# User Manual

## for S32K14X OCU Driver

# Contents

**User Manual, Rev. 1.0**

# Chapter 4
# Tresos Configuration Plug-in

**User Manual, Rev. 1.0**

# Chapter 1
# Revision History

### Table 1-1.  Revision History

| Revision | Date | Author | Description |
|---|---|---|---|
| 1.0 | 21/06/2019 | NXP MCAL Team | Updated version for ASR 4.3.1S32K14XR1.0.1 |

**User Manual, Rev. 1.0**

# Chapter 2
# Introduction

This User Manual describes NXP Semiconductors AUTOSAR Output Compare Unit ( Ocu ) for S32K14X .

AUTOSAR Ocu driver configuration parameters and deviations from the specification are described in Ocu 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 intented to be used with the following microcontroller devices of NXP Semiconductors .

**Table 2-1.  S32K14X Derivatives**

| NXP Semiconductors | s32k148_lqfp144, s32k148_lqfp176, s32k148_mapbga100, s32k146_lqfp144, s32k146_lqfp100, s32k146_lqfp64, s32k146_mapbga100, s32k144_lqfp100, s32k144_lqfp64, s32k144_mapbga100, s32k142_lqfp100, s32k142_lqfp64, s32k118_lqfp48, s32k118_lqfp64, s32k142_lqfp48, s32k144_lqfp48, s32k148_lqfp100 |
|---|---|

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

## 2.2  Overview

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

AUTOSAR

- paves the way for innovative electronic systems that further improve performance, safety and environmental friendliness.
- is a strong global partnership that creates one common standard: "Cooperate on standards, compete on implementation".
- is a key enabling technology to manage the growing electrics/electronics complexity. It aims to be prepared for the upcoming technologies and to improve cost-efficiency without making any compromise with respect to quality.
- facilitates the exchange and update of software and hardware over the service life of the vehicle.

## 2.3  About this Manual

This Technical Reference employs the following typographical conventions:

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

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

Notes and warnings are shown as below:

### Note

This is a note.

## 2.4  Acronyms and Definitions

### Table 2-2.  Acronyms and Definitions

| Term | Definition |
|------|------------|
| BSW | Basic Software |
| DEM | Diagnostic Event Manager |
| DET | Development Error Tracer |
| ECU | Electronic Control Unit |
| ISR | interrupt Service Routine |
| OCU | Output Compare Unit |
| OS | Operating System |
| RAM | Random Access Memory |
| ROM | Read-only Memory |

*Table continues on the next page...*

**User Manual, Rev. 1.0**

NXP Semiconductors

Table 2-2. Acronyms and Definitions (continued)

| Term | Definition |
|------|------------|
| N/A | Not Applicable |
| MCU | Microcontroller Unit |
| GUI | Graphical User Interface |
| EcuM | ECU state Manager |
| API | Application Programming Interface |
| PB Variant | Post Build Variant |
| PC Variant | Pre Compile Variant |
| SWS | Software Specification |
| VLE | Variable Length Encoding |
| XML | Extensible Markup Language |
| FTM | FlexTimer Module |

## 2.5 Reference List

Table 2-3. Reference List

| # | Title | Version |
|---|-------|---------|
| 1 | Specification of Ocu Driver | AUTOSAR Release 4.3.1 |
| 2 | S32K14X Reference Manual | Reference Manual, Rev. 9, 9/2018 |
| 3 | S32K142 Mask Set Errata for Mask 0N33V (0N33V) | 30/11/2017 |
| 4 | S32K144 Mask Set Errata for Mask 0N57U (0N57U) | 30/11/2017 |
| 5 | S32K146 Mask Set Errata for Mask 0N73V (0N73V) | 30/11/2017 |
| 6 | S32K148 Mask Set Errata for Mask 0N20V (0N20V) | 25/10/2018 |
| 7 | S32K118 Mask Set Errata for Mask 0N97V (0N97V) | 07/01/2019 |

**User Manual, Rev. 1.0**

# Chapter 3
# Driver

## 3.1  Requirements

Requirements for this driver are detailed in the AUTOSAR 4.3 Rev0001Ocu Driver Software Specification document (See Table Reference List ).

## 3.2  Driver Design Summary

The driver provides functions for initialization and control of the microcontroller internal OCU functionality. The OCU driver allows comparing and acting automatically when the value of a counter matches a defined threshold.The OCU driver provides services and configuration parameters for:

- Starting and stopping a comparison process
- Setting comparison threshold
- Enabling and disabling notification mechanisms
- Getting counter values
- Changing output pin states
- Triggering some hardware resources (ADC, DMA) if available

The tick duration of a channel counter depends on the channel specific settings (part of OCU driver) as well as on the system clock and settings of the clock tree controlled by the MCU module. Available FTM Clock sources can be selected in the Plugin.

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

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

Interrupts can be enabled or disabled for each channel in order to have user notifications. For each channel not configured by the user in Tresos Studio configuration tool, the code for interrupt handling is removed based on a series of ifdefs.

## 3.3   Hardware resources

### Table 3-1.   OCU Hardware channels availability for S32K14X family

| Device | Total FTM channels | Total External interrupt vector |
|---|---|---|
| Ocu_s32k142_lqfp64 | 30 ch, 16-bit | 16 vectors |
| Ocu_s32k142_lqfp100 | 32 ch, 16-bit | 16 vectors |
| Ocu_s32k144_lqfp64 | 30 ch, 16-bit | 16 vectors |
| Ocu_s32k144_lqfp100 | 32 ch, 16-bit | 16 vectors |
| Ocu_s32k144_mapbga100 | 32 ch, 16-bit | 16 vectors |
| Ocu_s32k146_lqfp64 | 38 ch, 16-bit | 21 vectors |
| Ocu_s32k146_lqfp100 | 39 ch, 16-bit | 21 vectors |
| Ocu_s32k146_lqfp144 | 48 ch, 16-bit | 24 vectors |
| Ocu_s32k146_mapbga100 | 39 ch, 16-bit | 21 vectors |
| Ocu_s32k148_lqfp144 | 61 ch, 16-bit | 32 vectors |
| Ocu_s32k148_lqfp176 | 64 ch, 16-bit | 32 vectors |
| Ocu_s32k148_mapbga100 | 39 ch, 16-bit | 32 vectors |
| Ocu_s32k118_lqfp48 | 39 ch, 16-bit | 32 vectors |
| Ocu_s32k118_lqfp64 | 39 ch, 16-bit | 32 vectors |
| Ocu_s32k142_lqfp48 | 32 ch, 16-bit | 16 vectors |
| Ocu_s32k144_lqfp48 | 32 ch, 16-bit | 16 vectors |
| Ocu_s32k148_lqfp100 | 39 ch, 16-bit | 32 vectors |

## 3.4   Deviation from Requirements

The driver deviates from the AUTOSAR Ocu Driver software specification in some places.

There are also some additional requirements (on top of requirements detailed in AUTOSAR Ocu Driver software specification) which need to be satisfied for correct operation.

1. Additional Requirement 1
2. Additional Requirement 2
3. Additional Requirement ...
4. Additional Requirement N

**User Manual, Rev. 1.0**

**Table 3-2.   Deviations Status Column Description**

| 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, or out of scope for the driver.

**Table 3-3.   Driver Deviations Table**

| Requirement | Status | Description | Notes |
|-------------|--------|-------------|-------|
| SWS_Ocu_00005 | N/A | Ocu_Lcfg.c shall include Ocu.h and Ocu_MemMap.h. | Link time config is considered an other form of Pb config. No separate files required. |
| SWS_Ocu_00008 | N/A | Ocu_Irq.c shall include Ocu_MemMap.h and Ocu.h. | Interrupts are defined at IP layer. |
| SWS_Ocu_00020 | N/A | The detection of production errors cannot be switched off. | DEM errors are not used |
| SWS_Ocu_00023 | N/A | 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/A | 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/A | 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/A | 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/A | If source code for caller and callee of Ocu_GetVersionInfo is available; the OCU driver should realize Ocu_GetVersionInfo 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/A | These requirements are not applicable to this specification. | Not a requirement |

# 3.5   Driver limitations

None

User Manual, Rev. 1.0

# 3.6 Driver usage and configuration tips

None.

# 3.7 Runtime Errors

None

# 3.8 Software specification

The following sections contains driver software specifications.

## 3.8.1 Define Reference

Constants supported by the driver are as per AUTOSAR Ocu Driver software specification Version 4.3 Rev0001 .

