User Manual

for S32K1 OCU Driver

Document Number: UM2OCUASR4.4 Rev0000R1.0.1 Rev. 1.0

Chapter 1

Revision History

Revision	Date	Author	Description
1.0	24.02.2022	NXP RTD Team	Prepared for release RTD S32K1 Version 1.0.1

Chapter 2

Introduction

This User Manual describes NXP Semiconductor AUTOSAR Ocu for S32. 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:

- s32k116_qfn32
- s32k116_lqfp48
- s32k118_lqfp48
- s32k118_lqfp64
- s32k142_lqfp48
- s32k142_lqfp64
- s32k142_lqfp100
- s32k142w_lqfp48
- s32k142w_lqfp64
- s32k144_lqfp48

- s32k144_lqfp64
- s32k144_lqfp100
- s32k144_mapbga100
- s32k144w_lqfp48
- s32k144w_lqfp64
- s32k146_lqfp64
- s32k146_lqfp100
- s32k146_mapbga100
- s32k146_lqfp144
- s32k148_lqfp100
- s32k148_mapbga100
- s32k148_lqfp144
- s32k148_lqfp176

All of the above microcontroller devices are collectively named as S32K1.

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 4.↩ 4.0
2	S32K1xx Series Reference Manual	Rev. 14, 09/2021
3	S32K1xx Data Sheet	Rev. 14, 08/2021
4	Errata S32K116_0N96V	Rev. 22/OCT/2021
5	Errata S32K118_0N97V	Rev. 22/OCT/2021
6	Errata S32K142_0N33V	Rev. 22/OCT/2021
7	Errata S32K144_0N57U	Rev. 22/OCT/2021
8	Errata S32K144W_0P64A	Rev. 22/OCT/2021
9	Errata S32K146_0N73V	Rev. 22/OCT/2021
10	Errata S32K148_0N20V	Rev. 22/OCT/2021

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 module provides the service for initializing the whole PORT structure of the microcontroller. Many ports and port pins can be assigned to various functionalities, e.g.

- General purpose I/O
- ADC
- SPI

- SCI
- PWM
- CAN
- LIN
- etc

For this reason, there is an overall configuration and initialization of this port structure. The configuration and mode of these port pins is microcontroller and ECU dependent. Port initialisation data are written to each port as efficiently as possible. This PORT driver module completes the overall configuration and initialisation of the port structure which is used in the DIO driver module. Therefore, the DIO driver works on pins and ports which are configured by the PORT driver. The PORT driver is initialised prior to use of the DIO functions. Otherwise DIO functions will exhibit undefined behaviour.

3.3 Hardware Resources

#	Hardware IP	Description
1	FTM	FlexTimer

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/A	Not available
N/T	Not testable
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

Requirement	Status	Description	Notes
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
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_00156	N/S	These requirements are not applicable to this specification	Not a requirement
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

None.

3.6 Driver usage and configuration tips

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

3.7 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 HwChannel
      - Parameter OcuDefaultThreshold
      - Parameter OcuHardwareTriggeredAdc
      - Parameter OcuHardwareTriggeredDMA
      - Parameter OcuMaxCounterValue
      - Parameter OcuNotification
      - Parameter OcuOuptutPinUsed
      - Parameter OcuOutputPinDefaultState
      - Parameter OcuOutputPinAction
      - Parameter OcuUseMcuReferencePoint
      - Parameter OcuChannelTickDuration
      - Reference OcuChannelEcucPartitionRef
      - Reference OcuMcuClockReferencePoint
    - Container OcuHWSpecificSettings
      - Parameter OcuHardwareChannelId

      - Parameter OcuFtmModule
      - Parameter OcuClockSource
      - Parameter OcuPrescale
      - Parameter OcuPrescale_Alternate
      - Parameter OcuDebugMode
  - Container OcuGeneral
    - Parameter OcuDevErrorDetect
    - Parameter OcuEnableDualClockMode
    - Parameter OcuEnableUserModeSupport
    - Parameter OcuMulticoreEnabled
    - 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	1
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.

NoteThis feature 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	Infinite
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.

NoteHardware channel should be enumaration paramenter and has been replaced by OcuHardwareChannel

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 HwChannel

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

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-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	FTM_0_CH_0
literals	['FTM_0_CH_0', 'FTM_0_CH_1', 'FTM_0_CH_2', 'FTM_0_CH_3', 'FT↩ M_0_CH_4', 'FTM_0_CH_5', 'FTM_0_CH_6', 'FTM_0_CH_7', 'FTM_↩ 1_CH_0', 'FTM_1_CH_1', 'FTM_1_CH_2', 'FTM_1_CH_3', 'FTM_1_C↩ H_4', 'FTM_1_CH_5', 'FTM_1_CH_6', 'FTM_1_CH_7', 'FTM_2_CH_0', 'FTM_2_CH_1', 'FTM_2_CH_2', 'FTM_2_CH_3', 'FTM_2_CH_4', 'FT↩ M_2_CH_5', 'FTM_2_CH_6', 'FTM_2_CH_7', 'FTM_3_CH_0', 'FTM_↩ 3_CH_1', 'FTM_3_CH_2', 'FTM_3_CH_3', 'FTM_3_CH_4', 'FTM_3_C↩ H_5', 'FTM_3_CH_6', 'FTM_3_CH_7', 'FTM_4_CH_0', 'FTM_4_CH_1', 'FTM_4_CH_2', 'FTM_4_CH_3', 'FTM_4_CH_4', 'FTM_4_CH_5', 'FT↩ M_4_CH_6', 'FTM_4_CH_7', 'FTM_5_CH_0', 'FTM_5_CH_1', 'FTM_↩ 5_CH_2', 'FTM_5_CH_3', 'FTM_5_CH_4', 'FTM_5_CH_5', 'FTM_5_C↩ H_6', 'FTM_5_CH_7', 'FTM_6_CH_0', 'FTM_6_CH_1', 'FTM_6_CH_2', 'FTM_6_CH_3', 'FTM_6_CH_4', 'FTM_6_CH_5', 'FTM_6_CH_6', 'FT↩ M_6_CH_7', 'FTM_7_CH_0', 'FTM_7_CH_1', 'FTM_7_CH_2', 'FTM_↩ 7_CH_3', 'FTM_7_CH_4', 'FTM_7_CH_5', 'FTM_7_CH_6', 'FTM_7_C↩ H_7']

4.17 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	65535
min	0

4.18 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
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 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

4.20 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
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	65535
min	1

4.21 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

4.22 Parameter OcuOuptutPinUsed

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

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

4.23 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']

4.24 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
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']

4.25 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.

