_PRESCALER_1_U32

```
#define MCL_MASTER_BUS_PRESCALER_1_U32
```

Initialization option to set the channel prescaler value to 1.

Definition at line 215 of file Emios_Ocu_Ip_Types.h.

6.1.3.9 MCL_MASTER_BUS_PRESCALER_2_U32

```
#define MCL_MASTER_BUS_PRESCALER_2_U32
```

Initialization option to set the channel prescaler value to 2.

Definition at line 220 of file Emios_Ocu_Ip_Types.h.

6.1.3.10 MCL_MASTER_BUS_PRESCALER_3_U32

```
#define MCL_MASTER_BUS_PRESCALER_3_U32
```

Initialization option to set the channel prescaler value to 3.

Definition at line 225 of file Emios_Ocu_Ip_Types.h.

6.1.3.11 MCL_MASTER_BUS_PRESCALER_4_U32

```
#define MCL_MASTER_BUS_PRESCALER_4_U32
```

Initialization option to set the channel prescaler value to 4.

Definition at line 230 of file Emios_Ocu_Ip_Types.h.

6.1.3.12 OCU_EMIOS_SET_LOW

```
#define OCU_EMIOS_SET_LOW
```

Edge Pin Action values.

Automatic action (by hardware) to be performed on a pin attached to an OCU channel.

The channel pin will be set HIGH upon compare match.

Definition at line 238 of file Emios_Ocu_Ip_Types.h.

6.1.3.13 OCU_EMIOS_SET_HIGH

```
#define OCU_EMIOS_SET_HIGH
```

The channel pin will be set LOW upon compare match.

Definition at line 240 of file Emios_Ocu_Ip_Types.h.

6.1.3.14 OCU_EMIOS_SET_TOGGLE

```
#define OCU_EMIOS_SET_TOGGLE
```

The channel pin will be set to the opposite of its current level HIGH upon compare match.

Definition at line 242 of file Emios_Ocu_Ip_Types.h.

6.1.3.15 OCU_EMIOS_SET_DISABLE

```
#define OCU_EMIOS_SET_DISABLE
```

The channel pin will remain at its current level upon compare match.

Definition at line 244 of file Emios_Ocu_Ip_Types.h.

6.1.3.16 OCU_EMIOS_LOW

```
#define OCU_EMIOS_LOW
```

Pin State levels.

Output state of the pin linked to an OCU channel.

Ocu Pin level is logic low

Definition at line 252 of file Emios_Ocu_Ip_Types.h.

6.1.3.17 OCU_EMIOS_HIGH

```
#define OCU_EMIOS_HIGH
```

Ocu Pin level is logic high.

Definition at line 254 of file Emios_Ocu_Ip_Types.h.

6.1.3.18 OCU_EMIOS_PRIMARY_PRESCALER

```
#define OCU_EMIOS_PRIMARY_PRESCALER
```

Prescaler values.

Possible values of prescallers used to configure base-clock timers

Selected value is the default/primary prescaler

Definition at line 262 of file Emios_Ocu_Ip_Types.h.

6.1.3.19 OCU_EMIOS_ALTERNATIVE_PRESCALER

```
#define OCU_EMIOS_ALTERNATIVE_PRESCALER
```

Selected value is the alternative configured prescaler.

Definition at line 264 of file Emios_Ocu_Ip_Types.h.

6.1.3.20 OCU_EMIOS_CM_IN_REF_INTERVAL

```
#define OCU_EMIOS_CM_IN_REF_INTERVAL
```

Ocu Return values.

Return information after setting a new threshold value.

The compare match will occur inside the current Reference Interval.

Definition at line 272 of file Emios_Ocu_Ip_Types.h.

6.1.3.21 OCU_EMIOS_CM_OUT_REF_INTERVAL

```
#define OCU_EMIOS_CM_OUT_REF_INTERVAL
```

The compare match will not occur inside the current Reference Interval.

Definition at line 274 of file Emios_Ocu_Ip_Types.h.

6.1.4 Types Reference

6.1.4.1 Emios_Ocu_Ip_PinActionType

```
typedef uint8 Emios_Ocu_Ip_PinActionType
```

Edge Pin Action type.

Automatic action (by hardware) to be performed on a pin attached to an OCU channel.

Definition at line 337 of file Emios_Ocu_Ip_Types.h.

6.1.4.2 Emios_Ocu_Ip_PinStateType

```
typedef uint8 Emios_Ocu_Ip_PinStateType
```

Pin State level.

Output state of the pin linked to an OCU channel.

Definition at line 344 of file Emios_Ocu_Ip_Types.h.

6.1.4.3 Emios_Ocu_Ip_SelectPrescalerType

```
typedef uint8 Emios_Ocu_Ip_SelectPrescalerType
```

Prescaler type.

Specifies the possible values of prescallers used to configure base-clock timers

Definition at line 351 of file Emios_Ocu_Ip_Types.h.

6.1.4.4 Emios_Ocu_Ip_ReturnType

```
typedef uint8 Emios_Ocu_Ip_ReturnType
```

Ocu Return Type.

Return information after setting a new threshold value.

Definition at line 358 of file Emios_Ocu_Ip_Types.h.

6.1.4.5 Emios_Ocu_Ip_xRegLayoutType

```
typedef eMIOS_Type Emios_Ocu_Ip_xRegLayoutType
```

Redefine eMIOS Register Layout Typedef from header file to comply with coding guidelines

Definition at line 361 of file Emios_Ocu_Ip_Types.h.

6.1.4.6 Emios_Ocu_Ip_CallbackType

```
typedef void(* Emios_Ocu_Ip_CallbackType) (uint16 CallbackParam)
```

Channel notification typedef.

Definition at line 364 of file Emios_Ocu_Ip_Types.h.

6.1.5 Enum Reference

6.1.5.1 Emios_Ocu_Ip_OperationModeType

```
enum Emios_Ocu_Ip_OperationModeType
```

UC modes of operation - Output compare mode.

