User Manual

for S32K3 PWM Driver

Document Number: UM34PWMASRR21-11 Rev0000R3.0.0 Rev. 1.0

Chapter 1

Revision History

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

Chapter 2

Introduction

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

2.1 Supported Derivatives

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

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

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

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

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

2.2 Overview

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

AUTOSAR:

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

2.3 About This Manual

This Technical Reference employs the following typographical conventions:

- **Boldface** style: Used for important terms, notes and warnings.
- *Italic* style: Used for code snippets in the text. Note that C language modifiers such "const" or "volatile" are sometimes omitted to improve readability of the presented code.

Notes and warnings are shown as below:

Note

This is a note.

Warning

This is a warning

2.4 Acronyms and Definitions

Term	Definition
API	Application Programming Interface
ASM	Assembler
BSMI	Basic Software Make file Interface
CAN	Controller Area Network
C/CPP	C and C++ Source Code
CS	Chip Select
CTU	Cross Trigger Unit
DEM	Diagnostic Event Manager
DET	Development Error Tracer
DMA	Direct Memory Access
ECU	Electronic Control Unit
FIFO	First In First Out
LSB	Least Signifigant Bit
MCU	Micro Controller Unit
MIDE	Multi Integrated Development Environment
MSB	Most Significant Bit
N/A	Not Applicable
RAM	Random Access Memory
SIU	Systems Integration Unit
SWS	Software Specification
VLE	Variable Length Encoding
XML	Extensible Markup Language

2.5 Reference List

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

Chapter 3

Driver

3.1 Requirements

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

3.2 Driver Design Summary

The driver provides functions for initialization and control of the micro-controller internal PWM stage (pulse width modulation). The PWM module generates pulses with variable pulse width. It allows the selection of the duty cycle and the signal period time. Each PWM channel is linked to a hardware channel capable of implementing PWM functionality, which belongs to the micro-controller.

The S32K39X micro-controller contains one instance of Emios modules and one instance of FlexIO module. The S32K314, S32K324, S32K328, S32K338, S32K344, S32K348, S32K358, S32K388 micro-controller contains three instances of Emios modules and one instance of FlexIO module. The S32K341, S32K342, S32K322, S32K310, S32↩ K311, S32K312 and S32M7X micro-controller contains two instances of Emios modules and one instance of FlexIO module.

The Emios module has the following major features in this release:

- Up to 24 channels chosen among Unified or Dedicated Channels, not necessarily numbered in a continuous sequence.
- Data registers of either 8-, 16-, 24-, or 32-bit width.
- Counter buses B, C, D, and E can be driven by Unified Channels 0, 8, 16, and 24, respectively.
- Counter bus A can be driven by Unified Channel 23.
- Counter bus F can be driven by a specified Unified Channel, defined by the system configuration.
- Each channel has its own timebase, alternative to the counter buses.
- Two global prescalers.
- One prescaler per channel (CP).
- State of the Unified Channels can be frozen for debug purposes.
- Output Pulse-Width and Frequency Modulation Buffered (OPWFMB).

The Flexio module has the following major features in this release:

- Up to 8 channels chosen among the Flexio timers.
- Up to 32 pins that can be chosen for any channel.
- Timer 8-bit High PWM Mode.
- Timer 8-bit Low PWM Mode.
- Pin output override mode to drive either HIGH or LOW signal on channel output.
- One prescaler per channel.
- State of the timers can be frozen for debug purposes.

3.3 Hardware Resources

Resources available on each derivative.

Derivatives	Pwm Module	Pwm Channel available
s32k310_lqfp48	Emios 0, Emios 1	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7
s32k310_mqfp100	Emios 0, Emios 1	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23

Derivatives	Pwm Module	Pwm Channel available
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7
s32k311_lqfp48	Emios 0, Emios 1	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7
s32k311_mqfp100	Emios 0, Emios 1	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7
s32k312_mqfp100	Emios 0, Emios 1	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7
s32k312_mqfp172	Emios 0, Emios 1	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7
s32k314_mapbga257	Emios 0, Emios 1, Emios 2	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7
s32k314_mqfp100	Emios 0, Emios 1, Emios 2	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7

Derivatives	Pwm Module	Pwm Channel available
s32k314_mqfp172	Emios 0, Emios 1, Emios 2	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7
s32k322_mqfp100	Emios 0, Emios 1	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7
s32k322_mqfp172	Emios 0, Emios 1	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7
s32k324_mapbga257	Emios 0, Emios 1, Emios 2	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7
s32k324_mqfp172	Emios 0, Emios 1, Emios 2	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7
s32k328_mapbga289	Emios 0, Emios 1, Emios 2	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7

Derivatives	Pwm Module	Pwm Channel available
s32k328_mqfp172	Emios 0, Emios 1, Emios 2	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7
s32k338_mapbga289	Emios 0, Emios 1, Emios 2	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7
s32k338_mqfp172	Emios 0, Emios 1, Emios 2	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7
s32k341_mqfp100	Emios 0, Emios 1	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7
s32k341_mqfp172	Emios 0, Emios 1	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7
s32k342_mqfp100	Emios 0, Emios 1	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7

Derivatives	Pwm Module	Pwm Channel available
s32k342_mqfp172	Emios 0, Emios 1	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7
s32k344_mapbga257	Emios 0, Emios 1, Emios 2	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7
s32k344_mqfp172	Emios 0, Emios 1, Emios 2	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7
s32k348_mapbga289	Emios 0, Emios 1, Emios 2	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7
s32k348_mqfp172	Emios 0, Emios 1, Emios 2	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7
s32k358_mapbga289	Emios 0, Emios 1, Emios 2	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7

Derivatives	Pwm Module	Pwm Channel available
s32k358_mqfp172	Emios 0, Emios 1, Emios 2	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7
s32k388_mapbga289	Emios 0, Emios 1, Emios 2	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7
s32k394_mapbga289	Emios 0	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7
s32k396_mapbga289	Emios 0	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7
s32k396_mqfp172	Emios 0	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7
s32m274_lqfp64	Emios 0, Emios 1	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7

Derivatives	Pwm Module	Pwm Channel available
s32m276_lqfp64	Emios 0, Emios 1	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7, Channel 8, Channel 9, Channel 10, Channel 11, Channel 12, Channel 13, Channel 14, Channel 15, Channel 16, Channel 17, Channel 18, Channel 19, Channel 20, Channel 21, Channel 22, Channel 23
	Flexio	Channel 0, Channel 1, Channel 2, Channel 3, Channel 4, Channel 5, Channel 6, Channel 7

3.4 Deviations from Requirements

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

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

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

Requirement	Status	Description	Notes
SWS_Pwm_00065	N/S	The Pwm SWS shall not define the code file structure.	Code file structure does always not define in the driver.
SWS_Pwm_70075	N/S	Pwm_Irq.c shall include Pwm.h.	Pwm_Irq.c is not needed. Autosar specific interrupt behaviour is implemented using a normal function placed in the Pwm.c file.
SWS_Pwm_00162	N/S	The Pwm Driver shall support synchronuous and asynchronous power state transitions, depending on the value of the configuration parameter PwmPower↩ StateAsynchTransitionMode.	Rejected: All PWM hardware do not support asynchronous power state transitions
SWS_Pwm_00164	N/S	In case the configuration parameter PwmPowerState↩ AsynchTransitionMode is set to TRUE, the preparation process shall continue in background after the relative API returns and its completion shall be notified by means of the configured callback.	Rejected: All PWM hardware do not support asynchronous power state transitions

Requirement	Status	Description	Notes
SWS_Pwm_00189	N/S	Service name: - Pwm_Main↩ _PowerTransitionManager - Syntax: - void Pwm_Main_↩ PowerTransitionManager(void) - Service ID[hex]: - 0x0d - Description: - This API is cyclically called and supervises the power state transitions, checking for the readiness of the module and issuing the callbacks IoHwAb_Pwm_↩ NotifyReadyForPowerState (see PwmPowerStateReadyCbkRef configuration parameter). - Available via: - SchM_Pwm.h -	Rejected: All PWM hardwares do not support asynchronous power state transitions
SWS_Pwm_00190	N/S	This API executes any non-immediate action needed to finalize a power state transition requested by Pwm_PreparePowerState().	Rejected: All PWM hardwares do not support asynchronous power state transitions
SWS_Pwm_00191	N/S	The rate of scheduling shall be defined by Pwm Main↩ SchedulePeriod and shall be variable, as the function only needs to be called if a transition has been requested.	Rejected: All PWM hardwares do not support asynchronous power state transitions
SWS_Pwm_00192	N/S	This API shall also issue call-back notifications to the eventually registered users (IoHwAbs) as configured, only in case the asynch mode is chosen.	Rejected: All PWM hardwares do not support asynchronous power state transitions

Requirement	Status	Description	Notes
SWS_Pwm_00193	N/S	In case the PWM module is not initialized, this function shall simply return without any further elaboration. This is needed to avoid to elaborate uninitialized variables. No DET error shall be entered, because this condition can easily be verified during the startup phase (tasks started before the initialization is complete).Rationale: during the startup phase it can happen that the OS already schedules tasks, which call main functions, while some modules are not initialised yet. This is no real error condition, although need handling, i.e. returning without execution.Although the transition state monitoring functionality is mandatory, the implementation of this API is optional, meaning that if the HW allows for other ways to deliver notification and watch the transition state the implementation of this function can be skipped.	Rejected: All PWM hardwares do not support asynchronous power state transitions
SWS_Pwm_00199	N/S	In case the PWM Driver is configured to support power state management with asynchronous transitions, this API shall be called to signal completion of the power transition preparation phase to the IoHwAbs module.This is a callback, this API is to be implemented in the IoHwAbs component.	Rejected: All PWM hardwares do not support asynchronous power state transitions
SWS_Pwm_00153	N/S	These requirements are not applicable to this specification.	not a requirement

Requirement	Status	Description	Notes
ECUC_Pwm_00143	N/S	Name - PwmPowerState↩ AsynchTransitionMode - Parent Container - PwmGeneral - Description - Enables / disables support of the PWM Driver to the asynchronous power state transition. - Multiplicity - 0..1 - Type - EcucBooleanParamDef - Default value - false - Post-Build Variant Multiplicity - false - Post-Build Variant Value - false - Multiplicity Configuration Class - Pre-compile time - X - All Variants - Link time - – - - Post-build time - – - - Value Configuration Class - Pre-compile time - X - All Variants - Link time - – - - Post-build time - – - - Scope / Dependency - scope: localdependency↩ : This parameter shall only be configured if the parameter PwmLowPowerStatesSupport is set to true. -	Rejected: All PWM hardwares do not support asynchronous power state transitions
ECUC_Pwm_00144	N/S	Container Name - PwmPower↩ StateConfig - Description - Each instance of this parameter defines a power state and the callback to be called when this power state is reached. - Configuration Parameters -	Rejected: All PWM hardwares do not support asynchronous power state transitions

Requirement	Status	Description	Notes
ECUC_Pwm_00146	N/S	Name - PwmPowerState - Parent Container - PwmPower↩ StateConfig - Description - Each instance of this parameter describes a different power state supported by the PWM HW. It should be defined by the HW supplier and used by the P↩ WMDriver to reference specific HW configurations which set the PWM HW module in the referenced power state.At least the power mode corresponding to full power state shall be always configured. - Multiplicity - 1 - Type - EcucIntegerParamDef (Symbolic Name generated for this parameter) - Range - 0 .. 18446744073709551615 - - Default value - – - Post-Build Variant Value - false - Value Configuration Class - Pre-compile time - X - All Variants - Link time - – - - Post-build time - – - - Scope / Dependency - scope↩ : localdependency: This parameter shall only be configured if the parameter PwmLowPower↩ StatesSupport is set to true. -	Rejected: All PWM hardwares do not support asynchronous power state transitions
ECUC_Pwm_00145	N/S	Name - PwmPowerState↩ ReadyCbkRef - Parent Container - PwmPowerStateConfig - Description - Each instance of this parameter contains a reference to a power mode callback defined in a CDD or IoHwAbs component. - Multiplicity - 1 - Type - EcucFunctionNameDef - Default value - – - maxLength - – - minLength - – - regular↩ Expression - – - Post-Build Variant Value - false - Value Configuration Class - Pre-compile time - X - All Variants - Link time - – - - Post-build time - – - - Scope / Dependency - scope: localdependency↩ : This parameter shall only be configured if the parameter PwmLowPowerStatesSupport is set to true. -	Rejected: All PWM hardwares do not support asynchronous power state transitions

Requirement	Status	Description	Notes
SWS_Pwm_CONSTR_00001	N/S	DRAFT: The ECUC partitions referenced by PwmKernel↩ EcucPartitionRef shall be a subset of the ECUC partitions referenced by PwmEcuc↩ PartitionRef.	Type IV Autosar multicore not implemented for current module (AAI-445), therefore Pwm↩ KernelEcucPartitionRef is not supported
SWS_Pwm_91003	N/S	Service name: - Pwm↩ _DisableNotification (draft) - Syntax: - void Pwm_DisableNotification(Pwm_ChannelType ChannelNumber) - Service ID[hex]: - 0x06 - Sync/Async: - Asynchronous - Reentrancy: - Reentrant for different channel numbers - Parameters (in): - Channel↩ Number - Numeric identifier of the PWM - Parameters (inout): - None - Parameters (out): - None - Return value: - None - Description: - Service to disable the PWM signal edge notification.Tags: atp.↩ Status=draft - Available via: - Pwm.h -	Description specified as draft is not clear. Should be re-assesed on next ASR version
SWS_Pwm_91004	N/S	Service name: - Pwm↩ _EnableNotification (draft) - Syntax: - void Pwm_EnableNotification(Pwm_ChannelType ChannelNumber, Pwm_EdgeNot - Service ID[hex]: - 0x07 - Sync/Async: - Asynchronous - Reentrancy: - Reentrant for different channel numbers - Parameters (in): - ChannelNumber - Numeric identifier of the PWM - Notification - Type of notification PWM_RISING_EDGE or PWM_FALLING_EDGE or PWM_BOTH_EDGES - Parameters (inout): - None - Parameters (out): - None - Return value: - None - Description: - Service to enable the PWM signal edge notification according to notification parameter.Tags: atp.Status=draft - Available via: - Pwm.h -	Description specified as draft is not clear. Should be re-assesed on next ASR version

Requirement	Status	Description	Notes
SWS_Pwm_91000	N/S	Service name: - Pwm_SetDutyCycle (draft) - Syntax: - void Pwm_SetDutyCycle(Pwm_ChannelType ChannelNumber, uint16 DutyCycle) - Service ID[hex]: - 0x02 - Sync/Async: - Asynchronous - Reentrancy: - Reentrant for different channel numbers - Parameters (in): - ChannelNumber - Numeric identifier of the PWM - DutyCycle - Min=0x0000 Max=0x8000 - Parameters (inout): - None - Parameters (out): - None - Return value: - None - Description: - Service sets the duty cycle of the PWM channel.Tags: atp.Status=draft - Available via: - Pwm.h -	Description specified as draft is not clear. Should be re-assesed on next ASR version
SWS_Pwm_91002	N/S	Service name: - Pwm_SetOutputToIdle (draft) - Syntax: - void Pwm_SetOutputToIdle(Pwm_ChannelType ChannelNumber) - Service ID[hex]: - 0x04 - Sync/Async: - Asynchronous - Reentrancy: - Reentrant for different channel numbers - Parameters (in): - ChannelNumber - Numeric identifier of the PWM - Parameters (inout): - None - Parameters (out): - None - Return value: - None - Description: - Service sets the PWM output to the configured Idle state.Tags: atp.Status=draft - Available via: - Pwm.h -	Description specified as draft is not clear. Should be re-assesed on next ASR version

Requirement	Status	Description	Notes
SWS_Pwm_91001	N/S	Service name: - Pwm↩ _SetPeriodAndDuty (draft) - Syntax: - void Pwm_SetPeriodAndDuty(Pwm_ChannelType ChannelNumber, Pwm_PeriodTy - Service ID[hex]: - 0x03 - Sync/Async: - Asynchronous - Reentrancy: - Reentrant for different channel numbers - Parameters (in): - Channel↩ Number - Numeric identifier of the PWM - Period - Period of the PWM signal - DutyCycle - Min=0x0000 Max=0x8000 - Parameters (inout): - None - Parameters (out): - None - Return value: - None - Description: - Service sets the period and the duty cycle of a PWM channelTags: atp.↩ Status=draft - Available via: - Pwm.h -	Description specified as draft is not clear. Should be re-assesed on next ASR version
ECUC_Pwm_00150	N/S	Name - PwmKernelEcuc↩ PartitionRef - Parent Container - PwmGeneral - Description - Maps the PWM 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 PWM driver is mapped to.Tags: atp.Status=draft - Multiplicity - 0..1 - Type - Reference to [EcucPartition] - Post-Build Variant Multiplicity - true - Post-Build Variant Value - true - Multiplicity Configuration Class - Pre-compile time - X - All Variants - Link time - – - - Post-build time - – - - Value Configuration Class - Pre-compile time - X - All Variants - Link time - – - - Post-build time - – - - Scope / Dependency - scope: ECU -	Type IV Autosar multicore not implemented for current module (AAI-445), therefore Pwm↩ KernelEcucPartitionRef is not supported
SWS_Pwm_00175	N/S	The API shall report the DET error PWM_E_TRANSITIO↩ N_NOT_POSSIBLE in case the requested power state cannot be directly reached from the current power state.	Currently, Pwm driver has only 2 powe state (High power and Low Power) so this DET error will not happen

Requirement	Status	Description	Notes
SWS_Pwm_00195	N/S	The API shall report the DET error PWM_E_TRANSITIO↩ N_NOT_POSSIBLE in case the requested power state cannot be directly reached from the current power state.	Currently, Pwm driver has only 2 powe state (High power and Low Power) so this DET error will not happen
SWS_Pwm_00188	N/S	The API shall report the DET error PWM_E_TRANSITIO↩ N_NOT_POSSIBLE in case the requested power state cannot be directly reached from the current power state.All asynchronous operation needed to reach the target power state can be executed in background in the context of Pwm_Main_↩ PowerTransitionManager.	s

3.5 Driver Limitations

When Pwm_SetDutyCycles and Pwm_SetPeriodAndDuty functions are used for setting the channel X of the Flex↩ PWM submodule N (N=0, 1, 2, 3), if duty cycle is changed from 0% to other values or from other duty cycles to 0%, the fisrt transforming pulse may not be as expected. This is due to a hardware limitation in which channel polarity bits are not bufferable, while channel value registers (VAL0 - VAL5) are buffered registers. In accordance with SWS_Pwm_00014, for 0% requested Duty Cycle the output will be the inverse of the configured polarity parameter. So changing the duty cycle from other values to 0% and vice versa requires polarity bits be reversed. Because of the hardware limitation, duty and period values will be applied in next period, but changing polarity will take effect immediately.

Current PWM driver does not support the OPWMC mode.

3.6 Driver usage and configuration tips

In order for the Pwm driver to function without problems the following steps must be taken:

- Initialize the MCU driver with the desired clock configuration
- Initialize the MCL driver with a valid Emios and Flexio instance configuration.
- Initialize the PORT driver with a desired pin configuration.
- Initialize the PWM driver.

The Global time base must be enabled by the MCL driver to provide a clock for each instance. In this release, we support two IPs: Emios, Flexio. Below's how to configure MCL driver for these IPs:

a) Emios IP:

Step 1: Enable Emios common support

-If users use EB tresos

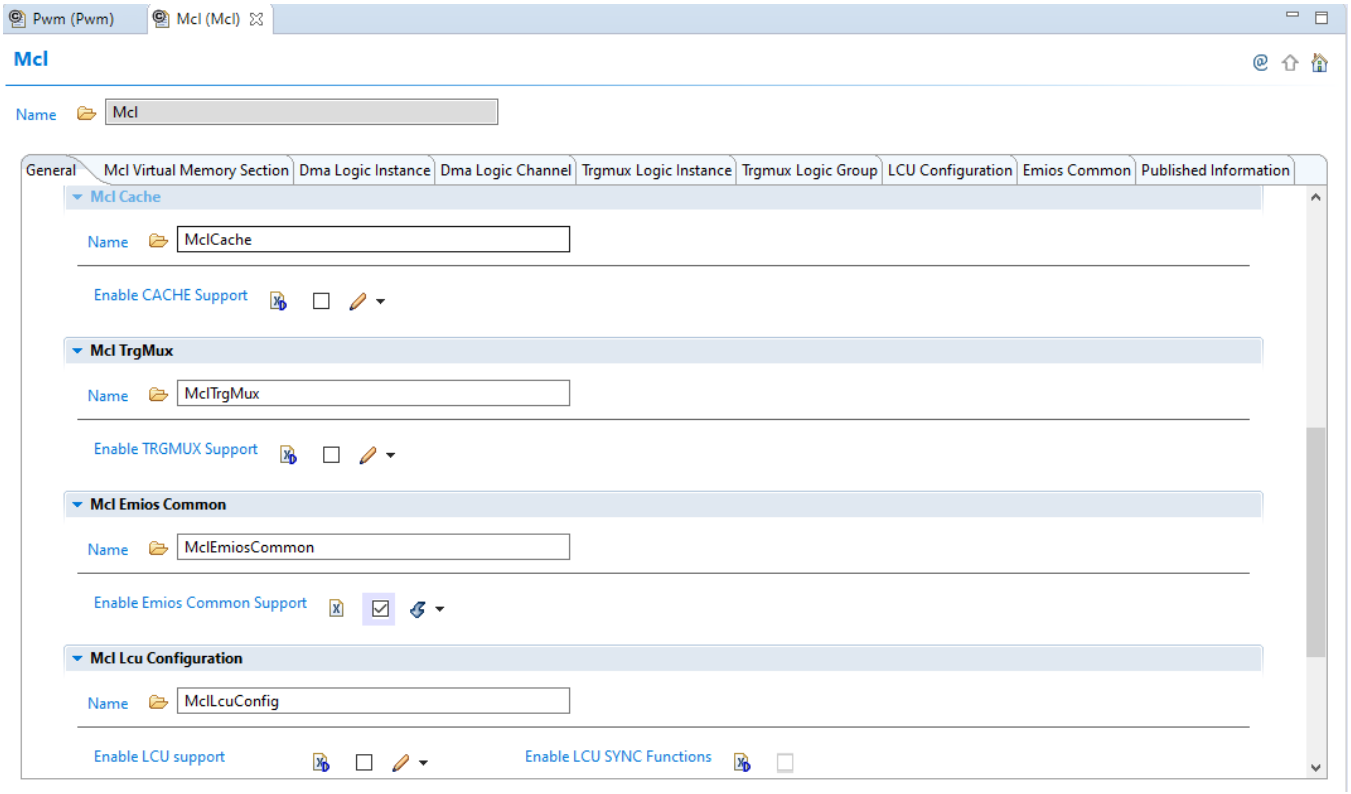

Figure 3.1 Enable Emios common support

-If users use S32DS with Pwm driver

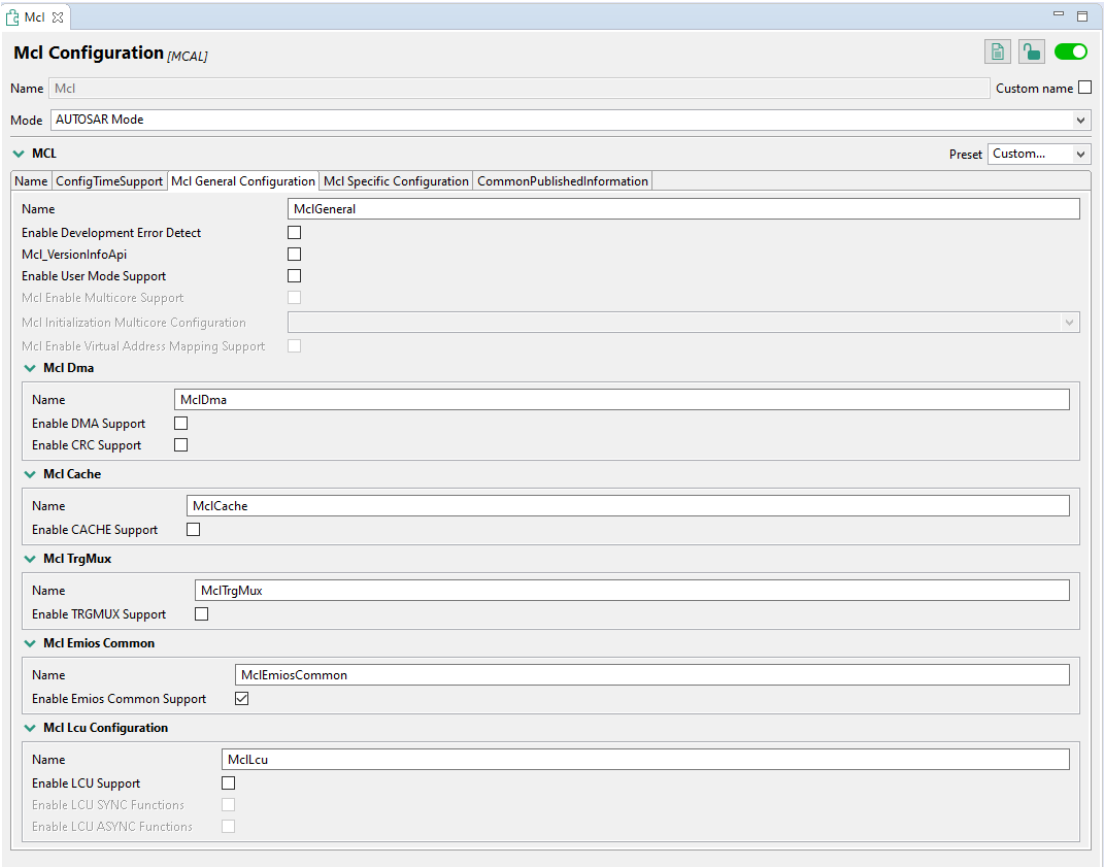

Figure 3.2 Enable Emios common support

-If users use S32DS with Emios_Pwm driver

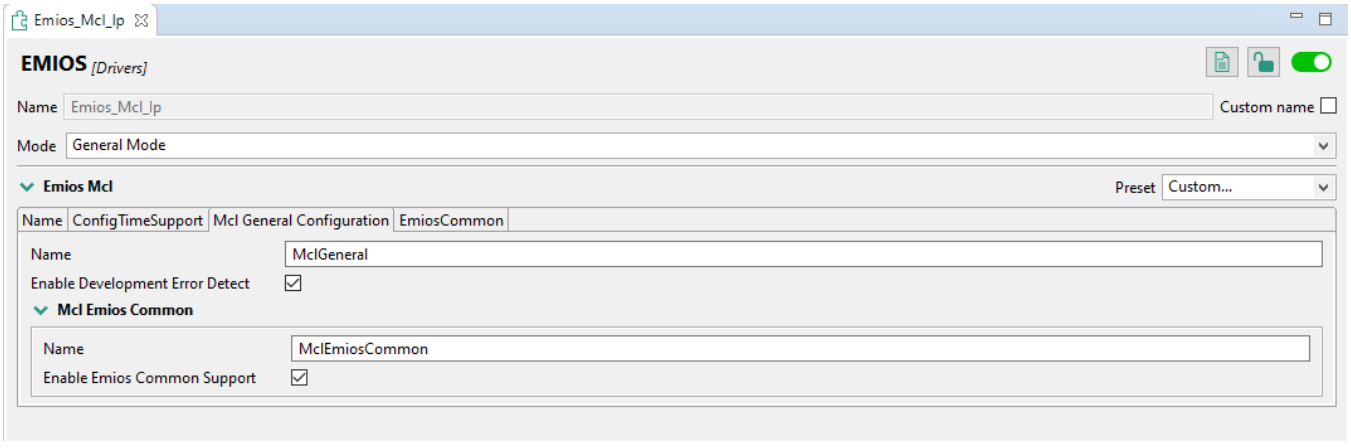

Figure 3.3 Enable Emios common support.

Step 2: Configure instance when Emios Common Support is checked

-If users use EB tresos

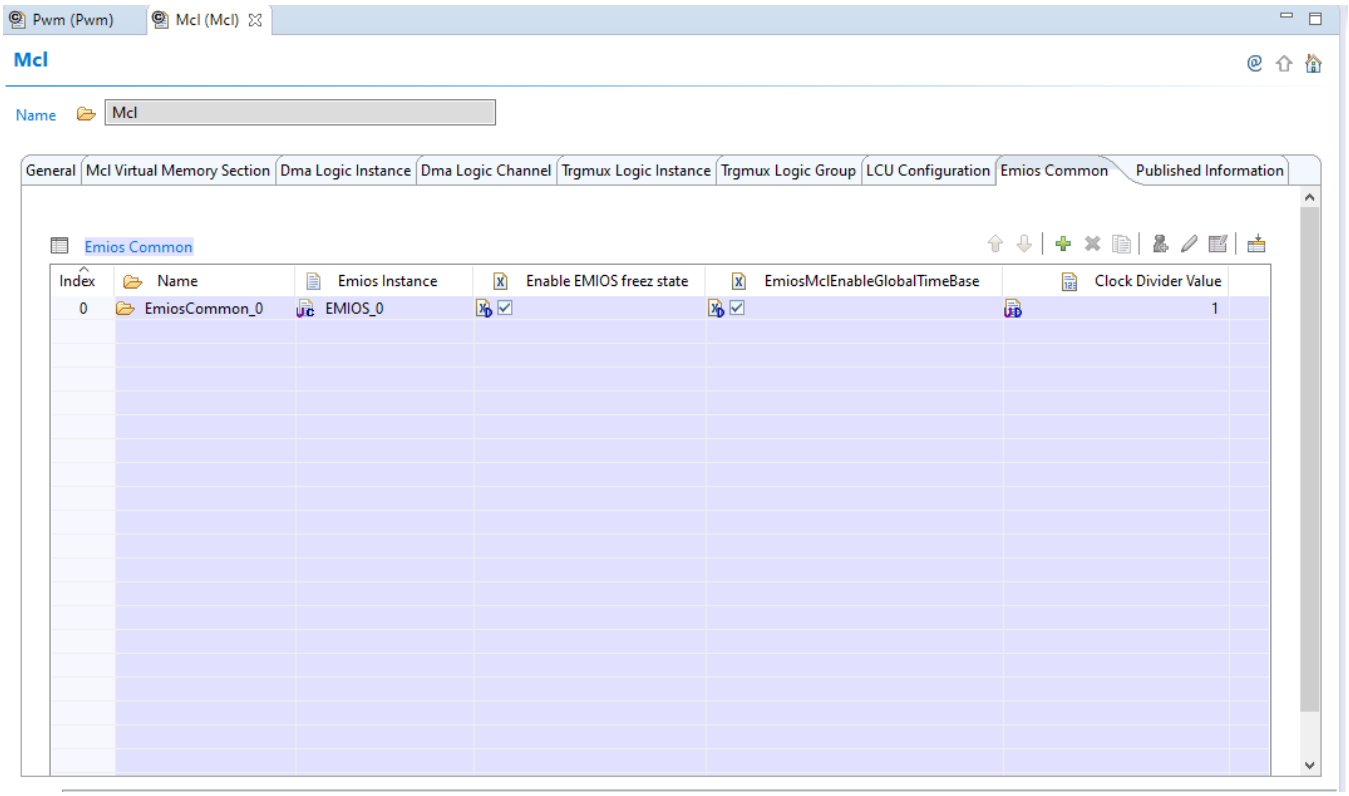

Figure 3.4 Configure instance.