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

4.26 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 Unit the Ocu Channel Tick Duration is nano seconds(ns).

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	9223372036854775807
max	N/A
min	1

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
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 Ftm 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

This container contains Ocu-specific parameters for selecting the clock source and setting optional prescalers if supported by hardware.

Included subcontainers:

- None

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

4.30 Parameter OcuHardwareChannelId

Channel ID of the OCU hardware channel. This ID is used to match with OcuChannel/OcuAssignedHardwareChannel

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

4.31 Parameter OcuFtmModule

Selects one Ftm modules available on the platform.

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

Property	Value
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	FTM_0
literals	['FTM_0', 'FTM_1', 'FTM_2', 'FTM_3', 'FTM_4', 'FTM_5', 'FTM_6', 'F↩ TM_7']

4.32 Parameter OcuClockSource

The OCU driver specific clock input for the unit can statically be configured to select different clock sources if provided by hardware. Enumeration literals are 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_EXTERNAL_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

Property	Value
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	DIV1
literals	['DIV1', 'DIV2', 'DIV4', 'DIV8', 'DIV16', 'DIV32', 'DIV64', 'DIV128']

4.34 Parameter OcuPrescale_Alternate

Select input count source (clock) used for this Ftm module. 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	DIV1
literals	['DIV1', 'DIV2', 'DIV4', 'DIV8', 'DIV16', 'DIV32', 'DIV64', 'DIV128']

4.35 Parameter OcuDebugMode

OcuDebugMode

These bits allow select the behavior of the FTM counter, the CH(n)F bit, the channels output, and the writing to the MOD, CNTIN, and C(n)V registers (wave form register)

CNT_ON_OUTPUTS_ON - Continue with normal operation during debug mode.

CNT_STOPED_OUTPUTS_FROZEN - Counter is stopped, channels outputs are frozen, writes to wave form registers bypass the registers buffers

CNT_STOPED_OUTPUTS_SAFE - Counter is stopped, channels outputs are forced to their safe value according to POLn bit, writes to wave form registers bypass the registers buffers

CNT_STOPED_FLAG_SET - Counter is stopped, channels outputs operate with its functionality, writes to wave form registers bypass the registers buffers

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	CNT_ON_OUTPUTS_ON
literals	['CNT_ON_OUTPUTS_ON', 'CNT_STOPED_OUTPUTS_FROZEN', 'CN↩ T_STOPED_OUTPUTS_SAFE', 'CNT_STOPED_FLAG_SET']

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 OcuDevErrorDetect

Switch for enabling the development error detection.

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

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

4.38 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

4.39 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 Ftm 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

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF

Property	Value
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

4.40 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

4.41 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

4.42 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

4.43 Container OcuHwResourceConfig

List of HW interrupts available for the entire platform.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	64
upperMultiplicity	64
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.44 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
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	FTM_0_CH_1

Property	Value
literals	['FTM_0_CH_0', 'FTM_0_CH_1', 'FTM_0_CH_2', 'FTM_0_CH_3', 'FT↩ M_0_CH_4', 'FTM_0_CH_5', 'FTM_0_CH_6', 'FTM_0_CH_7', 'FTM_↩ 1_CH_0', 'FTM_1_CH_1', 'FTM_1_CH_2', 'FTM_1_CH_3', 'FTM_1_C↩ H_4', 'FTM_1_CH_5', 'FTM_1_CH_6', 'FTM_1_CH_7', 'FTM_2_CH_0', 'FTM_2_CH_1', 'FTM_2_CH_2', 'FTM_2_CH_3', 'FTM_2_CH_4', 'FT↩ M_2_CH_5', 'FTM_2_CH_6', 'FTM_2_CH_7', 'FTM_3_CH_0', 'FTM_↩ 3_CH_1', 'FTM_3_CH_2', 'FTM_3_CH_3', 'FTM_3_CH_4', 'FTM_3_C↩ H_5', 'FTM_3_CH_6', 'FTM_3_CH_7', 'FTM_4_CH_0', 'FTM_4_CH_1', 'FTM_4_CH_2', 'FTM_4_CH_3', 'FTM_4_CH_4', 'FTM_4_CH_5', 'FT↩ M_4_CH_6', 'FTM_4_CH_7', 'FTM_5_CH_0', 'FTM_5_CH_1', 'FTM_↩ 5_CH_2', 'FTM_5_CH_3', 'FTM_5_CH_4', 'FTM_5_CH_5', 'FTM_5_C↩ H_6', 'FTM_5_CH_7', 'FTM_6_CH_0', 'FTM_6_CH_1', 'FTM_6_CH_2', 'FTM_6_CH_3', 'FTM_6_CH_4', 'FTM_6_CH_5', 'FTM_6_CH_6', 'FT↩ M_6_CH_7', 'FTM_7_CH_0', 'FTM_7_CH_1', 'FTM_7_CH_2', 'FTM_↩ 7_CH_3', 'FTM_7_CH_4', 'FTM_7_CH_5', 'FTM_7_CH_6', 'FTM_7_C↩ H_7']

4.45 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

4.46 Parameter OcuChannelIsUsed

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

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

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

4.47 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

4.48 Parameter ArReleaseMajorVersion

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

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

Property	Value
valueConfigClasses	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
defaultValue	4
max	4
min	4

4.49 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	4
max	4
min	4

4.50 Parameter ArReleaseRevisionVersion

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

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

Property	Value
defaultValue	0
max	0
min	0

4.51 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

4.52 Parameter SwMajorVersion

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

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

4.53 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

4.54 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
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
defaultValue	1
max	1
min	1

4.55 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	

4.56 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

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 FTM driver header file.