Enumerator

EMIOS_MODE_GPIO_IN	General Purpose Input In GPIO mode, all input capture and output compare functions of the Unified Channel are disabled, and the internal counter (EMIOSCNTn register) is cleared and disabled.
EMIOS_MODE_SAOC	Single-Action Output Compare
EMIOS_MODE_DAOC_FSET_TRAILING_MATCH	Double-Action Output Compare, FLAG set at Trailing edge match
EMIOS_MODE_DAOC_FSET_BOTH_MATCH	Double-Action Output Compare, FLAG set at both Leading & Trailing edge match
EMIOS_MODE_MCB_COUNTER_UP	Modulus Counter up mode

Definition at line 285 of file Emios_Ocu_Ip_Types.h.

6.1.5.2 Emios_Ocu_Ip_CountDirectionType

enum Emios_Ocu_Ip_CountDirectionType

Ocu Count direction.

This enum specifies the count direction for the whole Ocu driver.

Definition at line 301 of file Emios_Ocu_Ip_Types.h.

6.1.5.3 Emios_Ocu_Ip_BusSelectType

enum Emios_Ocu_Ip_BusSelectType

Counter bus select.

Select either one of counter bus or the internal counter to be used by the Unified Channel.

Enumerator

EMIOS_OCU_BUS_A	Global counter bus A
EMIOS_OCU_BUS_B	Local counter bus
EMIOS_OCU_BUS_C	Local counter bus
EMIOS_OCU_BUS_D	Local counter bus
EMIOS_OCU_BUS_F	Global counter bus F
EMIOS_OCU_BUS_INTERNAL_COUNTER	Internal counter bus

Definition at line 314 of file Emios_Ocu_Ip_Types.h.

6.1.6 Function Reference

6.1.6.1 Emios_Ocu_Ip_Init()

```
void Emios_Ocu_Ip_Init (
            const Emios_Ocu_Ip_ConfigType *const pEmiosIpConfig )
```

Ocu driver initialization function for eMios module.

This function is called once for all eMios hw channels corresponding to the configured timer channels, and:

- initializes the global data-stores
- disables the timer channel
- disables the timer compare interrupts corresponding to eMios channel
- clears the timer compare interrupt flags corresponding to eMios channel
- Select the clock divider value for the internal prescaler of Unified Channel
- select counter Bus (A, B, C, D, E or INTERNAL Bus)
- set the compare register A to configured default threshold value to eMios channel.
- set the compare register A of Counter Bus to configured max counter value.
- if output pin is enable, set the mode for SAOC and the output pin operates normally.
- clear pending interrupt serviced.
- enables the prescaler for counter bus (start run counter).
- configures UC Control n (C0 - C23) registers to expected values.

Parameters

in	*pEmiosIpConfig*	- Pointer to configuration structure of emios channels

6.1.6.2 Emios_Ocu_Ip_DeInit()

```
void Emios_Ocu_Ip_DeInit (
            uint8 InstNum )
```

Ocu driver de-initialization function for eMios module.

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

- disables the timer compare interrupts corresponding to eMios channel
- clears the timer compare interrupt flags corresponding to eMios channel
- resets eMIOS UC Control register.
- resets the compare register eMIOS UC A register .
- resets eMIOS UC Control register of Master Bus.
- resets the compare register eMIOS UC A register of Master Bus.

Parameters

in	*InstNum*	- eMIOS instance number

Returns

void

6.1.6.3 Emios_Ocu_Ip_StartChannel()

```
void Emios_Ocu_Ip_StartChannel (
             uint8 InstNum,
             uint8 ChNum )
```

Ocu driver function for starting the eMios timer channel.

This function enables the channel timer.

Parameters

in	*InstNum*	- eMIOS instance number
in	*ChNum*	- Channel number in the eMIOS instance

6.1.6.4 Emios_Ocu_Ip_StopChannel()

```
void Emios_Ocu_Ip_StopChannel (
             uint8 InstNum,
             uint8 ChNum )
```

Ocu driver function for stopping the eMios timer channel.

This function stops the timer channel

Parameters

in	*InstNum*	- eMIOS instance number
in	*ChNum*	- Channel number in the eMIOS instance

6.1.6.5 Emios_Ocu_Ip_GetCounter()

```
Emios_Ocu_Ip_ValueType Emios_Ocu_Ip_GetCounter (
            uint8 InstNum,
            uint8 ChNum )
```

Ocu driver function for getting the current counter value for a eMIOS timer channel.

This function is called for reading the eMIOS channel counter register.

Parameters

in	*InstNum*	- eMIOS instance number
in	*ChNum*	- Channel number in the eMIOS instance

return current value of the eMIOS channel counter register

6.1.6.6 Emios_Ocu_Ip_SetPinState()

```
void Emios_Ocu_Ip_SetPinState (
            uint8 InstNum,
            uint8 ChNum,
            Emios_Ocu_Ip_PinStateType PinState )
```

Ocu driver function for setting the Pin State for the current eMIOS channel.

Parameters

in	*InstNum*	- eMIOS instance number
in	*ChNum*	- Channel number in the eMIOS instance PinState - Expectation state of pin: OCU_EMIOS_HIGH, OCU_EMIOS_LOW

6.1.6.7 Emios_Ocu_Ip_SetPinAction()

```
void Emios_Ocu_Ip_SetPinAction (
            uint8 InstNum,
```

```
            uint8 ChNum,
            Emios_Ocu_Ip_PinActionType PinAction )
```

Ocu driver function for setting the Pin Action for the current eMIOS channel.

Parameters

in	*InstNum*	- eMIOS instance number
in	*ChNum*	- Channel number in the eMIOS instance PinAction - Type of action: OCU_EMIOS_SET_LOW, OCU_EMIOS_SET_HIGH, OCU_EMIOS_SET_TOGGLE

6.1.6.8 Emios_Ocu_Ip_SetAbsoluteThreshold()

```
Emios_Ocu_Ip_ReturnType Emios_Ocu_Ip_SetAbsoluteThreshold (
            uint8 InstNum,
            uint8 ChNum,
            Emios_Ocu_Ip_ValueType ReferenceVal,
            Emios_Ocu_Ip_ValueType AbsoluteVal )
```

The function changes the eMios compare register value.

This function:

- Clears the compare load register 1.
- sets the compare load register 2 to the new timeout value.