-If users use S32DS with Pwm driver

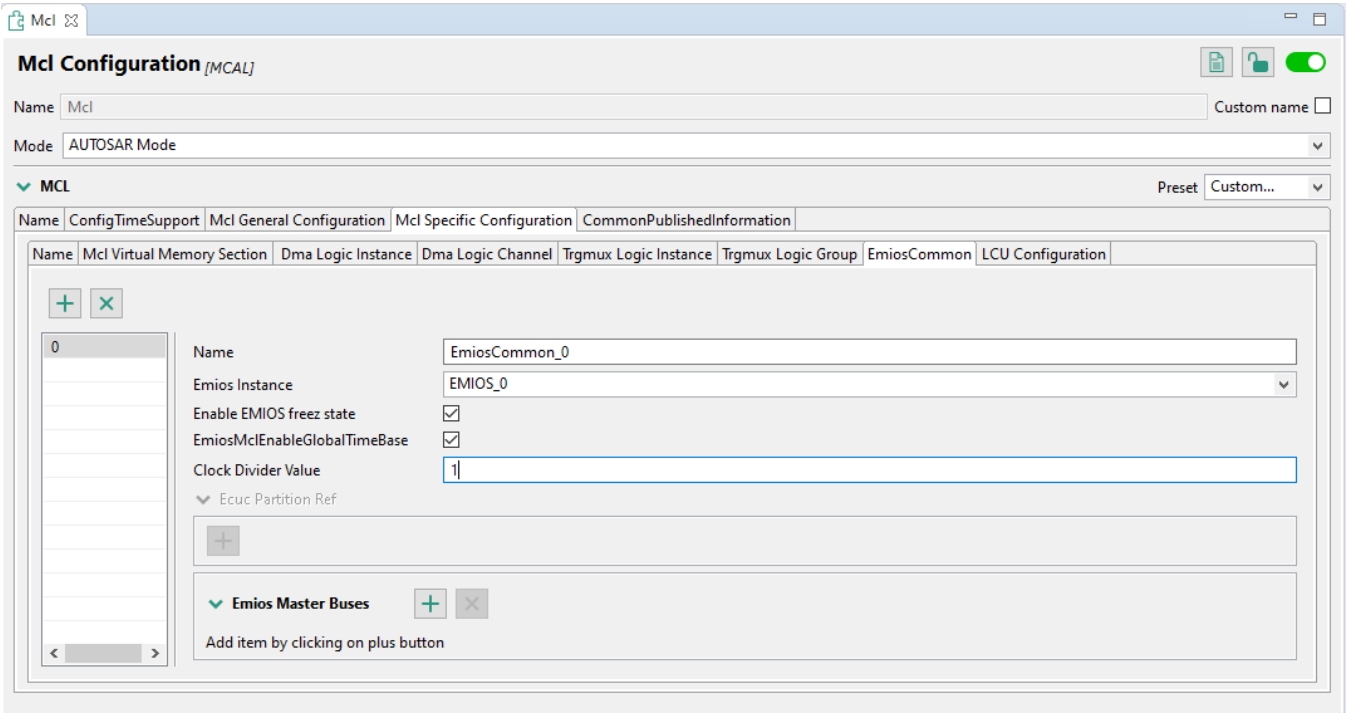

Figure 3.5 Configure instance.

-If users use S32DS with Emios_Pwm driver

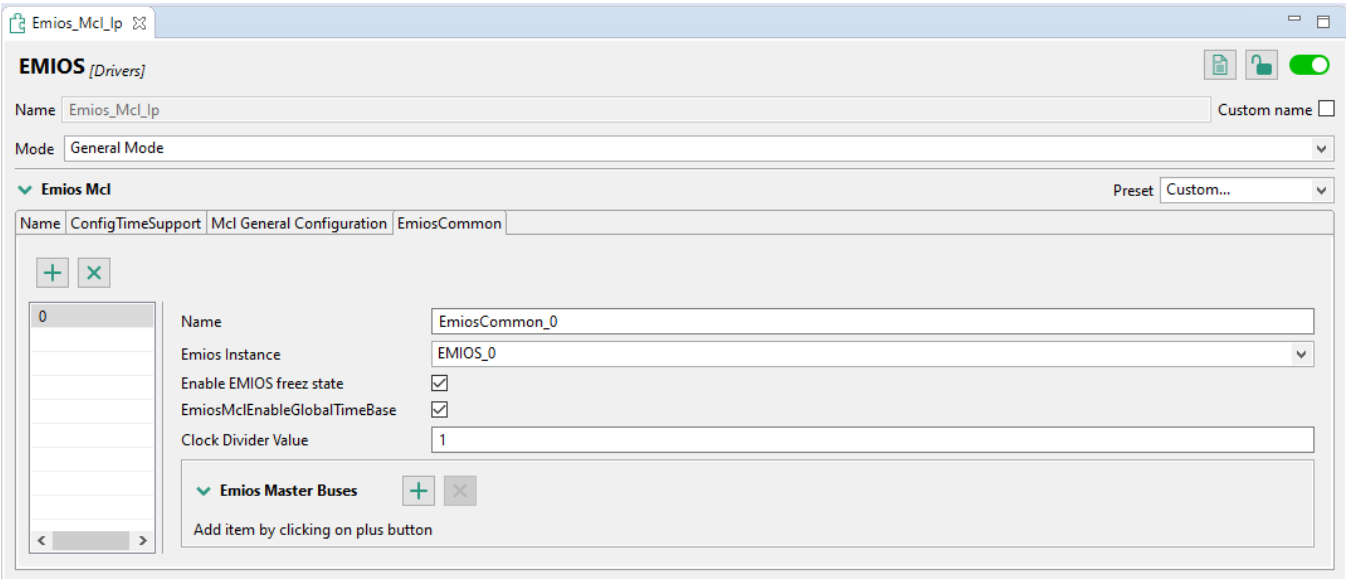

Figure 3.6 Configure instance.

Additional step: if users want to use the master bus (bus A, bus B, bus C, bus D, bus E or bus F)

-If users use EB tresos

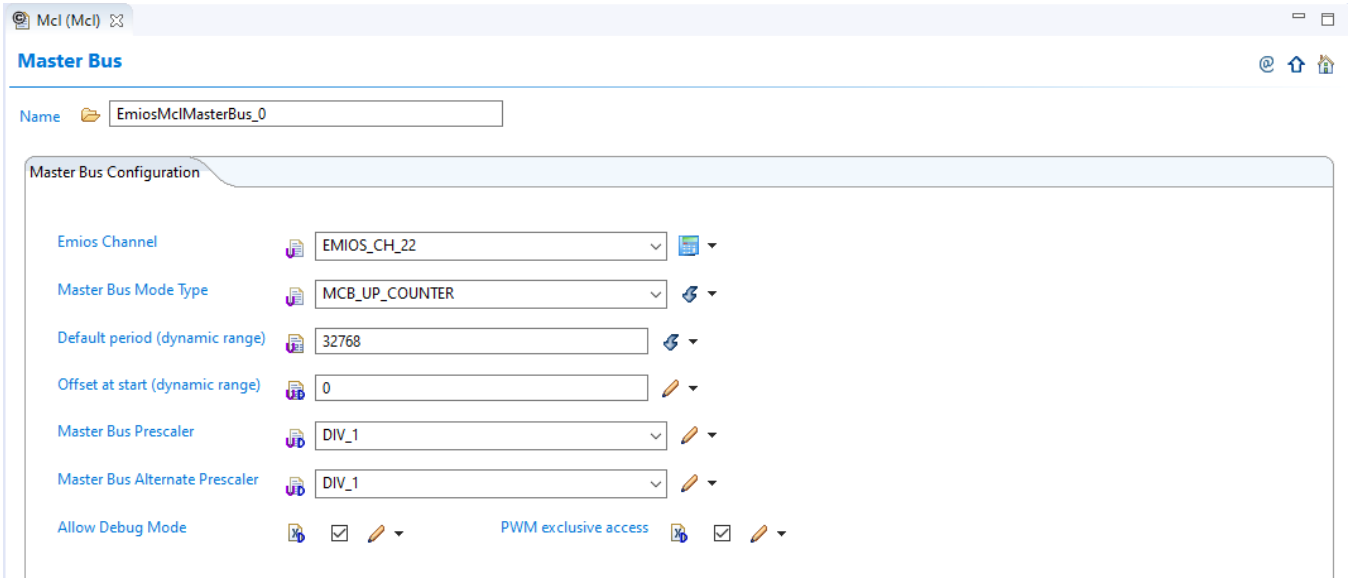

Figure 3.7 Configure Emios master bus.

-If users use S32DS with Pwm driver

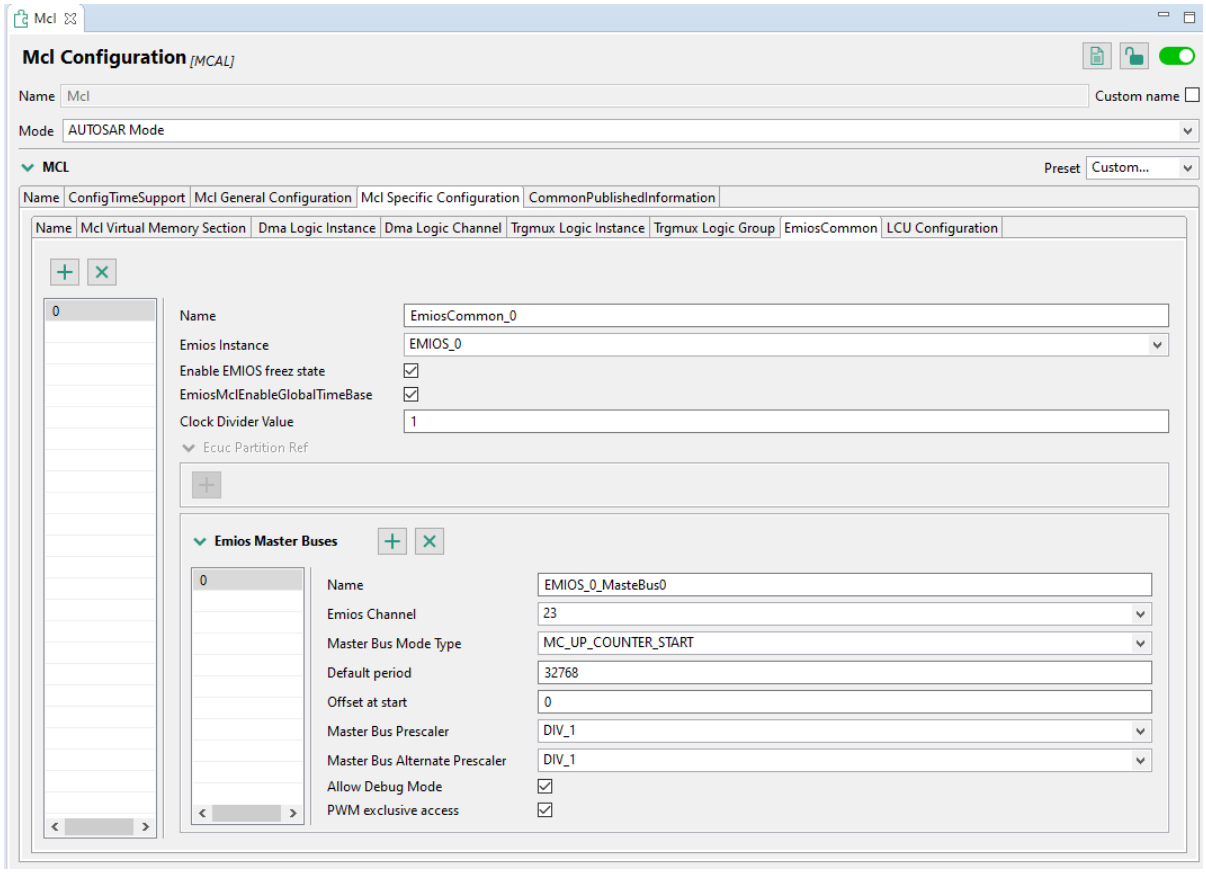

Figure 3.8 Configure instance.

-If users use S32DS with Emios_Pwm driver

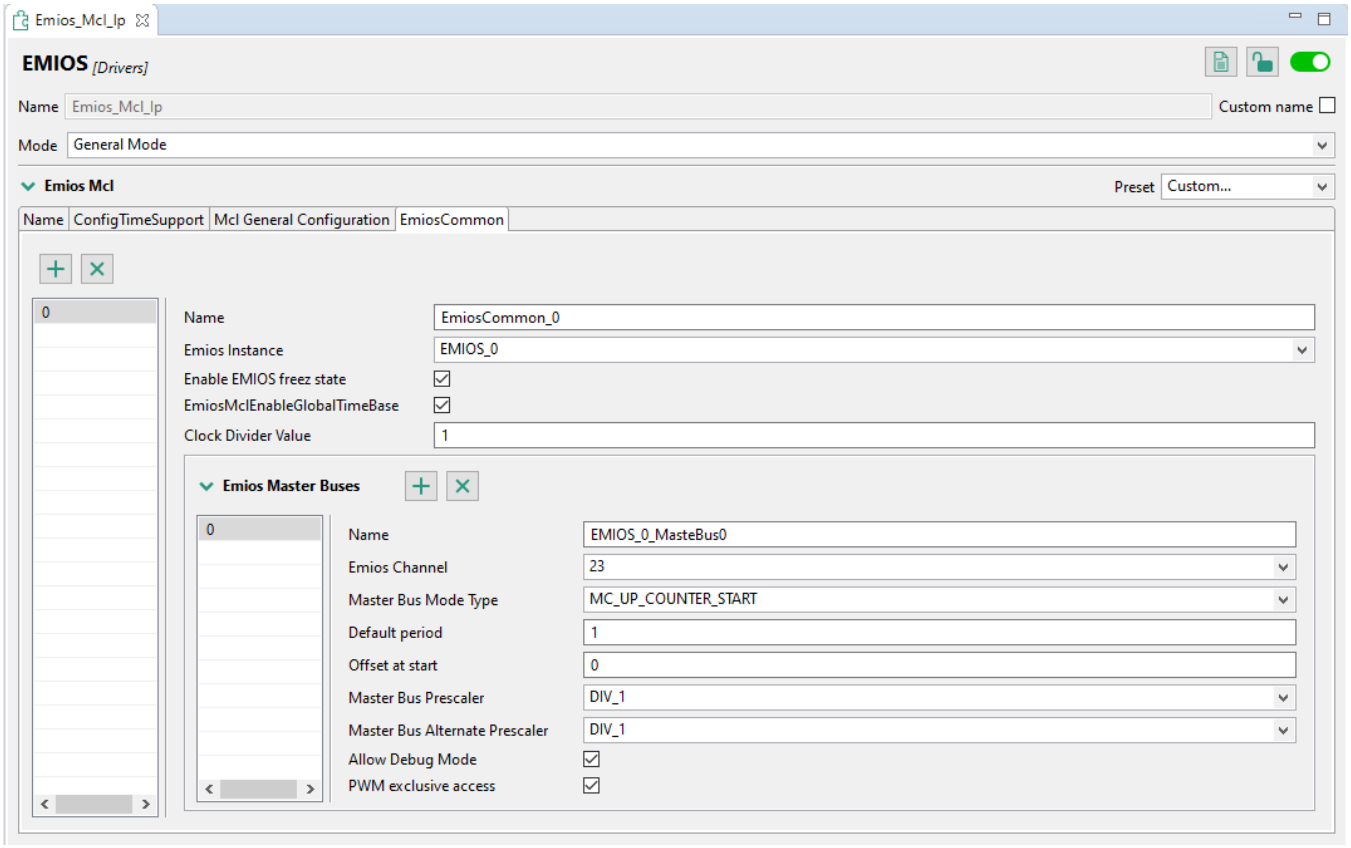

Figure 3.9 Configure instance.

b) Flexio IP:

Step 1: Enable Flexio common support

-If users use EB tresos

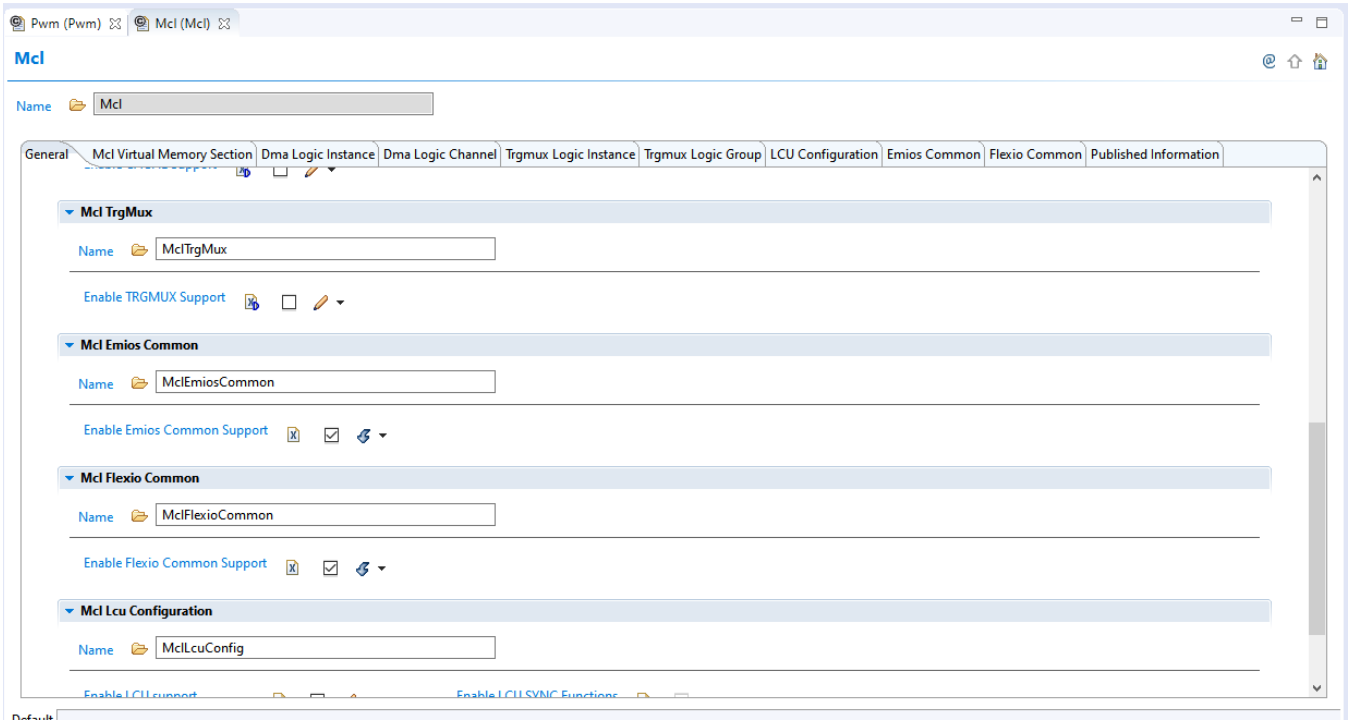

Figure 3.10 Enable Flexio common support

-If users use S32DS with Pwm driver

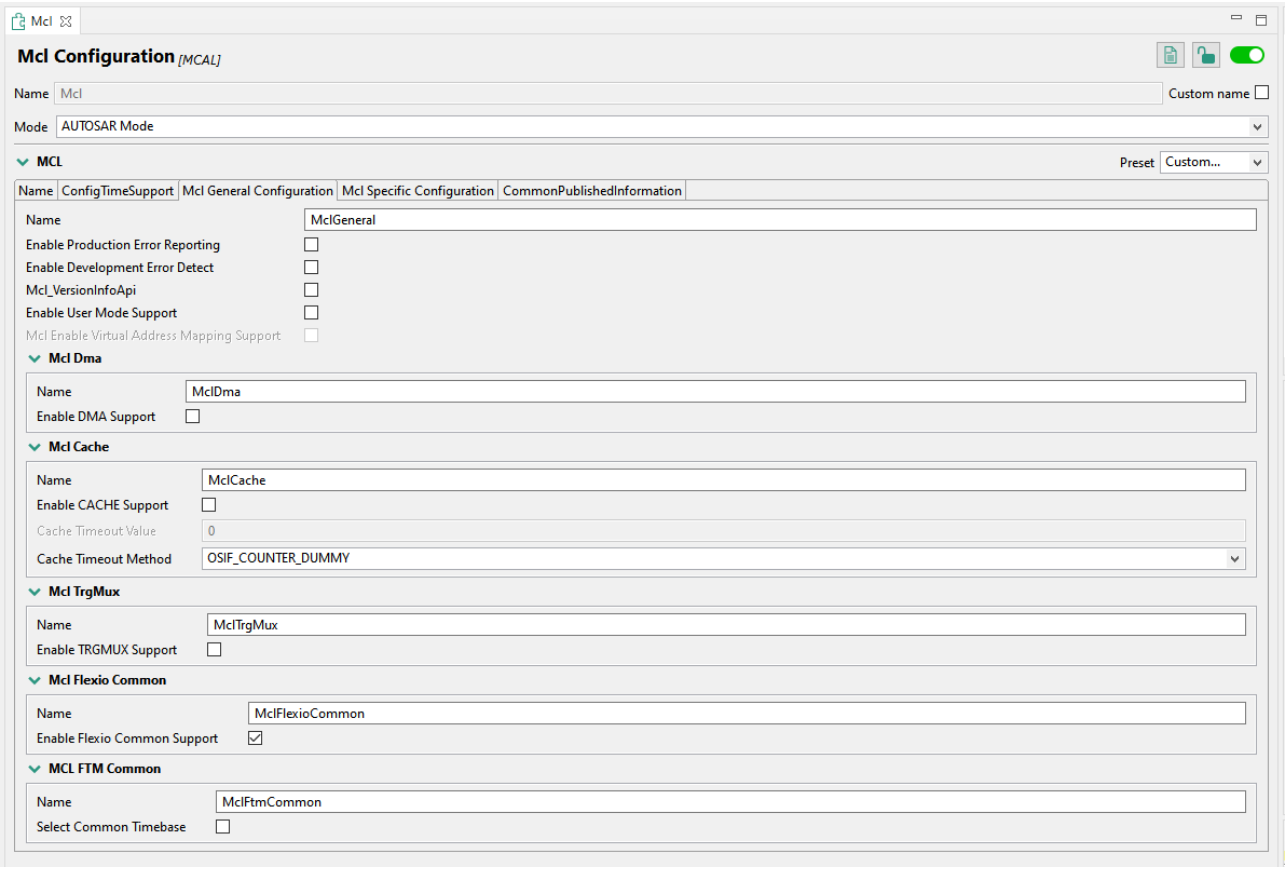

Figure 3.11 Enable Flexio common support

-If users use S32DS with Flexio_Pwm driver

Figure 3.12 Enable Flexio common support.

Step 2: Configure instance when Flexio Common Support is checked

-If users use EB tresos

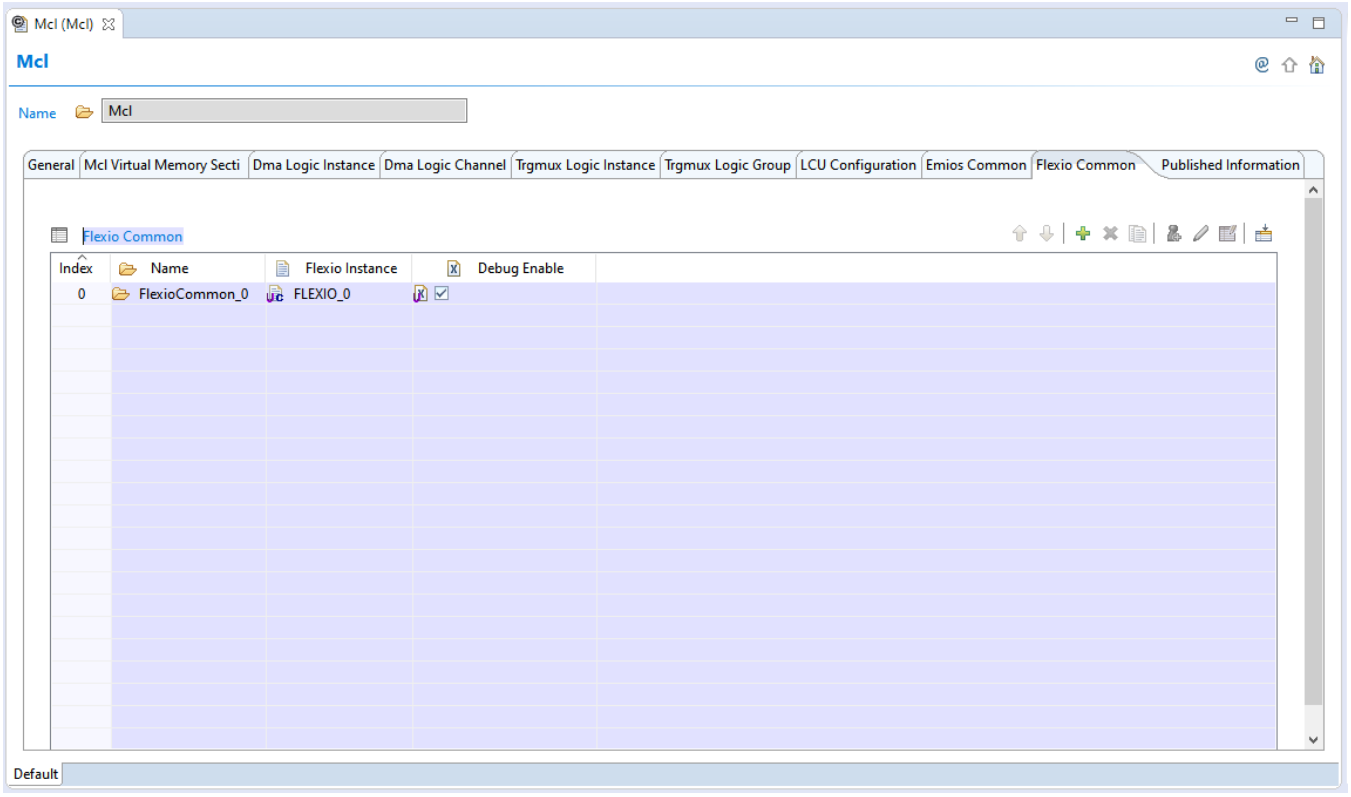

Figure 3.13 Configure instance.

-If users use S32DS with Pwm driver

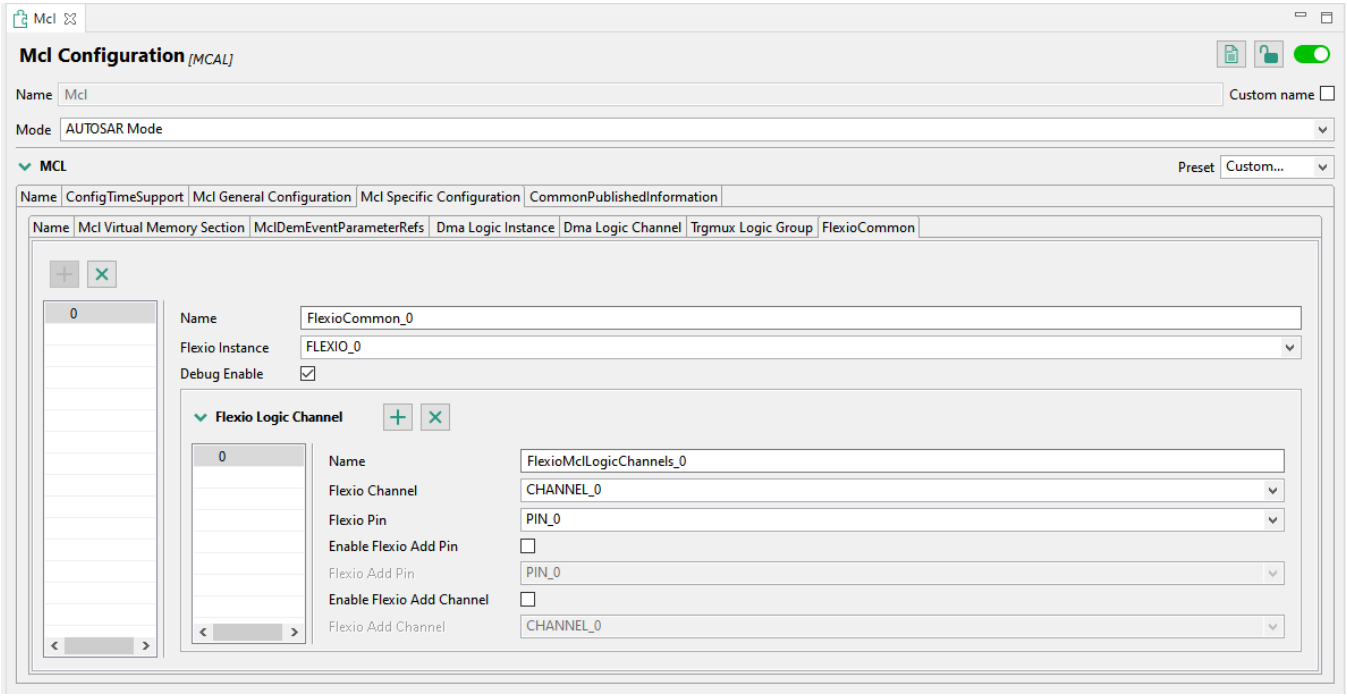

Figure 3.14 Configure instance.

-If users use S32DS with Flexio_Pwm driver

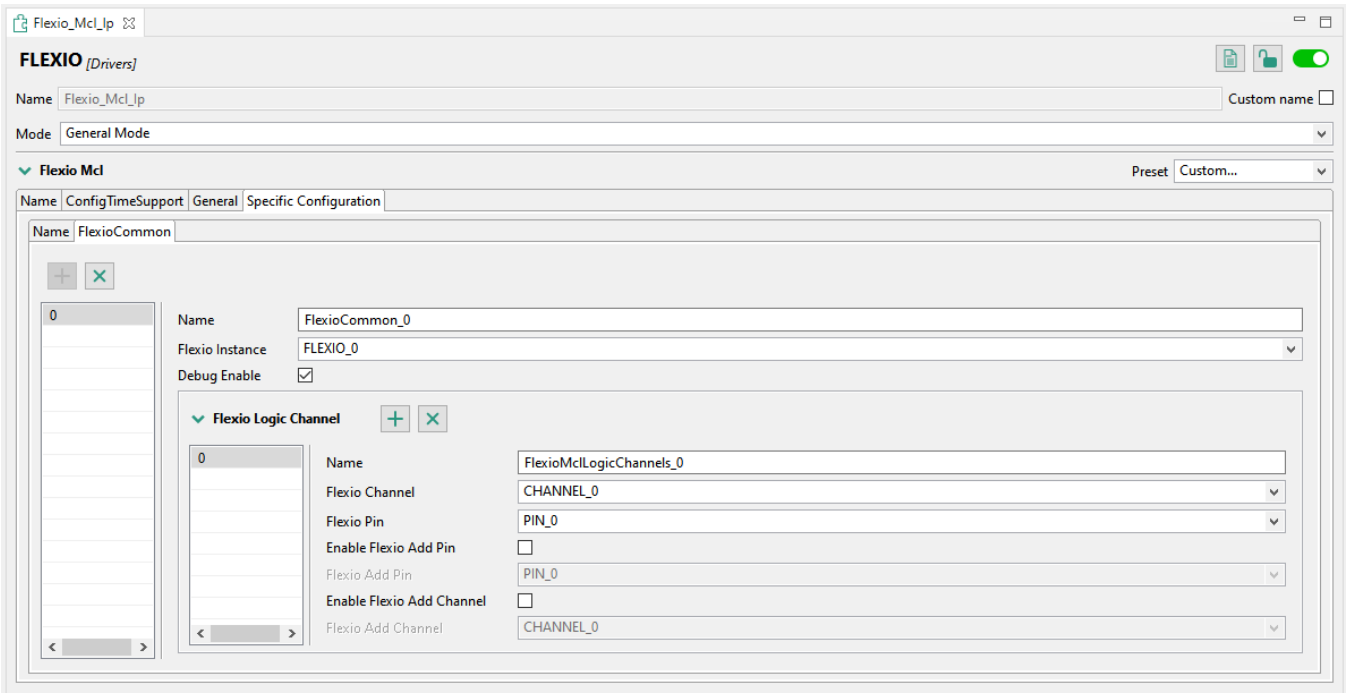

Figure 3.15 Configure instance.