### 3.8.1.1 Define OCU_VENDOR_ID

Table 3-4.   Define OCU_VENDOR_ID Description

| Name | OCU_VENDOR_ID |
|------|---------------|
| Initializer | 43 |

### 3.8.1.2 Define OCU_AR_RELEASE_MAJOR_VERSION

Table 3-5.   Define OCU_AR_RELEASE_MAJOR_VERSION Description

| Name | OCU_AR_RELEASE_MAJOR_VERSION |
|------|------------------------------|
| Initializer | 4 |

### 3.8.1.3  Define OCU_AR_RELEASE_MINOR_VERSION

**Table 3-6.  Define OCU_AR_RELEASE_MINOR_VERSION Description**

| Name | OCU_AR_RELEASE_MINOR_VERSION |
|---|---|
| Initializer | 3 |

### 3.8.1.4  Define OCU_AR_RELEASE_REVISION_VERSION

**Table 3-7.  Define OCU_AR_RELEASE_REVISION_VERSION Description**

| Name | OCU_AR_RELEASE_REVISION_VERSION |
|---|---|
| Initializer | 1 |

### 3.8.1.5  Define OCU_SW_MAJOR_VERSION

**Table 3-8.  Define OCU_SW_MAJOR_VERSION Description**

| Name | OCU_SW_MAJOR_VERSION |
|---|---|
| Initializer | 1 |

### 3.8.1.6  Define OCU_SW_MINOR_VERSION

**Table 3-9.  Define OCU_SW_MINOR_VERSION Description**

| Name | OCU_SW_MINOR_VERSION |
|---|---|
| Initializer | 0 |

### 3.8.1.7  Define OCU_SW_PATCH_VERSION

**Table 3-10.  Define OCU_SW_PATCH_VERSION Description**

| Name | OCU_SW_PATCH_VERSION |
|---|---|
| Initializer | 1 |

## 3.8.1.8   Define OCU_DEINIT_ID

API service ID of Ocu_DeInit function.

**Details:**

Parameters used when raising an error/exception

**Table 3-11.   Define OCU_DEINIT_ID Description**

| Name | OCU_DEINIT_ID |
|---|---|
| Initializer | (0x01U) |

## 3.8.1.9   Define OCU_DISABLENOTIFICATION_ID

API service ID of Ocu_DisableNotification function.

**Details:**

Parameters used when raising an error/exception

**Table 3-12.   Define OCU_DISABLENOTIFICATION_ID Description**

| Name | OCU_DISABLENOTIFICATION_ID |
|---|---|
| Initializer | (0x0AU) |

## 3.8.1.10   Define OCU_ENABLENOTIFICATION_ID

API service ID of Ocu_EnableNotification function.

**Details:**

Parameters used when raising an error/exception

**User Manual, Rev. 1.0**

#### Table 3-13.   Define OCU_ENABLENOTIFICATION_ID Description

| Name | OCU_ENABLENOTIFICATION_ID |
|---|---|
| Initializer | (0x0BU) |

## 3.8.1.11   Define OCU_GETCOUNTER_ID

API service ID of Ocu_GetCounter function.

### Details:

Parameters used when raising an error/exception

#### Table 3-14.   Define OCU_GETCOUNTER_ID Description

| Name | OCU_GETCOUNTER_ID |
|---|---|
| Initializer | (0x06U) |

## 3.8.1.12   Define OCU_GETVERSIONINFO_ID

API service ID of Ocu_GetVersionInfo function.

### Details:

Parameters used when raising an error/exception

#### Table 3-15.   Define OCU_GETVERSIONINFO_ID Description

| Name | OCU_GETVERSIONINFO_ID |
|---|---|
| Initializer | (0x09U) |

## 3.8.1.13   Define OCU_INIT_ID

API service ID of Ocu_Init function.

### Details:

Parameters used when raising an error/exception

**Table 3-16.   Define OCU_INIT_ID Description**

| Name | OCU_INIT_ID |
|------|-------------|
| Initializer | (0x00U) |

## 3.8.1.14   Define OCU_PROCESSNOTIFICATION_ID

API service ID.

**Table 3-17.   Define OCU_PROCESSNOTIFICATION_ID
Description**

| Name | OCU_PROCESSNOTIFICATION_ID |
|------|----------------------------|
| Initializer | 0x0AU |

## 3.8.1.15   Define OCU_SETABSOLUTETHRESHOLD_ID

API service ID of Ocu_SetAbsoluteThreshold function.

**Details:**

Parameters used when raising an error/exception

**Table 3-18.   Define OCU_SETABSOLUTETHRESHOLD_ID
Description**

| Name | OCU_SETABSOLUTETHRESHOLD_ID |
|------|-----------------------------|
| Initializer | (0x07U) |

## 3.8.1.16   Define OCU_SETCLOCKMODE_ID

API Ocu_SetClockMode service called with wrong parameter.

**Details:**

Parameters used when raising an error/exception

### Table 3-19.   Define OCU_SETCLOCKMODE_ID Description

| Name | OCU_SETCLOCKMODE_ID |
|------|---------------------|
| Initializer | (0x7AU) |

## 3.8.1.17   Define OCU_SETPINACTION_ID

API service ID of Ocu_SetPinAction function.

**<u>Details</u>:**

Parameters used when raising an error/exception

### Table 3-20.   Define OCU_SETPINACTION_ID Description

| Name | OCU_SETPINACTION_ID |
|------|---------------------|
| Initializer | (0x05U) |

## 3.8.1.18   Define OCU_SETPINSTATE_ID

API service ID of Ocu_SetPinState function.

**<u>Details</u>:**

Parameters used when raising an error/exception

### Table 3-21.   Define OCU_SETPINSTATE_ID Description

| Name | OCU_SETPINSTATE_ID |
|------|--------------------|
| Initializer | (0x04U) |

## 3.8.1.19   Define OCU_SETRELATIVETHRESHOLD_ID

API service ID of Ocu_SetRelativeThreshold function.

**<u>Details</u>:**

Parameters used when raising an error/exception

**Table 3-22. Define OCU_SETRELATIVETHRESHOLD_ID
Description**

| Name | OCU_SETRELATIVETHRESHOLD_ID |
|---|---|
| Initializer | (0x08U) |

## 3.8.1.20 Define OCU_STARTCHANNEL_ID

API service ID of Ocu_StartChannel function.

**Details:**

Parameters used when raising an error/exception

**Table 3-23. Define OCU_STARTCHANNEL_ID Description**

| Name | OCU_STARTCHANNEL_ID |
|---|---|
| Initializer | (0x02U) |

## 3.8.1.21 Define OCU_STOPCHANNEL_ID

API service ID of Ocu_StopChannel function.

**Details:**

Parameters used when raising an error/exception

**Table 3-24. Define OCU_STOPCHANNEL_ID Description**

| Name | OCU_STOPCHANNEL_ID |
|---|---|
| Initializer | (0x03U) |

## 3.8.1.22 Define OCU_E_ALREADY_INITIALIZED

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

**Details:**

**User Manual, Rev. 1.0**

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

**Implements:** Ocu_ErrorIds_define AUTOSAR

**Table 3-25.  Define OCU_E_ALREADY_INITIALIZED Description**

| Name | OCU_E_ALREADY_INITIALIZED |
|---|---|
| Initializer | (0x07U) |

## 3.8.1.23   Define OCU_E_BUSY

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

**Details:**

Errors and exceptions that will be detected by the OCU driver

**Implements:** Ocu_ErrorIds_define AUTOSAR

**Table 3-26.   Define OCU_E_BUSY Description**

| Name | OCU_E_BUSY |
|---|---|
| Initializer | (0x09U) |

## 3.8.1.24   Define OCU_E_INIT_FAILED

API Ocu_Init service called with wrong parameter.

**Details:**

Errors and exceptions that will be detected by the OCU driver

**Implements:** Ocu_ErrorIds_define AUTOSAR

**Table 3-27.   Define OCU_E_INIT_FAILED Description**

| Name | OCU_E_INIT_FAILED |
|---|---|
| Initializer | (0x0BU) |

**User Manual, Rev. 1.0**

## 3.8.1.25   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.

**Details:**

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

**Implements:** Ocu_ErrorIds_define AUTOSAR

**Table 3-28.   Define OCU_E_NO_VALID_NOTIF Description**

| Name | OCU_E_NO_VALID_NOTIF |
|---|---|
| Initializer | (0x06U) |

## 3.8.1.26   Define OCU_E_PARAM_INSTANCE

Generated when the module id is more than the number of module that supported by this platform.

**Details:**

Errors and exceptions that will be detected by the OCU driver

**Implements:** Ocu_ErrorIds_define Non-AUTOSAR

**Table 3-29.   Define OCU_E_PARAM_INSTANCE Description**

| Name | OCU_E_PARAM_INSTANCE |
|---|---|
| Initializer | (0x1DU) |

## 3.8.1.27   Define OCU_E_PARAM_INVALID_ACTION

API `Ocu_SetPinAction()` called with an invalid pin action.

**Details:**

Errors and exceptions that will be detected by the OCU driver

**Implements:** Ocu_ErrorIds_define AUTOSAR

**Table 3-30.   Define OCU_E_PARAM_INVALID_ACTION Description**

| Name | OCU_E_PARAM_INVALID_ACTION |
|------|----------------------------|
| Initializer | (0x05U) |

## 3.8.1.28   Define OCU_E_PARAM_INVALID_CHANNEL

API service used with an invalid channel Identifier.