Parameters

in	*InstNum*	- eMIOS instance number
in	*ChNum*	- Channel number in the eMIOS instance
in	*ReferenceVal*	- Value given by the upper layer and used as a base to determine
in	*AbsoluteVal*	- channel timeout value

return Tells the caller whether the compare match will occur (or has already occured) during the current Refence Interval, as a result of setting the new threshold value.

6.1.6.9 Emios_Ocu_Ip_SetRelativeThreshold()

```
Emios_Ocu_Ip_ReturnType Emios_Ocu_Ip_SetRelativeThreshold (
            uint8 InstNum,
            uint8 ChNum,
            Emios_Ocu_Ip_ValueType RelativeValue )
```

The function changes the eMios compare register value.

This function:

- Clears the compare load register 1.
- sets the compare load register 2 to the new timeout value.

Parameters

in	*InstNum*	- eMIOS instance number
in	*ChNum*	- Channel number in the eMIOS instance RelativeValue - Channel timeout value

return Tells the caller whether the compare match will occur (or has already occured) during the current Refence Interval, as a result of setting the new threshold value.

6.1.6.10 Emios_Ocu_Ip_SetClockMode()

```
void Emios_Ocu_Ip_SetClockMode (
             uint8 InstNum,
             Emios_Ocu_Ip_SelectPrescalerType Prescaler )
```

The function changes the prescaler of eMIOS channel.

This function changes the clock divider value for the UC internal prescaler

Parameters

in	*InstNum*	- eMIOS instance number
in	*Prescaler*	- Prescaler type: normal or alternative

6.1.6.11 Emios_Ocu_Ip_GetMaxCounterValue()

```
uint32 Emios_Ocu_Ip_GetMaxCounterValue (
             uint8 InstNum,
             uint8 ChNum )
```

Emios_Ocu_Ip_GetMaxCounterValue.

This function will return max counter value for given channel

Parameters

in	*InstNum*	- eMIOS instance number
in	*ChNum*	- Channel number in the eMIOS instance

Returns

uint32 - Max counter value for given channel

6.1.6.12 Emios_Ocu_Ip_DisableNotification()

```
void Emios_Ocu_Ip_DisableNotification (
            uint8 InstNum,
            uint8 ChNum )
```

This function is used to disable the OCU compare match notification.

Parameters

in	*InstNum*	- eMIOS instance number
in	*ChNum*	- Channel number in the eMIOS instance

Returns

none

6.1.6.13 Emios_Ocu_Ip_EnableNotification()

```
void Emios_Ocu_Ip_EnableNotification (
            uint8 InstNum,
            uint8 ChNum )
```

This function is used to enable the OCU compare match notification of the indexed channel.

Parameters

in	*InstNum*	- eMIOS instance number
in	*ChNum*	- Channel number in the eMIOS instance

Returns

none

6.1.6.14 Emios_Ocu_Ip_IrqHandler()

```
void Emios_Ocu_Ip_IrqHandler (
            uint8 InstNum,
            uint8 ChIdx )
```

Driver routine to process all the interrupts of eMios.

Support function used by interrupt service routines to implement eMios specific operations and call the EMIOS_↩ OCU_IP upper layer handler to implement non-hardware specific operations.

Parameters

in	*InstNum*	Instance number of eMIOS module
in	*ChIdx*	The channel in the instance

6.2 Ocu Driver

6.2.1 Detailed Description

Ocu HLD module API header.

Ocu HLD generic types file.

Ocu HLD notifications API header.

Ocu HLD Environment configuration header file.

Ocu module API header, containing Autosar API and function that are exported by OCU driver

Ocu environment configuration header.

Ocu notifications API header.

OCU header file containing generic Ocu types and structures.

Data Structures

- struct Ocu_ChannelConfigType

  Ocu high level channel configuration structure. More...
- struct Ocu_ConfigType

  Ocu high level configuration structure. More...

Macros

- #define OCU_VALID_CHANNEL_NUM

  Valid channel number.
- #define OCU_INVALID_CHANNEL_NUM

  Invalid channel number.
- #define OCU_INSTANCE_INDEX

  Instance ID of this OCU driver.
- #define OCU_E_UNINIT

  API service used without module initialization.
- #define OCU_E_PARAM_INVALID_CHANNEL

  API service used with an invalid channel Identifier.
- #define OCU_E_PARAM_INVALID_STATE

  API Ocu_SetPinState() called with an invalid pin state or when the channel is in the RUNNING state..
- #define OCU_E_PARAM_INVALID_ACTION

  API Ocu_SetPinAction() called with an invalid pin action.
- #define OCU_E_NO_VALID_NOTIF

  Usage of Ocu_DisableNotification() or Ocu_EnableNotification() on a channel where a NULL pointer is configured as the notification function.
- #define OCU_E_ALREADY_INITIALIZED

  API Ocu_Init() called while the OCU driver has already been initialized.
- #define OCU_E_PARAM_POINTER

API Ocu_GetVersionInfo() is called with a NULL parameter.

- #define OCU_E_BUSY

  API Ocu_StartChannel() called on a channel that is in state RUNNING.
- #define OCU_E_PARAM_NO_PIN

  Ocu_SetPinState() or Ocu_SetPinAction() called for a channel that doesn't have an associated output pin.
- #define OCU_E_INIT_FAILED

  API Ocu_Init service called with wrong parameter.
- #define OCU_E_PARAM_INVALID_VALUE

  Ocu_SetAbsoluteThreshold() or Ocu_SetRelativeThreshold() called for with a compare match parameter greater than maximum supported counter value for a given channel.
- #define OCU_E_PARAM_CONFIG

  API Ocu_Init service called with wrong Core number.
- #define OCU_INIT_ID

  API service ID of Ocu_Init() function.
- #define OCU_DEINIT_ID

  API service ID of Ocu_DeInit() function.
- #define OCU_STARTCHANNEL_ID

  API service ID of Ocu_StartChannel() function.
- #define OCU_STOPCHANNEL_ID

  API service ID of Ocu_StopChannel() function.
- #define OCU_SETPINSTATE_ID

  API service ID of Ocu_SetPinState() function.
- #define OCU_SETPINACTION_ID

  API service ID of Ocu_SetPinAction() function.
- #define OCU_GETCOUNTER_ID

  API service ID of Ocu_GetCounter() function.
- #define OCU_SETABSOLUTETHRESHOLD_ID

  API service ID of Ocu_SetAbsoluteThreshold() function.
- #define OCU_SETRELATIVETHRESHOLD_ID

  API service ID of Ocu_SetRelativeThreshold() function.
- #define OCU_GETVERSIONINFO_ID

  API service ID of Ocu_GetVersionInfo() function.
- #define OCU_DISABLENOTIFICATION_ID

  API service ID of Ocu_DisableNotification() function.
- #define OCU_ENABLENOTIFICATION_ID

  API service ID of Ocu_EnableNotification() function.
- #define OCU_SETCLOCKMODE_ID

  API Ocu_SetClockMode service called with wrong parameter.