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 Pwm
  - Container PwmChannelConfigSet
    - Container PwmChannel
      - Parameter PwmChannelId
      - Parameter PwmChannelClass
      - Parameter PwmPeriodInTicks
      - Parameter PwmPeriodDefault
      - Parameter PwmDutycycleDefault
      - Parameter PwmPolarity
      - Parameter PwmIdleState
      - Parameter PwmNotification
      - Reference PwmChannelEcucPartitionRef
      - Reference PwmHwChannel
      - Reference PwmMcuClockReferencePoint
    - Container PwmEmios
      - Parameter PwmHwInstance
      - Container PwmEmiosChannels
      - Parameter EmiosChId
      - Parameter EmiosChMode
      - Parameter EmiosChFlagGeneration
      - Parameter EmiosChCounterBus
      - Parameter EmiosChFreeze
      - Parameter EmiosChOutputDisable
      - Parameter EmiosChPrescaler
      - Parameter EmiosChPrescalerAlternate
      - Parameter EmiosChPrescalerSource
      - Parameter EmiosChPolarity
      - Parameter EmiosChInterrupt
      - Parameter EmiosChDutyCycle
      - Parameter EmiosChPeriod

- Parameter EmiosChPhaseShift
- Parameter EmiosChTrigger
- Parameter EmiosChDeadtime
- Reference PwmEmiosBusRef
- Container EmiosChIrqCallback
- Parameter EmiosChIrqFunctionCallback
- Parameter EmiosChIrqParameterCallback

* Container PwmFlexio
  - Parameter PwmHwInstance
  - Container PwmFlexioChannels
  - Parameter FlexioChId
  - Parameter FlexioPinId
  - Parameter FlexioChPrescaler
  - Parameter FlexioChPrescalerAlternate
  - Parameter FlexioChDutyCycle
  - Parameter FlexioChPeriod
  - Parameter FlexioChPolarity
  - Parameter FlexioChInterrupt
  - Reference FlexioMclChRef
  - Container FlexioChIrqCallback
  - Parameter FlexioChIrqFunctionCallback
  - Parameter FlexioChIrqParameterCallback

* Container FlexPwm
  - Parameter FlexPwmModule
  - Parameter FlexPwmPeriod
  - Parameter FlexPwmFaultFunctionality
  - Container FlexPwmFaultFilterSettings
  - Parameter FlexPwmFaultGlitchStretchEnable
  - Parameter FlexPwmFaultFilterCounter
  - Parameter FlexPwmFaultFilterPeriod
  - Parameter FlexPwmFaultCombinationalPath
  - Container FlexPwmFaultChannelSettings
  - Parameter FlexPwmFaultLevel
  - Parameter FlexPwmAutomaticFaultClearing
  - Parameter FlexPwmFaultSafetyMode
  - Parameter FlexPwmFullCycle
  - Parameter FlexPwmFaultInterruptEn
  - Parameter FlexPwmFaultNotification
  - Container FlexPwmSubModules
  - Parameter FlexPwmSubModule
  - Parameter FlexPwmCapabilities
  - Parameter FlexPwmClockSel
  - Parameter FlexPwmInitControlSrc
  - Parameter ReloadSrcSelect
  - Parameter ForceOutSelect

      - Parameter FlexPwmPrescaler
      - Parameter FlexPwmPrescaler_Alternate
      - Parameter FullCycleReload
      - Parameter HalfCycleReload
      - Parameter ReloadFrequency
      - Parameter FlexPwmInitVal
      - Parameter FlexPwmIndependent
      - Parameter FlexPwmPolarityPair
      - Parameter FlexPwmDeadTimeCount0
      - Parameter FlexPwmDeadTimeCount1
      - Parameter FlexPwmDebugEnable
      - Container FlexPwmChannels
      - Parameter FlexPwmChannel
      - Parameter FlexPwmChPolarity
      - Parameter FlexPwmChDutyCycle
      - Parameter FlexPwmPhaseShiftTicks
      - Parameter FlexPwm_CTU_Trigger
      - Parameter FlexPwmFaultOutputState
      - Parameter FlexPwmChInterrupt
      - Container FlexPwmChIrqCallback
      - Parameter FlexPwmChIrqFunctionCallback
      - Parameter FlexPwmChIrqParameterCallback
      - Container FlexPwmChannelFaultSettings
      - Parameter FlexPwmDisableOutputOnFault0
      - Parameter FlexPwmDisableOutputOnFault1
      - Parameter FlexPwmDisableOutputOnFault2
      - Parameter FlexPwmDisableOutputOnFault3
  - Container PwmGeneral
    * Parameter PwmMulticoreEnabled
    * Parameter PwmDevErrorDetect
    * Parameter PwmDutycycleUpdatedEndperiod
    * Parameter PwmPeriodUpdatedEndperiod
    * Parameter PwmNotificationSupported
    * Parameter PwmEnableUserModeSupport
    * Parameter PwmLowPowerStatesSupport
    * Parameter PwmPowerStateAsynchTransitionMode
    * Parameter PwmEnableDualClockMode
    * Parameter PwmMultiChannelSync
    * Parameter PwmIndex
    * Reference PwmEcucPartitionRef
    * Reference PwmKernelEcucPartitionRef
    * Container PwmPowerStateConfig
      - Parameter PwmPowerState
      - Parameter PwmPowerStateReadyCbkRef

- Container PwmConfigurationOfOptApiServices
  - Parameter PwmDeInitApi
  - Parameter PwmGetOutputState
  - Parameter PwmSetDutyCycle
  - Parameter PwmSetOutputToIdle
  - Parameter PwmSetPeriodAndDuty
  - Parameter PwmVersionInfoApi
  - Parameter PwmGetChannelStateApi
  - Parameter PwmSetDutyCycle_NoUpdate
  - Parameter PwmSetPeriodAndDuty_NoUpdate
  - Parameter PwmSetPhaseShift
  - Parameter PwmSetPhaseShift_NoUpdate
  - Parameter PwmSetDutyPhaseShift
  - Parameter PwmSetChannelDeadTime
  - Parameter PwmSetCounterBusApi
  - Parameter PwmSetChannelOutputApi
  - Parameter PwmSetTriggerDelayApi
  - Parameter PwmEmiosFastUpdateApi
- Container CommonPublishedInformation
  - Parameter ArReleaseMajorVersion
  - Parameter ArReleaseMinorVersion
  - Parameter ArReleaseRevisionVersion
  - Parameter ModuleId
  - Parameter SwMajorVersion
  - Parameter SwMinorVersion
  - Parameter SwPatchVersion
  - Parameter VendorApiInfix
  - Parameter VendorId

4.1 Module Pwm

Configuration of Pwm (Pulse Width Modulation) module.

Included containers:

- PwmChannelConfigSet
- PwmGeneral
- PwmConfigurationOfOptApiServices
- CommonPublishedInformation

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

4.2 Container PwmChannelConfigSet

Container contains the channel configuration parameter of the Pwm driver.

Included subcontainers:

- PwmChannel
- PwmEmios
- PwmFlexio
- FlexPwm

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

4.3 Container PwmChannel

Configuration of an individual Pwm 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

4.4 Parameter PwmChannelId

Channel ID of the Pwm channel. This value will be assigned to the symbolic name derived of the PwmChannel container short name.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	true
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	0
max	4294967295
min	0

4.5 Parameter PwmChannelClass

Class of Pwm Channel.

PWM_FIXED_PERIOD - Period of the channel will not be changed.

PWM_FIXED_PERIOD_SHIFTED - Period of the channel will not be changed, and support with phase shift feature.

PWM_VARIABLE_PERIOD - Period of the channel can be changed.

<note>

Due to Ftm hardware specific feature that the counter register and period register are shared common for all

channels in one Ftm instance, therefore if current channel is configured PWM_FIXED_PERIOD class, but another

channel in the same Ftm instance is configured PWM_VARIABLE_PERIOD, then when that channel call function to

change period, current channel will change period, too.

</note>

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF

Property	Value
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	true
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	PWM_FIXED_PERIOD
literals	['PWM_FIXED_PERIOD', 'PWM_FIXED_PERIOD_SHIFTED', 'PWM_↩ VARIABLE_PERIOD']

4.6 Parameter PwmPeriodInTicks

Check this option to configure Default Period unit in ticks, or uncheck this to configure Default Period unit in seconds.

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

4.7 Parameter PwmPeriodDefault

Default period value of Pwm channel at initialization.

The measure unit are in ticks (if Period In Ticks checked), or in seconds (if unchecked)

Valid range: [0, 0xFFFFFE]

<note>

The maximum period tick is 0xFFFFFE (instead of maximum value of period register 0xFFFFFF) in order to achieve

perfect 0 or 100% duty cycle.

All channels which are in the same FlexPwm sub-module with current channel must have the same Default Period

value, due to FlexPwm sub-module hardware specific feature that the period register is shared common for all channels

in one FlexPwm sub-module .

</note>

Property	Value
type	ECUC-FLOAT-PARAM-DEF
origin	AUTOSAR_ECUC
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	1.0
max	1.6777214E7
min	0.0

4.8 Parameter PwmDutycycleDefault

Default value for duty cycle of Pwm channel at initialization.

0 represents for 0% duty cycle

16384 (0x4000) represents for 50% duty cycle

32768 (0x8000) represents for 100% duty cycle

Valid value: [0,32768]

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

Property	Value
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	16384
max	32768
min	0

4.9 Parameter PwmPolarity

Define the polarity of Pwm channel at initialization.

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

4.10 Parameter PwmIdleState

Define Pwm channel state when the output is set to idle.

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

Property	Value
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	PWM_LOW
literals	['PWM_HIGH', 'PWM_LOW']

4.11 Parameter PwmNotification

User callback notification function.

This option is only activated when PwmGeneral/PwmNotificationSupported is checked.

<note>

Use NULL_PTR without any quotes to determine no notification function is used.

If the string is different from above, it will be used as the notification function name.

Notification does not apply to channel that its alignment type is PWM_CENTER_ALIGNED.

</note>

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

4.12 Reference PwmChannelEcucPartitionRef

Maps a Pwm 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 Pwm driver is mapped to.

When users choose ENABLE multicore feature by checking PwmMulticoreEnabled option, this will force to configure at least 1 ECUC partition in this list that is referenced from ECUC module; OR when DISABLE multicore feature, user have to remove all ECUC partitions in this list.

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

4.13 Reference PwmHwChannel

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

Property	Value
type	ECUC-CHOICE-REFERENCE-DEF
origin	NXP
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
requiresSymbolicNameValue	False
destinations	['/TS_T40D34M30I0R0/Pwm/PwmChannelConfigSet/PwmEmios/Pwm↩ EmiosChannels', '/TS_T40D34M30I0R0/Pwm/PwmChannelConfigSet/Pwm↩ Flexio/PwmFlexioChannels', '/TS_T40D34M30I0R0/Pwm/PwmChannel↩ ConfigSet/FlexPwm/FlexPwmSubModules/FlexPwmChannels']

4.14 Reference PwmMcuClockReferencePoint

Reference to the clock source configuration, which is set in the MCU driver configuration.

Property	Value
type	ECUC-REFERENCE-DEF
origin	AUTOSAR_ECUC
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
requiresSymbolicNameValue	False
destination	/AUTOSAR/EcucDefs/Mcu/McuModuleConfiguration/McuClockSetting↩ Config/McuClockReferencePoint

4.15 Container PwmEmios

Configuration of an Emios module available on the platform.

Included subcontainers:

- PwmEmiosChannels

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

4.16 Parameter PwmHwInstance

Select the hardware Emios module.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF

Property	Value
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	Emios_0
literals	['Emios_0', 'Emios_1', 'Emios_2']

4.17 Container PwmEmiosChannels

List of Emios channels available on the platform.

Included subcontainers:

- EmiosChIrqCallback

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

4.18 Parameter EmiosChId

Select one of the Emios channels available on the platform.

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

Property	Value
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	CH_0
literals	['CH_0', 'CH_1', 'CH_2', 'CH_3', 'CH_4', 'CH_5', 'CH_6', 'CH_7', 'CH_↩ 8', 'CH_9', 'CH_10', 'CH_11', 'CH_12', 'CH_13', 'CH_14', 'CH_15', 'CH_16', 'CH_17', 'CH_18', 'CH_19', 'CH_20', 'CH_21', 'CH_22', 'CH_23']

4.19 Parameter EmiosChMode

Select the mode for the Emios channel:

OPWFMB: Variable Period and Variable Duty Cycle. Bus: Internal only

OPWMB: Fixed Period, Variable Duty Cycle Bus: External bus only

OPWMCB: Variable Period and Variable Duty Cycle with dead time insertion Bus: External bus only

OPWMT: Fixed Period, Variable Duty Cycle with trigger generation Bus: External bus only

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	EMIOS_PWM_IP_MODE_OPWFMB
literals	['EMIOS_PWM_IP_MODE_OPWFMB', 'EMIOS_PWM_IP_MODE_OP↩ WMCB_TRAIL_EDGE', 'EMIOS_PWM_IP_MODE_OPWMCB_LEAD_↩ EDGE', 'EMIOS_PWM_IP_MODE_OPWMB', 'EMIOS_PWM_IP_MOD↩ E_OPWMT', 'EMIOS_PWM_IP_MODE_DAOC']

4.20 Parameter EmiosChFlagGeneration

Select when the flag will be set.

FLAG: FLAG will be set on trailing edge.

FLAG_BOTH: FLAG will be set on leading edge and trailing edge.

For OPWMT mode Flag will only be set when a trigger is generated.

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	Trailing_Edge
literals	['Trailing_Edge', 'Both_Trailing_and_Leading_Edge']

4.21 Parameter EmiosChCounterBus

Select the counter bus used by the channel.

EMIOS_PWM_IP_BUS_A - counter bus A, available for all channels

EMIOS_PWM_IP_BUS_F - counter bus F, available for all channels

EMIOS_PWM_IP_BUS_BCDE - counter bus B, C, D or E specific to each channel

Bus B is controlled by channel 0 and is available for channels 0->7

Bus C is controlled by channel 8 and is available for channels 8->15

Bus D is controlled by channel 16 and is available for channels 16->23

Bus E is controlled by channel 24 and is available for channels 24->31

EMIOS_PWM_IP_BUS_INTERNAL - unified channel internal counter

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

Property	Value
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	EMIOS_PWM_IP_BUS_INTERNAL
literals	['EMIOS_PWM_IP_BUS_A', 'EMIOS_PWM_IP_BUS_BCDE', 'EMIOS_↩ PWM_IP_BUS_F', 'EMIOS_PWM_IP_BUS_INTERNAL']

4.22 Parameter EmiosChFreeze

This parameter controls Freeze Enable bit (FREN), if set and validated by FRZ bit in EMIOS_MCR register, freezes all registers value of period used for initialization.

True - Freeze channel registers values.

False - Normal operation.

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

4.23 Parameter EmiosChOutputDisable

Select one of the four output disable input signals.

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

Property	Value
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	EMIOS_PWM_IP_OUTPUT_DISABLE_NONE
literals	['EMIOS_PWM_IP_OUTPUT_DISABLE_NONE', 'EMIOS_PWM_IP_O↩ UTPUT_DISABLE_0', 'EMIOS_PWM_IP_OUTPUT_DISABLE_1', 'EMI↩ OS_PWM_IP_OUTPUT_DISABLE_2', 'EMIOS_PWM_IP_OUTPUT_D↩ ISABLE_3']

4.24 Parameter EmiosChPrescaler

Select clock prescaler used for this Emios channel. The internal counter must be selected for this setting to take any effect.

Will only have an effect in OPWFMB and OPWMCB modes.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	EMIOS_PWM_IP_CLOCK_DIV_2
literals	['EMIOS_PWM_IP_CLOCK_DIV_1', 'EMIOS_PWM_IP_CLOCK_DIV↩ _2', 'EMIOS_PWM_IP_CLOCK_DIV_3', 'EMIOS_PWM_IP_CLOCK_D↩ IV_4', 'EMIOS_PWM_IP_CLOCK_DIV_5', 'EMIOS_PWM_IP_CLOCK↩ _DIV_6', 'EMIOS_PWM_IP_CLOCK_DIV_7', 'EMIOS_PWM_IP_CLO↩ CK_DIV_8', 'EMIOS_PWM_IP_CLOCK_DIV_9', 'EMIOS_PWM_IP_C↩ LOCK_DIV_10', 'EMIOS_PWM_IP_CLOCK_DIV_11', 'EMIOS_PWM_↩ IP_CLOCK_DIV_12', 'EMIOS_PWM_IP_CLOCK_DIV_13', 'EMIOS_P↩ WM_IP_CLOCK_DIV_14', 'EMIOS_PWM_IP_CLOCK_DIV_15', 'EMI↩ OS_PWM_IP_CLOCK_DIV_16']

4.25 Parameter EmiosChPrescalerAlternate

Select alternative clock prescaler used for this Emios channel. The internal counter must be selected for this setting to take any effect.

Will only have an effect in OPWFMB and OPWMCB modes.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	EMIOS_PWM_IP_CLOCK_DIV_2
literals	['EMIOS_PWM_IP_CLOCK_DIV_1', 'EMIOS_PWM_IP_CLOCK_DIV↩ _2', 'EMIOS_PWM_IP_CLOCK_DIV_3', 'EMIOS_PWM_IP_CLOCK_D↩ IV_4', 'EMIOS_PWM_IP_CLOCK_DIV_5', 'EMIOS_PWM_IP_CLOCK↩ _DIV_6', 'EMIOS_PWM_IP_CLOCK_DIV_7', 'EMIOS_PWM_IP_CLO↩ CK_DIV_8', 'EMIOS_PWM_IP_CLOCK_DIV_9', 'EMIOS_PWM_IP_C↩ LOCK_DIV_10', 'EMIOS_PWM_IP_CLOCK_DIV_11', 'EMIOS_PWM_↩ IP_CLOCK_DIV_12', 'EMIOS_PWM_IP_CLOCK_DIV_13', 'EMIOS_P↩ WM_IP_CLOCK_DIV_14', 'EMIOS_PWM_IP_CLOCK_DIV_15', 'EMI↩ OS_PWM_IP_CLOCK_DIV_16']

4.26 Parameter EmiosChPrescalerSource

Select clock source for the internal prescaler used for this Emios channel.

Will only have an effect in OPWFMB and OPWMCB modes.

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

Property	Value
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	EMIOS_PWM_IP_PS_SRC_MODULE_CLOCK
literals	['EMIOS_PWM_IP_PS_SRC_PRESCALED_CLOCK', 'EMIOS_PWM_I↩ P_PS_SRC_MODULE_CLOCK']

4.27 Parameter EmiosChPolarity

Define the output polarity of the 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-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	EMIOS_PWM_IP_ACTIVE_HIGH
literals	['EMIOS_PWM_IP_ACTIVE_HIGH', 'EMIOS_PWM_IP_ACTIVE_LOW']

4.28 Parameter EmiosChInterrupt

Define what happens when a flag event is generated.

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

Property	Value
defaultValue	EMIOS_PWM_IP_NOTIFICATION_DISABLED
literals	['EMIOS_PWM_IP_NOTIFICATION_DISABLED', 'EMIOS_PWM_IP_I↩ NTERRUPT_REQUEST', 'EMIOS_PWM_IP_DMA_REQUEST']

4.29 Parameter EmiosChDutyCycle

Value for duty cycle used for initialization.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	16777214
min	0

4.30 Parameter EmiosChPeriod

Period value used at initialization.

NOTE: Only used in OPWFMB mode.

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

Property	Value
defaultValue	32768
max	16777214
min	1

4.31 Parameter EmiosChPhaseShift

Phase Shift (in ticks) of the PWM output.

Please note that the counter will always start from 1, and the Phase Shift value is incremented by 1 in the code.

The Phase Shift value must be less than the period of the channel used as reference or else a DET error will be generated.

NOTE: The Phase Shift parameter can only be used in OPWMB or OPWMT mode.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	16777214
min	0

4.32 Parameter EmiosChTrigger

Select input count source (clock) used for this Emios channelDelay (in ticks) for generating the trigger event.

Please note that the counter will always start from 1, and the Trigger value is incremented by 1 in the code, then the value is updated into Alternate A register.

The programmed value in the A2 register must be less or equal to the value of the channel period used as reference or else the CTU triggers will not be generated.

NOTE: This parameter is used only in OPWMT mode.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	16777214
min	0

4.33 Parameter EmiosChDeadtime

Deadtime parameter controls the deadtime during transitions of the PWM output.

NOTE: This parameter is used only for OPWMCB mode.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	16777214
min	0

4.34 Reference PwmEmiosBusRef

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

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

4.35 Container EmiosChIrqCallback

Configure Notfication function and parameter for interrupt handler callback.

Included subcontainers:

- None

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

4.36 Parameter EmiosChIrqFunctionCallback

User callback function.

NOTE: Use NULL_PTR without any quotes. If the used string is different from NULL_PTR it will be used as the configured function name.

Property	Value
type	ECUC-FUNCTION-NAME-DEF
origin	NXP
symbolicNameValue	false

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

4.37 Parameter EmiosChIrqParameterCallback

User callback parameter.

NOTE: Use NULL_PTR without any quotes. If the used string is different from NULL_PTR it will be used as the configured parameter name.

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

4.38 Container PwmFlexio

Configuration of an Flexio module available on the platform.

Included subcontainers:

- PwmFlexioChannels

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

Property	Value
lowerMultiplicity	0
upperMultiplicity	Infinite
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD

4.39 Parameter PwmHwInstance

Select the hardware Flexio module.

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	Flexio_0
literals	['Flexio_0']

4.40 Container PwmFlexioChannels

List of Flexio channels available on the platform.

Included subcontainers:

- FlexioChIrqCallback

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

4.41 Parameter FlexioChId

Select one of the Flexio channels available on the platform.

NOTE: This also selects the used timer.

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	CH_0
literals	['CH_0', 'CH_1', 'CH_2', 'CH_3', 'CH_4', 'CH_5', 'CH_6', 'CH_7']

4.42 Parameter FlexioPinId

Select one of the Flexio Pins available on the platform.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	PIN_0
literals	['PIN_0', 'PIN_1', 'PIN_2', 'PIN_3', 'PIN_4', 'PIN_5', 'PIN_6', 'PIN_7', 'P↩IN_8', 'PIN_9', 'PIN_10', 'PIN_11', 'PIN_12', 'PIN_13', 'PIN_14', 'PIN_15', 'PIN_16', 'PIN_17', 'PIN_18', 'PIN_19', 'PIN_20', 'PIN_21', 'PIN_22', 'PIN↩_23', 'PIN_24', 'PIN_25', 'PIN_26', 'PIN_27', 'PIN_28', 'PIN_29', 'PIN_30', 'PIN_31']

4.43 Parameter FlexioChPrescaler

Select clock prescaler used for this Flexio 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-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	FLEXIO_PWM_IP_CLK_DIV_1
literals	['FLEXIO_PWM_IP_CLK_DIV_1', 'FLEXIO_PWM_IP_CLK_DIV_16', 'FLEXIO_PWM_IP_CLK_DIV_256']

4.44 Parameter FlexioChPrescalerAlternate

Select alternative clock prescaler used for this Flexio 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-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	FLEXIO_PWM_IP_CLK_DIV_1
literals	['FLEXIO_PWM_IP_CLK_DIV_1', 'FLEXIO_PWM_IP_CLK_DIV_16', 'FLEXIO_PWM_IP_CLK_DIV_256']

4.45 Parameter FlexioChDutyCycle

Value for duty cycle used for initialization.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	256
min	0

4.46 Parameter FlexioChPeriod

Period value used at initialization.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	512
min	0

4.47 Parameter FlexioChPolarity

Define the output polarity of the channel.

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

Property	Value
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	FLEXIO_PWM_IP_ACTIVE_HIGH
literals	['FLEXIO_PWM_IP_ACTIVE_HIGH', 'FLEXIO_PWM_IP_ACTIVE_LOW']

4.48 Parameter FlexioChInterrupt

Define what happens when a flag event is generated.

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	FLEXIO_PWM_IP_IRQ_DISABLED
literals	['FLEXIO_PWM_IP_IRQ_DISABLED', 'FLEXIO_PWM_IP_IRQ_ON_↩ RISING_EDGE', 'FLEXIO_PWM_IP_IRQ_ON_FALLING_EDGE', 'FLE↩ XIO_PWM_IP_IRQ_ON_BOTH_EDGES', 'FLEXIO_PWM_IP_IRQ_O↩ N_PERIOD_END']

4.49 Reference FlexioMclChRef

Select the Flexio MCL hw channel on which the the current PWM channel will be implemented.

Property	Value
type	ECUC-REFERENCE-DEF
origin	NXP

Property	Value
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
requiresSymbolicNameValue	False
destination	/TS_T40D34M30I0R0/Mcl/MclConfig/FlexioCommon/FlexioMclLogicChannels

4.50 Container FlexioChIrqCallback

Configure Notification function and parameter for interrupt handler callback.

Included subcontainers:

- None

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

4.51 Parameter FlexioChIrqFunctionCallback

User callback function.

NOTE: Use NULL_PTR without any quotes. If the used string is different from NULL_PTR it will be used as the configured function name.

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

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

4.52 Parameter FlexioChIrqParameterCallback

User callback parameter.

NOTE: Use NULL_PTR without any quotes. If the used string is different from NULL_PTR it will be used as the configured parameter name.

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

4.53 Container FlexPwm

Configuration of a FlexPWM module available on the platform.

Included subcontainers:

- FlexPwmFaultFilterSettings
- FlexPwmFaultChannelSettings
- FlexPwmSubModules

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	0

Property	Value
upperMultiplicity	Infinite
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE

4.54 Parameter FlexPwmModule

Select one of the FlexPWM modules available on the platform.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	FlexPwm_1
literals	['FlexPwm_1', 'FlexPwm_2']

4.55 Parameter FlexPwmPeriod

Period value used at initialization.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	65534
min	0

4.56 Parameter FlexPwmFaultFunctionality

Fault functionality is enabled for this FlexPWM module.

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

4.57 Container FlexPwmFaultFilterSettings

Container used for storing configuration for PWM Fault mechanism.

Included subcontainers:

- None

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

4.58 Parameter FlexPwmFaultGlitchStretchEnable

This parameter is used to enable the fault glitch stretching logic.

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

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

4.59 Parameter FlexPwmFaultFilterCounter

Specify the number of consecutive samples that must agree prior to the input filter accepting an input transition.

Value range: [0,7]

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	7
min	0

4.60 Parameter FlexPwmFaultFilterPeriod

Specify the sampling period (in IPBus clock cycles) of the fault pin input filter.

Value range: [0,255]

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

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

4.61 Parameter FlexPwmFaultCombinationalPath

This field is used to control the combinational path from the fault inputs to the PWM outputs

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

4.62 Container FlexPwmFaultChannelSettings

List of FlexPwm fault channels in a FlexPwm Module.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	4
upperMultiplicity	4
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.63 Parameter FlexPwmFaultLevel

Specify the logic level for Fault detection.

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	LOW
literals	['LOW', 'HIGH']

4.64 Parameter FlexPwmAutomaticFaultClearing

Enable the automatic fault clearing.

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

4.65 Parameter FlexPwmFaultSafetyMode

Enable the safe mode of operation, only for Manual fault clearing.

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

4.66 Parameter FlexPwmFullCycle

PWM output is re-enabled only at the start of a full cycle.

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

4.67 Parameter FlexPwmFaultInterruptEn

Enables/Disables the interrupt for each fault channels.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF

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

4.68 Parameter FlexPwmFaultNotification

User callback function on fault detection.

NOTE: Please use NULL or NULL_PTR without 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	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	NULL_PTR

4.69 Container FlexPwmSubModules

Select one of the FlexPWM submodules available on the current module.

Included subcontainers:

- FlexPwmChannels

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

4.70 Parameter FlexPwmSubModule

Select one of the 4 sub-modules available on the current FlexPWM module.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	SubModule_0
literals	['SubModule_0', 'SubModule_1', 'SubModule_2', 'SubModule_3']

4.71 Parameter FlexPwmCapabilities

This option will change the PWM generation mode for all channels in this FlexPWM submodule.

If the value is PWM_CENTER_ALIGNED or PWM_PHASE_SHIFTED, the complementary output and channel X will be disabled for this FlexPWM sub-module.

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

Property	Value
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	FLEXPWM_IP_EDGE_ALIGNED
literals	['FLEXPWM_IP_EDGE_ALIGNED', 'FLEXPWM_IP_PHASE_SHIFTED', 'FLEXPWM_IP_CENTER_ALIGNED']

4.72 Parameter FlexPwmClockSel

Select clock source for current FlexPWM submodule.

FLEXPWM_IP_CLKSOURCE_PERIPHERAL_CLK: Peripheral clock. For S32ZE it's system clock AE_CLK.

FLEXPWM_IP_CLKSOURCE_EXT_CLK: External clock. For S32ZE the external clock is from eTimer channel output.

FlexPwm_AE_1 external clock connected to eTimer_AE_1_1 channel 1.

FlexPwm_AE_2 external clock connected to eTimer_AE_2_1 channel 1.

FLEXPWM_IP_CLKSOURCE_AUX_CLK: Submodule 0's clock of its FlexPWM hardware. This setting should not be used in submodule 0 as it will force the clock to logic 0.

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	FLEXPWM_IP_CLKSOURCE_PERIPHERAL_CLK
literals	['FLEXPWM_IP_CLKSOURCE_PERIPHERAL_CLK', 'FLEXPWM_IP_↩ CLKSOURCE_EXT_CLK', 'FLEXPWM_IP_CLKSOURCE_AUX_CLK']

4.73 Parameter FlexPwmInitControlSrc

Initialization Control Select

FLEXPWM_IP_INIT_LOCAL_SYNC: Local sync (PWMX) causes initialization.

FLEXPWM_IP_INIT_MASTER_RELOAD: Master reload from sub-module 0 causes initialization.

