

---

# User Manual

for S32K14X PWM Driver

Document Number: UM2PWMA SR4.2 Rev0002 R1.0.2  
Rev. 1.0





# Contents

Section number	Title	Page
<b>Chapter 1</b>		
<b>Revision History</b>		
<b>Chapter 2</b>		
<b>Introduction</b>		
2.1	Supported Derivatives.....	19
2.2	Overview.....	19
2.3	About this Manual.....	20
2.4	Acronyms and Definitions.....	20
2.5	Reference List.....	21
<b>Chapter 3</b>		
<b>Driver</b>		
3.1	Requirements.....	23
3.2	Driver Design Summary.....	23
3.3	Hardware Resources.....	23
3.4	Deviation from Requirements.....	29
3.5	FlexIO specific implementation details.....	30
3.6	FTM specific implementation details.....	30
3.7	Driver limitations.....	31
3.8	Driver usage and configuration tips.....	32
3.8.1	Configure PwmHwConfiguration.....	32
3.8.2	Configure PwmModule.....	32
3.8.2.1	Configure PwmFtmModule.....	32
3.8.2.2	Configure PwmFlexIOModule.....	34
3.8.3	Configure logic channel in PwmChannel.....	36
3.8.4	Implement errata.....	37
3.9	Runtime Errors.....	37
3.10	Software specification.....	37
3.10.1	Define Reference.....	38

Section number	Title	Page
3.10.1.1	Define PWM_VENDOR_ID.....	38
3.10.1.2	Define PWM_MODULE_ID.....	38
3.10.1.3	Define PWM_AR_RELEASE_MAJOR_VERSION.....	38
3.10.1.4	Define PWM_AR_RELEASE_MINOR_VERSION.....	38
3.10.1.5	Define PWM_AR_RELEASE_REVISION_VERSION.....	39
3.10.1.6	Define PWM_SW_MAJOR_VERSION.....	39
3.10.1.7	Define PWM_SW_MINOR_VERSION.....	39
3.10.1.8	Define PWM_SW_PATCH_VERSION.....	39
3.10.1.9	Define PWM_DUTY_CYCLE_100.....	39
3.10.1.10	Define PWM_E_PARAM_CONFIG.....	40
3.10.1.11	Define PWM_E_UNINIT.....	40
3.10.1.12	Define PWM_E_PARAM_CHANNEL.....	40
3.10.1.13	Define PWM_E_PERIOD_UNCHANGEABLE.....	41
3.10.1.14	Define PWM_E_ALREADY_INITIALIZED.....	41
3.10.1.15	Define PWM_E_PARAM_POINTER.....	42
3.10.1.16	Define PWM_E_PARAM_NOTIFICATION.....	42
3.10.1.17	Define PWM_E_PARAM_NOTIFICATION_NULL.....	43
3.10.1.18	Define PWM_E_DUTYCYCLE_RANGE.....	43
3.10.1.19	Define PWM_E_COUNTERBUS.....	44
3.10.1.20	Define PWM_E_CHANNEL_OFFSET_VALUE.....	44
3.10.1.21	Define PWM_E_OPWMB_CHANNEL_OFFSET_DUTYCYCLE_RANGE.....	44
3.10.1.22	Define PWM_E_PARAM_INSTANCE.....	45
3.10.1.23	Define PWM_E_OPWMT_CHANNEL_TRIGGER_RANGE.....	45
3.10.1.24	Define PWM_E_OUTPUT_STATE.....	46
3.10.1.25	Define PWM_E_UNEXPECTED_ISR.....	46
3.10.1.26	Define PWM_E_PARAM_PHASESHIFT_RANGE.....	47
3.10.1.27	Define PWM_E_CHANNEL_PHASE_SHIFT_WITHOUT_COMBINE.....	47
3.10.1.28	Define PWM_E_PARAM_SYNCHRONOUS_MODIFIED_COMBINE.....	48
3.10.1.29	Define PWM_E_TRIGGER_MASK.....	48