**Details:**

Errors and exceptions that will be detected by the OCU driver

**Implements:** Ocu_ErrorIds_define AUTOSAR

**Table 3-31.   Define OCU_E_PARAM_INVALID_CHANNEL Description**

| Name | OCU_E_PARAM_INVALID_CHANNEL |
|------|-----------------------------|
| Initializer | (0x03U) |

## 3.8.1.29   Define OCU_E_PARAM_INVALID_STATE

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

**Details:**

Errors and exceptions that will be detected by the OCU driver

**Implements:** Ocu_ErrorIds_define AUTOSAR

**Table 3-32.   Define OCU_E_PARAM_INVALID_STATE Description**

| Name | OCU_E_PARAM_INVALID_STATE |
|------|---------------------------|
| Initializer | (0x04U) |

**User Manual, Rev. 1.0**

## 3.8.1.30  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.

**Details:**

Errors and exceptions that will be detected by the OCU driver

**Implements:** Ocu_ErrorIds_define Non-AUTOSAR

**Table 3-33.  Define OCU_E_PARAM_INVALID_VALUE Description**

| Name | OCU_E_PARAM_INVALID_VALUE |
|---|---|
| Initializer | (0x1BU) |

## 3.8.1.31  Define OCU_E_PARAM_NO_PIN

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

**Details:**

Errors and exceptions that will be detected by the OCU driver

**Implements:** Ocu_ErrorIds_define AUTOSAR

**Table 3-34.  Define OCU_E_PARAM_NO_PIN Description**

| Name | OCU_E_PARAM_NO_PIN |
|---|---|
| Initializer | (0x0AU) |

## 3.8.1.32  Define OCU_E_PARAM_POINTER

API `Ocu_GetVersionInfo()` is called with a NULL parameter.

**Details:**

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

**Implements:** Ocu_ErrorIds_define AUTOSAR

**Table 3-35.   Define OCU_E_PARAM_POINTER Description**

| Name | OCU_E_PARAM_POINTER |
|---|---|
| Initializer | (0x08U) |

## 3.8.1.33   Define OCU_E_UNINIT

API service used without module initialization.

**Details:**

Errors and exceptions that will be detected by the OCU driver

**Implements:** Ocu_ErrorIds_define AUTOSAR

**Table 3-36.   Define OCU_E_UNINIT Description**

| Name | OCU_E_UNINIT |
|---|---|
| Initializer | (0x02U) |

## 3.8.1.34   Define OCU_DEINIT_API

Switch to indicate that Ocu_DeInit API is supported.

**Table 3-37.   Define OCU_DEINIT_API Description**

| Name | OCU_DEINIT_API |
|---|---|
| Initializer | (STD_ON) |

## 3.8.1.35   Define OCU_GET_COUNTER_API

Switch to indicate that Ocu_GetCounter API is supported.

**Table 3-38.   Define OCU_GET_COUNTER_API Description**

| Name | OCU_GET_COUNTER_API |
|---|---|
| Initializer | (STD_ON) |

**User Manual, Rev. 1.0**

### 3.8.1.36   Define OCU_SET_ABSOLUTE_THRESHOLD_API

Switch to indicate that Ocu_SetAbsoluteThreshold API is supported.

**Table 3-39.   Define OCU_SET_ABSOLUTE_THRESHOLD_API Description**

| Name | OCU_SET_ABSOLUTE_THRESHOLD_API |
|------|--------------------------------|
| Initializer | (STD_ON) |

### 3.8.1.37   Define OCU_SET_CLOCK_MODE_API

Switch for enabling the dual clock functionality (Ocu_SetClockMode API).

**Table 3-40.   Define OCU_SET_CLOCK_MODE_API Description**

| Name | OCU_SET_CLOCK_MODE_API |
|------|------------------------|
| Initializer | (STD_OFF) |

### 3.8.1.38   Define OCU_SET_PIN_ACTION_API

Switch to indicate that Ocu_SetPinAction API is supported.

**Table 3-41.   Define OCU_SET_PIN_ACTION_API Description**

| Name | OCU_SET_PIN_ACTION_API |
|------|------------------------|
| Initializer | (STD_ON) |

### 3.8.1.39   Define OCU_SET_PIN_STATE_API

Switch to indicate that Ocu_SetPinState API is supported.

**Table 3-42.   Define OCU_SET_PIN_STATE_API Description**

| Name | OCU_SET_PIN_STATE_API |
|------|------------------------|
| Initializer | (STD_ON) |

### 3.8.1.40  Define OCU_SET_RELATIVE_THRESHOLD_API

Switch to indicate that Ocu_SetRelativeThreshold API is supported.

**Table 3-43.   Define OCU_SET_RELATIVE_THRESHOLD_API Description**

| Name | OCU_SET_RELATIVE_THRESHOLD_API |
|---|---|
| Initializer | (STD_ON) |

### 3.8.1.41  Define OCU_VERSION_INFO_API

Switch to indicate that Ocu_GetVersionInfo API is supported.

**Table 3-44.   Define OCU_VERSION_INFO_API Description**

| Name | OCU_VERSION_INFO_API |
|---|---|
| Initializer | (STD_ON) |

### 3.8.1.42  Define OCU_NOTIFICATION_SUPPORTED

Switch to indicate that the notifications are supported.

**Table 3-45.   Define OCU_NOTIFICATION_SUPPORTED Description**

| Name | OCU_NOTIFICATION_SUPPORTED |
|---|---|
| Initializer | (STD_ON) |

### 3.8.1.43  Define OCU_DEV_ERROR_DETECT

Switch for enabling the development error detection.

**Table 3-46.   Define OCU_DEV_ERROR_DETECT Description**

| Name | OCU_DEV_ERROR_DETECT |
|---|---|
| Initializer | (STD_ON) |

## 3.8.2   Enum Reference

Enumeration of all constants supported by the driver are as per AUTOSAR Ocu Driver software specification Version 4.3 Rev0001 .

### 3.8.2.1   Enumeration Ocu_ChannelStatusType

This enumerated type allows the selection of channel status type.

**Table 3-47.   Enumeration Ocu_ChannelStatusType Values**

| Name | Initializer | Description |
|---|---|---|
| OCU_STATUS_UNINITIALIZED | 0 | OCU channel status - uninitialized. |
| OCU_STATUS_INITIALIZED | 1 | OCU channel status - initialized. |
| OCU_STATUS_STOPPED | 2 | OCU channel status - stopped. |
| OCU_STATUS_RUNNING | 3 | OCU channel status - expired. |

### 3.8.2.2   Enumeration Ocu_CountDirectionType

Ocu Count direction.

**Details:**

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

**Implements:** Ocu_CountDirectionType_enumeration

**Table 3-48.   Enumeration Ocu_CountDirectionType Values**

| Name | Initializer | Description |
|---|---|---|
| OCU_UPCOUNTING | 0 | Counter in UP Counting. |
| OCU_DOWNCOUNTING | 1 | Counter in DOWN Counting. |

### 3.8.2.3   Enumeration Ocu_GlobalStateType

Enum containing the possible states of the Ocu driver.

**User Manual, Rev. 1.0**

**Table 3-49.  Enumeration Ocu_GlobalStateType Values**

| Name | Initializer | Description |
|------|------------|-------------|
| OCU_STATE_UNINIT | 0x00 | OCU STATE UNINIT |
| OCU_STATE_IDLE | 0x01 | OCU STATE IDLE |

## 3.8.2.4   Enumeration Ocu_PinActionType

Edge Pin Action type.

**Details:**

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

**Implements:** Ocu_PinActionType_enumeration

**Table 3-50.  Enumeration Ocu_PinActionType Values**

| Name | Initializer | Description |
|------|------------|-------------|
| OCU_SET_LOW | 0 | The channel pin will be set HIGH upon compare match. |
| OCU_SET_HIGH | 1 | The channel pin will be set LOW upon compare match. |
| OCU_TOGGLE | 2 | The channel pin will be set to the opposite of its current level HIGH upon compare match. |
| OCU_DISABLE | 3 | The channel pin will remain at its current level upon compare match. |

## 3.8.2.5   Enumeration Ocu_PinStateType

Pin State level.

**Details:**

Output state of the pin linked to an OCU channel.

**Implements:** Ocu_PinStateType_enumeration

**Table 3-51.   Enumeration Ocu_PinStateType Values**

| Name | Initializer | Description |
| --- | --- | --- |
| OCU_LOW | 0 | Ocu Pin level is logic low. |
| OCU_HIGH | 1 | Ocu Pin level is logic high. |

## 3.8.2.6   Enumeration Ocu_ReturnType

Ocu Return Type.

### Details:

Return information after setting a new threshold value.

**Implements:** Ocu_PinStateType_enumeration

**Table 3-52.   Enumeration Ocu_ReturnType Values**

| Name | Initializer | Description |
| --- | --- | --- |
| OCU_CM_IN_REF_INTERVAL | 0 | The compare match will occur inside the current Reference Interval. |
| OCU_CM_OUT_REF_INTERVAL | 1 | The compare match will not occur inside the current Reference Interval. |

## 3.8.2.7   Enumeration Ocu_SelectPrescalerType

Prescaler type.