FLEXPWM_IP_INIT_MASTER_SYNC: Master sync from sub-module 0 causes initialization.

FLEXPWM_IP_INIT_EXT_SYNC: EXT_SYNC causes initialization.

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	FLEXPWM_IP_INIT_LOCAL_SYNC
literals	['FLEXPWM_IP_INIT_LOCAL_SYNC', 'FLEXPWM_IP_INIT_MASTE↩ R_RELOAD', 'FLEXPWM_IP_INIT_MASTER_SYNC', 'FLEXPWM_IP_↩ INIT_EXT_SYNC']

4.74 Parameter ReloadSrcSelect

This field determines the source of the RELOAD signal for this submodule.

FLEXPWM_IP_LOCAL_RELOAD: The local RELOAD signal is used to reload registers.

FLEXPWM_IP_MASTER_RELOAD: The master RELOAD signal (from submodule 0) is used to reload registers.

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	FLEXPWM_IP_LOCAL_RELOAD
literals	['FLEXPWM_IP_LOCAL_RELOAD', 'FLEXPWM_IP_MASTER_RELOAD']

4.75 Parameter ForceOutSelect

This field determines the source of the FORCE OUTPUT signal for this submodule.

FLEXPWM_IP_LOCAL_FORCE: The local force signal, CTRL2[FORCE], from this sub-module is used to force updates.

FLEXPWM_IP_MASTER_FORCE: The master force signal from sub-module 0 is used to force updates.

FLEXPWM_IP_LOCAL_RELOAD_FORCE: The local reload signal from this sub-module is used to force updates without regard to the state of LDOK.

FLEXPWM_IP_MASTER_RELOAD_FORCE: The master reload signal from sub-module 0 is used to force updates if LDOK is set.

FLEXPWM_IP_LOCAL_SYNC: The local sync signal from this sub-module is used to force updates.

FLEXPWM_IP_MASTER_SYNC: The master sync signal from submodule0 is used to force updates.

FLEXPWM_IP_EXT_FORCE: The external force signal, EXT_FORCE, from outside the PWM module causes updates.

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	FLEXPWM_IP_LOCAL_FORCE
literals	['FLEXPWM_IP_LOCAL_FORCE', 'FLEXPWM_IP_MASTER_FORCE', 'FLEXPWM_IP_LOCAL_RELOAD_FORCE', 'FLEXPWM_IP_MASTER↩ _RELOAD_FORCE', 'FLEXPWM_IP_LOCAL_SYNC', 'FLEXPWM_IP_↩ MASTER_SYNC', 'FLEXPWM_IP_EXT_FORCE']

4.76 Parameter FlexPwmPrescaler

Select primary prescaler for clock source.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF

Property	Value
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	FLEXPWM_IP_DIV1
literals	['FLEXPWM_IP_DIV1', 'FLEXPWM_IP_DIV2', 'FLEXPWM_IP_DIV4', 'FLEXPWM_IP_DIV8', 'FLEXPWM_IP_DIV16', 'FLEXPWM_IP_DIV32', 'FLEXPWM_IP_DIV64', 'FLEXPWM_IP_DIV128']

4.77 Parameter FlexPwmPrescaler_Alternate

Select alternate prescaler for clock source. This option will be used when PwmEnableDualClockMode is selected.

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	FLEXPWM_IP_DIV1
literals	['FLEXPWM_IP_DIV1', 'FLEXPWM_IP_DIV2', 'FLEXPWM_IP_DIV4', 'FLEXPWM_IP_DIV8', 'FLEXPWM_IP_DIV16', 'FLEXPWM_IP_DIV32', 'FLEXPWM_IP_DIV64', 'FLEXPWM_IP_DIV128']

4.78 Parameter FullCycleReload

Enable this check, the PWM reload opportunity is generated at the end of every cycle.

By default full cycle reload is enabled.

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

4.79 Parameter HalfCycleReload

Enable this check, the pwm reload opportunity is generated at the half cycle.

By default half cycle reload is disabled.

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

4.80 Parameter ReloadFrequency

Half Cycle Reload or Full Cycle Reload.

NOTE: This option is available only for center-aligned mode.

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

Property	Value
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	FLEXPWM_IP_LDFQ_EACH1
literals	['FLEXPWM_IP_LDFQ_EACH1', 'FLEXPWM_IP_LDFQ_EACH2', 'FLE↩XPWM_IP_LDFQ_EACH3', 'FLEXPWM_IP_LDFQ_EACH4', 'FLEXPW↩M_IP_LDFQ_EACH5', 'FLEXPWM_IP_LDFQ_EACH6', 'FLEXPWM_I↩P_LDFQ_EACH7', 'FLEXPWM_IP_LDFQ_EACH8', 'FLEXPWM_IP_L↩DFQ_EACH9', 'FLEXPWM_IP_LDFQ_EACH10', 'FLEXPWM_IP_LDF↩Q_EACH11', 'FLEXPWM_IP_LDFQ_EACH12', 'FLEXPWM_IP_LDFQ_↩EACH13', 'FLEXPWM_IP_LDFQ_EACH14', 'FLEXPWM_IP_LDFQ_EA↩CH15', 'FLEXPWM_IP_LDFQ_EACH16']

4.81 Parameter FlexPwmInitVal

This property is used to shift simultaneously in time the rising edges of the PWM channels of 2 or more FlexPWM submodules.

Eg: If set to 0 in 2 FlexPWM submodules configured in edge-aligned mode, the rising edges of all the channels

of those submodules will take place at the same point in time. If for one of the 2 FlexPWM submodules configured

in edge-aligned mode, this offset is not 0, the rising edges of the channels of that FlexPWM submodule will take place

Offset ticks in time after the rising edges of the channels of the FlexPwm SubModule having offset set to 0.

Value range: [0,0xFFFFFE]

The property can be used only in edge-aligned and phase-shifted modes.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	16777214

4.82 Parameter FlexPwmIndependent

Options available for the PWM A & B pair operation.

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

4.83 Parameter FlexPwmPolarityPair

Source selection for the generation of complementary PWM pair output

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	FLEXPWM_IP_COMP_SOURCE23
literals	['FLEXPWM_IP_COMP_SOURCE23', 'FLEXPWM_IP_COMP_SOURCE45']

4.84 Parameter FlexPwmDeadTimeCount0

11-bit value written to Deadtime Count 0 (DTCNT0) register of the FlexPWM submodule.

This register is set in terms of IPBus clock cycles regardless of the setting of PRSC and/or CLK_SEL.

The DTCNT0 field is used to control the deadtime during 0 to 1 transitions of the PWM_A output, assuming normal polarity.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	2047
min	0

4.85 Parameter FlexPwmDeadTimeCount1

11-bit value written to Deadtime Count 1 (DTCNT1) register of the FlexPWM submodule.

This register is set in terms of IPBus clock cycles regardless of the setting of PRSC and/or CLK_SEL.

The DTCNT1 field is used to control the deadtime during 0 to 1 transitions of the PWM_B output, assuming normal polarity.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	2047
min	0

4.86 Parameter FlexPwmDebugEnable

Debug Enable

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

4.87 Container FlexPwmChannels

Container used for storing configuration for FlexPWM channel.

Included subcontainers:

- FlexPwmChIrqCallback
- FlexPwmChannelFaultSettings

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	3
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE

4.88 Parameter FlexPwmChannel

Selects the FlexPWM channel

Valid values:

FLEXPWM_IP_PWMX - FlexPWM PWM_X channel

FLEXPWM_IP_PWMA - FlexPWM PWM_A channel

FLEXPWM_IP_PWMB - FlexPWM PWM_B channel

Complementary output channels may be also used. Complementary channels are based on channels A and B of the same submodule.

In order to use complementary channels, select Channel A and make sure the property 'Channel B Relation To Channel A'

of SubModule the channel belongs to is set to COMPLEMENTARY. When a submodule has the 'Channel B Relation To Channel A'

property set to COMPLEMENTARY, its channel B can no longer be selected in configuration but channel X is still available.

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

4.89 Parameter FlexPwmChPolarity

Define the output polarity of the channel.

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

Property	Value
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	FLEXPWM_IP_POL_HIGH
literals	['FLEXPWM_IP_POL_HIGH', 'FLEXPWM_IP_POL_LOW']

4.90 Parameter FlexPwmChDutyCycle

Value for duty cycle used for initialization.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	16777214
min	0

4.91 Parameter FlexPwmPhaseShiftTicks

Defines the Phase Shift of the current FlexPWM channel in ticks.

Property applies only to Pwm_A and Pwm_B channels and only when FlexPWM submodule is in phase-shifted mode.

Valid values [0,0xFFFFFE]

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

Property	Value
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	16777214
min	0

4.92 Parameter FlexPwm_CTU_Trigger

Define the CTU trigger configuration for FlexPWM channels.

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	FLEXPWM_IP_NO_TRIGGER
literals	['FLEXPWM_IP_NO_TRIGGER', 'FLEXPWM_IP_FIRST_EDGE', 'FLE↩ XPWM_IP_SECOND_EDGE', 'FLEXPWM_IP_BOTH_EDGES']

4.93 Parameter FlexPwmFaultOutputState

Specify the fault state for the PWM channel output during fault, stop and debug conditions.

FLEXPWM_IP_OUTPUT_STATE_LOGIC_0 - Output is forced to logic 0 state prior to consideration of output polarity control

FLEXPWM_IP_OUTPUT_STATE_LOGIC_1 - Output is forced to logic 1 state prior to consideration of output polarity control

FLEXPWM_IP_OUTPUT_STATE_TRISTATED - Output is tri-stated

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	FLEXPWM_IP_OUTPUT_STATE_LOGIC_0
literals	['FLEXPWM_IP_OUTPUT_STATE_LOGIC_0', 'FLEXPWM_IP_OUTP↩ UT_STATE_LOGIC_1', 'FLEXPWM_IP_OUTPUT_STATE_TRISTAT↩ ED']

4.94 Parameter FlexPwmChInterrupt

Define what happens when a flag event is generated.

FLEXPWM_IP_DISABLE_INT - Disable all Interrupt

FLEXPWM_IP_RELOAD_INT - Reload Interrupt

FLEXPWM_IP_COMPARE_INT - Compare Interrupt

Property	Value
type	ECUC-ENUMERATION-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	FLEXPWM_IP_DISABLE_INT
literals	['FLEXPWM_IP_DISABLE_INT', 'FLEXPWM_IP_COMPARE_INT', 'F↩ LEXPWM_IP_RELOAD_INT']

4.95 Container FlexPwmChIrqCallback

Configure Notfication function and parameter for interrupt handler callback.

Included subcontainers:

- None

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

4.96 Parameter FlexPwmChIrqFunctionCallback

User callback function.

NOTE: Use NULL_PTR without any quotes. If the used string is different from NULL_PTR it will be used as the configured function name.

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

4.97 Parameter FlexPwmChIrqParameterCallback

User callback parameter.

NOTE: Use NULL_PTR without any quotes. If the used string is different from NULL_PTR it will be used as the configured parameter name.

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

4.98 Container FlexPwmChannelFaultSettings

Container used for storing global fault configuration.

Included subcontainers:

- None

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

4.99 Parameter FlexPwmDisableOutputOnFault0

Disable the PWM output on detection of fault on fault channel 0.

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

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

4.100 Parameter FlexPwmDisableOutputOnFault1

Disable the PWM output on detection of fault on fault channel 1.

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

4.101 Parameter FlexPwmDisableOutputOnFault2

Disable the PWM output on detection of fault on fault channel 2.

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

4.102 Parameter FlexPwmDisableOutputOnFault3

Disable the PWM output on detection of fault on fault channel 3.

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

4.103 Container PwmGeneral

Container used for storing the general configuration of Pwm.

Included subcontainers:

- PwmPowerStateConfig

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

4.104 Parameter PwmMulticoreEnabled

Switch to enable/disable multicore feature.

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

PwmChannelEcucPartitionRef, and each Pwm channel in PwmChannel to configure at least 1 ECUC partition reference

in PwmChannelEcucPartitionRef container to fulfill generating code condition; OR uncheck this option to DISABLE

multicore feature, performing this action will force user to remove all ECUC partition reference in every Pwm channels contained

in PwmChannel and in PwmChannelEcucPartitionRef.

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

4.105 Parameter PwmDevErrorDetect

Switch to enable/disable the development error detection.

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

4.106 Parameter PwmDutycycleUpdatedEndperiod

Switch to enable the update of the duty cycle parameter at the end of the current period.

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

4.107 Parameter PwmPeriodUpdatedEndperiod

Switch to enable the update of the period parameter at the end of the current period.

<note>In current implementation, this option is locked and always enable by default.</note>

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

4.108 Parameter PwmNotificationSupported

Switch to indicate that the notifications are supported.

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

Property	Value
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

4.109 Parameter PwmEnableUserModeSupport

If enabled, the Pwm module will adapt to run from User Mode.

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

4.110 Parameter PwmLowPowerStatesSupport

If enabled, hardware offers low power mode and adds all power state management related APIs

(Pwm_SetPowerState, Pwm_GetCurrentPowerState, Pwm_GetTargetPowerState,

Pwm_PreparePowerState, Pwm_Main_PowerTransitionManager),

indicating if the hardware offers low power state management.

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

Property	Value
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	false

4.111 Parameter PwmPowerStateAsynchTransitionMode

Enable the support of the Pwm driver to the asynchronous power state transition.

<note> This feature is not supported and is rejected, as all hardware modules do not support asynchronous power state transitions.</note>

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

4.112 Parameter PwmEnableDualClockMode

Switch to enable/disable dual clock mode feature, which will add/remove the service Pwm_SetClockMode() from the code.

This feature is used when the prescaler value needs to be changed to maintain same period at different frequency.

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

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

4.113 Parameter PwmMultiChannelSync

Enable the update synchronous feature for several channels.

This option will activate the use of PwmSetDutyCycle_NoUpdate, PwmSetPeriodAndDuty_NoUpdate and PwmSetPhaseShift_NoUpdate or PwmSetDutyPhaseShift.

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

4.114 Parameter PwmIndex

Specify the InstanceId of this module instance. If only one instance is present it shall have the ID 0.

<note>In current implementation, this feature is not used.</note>

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

Property	Value
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	0
max	4294967295
min	0

4.115 Reference PwmEcucPartitionRef

Maps the Pwm 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.

When users choose ENABLE multicore feature by checking PwmMulticoreEnabled option, this will force to configure at least 1 ECUC partition in this list that is referenced from ECUC module; OR when DISABLE multicore feature, user have to remove all ECUC partitions in this list.

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

4.116 Reference PwmKernelEcucPartitionRef

Maps the Pwm 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 Pwm driver is mapped to.

<note>This feature is not supported and is rejected.</note>

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

4.117 Container PwmPowerStateConfig

Each instance of this parameter defines a power state and the callback to be called when this power state is reached.

<note>This feature is not supported and is rejected, as all hardware modules do not support asynchronous power state transitions.</note>

Included subcontainers:

- None

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

4.118 Parameter PwmPowerState

Each instance of this parameter describes a different power state supported by the Pwm hardware.

It should be defined by the hardware supplier and used by the Pwm Driver to reference specific HW configurations which set the Pwm HW module in the referenced power state.

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	0
max	9223372036854775807
min	0

4.119 Parameter PwmPowerStateReadyCbkRef

Each instance of this parameter contains a reference to a power mode callback defined in a CDD or IoHwAbs component.

Property	Value
type	ECUC-FUNCTION-NAME-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	NULL_PTR

4.120 Container PwmConfigurationOfOptApiServices

Container used for storing the configuration of all optional API's.

Included subcontainers:

- None

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

4.121 Parameter PwmDeInitApi

Add/remove the service Pwm_DeInit() from the code.

This option will toggle PWM_DE_INIT_API define macro in Pwm_Cfg.h file.

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

4.122 Parameter PwmGetOutputState

Add/remove the service Pwm_GetOutputState() from the code.

This option will toggle PWM_GET_OUTPUT_STATE_API define macro in Pwm_Cfg.h file.

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

4.123 Parameter PwmSetDutyCycle

Add/remove the service Pwm_SetDutyCycle() from the code.

This option will toggle PWM_SET_DUTY_CYCLE_API define macro in Pwm_Cfg.h file.

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

4.124 Parameter PwmSetOutputToIdle

Add/remove the service Pwm_SetOutputToIdle() from the code.

This option will toggle PWM_SET_OUTPUT_TO_IDLE_API define macro in Pwm_Cfg.h file.

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

4.125 Parameter PwmSetPeriodAndDuty

Add/remove the service Pwm_SetPeriodAndDuty() from the code.

This option will toggle PWM_SET_PERIOD_AND_DUTY_API define macro in Pwm_Cfg.h file.

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

4.126 Parameter PwmVersionInfoApi

Add/remove the service Pwm_GetVersionInfo() from the code.

This option will toggle PWM_VERSION_INFO_API define macro in Pwm_Cfg.h file.

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

4.127 Parameter PwmGetChannelStateApi

Add/remove the service Pwm_GetChannelState() from the code.

This option will toggle PWM_GET_CHANNEL_STATE_API define macro in Pwm_Cfg.h file.

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

4.128 Parameter PwmSetDutyCycle_NoUpdate

Add/remove the service Pwm_SetDutyCycle_NoUpdate() from the code.

This option will toggle PWM_SET_DUTY_CYCLE_NO_UPDATE_API define macro in Pwm_Cfg.h file.

<note>This option is activated only when PwmGeneral/PwmMultiChannelSync is enabled.</note>

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

4.129 Parameter PwmSetPeriodAndDuty_NoUpdate

Add/remove the service Pwm_SetPeriodAndDuty_NoUpdate() from the code.

This option will toggle PWM_SET_PERIOD_AND_DUTY_NO_UPDATE_API define macro in Pwm_Cfg.h file.

<note>This option is activated only when PwmGeneral/PwmMultiChannelSync is enabled.</note>

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

4.130 Parameter PwmSetPhaseShift

Add/remove the service Pwm_SetPhaseShift() from the code.

This option will toggle PWM_SET_PHASE_SHIFT_API define macro in Pwm_Cfg.h file.

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

4.131 Parameter PwmSetPhaseShift_NoUpdate

Add/remove the service Pwm_SetPhaseShift_NoUpdate() from the code.

This option will toggle PWM_SET_PHASE_SHIFT_NO_UPDATE_API define macro in Pwm_Cfg.h file.

<note>This option is activated only when PwmGeneral/PwmMultiChannelSync is enabled.</note>

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

4.132 Parameter PwmSetDutyPhaseShift

Add/remove the service Pwm_SetDutyPhaseShift() from the code.

This option will toggle PWM_SET_DUTY_PHASE_SHIFT_API define macro in Pwm_Cfg.h file.

<note>This option is activated only when PwmGeneral/PwmMultiChannelSync is enabled.</note>

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

4.133 Parameter PwmSetChannelDeadTime

Add/remove the service Pwm_SetChannelDeadTime() from the code.

This option will toggle PWM_SETCHANNELDEADTIME_API define macro.

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

4.134 Parameter PwmSetCounterBusApi

Switch to indicate that the Pwm_SetCounterBus is supported.

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

4.135 Parameter PwmSetChannelOutputApi

Switch to indicate that the Pwm_SetChannelOutput is supported.

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

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

4.136 Parameter PwmSetTriggerDelayApi

Switch to indicate that the PwmSetTriggerDelay is supported.

This function is called when the prescaler value needs to be change to maintain same period at different frequency.

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

4.137 Parameter PwmEmiosFastUpdateApi

Switch to indicate that the PwmEmiosFastUpdateApi is supported.

API for fast PWM update, suitable for multi-phase motor control.

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

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

4.138 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

4.139 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
valueConfigClasses	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
defaultValue	4
max	4
min	4

4.140 Parameter ArReleaseMinorVersion

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

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

4.141 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
defaultValue	0
max	0
min	0

4.142 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	121
max	121
min	121

4.143 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	3
max	3
min	3

4.144 Parameter SwMinorVersion

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

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

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

4.145 Parameter SwPatchVersion

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

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

4.146 Parameter VendorApiInfix

In driver modules which can be instantiated several times on a single ECU, BSW00347 requires that the name of APIs is extended by the VendorId and a vendor specific name.

This parameter is used to specify the vendor specific name. In total, the implementation specific name is generated as follows:

<ModuleName>_>VendorId>_<VendorApiInfix>.

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

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

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

4.147 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

Data Structure Index

6.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 7

Module Documentation

7.1 Emios Pwm IPL

7.1.1 Detailed Description

Data Structures

- struct Emios_Pwm_Ip_NotificationType
  Structure for notification. More...

Types Reference

- typedef void(* Emios_Pwm_Ip_CallbackType) (uint8 param)
  Notification function callback type.

Enum Reference

- enum Emios_Pwm_Ip_PwmType
  Emios PWM Channel modes.
- enum Emios_Pwm_Ip_MasterBusModeType
  Emios PWM master bus modes.
- enum Emios_Pwm_Ip_StatusType
  Status return codes.
- enum Emios_Pwm_Ip_PolarityType
  PWM output polarity.
- enum Emios_Pwm_Ip_OutputStateType
  PWM output pin state.
- enum Emios_Pwm_Ip_InternalClkPsType
  Internal pre-scaler factor selection for the clock source.
- enum Emios_Pwm_Ip_InternalPsSrcType

Internal prescaler source.

- enum Emios_Pwm_Ip_OutDisableSourceType

  Output Disable select.
- enum Emios_Pwm_Ip_InterruptType

  Interrupt types enabled for the channel.
- enum Emios_Pwm_Ip_CounterBusSourceType

  Counter bus select.
- enum Emios_Pwm_Ip_PwmModeType

  Supported channel PWM modes.

Function Reference

- void Emios_Pwm_Ip_InitChannel (uint8 Instance, Emios_Pwm_Ip_ChannelConfigType const *UserChCfg)

  Initialize PWM Mode.
- void Emios_Pwm_Ip_DeInitChannel (uint8 Instance, uint8 Channel)

  Reset eMIOS Channel to GPIO mode (reset default)
- void Emios_Pwm_Ip_ForceMatchLeadingEdge (uint8 Instance, uint8 Channel, boolean Enable)

  Allow the software to force the output flip-flop to the level corresponding to a match on leading edge. The FLAG bit is not set.
- void Emios_Pwm_Ip_ForceMatchTrailingEdge (uint8 Instance, uint8 Channel, boolean Enable)

  Allow the software to force the output flip-flop to the level corresponding to a match on trailing edge. The FLAG bit is not set.
- Emios_Pwm_Ip_PeriodType Emios_Pwm_Ip_GetPeriod (uint8 Instance, uint8 Channel)

  Get Period value in PWM mode.
- void Emios_Pwm_Ip_SetPeriod (uint8 Instance, uint8 Channel, Emios_Pwm_Ip_PeriodType NewPeriod)

  Set new Period value in PWM mode.
- Emios_Pwm_Ip_DutyType Emios_Pwm_Ip_GetDutyCycle (uint8 Instance, uint8 Channel)

  Get Duty Cycle value in PWM mode.
- Emios_Pwm_Ip_StatusType Emios_Pwm_Ip_SetDutyCycle (uint8 Instance, uint8 Channel, Emios_Pwm↩ _Ip_DutyType NewDutyCycle)

  Set new Duty Cycle value in PWM mode.
- Emios_Pwm_Ip_PeriodType Emios_Pwm_Ip_GetPhaseShift (uint8 Instance, uint8 Channel)

  Get Leading Edge Placement value in PWM mode.
- Emios_Pwm_Ip_StatusType Emios_Pwm_Ip_SetPhaseShift (uint8 Instance, uint8 Channel, Emios_Pwm↩ _Ip_DutyType PhaseShift)

  Set new Leading edge placement value in PWM mode.
- Emios_Pwm_Ip_PeriodType Emios_Pwm_Ip_GetDeadTime (uint8 Instance, uint8 Channel)

  Get dead time value in PWM mode.
- void Emios_Pwm_Ip_SetDeadTime (uint8 Instance, uint8 Channel, Emios_Pwm_Ip_PeriodType New↩ DeadTime)

  Set new dead time value in PWM mode.
- uint32 Emios_Pwm_Ip_GetTriggerPlacement (uint8 Instance, uint8 Channel)

  Get Trigger Placement value in PWM mode.
- void Emios_Pwm_Ip_SetTriggerPlacement (uint8 Instance, uint8 Channel, Emios_Pwm_Ip_PeriodType NewTriggerPlacement)

  Set new Trigger Placement value in PWM mode.
- Emios_Pwm_Ip_StatusType Emios_Pwm_Ip_ChannelEnterDebugMode (uint8 Instance, uint8 Channel)

Set a Channel enters freeze state, should be setting EMIOS_AllowEnterDebugMode first.

- void Emios_Pwm_Ip_ChannelStopDebugMode (uint8 Instance, uint8 Channel)

  Release a Channel from freeze state.
- Emios_Pwm_Ip_InterruptType Emios_Pwm_Ip_GetFlagRequest (uint8 Instance, uint8 Channel)

  Get the Unified Channel FLAG event generated. Interrupt or DMA request.
- void Emios_Pwm_Ip_SetFlagRequest (uint8 Instance, uint8 Channel, Emios_Pwm_Ip_InterruptType Event)

  Allow the Unified Channel FLAG bit to generate an interrupt signal or a DMA request signal.
- Emios_Pwm_Ip_OutputStateType Emios_Pwm_Ip_GetOutputState (uint8 Instance, uint8 Channel)

  Get the Unified Channel output pin logic level.
- void Emios_Pwm_Ip_SetOutputState (uint8 Instance, uint8 Channel, Emios_Pwm_Ip_OutputStateType OutputState)

  Set the state of output pin.
- void Emios_Pwm_Ip_SetOutputToNormal (uint8 Instance, uint8 Channel, uint16 DutyPercent, Emios_Pwm_Ip_Polarity Polarity, Emios_Pwm_Ip_PwmModeType Mode)

  Set the polarity and mode for current Channel as normal.
- Emios_Pwm_Ip_PwmModeType Emios_Pwm_Ip_GetChannelMode (uint8 Instance, uint8 Channel)

  Get mode of operation of the Unified Channel.
- uint8 Emios_Pwm_Ip_GetMasterBusChannel (uint8 Instance, uint8 Channel)

  Get master bus Channel.
- void Emios_Pwm_Ip_SetPreEnableClock (uint8 Instance, uint8 Channel, boolean Value)

  Set Prescaler Enable bit.
- void Emios_Pwm_Ip_SetBusSelected (uint8 Instance, uint8 Channel, Emios_Pwm_Ip_CounterBusSourceType Value)

  Set Bus Select bits.
- void Emios_Pwm_Ip_SetClockPs (uint8 Instance, uint8 Channel, Emios_Pwm_Ip_InternalClkPsType Value)

  This function set the value of the prescaler on eMios channels.
- void Emios_Pwm_Ip_ComparatorTransferEnable (uint8 Instance, uint32 ChannelMask)

  The function shall Enable the output update for the corresponding Channel.
- void Emios_Pwm_Ip_ComparatorTransferDisable (uint8 Instance, uint32 ChannelMask)

  The function shall disable the output update for the corresponding Channel.
- void Emios_Pwm_Ip_SyncUpdate (uint8 Instance)

  This function updates the duty cycle and-or period for the specified PWM Channel. The value written does not take effect until calling SyncUpdate API.
- void Emios_Pwm_Ip_UpdateUCRegA (uint8 Instance, uint8 Channel, Emios_Pwm_Ip_PeriodType Value)

  This function updates the value of UCRegA. It may be used to change duty cycle or phase shift with minimum overhead.
- void Emios_Pwm_Ip_UpdateUCRegB (uint8 Instance, uint8 Channel, Emios_Pwm_Ip_PeriodType Value)

  This function updates the value of UCRegB. It may be used to change duty cycle, phase shift or inserted dead time buffer with minimum overhead.
- void Emios_Pwm_Ip_IrqHandler (uint8 Instance, uint8 Channel)

  Interrupt handler for Emios Pwm channels.

7.1.2 Data Structure Documentation

7.1.2.1 struct Emios_Pwm_Ip_NotificationType

Structure for notification.

The structure used to notification

Definition at line 352 of file Emios_Pwm_Ip_Types.h.

Data Fields

Type	Name	Description
Emios_Pwm_Ip_CallbackType	CbFunction	Callback function pointer.
uint8	CbParameter	Callback function parameter pointer.

7.1.3 Types Reference

7.1.3.1 Emios_Pwm_Ip_CallbackType

```
typedef void(* Emios_Pwm_Ip_CallbackType) (uint8 param)
```

Notification function callback type.

Definition at line 346 of file Emios_Pwm_Ip_Types.h.

7.1.4 Enum Reference

7.1.4.1 Emios_Pwm_Ip_PwmType

```
enum Emios_Pwm_Ip_PwmType
```

Emios PWM Channel modes.

Enumerator

EMIOS_PWM_IP_HW_MODE_OPWFMB	Output Pulse Width and Frequency Modulation Buffered.
EMIOS_PWM_IP_HW_MODE_OPWMCB	Center Aligned Output Pulse Width Modulation Buffered.
EMIOS_PWM_IP_HW_MODE_OPWMB	Output Pulse Width Modulation Buffered.
EMIOS_PWM_IP_HW_MODE_OPWMT	Output Pulse-Width Modulation with Trigger.
EMIOS_PWM_IP_HW_MODE_DAOC	Double Action Output Compare.
EMIOS_PWM_IP_HW_MODE_OPWMC	Center Aligned Output Pulse Width Modulation
EMIOS_PWM_IP_HW_MODE_OPWM	Output Pulse Width Modulation.
EMIOS_PWM_IP_HW_MODE_OPWFM	Output Pulse Width and Frequency Modulation.

Definition at line 105 of file Emios_Pwm_Ip_HwAccess.h.

7.1.4.2 Emios_Pwm_Ip_MasterBusModeType

enum Emios_Pwm_Ip_MasterBusModeType

Emios PWM master bus modes.

Definition at line 128 of file Emios_Pwm_Ip_HwAccess.h.

7.1.4.3 Emios_Pwm_Ip_StatusType

enum Emios_Pwm_Ip_StatusType

Status return codes.

Common error codes will be a unified enumeration (C enum) that will contain all error codes (common and specific). There will be separate "error values spaces" (or slots), each of 256 positions, allocated per functionality.

Enumerator

EMIOS_PWM_IP_STATUS_SUCCESS	Generic operation success status.
EMIOS_PWM_IP_STATUS_ERROR	Generic operation failure status.
EMIOS_PWM_IP_STATUS_BUSY	Generic operation busy status.
EMIOS_PWM_IP_STATUS_TIMEOUT	Generic operation timeout status.
EMIOS_PWM_IP_STATUS_UNSUPPORTED	Generic operation unsupported status.
EMIOS_PWM_IP_STATUS_WRONG_MODE	EMIOS unsuccessful attempt selecting wrong mode.
EMIOS_PWM_IP_STATUS_CNT_BUS_OVER↩FLOW	EMIOS counter bus overflow.
EMIOS_PWM_IP_STATUS_WRONG_CNT_BUS	EMIOS unsuccessful attempt selecting wrong counter bus.
EMIOS_PWM_IP_STATUS_GLOBAL_FREEZ↩E_DISABLED	EMIOS must set global allow enter debug mode first.