Types Reference

- typedef void(* Ocu_NotificationType) (void)

  Ocu channel notification typedef.
- typedef uint32 Ocu_ValueType

  Ocu Value type (the value of the period is platform dependent and thus configurable)
- typedef uint16 Ocu_ChannelType

  Ocu channel type.

Enum Reference

- enum Ocu_ReturnType

  Ocu Return Type.
- enum Ocu_PinActionType

  Edge Pin Action type.
- enum Ocu_PinStateType

  Pin State level.
- enum Ocu_SelectPrescalerType

  Prescaler type.

Function Reference

- void Ocu_Init (const Ocu_ConfigType *ConfigPtr)

  This function initializes the Ocu driver.
- void Ocu_DeInit (void)

  This function deinitializes the Ocu driver.
- Std_ReturnType Ocu_StartChannel (Ocu_ChannelType ChannelNumber)

  This function starts a specified Ocu channel.
- void Ocu_StopChannel (Ocu_ChannelType ChannelNumber)

  This function stops a specified Ocu channel.
- void Ocu_SetPinState (Ocu_ChannelType ChannelNumber, Ocu_PinStateType PinState)

  Service to set immediately the level of the pin associated to an OCU channel.
- void Ocu_SetPinAction (Ocu_ChannelType ChannelNumber, Ocu_PinActionType PinAction)

  Service to indicate the driver what shall be done automatically by hardware (if supported) upon compare match.
- Ocu_ValueType Ocu_GetCounter (Ocu_ChannelType ChannelNumber)

  Service to read the current value of the counter.
- Ocu_ReturnType Ocu_SetAbsoluteThreshold (Ocu_ChannelType ChannelNumber, Ocu_ValueType ReferenceValue, Ocu_ValueType AbsoluteValue)

  Service to set the value of the channel threshold using an absolute input data.
- Ocu_ReturnType Ocu_SetRelativeThreshold (Ocu_ChannelType ChannelNumber, Ocu_ValueType RelativeValue)

  Service to set the value of the channel threshold relative to the current value of the counter.
- void Ocu_DisableNotification (Ocu_ChannelType ChannelNumber)

  This service is used to disable notifications from an OCU channel.
- void Ocu_EnableNotification (Ocu_ChannelType ChannelNumber)

  This service is used to enable notifications from an OCU channel.
- void Ocu_GetVersionInfo (Std_VersionInfoType *versioninfo)

  This function returns Ocu driver version details.
- void Ocu_SetClockMode (Ocu_SelectPrescalerType Prescaler)

  Implementation specific function to change the peripheral clock frequency.
- void Ocu_ProcessChannelNotification (uint16 Channel)

  Ocu common handler to implements generic part of the ISR.

Variables

- const Ocu_ConfigType Ocu_Config_VS_0
  Export Post-Build Ocu configurations.

6.2.2 Data Structure Documentation

6.2.2.1 struct Ocu_ChannelConfigType

Ocu high level channel configuration structure.

Definition at line 387 of file Ocu_Types.h.

Data Fields

Type	Name	Description
boolean	OutputPinEnable	Channel output pin enable.
const Ocu_NotificationType	pfChNotification	Pointer to channel notification function.

6.2.2.2 struct Ocu_ConfigType

Ocu high level configuration structure.

Definition at line 401 of file Ocu_Types.h.

Data Fields

- const Ocu_ChannelType NumChannels
  Number of OCU channels (configured in tresos plugin builder)
- const Ocu_ChannelConfigType(* pOcuChannelsConfig)[]
  Pointer to the OCU channel configuration.
- const Ocu_Ipw_IpConfigType * pcxIpConfig
  Combined IP specific configuration structure.
- const Ocu_ChannelType(* HwToLogicChannelMap)[]
  Index table to translate HW channels to logical used to process interrupts for notifications.
- uint8 CoreId
  Index Core.

6.2.2.2.1 Field Documentation

6.2.2.2.1.1 NumChannels const Ocu_ChannelType NumChannels

Number of OCU channels (configured in tresos plugin builder)

Definition at line 404 of file Ocu_Types.h.

6.2.2.2.1.2 pOcuChannelsConfig const Ocu_ChannelConfigType(* pOcuChannelsConfig)[]

Pointer to the OCU channel configuration.

Definition at line 406 of file Ocu_Types.h.

6.2.2.2.1.3 pcxIpConfig const Ocu_Ipw_IpConfigType* pcxIpConfig

Combined IP specific configuration structure.

Definition at line 408 of file Ocu_Types.h.

6.2.2.2.1.4 HwToLogicChannelMap const Ocu_ChannelType(* HwToLogicChannelMap)[]

Index table to translate HW channels to logical used to process interrupts for notifications.

Definition at line 411 of file Ocu_Types.h.

6.2.2.2.1.5 CoreId uint8 CoreId

Index Core.

Definition at line 415 of file Ocu_Types.h.

6.2.3 Macro Definition Documentation

6.2.3.1 OCU_VALID_CHANNEL_NUM

```
#define OCU_VALID_CHANNEL_NUM
```

Valid channel number.

Definition at line 120 of file Ocu_Types.h.

6.2.3.2 OCU_INVALID_CHANNEL_NUM

```
#define OCU_INVALID_CHANNEL_NUM
```

Invalid channel number.

Definition at line 123 of file Ocu_Types.h.

6.2.3.3 OCU_INSTANCE_INDEX

```
#define OCU_INSTANCE_INDEX
```

Instance ID of this OCU driver.

Definition at line 126 of file Ocu_Types.h.