Section number	Title	Page
3.10.1.30	Define PWM_E_NOT_DISENGAGED.....	49
3.10.1.31	Define PWM_E_POWER_STATE_NOT_SUPPORTED.....	49
3.10.1.32	Define PWM_E_TRANSITION_NOT_POSSIBLE.....	49
3.10.1.33	Define PWM_E_PERIPHERAL_NOT_PREPARED.....	50
3.10.1.34	Define PWM_INIT_ID.....	50
3.10.1.35	Define PWM_DEINIT_ID.....	50
3.10.1.36	Define PWM_SETDUTYCYCLE_ID.....	51
3.10.1.37	Define PWM_SETPERIODANDDUTY_ID.....	51
3.10.1.38	Define PWM_SETOUTPUTTOIDLE_ID.....	52
3.10.1.39	Define PWM_GETOUTPUTSTATE_ID.....	52
3.10.1.40	Define PWM_DISABLENOTIFICATION_ID.....	52
3.10.1.41	Define PWM_ENABLENOTIFICATION_ID.....	53
3.10.1.42	Define PWM_GETVERSIONINFO_ID.....	53
3.10.1.43	Define PWM_SETPOWERSTATE_ID.....	53
3.10.1.44	Define PWM_GETCURRENTPOWERSTATE_ID.....	54
3.10.1.45	Define PWM_GETTARGETPOWERSTATE_ID.....	54
3.10.1.46	Define PWM_PREPAREPOWERSTATE_ID.....	54
3.10.1.47	Define PWM_GETCHANNELSTATE_ID.....	55
3.10.1.48	Define PWM_FORCEOUTPUTTOZERO_ID.....	55
3.10.1.49	Define PWM_SETCOUNTERBUS_ID.....	55
3.10.1.50	Define PWM_SETCHANNELOUTPUT_ID.....	56
3.10.1.51	Define PWM_SETTRIGGERDELAY_ID.....	56
3.10.1.52	Define PWM_BUFFERTRANSFERENDIS_ID.....	57
3.10.1.53	Define PWM_SETCLOCKMODE_ID.....	57
3.10.1.54	Define PWM_SYNCUPDATE_ID.....	57
3.10.1.55	Define PWM_SETPERIODANDDUTY_NO_UPDATE_ID.....	58
3.10.1.56	Define PWM_SETDUTYCYCLE_NO_UPDATE_ID.....	58
3.10.1.57	Define PWM_SETPHASESHIFT_ID.....	58
3.10.1.58	Define PWM_SETPHASESHIFTNOUPDATE_ID.....	59

Section number	Title	Page
3.10.1.59	Define PWM_ENABLETRIGGER_ID.....	59
3.10.1.60	Define PWM_DISABLETRIGGER_ID.....	59
3.10.1.61	Define PWM_RESETCOUNTERENABLE_ID.....	60
3.10.1.62	Define PWM_RESETCOUNTERDISABLE_ID.....	60
3.10.1.63	Define PWM_MASKOUTPUT_ID.....	61
3.10.1.64	Define PWM_UNMASKOUTPUT_ID.....	61
3.10.1.65	Define PWM_DISABLERELOADNOTIF_ID.....	61
3.10.1.66	Define PWM_ENABLERELOADNOTIF_ID.....	62
3.10.1.67	Define PWM_FLEXIO_USED.....	62
3.10.1.68	Define PWM_FTM_USED.....	62
3.10.1.69	Define PWM_FLEXIO_0_CH_0_1_USED.....	63
3.10.1.70	Define PWM_FLEXIO_0_CH_2_3_USED.....	63
3.10.1.71	Define PWM_FLEXIO_0_CH_0_1_ISR_USED.....	63
3.10.1.72	Define PWM_FLEXIO_0_CH_2_3_ISR_USED.....	63
3.10.1.73	Define PWM_FLEXIO_0_PIN_0_USED.....	64
3.10.1.74	Define PWM_FLEXIO_0_PIN_1_USED.....	64
3.10.1.75	Define PWM_FLEXIO_0_PIN_2_USED.....	64
3.10.1.76	Define PWM_FLEXIO_0_PIN_3_USED.....	64
3.10.1.77	Define PWM_FLEXIO_0_PIN_4_USED.....	65
3.10.1.78	Define PWM_FLEXIO_0_PIN_5_USED.....	65
3.10.1.79	Define PWM_FLEXIO_0_PIN_6_USED.....	65
3.10.1.80	Define PWM_FLEXIO_0_PIN_7_USED.....	65
3.10.1.81	Define PWM_FTM_0_USED.....	66
3.10.1.82	Define PWM_FTM_1_USED.....	66
3.10.1.83	Define PWM_FTM_2_USED.....	66
3.10.1.84	Define PWM_FTM_3_USED.....	66
3.10.1.85	Define PWM_FTM_4_USED.....	66
3.10.1.86	Define PWM_FTM_5_USED.....	67
3.10.1.87	Define PWM_FTM_6_USED.....	67