Definition at line 107 of file Emios_Pwm_Ip_Types.h.

7.1.4.4 Emios_Pwm_Ip_PolarityType

enum Emios_Pwm_Ip_PolarityType

PWM output polarity.

This enumeration specifies polarity type of Emios

Enumerator

EMIOS_PWM_IP_ACTIVE_LOW	Output signal active low.
EMIOS_PWM_IP_ACTIVE_HIGH	Output signal active high.

Definition at line 136 of file Emios_Pwm_Ip_Types.h.

7.1.4.5 Emios_Pwm_Ip_OutputStateType

enum Emios_Pwm_Ip_OutputStateType

PWM output pin state.

This enumeration specifies output state of Emios channel pin

Enumerator

EMIOS_PWM_IP_OUTPUT_STATE_LOW	Output signal low.
EMIOS_PWM_IP_OUTPUT_STATE_HIGH	Output signal high.

Definition at line 148 of file Emios_Pwm_Ip_Types.h.

7.1.4.6 Emios_Pwm_Ip_InternalClkPsType

enum Emios_Pwm_Ip_InternalClkPsType

Internal pre-scaler factor selection for the clock source.

This enumeration specifies the clock divider value for the internal prescaler of Emios

Enumerator

EMIOS_PWM_IP_CLOCK_DIV_1	Divide by 1.
EMIOS_PWM_IP_CLOCK_DIV_2	Divide by 2.
EMIOS_PWM_IP_CLOCK_DIV_3	Divide by 3.
EMIOS_PWM_IP_CLOCK_DIV_4	Divide by 4.
EMIOS_PWM_IP_CLOCK_DIV_5	Divide by 5.
EMIOS_PWM_IP_CLOCK_DIV_6	Divide by 6.
EMIOS_PWM_IP_CLOCK_DIV_7	Divide by 7.
EMIOS_PWM_IP_CLOCK_DIV_8	Divide by 8.
EMIOS_PWM_IP_CLOCK_DIV_9	Divide by 9.

Enumerator

EMIOS_PWM_IP_CLOCK_DIV_10	Divide by 10.
EMIOS_PWM_IP_CLOCK_DIV_11	Divide by 11.
EMIOS_PWM_IP_CLOCK_DIV_12	Divide by 12.
EMIOS_PWM_IP_CLOCK_DIV_13	Divide by 13.
EMIOS_PWM_IP_CLOCK_DIV_14	Divide by 14.
EMIOS_PWM_IP_CLOCK_DIV_15	Divide by 15.
EMIOS_PWM_IP_CLOCK_DIV_16	Divide by 16.
EMIOS_PWM_IP_CLOCK_NONE	Prescaler Disabled.

Definition at line 161 of file Emios_Pwm_Ip_Types.h.

7.1.4.7 Emios_Pwm_Ip_InternalPsSrcType

enum Emios_Pwm_Ip_InternalPsSrcType

Internal prescaler source.

Definition at line 205 of file Emios_Pwm_Ip_Types.h.

7.1.4.8 Emios_Pwm_Ip_OutDisableSourceType

enum Emios_Pwm_Ip_OutDisableSourceType

Output Disable select.

Select one of the four output disable input signals

Enumerator

EMIOS_PWM_IP_OUTPUT_DISABLE_0	Channel output disable source 0.
EMIOS_PWM_IP_OUTPUT_DISABLE_1	Channel output disable source 1.
EMIOS_PWM_IP_OUTPUT_DISABLE_2	Channel output disable source 2.
EMIOS_PWM_IP_OUTPUT_DISABLE_3	Channel output disable source 3.
EMIOS_PWM_IP_OUTPUT_DISABLE_NONE	Channel output disable not used.

Definition at line 215 of file Emios_Pwm_Ip_Types.h.

7.1.4.9 Emios_Pwm_Ip_InterruptType

enum Emios_Pwm_Ip_InterruptType

Interrupt types enabled for the channel.

This enumeration specifies interrupt type of Emios

Enumerator

EMIOS_PWM_IP_NOTIFICATION_DISABLED	Interrupt/DMA requests are disabled.
EMIOS_PWM_IP_INTERRUPT_REQUEST	Interrupt requests are generated on FLAGs.
EMIOS_PWM_IP_DMA_REQUEST	DMA requests are generated on FLAGs.

Definition at line 233 of file Emios_Pwm_Ip_Types.h.

7.1.4.10 Emios_Pwm_Ip_CounterBusSourceType

enum Emios_Pwm_Ip_CounterBusSourceType

Counter bus select.

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

Enumerator

EMIOS_PWM_IP_BUS_A	Global counter bus A.
EMIOS_PWM_IP_BUS_BCDE	Local group counter bus.
EMIOS_PWM_IP_BUS_F	Global counter bus F.
EMIOS_PWM_IP_BUS_INTERNAL	Internal counter bus.

Definition at line 247 of file Emios_Pwm_Ip_Types.h.

7.1.4.11 Emios_Pwm_Ip_PwmModeType

enum Emios_Pwm_Ip_PwmModeType

Supported channel PWM modes.

This enumeration specifies mode type of Emios

Enumerator

EMIOS_PWM_IP_MODE_GPO	GPIO (output)
EMIOS_PWM_IP_MODE_DAOC_FLAG	Double Action Output Compare. FLAGs are generated only on B1 matches.
EMIOS_PWM_IP_MODE_DAOC_FLAG_BOTH	Double Action Output Compare. FLAGs are generated only on A1 & B1 matches.
EMIOS_PWM_IP_MODE_OPWMC_TRAIL_E↩ DGE_FLAG	Center Aligned Output Pulse Width Modulation (with trail edge dead-time). FLAGs are generated on the trailing edge.
EMIOS_PWM_IP_MODE_OPWMC_TRAIL_E↩ DGE_FLAG_BOTH	Center Aligned Output Pulse Width Modulation (with trail edge dead-time). FLAGs are generated on the both edges.
EMIOS_PWM_IP_MODE_OPWMC_LEAD_E↩ DGE_FLAG	Center Aligned Output Pulse Width Modulation (with lead edge dead-time). FLAG are generated on the leading edge.
EMIOS_PWM_IP_MODE_OPWMC_LEAD_E↩ DGE_FLAG_BOTH	Center Aligned Output Pulse Width Modulation (with lead edge dead-time). FLAG are generated in the both edges.
EMIOS_PWM_IP_MODE_OPWMT	Output Pulse-Width Modulation with Trigger.
EMIOS_PWM_IP_MODE_OPWFMB_FLAG	Output Pulse Width and Frequency Modulation Buffered. FLAGs are generated only on B1 matches.
EMIOS_PWM_IP_MODE_OPWFMB_FLAG_↩ BOTH	Output Pulse Width and Frequency Modulation Buffered. FLAGs are generated on both A1 & B1 matches.
EMIOS_PWM_IP_MODE_OPWMCB_TRAIL_↩ EDGE_FLAG	Center Aligned Output Pulse Width Modulation Buffered (with trail edge dead-time). FLAGs are generated on the trailing edge.
EMIOS_PWM_IP_MODE_OPWMCB_TRAIL_↩ EDGE_FLAG_BOTH	Center Aligned Output Pulse Width Modulation Buffered (with trail edge dead-time). FLAGs are generated on the both edges.
EMIOS_PWM_IP_MODE_OPWMCB_LEAD_↩ EDGE_FLAG	Center Aligned Output Pulse Width Modulation Buffered (with lead edge dead-time). FLAG are generated on the leading edge.
EMIOS_PWM_IP_MODE_OPWMCB_LEAD_↩ EDGE_FLAG_BOTH	Center Aligned Output Pulse Width Modulation Buffered (with lead edge dead-time). FLAG are generated in the both edges.
EMIOS_PWM_IP_MODE_OPWMB_FLAG	Output Pulse Width Modulation Buffered. FLAGs are generated only on trailing matches.
EMIOS_PWM_IP_MODE_OPWMB_FLAG_B↩ OTH	Output Pulse Width Modulation Buffered. FLAGs are generated on both leading and trailing matches.
EMIOS_PWM_IP_MODE_OPWM_IMMEDIAT↩ E_UPDATE_FLAG	Output Pulse Width Modulation (immediate update). FLAGs are generated only on trailing matches.
EMIOS_PWM_IP_MODE_OPWM_IMMEDIAT↩ E_UPDATE_FLAG_BOTH	Output Pulse Width Modulation (immediate update). FLAGs are generated on both leading and trailing matches.
EMIOS_PWM_IP_MODE_OPWM_NEXT_PE↩ RIOD_UPDATE_FLAG	Output Pulse Width Modulation (next period update). FLAGs are generated only on trailing matches.

Enumerator

EMIOS_PWM_IP_MODE_OPWM_NEXT_PE↩ RIOD_UPDATE_FLAG_BOTH	Output Pulse Width Modulation (next period update). FLAGs are generated on both leading and trailing matches.
EMIOS_PWM_IP_MODE_OPWFM_IMMEDIA↩ TE_UPDATE_FLAG	Output Pulse Width and Frequency Modulation (immediate update). FLAGs are generated only on BS1 matches.
EMIOS_PWM_IP_MODE_OPWFM_IMMEDIA↩ TE_UPDATE_FLAG_BOTH	Output Pulse Width and Frequency Modulation (immediate update). FLAGs are generated on both AS1 & BS1 matches.
EMIOS_PWM_IP_MODE_OPWFM_NEXT_P↩ ERIOD_UPDATE_FLAG	Output Pulse Width and Frequency Modulation (next period update). FLAGs are generated only on BS1 matches.
EMIOS_PWM_IP_MODE_OPWFM_NEXT_P↩ ERIOD_UPDATE_FLAG_BOTH	Output Pulse Width and Frequency Modulation (next period update). FLAGs are generated on both AS1 & BS1 matches.

Definition at line 263 of file Emios_Pwm_Ip_Types.h.

7.1.5 Function Reference

7.1.5.1 Emios_Pwm_Ip_InitChannel()

```
void Emios_Pwm_Ip_InitChannel (
            uint8 Instance,
            Emios_Pwm_Ip_ChannelConfigType const * UserChCfg )
```

Initialize PWM Mode.

Parameters

in	*Instance*	The eMIOS group id
in	*UserChCfg*	A pointer to the PWM configuration structure

Returns

void

7.1.5.2 Emios_Pwm_Ip_DeInitChannel()

```
void Emios_Pwm_Ip_DeInitChannel (
            uint8 Instance,
            uint8 Channel )
```

Reset eMIOS Channel to GPIO mode (reset default)

Parameters

in	*Instance*	The eMIOS group id
in	*Channel*	The Channel in this eMIOS group

Returns

void

7.1.5.3 Emios_Pwm_Ip_ForceMatchLeadingEdge()

```
void Emios_Pwm_Ip_ForceMatchLeadingEdge (
            uint8 Instance,
            uint8 Channel,
            boolean Enable )
```

Allow the software to force the output flip-flop to the level corresponding to a match on leading edge. The FLAG bit is not set.

Parameters

in	*Instance*	The eMIOS group id
in	*Channel*	The Channel in this eMIOS group
in	*Enable*	The Channel in Force Match Leading Edge or not

Returns

void

7.1.5.4 Emios_Pwm_Ip_ForceMatchTrailingEdge()

```
void Emios_Pwm_Ip_ForceMatchTrailingEdge (
            uint8 Instance,
            uint8 Channel,
            boolean Enable )
```

Allow the software to force the output flip-flop to the level corresponding to a match on trailing edge. The FLAG bit is not set.

Parameters

in	*Instance*	The eMIOS group id
in	*Channel*	The Channel in this eMIOS group
in	*Enable*	The Channel in Force Match Leading Edge or not

Returns

void

7.1.5.5 Emios_Pwm_Ip_GetPeriod()

```
Emios_Pwm_Ip_PeriodType Emios_Pwm_Ip_GetPeriod (
            uint8 Instance,
            uint8 Channel )
```

Get Period value in PWM mode.

Parameters

in	*Instance*	The eMIOS group id
in	*Channel*	The Channel in this eMIOS group
out	*RetPeriod*	A pointer to return period value

Returns

Emios_Pwm_Ip_PeriodType Value of period

7.1.5.6 Emios_Pwm_Ip_SetPeriod()

```
void Emios_Pwm_Ip_SetPeriod (
            uint8 Instance,
            uint8 Channel,
            Emios_Pwm_Ip_PeriodType NewPeriod )
```

Set new Period value in PWM mode.

Parameters

in	*Instance*	The eMIOS group id
in	*Channel*	The Channel in this eMIOS group
in	*NewPeriod*	New Period value

Returns

void

7.1.5.7 Emios_Pwm_Ip_GetDutyCycle()

```
Emios_Pwm_Ip_DutyType Emios_Pwm_Ip_GetDutyCycle (
            uint8 Instance,
            uint8 Channel )
```

Get Duty Cycle value in PWM mode.

Parameters

in	*Instance*	The eMIOS group id
in	*Channel*	The Channel in this eMIOS group

Returns

Emios_Pwm_Ip_DutyType Value of duty cycle

7.1.5.8 Emios_Pwm_Ip_SetDutyCycle()

```
Emios_Pwm_Ip_StatusType Emios_Pwm_Ip_SetDutyCycle (
            uint8 Instance,
            uint8 Channel,
            Emios_Pwm_Ip_DutyType NewDutyCycle )
```

Set new Duty Cycle value in PWM mode.

Parameters

in	*Instance*	The eMIOS group id
in	*Channel*	The Channel in this eMIOS group
in	*NewDutyCycle*	New duty cycle value

Returns

operation status

- EMIOS_PWM_IP_STATUS_SUCCESS : Operation was successful.
- EMIOS_PWM_IP_STATUS_ERROR : Operation failed, invalid input value.

7.1.5.9 Emios_Pwm_Ip_GetPhaseShift()

```
Emios_Pwm_Ip_PeriodType Emios_Pwm_Ip_GetPhaseShift (
            uint8 Instance,
            uint8 Channel )
```

Get Leading Edge Placement value in PWM mode.

Parameters

in	*Instance*	The eMIOS group id
in	*Channel*	The Channel in this eMIOS group

Returns

Emios_Pwm_Ip_PeriodType Value of leading edge placement in counter bus time base

7.1.5.10 Emios_Pwm_Ip_SetPhaseShift()

```
Emios_Pwm_Ip_StatusType Emios_Pwm_Ip_SetPhaseShift (
            uint8 Instance,
            uint8 Channel,
            Emios_Pwm_Ip_DutyType PhaseShift )
```

Set new Leading edge placement value in PWM mode.

Parameters

in	*Instance*	The eMIOS group id
in	*Channel*	The Channel in this eMIOS group
in	*PhaseShift*	New Phase Shift value

Returns

void

7.1.5.11 Emios_Pwm_Ip_GetDeadTime()

```
Emios_Pwm_Ip_PeriodType Emios_Pwm_Ip_GetDeadTime (
            uint8 Instance,
            uint8 Channel )
```

Get dead time value in PWM mode.

Parameters

in	*Instance*	The eMIOS group id
in	*Channel*	The Channel in this eMIOS group

Returns

Emios_Pwm_Ip_PeriodType Value of Dead Time

7.1.5.12 Emios_Pwm_Ip_SetDeadTime()

```
void Emios_Pwm_Ip_SetDeadTime (
            uint8 Instance,
            uint8 Channel,
            Emios_Pwm_Ip_PeriodType NewDeadTime )
```

Set new dead time value in PWM mode.

Parameters

in	*Instance*	The eMIOS group id
in	*Channel*	The Channel in this eMIOS group
in	*NewDeadTime*	New Dead Time value

Returns

void

7.1.5.13 Emios_Pwm_Ip_GetTriggerPlacement()

```
uint32 Emios_Pwm_Ip_GetTriggerPlacement (
            uint8 Instance,
            uint8 Channel )
```

Get Trigger Placement value in PWM mode.

Parameters

in	*Instance*	The eMIOS group id
in	*Channel*	The Channel in this eMIOS group

Returns

uint32 Value of Trigger Placement

7.1.5.14 Emios_Pwm_Ip_SetTriggerPlacement()

```
void Emios_Pwm_Ip_SetTriggerPlacement (
            uint8 Instance,
            uint8 Channel,
            Emios_Pwm_Ip_PeriodType NewTriggerPlacement )
```

Set new Trigger Placement value in PWM mode.

Parameters

in	*Instance*	The eMIOS group id
in	*Channel*	The Channel in this eMIOS group
in	*NewTriggerPlacement*	New Trigger Placement value

Returns

void

7.1.5.15 Emios_Pwm_Ip_ChannelEnterDebugMode()

```
Emios_Pwm_Ip_StatusType Emios_Pwm_Ip_ChannelEnterDebugMode (
            uint8 Instance,
            uint8 Channel )
```

Set a Channel enters freeze state, should be setting EMIOS_AllowEnterDebugMode first.

Parameters

in	*Instance*	The eMIOS group id
in	*Channel*	The Channel in this eMIOS group

Returns

operation status

- EMIOS_PWM_IP_STATUS_SUCCESS : Operation was successful.
- EMIOS_PWM_IP_STATUS_ERROR : Operation failed, invalid input value.

- EMIOS_PWM_IP_STATUS_ENABLE_GLOBAL_FRZ : Need call EMIOS_AllowEnterDebugMode first.

7.1.5.16 Emios_Pwm_Ip_ChannelStopDebugMode()

```
void Emios_Pwm_Ip_ChannelStopDebugMode (
            uint8 Instance,
            uint8 Channel )
```

Release a Channel from freeze state.

Parameters

in	*Instance*	The eMIOS group id
in	*Channel*	The Channel in this eMIOS group

Returns

void

7.1.5.17 Emios_Pwm_Ip_GetFlagRequest()

```
Emios_Pwm_Ip_InterruptType Emios_Pwm_Ip_GetFlagRequest (
            uint8 Instance,
            uint8 Channel )
```

Get the Unified Channel FLAG event generated. Interrupt or DMA request.

Parameters

in	*Instance*	The eMIOS group id
in	*Channel*	The Channel in this eMIOS group

Returns

The FLAG event response type

7.1.5.18 Emios_Pwm_Ip_SetFlagRequest()

```
void Emios_Pwm_Ip_SetFlagRequest (
            uint8 Instance,
            uint8 Channel,
            Emios_Pwm_Ip_InterruptType Event )
```

Allow the Unified Channel FLAG bit to generate an interrupt signal or a DMA request signal.

Parameters

in	*Instance*	The eMIOS group id
in	*Channel*	The Channel in this eMIOS group
in	*Event*	The FLAG event response type

Returns

void

7.1.5.19 Emios_Pwm_Ip_GetOutputState()

```
Emios_Pwm_Ip_OutputStateType Emios_Pwm_Ip_GetOutputState (
            uint8 Instance,
            uint8 Channel )
```

Get the Unified Channel output pin logic level.

Parameters

in	*Instance*	The eMIOS group id
in	*Channel*	The Channel in this eMIOS group

Returns

The Emios Channel output pin state HIGH/LOW

7.1.5.20 Emios_Pwm_Ip_SetOutputState()

```
void Emios_Pwm_Ip_SetOutputState (
            uint8 Instance,
            uint8 Channel,
            Emios_Pwm_Ip_OutputStateType OutputState )
```

Set the state of output pin.

Parameters

in	*Instance*	The eMIOS group id
in	*Channel*	The Channel in this eMIOS group
in	*OutputState*	The state of output pin

Returns

void

7.1.5.21 Emios_Pwm_Ip_SetOutputToNormal()

```
void Emios_Pwm_Ip_SetOutputToNormal (
            uint8 Instance,
            uint8 Channel,
            uint16 DutyPercent,
            Emios_Pwm_Ip_PolarityType Polarity,
            Emios_Pwm_Ip_PwmModeType Mode )
```

Set the polarity and mode for current Channel as normal.

Parameters

in	*Instance*	The eMIOS group id
in	*Channel*	The Channel in this eMIOS group
in	*DutyPercent*	The range of duty cycle value :0x00(0%) ->0x8000(100%)
in	*Polarity*	The polarity of Channel
in	*Mode*	Mode of Channel

Returns

void

7.1.5.22 Emios_Pwm_Ip_GetChannelMode()

```
Emios_Pwm_Ip_PwmModeType Emios_Pwm_Ip_GetChannelMode (
            uint8 Instance,
            uint8 Channel )
```

Get mode of operation of the Unified Channel.

Parameters

in	*Instance*	The eMIOS group id
in	*Channel*	The Channel in this eMIOS group

Returns

Emios_Pwm_Ip_PwmModeType

7.1.5.23 Emios_Pwm_Ip_GetMasterBusChannel()

```
uint8 Emios_Pwm_Ip_GetMasterBusChannel (
            uint8 Instance,
            uint8 Channel )
```

Get master bus Channel.

Parameters

in	*Instance*	The eMIOS group id
in	*Channel*	The Channel in this eMIOS group

Returns

Emios_Pwm_Ip_PwmModeType

7.1.5.24 Emios_Pwm_Ip_SetPreEnableClock()

```
void Emios_Pwm_Ip_SetPreEnableClock (
            uint8 Instance,
            uint8 Channel,
            boolean Value )
```

Set Prescaler Enable bit.

Parameters

in	*Instance*	The eMIOS group id
in	*Channel*	The Channel in this eMIOS group
in	*Value*	The value to set • 0 Prescaler disabled (no clock) • 1 Prescaler enabled

Returns

void

7.1.5.25 Emios_Pwm_Ip_SetBusSelected()

```
void Emios_Pwm_Ip_SetBusSelected (
            uint8 Instance,
            uint8 Channel,
            Emios_Pwm_Ip_CounterBusSourceType Value )
```

Set Bus Select bits.

Parameters

in	*Instance*	The eMIOS group id
in	*Channel*	The Channel in this eMIOS group
in	*Value*	The value to set

Returns

void

7.1.5.26 Emios_Pwm_Ip_SetClockPs()

```
void Emios_Pwm_Ip_SetClockPs (
            uint8 Instance,
            uint8 Channel,
            Emios_Pwm_Ip_InternalClkPsType Value )
```

This function set the value of the prescaler on eMios channels.

Parameters

in	*Instance*	The eMIOS group id
in	*Channel*	The Channel in this eMIOS group
in	*Value*	The value to set

Returns

void

7.1.5.27 Emios_Pwm_Ip_ComparatorTransferEnable()

```
void Emios_Pwm_Ip_ComparatorTransferEnable (
            uint8 Instance,
            uint32 ChannelMask )
```

The function shall Enable the output update for the corresponding Channel.

Parameters

in	*Instance*	Instance of EMIOS used.
in	*ChannelMask*	EMIOS hardware mask Channel used.

7.1.5.28 Emios_Pwm_Ip_ComparatorTransferDisable()

```
void Emios_Pwm_Ip_ComparatorTransferDisable (
            uint8 Instance,
            uint32 ChannelMask )
```

The function shall disable the output update for the corresponding Channel.

Parameters

in	*Instance*	Instance of EMIOS used.
in	*ChannelMask*	EMIOS hardware mask Channel used.

7.1.5.29 Emios_Pwm_Ip_SyncUpdate()

```
void Emios_Pwm_Ip_SyncUpdate (
            uint8 Instance )
```

This function updates the duty cycle and-or period for the specified PWM Channel. The value written does not take effect until calling SyncUpdate API.

Parameters

Instance	eMIOS hardware module index

Returns

void

7.1.5.30 Emios_Pwm_Ip_UpdateUCRegA()

```
void Emios_Pwm_Ip_UpdateUCRegA (
            uint8 Instance,
            uint8 Channel,
            Emios_Pwm_Ip_PeriodType Value )
```

This function updates the value of UCRegA. It may be used to change duty cycle or phase shift with minimum overhead.

Parameters

Instance	eMIOS hardware module index
Channel	The Channel in this eMIOS group
Value	The value to set

Returns

void

7.1.5.31 Emios_Pwm_Ip_UpdateUCRegB()

```
void Emios_Pwm_Ip_UpdateUCRegB (
            uint8 Instance,
            uint8 Channel,
            Emios_Pwm_Ip_PeriodType Value )
```

This function updates the value of UCRegB. It may be used to change duty cycle, phase shift or inserted dead time buffer with minimum overhead.

Parameters

Instance	eMIOS hardware module index
Channel	The Channel in this eMIOS group
Value	The value to set

Returns

void

7.1.5.32 Emios_Pwm_Ip_IrqHandler()

```
void Emios_Pwm_Ip_IrqHandler (
            uint8 Instance,
            uint8 Channel )
```

Interrupt handler for Emios Pwm channels.

Interrupt handler that clears the flags and calls the user notification function.

Parameters

in	*instance*	Emios instance id on which the interrupt occured.
in	*channel*	Channell id within the emios instance that triggered the interrupt.

7.2 FlexIO Pwm IPL

7.2.1 Detailed Description

Function Reference

- Flexio_Pwm_Ip_StatusType Flexio_Pwm_Ip_InitChannel (uint8 InstanceId, const Flexio_Pwm_Ip_ChannelConfigTyp *const UserCfg)

  Initialize a flexio channel in pwm mode.
- Flexio_Pwm_Ip_StatusType Flexio_Pwm_Ip_DeInitChannel (uint8 InstanceId, uint8 Channel)

  Deinitialize a flexio channel.
- Flexio_Pwm_Ip_StatusType Flexio_Pwm_Ip_UpdateClockPrescaler (uint8 InstanceId, uint8 Channel, Flexio_Pwm_Ip_ClockPrescalerType Prescaler)

  Set clock prescaler for a flexio channel.
- Flexio_Pwm_Ip_StatusType Flexio_Pwm_Ip_ForceOuputLevel (uint8 InstanceId, uint8 Channel, boolean Level)

  Force the pin ouput to logic one or zero.
- Flexio_Pwm_Ip_StatusType Flexio_Pwm_Ip_UpdatePeriodDuty (uint8 InstanceId, uint8 Channel, uint16 Period, uint16 DutyCycle)

  Set a new value for duty cycle and period of the channel.
- boolean Flexio_Pwm_Ip_GetOutputState (uint8 InstanceId, uint8 Channel)

  Get the logic level of the channel ouput.
- Flexio_Pwm_Ip_StatusType Flexio_Pwm_Ip_UpdateInterruptMode (uint8 InstanceId, uint8 Channel, Flexio_Pwm_Ip_InterruptType IrqMode)

  Update the interrupt mode for a channel.
- uint16 Flexio_Pwm_Ip_GetPeriod (uint8 InstanceId, uint8 Channel)

  Getting the period for a channel.

7.2.2 Function Reference

7.2.2.1 Flexio_Pwm_Ip_InitChannel()

```
Flexio_Pwm_Ip_StatusType Flexio_Pwm_Ip_InitChannel (
            uint8 InstanceId,
            const Flexio_Pwm_Ip_ChannelConfigType *const UserCfg )
```

Initialize a flexio channel in pwm mode.

The function will initialize one timer and pin of the selected flexio channel in pwm mode, with the configuration of the user. The interrupts will be disabled.

Parameters

in	*InstanceId*	The Flexio instance id
in	*UserCfg*	The channel configuration for the selected Flexio instance

Returns

FLEXIO_PWM_IP_STATUS_SUCCESS - if the initialization was successfull

7.2.2.2 Flexio_Pwm_Ip_DeInitChannel()

```
Flexio_Pwm_Ip_StatusType Flexio_Pwm_Ip_DeInitChannel (
            uint8 InstanceId,
            uint8 Channel )
```

Deinitialize a flexio channel.

The function will reset the timer and pin of the selected flexio channel.

Parameters

in	*Instance↩ Id*	The Flexio instance id
in	*Channel*	The channel id for the selected Flexio instance

Returns

FLEXIO_PWM_IP_STATUS_SUCCESS - if the deinitialization was successfull

7.2.2.3 Flexio_Pwm_Ip_UpdateClockPrescaler()

```
Flexio_Pwm_Ip_StatusType Flexio_Pwm_Ip_UpdateClockPrescaler (
            uint8 InstanceId,
            uint8 Channel,
            Flexio_Pwm_Ip_ClockPrescalerType Prescaler )
```

Set clock prescaler for a flexio channel.

The function will change the prescaller value for the selected flexio channel.

Parameters

in	*Instance↩ Id*	The Flexio instance id
in	*Channel*	The channel id for the selected Flexio instance
in	*Prescaler*	The new prescaler value

Returns

FLEXIO_PWM_IP_STATUS_SUCCESS - if the initialization was successfull

7.2.2.4 Flexio_Pwm_Ip_ForceOuputLevel()

```
Flexio_Pwm_Ip_StatusType Flexio_Pwm_Ip_ForceOuputLevel (
            uint8 InstanceId,
            uint8 Channel,
            boolean Level )
```

Force the pin ouput to logic one or zero.

The function will override the pin ouput of the selected flexio channel to the desired logic level.

Parameters

in	*InstanceId*	The Flexio instance id
in	*Channel*	The channel id for the selected Flexio instance
in	*Level*	The logic level that the pin ouput should be set at.

Returns

FLEXIO_PWM_IP_STATUS_SUCCESS - if the initialization was successfull

7.2.2.5 Flexio_Pwm_Ip_UpdatePeriodDuty()

```
Flexio_Pwm_Ip_StatusType Flexio_Pwm_Ip_UpdatePeriodDuty (
            uint8 InstanceId,
            uint8 Channel,
            uint16 Period,
            uint16 DutyCycle )
```

Set a new value for duty cycle and period of the channel.

The function will update the selected flexio channel with the new values for the duty cycle and period.

Parameters

in	*InstanceId*	The Flexio instance id
in	*Channel*	The channel id for the selected Flexio instance
in	*Period*	The new value for the period
in	*DutyCycle*	The new value for the duty cycle

Returns

FLEXIO_PWM_IP_STATUS_SUCCESS - if the initialization was successfull

7.2.2.6 Flexio_Pwm_Ip_GetOutputState()

```
boolean Flexio_Pwm_Ip_GetOutputState (
            uint8 InstanceId,
            uint8 Channel )
```

Get the logic level of the channel ouput.

The function will return the logic level that the selected flexio channel is driving on on the output pin.

Parameters

in	*Instance↩ Id*	The Flexio instance id
in	*Channel*	The channel id for the selected Flexio instance

Returns

FLEXIO_PWM_IP_STATUS_SUCCESS - if the initialization was successfull

7.2.2.7 Flexio_Pwm_Ip_UpdateInterruptMode()

```
Flexio_Pwm_Ip_StatusType Flexio_Pwm_Ip_UpdateInterruptMode (
            uint8 InstanceId,
            uint8 Channel,
            Flexio_Pwm_Ip_InterruptType IrqMode )
```

Update the interrupt mode for a channel.

The function will set a new mode for the flag event response on the selected channel.

Parameters

in	*Instance↩ Id*	The Flexio instance id
in	*Channel*	The channel id for the selected Flexio instance
in	*IrqMode*	The new irq mode for the channel

Returns

FLEXIO_PWM_IP_STATUS_SUCCESS - if the initialization was successfull

7.2.2.8 Flexio_Pwm_Ip_GetPeriod()

```
uint16 Flexio_Pwm_Ip_GetPeriod (
            uint8 InstanceId,
            uint8 Channel )
```

Getting the period for a channel.

The function will get the period on the selected channel.