6.2.3.4 OCU_E_UNINIT

```
#define OCU_E_UNINIT
```

API service used without module initialization.

Errors and exceptions that will be detected by the OCU driver AUTOSAR

Definition at line 134 of file Ocu_Types.h.

6.2.3.5 OCU_E_PARAM_INVALID_CHANNEL

```
#define OCU_E_PARAM_INVALID_CHANNEL
```

API service used with an invalid channel Identifier.

Errors and exceptions that will be detected by the OCU driver AUTOSAR

Definition at line 141 of file Ocu_Types.h.

6.2.3.6 OCU_E_PARAM_INVALID_STATE

```
#define OCU_E_PARAM_INVALID_STATE
```

API Ocu_SetPinState() called with an invalid pin state or when the channel is in the RUNNING state..

Errors and exceptions that will be detected by the OCU driver AUTOSAR

Definition at line 148 of file Ocu_Types.h.

6.2.3.7 OCU_E_PARAM_INVALID_ACTION

```
#define OCU_E_PARAM_INVALID_ACTION
```

API Ocu_SetPinAction() called with an invalid pin action.

Errors and exceptions that will be detected by the OCU driver AUTOSAR

Definition at line 155 of file Ocu_Types.h.

6.2.3.8 OCU_E_NO_VALID_NOTIF

```
#define OCU_E_NO_VALID_NOTIF
```

Usage of Ocu_DisableNotification() or Ocu_EnableNotification() on a channel where a NULL pointer is configured as the notification function.

Errors and exceptions that will be detected by the OCU driver. AUTOSAR

Definition at line 163 of file Ocu_Types.h.

6.2.3.9 OCU_E_ALREADY_INITIALIZED

```
#define OCU_E_ALREADY_INITIALIZED
```

API Ocu_Init() called while the OCU driver has already been initialized.

Errors and exceptions that will be detected by the OCU driver. AUTOSAR

Definition at line 170 of file Ocu_Types.h.

6.2.3.10 OCU_E_PARAM_POINTER

```
#define OCU_E_PARAM_POINTER
```

API Ocu_GetVersionInfo() is called with a NULL parameter.

Errors and exceptions that will be detected by the OCU driver. AUTOSAR

Definition at line 177 of file Ocu_Types.h.

6.2.3.11 OCU_E_BUSY

```
#define OCU_E_BUSY
```

API Ocu_StartChannel() called on a channel that is in state RUNNING.

Errors and exceptions that will be detected by the OCU driver AUTOSAR

Definition at line 184 of file Ocu_Types.h.

6.2.3.12 OCU_E_PARAM_NO_PIN

```
#define OCU_E_PARAM_NO_PIN
```

Ocu_SetPinState() or Ocu_SetPinAction() called for a channel that doesn't have an associated output pin.

Errors and exceptions that will be detected by the OCU driver AUTOSAR

Definition at line 192 of file Ocu_Types.h.

6.2.3.13 OCU_E_INIT_FAILED

```
#define OCU_E_INIT_FAILED
```

API Ocu_Init service called with wrong parameter.

Errors and exceptions that will be detected by the OCU driver AUTOSAR

Definition at line 200 of file Ocu_Types.h.

6.2.3.14 OCU_E_PARAM_INVALID_VALUE

```
#define OCU_E_PARAM_INVALID_VALUE
```

Ocu_SetAbsoluteThreshold() or Ocu_SetRelativeThreshold() called for with a compare match parameter greater than maximum supported counter value for a given channel.

Errors and exceptions that will be detected by the OCU driver Non-AUTOSAR

Definition at line 208 of file Ocu_Types.h.

6.2.3.15 OCU_E_PARAM_CONFIG

```
#define OCU_E_PARAM_CONFIG
```

API Ocu_Init service called with wrong Core number.

Errors and exceptions that will be detected by the OCU driver when Non-AUTOSAR

Definition at line 215 of file Ocu_Types.h.

6.2.3.16 OCU_INIT_ID

```
#define OCU_INIT_ID
```

API service ID of Ocu_Init() function.

Parameters used when raising an error/exception

Definition at line 221 of file Ocu_Types.h.

6.2.3.17 OCU_DEINIT_ID

```
#define OCU_DEINIT_ID
```

API service ID of Ocu_DeInit() function.

Parameters used when raising an error/exception

Definition at line 227 of file Ocu_Types.h.

6.2.3.18 OCU_STARTCHANNEL_ID

```
#define OCU_STARTCHANNEL_ID
```

API service ID of Ocu_StartChannel() function.

Parameters used when raising an error/exception

Definition at line 233 of file Ocu_Types.h.

6.2.3.19 OCU_STOPCHANNEL_ID

```
#define OCU_STOPCHANNEL_ID
```

API service ID of Ocu_StopChannel() function.

Parameters used when raising an error/exception

Definition at line 239 of file Ocu_Types.h.

6.2.3.20 OCU_SETPINSTATE_ID

```
#define OCU_SETPINSTATE_ID
```

API service ID of Ocu_SetPinState() function.

Parameters used when raising an error/exception

Definition at line 244 of file Ocu_Types.h.

6.2.3.21 OCU_SETPINACTION_ID

```
#define OCU_SETPINACTION_ID
```

API service ID of Ocu_SetPinAction() function.

Parameters used when raising an error/exception

Definition at line 250 of file Ocu_Types.h.

6.2.3.22 OCU_GETCOUNTER_ID

```
#define OCU_GETCOUNTER_ID
```

API service ID of Ocu_GetCounter() function.

Parameters used when raising an error/exception

Definition at line 256 of file Ocu_Types.h.

6.2.3.23 OCU_SETABSOLUTETHRESHOLD_ID

```
#define OCU_SETABSOLUTETHRESHOLD_ID
```

API service ID of Ocu_SetAbsoluteThreshold() function.

Parameters used when raising an error/exception

Definition at line 262 of file Ocu_Types.h.

6.2.3.24 OCU_SETRELATIVETHRESHOLD_ID

```
#define OCU_SETRELATIVETHRESHOLD_ID
```

API service ID of Ocu_SetRelativeThreshold() function.

Parameters used when raising an error/exception

Definition at line 268 of file Ocu_Types.h.