Ocu FTM driver header file specific to OCU driver.

Ocu FTM header file support for OCU driver.

FTM driver interface.

FTM specific defines which need to be exported to external application

Data Structures

- struct Ftm_Ocu_Ip_ChannelConfigType

  FTM IP channel specific configuration structure for the Ocu functionality. More...
- struct Ftm_Ocu_Ip_SpecificIpConfigType

  FTM IP module specific configuration structure for the Ocu functionality. More...
- struct Ftm_Ocu_Ip_ModuleConfigType

  FTM IP specific configuration structure type. More...

Macros

- #define FTM_COMBINE_CH_CTRL_WIDTH

  Combine control width for channel combine function.
- #define OCU_FTM_INVALID_CHANNEL_NUM

  invalid FTM channel number
- #define FTM_0_CH_0

  FTM 0 channel 0.
- #define FTM_0_CH_1

  FTM 0 channel 1.
- #define FTM_0_CH_2

FTM 0 channel 2.
- #define FTM_0_CH_3

  FTM 0 channel 3.
- #define FTM_0_CH_4

  FTM 0 channel 4.
- #define FTM_0_CH_5

  FTM 0 channel 5.
- #define FTM_0_CH_6

  FTM 0 channel 6.
- #define FTM_0_CH_7

  FTM 0 channel 7.
- #define FTM_1_CH_0

  FTM 1 channel 0.
- #define FTM_1_CH_1

  FTM 1 channel 1.
- #define FTM_1_CH_2

  FTM 1 channel 2.
- #define FTM_1_CH_3

  FTM 1 channel 3.
- #define FTM_1_CH_4

  FTM 1 channel 4.
- #define FTM_1_CH_5

  FTM 1 channel 5.
- #define FTM_1_CH_6

  FTM 1 channel 6.
- #define FTM_1_CH_7

  FTM 1 channel 7.
- #define FTM_2_CH_0

  FTM 2 channel 0.
- #define FTM_2_CH_1

  FTM 2 channel 1.
- #define FTM_2_CH_2

  FTM 2 channel 2.
- #define FTM_2_CH_3

  FTM 2 channel 3.
- #define FTM_2_CH_4

  FTM 2 channel 4.
- #define FTM_2_CH_5

  FTM 2 channel 5.
- #define FTM_2_CH_6

  FTM 2 channel 6.
- #define FTM_2_CH_7

  FTM 2 channel 7.
- #define FTM_3_CH_0

  FTM 3 channel 0.
- #define FTM_3_CH_1

  FTM 3 channel 1.

- #define FTM_3_CH_2
  FTM 3 channel 2.
- #define FTM_3_CH_3
  FTM 3 channel 3.
- #define FTM_3_CH_4
  FTM 3 channel 4.
- #define FTM_3_CH_5
  FTM 3 channel 5.
- #define FTM_3_CH_6
  FTM 3 channel 6.
- #define FTM_3_CH_7
  FTM 3 channel 7.
- #define FTM_4_CH_0
  FTM 4 channel 0.
- #define FTM_4_CH_1
  FTM 4 channel 1.
- #define FTM_4_CH_2
  FTM 4 channel 2.
- #define FTM_4_CH_3
  FTM 4 channel 3.
- #define FTM_4_CH_4
  FTM 4 channel 4.
- #define FTM_4_CH_5
  FTM 4 channel 5.
- #define FTM_4_CH_6
  FTM 4 channel 6.
- #define FTM_4_CH_7
  FTM 4 channel 7.
- #define FTM_5_CH_0
  FTM 5 channel 0.
- #define FTM_5_CH_1
  FTM 5 channel 1.
- #define FTM_5_CH_2
  FTM 5 channel 2.
- #define FTM_5_CH_3
  FTM 5 channel 3.
- #define FTM_5_CH_4
  FTM 5 channel 4.
- #define FTM_5_CH_5
  FTM 5 channel 5.
- #define FTM_5_CH_6
  FTM 5 channel 6.
- #define FTM_5_CH_7
  FTM 5 channel 7.
- #define FTM_6_CH_0
  FTM 6 channel 0.
- #define FTM_6_CH_1

FTM 6 channel 1.
- #define FTM_6_CH_2

  FTM 6 channel 2.
- #define FTM_6_CH_3

  FTM 6 channel 3.
- #define FTM_6_CH_4

  FTM 6 channel 4.
- #define FTM_6_CH_5

  FTM 6 channel 5.
- #define FTM_6_CH_6

  FTM 6 channel 6.
- #define FTM_6_CH_7

  FTM 6 channel 7.
- #define FTM_7_CH_0

  FTM 7 channel 0.
- #define FTM_7_CH_1

  FTM 7 channel 1.
- #define FTM_7_CH_2

  FTM 7 channel 2.
- #define FTM_7_CH_3

  FTM 7 channel 3.
- #define FTM_7_CH_4

  FTM 7 channel 4.
- #define FTM_7_CH_5

  FTM 7 channel 5.
- #define FTM_7_CH_6

  FTM 7 channel 6.
- #define FTM_7_CH_7

  FTM 7 channel 7.
- #define FTM_BDM_MODE_00

  Options for the FlexTimer behavior in BDM Mode.
- #define FTM_BDM_MODE_01
- #define FTM_BDM_MODE_10
- #define FTM_BDM_MODE_11
- #define FTM_CLOCK_SOURCE_NONE

  FlexTimer Clock Source Selection.
- #define FTM_CLOCK_SOURCE_SYSTEMCLK
- #define FTM_CLOCK_SOURCE_FIXEDCLK
- #define FTM_CLOCK_SOURCE_EXTERNALCLK
- #define FTM_CLOCK_DIVID_BY_1