Section number	Title	Page
3.10.1.88	Define PWM_FTM_7_USED.....	67
3.10.1.89	Define PWM_FTM_0_CH_0_CH_1_ISR_USED.....	67
3.10.1.90	Define PWM_FTM_0_CH_2_CH_3_ISR_USED.....	68
3.10.1.91	Define PWM_FTM_0_CH_4_CH_5_ISR_USED.....	68
3.10.1.92	Define PWM_FTM_0_CH_6_CH_7_ISR_USED.....	68
3.10.1.93	Define PWM_FTM_0_OVF_ISR_USED.....	68
3.10.1.94	Define PWM_FTM_0_FAULT_ISR_USED.....	69
3.10.1.95	Define PWM_FTM_1_CH_0_CH_1_ISR_USED.....	69
3.10.1.96	Define PWM_FTM_1_CH_2_CH_3_ISR_USED.....	69
3.10.1.97	Define PWM_FTM_1_CH_4_CH_5_ISR_USED.....	69
3.10.1.98	Define PWM_FTM_1_CH_6_CH_7_ISR_USED.....	70
3.10.1.99	Define PWM_FTM_1_OVF_ISR_USED.....	70
3.10.1.100	Define PWM_FTM_1_FAULT_ISR_USED.....	70
3.10.1.101	Define PWM_FTM_2_CH_0_CH_1_ISR_USED.....	70
3.10.1.102	Define PWM_FTM_2_CH_2_CH_3_ISR_USED.....	71
3.10.1.103	Define PWM_FTM_2_CH_4_CH_5_ISR_USED.....	71
3.10.1.104	Define PWM_FTM_2_CH_6_CH_7_ISR_USED.....	71
3.10.1.105	Define PWM_FTM_2_OVF_ISR_USED.....	71
3.10.1.106	Define PWM_FTM_2_FAULT_ISR_USED.....	72
3.10.1.107	Define PWM_FTM_3_CH_0_CH_1_ISR_USED.....	72
3.10.1.108	Define PWM_FTM_3_CH_2_CH_3_ISR_USED.....	72
3.10.1.109	Define PWM_FTM_3_CH_4_CH_5_ISR_USED.....	72
3.10.1.110	Define PWM_FTM_3_CH_6_CH_7_ISR_USED.....	73
3.10.1.111	Define PWM_FTM_3_OVF_ISR_USED.....	73
3.10.1.112	Define PWM_FTM_3_FAULT_ISR_USED.....	73
3.10.1.113	Define PWM_FTM_4_CH_0_CH_1_ISR_USED.....	73
3.10.1.114	Define PWM_FTM_4_CH_2_CH_3_ISR_USED.....	74
3.10.1.115	Define PWM_FTM_4_CH_4_CH_5_ISR_USED.....	74
3.10.1.116	Define PWM_FTM_4_CH_6_CH_7_ISR_USED.....	74