### Details:

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

**Table 3-53.   Enumeration Ocu_SelectPrescalerType Values**

| Name | Initializer | Description |
| --- | --- | --- |
| OCU_PRIMARY_PRESCALER | 0 | Selected value is the default/primary prescaler. |
| OCU_ALTERNATIVE_PRESCALER | 1 | Selected value is the alternative configured prescaler. |

### 3.8.3  Function Reference

Functions of all functions supported by the driver are as per AUTOSAR Ocu Driver software specification Version 4.3 Rev0001 .

#### 3.8.3.1  Function Ocu_Init

This function initializes the Ocu driver.

**Details:**

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

**Return:** void.

**Implements:** Ocu_Init_Activity

**Violates:** Violates MISRA 2004 Required Rule 8.10 could be made static

**Prototype:** `void Ocu_Init(const Ocu_ConfigType *ConfigPtr);`

**Table 3-54.   Ocu_Init Arguments**

| Type | Name | Direction | Description |
|------|------|-----------|-------------|
| const`Ocu_ConfigType`* | ConfigPtr | **input** | Pointer to OCU top configuration structure. |

#### 3.8.3.2  Function Ocu_DeInit

This function deinitializes the Ocu driver.

**Details:**

**User Manual, Rev. 1.0**

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.

**Return:** void.

**Implements:** Ocu_DeInit_Activity

**Prototype:** `void Ocu_DeInit(void);`

### 3.8.3.3 Function Ocu_GetVersionInfo

This function returns Ocu driver version details.

**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.

**Return:** void.

**Implements:** Ocu_GetVersionInfo_Activity

**Prototype:** `void Ocu_GetVersionInfo(Std_VersionInfoType *versioninfo);`

| Type | Name | Direction | Description |
|------|------|-----------|-------------|
| Std_VersionInfoType * | versioninfo | **input, output** | - pointer to Std_VersionInfoType output variable. |

## 3.8.3.4  Function Ocu_DisableNotification

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

**Details:**

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.

**Return:** void.

**Implements:** Ocu_DisableNotification_Activity

**Prototype:** `void Ocu_DisableNotification(Ocu_ChannelType ChannelNumber);`

**Table 3-56.  Ocu_DisableNotification Arguments**

| Type | Name | Direction | Description |
|------|------|-----------|-------------|
| `Ocu_ChannelType` | ChannelNumber | **input** | - Ocu channel id. |

## 3.8.3.5  Function Ocu_EnableNotification

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

**Details:**

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_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_EnableNotification shall raise the error OCU_E_NO_VALID_NOTIF and return without any action.

**Return:** void.

**Implements:** Ocu_EnableNotification_Activity

**Prototype:** `void Ocu_EnableNotification(Ocu_ChannelType ChannelNumber);`

**Table 3-57.   Ocu_EnableNotification Arguments**

| Type | Name | Direction | Description |
|------|------|-----------|-------------|
| `Ocu_ChannelType` | ChannelNumber | **input** | - Ocu channel id. |
|  | Notification | **input** | - notification type to be enabled. |

## 3.8.3.6   Function Ocu_StartChannel

This function starts a specified Ocu channel.

**Details:**

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 "RUNINNG", 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.

**Return:** void.

**Implements:** Ocu_StartChannel_Activity

**Prototype:** `void Ocu_StartChannel(Ocu_ChannelType ChannelNumber);`

**Table 3-58.   Ocu_StartChannel Arguments**

| Type | Name | Direction | Description |
|------|------|-----------|-------------|
| Ocu_ChannelType | ChannelNumber | **input** | Ocu channel id. |

## 3.8.3.7   Function Ocu_StopChannel

This function stops a specified Ocu channel.

**Details:**

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.

**Return:** void.

**Implements:** Ocu_StopChannel_Activity

**Prototype:** `void Ocu_StopChannel(Ocu_ChannelType ChannelNumber);`

**Table 3-59.   Ocu_StopChannel Arguments**

| Type | Name | Direction | Description |
|------|------|-----------|-------------|
| Ocu_ChannelType | ChannelNumber | **input** | - Ocu channel id. |

## 3.8.3.8   Function Ocu_SetPinState

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

**Details:**

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_INVALID_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_SetPinState 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.

**Return:** void.

**Implements:** Ocu_SetPinState_Activity

**Prototype:** `void Ocu_SetPinState(Ocu_ChannelType ChannelNumber, Ocu_PinStateType PinState);`

**Table 3-60.  Ocu_SetPinState Arguments**

| Type | Name | Direction | Description |
|------|------|-----------|-------------|
| `Ocu_ChannelType` | ChannelNumber | **input** | - Ocu channel id. |
| `Ocu_PinStateType` | PinState | **input** | - Output Pin State. |

## 3.8.3.9  Function Ocu_SetPinAction

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

**Details:**

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.

**User Manual, Rev. 1.0**

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_SetPinAction 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.

**<u>Return:</u>** void.

**<u>Implements:</u>** Ocu_SetPinAction_Activity

**<u>Prototype:</u>** `void Ocu_SetPinAction(Ocu_ChannelType ChannelNumber, Ocu_PinActionType PinAction);`

**Table 3-61.   Ocu_SetPinAction Arguments**

| Type | Name | Direction | Description |
|------|------|-----------|-------------|
| `Ocu_ChannelType` | ChannelNumber | **input** | - Ocu channel id. |
| `Ocu_PinActionType` | PinAction | **input** | - Pin Action (OCU_SET_LOW, OCU_SET_HIGH, OCU_TOGGLE, OCU_DISABLE). |

## 3.8.3.10  Function Ocu_GetCounter

Service to read the current value of the counter.

**<u>Details:</u>**

The function Ocu_GetCounter shall read and return the value of the counter of the channel indicated by ChannelNumber.

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".

**Return:** Ocu_ValueType the value of the hardware counter.

**Implements:** Ocu_GetCounter_Activity

**Prototype:** `Ocu_ValueType Ocu_GetCounter(Ocu_ChannelType ChannelNumber);`

**Table 3-62.   Ocu_GetCounter Arguments**

| Type | Name | Direction | Description |
|------|------|-----------|-------------|
| Ocu_ChannelType | ChannelNumber | **input** | - Ocu channel id. |

## 3.8.3.11   Function Ocu_SetAbsoluteThreshold

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

**Details:**

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_SetAbsoluteThreshold 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)

**Return:** 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.

**Implements:** Ocu_SetAbsoluteThreshold_Activity

**Prototype:** `Ocu_ReturnType Ocu_SetAbsoluteThreshold(Ocu_ChannelType ChannelNumber,`
`Ocu_ValueType ReferenceValue, Ocu_ValueType AbsoluteValue);`

**User Manual, Rev. 1.0**

**Table 3-63.   Ocu_SetAbsoluteThreshold Arguments**

| Type | Name | Direction | Description |
|------|------|-----------|-------------|
| Ocu_ChannelType | ChannelNumber | **input** | - Ocu channel id. |
| Ocu_ValueType | ReferenceValue | **input** | - Value given by the upper layer and used as a base to determine whether to call the notification before the function exits or not. |
| Ocu_ValueType | AbsoluteValue | **input** | - Value to compare with the content of the counter. This value is in ticks. |

## 3.8.3.12   Function Ocu_SetRelativeThreshold

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

**Details:**

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_SetRelativeThreshold 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.

**Return:** 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.

**Implements:** Ocu_SetRelativeThreshold_Activity

**Prototype:** `Ocu_ReturnType Ocu_SetRelativeThreshold(Ocu_ChannelType ChannelNumber, Ocu_ValueType RelativeValue);`

**Table 3-64.  Ocu_SetRelativeThreshold Arguments**

| Type | Name | Direction | Description |
|------|------|-----------|-------------|
| `Ocu_ChannelType` | ChannelNumber | **input** | - Ocu channel id. |
| `Ocu_ValueType` | RelativeValue | **input** | - Value to use for computing the new threshold. |

## 3.8.3.13   Function Ocu_SetClockMode

This function sets the clock mode for the driver

**Details:**

This function is reentrant and it changes the prescaler for the driver, changing the power consumption of the driver

**Return:** void.

**Pre:** Ocu_Init must be called before.

**Implements:** Ocu_SetClockMode_Activity

**Violates:** Violates MISRA 2004 Required Rule 8.10 could be made static

**Prototype:** `void Ocu_SetClockMode(Ocu_SelectPrescalerType Prescaler);`

**Table 3-65.  Ocu_SetClockMode Arguments**

| Type | Name | Direction | Description |
|------|------|-----------|-------------|
| `Ocu_SelectPrescalerType` | Prescaler | **input** | Prescaler type: Normal or Alternate |

## 3.8.4   Structs Reference

Data structures supported by the driver are as per AUTOSAR Ocu Driver software specification Version 4.3 Rev0001 .

## 3.8.4.1   Structure Ocu_ConfigType

Structure that contains OCU channel configuration.

**Details:**

It contains the information like Ocu Channel Numbers, Ocu Channel Configuration Module hardware configuration.