  FlexTimer Prescale Factor Selections bits (PS).
- #define FTM_CLOCK_DIVID_BY_2
- #define FTM_CLOCK_DIVID_BY_4
- #define FTM_CLOCK_DIVID_BY_8
- #define FTM_CLOCK_DIVID_BY_16
- #define FTM_CLOCK_DIVID_BY_32
- #define FTM_CLOCK_DIVID_BY_64
- #define FTM_CLOCK_DIVID_BY_128

- #define FTM_OCU_SET_LOW

  Edge Pin Action values.
- #define FTM_OCU_SET_HIGH

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

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

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

  Pin State levels.
- #define OCU_FTM_HIGH

  Ocu Pin level is logic high.
- #define FTM_OCU_PRIMARY_PRESCALER

  Prescaler values.
- #define FTM_OCU_ALTERNATIVE_PRESCALER

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

  Ocu Return values.
- #define OCU_FTM_CM_OUT_REF_INTERVAL

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

Types Reference

- typedef uint8 Ftm_Ocu_Ip_PinActionType

  Edge Pin Action type.
- typedef uint8 Ftm_Ocu_Ip_PinStateType

  Pin State level.
- typedef uint8 Ftm_Ocu_Ip_SelectPrescalerType

  Prescaler type.
- typedef uint8 Ftm_Ocu_Ip_ReturnType

  Ocu Return Type.
- typedef FTM_Type Ftm_Ocu_Ip_xRegLayoutType
- typedef void(∗ Ftm_Ocu_Ip_CallbackType) (uint16 CallbackParam)

  Channel notification typedef.

Enum Reference

- enum Ftm_Ocu_Ip_OutputCompareModeType

  FlexTimer output compare edge mode. Toggle, clear or set.

Function Reference

- void Ftm_Ocu_Ip_Init (const Ftm_Ocu_Ip_ModuleConfigType *const pFtmIpConfig)

  This function initializes all internals variables of the driver.
- void Ftm_Ocu_Ip_StartChannel (uint8 InstNum, uint8 ChNum)

  Ocu driver function for starting the FTM timer channel.
- void Ftm_Ocu_Ip_StopChannel (uint8 InstNum, uint8 ChNum)

  Ocu driver function for stopping the FTM timer channel.

- #define FTM_MOD_MASK_U8

  FTM channels defines.

- #define OCU_FTM_OUTPUTDISABLED

  FTM register define used to configure counter clock source.

6.1.2 Data Structure Documentation

6.1.2.1 struct Ftm_Ocu_Ip_ChannelConfigType

FTM IP channel specific configuration structure for the Ocu functionality.

Definition at line 606 of file Ftm_Ocu_Ip_Types.h.

Data Fields

Type	Name	Description
const uint16	HwChannel	Assigned FTM channel Id.
const uint16	DefaultThreshold	Compare match threshold for the current channel.
const uint8	ChannelControlValue	FTM channel parameters. Bits 7 --> 1: Output Pin is Used / 0: Output Pin is not used Bit 6 --> Default Pin State: 0: PIN_LOW; 1: PIN_HIGH Bits 5 .. 4 --> Pin Action behaviour on compare match: SET_PIN_LOW; 2: SET_PIN_HIGH; 1: PIN_TOGLE; 0: OUTPUT_DISABLE Bits 3 .. 0 --> Reserved for future use

6.1.2.2 struct Ftm_Ocu_Ip_SpecificIpConfigType

FTM IP module specific configuration structure for the Ocu functionality.

Definition at line 627 of file Ftm_Ocu_Ip_Types.h.

Data Fields

Type	Name	Description
const uint8	ModuleId	Assigned FTM module Id.
const uint16	MaxCounterValue	Assigned FTM module Id.
const uint8	ModuleControlValue	FTM module parameters. Bits 7 .. 6 --> Clock source Bits 5 .. 3 --> Normal prescale Bit 2 .. 1 --> Debug mode config Bits 0 --> Reserved for future use

6.1.2.3 struct Ftm_Ocu_Ip_ModuleConfigType

FTM IP specific configuration structure type.

Definition at line 648 of file Ftm_Ocu_Ip_Types.h.

Data Fields

Type	Name	Description
const uint8	NumChannels	Number of FTM channels in the Ocu configuration.
const uint8	NumModules	Number of FTM modules in the Ocu configuration.
const Ftm_Ocu_Ip_ChannelConfigType(∗	pcxChannelsConfig)[ ]	Pointer to the configured channels for FTM.
const Ftm_Ocu_Ip_SpecificIpConfigType(∗	pcxModulesConfig)[ ]	Pointer to the configured channels for FTM.

6.1.3 Macro Definition Documentation

6.1.3.1 FTM_COMBINE_CH_CTRL_WIDTH

```
#define FTM_COMBINE_CH_CTRL_WIDTH
```

Combine control width for channel combine function.

Definition at line 89 of file Ftm_Ocu_Ip_Types.h.

6.1.3.2 OCU_FTM_INVALID_CHANNEL_NUM

```
#define OCU_FTM_INVALID_CHANNEL_NUM
```

invalid FTM channel number

Definition at line 92 of file Ftm_Ocu_Ip_Types.h.

6.1.3.3 FTM_MOD_MASK_U8

```
#define FTM_MOD_MASK_U8
```

FTM channels defines.

There are defines used for the FTM channel encoding -> channel_id

Note

> a FTM module generally has only 6 channels so by using this method there will be gaps of 2 channels between different FTM modules because [5 bit | 3 bit] [module id: 31 ... 0 | channel id: 7 ... 0]

Definition at line 104 of file Ftm_Ocu_Ip_Types.h.

6.1.3.4 FTM_0_CH_0

```
#define FTM_0_CH_0
```

FTM 0 channel 0.

Definition at line 117 of file Ftm_Ocu_Ip_Types.h.

6.1.3.5 FTM_0_CH_1

```
#define FTM_0_CH_1
```

FTM 0 channel 1.

Definition at line 121 of file Ftm_Ocu_Ip_Types.h.

6.1.3.6 FTM_0_CH_2

```
#define FTM_0_CH_2
```

FTM 0 channel 2.

Definition at line 125 of file Ftm_Ocu_Ip_Types.h.