6.2.3.25 OCU_GETVERSIONINFO_ID

```
#define OCU_GETVERSIONINFO_ID
```

API service ID of Ocu_GetVersionInfo() function.

Parameters used when raising an error/exception

Definition at line 274 of file Ocu_Types.h.

6.2.3.26 OCU_DISABLENOTIFICATION_ID

```
#define OCU_DISABLENOTIFICATION_ID
```

API service ID of Ocu_DisableNotification() function.

Parameters used when raising an error/exception

Definition at line 280 of file Ocu_Types.h.

6.2.3.27 OCU_ENABLENOTIFICATION_ID

```
#define OCU_ENABLENOTIFICATION_ID
```

API service ID of Ocu_EnableNotification() function.

Parameters used when raising an error/exception

Definition at line 286 of file Ocu_Types.h.

6.2.3.28 OCU_SETCLOCKMODE_ID

```
#define OCU_SETCLOCKMODE_ID
```

API Ocu_SetClockMode service called with wrong parameter.

Parameters used when raising an error/exception

Definition at line 292 of file Ocu_Types.h.

6.2.4 Types Reference

6.2.4.1 Ocu_NotificationType

```
typedef void(* Ocu_NotificationType) (void)
```

Ocu channel notification typedef.

The callback notifications shall be configurable as pointers to user defined functions within the configuration structure.

Definition at line 307 of file Ocu_Types.h.

6.2.4.2 Ocu_ValueType

```
typedef uint32 Ocu_ValueType
```

Ocu Value type (the value of the period is platform dependent and thus configurable)

Type for reading the counter and writing the threshold values (in number of ticks)

Definition at line 314 of file Ocu_Types.h.

6.2.4.3 Ocu_ChannelType

```
typedef uint16 Ocu_ChannelType
```

Ocu channel type.

Numeric identifier of an OCU channel.

Definition at line 320 of file Ocu_Types.h.

6.2.5 Enum Reference

6.2.5.1 Ocu_ReturnType

```
enum Ocu_ReturnType
```

Ocu Return Type.

Return information after setting a new threshold value.

Enumerator

OCU_CM_IN_REF_INTERVAL	The compare match will occur inside the current Reference Interval.
OCU_CM_OUT_REF_INTERVAL	The compare match will not occur inside the current Reference Interval.

Definition at line 329 of file Ocu_Types.h.

6.2.5.2 Ocu_PinActionType

```
enum Ocu_PinActionType
```

Edge Pin Action type.

Automatic action (by hardware) to be performed on a pin attached to an OCU channel.

Enumerator

OCU_SET_HIGH	The channel pin will be set HIGH upon compare match.
OCU_SET_LOW	The channel pin will be set LOW upon compare match.
OCU_TOGGLE	The channel pin will be set to the opposite of its current level HIGH upon compare match.
OCU_DISABLE	The channel pin will remain at its current level upon compare match.

Definition at line 342 of file Ocu_Types.h.

6.2.5.3 Ocu_PinStateType

enum Ocu_PinStateType

Pin State level.

Output state of the pin linked to an OCU channel.

Enumerator

OCU_HIGH	The pin associated to an OCU channel is in high state.
OCU_LOW	The pin associated to an OCU channel is in low state.

Definition at line 361 of file Ocu_Types.h.

6.2.5.4 Ocu_SelectPrescalerType

enum Ocu_SelectPrescalerType

Prescaler type.

This enum specifies the possible types of prescallers used to configure base-clock timers

Enumerator

OCU_PRIMARY_PRESCALER	Selected value is the default/primary prescaler.
OCU_ALTERNATIVE_PRESCALER	Selected value is the alternative configured prescaler.

Definition at line 374 of file Ocu_Types.h.

6.2.6 Function Reference

6.2.6.1 Ocu_Init()

```
void Ocu_Init (
            const Ocu_ConfigType * ConfigPtr )
```

This function initializes the Ocu driver.

The function Ocu_Init shall initialize all internals variables and the used OCU structure of the microcontroller according to the parameters specified in ConfigPtr. Ocu shall initialize the free-running timers that are used by the driver.

If development error detection is enabled, calling the routine with a NULL ConfigPtr, Ocu_Init shall raise the development error OCU_E_INIT_FAILED and return without any action.

If development error detection is enabled, calling the routine Ocu_Init while the OCU driver and hardware are already initialized will cause a development error OCU_E_ALREADY_INITIALIZED. The desired functionality shall be left without any action.

For pre-compile and link time configuration variants, a NULL pointer shall be passed to the initialization routine. In this case the check for this NULL pointer has to be omitted.

If development error detection for the Ocu module is enabled, if any function (except Ocu_Init) is called before Ocu_Init has been called, the called function shall raise development error OCU_E_UNINIT.

Parameters

in	*ConfigPtr*	pointer to OCU top configuration structure

Returns

void

6.2.6.2 Ocu_DeInit()

```
void Ocu_DeInit (
            void  )
```

This function deinitializes the Ocu driver.

The function Ocu_DeInit shall deinitialize the OCU module.

```
The function Ocu_DeInit shall deinitialize the OCU variables and registers that were
initialized by Ocu_Init to a state comparable to their power on reset state.
The function Ocu_DeInit shall disable OCU interrupts and OCU signal edge notifications.
The function Ocu_DeInit shall stop all free-running counters, which are exclusively
used by this driver.
If development error detection for the Ocu module is enabled,
when a development error occurs, the corresponding OCU function shall:
     Report the error to the Development Error Tracer.
```

```
      Skip the desired functionality in order to avoid any corruptions
         of data or hardware registers (this means leave the function without any actions).
If development error detection is enabled for the OCU driver: if a channel is still in
the RUNNING state when the function Ocu_DeInit is called, then the function shall raise
the development error OCU_E_PARAM_INVALID_STATE and return without any action.
If development error detection for the Ocu module is enabled, if any function
(except Ocu_Init) is called before Ocu_Init has been called, the called function
shall raise development error OCU_E_UNINIT.
```

Returns

void

Precondition

Ocu_Init

6.2.6.3 Ocu_StartChannel()

```
Std_ReturnType Ocu_StartChannel (
            Ocu_ChannelType ChannelNumber )
```

This function starts a specified Ocu channel.

The function Ocu_StartChannel shall start an OCU channel by allowing all compare match configured actions to be performed.