**Implements:** Ocu_ConfigType_struct

**Declaration:**

```
typedef struct
                     {
                       Ocu_ChannelType nNumChannels,
                       Ocu_ChannelConfigType pOcuChannelsConfig,
                       Ocu_IpConfigType IpConfig,
                       Ocu_ChannelType HwToLogicChannelMap[]
                     } Ocu_ConfigType;
```

**Table 3-66.   Structure Ocu_ConfigType member description**

| Member | Description |
|---|---|
| nNumChannels | Number of channels for a given configuration |
| pOcuChannelsConfig | Pointer to the array of channels configured |
| IpConfig | structure of the IP Configuration data |
| HwToLogicChannelMap[] | Array for the hardware channel to logical channel mapping. It's used durring the notification callback from the ISR |

## 3.8.4.2   Structure Ocu_ChannelConfigType

Structure that contains OCU channel configuration.

**Details:**

It contains the information like Ocu Output Pin Enabled

**Implements:** Ocu_ChannelConfigType_struct

**Declaration:**

```
typedef struct
                     {
                       boolean bOcuOutputPinEnable,
                       Ocu_NotifyType pfOcuChannelNotification
                     } Ocu_ChannelConfigType;
```

**Table 3-67.   Structure Ocu_ChannelConfigType member description**

| Member | Description |
|---|---|
| bOcuOutputPinEnable | The measurement mode for a channel (Timestamp, Signal Measurement, Signal Edge Detect, Edge Counter) |
| pfOcuChannelNotification | The configured notification for OcuChannel |

**User Manual, Rev. 1.0**

### 3.8.4.3   Structure Ocu_IpConfigType

Structure that contains Ocu parameters. It represents a link from the logical configuration to the hardware config.

**Implements:** Ocu_IpConfigType_struct

**Declaration:**

```
typedef struct
                    {
                      Ocu_Ftm_IpConfigType pFtmIpConfig,
                          Ocu_IpChannelConfigType (*pIpChannelsConfig)[]
                    } Ocu_IpConfigType;
```

**Table 3-68.   Structure Ocu_IpConfigType member description**

| Member | Description |
|---|---|
| pFtmIpConfig | Pointer to the FTM IP configuration |
| (*pIpChannelsConfig)[]; | Array of the configured hardware channel |

### 3.8.4.4   Structure Ocu_IpChannelConfigType

Structure that combine configuration of each FTM channels in the Ocu.

**Implements:** Ocu_IpChannelConfigType_struct

**Declaration:**

```
typedef struct
                    {
                      uint8 u8IdxChannelConfig
                    } Ocu_IpChannelConfigType;
```

**Table 3-69.   Structure Ocu_IpChannelConfigType member description**

| Member | Description |
|---|---|
| u8IdxChannelConfig | Assigned FTM channel id. |

**User Manual, Rev. 1.0**

# 3.8.4.5  Structure Ocu_Ftm_IpConfigType

Structure that contains the Ocu general IP configuration.

**Implements:** Ocu_Ftm_IpConfigType

**Declaration:**

```
typedef struct
                    {
                      Ocu_Ftm_ChannelType u8NumChannels,
                      Ocu_Ftm_ModuleType u8NumModules,
                      Ocu_Ftm_ChannelConfigType (*pChannelsConfig)[],
                      Ocu_Ftm_ModuleConfigType (*pModulesConfig)[];
                    } Ocu_Ftm_IpConfigType;
```

**Table 3-70.  Structure Ocu_Ftm_IpConfigType member description**

| Member | Description |
|---|---|
| u8NumChannels | Number of the hardware configured channels |
| u8NumModules | Number of the hardware configured modules |
| (*pChannelsConfig)[]; | List of the hardware channel configurations |
| (*pModulesConfig)[]; | List of the hardware module configurations |

# 3.8.4.6  Structure Ocu_Ftm_ChannelConfigType

Structure that contains the Ocu general IP configuration.

**Implements:** Ocu_Ftm_ChannelConfigType

**Declaration:**

```
typedef struct
                    {
                      Ocu_Ftm_ChannelType u8EncodedHwChannel,
                      Ocu_ValueType u16DefaultThreshold;
                      Ocu_Ftm_ChannelControlType u8ChannelControlValue;
                    } Ocu_Ftm_ChannelConfigType;
```

**Table 3-71.  Structure Ocu_Ftm_ChannelConfigType member description**

| Member | Description |
|---|---|
| u8EncodedHwChannel | NAssigned Ftm channel id |
| u16DefaultThreshold | Compare match threshold for the current channe |
| u8ChannelControlValue | Ftm channel parameters : |
| | Bits 7 --> 1: Output Pin is Used / 0: Output Pin is not used |

**Table 3-71. Structure Ocu_Ftm_ChannelConfigType member description**

| Member | Description |
|---|---|
| | 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 |

## 3.8.4.7 Structure Ocu_Ftm_ModuleConfigType

Structure that contains the Ocu general IP configuration.

**Implements:** Ocu_Ftm_ModuleConfigType

**Declaration:**

```
typedef struct
                        {
                          uint8 u8ModuleId,
                          uint16 u16MaxCounterValue;
                          Ocu_Ftm_ModuleControlType u8ModuleControlValue;
                          #if (OCU_SET_CLOCK_MODE_API == STD_ON)
                          uint8 u8AltControlValue;
                          #endif
                        } Ocu_Ftm_ModuleConfigType;
```

**Table 3-72. Structure Ocu_Ftm_ModuleConfigType member description**

| Member | Description |
|---|---|
| u8ModuleId | Assigned Ftm module id |
| u16MaxCounterValue | Maximum counter value |
| u8ModuleControlValue | Ftm module parameters :<br><br>Bits 7 .. 6 --> Clock source<br><br>Bits 5 .. 3 --> Normal prescale<br><br>Bit 2 .. 1 --> Debug mode config<br><br>Bits 0 --> Reserved for future use |
| u8AltControlValue | Ftm channel parameters :<br><br>Bits 7 .. 5 --> Alternate Prescaler configuration<br><br>Bits 4 .. 0 --> Reserved for future use |

## 3.8.5   Types Reference

Types supported by the driver are as per AUTOSAR Ocu Driver software specification Version 4.3 Rev0001 .

### 3.8.5.1   Typedef Ocu_ChannelIpType

IP type used to implement a Ocu channel.

**Type:** uint8

### 3.8.5.2   Typedef Ocu_ChannelType

Ocu channel type.

**Implements:** Ocu_ChannelType_typedef

**Type:** uint16

### 3.8.5.3   Typedef Ocu_Ftm_ChannelType

Ftm HW module/channel id type

**Type:** uint8

### 3.8.5.4   Typedef Ocu_Ftm_ModuleType

Ftm HW module/channel id type

**Type:** uint8

### 3.8.5.5   Typedef Ocu_Ftm_ChannelControlType

Ftm unified channel control register value

**Type:** uint8

### 3.8.5.6   Typedef Ocu_Ftm_ModuleControlType

Ftm unified module control register value

**Type:** uint8

### 3.8.5.7   Typedef Ocu_NotifyType

Channel notification typedef.

**Type:** void(*)

### 3.8.5.8   Typedef Ocu_ValueType

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

**Implements:** Ocu_ValueType_typedef

**Type:** uint16

## 3.8.6   Variables Reference

Variables supported by the driver are as per AUTOSAR Ocu Driver software specification Version 4.3 Rev0001 .

# 3.9   Symbolic Names Disclaimer

All containers having the symbolic name tag set as true in the Autosar schema will generate defines like:

#define <Container_Short_Name> <Container_ID>

For this reason it is forbidden to duplicate the name of such containers across the MCAL configuration, 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 Ocu Driver. The most of the parameters are described below.

## 4.1   Configuration elements of Ocu

**Included forms :**

- IMPLEMENTATION_CONFIG_VARIANT
- OcuConfigurationOfOptApiServices
- OcuGeneral
- CommonPublishedInformation
- OcuGeneral

## 4.2   Form IMPLEMENTATION_CONFIG_VARIANT



**Figure 4-1. Tresos Plugin snapshot for IMPLEMENTATION_CONFIG_VARIANT form.**

**Table 4-1.   Attribute IMPLEMENTATION_CONFIG_VARIANT detailed description**

| Property | Value |
|---|---|
| Label | Config Variant |
| Default | VariantPostBuild |
| Range | VariantPostBuild VariantPreCompile |