6.1.3.7 FTM_0_CH_3

```
#define FTM_0_CH_3
```

FTM 0 channel 3.

Definition at line 129 of file Ftm_Ocu_Ip_Types.h.

6.1.3.8 FTM_0_CH_4

```
#define FTM_0_CH_4
```

FTM 0 channel 4.

Definition at line 133 of file Ftm_Ocu_Ip_Types.h.

6.1.3.9 FTM_0_CH_5

```
#define FTM_0_CH_5
```

FTM 0 channel 5.

Definition at line 137 of file Ftm_Ocu_Ip_Types.h.

6.1.3.10 FTM_0_CH_6

```
#define FTM_0_CH_6
```

FTM 0 channel 6.

Definition at line 141 of file Ftm_Ocu_Ip_Types.h.

6.1.3.11 FTM_0_CH_7

```
#define FTM_0_CH_7
```

FTM 0 channel 7.

Definition at line 145 of file Ftm_Ocu_Ip_Types.h.

6.1.3.12 FTM_1_CH_0

```
#define FTM_1_CH_0
```

FTM 1 channel 0.

Definition at line 150 of file Ftm_Ocu_Ip_Types.h.

6.1.3.13 FTM_1_CH_1

```
#define FTM_1_CH_1
```

FTM 1 channel 1.

Definition at line 154 of file Ftm_Ocu_Ip_Types.h.

6.1.3.14 FTM_1_CH_2

```
#define FTM_1_CH_2
```

FTM 1 channel 2.

Definition at line 158 of file Ftm_Ocu_Ip_Types.h.

6.1.3.15 FTM_1_CH_3

```
#define FTM_1_CH_3
```

FTM 1 channel 3.

Definition at line 162 of file Ftm_Ocu_Ip_Types.h.

6.1.3.16 FTM_1_CH_4

```
#define FTM_1_CH_4
```

FTM 1 channel 4.

Definition at line 166 of file Ftm_Ocu_Ip_Types.h.

6.1.3.17 FTM_1_CH_5

```
#define FTM_1_CH_5
```

FTM 1 channel 5.

Definition at line 170 of file Ftm_Ocu_Ip_Types.h.

6.1.3.18 FTM_1_CH_6

```
#define FTM_1_CH_6
```

FTM 1 channel 6.

Definition at line 174 of file Ftm_Ocu_Ip_Types.h.

6.1.3.19 FTM_1_CH_7

```
#define FTM_1_CH_7
```

FTM 1 channel 7.

Definition at line 178 of file Ftm_Ocu_Ip_Types.h.

6.1.3.20 FTM_2_CH_0

```
#define FTM_2_CH_0
```

FTM 2 channel 0.

Definition at line 183 of file Ftm_Ocu_Ip_Types.h.

6.1.3.21 FTM_2_CH_1

```
#define FTM_2_CH_1
```

FTM 2 channel 1.

Definition at line 187 of file Ftm_Ocu_Ip_Types.h.

6.1.3.22 FTM_2_CH_2

```
#define FTM_2_CH_2
```

FTM 2 channel 2.

Definition at line 191 of file Ftm_Ocu_Ip_Types.h.

6.1.3.23 FTM_2_CH_3

```
#define FTM_2_CH_3
```

FTM 2 channel 3.

Definition at line 195 of file Ftm_Ocu_Ip_Types.h.

6.1.3.24 FTM_2_CH_4

```
#define FTM_2_CH_4
```

FTM 2 channel 4.

Definition at line 199 of file Ftm_Ocu_Ip_Types.h.

6.1.3.25 FTM_2_CH_5

```
#define FTM_2_CH_5
```

FTM 2 channel 5.

Definition at line 203 of file Ftm_Ocu_Ip_Types.h.

6.1.3.26 FTM_2_CH_6

```
#define FTM_2_CH_6
```

FTM 2 channel 6.

Definition at line 207 of file Ftm_Ocu_Ip_Types.h.

6.1.3.27 FTM_2_CH_7

```
#define FTM_2_CH_7
```

FTM 2 channel 7.

Definition at line 211 of file Ftm_Ocu_Ip_Types.h.

6.1.3.28 FTM_3_CH_0

```
#define FTM_3_CH_0
```

FTM 3 channel 0.

Definition at line 216 of file Ftm_Ocu_Ip_Types.h.

6.1.3.29 FTM_3_CH_1

```
#define FTM_3_CH_1
```

FTM 3 channel 1.

Definition at line 220 of file Ftm_Ocu_Ip_Types.h.

6.1.3.30 FTM_3_CH_2

```
#define FTM_3_CH_2
```

FTM 3 channel 2.

Definition at line 224 of file Ftm_Ocu_Ip_Types.h.

6.1.3.31 FTM_3_CH_3

```
#define FTM_3_CH_3
```

FTM 3 channel 3.

Definition at line 228 of file Ftm_Ocu_Ip_Types.h.

6.1.3.32 FTM_3_CH_4

```
#define FTM_3_CH_4
```

FTM 3 channel 4.

Definition at line 232 of file Ftm_Ocu_Ip_Types.h.

6.1.3.33 FTM_3_CH_5

```
#define FTM_3_CH_5
```

FTM 3 channel 5.

Definition at line 236 of file Ftm_Ocu_Ip_Types.h.

6.1.3.34 FTM_3_CH_6

```
#define FTM_3_CH_6
```

FTM 3 channel 6.

Definition at line 240 of file Ftm_Ocu_Ip_Types.h.

6.1.3.35 FTM_3_CH_7

```
#define FTM_3_CH_7
```

FTM 3 channel 7.

Definition at line 244 of file Ftm_Ocu_Ip_Types.h.

6.1.3.36 FTM_4_CH_0

```
#define FTM_4_CH_0
```

FTM 4 channel 0.

Definition at line 249 of file Ftm_Ocu_Ip_Types.h.

6.1.3.37 FTM_4_CH_1

```
#define FTM_4_CH_1
```

FTM 4 channel 1.

Definition at line 253 of file Ftm_Ocu_Ip_Types.h.