Section number	Title	Page
3.10.1.117	Define PWM_FTM_4_OVF_ISR_USED.....	74
3.10.1.118	Define PWM_FTM_4_FAULT_ISR_USED.....	75
3.10.1.119	Define PWM_FTM_5_CH_0_CH_1_ISR_USED.....	75
3.10.1.120	Define PWM_FTM_5_CH_2_CH_3_ISR_USED.....	75
3.10.1.121	Define PWM_FTM_5_CH_4_CH_5_ISR_USED.....	75
3.10.1.122	Define PWM_FTM_5_CH_6_CH_7_ISR_USED.....	76
3.10.1.123	Define PWM_FTM_5_OVF_ISR_USED.....	76
3.10.1.124	Define PWM_FTM_5_FAULT_ISR_USED.....	76
3.10.1.125	Define PWM_FTM_6_CH_0_CH_1_ISR_USED.....	76
3.10.1.126	Define PWM_FTM_6_CH_2_CH_3_ISR_USED.....	77
3.10.1.127	Define PWM_FTM_6_CH_4_CH_5_ISR_USED.....	77
3.10.1.128	Define PWM_FTM_6_CH_6_CH_7_ISR_USED.....	77
3.10.1.129	Define PWM_FTM_6_OVF_ISR_USED.....	77
3.10.1.130	Define PWM_FTM_6_FAULT_ISR_USED.....	78
3.10.1.131	Define PWM_FTM_7_CH_0_CH_1_ISR_USED.....	78
3.10.1.132	Define PWM_FTM_7_CH_2_CH_3_ISR_USED.....	78
3.10.1.133	Define PWM_FTM_7_CH_4_CH_5_ISR_USED.....	78
3.10.1.134	Define PWM_FTM_7_CH_6_CH_7_ISR_USED.....	79
3.10.1.135	Define PWM_FTM_7_OVF_ISR_USED.....	79
3.10.1.136	Define PWM_FTM_7_FAULT_ISR_USED.....	79
3.10.1.137	Define PWM_FTM_CHANNEL.....	79
3.10.1.138	Define PWM_FLEXIO_CHANNEL.....	80
3.10.1.139	Define PWM_FTM_CHANNELS_NO.....	80
3.10.1.140	Define PWM_FTM_CHANNELS_MAX_U8.....	80
3.10.1.141	Define PWM_FTM_MODULE_NO.....	80
3.10.1.142	Define PWM_FLEXIO_MODULE_NO.....	81
3.10.1.143	Define PWM_FLEXIO_CHANNEL_NO.....	81
3.10.1.144	Define PWM_FLEXIO_CHANNELS_MAX_U8.....	81
3.10.1.145	Define PWM_HW_CHANNELS_NO_U8.....	81



Section number	Title	Page
3.10.1.146	Define PWM_FTM_MODULE_CHANNELS_NO.....	82
3.10.1.147	Define PWM_FTM_MODULE_FAULT_NO.....	82
3.10.1.148	Define PWM_DEV_ERROR_DETECT.....	82
3.10.1.149	Define PWM_DUTY_PERIOD_UPDATED_ENDPERIOD.....	82
3.10.1.150	Define PWM_DUTYCYCLE_UPDATED_ENDPERIOD.....	83
3.10.1.151	Define PWM_FAULT_SUPPORTED.....	83
3.10.1.152	Define PWM_INDEX.....	83
3.10.1.153	Define PWM_NOTIFICATION_SUPPORTED.....	84
3.10.1.154	Define PWM_PRECOMPILE_SUPPORT.....	84
3.10.1.155	Define PWM_FTM_ENABLE_EXT_TRIGGERS.....	84
3.10.1.156	Define PWM_SET_DUTY_CYCLE_API.....	84
3.10.1.157	Define PWM_SET_OUTPUT_TO_IDLE_API.....	85
3.10.1.158	Define PWM_SET_PERIOD_AND_DUTY_API.....	85
3.10.1.159	Define PWM_SET_CLOCK_MODE_API.....	85
3.10.1.160	Define PWM_VERSION_INFO_API.....	85
3.10.1.161	Define PWM_GET_CHANNEL_STATE_API.....	86
3.10.1.162	Define PWM_GET_OUTPUT_STATE_API.....	86
3.10.1.163	Define PWM_FORCE_OUTPUT_TO_ZERO_API.....	86
3.10.1.164	Define PWM_DE_INIT_API.....	87
3.10.1.165	Define PWM_SET_PHASE_SHIFT_API.....	87
3.10.1.166	Define PWM_SET_PHASE_SHIFT_NO_UPDATE_API.....	87
3.10.1.167	Define PWM_ENABLE_TRIGEER_API.....	87
3.10.1.168	Define PWM_DIABLE_TRIGEER_API.....	88
3.10.1.169	Define PWM_RESET_COUNTER_API.....	88
3.10.1.170	Define PWM_ENABLE_MASKING_OPERATIONS.....	88
3.10.1.171	Define PWM_SYNC_UPDATE_API.....	88
3.10.1.172	Define PWM_UPDATE_DUTY_SYNCHRONOUS.....	89
3.10.1.173	Define PWM_SET_DUTY_CYCLE_NO_UPDATE_API.....	89
3.10.1.174	Define PWM_SET_PERIOD_AND_DUTY_NO_UPDATE_API.....	89