# 4.3   Form OcuConfigurationOfOptionalApis
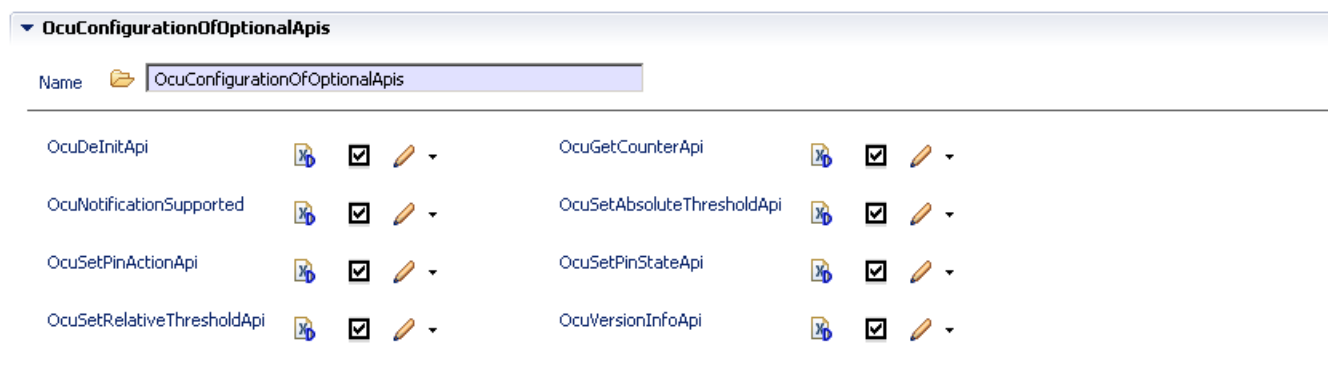


**Figure 4-2. Tresos Plugin snapshot for OcuConfigurationOfOptionalApis form.**

## 4.3.1   OcuDeInitApi (OcuConfigurationOfOptionalApis)

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

**Table 4-2.   Attribute OcuDeInitApi (OcuConfigurationOfOptionalApis) detailed description**

| Property | Value |
|---|---|
| Type | BOOLEAN |
| Origin | AUTOSAR_ECUC |
| Symbolic Name | false |

## 4.3.2   OcuGetCounterApi (OcuConfigurationOfOptionalApis)

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

**Table 4-3.   Attribute OcuGetCounterApi (OcuConfigurationOfOptionalApis) detailed description**

| Property | Value |
|---|---|
| Type | BOOLEAN |
| Origin | AUTOSAR_ECUC |
| Symbolic Name | false |

### 4.3.3 OcuNotificationSupported (OcuConfigurationOfOptionalApis)

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

**Table 4-4.  Attribute OcuNotificationSupported (OcuConfigurationOfOptionalApis) detailed description**

| Property | Value |
|---|---|
| Type | BOOLEAN |
| Origin | AUTOSAR_ECUC |
| Symbolic Name | false |

### 4.3.4 OcuSetAbsoluteThresholdApi (OcuConfigurationOfOptionalApis)

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

**Table 4-5.  Attribute OcuSetAbsoluteThresholdApi (OcuConfigurationOfOptionalApis) detailed description**

| Property | Value |
|---|---|
| Type | BOOLEAN |
| Origin | AUTOSAR_ECUC |
| Symbolic Name | false |

### 4.3.5 OcuSetPinActionApi (OcuConfigurationOfOptionalApis)

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

**Table 4-6.  Attribute OcuSetPinActionApi (OcuConfigurationOfOptionalApis) detailed description**

| Property | Value |
|---|---|
| Type | BOOLEAN |
| Origin | AUTOSAR_ECUC |
| Symbolic Name | false |

## 4.3.6  OcuSetPinStateApi (OcuConfigurationOfOptionalApis)

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

**Table 4-7.  Attribute OcuSetPinStateApi (OcuConfigurationOfOptionalApis) detailed description**

| Property | Value |
|---|---|
| Type | BOOLEAN |
| Origin | AUTOSAR_ECUC |
| Symbolic Name | false |

## 4.3.7  OcuSetRelativeThresholdApi (OcuConfigurationOfOptionalApis)

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

**Table 4-8.  Attribute OcuSetRelativeThresholdApi (OcuConfigurationOfOptionalApis) detailed description**

| Property | Value |
|---|---|
| Type | BOOLEAN |
| Origin | AUTOSAR_ECUC |
| Symbolic Name | false |

## 4.3.8  OcuVersionInfoApi (OcuConfigurationOfOptionalApis)

Switch to indicate that the Ocu_GetVersionInfo() is supported.

**Table 4-9.  Attribute OcuVersionInfoApi (OcuConfigurationOfOptionalApis) detailed description**

| Property | Value |
|---|---|
| Type | BOOLEAN |
| Origin | AUTOSAR_ECUC |
| Symbolic Name | false |

# 4.4 Form OcuGeneral

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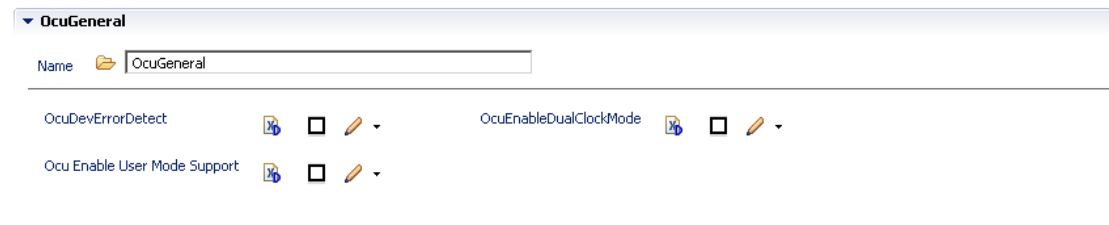


**Figure 4-3. Tresos Plugin snapshot for OcuGeneral form.**

## 4.4.1 OcuDevErrorDetect (OcuGeneral)

Enables/Disables development error detection.

**Table 4-10. Attribute OcuDevErrorDetect (OcuGeneral) detailed description**

| Property | Value |
|---|---|
| Type | BOOLEAN |
| Origin | AUTOSAR_ECUC |
| Symbolic Name | false |
| Default | true |

## 4.4.2 OcuEnableDualClockMode (OcuGeneral)

Enables/Disables API Ocu_SetClockMode().

**Table 4-11. Attribute OcuEnableDualClockMode (OcuGeneral) detailed description**

| Property | Value |
|---|---|
| Type | BOOLEAN |
| Origin | Custom |
| Default | false |

## 4.4.3 OcuEnableUserModeSupport (OcuGeneral)

When this parameter is enabled, the Ocu module will adapt to run from User Mode. Ocu driver does not perform any further actions when running in User Mode. Note: Implementation Specific Parameter.

**User Manual, Rev. 1.0**

**Table 4-12. Attribute OcuEnableDualClockMode (OcuGeneral) detailed description**

| Property | Value |
|---|---|
| Type | BOOLEAN |
| Origin | Custom |
| Default | false |

## 4.4.4   Form OcuHwResourceConfig

This container contains list of available interrupt sources supported by hardware. This tab also enables or disables interrupts for channels which are used in application.
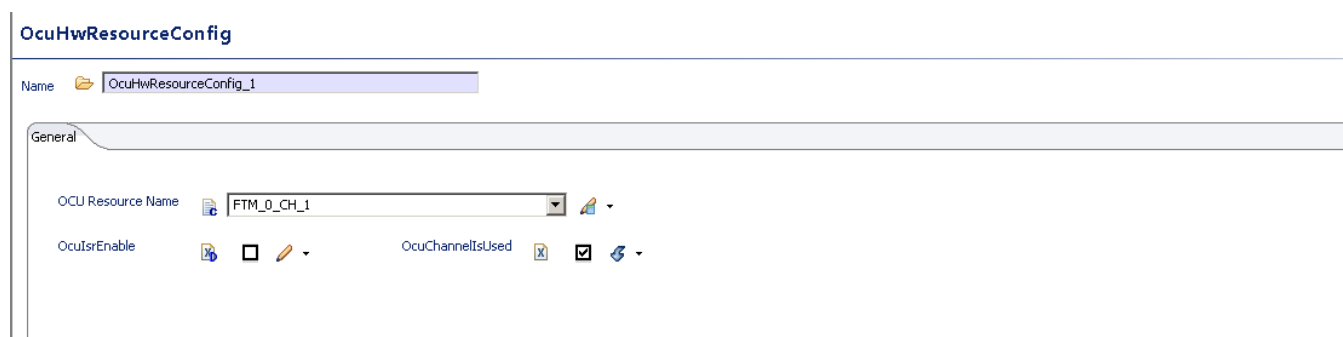


**Figure 4-4. Tresos Plugin snapshot for OcuHwResourceConfig form.**

## 4.4.4.1   OcuIsrHwId (OcuHwResourceConfig)

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

**Table 4-13. Attribute OcuIsrHwId (OcuGeneral) detailed description**

| Property | Value |
|---|---|
| Type | STRING |
| Origin | Custom |
| Symbolic Name | false |
| Range | FTM_0_CH_0,FTM_0_CH_1,FTM_0_CH_2,FTM_0_CH_3,FTM_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_CH_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,FTM_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_CH_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,FTM_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,F |

**Table 4-13.   Attribute OcuIsrHwId (OcuGeneral) detailed description**

| Property | Value |
|---|---|
|  | TM_5_CH_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,FTM_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_CH_7 |

## 4.4.4.2   OcuIsrEnable (OcuHwResourceConfig)

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

**Table 4-14.   Attribute OcuIsrEnable (OcuGeneral) detailed description**

| Property | Value |
|---|---|
| Type | BOOLEAN |
| Origin | Custom |
| Symbolic Name | false |
| Default | false |

## 4.4.4.3   OcuChannelIsUsed (OcuHwResourceConfig)

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

**Table 4-15.   Attribute OcuChannelIsUsed (OcuGeneral) detailed description**

| Property | Value |
|---|---|
| Type | BOOLEAN |
| Origin | Custom |
| Symbolic Name | false |
| Default | false |