6.1.3.38 FTM_4_CH_2

```
#define FTM_4_CH_2
```

FTM 4 channel 2.

Definition at line 257 of file Ftm_Ocu_Ip_Types.h.

6.1.3.39 FTM_4_CH_3

```
#define FTM_4_CH_3
```

FTM 4 channel 3.

Definition at line 261 of file Ftm_Ocu_Ip_Types.h.

6.1.3.40 FTM_4_CH_4

```
#define FTM_4_CH_4
```

FTM 4 channel 4.

Definition at line 265 of file Ftm_Ocu_Ip_Types.h.

6.1.3.41 FTM_4_CH_5

```
#define FTM_4_CH_5
```

FTM 4 channel 5.

Definition at line 269 of file Ftm_Ocu_Ip_Types.h.

6.1.3.42 FTM_4_CH_6

```
#define FTM_4_CH_6
```

FTM 4 channel 6.

Definition at line 273 of file Ftm_Ocu_Ip_Types.h.

6.1.3.43 FTM_4_CH_7

```
#define FTM_4_CH_7
```

FTM 4 channel 7.

Definition at line 277 of file Ftm_Ocu_Ip_Types.h.

6.1.3.44 FTM_5_CH_0

```
#define FTM_5_CH_0
```

FTM 5 channel 0.

Definition at line 282 of file Ftm_Ocu_Ip_Types.h.

6.1.3.45 FTM_5_CH_1

```
#define FTM_5_CH_1
```

FTM 5 channel 1.

Definition at line 286 of file Ftm_Ocu_Ip_Types.h.

6.1.3.46 FTM_5_CH_2

```
#define FTM_5_CH_2
```

FTM 5 channel 2.

Definition at line 290 of file Ftm_Ocu_Ip_Types.h.

6.1.3.47 FTM_5_CH_3

```
#define FTM_5_CH_3
```

FTM 5 channel 3.

Definition at line 294 of file Ftm_Ocu_Ip_Types.h.

6.1.3.48 FTM_5_CH_4

```
#define FTM_5_CH_4
```

FTM 5 channel 4.

Definition at line 298 of file Ftm_Ocu_Ip_Types.h.

6.1.3.49 FTM_5_CH_5

```
#define FTM_5_CH_5
```

FTM 5 channel 5.

Definition at line 302 of file Ftm_Ocu_Ip_Types.h.

6.1.3.50 FTM_5_CH_6

```
#define FTM_5_CH_6
```

FTM 5 channel 6.

Definition at line 306 of file Ftm_Ocu_Ip_Types.h.

6.1.3.51 FTM_5_CH_7

```
#define FTM_5_CH_7
```

FTM 5 channel 7.

Definition at line 310 of file Ftm_Ocu_Ip_Types.h.

6.1.3.52 FTM_6_CH_0

```
#define FTM_6_CH_0
```

FTM 6 channel 0.

Definition at line 315 of file Ftm_Ocu_Ip_Types.h.

6.1.3.53 FTM_6_CH_1

```
#define FTM_6_CH_1
```

FTM 6 channel 1.

Definition at line 319 of file Ftm_Ocu_Ip_Types.h.

6.1.3.54 FTM_6_CH_2

```
#define FTM_6_CH_2
```

FTM 6 channel 2.

Definition at line 323 of file Ftm_Ocu_Ip_Types.h.

6.1.3.55 FTM_6_CH_3

```
#define FTM_6_CH_3
```

FTM 6 channel 3.

Definition at line 327 of file Ftm_Ocu_Ip_Types.h.

6.1.3.56 FTM_6_CH_4

```
#define FTM_6_CH_4
```

FTM 6 channel 4.

Definition at line 331 of file Ftm_Ocu_Ip_Types.h.

6.1.3.57 FTM_6_CH_5

```
#define FTM_6_CH_5
```

FTM 6 channel 5.

Definition at line 335 of file Ftm_Ocu_Ip_Types.h.

6.1.3.58 FTM_6_CH_6

```
#define FTM_6_CH_6
```

FTM 6 channel 6.

Definition at line 339 of file Ftm_Ocu_Ip_Types.h.

6.1.3.59 FTM_6_CH_7

```
#define FTM_6_CH_7
```

FTM 6 channel 7.

Definition at line 343 of file Ftm_Ocu_Ip_Types.h.

6.1.3.60 FTM_7_CH_0

```
#define FTM_7_CH_0
```

FTM 7 channel 0.

Definition at line 348 of file Ftm_Ocu_Ip_Types.h.

6.1.3.61 FTM_7_CH_1

```
#define FTM_7_CH_1
```

FTM 7 channel 1.

Definition at line 352 of file Ftm_Ocu_Ip_Types.h.

6.1.3.62 FTM_7_CH_2

```
#define FTM_7_CH_2
```

FTM 7 channel 2.

Definition at line 356 of file Ftm_Ocu_Ip_Types.h.

6.1.3.63 FTM_7_CH_3

```
#define FTM_7_CH_3
```

FTM 7 channel 3.

Definition at line 360 of file Ftm_Ocu_Ip_Types.h.

6.1.3.64 FTM_7_CH_4

```
#define FTM_7_CH_4
```

FTM 7 channel 4.

Definition at line 364 of file Ftm_Ocu_Ip_Types.h.

6.1.3.65 FTM_7_CH_5

```
#define FTM_7_CH_5
```

FTM 7 channel 5.

Definition at line 368 of file Ftm_Ocu_Ip_Types.h.

6.1.3.66 FTM_7_CH_6

```
#define FTM_7_CH_6
```

FTM 7 channel 6.

Definition at line 372 of file Ftm_Ocu_Ip_Types.h.

6.1.3.67 FTM_7_CH_7

```
#define FTM_7_CH_7
```

FTM 7 channel 7.

Definition at line 376 of file Ftm_Ocu_Ip_Types.h.

6.1.3.68 OCU_FTM_OUTPUTDISABLED

```
#define OCU_FTM_OUTPUTDISABLED
```

FTM register define used to configure counter clock source.

Definition at line 383 of file Ftm_Ocu_Ip_Types.h.