Section number	Title	Page
3.10.1.175	Define PWM_ENABLE_PHASE_SHIFT.....	89
3.10.2	Enum Reference.....	90
3.10.2.1	Enumeration Pwm_ChannelClassType.....	90
3.10.2.2	Enumeration Pwm_OutputStateType.....	90
3.10.2.3	Enumeration Pwm_EdgeNotificationType.....	91
3.10.2.4	Enumeration Pwm_PrescalerType.....	91
3.10.2.5	Enumeration Pwm_GlobalStateType.....	92
3.10.2.6	Enumeration Pwm_AlignmentType.....	92
3.10.2.7	Enumeration Pwm_PowerStateRequestResultType.....	93
3.10.2.8	Enumeration Pwm_PowerStateType.....	93
3.10.2.9	Enumeration Pwm_DataUpdateType.....	94
3.10.3	Structs Reference.....	94
3.10.3.1	Structure Pwm_FlexIO_ChannelConfigType.....	94
3.10.3.2	Structure Pwm_FlexIO_IpConfigType.....	95
3.10.3.3	Structure Pwm_Ftm_ChannelConfigType.....	96
3.10.3.4	Structure Pwm_Ftm_ModuleConfigType.....	97
3.10.3.5	Structure Pwm_Ftm_IpConfigType.....	98
3.10.3.6	Structure Pwm_IpConfigType.....	99
3.10.3.7	Structure Pwm_IpChannelConfigType.....	99
3.10.3.8	Structure Pwm_ChannelConfigType.....	100
3.10.3.9	Structure Pwm_ConfigType.....	101
3.10.4	Types Reference.....	102
3.10.4.1	Typedef Pwm_FlexIO_ChannelType.....	103
3.10.4.2	Typedef Pwm_FlexIO_TimerType.....	103
3.10.4.3	Typedef Pwm_NotifyType.....	103
3.10.4.4	Typedef Pwm_ChannelType.....	103
3.10.4.5	Typedef Pwm_PeriodType.....	103
3.10.4.6	Typedef Pwm_ChannelIpType.....	104
3.10.5	Function Reference.....	104