Parameters

in	*Instance↩ Id*	The Flexio instance id
in	*Channel*	The channel id for the selected Flexio instance

Returns

uint16

7.3 FlexPwm IPL

7.3.1 Detailed Description

Data Structures

- struct FlexPwm_Ip_ChannelCfgTypes
  FlexPwm channel configuration parameters structure. More...
- struct FlexPwm_Ip_FaultChCfgTypes
  Fault channels configuration parameters structure. More...
- struct FlexPwm_Ip_SubModuleCfgTypes
  Configuration Sub Module parameters structure. More...
- struct FlexPwm_Ip_InstanceCfgTypes
  Configuration Instance parameters structure. More...

Types Reference

- typedef void(∗ FlexPwm_Ip_FaultCallbackType) (void)
  Fault channel notification typedef.

Enum Reference

- enum FlexPwm_Ip_StateTypes
  FlexPWM state Type options. Implements : FlexPwm_Ip_StateTypes.
- enum FlexPwm_Ip_ChannelTypes
  FlexPWM channel Type options. Implements : FlexPwm_Ip_ChannelTypes.
- enum FlexPwm_Ip_PolarityTypes
  FlexPWM polarity Type options. Implements : FlexPwm_Ip_PolarityTypes.
- enum FlexPwm_Ip_OutputStateTypes
  FlexPWM output state Type options. Implements : FlexPwm_Ip_OutputStateTypes.
- enum FlexPwm_Ip_InterruptTypes
  FlexPWM Interrupts Type options. Implements : FlexPwm_Ip_InterruptTypes.
- enum FlexPwm_Ip_OuputTriggerTypes
  FlexPWM Output Trigger control Implements : FlexPwm_Ip_OuputTriggerTypes.
- enum FlexPwm_Ip_FaultStateChTypes
  PWM output state during fault conditions Implements : FlexPwm_Ip_FaultStateChTypes.
- enum FlexPwm_Ip_SignalPwmTypes
  FlexPWM Signal Type options. Implements : FlexPwm_Ip_SignalPwmTypes.
- enum FlexPwm_Ip_ClockSourceTypes
  PWM clock source selection. Implements : FlexPwm_Ip_ClockSourceTypes.
- enum FlexPwm_Ip_PrescalerTypes
  PWM pre scaler factor selection for clock source. Implements : FlexPwm_Ip_PrescalerTypes.
- enum FlexPwm_Ip_PrescalerModeTypes

Prescaler mode type.

- enum FlexPwm_Ip_ReloadSourceSelTypes

  Options available to select reload signal for loading the buffered-registers with new values. Implements : FlexPwm_Ip_ReloadSourceSelTypes.
- enum FlexPwm_Ip_ReloadTypes

  Options available on how to load the buffered-registers with new values. Implements : FlexPwm_Ip_ReloadTypes.
- enum FlexPwm_Ip_LoadFrequencyTypes

  PWM load frequency selection. Implements :FlexPwm_Ip_LoadFrequencyTypes.
- enum FlexPwm_Ip_ForceSourceSelTypes

  Options that can trigger a PWM FORCE_OUT. Implements :FlexPwm_Ip_ForceSourceSelTypes.
- enum FlexPwm_Ip_ChannelPairTypes

  Options available for the PWM A & B pair operation. Implements : FlexPwm_Ip_ChannelPairTypes.
- enum FlexPwm_Ip_SrcCompSelTypes

  Source selection for the generation of complementary PWM pair output. Implements : FlexPwm_Ip_SrcCompSelTypes.
- enum FlexPwm_Ip_InitControlSelTypes

  PWM counter initialization options. Implements : FlexPwm_Ip_InitControlSelTypes.
- enum FlexPwm_Ip_DeadTimeCountTypes

  FlexPWM dead time control available counter registers. Implements : FlexPwm_Ip_DeadTimeCountTypes.

7.3.2 Data Structure Documentation

7.3.2.1 struct FlexPwm_Ip_ChannelCfgTypes

FlexPwm channel configuration parameters structure.

FlexTimer Channel configuration parameters structure type

Definition at line 382 of file FlexPwm_Ip_Types.h.

Data Fields

Type	Name	Description
FlexPwm_Ip_ChannelTypes	ChannelId	FlexPWM channel Type options.
FlexPwm_Ip_PolarityTypes	Polarity	Output Polarity
uint16	DutyCycle	Duty Cycle
uint16	PhaseShiftTicks	The Phase Shift of the current FlexPWM channel in ticks.
FlexPwm_Ip_OuputTriggerTypes	OutputTrig	Define the CTU trigger configuration for FlexPWM channels.
FlexPwm_Ip_FaultStateChTypes	FaultState	Specify the fault state for the PWM channel output during fault, stop and debug conditions.
uint8	DisOutputFault	Disable the PWM output on detection of fault on fault channel.
FlexPwm_Ip_InterruptTypes	InterruptType	Type of interrupts
FlexPwm_Ip_NotificationType	ChannelCb	Callback for the flexPwm channels

7.3.2.2 struct FlexPwm_Ip_FaultChCfgTypes

Fault channels configuration parameters structure.

FlexTimer fault channels configuration parameters structure type

Definition at line 406 of file FlexPwm_Ip_Types.h.

Data Fields

Type	Name	Description
uint8	FaultLevel	Select the active logic level of the individual fault inputs.
uint8	AutoFaultClearing	Select automatic or manual clearing of faults.
uint8	FaultSafetyMode	select the safety mode during manual fault clearing.
uint8	FullCycle	This is used to control the timing for re-enabling the PWM outputs after a fault condition.
uint8	FaultInterruptEn	Enables fault interrupt.
FlexPwm_Ip_FaultCallbackType	FaultNotification[(uint8) 1U]	Fault notification callbacks

7.3.2.3 struct FlexPwm_Ip_SubModuleCfgTypes

Configuration Sub Module parameters structure.

FlexPwm IP specific Sub Module configuration structure type

Definition at line 421 of file FlexPwm_Ip_Types.h.

Data Fields

Type	Name	Description
uint8	SubModuleId	FlexPWM submodule Id
FlexPwm_Ip_ClockSourceTypes	ClkSource	Select clock source for current FlexPWM submodule.
FlexPwm_Ip_InitControlSelTypes	InitControl	Option to initialize the counter.
FlexPwm_Ip_PrescalerTypes	Prescaler	Select pre-scaler for clock source.
FlexPwm_Ip_PrescalerTypes	PrescalerAlt	Select pre-scaler alternate for clock source.
FlexPwm_Ip_ReloadSourceSelTypes	ReloadSrc	Select reload signal for loading the buffered-registers with new values.
FlexPwm_Ip_ReloadTypes	Reload	Select the options how to load the buffered-registers with new values.
FlexPwm_Ip_LoadFrequencyTypes	LoadFrq	Load frequency selection.
FlexPwm_Ip_ForceSourceSelTypes	ForceSrc	Options that can trigger a PWM FORCE_OUT.

Data Fields

Type	Name	Description
FlexPwm_Ip_ChannelPairTypes	ChPair	Options available for the PWM A & B pair operation.
FlexPwm_Ip_SignalPwmTypes	SigPwm	Signal Type options
uint16	InitVal	Shift simultaneously in time the rising edges of the PWM channels of 2 or more FlexPWM submodules
FlexPwm_Ip_SrcCompSelTypes	CompSrc	Source selection for the generation of complementary PWM pair output.
uint16	DeadTimeCount0	Controls the dead time during 0 to 1 transitions of the PWMA output
uint16	DeadTimeCount1	Controls the dead time during 0 to 1 transitions of the complementary PWMB output
uint8	DebugModeEnable	Enable/disable Debug Mode
const FlexPwm_Ip_ChannelCfgTypes *const *	ChannelCfgArray	Configure the channels in each subModule
uint8	NumChannelCfg	Number of the channels that is configured
uint16	Period	Period in ticks

7.3.2.4 struct FlexPwm_Ip_InstanceCfgTypes

Configuration Instance parameters structure.

FlexPwm IP specific instance configuration structure type

Definition at line 449 of file FlexPwm_Ip_Types.h.

Data Fields

Type	Name	Description
uint16	OutputEnable	Enable the channel outputs of each instance.
uint16	MasterControlRun	Enables the clocks to the PWM generator in each instance
boolean	FaultFunctionalityEnable	Allow to use fault Functionality or not
uint8	FaultGlitchStretchEnable	Enable the fault glitch stretching logic
uint8	FaultFilterCounter	Represents the number of consecutive samples
uint8	FaultFilterPeriod	Represent the sampling period
uint8	NoCombinationalPath	This is used to control the combinational path from the fault inputs to the PWM outputs.
const FlexPwm_Ip_FaultChCfgTypes *	FaultChCfg	Configure the channel fault in each instance

Data Fields

Type	Name	Description
const FlexPwm_Ip_SubModuleCfgTypes *const *	SubModuleCfgArray	Configure the SubModule in each instance
uint8	NumSubModuleCfg	Number of the subModule that is configured

7.3.3 Types Reference

7.3.3.1 FlexPwm_Ip_FaultCallbackType

```
typedef void(* FlexPwm_Ip_FaultCallbackType) (void)
```

Fault channel notification typedef.

Definition at line 398 of file FlexPwm_Ip_Types.h.

7.3.4 Enum Reference

7.3.4.1 FlexPwm_Ip_StateTypes

```
enum FlexPwm_Ip_StateTypes
```

FlexPWM state Type options. Implements : FlexPwm_Ip_StateTypes.

Enumerator

FLEXPWM_IP_UNINIT_STATE	uninit state
FLEXPWM_IP_INIT_STATE	init state
FLEXPWM_IP_IDLE_STATE	idle state

Definition at line 105 of file FlexPwm_Ip_Types.h.

7.3.4.2 FlexPwm_Ip_ChannelTypes

```
enum FlexPwm_Ip_ChannelTypes
```

FlexPWM channel Type options. Implements : FlexPwm_Ip_ChannelTypes.

Enumerator

FLEXPWM_IP_PWMX	channel X
FLEXPWM_IP_PWMA	channel A
FLEXPWM_IP_PWMB	channel B

Definition at line 116 of file FlexPwm_Ip_Types.h.

7.3.4.3 FlexPwm_Ip_PolarityTypes

`enum FlexPwm_Ip_PolarityTypes`

FlexPWM polarity Type options. Implements : FlexPwm_Ip_PolarityTypes.

Enumerator

FLEXPWM_IP_POL_HIGH	output not inverted
FLEXPWM_IP_POL_LOW	output inverted

Definition at line 127 of file FlexPwm_Ip_Types.h.

7.3.4.4 FlexPwm_Ip_OutputStateTypes

`enum FlexPwm_Ip_OutputStateTypes`

FlexPWM output state Type options. Implements : FlexPwm_Ip_OutputStateTypes.

Enumerator

FLEXPWM_IP_OUT_LOW	output low
FLEXPWM_IP_OUT_HIGH	output high

Definition at line 137 of file FlexPwm_Ip_Types.h.

7.3.4.5 FlexPwm_Ip_InterruptTypes

enum FlexPwm_Ip_InterruptTypes

FlexPWM Interrupts Type options. Implements : FlexPwm_Ip_InterruptTypes.

Enumerator

FLEXPWM_IP_DISABLE_INT	Disable all Interrupt
FLEXPWM_IP_RELOAD_INT	Reload Interrupt
FLEXPWM_IP_COMPARE_INT	Compare Interrupt

Definition at line 147 of file FlexPwm_Ip_Types.h.

7.3.4.6 FlexPwm_Ip_OuputTriggerTypes

enum FlexPwm_Ip_OuputTriggerTypes

FlexPWM Output Trigger control Implements : FlexPwm_Ip_OuputTriggerTypes.

Enumerator

FLEXPWM_IP_NO_TRIGGER	No trigger
FLEXPWM_IP_FIRST_EDGE_X	Enable the generation of OUT_TRIG outputs based on the counter value matching VAL0 registers
FLEXPWM_IP_SECOND_EDGE_X	Enable the generation of OUT_TRIG outputs based on the counter value matching VAL1 registers
FLEXPWM_IP_BOTH_EDGES_X	Enable the generation of OUT_TRIG outputs based on the counter value matching VAL0 and VAL1 registers
FLEXPWM_IP_FIRST_EDGE_A	Enable the generation of OUT_TRIG outputs based on the counter value matching VAL2 registers
FLEXPWM_IP_SECOND_EDGE_A	Enable the generation of OUT_TRIG outputs based on the counter value matching VAL3 registers
FLEXPWM_IP_BOTH_EDGES_A	Enable the generation of OUT_TRIG outputs based on the counter value matching VAL2 and VAL3 registers
FLEXPWM_IP_FIRST_EDGE_B	Enable the generation of OUT_TRIG outputs based on the counter value matching VAL4 registers
FLEXPWM_IP_SECOND_EDGE_B	Enable the generation of OUT_TRIG outputs based on the counter value matching VAL5 registers
FLEXPWM_IP_BOTH_EDGES_B	Enable the generation of OUT_TRIG outputs based on the counter value matching VAL4 and VAL5 registers

Definition at line 158 of file FlexPwm_Ip_Types.h.

7.3.4.7 FlexPwm_Ip_FaultStateChTypes

enum FlexPwm_Ip_FaultStateChTypes

PWM output state during fault conditions Implements : FlexPwm_Ip_FaultStateChTypes.

Enumerator

FLEXPWM_IP_OUTPUT_STATE_LOGIC_0	Output is forced to logic 0 state prior to consideration of output polarity control
FLEXPWM_IP_OUTPUT_STATE_LOGIC_1	Output is forced to logic 1 state prior to consideration of output polarity control
FLEXPWM_IP_OUTPUT_STATE_TRISTATED	Output is tri-stated

Definition at line 176 of file FlexPwm_Ip_Types.h.

7.3.4.8 FlexPwm_Ip_SignalPwmTypes

enum FlexPwm_Ip_SignalPwmTypes

FlexPWM Signal Type options. Implements : FlexPwm_Ip_SignalPwmTypes.

Enumerator

FLEXPWM_IP_CENTER_ALIGNED	Center-aligned PWM
FLEXPWM_IP_EDGE_ALIGNED	Edge aligned PWM
FLEXPWM_IP_PHASE_SHIFTED	Phase shifted PWM
FLEXPWM_IP_DOUBLE_SWITCHING	Double switching PWM

Definition at line 187 of file FlexPwm_Ip_Types.h.

7.3.4.9 FlexPwm_Ip_ClockSourceTypes

enum FlexPwm_Ip_ClockSourceTypes

PWM clock source selection. Implements : FlexPwm_Ip_ClockSourceTypes.

Enumerator

FLEXPWM_IP_CLKSOURCE_PERIPHERAL_↩ CLK	The peripheral clock is used as the clock.
FLEXPWM_IP_CLKSOURCE_EXT_CLK	EXT_CLK is used as the clock.
FLEXPWM_IP_CLKSOURCE_AUX_CLK	Clock of the submodule 0 (AUX_CLK) is used as the source clock.

Definition at line 199 of file FlexPwm_Ip_Types.h.

7.3.4.10 FlexPwm_Ip_PrescalerTypes

enum FlexPwm_Ip_PrescalerTypes

PWM pre scaler factor selection for clock source. Implements : FlexPwm_Ip_PrescalerTypes.

Enumerator

FLEXPWM_IP_DIV1	PWM clock frequency = fclk/1.
FLEXPWM_IP_DIV2	PWM clock frequency = fclk/2.
FLEXPWM_IP_DIV4	PWM clock frequency = fclk/4.
FLEXPWM_IP_DIV8	PWM clock frequency = fclk/8.
FLEXPWM_IP_DIV16	PWM clock frequency = fclk/16.
FLEXPWM_IP_DIV32	PWM clock frequency = fclk/32.
FLEXPWM_IP_DIV64	PWM clock frequency = fclk/64.
FLEXPWM_IP_DIV128	PWM clock frequency = fclk/128.

Definition at line 210 of file FlexPwm_Ip_Types.h.

7.3.4.11 FlexPwm_Ip_PrescalerModeTypes

enum FlexPwm_Ip_PrescalerModeTypes

Prescaler mode type.

This enumeration specifies the possible types of prescalers used to configure base-clock timers

Enumerator

FLEXPWM_IP_PRIMARY_PRESCALER	Selected value is the default/primary prescaler.
FLEXPWM_IP_ALTERNATIVE_PRESCALER	Selected value is the alternative configured prescaler.

Definition at line 227 of file FlexPwm_Ip_Types.h.

7.3.4.12 FlexPwm_Ip_ReloadSourceSelTypes

enum FlexPwm_Ip_ReloadSourceSelTypes

Options available to select reload signal for loading the buffered-registers with new values. Implements : FlexPwm↩ _Ip_ReloadSourceSelTypes.

Enumerator

FLEXPWM_IP_LOCAL_RELOAD	The local RELOAD signal is used to reload registers.
FLEXPWM_IP_MASTER_RELOAD	The master RELOAD signal (from submodule 0) is used to reload registers

Definition at line 238 of file FlexPwm_Ip_Types.h.

7.3.4.13 FlexPwm_Ip_ReloadTypes

enum FlexPwm_Ip_ReloadTypes

Options available on how to load the buffered-registers with new values. Implements : FlexPwm_Ip_ReloadTypes.

Enumerator

FLEXPWM_IP_RELOAD_IMMEDIATE	Buffered-registers get loaded with new values as soon as LDOK bit is set.
FLEXPWM_IP_RELOAD_FULL	Registers loaded on a PWM full cycle.
FLEXPWM_IP_RELOAD_HALF	Registers loaded on a PWM half cycle.
FLEXPWM_IP_RELOAD_FULL_HALF	Registers loaded on a PWM half & full cycle.

Definition at line 248 of file FlexPwm_Ip_Types.h.

7.3.4.14 FlexPwm_Ip_LoadFrequencyTypes

enum FlexPwm_Ip_LoadFrequencyTypes

PWM load frequency selection. Implements :FlexPwm_Ip_LoadFrequencyTypes.

Enumerator

FLEXPWM_IP_LDFQ_EACH1	Every 1 PWM opportunity.
FLEXPWM_IP_LDFQ_EACH2	Every 2 PWM opportunities.
FLEXPWM_IP_LDFQ_EACH3	Every 3 PWM opportunities.
FLEXPWM_IP_LDFQ_EACH4	Every 4 PWM opportunities.
FLEXPWM_IP_LDFQ_EACH5	Every 5 PWM opportunities.
FLEXPWM_IP_LDFQ_EACH6	Every 6 PWM opportunities.
FLEXPWM_IP_LDFQ_EACH7	Every 7 PWM opportunities.
FLEXPWM_IP_LDFQ_EACH8	Every 8 PWM opportunities.
FLEXPWM_IP_LDFQ_EACH9	Every 9 PWM opportunities.
FLEXPWM_IP_LDFQ_EACH10	Every 10 PWM opportunities.
FLEXPWM_IP_LDFQ_EACH11	Every 11 PWM opportunities.
FLEXPWM_IP_LDFQ_EACH12	Every 12 PWM opportunities.
FLEXPWM_IP_LDFQ_EACH13	Every 13 PWM opportunities.
FLEXPWM_IP_LDFQ_EACH14	Every 14 PWM opportunities.
FLEXPWM_IP_LDFQ_EACH15	Every 15 PWM opportunities.
FLEXPWM_IP_LDFQ_EACH16	Every 16 PWM opportunities.

Definition at line 260 of file FlexPwm_Ip_Types.h.

7.3.4.15 FlexPwm_Ip_ForceSourceSelTypes

enum FlexPwm_Ip_ForceSourceSelTypes

Options that can trigger a PWM FORCE_OUT. Implements :FlexPwm_Ip_ForceSourceSelTypes.

Enumerator

FLEXPWM_IP_LOCAL_FORCE	The local force signal, CTRL2[FORCE], from this sub-module is used to force updates.
FLEXPWM_IP_MASTER_FORCE	The master force signal from sub-module 0 is used to force updates.
FLEXPWM_IP_LOCAL_RELOAD_FORCE	The local reload signal from this sub-module is used to force updates without regard to the state of LDOK.
FLEXPWM_IP_MASTER_RELOAD_FORCE	The master reload signal from sub-module 0 is used to force updates if LDOK is set.
FLEXPWM_IP_LOCAL_SYNC	The local sync signal from this sub-module is used to force updates.
FLEXPWM_IP_MASTER_SYNC	The master sync signal from submodule0 is used to force updates.
FLEXPWM_IP_EXT_FORCE	The external force signal, EXT_FORCE, from outside the PWM module causes updates.

Definition at line 284 of file FlexPwm_Ip_Types.h.

7.3.4.16 FlexPwm_Ip_ChannelPairTypes

enum FlexPwm_Ip_ChannelPairTypes

Options available for the PWM A & B pair operation. Implements : FlexPwm_Ip_ChannelPairTypes.

Enumerator

FLEXPWM_IP_COMPLEMENTARY	PWM A & PWM B are complementary channels.
FLEXPWM_IP_INDEPENDENT	PWM A & PWM B operation as 2 independent channels.

Definition at line 299 of file FlexPwm_Ip_Types.h.

7.3.4.17 FlexPwm_Ip_SrcCompSelTypes

enum FlexPwm_Ip_SrcCompSelTypes

Source selection for the generation of complementary PWM pair output. Implements : FlexPwm_Ip_SrcComp↩ SelTypes.

Enumerator

FLEXPWM_IP_COMP_SOURCE23	PWM23 is used as the source for the generation
FLEXPWM_IP_COMP_SOURCE45	PWM45 is used as the source for the generation

Definition at line 309 of file FlexPwm_Ip_Types.h.

7.3.4.18 FlexPwm_Ip_InitControlSelTypes

enum FlexPwm_Ip_InitControlSelTypes

PWM counter initialization options. Implements : FlexPwm_Ip_InitControlSelTypes.

Enumerator

FLEXPWM_IP_INIT_LOCAL_SYNC	Local sync (PWMX) causes initialization.
FLEXPWM_IP_INIT_MASTER_RELOAD	Master reload from sub-module 0 causes initialization.
FLEXPWM_IP_INIT_MASTER_SYNC	Master sync from sub-module 0 causes initialization.
FLEXPWM_IP_INIT_EXT_SYNC	EXT_SYNC causes initialization.

Definition at line 319 of file FlexPwm_Ip_Types.h.

7.3.4.19 FlexPwm_Ip_DeadTimeCountTypes

enum FlexPwm_Ip_DeadTimeCountTypes

FlexPWM dead time control available counter registers. Implements : FlexPwm_Ip_DeadTimeCountTypes.

Enumerator

FLEXPWM_IP_DEADTIME_COUNT_0	Abstract of DTCNT0.
FLEXPWM_IP_DEADTIME_COUNT_1	Abstract of DTCNT1.

Definition at line 331 of file FlexPwm_Ip_Types.h.

7.4 Pwm Driver

7.4.1 Detailed Description

Data Structures

- struct Pwm_ChannelConfigType

  Pwm channel high level configuration structure. *More...*
- struct Pwm_ConfigType

  Pwm high level configuration structure. *More...*

Macros

- #define PWM_DUTY_CYCLE_100

  100% duty cycle
- #define PWM_E_INIT_FAILED

  API Pwm_Init service called with wrong parameter.
- #define PWM_E_UNINIT

  API service used without module initialization.
- #define PWM_E_PARAM_CHANNEL

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

  Usage of unauthorized PWM service on PWM channel configured a fixed period.
- #define PWM_E_ALREADY_INITIALIZED

  API Pwm_Init service called while the PWM driver has already been initialized.
- #define PWM_E_PARAM_POINTER

  Generated when a NULL_PTR pointer is passed to Pwm_GetVersionInfo function.
- #define PWM_E_NOT_DISENGAGED

  Generated when Pwm_SetPowerState is called while the PWM module is still in use.
- #define PWM_E_POWER_STATE_NOT_SUPPORTED

  The requested power state is not supported by the PWM module.
- #define PWM_E_TRANSITION_NOT_POSSIBLE

  Generated The requested power state is not reachable from the current one.
- #define PWM_E_PERIPHERAL_NOT_PREPARED

  Generated when Pwm_SetPowerState has been called without having called the API Pwm_PreparePowerState before.
- #define PWM_E_PERIODVALUE

  Pwm_SetPeriodAndDuty called with invalid period range.
- #define PWM_E_PARAM_NOTIFICATION

  Invalid polarity selected for edge notification.
- #define PWM_E_PARAM_NOTIFICATION_NULL

  NULL_PTR function is configured as notification callback.
- #define PWM_E_DUTYCYCLE_RANGE

  Pwm_SetDutyCycle or Pwm_SetPeriodAndDuty called with invalid duty cycle range.
- #define PWM_E_COUNTERBUS

Generated when Pwm_SetCounterBus is called with an invalid Bus.

- #define PWM_E_CHANNEL_OFFSET_VALUE

  Generated when the configured offset for the OPWMB channel is more than the period of the associated MCB channel.
- #define PWM_E_OPWMB_CHANNEL_OFFSET_DUTYCYCLE_RANGE

  Generated when the requested offset value plus the current requested duty cycle leads to programming event B over the Period value leading to unexpected behavior of the PWM signal.
- #define PWM_E_PARAM_INSTANCE

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

  Generated when the configured trigger value for the OPWMT channel is equal or greater than the period of the channel.
- #define PWM_E_SET_CHANNEL_OUTPUT

  Generated when the output state value for the SetChannelOutput of the channel.
- #define PWM_E_UNEXPECTED_ISR

  Generated when an ISR has been triggered.
- #define PWM_E_PARAM_PHASESHIFT_RANGE

  Generated when requested phase shift value greater than 0x4000 (50%)
- #define PWM_E_CHANNEL_PHASE_SHIFT_NOT_SUPPORTED

  Generated when given channel does not support phase shift feature.
- #define PWM_E_DUTY_SYNCHRONOUS_NOT_SUPPORTED

  Generated when given channel does not support duty synchronous feature. (PHASE_SHIFTED_SYNCED and P↩ HASE_SHIFTED_COMPLEMENTARY)
- #define PWM_E_TRIGGER_MASK

  Generated when bit mask is not compatible with hardware register.
- #define PWM_E_FORCE_OUTPUT_NOT_SUPPORTED

  Generated when given channel does not support force output to zero feature.
- #define PWM_E_FORCE_OUT

  Generated when given channel does not support set force out feature.
- #define PWM_E_PARAM_CONFIG

  Generated when the requested resource is configured to be unavailable on the current core.
- #define PWM_E_DEADTIME_RANGE

  Generated when the configured dead time value is not valid.
- #define PWM_E_SETOUTPUTTOIDLE_NOT_SUPPORTED

  Generated when the configured dead time value is not valid.
- #define PWM_INIT_ID

  API service ID of Pwm_Init function.
- #define PWM_DEINIT_ID

  API service ID of Pwm_DeInit function.
- #define PWM_SETDUTYCYCLE_ID

  API service ID of Pwm_SetDutyCycle function.
- #define PWM_SETPERIODANDDUTY_ID

  API service ID of Pwm_SetPeriodAndDuty function.
- #define PWM_SETOUTPUTTOIDLE_ID

  API service ID of Pwm_SetOutputToIdle function.
- #define PWM_GETOUTPUTSTATE_ID

  API service ID of Pwm_GetOutputState function.
- #define PWM_DISABLENOTIFICATION_ID

API service ID of Pwm_DisableNotification function.

- #define PWM_ENABLENOTIFICATION_ID

  API service ID of Pwm_EnableNotification function.
- #define PWM_GETVERSIONINFO_ID

  API service ID of Pwm_GetVersionInfo function.
- #define PWM_SETPOWERSTATE_ID

  API service ID of Pwm_SetPowerState function.
- #define PWM_GETCURRENTPOWERSTATE_ID

  API service ID of Pwm_GetCurrentPowerState function.
- #define PWM_GETTARGETPOWERSTATE_ID

  API service ID of Pwm_GetTargetPowerState function.
- #define PWM_PREPAREPOWERSTATE_ID

  API service ID of Pwm_PreparePowerState function.
- #define PWM_MAIN_POWERTRANSITIONMANAGER_ID

  API service ID of Pwm_Main_PowerTransitionManager function.
- #define PWM_GETCHANNELSTATE_ID

  API service ID of Pwm_GetChannelState function.
- #define PWM_FORCEOUTPUTTOZERO_ID

  API service ID of Pwm_ForceOutputToZero function.
- #define PWM_SETCOUNTERBUS_ID

  API service ID of Pwm_SetCounterBus function.
- #define PWM_SETCHANNELOUTPUT_ID

  API service ID of Pwm_SetChannelOutput function.
- #define PWM_SETTRIGGERDELAY_ID

  API service ID of Pwm_SetTriggerDelay function.
- #define PWM_SETCLOCKMODE_ID

  API service ID of Pwm_SetClockMode function.
- #define PWM_SYNCUPDATE_ID

  API service ID of Pwm_SyncUpdate function.
- #define PWM_SETPERIODANDDUTY_NO_UPDATE_ID

  API service ID of Pwm_SetPeriodAndDuty_NoUpdate function.
- #define PWM_SETDUTYCYCLE_NO_UPDATE_ID

  API service ID of Pwm_SetDutyCycle_NoUpdate function.
- #define PWM_SETCHANNELDEADTIME_ID

  API service ID of Pwm_SetChannelDeadTime function.
- #define PWM_SETPHASESHIFT_ID

  API service ID of Pwm_SetPhaseShift function.
- #define PWM_SETPHASESHIFTNOUPDATE_ID

  API service ID of Pwm_SetPhaseShift function.
- #define PWM_ENABLETRIGGER_ID

  API service ID of Pwm_EnableTrigger function.
- #define PWM_DISABLETRIGGER_ID

  API service ID of Pwm_DisableTrigger function.
- #define PWM_RESETCOUNTERENABLE_ID

  API service ID of Pwm_ResetCounterEnable function.
- #define PWM_RESETCOUNTERDISABLE_ID

  API service ID of Pwm_ResetCounterDisable function.

- #define PWM_MASKOUTPUT_ID
  API service ID of Pwm_MaskOutputs function.
- #define PWM_UNMASKOUTPUT_ID
  API service ID of Pwm_UnMaskOutputs function.
- #define PWM_DISABLERELOADNOTIF_ID
  API service ID of Pwm_DisableReloadNotification function.
- #define PWM_ENABLERELOADNOTIF_ID
  API service ID of Pwm_EnableReloadNotification function.
- #define PWM_SETCHANNELFORCEOUT_ID
  API service ID of Pwm_SetChannelForceOut function.
- #define PWM_SETDUTYPHASESHIFT_ID
  API service ID of Pwm_SetDutyPhaseShift function.
- #define PWM_SETUCREGA_ID
  API service ID of Pwm_FastUpdateSetUCRegA function.
- #define PWM_SETUCREGB_ID
  API service ID of Pwm_FastUpdateSetUCRegB function.
- #define PWM_DISABLEOU_ID
  API service ID of Pwm_FastUpdateDisableOU function.
- #define PWM_ENABLEOU_ID
  API service ID of Pwm_FastUpdateEnableOU function.

Types Reference

- typedef uint8 Pwm_ChannelType
  PWM channel type.
- typedef uint8 Pwm_InstanceType
  PWM channel type.
- typedef Pwm_Ipw_PeriodType Pwm_PeriodType
  PWM period type.
- typedef Pwm_Ipw_DutyType Pwm_DutyType
  PWM duty type.
- typedef void(* Pwm_NotifyType) (void)
  Channel notification typedef.