6.1.3.69 FTM_BDM_MODE_00

```
#define FTM_BDM_MODE_00
```

Options for the FlexTimer behavior in BDM Mode.

FTM counter stopped, CH(n)F bit can be set, FTM channels in functional mode, writes to MOD, CNTIN and C(n)V registers bypass the register buffers

Definition at line 443 of file Ftm_Ocu_Ip_Types.h.

6.1.3.70 FTM_BDM_MODE_01

```
#define FTM_BDM_MODE_01
```

FTM counter stopped, CH(n)F bit is not set, FTM channels outputs are forced to their safe value, writes to MOD, CNTIN and C(n)V registers bypass the register buffers

Definition at line 446 of file Ftm_Ocu_Ip_Types.h.

6.1.3.71 FTM_BDM_MODE_10

```
#define FTM_BDM_MODE_10
```

FTM counter stopped, CH(n)F bit is not set, FTM channels outputs are frozen when chip enters in BDM mode, writes to MOD, CNTIN and C(n)V registers bypass the register buffers

Definition at line 449 of file Ftm_Ocu_Ip_Types.h.

6.1.3.72 FTM_BDM_MODE_11

```
#define FTM_BDM_MODE_11
```

FTM counter in functional mode, CH(n)F bit can be set, FTM channels in functional mode, writes to MOD, CNTIN and C(n)V registers is in fully functional mode

Definition at line 452 of file Ftm_Ocu_Ip_Types.h.

6.1.3.73 FTM_CLOCK_SOURCE_NONE

```
#define FTM_CLOCK_SOURCE_NONE
```

FlexTimer Clock Source Selection.

None use clock for FTM

Definition at line 458 of file Ftm_Ocu_Ip_Types.h.

6.1.3.74 FTM_CLOCK_SOURCE_SYSTEMCLK

```
#define FTM_CLOCK_SOURCE_SYSTEMCLK
```

System clock

Definition at line 459 of file Ftm_Ocu_Ip_Types.h.

6.1.3.75 FTM_CLOCK_SOURCE_FIXEDCLK

```
#define FTM_CLOCK_SOURCE_FIXEDCLK
```

Fixed clock

Definition at line 460 of file Ftm_Ocu_Ip_Types.h.

6.1.3.76 FTM_CLOCK_SOURCE_EXTERNALCLK

```
#define FTM_CLOCK_SOURCE_EXTERNALCLK
```

External clock

Definition at line 461 of file Ftm_Ocu_Ip_Types.h.

6.1.3.77 FTM_CLOCK_DIVID_BY_1

```
#define FTM_CLOCK_DIVID_BY_1
```

FlexTimer Prescale Factor Selections bits (PS).

Selects one of 8 division factors for the clock source selected by CLKS. The new prescaler factor affects the clock source on the next FTM input clock cycle after the new value is updated into the register bits. This field is write protected. It can be written only when MODE[WPDIS] = 1. 000b - Divide by 1 001b - Divide by 2 010b - Divide by 4 011b - Divide by 8 100b - Divide by 16 101b - Divide by 32 110b - Divide by 64 111b - Divide by 128 Divide by 1

Definition at line 479 of file Ftm_Ocu_Ip_Types.h.

6.1.3.78 FTM_CLOCK_DIVID_BY_2

```
#define FTM_CLOCK_DIVID_BY_2
```

Divide by 2

Definition at line 480 of file Ftm_Ocu_Ip_Types.h.

6.1.3.79 FTM_CLOCK_DIVID_BY_4

```
#define FTM_CLOCK_DIVID_BY_4
```

Divide by 4

Definition at line 481 of file Ftm_Ocu_Ip_Types.h.

6.1.3.80 FTM_CLOCK_DIVID_BY_8

```
#define FTM_CLOCK_DIVID_BY_8
```

Divide by 8

Definition at line 482 of file Ftm_Ocu_Ip_Types.h.

6.1.3.81 FTM_CLOCK_DIVID_BY_16

```
#define FTM_CLOCK_DIVID_BY_16
```

Divide by 16

Definition at line 483 of file Ftm_Ocu_Ip_Types.h.

6.1.3.82 FTM_CLOCK_DIVID_BY_32

```
#define FTM_CLOCK_DIVID_BY_32
```

Divide by 32

Definition at line 484 of file Ftm_Ocu_Ip_Types.h.

6.1.3.83 FTM_CLOCK_DIVID_BY_64

```
#define FTM_CLOCK_DIVID_BY_64
```

Divide by 64

Definition at line 485 of file Ftm_Ocu_Ip_Types.h.

6.1.3.84 FTM_CLOCK_DIVID_BY_128

```
#define FTM_CLOCK_DIVID_BY_128
```

Divide by 128

Definition at line 486 of file Ftm_Ocu_Ip_Types.h.

6.1.3.85 FTM_OCU_SET_LOW

```
#define FTM_OCU_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 493 of file Ftm_Ocu_Ip_Types.h.

6.1.3.86 FTM_OCU_SET_HIGH

```
#define FTM_OCU_SET_HIGH
```

The channel pin will be set LOW upon compare match.

Definition at line 495 of file Ftm_Ocu_Ip_Types.h.

6.1.3.87 FTM_OCU_SET_TOGGLE

```
#define FTM_OCU_SET_TOGGLE
```

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

Definition at line 497 of file Ftm_Ocu_Ip_Types.h.

6.1.3.88 FTM_OCU_SET_DISABLE

```
#define FTM_OCU_SET_DISABLE
```

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

Definition at line 499 of file Ftm_Ocu_Ip_Types.h.

6.1.3.89 OCU_FTM_LOW

```
#define OCU_FTM_LOW
```

Pin State levels.

Output state of the pin linked to an OCU channel.

Ocu Pin level is logic low

Definition at line 507 of file Ftm_Ocu_Ip_Types.h.

6.1.3.90 OCU_FTM_HIGH

```
#define OCU_FTM_HIGH
```

Ocu Pin level is logic high.

Definition at line 509 of file Ftm_Ocu_Ip_Types.h.

6.1.3.91 FTM_OCU_PRIMARY_PRESCALER

```
#define FTM_OCU_PRIMARY_PRESCALER
```

Prescaler values.

Possible values of prescallers used to configure base-clock timers

Selected value is the default/primary prescaler

Definition at line 517 of file Ftm_Ocu_Ip_Types.h.

6.1.3.92 FTM_OCU_ALTERNATIVE_PRESCALER

```
#define FTM_OCU_ALTERNATIVE_PRESCALER
```

Selected value is the alternative configured prescaler.

Definition at line 519 of file Ftm_Ocu_Ip_Types.h.

6.1.3.93 OCU_FTM_CM_IN_REF_INTERVAL

```
#define OCU_FTM_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 527 of file Ftm_Ocu_Ip_Types.h.