The state of the selected channel shall be set to RUNNING If the function Ocu_StartChannel has been successfully performed.

If development error detection is enabled for the OCU driver: If the function Ocu_StartChannel is called on a channel in the state RUNNING, then the function shall raise the error OCU_E_BUSY and return without any action.

If development error detection is enabled for the OCU driver: If the parameter ChannelNumber is invalid (not within the range specified by the configuration), the function Ocu_StartChannel shall raise the error OCU_E_PARAM↩ _INVALID_CHANNEL and return without any action.

If development error detection is enabled for the OCU driver: If the driver is not initialized, the function Ocu_↩ StartChannel shall raise the error OCU_E_UNINIT and return without any action.

Parameters

in	*ChannelNumber*	Ocu channel Id

Returns

E_NOT_OK if call to Ocu_ValidateChannelConfigCall() or Ocu_ValidateRunningState() has failed, E_OK otherwise.

Precondition

Ocu_Init

6.2.6.4 Ocu_StopChannel()

```
void Ocu_StopChannel (
            Ocu_ChannelType ChannelNumber )
```

This function stops a specified Ocu channel.

The function Ocu_StopChannel shall stop an OCU channel by halting compare match configured actions for this channel.

The state of the selected channel shall be set to STOPPED if the function Ocu_StopChannel is successfully performed. If the function Ocu_StopChannel is called on a channel in the state STOPPED, then the function shall leave without any action (no change of the channel state), and shall not raise a development error.

If development error detection is enabled for the OCU driver: If the parameter ChannelNumber is invalid (not within the range specified by the configuration), the function Ocu_StopChannel shall raise the error OCU_E_PARAM↩ _INVALID_CHANNEL and return without any action.

If development error detection is enabled for the OCU driver: If the driver is not initialized, the function Ocu_↩ StopChannel shall raise the error OCU_E_UNINIT and return without any action.

Parameters

in	*ChannelNumber*	- Ocu channel Id

Returns

void

Precondition

Ocu_Init

6.2.6.5 Ocu_SetPinState()

```
void Ocu_SetPinState (
            Ocu_ChannelType ChannelNumber,
            Ocu_PinStateType PinState )
```

Service to set immediately the level of the pin associated to an OCU channel.

The function Ocu_SetPinState shall set the pin associated with the channel to the level indicated by PinState.

The function Ocu_SetPinState shall be used only if the channel is not in the RUNNING state.

If development error detection is enabled for the OCU driver: If the parameter ChannelNumber is invalid (not within the range specified by the configuration), the function Ocu_SetPinState shall raise the error OCU_E_PARAM_↩ INVALID_CHANNEL and return without any action

If development error detection is enabled for the OCU driver: If a pin is not associated with the channel (not defined in the configuration of the channel), the function Ocu_SetPinState shall raise the error OCU_E_PARAM_NO_PIN and return without any action.

If development error detection is enabled for the OCU driver: If the parameter PinState is invalid (not within the range specified by the configuration), the function Ocu_SetPinState shall raise the error OCU_E_PARAM_INV↩ ALID_STATE and return without any action.

If development error detection is enabled for the OCU driver: If the driver is not initialized, the function Ocu_Set↩ PinState shall raise the error OCU_E_UNINIT and return without any action.

If development error detection is enabled for the OCU driver: If the channel is in the RUNNING state, the function Ocu_SetPinState shall raise the error OCU_E_PARAM_INVALID_STATE and return without any action.

Parameters

in	*ChannelNumber*	- Ocu channel Id
in	*PinState*	- Output Pin State

Returns

void

Precondition

Ocu_Init

6.2.6.6 Ocu_SetPinAction()

```
void Ocu_SetPinAction (
            Ocu_ChannelType ChannelNumber,
            Ocu_PinActionType PinAction )
```

Service to indicate the driver what shall be done automatically by hardware (if supported) upon compare match.

The function Ocu_SetPinAction shall set the action to be performed by hardware automatically, at the next compare match in the corresponding OCU channel.

If development error detection is enabled for the OCU driver: If the parameter ChannelNumber is invalid (not within the range specified by the configuration), the function Ocu_SetPinAction shall raise the error OCU_E_PARAM↩ _INVALID_CHANNEL and return without any action.

If development error detection is enabled for the OCU driver: If a pin is not associated with the channel (not defined in the configuration of the channel), the function Ocu_SetPinAction shall raise the error OCU_E_PARAM_NO_PIN and return without any action.

If development error detection is enabled for the OCU driver: If the parameter PinAction is invalid (not within the range specified by the type), the function Ocu_SetPinAction shall raise the error OCU_E_PARAM_INVALID↩ _ACTION and return without any action.

If development error detection is enabled for the OCU driver: If the driver is not initialized, the function Ocu_Set↩ PinAction shall raise the error OCU_E_UNINIT and return without any action.

If a pin is associated with a channel; the relevant action with this pin shall be performed upon compare match.

Parameters

in	*ChannelNumber*	- Ocu channel Id
in	*PinAction*	- Pin Action (OCU_SET_LOW, OCU_SET_HIGH, OCU_TOGGLE, OCU_DISABLE)

Returns

void

Precondition

Ocu_Init

6.2.6.7 Ocu_GetCounter()

```
Ocu_ValueType Ocu_GetCounter (
            Ocu_ChannelType ChannelNumber )
```

Service to read the current value of the counter.

The function Ocu_GetCounter shall read and return the value of the counter of the channel indicated by Channel↩ Number.

If development error detection is enabled for the OCU driver: If the parameter ChannelNumber is invalid (not within the range specified by the configuration), the function Ocu_GetCounter shall raise the error OCU_E_PARAM_↩ INVALID_CHANNEL and shall return the value 0.

If development error detection is enabled for the OCU driver: if the driver is not initialized, then the function Ocu_GetCounterValue shall raise the error OCU_E_UNINIT and shall return the value 0.

Parameters

in	*ChannelNumber*	Ocu channel Id

Returns

Ocu_ValueType the value of the hardware counter.