# 4.5   Form 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

**Figure 4-5. Tresos Plugin snapshot for OcuConfigSet form.**

# 4.5.1   OcuCountdirection (OcuConfigSet)

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

**Table 4-16.   Attribute OcuCountdirection (OcuConfigSet) detailed description**

| Property | Value |
|----------|-------|
| Type | ENUMERATION |
| Origin | AUTOSAR_ECUC |
| Optional | true |
| Default | OCU_UPCOUNTING |
| Range | OCU_UPCOUNTING<br>OCU_DOWNCOUNTING |

# 4.5.2   Form OcuChannel

This container contains the channel-wide configuration (parameters) of the OCU Driver

**Is included by form :** Form OcuConfigSet
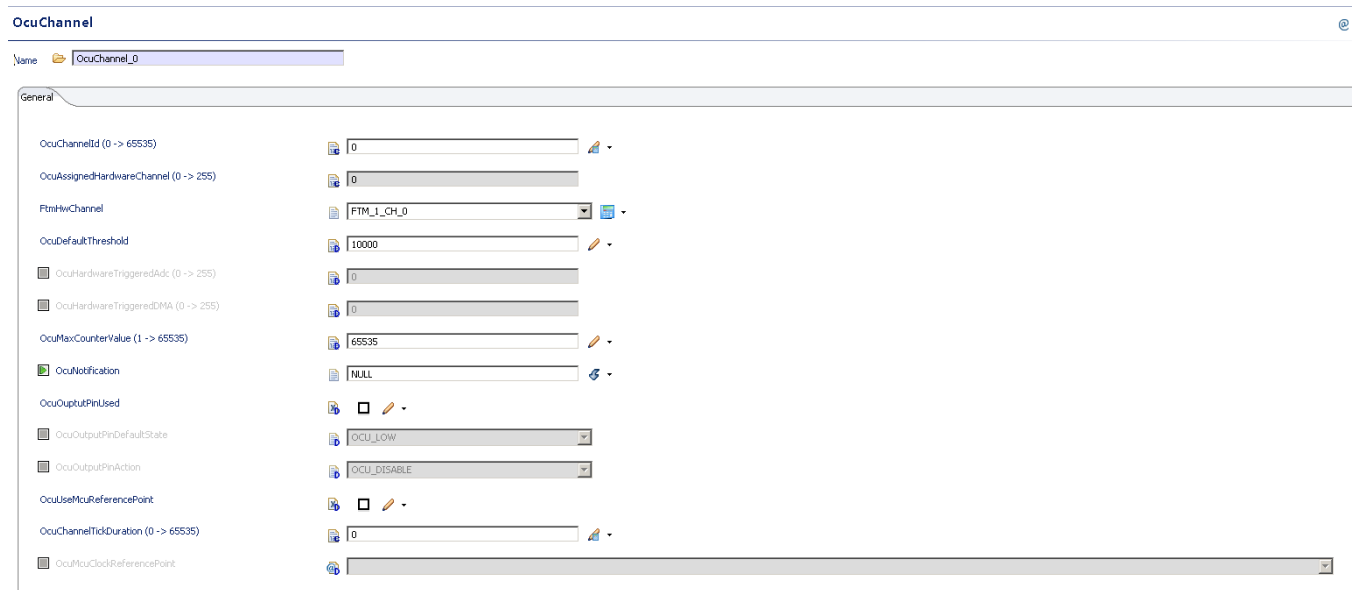
**Figure 4-6. Tresos Plugin snapshot for OcuChannel form.**

## 4.5.2.1   OcuChannelId (OcuChannel)

Channel Id of the OCU channel. This value will be assigned to the symbolic name derived of the OcuChannel container short name.

**Table 4-17.   Attribute OcuChannelId (OcuChannel) detailed description**

| Property | Value |
|---|---|
| Type | INTEGER |
| Origin | AUTOSAR_ECUC |
| Symbolic Name | true |
| Invalid | Range<br>    <=65535<br>    >=0 |

## 4.5.2.2   OcuAssignedHardwareChannel (OcuChannel)

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

**Table 4-18.   Attribute OcuAssignedHardwareChannel (OcuChannel) detailed description**

| Property | Value |
|---|---|
| Type | INTEGER |

*Table continues on the next page...*

**User Manual, Rev. 1.0**

**Table 4-18. Attribute OcuAssignedHardwareChannel (OcuChannel) detailed description (continued)**

| Property | Value |
|---|---|
| Origin | AUTOSAR_ECUC |
| Symbolic Name | false |
| Range | <=255<br>>=0 |

### 4.5.2.3 FtmHwChannel (OcuChannel)

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

**Table 4-19. Attribute FtmHwChannel (OcuChannel) detailed description**

| Property | Value |
|---|---|
| Type | STRING |
| Origin | Custom |
| Symbolic Name | false |
| Optional | true |

### 4.5.2.4 OcuDefaultThreshold (OcuChannel)

Value of comparison threshold used for Initialization.

**Table 4-20. Attribute OcuDefaultThreshold (OcuChannel) detailed description**

| Property | Value |
|---|---|
| Type | ENUMERATION |
| Origin | AUTOSAR_ECUC |
| Symbolic Name | false |
| Default | 10000 |
| Range | <=65535<br>>=0 |

### 4.5.2.5 OcuMaxCounterValue (OcuChannel)

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

**User Manual, Rev. 1.0**

**Table 4-21. Attribute OcuMaxCounterValue (OcuChannel) detailed description**

| Property | Value |
|---|---|
| Type | AUTOSAR_ECUC |
| Origin | AUTOSAR_ECUC |
| Symbolic Name | false |
| Default | 65535 |
| Invalid | Range<br>    <=65535<br>        >=0 |

## 4.5.2.6   OcuNotification (OcuChannel)

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.

**Table 4-22. Attribute OcuNotification (OcuChannel) detailed description**

| Property | Value |
|---|---|
| Type | FUNCTION-NAME |
| Origin | AUTOSAR_ECUC |
| Symbolic Name | false |
| Default | NULL_PTR |

## 4.5.2.7   OcuOuptutPinUsed (OcuChannel)

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

**Table 4-23. Attribute OcuOuptutPinUsed (OcuChannel) detailed description**

| Property | Value |
|---|---|
| Type | BOOLEAN |
| Origin | AUTOSAR_ECUC |
| Symbolic Name | false |
| Default | false |

**User Manual, Rev. 1.0**

## 4.5.2.8  OcuOutputPinDefaultState (OcuChannel)

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

**Table 4-24.  Attribute OcuOutputPinDefaultState (OcuChannel) detailed description**

| Property | Value |
|---|---|
| Type | ENUMERATION |
| Origin | AUTOSAR_ECUC |
| Symbolic Name | false |
| Default | OCU_LOW |
| Enable | false |
| Range | OCU_HIGH OCU_LOW |

## 4.5.2.9  OcuOutputPinAction (OcuChannel)

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

**Table 4-25.  Attribute OcuOutputPinAction (OcuChannel) detailed description**

| Property | Value |
|---|---|
| Type | ENUMERATION |
| Origin | Custom |
| Symbolic Name | false |
| Default | OCU_DISABLE |
| Enable | false |
| Range | OCU_SET_LOW OCU_SET_HIGH OCU_TOGGLE OCU_DISABLE |

## 4.5.2.10  OcuUseMcuReferencePoint (OcuChannel)

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.

**User Manual, Rev. 1.0**

**Table 4-26.   Attribute OcuUseMcuReferencePoint (OcuChannel) detailed description**

| Property | Value |
|---|---|
| Type | BOOLEAN |
| Origin | Custom |
| Symbolic Name | false |
| Default | false |

## 4.5.2.11   OcuChannelTickDuration (OcuChannel)

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.

**Table 4-27.   Attribute OcuChannelTickDuration (OcuChannel) detailed description**

| Property | Value |
|---|---|
| Type | INTEGER |
| Origin | AUTOSAR_ECUC |
| Symbolic Name | false |
| Invalid | XPath Range<br>        <=65535<br>        >=0 |

## 4.5.2.12   OcuMcuClockReferencePoint (OcuChannel)

Reference to the Ftm clock source configuration, which will be configured from the MCU driver, the McuClockReferencePoint container, in McuClockSettingConfig. Selecting the desired Clock source from MCU is selectable by the Ocu Clock Source from the OcuHwSpecificSettings. For example, for OCU_EXTERNAL_CLOCK configured in Ocu Clock Source, only FLEXTIMER0_EXT_CLK may be selected as a clock source from MCU. For more information please see the Ftm subchapter from the Clocking chapter in the RM.

**Table 4-28.   Attribute OcuMcuClockReferencePoint (OcuChannel) detailed description**

| Property | Value |
|---|---|
| Type | REFERENCE |
| Origin | Custom |

## 4.5.3   Form OcuHWSpecificSettings

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

**Is included by form :** Form OcuConfigSet



**Figure 4-7. Tresos Plugin snapshot for OcuHWSpecificSettings form.**

### 4.5.3.1   OcuHardwareChannelId (OcuHWSpecificSettings)

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

### 4.5.3.2   OcuFtmModule (OcuHWSpecificSettings)

Selects one Ftm modules available on the platform.