Enum Reference

- enum Pwm_OutputStateType
  Output signal level.
- enum Pwm_EdgeNotificationType
  Edge notification type.
- enum Pwm_ChannelClassType
  PWM channel class type.
- enum Pwm_PowerStateType
  Power state type.
- enum Pwm_PowerStateRequestResultType
  Result of power state type.
- enum Pwm_PrescalerType
  Prescaler type.

Function Reference

- void Pwm_Init (const Pwm_ConfigType *ConfigPtr)

  This function initializes the Pwm driver.
- void Pwm_DeInit (void)

  This function deinitializes the Pwm driver.
- void Pwm_SetDutyCycle (Pwm_ChannelType ChannelNumber, uint16 DutyCycle)

  This function sets the dutycycle for the specified Pwm channel.
- void Pwm_SetPeriodAndDuty (Pwm_ChannelType ChannelNumber, Pwm_PeriodType Period, uint16 DutyCycle)

  This function sets the period and the dutycycle for the specified Pwm channel.
- void Pwm_SetOutputToIdle (Pwm_ChannelType ChannelNumber)

  This function sets the generated pwm signal to the idle value configured.
- Pwm_OutputStateType Pwm_GetOutputState (Pwm_ChannelType ChannelNumber)

  This function returns the signal output state.
- void Pwm_EnableNotification (Pwm_ChannelType ChannelNumber, Pwm_EdgeNotificationType Notification)

  This function enables the user notifications.
- void Pwm_DisableNotification (Pwm_ChannelType ChannelNumber)

  This function disables the user notifications.
- void Pwm_GetVersionInfo (Std_VersionInfoType *versioninfo)

  This function returns Pwm driver version details.
- uint16 Pwm_GetChannelState (Pwm_ChannelType ChannelNumber)

  This function returns the duty cycle of the channel passed as parameter.
- void Pwm_SetCounterBus (Pwm_ChannelType ChannelNumber, uint32 Bus)

  This function will change the bus of pwm channels running.
- void Pwm_SetChannelOutput (Pwm_ChannelType ChannelNumber, Pwm_StateType State)

  function to set the state of the PWM pin as requested for the current cycle
- void Pwm_SetTriggerDelay (Pwm_ChannelType ChannelNumber, Pwm_PeriodType TriggerDelay)

  Implementation specific function to change the trigger delay.
- void Pwm_SetClockMode (Pwm_PrescalerType Prescaler)

  Implementation specific function to change the peripheral clock frequency.
- void Pwm_SetChannelDeadTime (Pwm_ChannelType ChannelNumber, Pwm_PeriodType DeadTimeTicks)
- void Pwm_SetDutyCycle_NoUpdate (Pwm_ChannelType ChannelNumber, uint16 DutyCycle)

  This function sets the values of dutycycle for the specified Pwm channel but without updating the PWM output.
- void Pwm_SetPeriodAndDuty_NoUpdate (Pwm_ChannelType ChannelNumber, Pwm_PeriodType Period, uint16 DutyCycle)

  This function sets the values of the period and the dutycycle for the specified Pwm channel into the hardware buffers but without updating the PWM output..
- void Pwm_SetPhaseShift_NoUpdate (Pwm_ChannelType ChannelNumber, Pwm_PeriodType Period, uint16 PhaseShift)

  This function set phase shift value and also force duty cycle to 50%. The output will take effect after Pwm_Sync↩ Update be called.
- void Pwm_SyncUpdate (uint8 ModuleId)

  Implementation specific function to updates duty synchronization.
- void Pwm_SetPhaseShift (Pwm_ChannelType ChannelNumber, Pwm_PeriodType Period, uint16 Phase↩ Shift)

  This function set phase shift value and also force duty cycle to 50%.

- void Pwm_SetDutyPhaseShift (Pwm_ChannelType ChannelNumber, uint16 DutyCycle, Pwm_DutyType PhaseShift, boolean SyncUpdate)

  This function set phase shift and duty cycle value (as immediate or synchronized base on API parameter SyncUpdate)
- void Pwm_FastUpdateSetUCRegA (Pwm_ChannelType ChannelNumber, Pwm_PeriodType Value)

  This function shall be used to change duty cycle or phase shift with minimum overhead.
- void Pwm_FastUpdateSetUCRegB (Pwm_ChannelType ChannelNumber, Pwm_PeriodType Value)

  This function shall be used to change duty cycle, phase shift or inserted dead time buffer with minimum overhead.
- void Pwm_FastUpdateDisableOU (uint8 ModuleId, uint32 ChannelMask)

  This function shall be used to disable output update for selected Emios channels.
- void Pwm_FastUpdateEnableOU (uint8 ModuleId, uint32 ChannelMask)

  This function shall be used to enable output update for selected Emios channels.
- Std_ReturnType Pwm_SetPowerState (Pwm_PowerStateRequestResultType *Result)

  Enters the already prepared power state.
- Std_ReturnType Pwm_GetCurrentPowerState (Pwm_PowerStateType *CurrentPowerState, Pwm_PowerStateRequestRe *Result)

  Get the current power state of the Pwm HW unit.
- Std_ReturnType Pwm_GetTargetPowerState (Pwm_PowerStateType *TargetPowerState, Pwm_PowerStateRequestResu *Result)

  Get the target power state of the Pwm HW unit.
- Std_ReturnType Pwm_PreparePowerState (Pwm_PowerStateType PowerState, Pwm_PowerStateRequestResultType *Result)

  Starts the needed process to allow the Pwm HW module to enter the requested power state.

7.4.2 Data Structure Documentation

7.4.2.1 struct Pwm_ChannelConfigType

Pwm channel high level configuration structure.

Definition at line 749 of file Pwm.h.

Data Fields

- const Pwm_ChannelType ChannelId

  Id for the logical channel.
- const Pwm_ChannelClassType PwmChannelClass

  Channel class type: Variable/Fixed period.
- const Pwm_IpwChannelConfigType IpwChannelCfg

  The type of ip channel configured.
- const Pwm_OutputStateType ChannelIdleState

  The state of the channel output in idle mode.

7.4.2.1.1 Field Documentation

7.4.2.1.1.1 ChannelId const Pwm_ChannelType ChannelId

Id for the logical channel.

Definition at line 752 of file Pwm.h.

7.4.2.1.1.2 PwmChannelClass const Pwm_ChannelClassType PwmChannelClass

Channel class type: Variable/Fixed period.

Definition at line 754 of file Pwm.h.

7.4.2.1.1.3 IpwChannelCfg const Pwm_IpwChannelConfigType IpwChannelCfg

The type of ip channel configured.

Definition at line 756 of file Pwm.h.

7.4.2.1.1.4 ChannelIdleState const Pwm_OutputStateType ChannelIdleState

The state of the channel output in idle mode.

Definition at line 758 of file Pwm.h.

7.4.2.2 struct Pwm_ConfigType

Pwm high level configuration structure.

Definition at line 768 of file Pwm.h.

Data Fields

- const Pwm_ChannelType NumChannels
  Number of Pwm configured channels.
- const Pwm_ChannelConfigType(* PwmChannelsConfig)[]
  Pointer to the list of Pwm configured channels.
- const Pwm_ChannelType HwToLogicChannelMap [(80U)]
  Index table to translate HW channels to logical used to process interrupts for notifications.

7.4.2.2.1 Field Documentation

7.4.2.2.1.1 NumChannels `const Pwm_ChannelType NumChannels`

Number of Pwm configured channels.

Definition at line 775 of file Pwm.h.

7.4.2.2.1.2 PwmChannelsConfig `const Pwm_ChannelConfigType(* PwmChannelsConfig)[]`

Pointer to the list of Pwm configured channels.

Definition at line 777 of file Pwm.h.

7.4.2.2.1.3 HwToLogicChannelMap `const Pwm_ChannelType HwToLogicChannelMap[(80U)]`

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

Definition at line 790 of file Pwm.h.

7.4.3 Macro Definition Documentation

7.4.3.1 PWM_DUTY_CYCLE_100

```
#define PWM_DUTY_CYCLE_100
```

100% duty cycle

Errors and exceptions that will be detected by the PWM driver generated when Pwm_SetDutyCycle or Pwm_↩ SetPeriodAndDuty are called with a value for duty cycle out of valid range [0x0000, 0x8000]

Definition at line 144 of file Pwm.h.

7.4.3.2 PWM_E_INIT_FAILED

```
#define PWM_E_INIT_FAILED
```

API Pwm_Init service called with wrong parameter.

Errors and exceptions that will be detected by the PWM driver

Definition at line 151 of file Pwm.h.

7.4.3.3 PWM_E_UNINIT

```
#define PWM_E_UNINIT
```

API service used without module initialization.

Errors and exceptions that will be detected by the PWM driver

Definition at line 157 of file Pwm.h.

7.4.3.4 PWM_E_PARAM_CHANNEL

```
#define PWM_E_PARAM_CHANNEL
```

API service used with an invalid channel Identifier.

Errors and exceptions that will be detected by the PWM driver

Definition at line 163 of file Pwm.h.

7.4.3.5 PWM_E_PERIOD_UNCHANGEABLE

```
#define PWM_E_PERIOD_UNCHANGEABLE
```

Usage of unauthorized PWM service on PWM channel configured a fixed period.

Errors and exceptions that will be detected by the PWM driver

Definition at line 169 of file Pwm.h.

7.4.3.6 PWM_E_ALREADY_INITIALIZED

```
#define PWM_E_ALREADY_INITIALIZED
```

API Pwm_Init service called while the PWM driver has already been initialized.

Errors and exceptions that will be detected by the PWM driver

Definition at line 175 of file Pwm.h.

7.4.3.7 PWM_E_PARAM_POINTER

```
#define PWM_E_PARAM_POINTER
```

Generated when a NULL_PTR pointer is passed to Pwm_GetVersionInfo function.

Errors and exceptions that will be detected by the PWM driver

Definition at line 181 of file Pwm.h.

7.4.3.8 PWM_E_NOT_DISENGAGED

```
#define PWM_E_NOT_DISENGAGED
```

Generated when Pwm_SetPowerState is called while the PWM module is still in use.

Errors and exceptions that will be detected by the PWM driver

Definition at line 187 of file Pwm.h.

7.4.3.9 PWM_E_POWER_STATE_NOT_SUPPORTED

```
#define PWM_E_POWER_STATE_NOT_SUPPORTED
```

The requested power state is not supported by the PWM module.

Errors and exceptions that will be detected by the PWM driver

Definition at line 193 of file Pwm.h.

7.4.3.10 PWM_E_TRANSITION_NOT_POSSIBLE

```
#define PWM_E_TRANSITION_NOT_POSSIBLE
```

Generated The requested power state is not reachable from the current one.

Errors and exceptions that will be detected by the PWM driver

Definition at line 199 of file Pwm.h.

7.4.3.11 PWM_E_PERIPHERAL_NOT_PREPARED

```
#define PWM_E_PERIPHERAL_NOT_PREPARED
```

Generated when Pwm_SetPowerState has been called without having called the API Pwm_PreparePowerState before.

Errors and exceptions that will be detected by the PWM driver

Definition at line 206 of file Pwm.h.

7.4.3.12 PWM_E_PERIODVALUE

```
#define PWM_E_PERIODVALUE
```

Pwm_SetPeriodAndDuty called with invalid period range.

Generated when Pwm_SetPeriodAndDuty is called with a value for period out of valid range [0x0000, PWM_M↩ AX_PERIOD]

Definition at line 213 of file Pwm.h.

7.4.3.13 PWM_E_PARAM_NOTIFICATION

```
#define PWM_E_PARAM_NOTIFICATION
```

Invalid polarity selected for edge notification.

Will be generated when an invalid polarity, edge notification is requested for one PWM channel. Due to the limitations that are present in the eMIOS implementation not all the polarity notifications combinations can be supported.

Definition at line 221 of file Pwm.h.

7.4.3.14 PWM_E_PARAM_NOTIFICATION_NULL

```
#define PWM_E_PARAM_NOTIFICATION_NULL
```

NULL_PTR function is configured as notification callback.

Will be generated when a NULL_PTR function is configured as notification callback for one PWM channel and Pwm_EnableNotification is called for that channel

Definition at line 228 of file Pwm.h.

7.4.3.15 PWM_E_DUTYCYCLE_RANGE

```
#define PWM_E_DUTYCYCLE_RANGE
```

Pwm_SetDutyCycle or Pwm_SetPeriodAndDuty called with invalid duty cycle range.

Generated when Pwm_SetDutyCycle or Pwm_SetPeriodAndDuty are called with a value for duty cycle out of valid range [0x0000, 0x8000]

Definition at line 235 of file Pwm.h.

7.4.3.16 PWM_E_COUNTERBUS

```
#define PWM_E_COUNTERBUS
```

Generated when Pwm_SetCounterBus is called with an invalid Bus.

Errors and exceptions that will be detected by the PWM driver

Definition at line 241 of file Pwm.h.

7.4.3.17 PWM_E_CHANNEL_OFFSET_VALUE

```
#define PWM_E_CHANNEL_OFFSET_VALUE
```

Generated when the configured offset for the OPWMB channel is more than the period of the associated MCB channel.

Errors and exceptions that will be detected by the PWM driver

Definition at line 248 of file Pwm.h.

7.4.3.18 PWM_E_OPWMB_CHANNEL_OFFSET_DUTYCYCLE_RANGE

```
#define PWM_E_OPWMB_CHANNEL_OFFSET_DUTYCYCLE_RANGE
```

Generated when the requested offset value plus the current requested duty cycle leads to programming event B over the Period value leading to unexpected behavior of the PWM signal.

Errors and exceptions that will be detected by the PWM driver

Definition at line 255 of file Pwm.h.

7.4.3.19 PWM_E_PARAM_INSTANCE

```
#define PWM_E_PARAM_INSTANCE
```

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

Errors and exceptions that will be detected by the PWM driver

Definition at line 262 of file Pwm.h.

7.4.3.20 PWM_E_OPWMT_CHANNEL_TRIGGER_RANGE

```
#define PWM_E_OPWMT_CHANNEL_TRIGGER_RANGE
```

Generated when the configured trigger value for the OPWMT channel is equal or greater than the period of the channel.

Errors and exceptions that will be detected by the PWM driver

Definition at line 269 of file Pwm.h.

7.4.3.21 PWM_E_SET_CHANNEL_OUTPUT

```
#define PWM_E_SET_CHANNEL_OUTPUT
```

Generated when the output state value for the SetChannelOutput of the channel.

Errors and exceptions that will be detected by the PWM driver

Definition at line 276 of file Pwm.h.

7.4.3.22 PWM_E_UNEXPECTED_ISR

```
#define PWM_E_UNEXPECTED_ISR
```

Generated when an ISR has been triggered.

1. when the driver is not initialized
2. for a HW channel that is not used by any logic channel
3. for a logic channel that has no notification configured

Errors and exceptions that will be detected by the PWM driver

Definition at line 285 of file Pwm.h.

7.4.3.23 PWM_E_PARAM_PHASESHIFT_RANGE

```
#define PWM_E_PARAM_PHASESHIFT_RANGE
```

Generated when requested phase shift value greater than 0x4000 (50%)

Pwm_SetPhaseShift only works with Combine channel (COMBINED_SYNCED or COMBINED_COMPLEME↩ NTARY) Which do not support matching at next cycle. The duty cycle is always fixed value at 50%, so Phase Shift cannot greater than 50%

Definition at line 293 of file Pwm.h.

7.4.3.24 PWM_E_CHANNEL_PHASE_SHIFT_NOT_SUPPORTED

```
#define PWM_E_CHANNEL_PHASE_SHIFT_NOT_SUPPORTED
```

Generated when given channel does not support phase shift feature.

For FTM, only combine mode (COMBINED_SYNCED and COMBINED_COMPLEMENTARY) is supported. For eMIOS, only OPWMB, OPWM and OPWMT mode is supported.

Definition at line 300 of file Pwm.h.

7.4.3.25 PWM_E_DUTY_SYNCHRONOUS_NOT_SUPPORTED

```
#define PWM_E_DUTY_SYNCHRONOUS_NOT_SUPPORTED
```

Generated when given channel does not support duty synchronous feature. (PHASE_SHIFTED_SYNCED and PHASE_SHIFTED_COMPLEMENTARY)

For FTM, please note that Modified Combine mode does not support synchronous update Cn Therefore Pwm_↩ SetDutyCycle_NoUpdate and Pwm_SetPeriodAndDuty_NoUpdate should not be called in this case. For eMIOS, please note that channels using DAOC mode or channels in idle state does not support.

Definition at line 311 of file Pwm.h.

7.4.3.26 PWM_E_TRIGGER_MASK

```
#define PWM_E_TRIGGER_MASK
```

Generated when bit mask is not compatible with hardware register.

Definition at line 317 of file Pwm.h.

7.4.3.27 PWM_E_FORCE_OUTPUT_NOT_SUPPORTED

```
#define PWM_E_FORCE_OUTPUT_NOT_SUPPORTED
```

Generated when given channel does not support force output to zero feature.

Only channels in FlexPWM and FTM module are supported.

Definition at line 323 of file Pwm.h.

7.4.3.28 PWM_E_FORCE_OUT

```
#define PWM_E_FORCE_OUT
```

Generated when given channel does not support set force out feature.

Only channels in FlexPWM is supported.

Definition at line 329 of file Pwm.h.

7.4.3.29 PWM_E_PARAM_CONFIG

```
#define PWM_E_PARAM_CONFIG
```

Generated when the requested resource is configured to be unavailable on the current core.

Only multi-core configuration is available.

Definition at line 335 of file Pwm.h.

7.4.3.30 PWM_E_DEADTIME_RANGE

```
#define PWM_E_DEADTIME_RANGE
```

Generated when the configured dead time value is not valid.

Errors and exceptions that will be detected by the PWM driver

Definition at line 341 of file Pwm.h.

7.4.3.31 PWM_E_SETOUTPUTTOIDLE_NOT_SUPPORTED

```
#define PWM_E_SETOUTPUTTOIDLE_NOT_SUPPORTED
```

Generated when the configured dead time value is not valid.

Errors and exceptions that will be detected by the PWM driver

Definition at line 347 of file Pwm.h.

7.4.3.32 PWM_INIT_ID

```
#define PWM_INIT_ID
```

API service ID of Pwm_Init function.

Parameters used when raising an error/exception

Definition at line 353 of file Pwm.h.

7.4.3.33 PWM_DEINIT_ID

```
#define PWM_DEINIT_ID
```

API service ID of Pwm_DeInit function.

Parameters used when raising an error/exception

Definition at line 359 of file Pwm.h.

7.4.3.34 PWM_SETDUTYCYCLE_ID

```
#define PWM_SETDUTYCYCLE_ID
```

API service ID of Pwm_SetDutyCycle function.

Parameters used when raising an error/exception

Definition at line 365 of file Pwm.h.

7.4.3.35 PWM_SETPERIODANDDUTY_ID

```
#define PWM_SETPERIODANDDUTY_ID
```

API service ID of Pwm_SetPeriodAndDuty function.

Parameters used when raising an error/exception

Definition at line 371 of file Pwm.h.

7.4.3.36 PWM_SETOUTPUTTOIDLE_ID

```
#define PWM_SETOUTPUTTOIDLE_ID
```

API service ID of Pwm_SetOutputToIdle function.

Parameters used when raising an error/exception

Definition at line 377 of file Pwm.h.

7.4.3.37 PWM_GETOUTPUTSTATE_ID

```
#define PWM_GETOUTPUTSTATE_ID
```

API service ID of Pwm_GetOutputState function.

Parameters used when raising an error/exception

Definition at line 383 of file Pwm.h.

7.4.3.38 PWM_DISABLENOTIFICATION_ID

```
#define PWM_DISABLENOTIFICATION_ID
```

API service ID of Pwm_DisableNotification function.

Parameters used when raising an error/exception

Definition at line 389 of file Pwm.h.

7.4.3.39 PWM_ENABLENOTIFICATION_ID

```
#define PWM_ENABLENOTIFICATION_ID
```

API service ID of Pwm_EnableNotification function.

Parameters used when raising an error/exception

Definition at line 395 of file Pwm.h.

7.4.3.40 PWM_GETVERSIONINFO_ID

```
#define PWM_GETVERSIONINFO_ID
```

API service ID of Pwm_GetVersionInfo function.

Parameters used when raising an error/exception

Definition at line 401 of file Pwm.h.

7.4.3.41 PWM_SETPOWERSTATE_ID

```
#define PWM_SETPOWERSTATE_ID
```

API service ID of Pwm_SetPowerState function.

Parameters used when raising an error/exception

Definition at line 407 of file Pwm.h.

7.4.3.42 PWM_GETCURRENTPOWERSTATE_ID

```
#define PWM_GETCURRENTPOWERSTATE_ID
```

API service ID of Pwm_GetCurrentPowerState function.

Parameters used when raising an error/exception

Definition at line 413 of file Pwm.h.

7.4.3.43 PWM_GETTARGETPOWERSTATE_ID

```
#define PWM_GETTARGETPOWERSTATE_ID
```

API service ID of Pwm_GetTargetPowerState function.

Parameters used when raising an error/exception

Definition at line 419 of file Pwm.h.

7.4.3.44 PWM_PREPAREPOWERSTATE_ID

```
#define PWM_PREPAREPOWERSTATE_ID
```

API service ID of Pwm_PreparePowerState function.

Parameters used when raising an error/exception

Definition at line 425 of file Pwm.h.

7.4.3.45 PWM_MAIN_POWERTRANSITIONMANAGER_ID

```
#define PWM_MAIN_POWERTRANSITIONMANAGER_ID
```

API service ID of Pwm_Main_PowerTransitionManager function.

Parameters used when raising an error/exception

Definition at line 431 of file Pwm.h.

7.4.3.46 PWM_GETCHANNELSTATE_ID

```
#define PWM_GETCHANNELSTATE_ID
```

API service ID of Pwm_GetChannelState function.

Parameters used when raising an error/exception

Definition at line 437 of file Pwm.h.

7.4.3.47 PWM_FORCEOUTPUTTOZERO_ID

```
#define PWM_FORCEOUTPUTTOZERO_ID
```

API service ID of Pwm_ForceOutputToZero function.

Parameters used when raising an error/exception

Definition at line 443 of file Pwm.h.

7.4.3.48 PWM_SETCOUNTERBUS_ID

```
#define PWM_SETCOUNTERBUS_ID
```

API service ID of Pwm_SetCounterBus function.

Parameters used when raising an error/exception

Definition at line 449 of file Pwm.h.

7.4.3.49 PWM_SETCHANNELOUTPUT_ID

```
#define PWM_SETCHANNELOUTPUT_ID
```

API service ID of Pwm_SetChannelOutput function.

Parameters used when raising an error/exception

Definition at line 455 of file Pwm.h.

7.4.3.50 PWM_SETTRIGGERDELAY_ID

```
#define PWM_SETTRIGGERDELAY_ID
```

API service ID of Pwm_SetTriggerDelay function.

Parameters used when raising an error/exception

Definition at line 461 of file Pwm.h.

7.4.3.51 PWM_SETCLOCKMODE_ID

```
#define PWM_SETCLOCKMODE_ID
```

API service ID of Pwm_SetClockMode function.

Parameters used when raising an error/exception

Definition at line 467 of file Pwm.h.

7.4.3.52 PWM_SYNCUPDATE_ID

```
#define PWM_SYNCUPDATE_ID
```

API service ID of Pwm_SyncUpdate function.

Parameters used when raising an error/exception

Definition at line 473 of file Pwm.h.

7.4.3.53 PWM_SETPERIODANDDUTY_NO_UPDATE_ID

```
#define PWM_SETPERIODANDDUTY_NO_UPDATE_ID
```

API service ID of Pwm_SetPeriodAndDuty_NoUpdate function.

Parameters used when raising an error/exception

Definition at line 479 of file Pwm.h.

7.4.3.54 PWM_SETDUTYCYCLE_NO_UPDATE_ID

```
#define PWM_SETDUTYCYCLE_NO_UPDATE_ID
```

API service ID of Pwm_SetDutyCycle_NoUpdate function.

Parameters used when raising an error/exception

Definition at line 485 of file Pwm.h.

7.4.3.55 PWM_SETCHANNELDEADTIME_ID

```
#define PWM_SETCHANNELDEADTIME_ID
```

API service ID of Pwm_SetChannelDeadTime function.

Parameters used when raising an error/exception

Definition at line 491 of file Pwm.h.

7.4.3.56 PWM_SETPHASESHIFT_ID

```
#define PWM_SETPHASESHIFT_ID
```

API service ID of Pwm_SetPhaseShift function.

Parameters used when raising an error/exception

Definition at line 497 of file Pwm.h.

7.4.3.57 PWM_SETPHASESHIFTNOUPDATE_ID

```
#define PWM_SETPHASESHIFTNOUPDATE_ID
```

API service ID of Pwm_SetPhaseShift function.

Parameters used when raising an error/exception

Definition at line 503 of file Pwm.h.

7.4.3.58 PWM_ENABLETRIGGER_ID

```
#define PWM_ENABLETRIGGER_ID
```

API service ID of Pwm_EnableTrigger function.

Parameters used when raising an error/exception

Definition at line 509 of file Pwm.h.

7.4.3.59 PWM_DISABLETRIGGER_ID

```
#define PWM_DISABLETRIGGER_ID
```

API service ID of Pwm_DisableTrigger function.

Parameters used when raising an error/exception

Definition at line 515 of file Pwm.h.

7.4.3.60 PWM_RESETCOUNTERENABLE_ID

```
#define PWM_RESETCOUNTERENABLE_ID
```

API service ID of Pwm_ResetCounterEnable function.

Parameters used when raising an error/exception

Definition at line 521 of file Pwm.h.

7.4.3.61 PWM_RESETCOUNTERDISABLE_ID

```
#define PWM_RESETCOUNTERDISABLE_ID
```

API service ID of Pwm_ResetCounterDisable function.

Parameters used when raising an error/exception

Definition at line 527 of file Pwm.h.

7.4.3.62 PWM_MASKOUTPUT_ID

```
#define PWM_MASKOUTPUT_ID
```

API service ID of Pwm_MaskOutputs function.

Parameters used when raising an error/exception

Definition at line 533 of file Pwm.h.

7.4.3.63 PWM_UNMASKOUTPUT_ID

```
#define PWM_UNMASKOUTPUT_ID
```

API service ID of Pwm_UnMaskOutputs function.

Parameters used when raising an error/exception

Definition at line 539 of file Pwm.h.

7.4.3.64 PWM_DISABLERELOADNOTIF_ID

```
#define PWM_DISABLERELOADNOTIF_ID
```

API service ID of Pwm_DisableReloadNotification function.

Parameters used when raising an error/exception

Definition at line 545 of file Pwm.h.

7.4.3.65 PWM_ENABLERELOADNOTIF_ID

```
#define PWM_ENABLERELOADNOTIF_ID
```

API service ID of Pwm_EnableReloadNotification function.

Parameters used when raising an error/exception

Definition at line 551 of file Pwm.h.

7.4.3.66 PWM_SETCHANNELFORCEOUT_ID

```
#define PWM_SETCHANNELFORCEOUT_ID
```

API service ID of Pwm_SetChannelForceOut function.

Parameters used when raising an error/exception

Definition at line 557 of file Pwm.h.

7.4.3.67 PWM_SETDUTYPHASESHIFT_ID

```
#define PWM_SETDUTYPHASESHIFT_ID
```

API service ID of Pwm_SetDutyPhaseShift function.

Parameters used when raising an error/exception

Definition at line 563 of file Pwm.h.

7.4.3.68 PWM_SETUCREGA_ID

```
#define PWM_SETUCREGA_ID
```

API service ID of Pwm_FastUpdateSetUCRegA function.

Parameters used when raising an error/exception

Definition at line 569 of file Pwm.h.

7.4.3.69 PWM_SETUCREGB_ID

```
#define PWM_SETUCREGB_ID
```

API service ID of Pwm_FastUpdateSetUCRegB function.

Parameters used when raising an error/exception

Definition at line 575 of file Pwm.h.

7.4.3.70 PWM_DISABLEOU_ID

```
#define PWM_DISABLEOU_ID
```

API service ID of Pwm_FastUpdateDisableOU function.

Parameters used when raising an error/exception

Definition at line 581 of file Pwm.h.

7.4.3.71 PWM_ENABLEOU_ID

```
#define PWM_ENABLEOU_ID
```

API service ID of Pwm_FastUpdateEnableOU function.

Parameters used when raising an error/exception

Definition at line 587 of file Pwm.h.

7.4.4 Types Reference

7.4.4.1 Pwm_ChannelType

```
typedef uint8 Pwm_ChannelType
```

PWM channel type.

Definition at line 717 of file Pwm.h.

7.4.4.2 Pwm_InstanceType

```
typedef uint8 Pwm_InstanceType
```

PWM channel type.

Definition at line 723 of file Pwm.h.

7.4.4.3 Pwm_PeriodType

```
typedef Pwm_Ipw_PeriodType Pwm_PeriodType
```

PWM period type.

Definition at line 729 of file Pwm.h.

7.4.4.4 Pwm_DutyType

```
typedef Pwm_Ipw_DutyType Pwm_DutyType
```

PWM duty type.

Definition at line 735 of file Pwm.h.

7.4.4.5 Pwm_NotifyType

```
typedef void(* Pwm_NotifyType) (void)
```

Channel notification typedef.

Definition at line 742 of file Pwm.h.

7.4.5 Enum Reference

7.4.5.1 Pwm_OutputStateType

```
enum Pwm_OutputStateType
```

Output signal level.

This enumeration specifies the return type of Pwm_GetOutputState

Enumerator

PWM_HIGH	PWM level is logic high.
PWM_LOW	PWM level is logic low.

Definition at line 598 of file Pwm.h.

7.4.5.2 Pwm_EdgeNotificationType

```
enum Pwm_EdgeNotificationType
```

Edge notification type.

This enumeration defines the type of edge transition that can generate a notification

Enumerator

PWM_RISING_EDGE	A notification will be generated on the rising edge.
PWM_FALLING_EDGE	A notification will be generated on the falling edge.
PWM_BOTH_EDGES	A notification will be generated on any state transition.

Definition at line 612 of file Pwm.h.

7.4.5.3 Pwm_ChannelClassType

enum Pwm_ChannelClassType

PWM channel class type.

This field will specify what parameters can be altered for the selected channel

Enumerator

PWM_VARIABLE_PERIOD	The period and duty cycle can be altered.
PWM_FIXED_PERIOD	Only the duty cycle can be altered.
PWM_FIXED_PERIOD_SHIFTED	Only the duty cycle can be altered.

Definition at line 628 of file Pwm.h.

7.4.5.4 Pwm_PowerStateType

enum Pwm_PowerStateType

Power state type.

Power state currently active or set as target power state.

Enumerator

PWM_FULL_POWER	PWM full power mode.
PWM_LOW_POWER	PWM low power mode.
PWM_NODEFINE_POWER	PWM no define power mode.

Definition at line 644 of file Pwm.h.

7.4.5.5 Pwm_PowerStateRequestResultType

```
enum Pwm_PowerStateRequestResultType
```

Result of power state type.