Precondition

Ocu_Init

6.2.6.8 Ocu_SetAbsoluteThreshold()

```
Ocu_ReturnType Ocu_SetAbsoluteThreshold (
            Ocu_ChannelType ChannelNumber,
            Ocu_ValueType ReferenceValue,
            Ocu_ValueType AbsoluteValue )
```

Service to set the value of the channel threshold using an absolute input data.

The function Ocu_SetAbsoluteThreshold shall set the channel threshold (the compare value) to the value given by AbsoluteValue.

If development error detection is enabled for the OCU driver: If the driver is not initialized, the function Ocu_Set↩ AbsoluteThreshold shall raise the error OCU_E_UNINIT and return without any action.

If development error detection is enabled for the OCU driver: If the parameter ChannelNumber is invalid (not within the range specified by the configuration), the function Ocu_SetAbsoluteThreshold shall raise the error OCU_E_↩ PARAM_INVALID_CHANNEL and return without any action.

After setting a new threshold value, the API Ocu_SetAbsoluteThreshold shall return a status to inform the caller whether the compare match will occur (or has already occurred) during the current Reference Interval, as a result of setting the new threshold value. (see OCU_SWS for details)

Parameters

in	*ChannelNumber*	- Ocu channel Id
in	*ReferenceValue*	- Value given by the upper layer and used as a base to determine whether to call the notification before the function exits or not.
in	*AbsoluteValue*	- Value to compare with the content of the counter. This value is in ticks.

Returns

Ocu_ReturnType - Tells the caller whether the compare match will occur (or has already occurred) during the current Reference Interval, as a result of setting the new threshold value.

Precondition

Ocu_Init

6.2.6.9 Ocu_SetRelativeThreshold()

```
Ocu_ReturnType Ocu_SetRelativeThreshold (
            Ocu_ChannelType ChannelNumber,
            Ocu_ValueType RelativeValue )
```

Service to set the value of the channel threshold relative to the current value of the counter.

The function Ocu_SetAbsoluteThreshold shall set the channel threshold (the compare value) to the value given by RelativeValue plus the current counter value read from hw.

After setting a new threshold value, the API Ocu_SetRelativeThreshold shall return a status to inform the caller whether the compare match will occur (or has already occurred) during the current Reference Interval, as a result of setting the new threshold value.

If development error detection is enabled for the OCU driver: If the driver is not initialized, the function Ocu_Set↩ RelativeThreshold shall raise the error OCU_E_UNINIT and return without any action.

After setting a new threshold value, the API Ocu_SetRelativeThreshold shall return a status to inform the caller whether the compare match will occur (or has already occurred) during the current Reference Interval, as a result of setting the new threshold value.

Parameters

in	*ChannelNumber*	- Ocu channel Id
in	*RelativeValue*	- Value to use for computing the new threshold.

Returns

Ocu_ReturnType - Tells the caller whether the compare match will occur (or has already occurred) during the current Reference Interval, as a result of setting the new threshold value.

Precondition

Ocu_Init

6.2.6.10 Ocu_DisableNotification()

```
void Ocu_DisableNotification (
            Ocu_ChannelType ChannelNumber )
```

This service is used to disable notifications from an OCU channel.

The function Ocu_DisableNotification shall disable the OCU compare match notification.

If development error detection is enabled for the OCU driver: If the driver is not initialized, the function Ocu_↩ DisableNotification shall raise the error OCU_E_UNINIT and return without any action.

If development error detection is enabled for the OCU driver: If the parameter Channel is invalid (not within the range specified by configuration), the function Ocu_DisableNotification shall raise the error OCU_E_PARAM_↩ INVALID_CHANNEL and return without any action.

If development error detection is enabled for the OCU driver: If the notification function is the NULL pointer, the function Ocu_DisableNotification shall raise the error OCU_E_NO_VALID_NOTIF and return without any action.

Parameters

in	*ChannelNumber*	- Ocu channel Id

Returns

void

Precondition

Ocu_Init

6.2.6.11 Ocu_EnableNotification()

```
void Ocu_EnableNotification (
            Ocu_ChannelType ChannelNumber )
```

This service is used to enable notifications from an OCU channel.

The function Ocu_EnableNotification shall enable the OCU compare match notification of the indexed channel.

If development error detection is enabled for the OCU driver: If the driver is not initialized, the function Ocu_↩ EnableNotification shall raise the error OCU_E_UNINIT and return without any action.

If development error detection is enabled for the OCU driver: If the parameter Channel is invalid (not within the range specified by configuration), then the function Ocu_EnableNotification shall raise the error OCU_E_PARA↩ M_INVALID_CHANNEL and return without any action.

If development error detection is enabled for the OCU driver: If the notification function is the NULL pointer, the function Ocu_EnableNotification shall raise the error OCU_E_NO_VALID_NOTIF and return without any action.

Parameters

in	*ChannelNumber*	- Ocu channel Id
in	*Notification*	- notification type to be enabled

Returns

void

Precondition

Ocu_Init

6.2.6.12 Ocu_GetVersionInfo()

```
void Ocu_GetVersionInfo (
            Std_VersionInfoType * versioninfo )
```

This function returns Ocu driver version details.

The function Ocu_GetVersionInfo shall return the version information of this module. The version information includes: Module Id, Vendor Id, Vendor specific version number.

Parameters

in,out	*versioninfo*	- pointer to Std_VersionInfoType output variable

Returns

void

6.2.6.13 Ocu_SetClockMode()

```
void Ocu_SetClockMode (
            Ocu_SelectPrescalerType Prescaler )
```

Implementation specific function to change the peripheral clock frequency.

This function is useful to set the prescalers that divide the OCU channels clock frequency.

Parameters

in	*Prescaler*	- prescaler type Possible values: • OCU_PRIMARY_PRESCALER for normal clock mode or • OCU_ALTERNATIVE_PRESCALER for a different prescaler for the driver

6.2.6.14 Ocu_ProcessChannelNotification()

```
void Ocu_ProcessChannelNotification (
            uint16 Channel )
```

Ocu common handler to implements generic part of the ISR.

Generic function used by all interrupt service routines to call notification

Parameters

in	*Channel*	- logic channel number

6.2.7 Variable Documentation

6.2.7.1 Ocu_Config_VS_0

```
const Ocu_ConfigType Ocu_Config_VS_0  [extern]
```

Export Post-Build Ocu configurations.

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.