6.1.3.94 OCU_FTM_CM_OUT_REF_INTERVAL

```
#define OCU_FTM_CM_OUT_REF_INTERVAL
```

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

Definition at line 529 of file Ftm_Ocu_Ip_Types.h.

6.1.4 Types Reference

6.1.4.1 Ftm_Ocu_Ip_PinActionType

```
typedef uint8 Ftm_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 552 of file Ftm_Ocu_Ip_Types.h.

6.1.4.2 Ftm_Ocu_Ip_PinStateType

```
typedef uint8 Ftm_Ocu_Ip_PinStateType
```

Pin State level.

Output state of the pin linked to an OCU channel.

Definition at line 559 of file Ftm_Ocu_Ip_Types.h.

6.1.4.3 Ftm_Ocu_Ip_SelectPrescalerType

```
typedef uint8 Ftm_Ocu_Ip_SelectPrescalerType
```

Prescaler type.

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

Definition at line 566 of file Ftm_Ocu_Ip_Types.h.

6.1.4.4 Ftm_Ocu_Ip_ReturnType

```
typedef uint8 Ftm_Ocu_Ip_ReturnType
```

Ocu Return Type.

Return information after setting a new threshold value.

Definition at line 573 of file Ftm_Ocu_Ip_Types.h.

6.1.4.5 Ftm_Ocu_Ip_xRegLayoutType

```
typedef FTM_Type Ftm_Ocu_Ip_xRegLayoutType
```

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

Definition at line 576 of file Ftm_Ocu_Ip_Types.h.

6.1.4.6 Ftm_Ocu_Ip_CallbackType

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

Channel notification typedef.

Definition at line 579 of file Ftm_Ocu_Ip_Types.h.

6.1.5 Enum Reference

6.1.5.1 Ftm_Ocu_Ip_OutputCompareModeType

```
enum Ftm_Ocu_Ip_OutputCompareModeType
```

FlexTimer output compare edge mode. Toggle, clear or set.

Definition at line 536 of file Ftm_Ocu_Ip_Types.h.

6.1.6 Function Reference

6.1.6.1 Ftm_Ocu_Ip_Init()

```
void Ftm_Ocu_Ip_Init (
             const Ftm_Ocu_Ip_ModuleConfigType *const pFtmIpConfig )
```

This function initializes all internals variables of the driver.

This function will initialize with default values all FTM registers. Will loop through all FTM modules and channels in the configuration structure and will setup required registers for each channel and module

Parameters

in	*pFtmIpConfig*	- Pointer to FTM IP configuration structure

6.1.6.2 Ftm_Ocu_Ip_StartChannel()

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

Ocu driver function for starting the FTM timer channel.

This function: Configure ELSB:ELSA Release software control Clear flag and enable interrupt

Parameters

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

6.1.6.3 Ftm_Ocu_Ip_StopChannel()

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

Ocu driver function for stopping the FTM timer channel.

This function:

- stop the timer channel by writing the ELSB:ELSA = 0:0
- Clear flag and disable interrupt

Parameters

in	*InstNum*	- FTM instance number
in	*ChNum*	- Channel number in the FTM 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_STOP_SEC_CONFIG_DATA_UNSPECIFIED
  Export Post-Build Ocu configurations.
- #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 doesnt 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 uint16 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_StartChannel (Ocu_ChannelType ChannelNumber)

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

  This function stops a specified Ocu channel.

6.2.2 Data Structure Documentation

6.2.2.1 struct Ocu_ChannelConfigType

Ocu high level channel configuration structure.

Definition at line 385 of file Ocu_Types.h.

Data Fields

Type	Name	Description
boolean	OutputPinEnable	Channel output pin enable.

6.2.2.2 struct Ocu_ConfigType

Ocu high level configuration structure.

Definition at line 399 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.
- 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 402 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 404 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 406 of file Ocu_Types.h.

6.2.2.2.1.4 CoreId `uint8 CoreId`

Index Core.

Definition at line 413 of file Ocu_Types.h.

6.2.3 Macro Definition Documentation

6.2.3.1 OCU_STOP_SEC_CONFIG_DATA_UNSPECIFIED

```
#define OCU_STOP_SEC_CONFIG_DATA_UNSPECIFIED
```

Export Post-Build Ocu configurations.

Definition at line 166 of file Ocu.h.

6.2.3.2 OCU_INVALID_CHANNEL_NUM

```
#define OCU_INVALID_CHANNEL_NUM
```

Invalid channel number.

Definition at line 121 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 124 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 132 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 139 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 146 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 153 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 161 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 168 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 175 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 182 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 doesnt have an associated output pin.

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

Definition at line 190 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 198 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 206 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 213 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 219 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 225 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 231 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 237 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 242 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 248 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 254 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 260 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 266 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 272 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 278 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 284 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 290 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 305 of file Ocu_Types.h.

6.2.4.2 Ocu_ValueType

```
typedef uint16 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 312 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 318 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 327 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_LOW	The channel pin will be set LOW upon compare match.
OCU_SET_HIGH	The channel pin will be set HIGH 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 340 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_LOW	Ocu Pin level is logic low.
OCU_HIGH	Ocu Pin level is logic high.

Definition at line 359 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 372 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_StartChannel()

```
void 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

void

Precondition

Ocu_Init

6.2.6.3 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