Result of the requests related to power state transitions.

Enumerator

PWM_SERVICE_ACCEPTED	Power state change executed.
PWM_NOT_INIT	Module not initialized.
PWM_SEQUENCE_ERROR	Wrong API call sequence.
PWM_HW_FAILURE	The HW module has a failure which prevents it to enter the required power state.
PWM_POWER_STATE_NOT_SUPP	Module does not support the requested power state.
PWM_TRANS_NOT_POSSIBLE	Module cannot transition directly from the current power state to the requested power state.

Definition at line 660 of file Pwm.h.

7.4.5.6 Pwm_PrescalerType

```
enum Pwm_PrescalerType
```

Prescaler type.

This enumeration specifies the possible types of prescalers used to configure base-clock timers

Enumerator

PWM_PRIMARY_PRESCALER	Selected value is the default/primary prescaler.
PWM_ALTERNATIVE_PRESCALER	Selected value is the alternative configured prescaler.

Definition at line 681 of file Pwm.h.

7.4.6 Function Reference

7.4.6.1 Pwm_Init()

```
void Pwm_Init (
            const Pwm_ConfigType * ConfigPtr )
```

This function initializes the Pwm driver.

The function Pwm_Init shall initialize all internals variables and the used PWM structure of the microcontroller according to the parameters specified in configPtr. If the duty cycle parameter equals:

```
- 0% or 100% : Then the PWM output signal shall be in the state according
    to the configured polarity parameter;

- >0% and <100%: Then the PWM output signal shall be modulated according
    to parameters period, duty cycle and configured polarity.
```

The function Pwm_SetDutyCycle shall update the duty cycle always at the end of the period if supported by the implementation and configured with PwmDutycycleUpdatedEndperiod.

The driver shall avoid spikes on the PWM output signal when updating the PWM period and duty.

If development error detection for the Pwm module is enabled, the PWM functions shall check the channel class type and raise development error PWM_E_PERIOD_UNCHANGEABLE if the PWM channel is not declared as a variable period type.

If development error detection for the Pwm module is enabled, the PWM functions shall check the parameter channelNumber and raise development error PWM_E_PARAM_CHANNEL if the parameter channelNumber is invalid.

If development error detection for the Pwm module is enabled, when a development error occurs, the corresponding PWM 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).
- Return pwm level low for the function Pwm_GetOutputState.
```

The function Pwm_Init shall disable all notifications. The reason is that the users of these notifications may not be ready. They can call Pwm_EnableNotification to start notifications.

The function Pwm_Init shall only initialize the configured resources and shall not touch resources that are not configured in the configuration file.

If the PwmDevErorDetect switch is enabled, API parameter checking is enabled. The detailed description of the detected errors can be found in chapter Error classification and chapter API specification (see PWM_SWS).

If development error detection is enabled, calling the routine Pwm_Init while the PWM driver and hardware are already initialized will cause a development error PWM_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 Pwm module is enabled, if any function (except Pwm_Init) is called before Pwm_Init has been called, the called function shall raise development error PWM_E_UNINIT.

Parameters

in	*ConfigPtr*	Pointer to PWM top configuration structure

Returns

void

7.4.6.2 Pwm_DeInit()

```
void Pwm_DeInit (
            void  )
```

This function deinitializes the Pwm driver.

The function Pwm_DeInit shall deinitialize the PWM module.

```
The function Pwm_DeInit shall set the state of the PWM output signals
to the idle state.
The function Pwm_DeInit shall disable PWM interrupts and PWM signal edge notifications.
The function Pwm_DeInit shall be pre-compile time configurable On-Off by the
configuration parameter PwmDeInitApi function prototype.
If development error detection for the Pwm module is enabled,
when a development error occurs, the corresponding PWM 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).
    - Return pwm level low for the function Pwm_GetOutputState.

If development error detection for the Pwm module is enabled, if any function
(except Pwm_Init) is called before Pwm_Init has been called, the called function
shall raise development error PWM_E_UNINIT.
```

Returns

void

7.4.6.3 Pwm_SetDutyCycle()

```
void Pwm_SetDutyCycle (
            Pwm_ChannelType ChannelNumber,
            uint16 DutyCycle )
```

This function sets the dutycycle for the specified Pwm channel.

The function Pwm_SetDutyCycle shall set the duty cycle of the PWM channel.

```
The function Pwm_SetDutyCycle shall set the PWM output state according
to the configured polarity parameter, when the duty cycle = 0% or 100%.
The function Pwm_SetDutyCycle shall modulate the PWM output signal according
to parameters period, duty cycle and configured polarity,
when the duty cycle > 0 % and < 100%.

If development error detection for the Pwm module is enabled,
the PWM functions shall check the parameter channelNumber and raise development error
PWM_E_PARAM_CHANNEL if the parameter channelNumber is invalid.

If development error detection for the Pwm module is enabled,
when a development error occurs, the corresponding PWM 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).
    - Return pwm level low for the function Pwm_GetOutputState.

The Pwm module shall comply with the following scaling scheme for the duty cycle:
    - 0x0000 means 0%.
    - 0x8000 means 100%.
    - 0x8000 gives the highest resolution while allowing 100% duty cycle to be
        represented with a 16 bit value.
        As an implementation guide, the following source code example is given:
        AbsoluteDutyCycle = ((uint32)AbsolutePeriodTime * RelativeDutyCycle) >> 15;

If the PwmDevErorDetect switch is enabled, API parameter checking is enabled.
The detailed description of the detected errors can be found in chapter
Error classification and chapter API specification (see PWM_SWS).
```

Parameters

in	*ChannelNumber*	Pwm Channel Id in the configuration
in	*DutyCycle*	Pwm dutycycle value 0x0000 for 0% ... 0x8000 for 100%

Returns

void

7.4.6.4 Pwm_SetPeriodAndDuty()

```
void Pwm_SetPeriodAndDuty (
            Pwm_ChannelType ChannelNumber,
            Pwm_PeriodType Period,
            uint16 DutyCycle )
```

This function sets the period and the dutycycle for the specified Pwm channel.

The function Pwm_SetPeriodAndDuty shall set the duty cycle of the PWM channel.

```
If development error detection for the Pwm module is enabled, the PWM functions
shall check the channel class type and raise development error PWM_E_PERIOD_UNCHANGEABLE
if the PWM channel is not declared as a variable period type.

If development error detection for the Pwm module is enabled,
```

```
the PWM functions shall check the parameter channelNumber and raise development error
PWM_E_PARAM_CHANNEL if the parameter channelNumber is invalid.

If development error detection for the Pwm module is enabled,
when a development error occurs, the corresponding PWM 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).
    - Return pwm level low for the function Pwm_GetOutputState.

The Pwm module shall comply with the following scaling scheme for the duty cycle:
    - 0x0000 means 0%.
    - 0x8000 means 100%.
    - 0x8000 gives the highest resolution while allowing 100% duty cycle to be
        represented with a 16 bit value.
        As an implementation guide, the following source code example is given:
        AbsoluteDutyCycle = ((uint32)AbsolutePeriodTime * RelativeDutyCycle) >> 15;

If the PwmDevErorDetect switch is enabled, API parameter checking is enabled.
The detailed description of the detected errors can be found in chapter
Error classification and chapter API specification (see PWM_SWS).

If development error detection for the Pwm module is enabled, if any function
(except Pwm_Init) is called before Pwm_Init has been called, the called function shall
raise development error PWM_E_UNINIT.
```

Parameters

in	*ChannelNumber*	Pwm Channel Id in the configuration
in	*Period*	Pwm signal period value
in	*DutyCycle*	Pwm dutycycle value 0x0000 for 0% ... 0x8000 for 100%

Returns

void

7.4.6.5 Pwm_SetOutputToIdle()

```
void Pwm_SetOutputToIdle (
            Pwm_ChannelType ChannelNumber )
```

This function sets the generated pwm signal to the idle value configured.

The function Pwm_SetOutputToIdle shall set immediately the PWM output to the configured Idle state.

If development error detection for the Pwm module is enabled, the PWM functions shall check the parameter channelNumber and raise development error PWM_E_PARAM_CHANNEL if the parameter channelNumber is invalid.

If development error detection for the Pwm module is enabled, when a development error occurs, the corresponding PWM 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 the PwmDevErorDetect switch is enabled, API parameter checking is enabled. The detailed description of the detected errors can be found in chapter Error classification and chapter API specification (see PWM_SWS).

After the call of the function Pwm_SetOutputToIdle, variable period type channels shall be reactivated either using the Api Pwm_SetPeriodAndDuty() to activate the PWM channel with the new passed period or Api Pwm_SetDutyCycle() to activate the PWM channel with the old period.

After the call of the function Pwm_SetOutputToIdle, fixed period type channels shall be reactivated using only the API Api Pwm_SetDutyCycle() to activate the PWM channel with the old period.

If development error detection for the Pwm module is enabled, if any function (except Pwm_Init) is called before Pwm_Init has been called, the called function shall raise development error PWM_E_UNINIT.

Parameters

in	*ChannelNumber*	Pwm Channel Id in the configuration

Returns

void

7.4.6.6 Pwm_GetOutputState()

```
Pwm_OutputStateType Pwm_GetOutputState (
            Pwm_ChannelType ChannelNumber )
```

This function returns the signal output state.

The function Pwm_GetOutputState shall read the internal state of the PWM output signal and return it as defined in the diagram below (see PWM_SWS).

If development error detection for the Pwm module is enabled, the PWM functions shall check the parameter channelNumber and raise development error PWM_E_PARAM_CHANNEL if the parameter channelNumber is invalid.

If development error detection for the Pwm module is enabled, when a development error occurs, the corresponding PWM 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).

- Return pwm level low for the function Pwm_GetOutputState.

If the PwmDevErorDetect switch is enabled, API parameter checking is enabled. The detailed description of the detected errors can be found in chapter Error classification and chapter API specification (see PWM_SWS).

Due to real time constraint and setting of the PWM channel (project dependant), the output state can be modified just after the call of the service Pwm_GetOutputState.

If development error detection for the Pwm module is enabled, if any function (except Pwm_Init) is called before Pwm_Init has been called, the called function shall raise development error PWM_E_UNINIT.

Parameters

in	*ChannelNumber*	Pwm Channel Id in the configuration

Returns

Pwm_OutputStateType Pwm signal output logic value

Return values

PWM_LOW	The output state of PWM channel is low
PWM_HIGH	The output state of PWM channel is high

7.4.6.7 Pwm_EnableNotification()

```
void Pwm_EnableNotification (
             Pwm_ChannelType ChannelNumber,
             Pwm_EdgeNotificationType Notification )
```

This function enables the user notifications.

The function Pwm_EnableNotification shall enable the PWM signal edge notification according to notification parameter. If development error detection for the Pwm module is enabled:

- The PWM functions shall check the parameter channelNumber and raise development error PWM_E_PA↩ RAM_CHANNEL if the parameter channelNumber is invalid.

If development error detection for the Pwm module is enabled, when a development error occurs, the corresponding PWM 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).
- Return pwm level low for the function Pwm_GetOutputState.
```

If the PwmDevErorDetect switch is enabled, API parameter checking is enabled. The detailed description of the detected errors can be found in chapter Error classification and chapter API specification (see PWM_SWS).

If development error detection for the Pwm module is enabled, if any function (except Pwm_Init) is called before Pwm_Init has been called, the called function shall raise development error PWM_E_UNINIT.

Parameters

in	*ChannelNumber*	Pwm Channel Id in the configuration
in	*Notification*	Notification type to be enabled

Returns

void

7.4.6.8 Pwm_DisableNotification()

```
void Pwm_DisableNotification (
            Pwm_ChannelType ChannelNumber )
```

This function disables the user notifications.

If development error detection for the Pwm module is enabled:

- The PWM functions shall check the parameter channelNumber and raise development error PWM_E_PA↩ RAM_CHANNEL if the parameter channelNumber is invalid.

If development error detection for the Pwm module is enabled, when a development error occurs, the corresponding PWM 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).
- Return pwm level low for the function Pwm_GetOutputState.

If the PwmDevErorDetect switch is enabled, API parameter checking is enabled. The detailed description of the detected errors can be found in chapter Error classification and chapter API specification (see PWM_SWS).

All functions from the PWM module except Pwm_Init, Pwm_DeInit and Pwm_GetVersionInfo shall be re-entrant for different PWM channel numbers. In order to keep a simple module implementation, no check of PWM088 must be performed by the module. The function Pwm_DisableNotification shall be pre compile time configurable On-Off by the configuration parameter: PwmNotificationSupported.

If development error detection for the Pwm module is enabled, if any function (except Pwm_Init) is called before Pwm_Init has been called, the called function shall raise development error PWM_E_UNINIT.

Parameters

in	*ChannelNumber*	Pwm Channel Id in the configuration

Returns

void

7.4.6.9 Pwm_GetVersionInfo()

```
void Pwm_GetVersionInfo (
            Std_VersionInfoType * versioninfo )
```

This function returns Pwm driver version details.

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

Parameters

out	*versioninfo*	Pointer to Std_VersionInfoType output variable

Returns

void

7.4.6.10 Pwm_GetChannelState()

```
uint16 Pwm_GetChannelState (
            Pwm_ChannelType ChannelNumber )
```

This function returns the duty cycle of the channel passed as parameter.

The function Pwm_GetChannelState shall return the DutyCycle of the channel. In case the channel is idle, the returned value will be zero.

Parameters

in	*ChannelNumber*	Pwm Channel Id in the configuration

Returns

uint16 DutyCycle of the requested channel

7.4.6.11 Pwm_SetCounterBus()

```
void Pwm_SetCounterBus (
            Pwm_ChannelType ChannelNumber,
            uint32 Bus )
```

This function will change the bus of pwm channels running.

This function is useful to change the frequency of the output PWM signal between two counter buses frequency

Parameters

in	*ChannelNumber*	Pwm Channel Id in the configuration
in	*Bus*	The eMIOS bus will be selected to change

Returns

void

7.4.6.12 Pwm_SetChannelOutput()

```
void Pwm_SetChannelOutput (
            Pwm_ChannelType ChannelNumber,
            Pwm_StateType State )
```

function to set the state of the PWM pin as requested for the current cycle

This function is useful to set the state of the PWM pin as requested for the current cycle and continues with normal PWM operation from the next cycle

Parameters

in	*ChannelNumber*	Pwm Channel Id in the configuration
in	*State*	Active-Inactive state of the channel

Returns

void

7.4.6.13 Pwm_SetTriggerDelay()

```
void Pwm_SetTriggerDelay (
            Pwm_ChannelType ChannelNumber,
            Pwm_PeriodType TriggerDelay )
```

Implementation specific function to change the trigger delay.

This function is useful to set the trigger delay to opwmt mode. If no DET error reported then the trigger delay for the PWM channels will be set. If development error detection for the Pwm module is enabled:

- 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.

Parameters

in	*ChannelNumber*	Pwm Channel Id in the configuration
in	*TriggerDelay*	Trigger delay value will be updated

Returns

void

7.4.6.14 Pwm_SetClockMode()

```
void Pwm_SetClockMode (
            Pwm_PrescalerType Prescaler )
```

Implementation specific function to change the peripheral clock frequency.

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

Parameters

in	*Prescaler*	Prescaler type

Returns

void

7.4.6.15 Pwm_SetChannelDeadTime()

```
void Pwm_SetChannelDeadTime (
            Pwm_ChannelType ChannelNumber,
            Pwm_PeriodType DeadTimeTicks )
```

This function is used to update the deadtime at runtime for Pwm channels.

Parameters

in	*ChannelNumber*	Pwm channel id
in	*DeadTimeTicks*	Dead Time value in ticks

Returns

void

7.4.6.16 Pwm_SetDutyCycle_NoUpdate()

```
void Pwm_SetDutyCycle_NoUpdate (
            Pwm_ChannelType ChannelNumber,
            uint16 DutyCycle )
```

This function sets the values of dutycycle for the specified Pwm channel but without updating the PWM output.

The function Pwm_SetDutyCycle_NoUpdate shall set the duty cycle of the PWM channel to the coresponding hardware buffers without updating the wave form on the output pin. This feature will allow a pre-buffering of new PWM duty cycle values for several channel, which can all be updated synchronos by calling Pwm_SyncUpdate.

The function Pwm_SetDutyCycle_NoUpdate shall set the PWM output state according to the configured polarity parameter, when the duty cycle = 0% or 100%. The function Pwm_SetDutyCycle_NoUpdate shall modulate the PWM output signal according to parameters period, duty cycle and configured polarity, when the duty cycle > 0 % and < 100%.

If development error detection for the Pwm module is enabled, the PWM functions shall check the parameter channelNumber and raise development error PWM_E_PARAM_CHANNEL if the parameter channelNumber is invalid.

If development error detection for the Pwm module is enabled, when a development error occurs, the corresponding PWM 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).
- Return pwm level low for the function Pwm_GetOutputState.

The Pwm module shall comply with the following scaling scheme for the duty cycle:

- 0x0000 means 0%.
- 0x8000 means 100%.
- 0x8000 gives the highest resolution while allowing 100% duty cycle to be represented with a 16 bit value. As an implementation guide, the following source code example is given: AbsoluteDutyCycle = ((uint32)Absolute↩ PeriodTime ∗ RelativeDutyCycle) >> 15;

If the PwmDevErorDetect switch is enabled, API parameter checking is enabled. The detailed description of the detected errors can be found in chapter Error classification and chapter API specification (see PWM_SWS).

Parameters

in	*ChannelNumber*	Pwm Channel Id in the configuration
in	*DutyCycle*	Pwm dutycycle value 0x0000 for 0% ... 0x8000 for 100%

Returns

void

7.4.6.17 Pwm_SetPeriodAndDuty_NoUpdate()

```
void Pwm_SetPeriodAndDuty_NoUpdate (
            Pwm_ChannelType ChannelNumber,
            Pwm_PeriodType Period,
            uint16 DutyCycle )
```

This function sets the values of the period and the dutycycle for the specified Pwm channel into the hardware buffers but without updating the PWM output..

The function Pwm_SetPeriodAndDuty_NoUpdate shall set the period and duty cycle of the PWM channel to the coresponding hardware buffers without updating the wave form on the output pin. This feature will allow a pre-buffering of new PWM duty cycle values for several channel, which can all be updated synchronos by calling Pwm_SyncUpdate.

If development error detection for the Pwm module is enabled, the PWM functions shall check the channel class type and raise development error PWM_E_PERIOD_UNCHANGEABLE if the PWM channel is not declared as a variable period type.

If development error detection for the Pwm module is enabled, the PWM functions shall check the parameter channelNumber and raise development error PWM_E_PARAM_CHANNEL if the parameter channelNumber is invalid.

If development error detection for the Pwm module is enabled, when a development error occurs, the corresponding PWM 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).
- Return pwm level low for the function Pwm_GetOutputState.

The Pwm module shall comply with the following scaling scheme for the duty cycle:

- 0x0000 means 0%.
- 0x8000 means 100%.
- 0x8000 gives the highest resolution while allowing 100% duty cycle to be represented with a 16 bit value. As an implementation guide, the following source code example is given: AbsoluteDutyCycle = ((uint32)Absolute↩ PeriodTime ∗ RelativeDutyCycle) >> 15;

If the PwmDevErorDetect switch is enabled, API parameter checking is enabled. The detailed description of the detected errors can be found in chapter Error classification and chapter API specification (see PWM_SWS).

If development error detection for the Pwm module is enabled, if any function (except Pwm_Init) is called before Pwm_Init has been called, the called function shall raise development error PWM_E_UNINIT.

Parameters

in	*ChannelNumber*	Pwm Channel Id in the configuration
in	*Period*	Pwm signal period value
in	*DutyCycle*	Pwm dutycycle value 0x0000 for 0% ... 0x8000 for 100%

Returns

void

7.4.6.18 Pwm_SetPhaseShift_NoUpdate()

```
void Pwm_SetPhaseShift_NoUpdate (
            Pwm_ChannelType ChannelNumber,
            Pwm_PeriodType Period,
            uint16 PhaseShift )
```

This function set phase shift value and also force duty cycle to 50%. The output will take effect after Pwm_Sync↩ Update be called.

In order to have Phase-Shifted Full-Bridge controller, Pwm_SetPhaseShift/Pwm_SetPhaseShift_NoUpdate is introduced. Pwm_SetPhaseShift allows to set both phase shift value and period, the duty value is fixed to 50%.

Parameters

in	*ChannelNumber*	Pwm Channel Id in the configuration
in	*Period*	Pwm signal period value
in	*PhaseShift*	Phase shift value

Returns

void

7.4.6.19 Pwm_SyncUpdate()

```
void Pwm_SyncUpdate (
            uint8 ModuleId )
```

Implementation specific function to updates duty synchronization.

This function is used to update duty synchronization for channels in given module, this should be called after calling Pwm_SetPeriodAndDuty_NoUpdate() or Pwm_SetDutyCycle_NoUpdate() API.

Parameters

in	*Module↩ Id*	pwm module id(instance ID) Ex : PWM_EMIOS_INSTANCE_0 PWM_EMIOS_INSTANCE_1 ... PWM_FTM_INSTANCE_0

Returns

void

7.4.6.20 Pwm_SetPhaseShift()

```
void Pwm_SetPhaseShift (
            Pwm_ChannelType ChannelNumber,
            Pwm_PeriodType Period,
            uint16 PhaseShift )
```

This function set phase shift value and also force duty cycle to 50%.

In order to have Phase-Shifted Full-Bridge controller, Pwm_SetPhaseShift is introduced. Pwm_SetPhaseShift allows to set both phase shift value and period, the duty value is fixed to 50%.

Parameters

in	*ChannelNumber*	Pwm Channel Id in the configuration
in	*Period*	Pwm signal period value
in	*PhaseShift*	Phase shift value

Returns

void

7.4.6.21 Pwm_SetDutyPhaseShift()

```
void Pwm_SetDutyPhaseShift (
            Pwm_ChannelType ChannelNumber,
            uint16 DutyCycle,
            Pwm_DutyType PhaseShift,
            boolean SyncUpdate )
```

This function set phase shift and duty cycle value (as immediate or synchronized base on API parameter SyncUpdate)

Pwm_SetDutyPhaseShift allows to set both phase shift and duty cycle value, The phase shift is the offset of the leading edge of the signal in respect to period starting point.

Parameters

in	*ChannelNumber*	Pwm Channel Id in the configuration
in	*DutyCycle*	Pwm duty cycle value 0x0000 for 0% ... 0x8000 for 100%
in	*PhaseShift*	Phase shift value (in ticks)
in	*SyncUpdate*	Update duty and phases shift value synchronization for channels in given module or not TRUE Set the phase shift and duty cycle value base on the synchronization when calling Pwm_SyncUpdate. FALSE Set phase shift and duty cycle value immediately

Returns

void

7.4.6.22 Pwm_FastUpdateSetUCRegA()

```
void Pwm_FastUpdateSetUCRegA (
            Pwm_ChannelType ChannelNumber,
            Pwm_PeriodType Value )
```

This function shall be used to change duty cycle or phase shift with minimum overhead.

Fast update API is only supported for Emios Ip

Parameters

in	*ChannelNumber*	Pwm logic channel id
in	*Value*	Value to write in register

Returns

void

7.4.6.23 Pwm_FastUpdateSetUCRegB()

```
void Pwm_FastUpdateSetUCRegB (
            Pwm_ChannelType ChannelNumber,
            Pwm_PeriodType Value )
```

This function shall be used to change duty cycle, phase shift or inserted dead time buffer with minimum overhead.

Fast update API is only supported for Emios Ip

Parameters

in	*ChannelNumber*	Pwm logic channel id
in	*Value*	Value to write in register

Returns

void

7.4.6.24 Pwm_FastUpdateDisableOU()

```
void Pwm_FastUpdateDisableOU (
            uint8 ModuleId,
            uint32 ChannelMask )
```

This function shall be used to disable output update for selected Emios channels.

Fast update API is only supported for Emios Ip

Parameters

in	*ModuleId*	Pwm hardware module id
in	*ChannelMask*	Bit mask of Emios hardware channels

Returns

void

7.4.6.25 Pwm_FastUpdateEnableOU()

```
void Pwm_FastUpdateEnableOU (
            uint8 ModuleId,
            uint32 ChannelMask )
```

This function shall be used to enable output update for selected Emios channels.

Fast update API is only supported for Emios Ip

Parameters

in	*ModuleId*	Pwm hardware module id
in	*ChannelMask*	Bit mask of Emios hardware channels

Returns

void

7.4.6.26 Pwm_SetPowerState()

```
Std_ReturnType Pwm_SetPowerState (
            Pwm_PowerStateRequestResultType * Result )
```

Enters the already prepared power state.

This API configures the Pwm module so that it enters the already prepared power state, chosen between a predefined set of configured ones.

Parameters

out	*Result*	Pointer to a variable to store the result of this function

Returns

Std_ReturnType Standard return type.

Return values

E_OK	Power Mode changed.
E_NOT_OK	Request rejected.

7.4.6.27 Pwm_GetCurrentPowerState()

```
Std_ReturnType Pwm_GetCurrentPowerState (
            Pwm_PowerStateType * CurrentPowerState,
            Pwm_PowerStateRequestResultType * Result )
```

Get the current power state of the Pwm HW unit.

This API returns the current power state of the Pwm HW unit.

Parameters

out	*CurrentPowerState*	The current power mode of the Pwm HW Unit is returned in this parameter
out	*Result*	Pointer to a variable to store the result of this function

Returns

Std_ReturnType Standard return type.

Return values

E_OK	Mode could be read.
E_NOT_OK	Service is rejected.

7.4.6.28 Pwm_GetTargetPowerState()

```
Std_ReturnType Pwm_GetTargetPowerState (
            Pwm_PowerStateType * TargetPowerState,
            Pwm_PowerStateRequestResultType * Result )
```

Get the target power state of the Pwm HW unit.

This API returns the target power state of the Pwm HW unit.

Parameters

out	*TargetPowerState*	The Target power mode of the Pwm HW Unit is returned in this parameter.
out	*Result*	Pointer to a variable to store the result of this function.

Returns

Std_ReturnType Standard return type.

Return values

E_OK	Mode could be read.
E_NOT_OK	Service is rejected.

7.4.6.29 Pwm_PreparePowerState()

```
Std_ReturnType Pwm_PreparePowerState (
            Pwm_PowerStateType PowerState,
            Pwm_PowerStateRequestResultType * Result )
```

Starts the needed process to allow the Pwm HW module to enter the requested power state.

This API starts the needed process to allow the Pwm HW module to enter the requested power state.

Parameters

in	*PowerState*	The target power state intended to be attained.
out	*Result*	Pointer to a variable to store the result of this function.

Returns

Std_ReturnType Standard return type.

Return values

E_OK	Mode could be read.
E_NOT_OK	Service is rejected.

Chapter 8

Data Structure Documentation

8.1 Flexio_Pwm_Ip_ChannelConfigType Struct Reference

PWM configuration parameters structure.

`#include <Flexio_Pwm_Ip_Types.h>`

Data Fields

- uint8 TimerId
  Flexio used timer index.
- uint8 PinId
  Flexio used pin index.
- Flexio_Pwm_Ip_ClockPrescalerType Prescaler
  Counter decrement clock prescaler.
- uint16 Period
  Pwm period in ticks.
- uint16 DutyCycle
  Pwm duty cycle in ticks.
- Flexio_Pwm_Ip_PolarityType Polarity
  Pwm output polarity.
- Flexio_Pwm_Ip_InterruptType IrqMode
  Interrupt mode.
- Flexio_Pwm_Ip_IplNotificationType IplCallback
  User notification callback for IPL.
- Flexio_Pwm_Ip_HldNotificationType HldCallback
  User notification callback for HLD.

8.1.1 Detailed Description PWM configuration parameters structure.

Flexio IP specific channel configuration structure type

Definition at line 249 of file Flexio_Pwm_Ip_Types.h.

8.1.2 Field Documentation

8.1.2.1 TimerId

```
uint8 TimerId
```

Flexio used timer index.

Definition at line 252 of file Flexio_Pwm_Ip_Types.h.

8.1.2.2 PinId

```
uint8 PinId
```

Flexio used pin index.

Definition at line 254 of file Flexio_Pwm_Ip_Types.h.

8.1.2.3 Prescaler

```
Flexio_Pwm_Ip_ClockPrescalerType Prescaler
```

Counter decrement clock prescaler.

Definition at line 257 of file Flexio_Pwm_Ip_Types.h.

8.1.2.4 Period

```
uint16 Period
```

Pwm period in ticks.

Definition at line 260 of file Flexio_Pwm_Ip_Types.h.

8.1.2.5 DutyCycle

`uint16 DutyCycle`

Pwm duty cycle in ticks.

Definition at line 262 of file Flexio_Pwm_Ip_Types.h.

8.1.2.6 Polarity

`Flexio_Pwm_Ip_PolarityType Polarity`

Pwm output polarity.

Definition at line 265 of file Flexio_Pwm_Ip_Types.h.

8.1.2.7 IrqMode

`Flexio_Pwm_Ip_InterruptType IrqMode`

Interrupt mode.

Definition at line 268 of file Flexio_Pwm_Ip_Types.h.

8.1.2.8 IplCallback

`Flexio_Pwm_Ip_IplNotificationType IplCallback`

User notification callback for IPL.

Definition at line 270 of file Flexio_Pwm_Ip_Types.h.

8.1.2.9 HldCallback

`Flexio_Pwm_Ip_HldNotificationType HldCallback`

User notification callback for HLD.

Definition at line 272 of file Flexio_Pwm_Ip_Types.h.

8.2 Flexio_Pwm_Ip_HldNotificationType Struct Reference

Structure for notification.

`#include <Flexio_Pwm_Ip_Types.h>`

Data Fields

- Flexio_Pwm_Ip_HldCallbackType CbFunction
  Callback function pointer.
- uint8 CbParameter
  Callback function parameter pointer.

8.2.1 Detailed Description

Structure for notification.

The structure used to notification

Definition at line 235 of file Flexio_Pwm_Ip_Types.h.

8.2.2 Field Documentation

8.2.2.1 CbFunction

`Flexio_Pwm_Ip_HldCallbackType CbFunction`

Callback function pointer.

Definition at line 238 of file Flexio_Pwm_Ip_Types.h.

8.2.2.2 CbParameter

`uint8 CbParameter`

Callback function parameter pointer.

Definition at line 240 of file Flexio_Pwm_Ip_Types.h.

8.3 Flexio_Pwm_Ip_IplNotificationType Struct Reference

Structure for notification.

`#include <Flexio_Pwm_Ip_Types.h>`

Data Fields

- Flexio_Pwm_Ip_IplCallbackType CbFunction
  Callback function pointer.
- void * CbParameter
  Callback function parameter pointer.

8.3.1 Detailed Description Structure for notification.

The structure used to notification

Definition at line 221 of file Flexio_Pwm_Ip_Types.h.

8.3.2 Field Documentation

8.3.2.1 CbFunction

```
Flexio_Pwm_Ip_IplCallbackType CbFunction
```

Callback function pointer.

Definition at line 224 of file Flexio_Pwm_Ip_Types.h.

8.3.2.2 CbParameter

```
void* CbParameter
```

Callback function parameter pointer.

Definition at line 226 of file Flexio_Pwm_Ip_Types.h.

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

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

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