**Table 4-29.   Attribute OcuFtmModule (OcuHWSpecificSettings) detailed description**

| Property | Value |
|----------|-------|
| Type | STRING |
| Origin | NXP |

*Table continues on the next page...*

**User Manual, Rev. 1.0**

**Table 4-29.** **Attribute OcuFtmModule (OcuHWSpecificSettings) detailed description (continued)**

| Property | Value |
|----------|-------|
| Symbolic Name | false |
| Range | FTM_0, FTM_1, FTM_2, FTM_3, FTM_4, FTM_5, FTM_6, FTM_7 |

### 4.5.3.3  OcuClockSource (OcuHWSpecificSettings)

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.

**Table 4-30.** **Attribute OcuClockSource (OcuHWSpecificSettings) detailed description**

| Property | Value |
|----------|-------|
| Type | ENUMERATION |
| Origin | Custom |
| Symbolic Name | false |
| Optional | true |
| Range | OCU_SYSTEM_CLOCK, OCU_EXTERNAL_CLOCK |

### 4.5.3.4  OcuPrescale (OcuHWSpecificSettings)

Specifies the default threshold for the channel after Ocu_StartChannel.

**Table 4-31.** **Attribute OcuPrescale (OcuHWSpecificSettings) detailed description**

| Property | Value |
|----------|-------|
| Type | ENUMERATION |
| Origin | Custom |
| Symbolic Name | false |
| Optional | true |
| Range | DIV1<br>DIV2<br>DIV4<br>DIV8<br>DIV16<br>DIV32<br>DIV64<br>DIV128 |

**User Manual, Rev. 1.0**

## 4.5.3.5   OcuPrescale_Alternate (OcuHWSpecificSettings)

Select input count source (clock) used for this eTimer channel. This parameter will be used by the Ocu_SetClockMode function.

**Table 4-32.   Attribute OcuPrescale_Alternate (OcuHWSpecificSettings) detailed description**

| Property | Value |
|---|---|
| Type | ENUMERATION |
| Origin | Custom |
| Symbolic Name | false |
| Optional | true |
| Range | DIV1<br>DIV2<br>DIV4<br>DIV8<br>DIV16<br>DIV32<br>DIV64<br>DIV128 |

## 4.5.3.6   OcuPrescale_Alternate (OcuHWSpecificSettings)

Select input count source (clock) used for this FTM module. This parameter will be used by the Ocu_SetClockMode function.

**Table 4-33.   Attribute OcuPrescale_Alternate (OcuHWSpecificSettings) detailed description**

| Property | Value |
|---|---|
| Type | ENUMERATION |
| Origin | Custom |
| Symbolic Name | false |
| Range | DIV1<br>DIV2<br>DIV4<br>DIV8<br>DIV16<br>DIV32<br>DIV64<br>DIV128 |

### 4.5.3.7 OcuDebugMode (OcuHWSpecificSettings)

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

**Table 4-34.  Attribute OcuDebugMode (OcuHWSpecificSettings) detailed description**

| Property | Value |
|---|---|
| Type | ENUMERATION |
| Origin | Custom |
| Symbolic Name | false |
| Range | CNT_ON_OUTPUTS_ON, CNT_STOPED_OUTPUTS_FROZEN, CNT_STOPED_OUTPUTS_SAFE, CNT_STOPED_FLAG_SET |

## 4.6  Form CommonPublishedInformation

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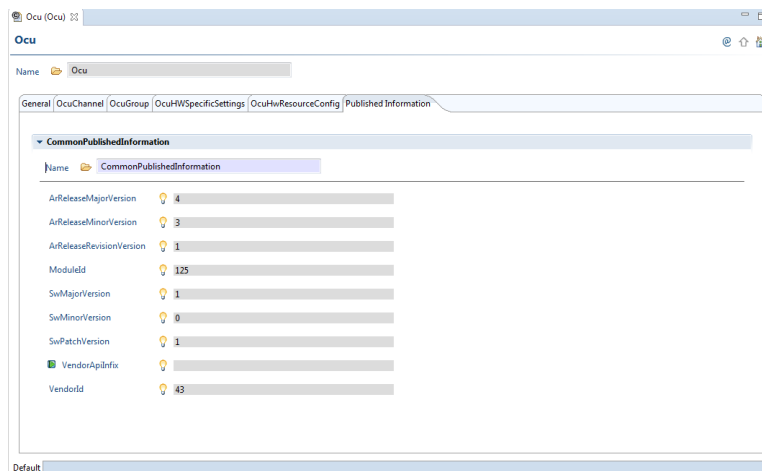


**Figure 4-8. Tresos Plugin snapshot for CommonPublishedInformation form.**

## 4.6.1   ArReleaseMajorVersion (CommonPublishedInformation)

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

**Table 4-35.   Attribute ArReleaseMajorVersion (CommonPublishedInformation) detailed description**

| Property | Value |
|---|---|
| Label | AUTOSAR Major Version |
| Type | INTEGER_LABEL |
| Origin | Custom |
| Symbolic Name | false |
| Default | 4 |
| Invalid | Range<br>        >=4<br>        <=4 |

## 4.6.2   ArReleaseMinorVersion (CommonPublishedInformation)

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

**Table 4-36.   Attribute ArReleaseMinorVersion (CommonPublishedInformation) detailed description**

| Property | Value |
|---|---|
| Label | AUTOSAR Minor Version |
| Type | INTEGER_LABEL |
| Origin | Custom |
| Symbolic Name | false |
| Default | 3 |
| Invalid | Range<br>        >=3<br>        <=3 |

## 4.6.3   ArReleaseRevisionVersion (CommonPublishedInformation)

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

**User Manual, Rev. 1.0**

NXP Semiconductors

**Table 4-37.   Attribute ArReleaseRevisionVersion (CommonPublishedInformation) detailed description**

| Property | Value |
| --- | --- |
| Label | AUTOSAR Release Revision Version |
| Type | INTEGER_LABEL |
| Origin | Custom |
| Symbolic Name | false |
| Default | 1 |
| Invalid | Range<br>        >=1<br>        <=1 |

## 4.6.4   ModuleId (CommonPublishedInformation)

Module ID of this module from Module List.

**Table 4-38.   Attribute ModuleId (CommonPublishedInformation) detailed description**

| Property | Value |
| --- | --- |
| Label | Module Id |
| Type | INTEGER_LABEL |
| Origin | Custom |
| Symbolic Name | false |
| Default | 125 |
| Invalid | Range<br>        >=125<br>        <=125 |

## 4.6.5   SwMajorVersion (CommonPublishedInformation)

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

**Table 4-39.   Attribute SwMajorVersion (CommonPublishedInformation) detailed description**

| Property | Value |
| --- | --- |
| Label | Software Major Version |
| Type | INTEGER_LABEL |
| Origin | Custom |
| Symbolic Name | false |

*Table continues on the next page...*

**User Manual, Rev. 1.0**

**Table 4-39. Attribute SwMajorVersion (CommonPublishedInformation) detailed description (continued)**

| Property | Value |
|---|---|
| Default | 1 |
| Invalid | Range<br>>=1<br><=1 |

## 4.6.6 SwMinorVersion (CommonPublishedInformation)

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

**Table 4-40. Attribute SwMinorVersion (CommonPublishedInformation) detailed description**

| Property | Value |
|---|---|
| Label | Software Minor Version |
| Type | INTEGER_LABEL |
| Origin | Custom |
| Symbolic Name | false |
| Default | 0 |
| Invalid | Range<br>>=0<br><=0 |

## 4.6.7 SwPatchVersion (CommonPublishedInformation)

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

**Table 4-41. Attribute SwPatchVersion (CommonPublishedInformation) detailed description**

| Property | Value |
|---|---|
| Label | Software Patch Version |
| Type | INTEGER_LABEL |
| Origin | Custom |
| Symbolic Name | false |
| Default | 1 |
| Invalid | Range<br>>=1<br><=1 |

## 4.6.8 VendorApiInfix (CommonPublishedInformation)

In driver modules which can be instantiated several times on a single ECU, BSW00347 requires that the name of APIs is extended by the VendorId and a vendor specific name. This parameter is used to specify the vendor specific name. In total, the implementation specific name is generated as follows:
<ModuleName>_<VendorId>_<VendorApiInfix><Api name from SWS>. 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.

**Table 4-42.   Attribute VendorApiInfix (CommonPublishedInformation) detailed description**

| Property | Value |
|---|---|
| Label | Vendor Api Infix |
| Type | STRING_LABEL |
| Origin | Custom |
| Symbolic Name | false |
| Default | |
| Enable | false |

## 4.6.9 VendorId (CommonPublishedInformation)

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

**Table 4-43.   Attribute VendorId (CommonPublishedInformation) detailed description**

| Property | Value |
|---|---|
| Label | Vendor Id |
| Type | INTEGER_LABEL |
| Origin | Custom |
| Symbolic Name | false |
| Default | 43 |
| Invalid | Range<br>    >=43<br>    <=43 |