Section number	Title	Page
3.10.5.1	Function Pwm_Init.....	104
3.10.5.2	Function Pwm_DeInit.....	106
3.10.5.3	Function Pwm_SetPeriodAndDuty.....	106
3.10.5.4	Function Pwm_SetDutyCycle.....	108
3.10.5.5	Function Pwm_SetOutputToIdle.....	109
3.10.5.6	Function Pwm_DisableNotification.....	110
3.10.5.7	Function Pwm_EnableNotification.....	111
3.10.5.8	Function Pwm_GetChannelState.....	112
3.10.5.9	Function Pwm_GetOutputState.....	113
3.10.5.10	Function Pwm_ForceOutputToZero.....	114
3.10.5.11	Function Pwm_SetClockMode.....	114
3.10.5.12	Function Pwm_GetVersionInfo.....	115
3.10.5.13	Function Pwm_DisableTrigger.....	115
3.10.5.14	Function Pwm_EnableTrigger.....	117
3.10.5.15	Function Pwm_MaskOutputs.....	118
3.10.5.16	Function Pwm_UnMaskOutputs.....	119
3.10.5.17	Function Pwm_ResetCounter.....	120
3.10.5.18	Function Pwm_ResetCounterEnable.....	121
3.10.5.19	Function Pwm_ResetCounterDisable.....	121
3.10.5.20	Function Pwm_SetDutyCycle_NoUpdate.....	121
3.10.5.21	Function Pwm_SetPeriodAndDuty_NoUpdate.....	123
3.10.5.22	Function Pwm_SetPhaseShift.....	124
3.10.5.23	Function Pwm_SetPhaseShift_NoUpdate.....	126
3.10.5.24	Function Pwm_SyncUpdate.....	127
3.10.5.25	Function Pwm_SetPowerState.....	127
3.10.5.26	Function Pwm_GetCurrentPowerState.....	128
3.10.5.27	Function Pwm_GetTargetPowerState.....	129
3.10.5.28	Function Pwm_PreparePowerState.....	130
3.10.6	Variables Reference.....	131

Section number	Title	Page
3.11	Symbolic Names Disclaimer.....	131

## Chapter 4

### Tresos Configuration Plug-in

4.1	Configuration elements of PWM.....	133
4.2	Form IMPLEMENTATION_CONFIG_VARIANT.....	133
4.3	Form PwmConfigurationOfOptApiServices.....	134
4.3.1	PwmDeInitApi (PwmConfigurationOfOptApiServices).....	134
4.3.2	PwmGetOutputState (PwmConfigurationOfOptApiServices).....	134
4.3.3	PwmSetDutyCycle (PwmConfigurationOfOptApiServices).....	135
4.3.4	PwmSetOutputToIdle (PwmConfigurationOfOptApiServices).....	135
4.3.5	PwmSetPeriodAndDuty (PwmConfigurationOfOptApiServices).....	135
4.3.6	PwmVersionInfoApi (PwmConfigurationOfOptApiServices).....	136
4.3.7	PwmGetChannelStateApi (PwmConfigurationOfOptApiServices).....	136
4.3.8	PwmForceOutputToZeroApi (PwmConfigurationOfOptApiServices).....	137
4.3.9	PwmSetDutyCycle_NoUpdate (PwmConfigurationOfOptApiServices).....	137
4.3.10	PwmSetPeriodAndDuty_NoUpdate (PwmConfigurationOfOptApiServices).....	137
4.3.11	PwmSetPhaseShift (PwmConfigurationOfOptApiServices).....	138
4.3.12	PwmSetPhaseShiftNoUpdate (PwmConfigurationOfOptApiServices).....	138
4.3.13	PwmEnableTrigger (PwmConfigurationOfOptApiServices).....	138
4.3.14	PwmDiableTrigger (PwmConfigurationOfOptApiServices).....	139
4.3.15	PwmSwResetCounter (PwmConfigurationOfOptApiServices).....	139
4.4	Form PwmGeneral.....	139
4.4.1	PwmDevErrorDetect (PwmGeneral).....	140
4.4.2	PwmDutycycleUpdatedEndperiod (PwmGeneral).....	140
4.4.3	PwmNotificationSupported (PwmGeneral).....	141
4.4.4	PwmEnableDualClockMode (PwmGeneral).....	141
4.4.5	PwmPeriodUpdatedEndperiod (PwmGeneral).....	141
4.4.6	PwmIndex (PwmGeneral).....	142
4.4.7	PwmFaultSupport (PwmGeneral).....	142

Section number	Title	Page
4.4.8	PwmFtmEnableExtTrigger (PwmGeneral).....	143
4.4.9	PwmEnablePhaseShift (PwmGeneral).....	143
4.4.10	PwmMultiChannelSynch (PwmGeneral).....	143
4.4.11	EnableMaskingOperations (PwmGeneral).....	144
4.4.12	PwmLowPowerStatesSupport (PwmGeneral).....	144
4.4.13	PwmPowerStateAsynchTransitionMode (PwmGeneral).....	144
4.5	Form CommonPublishedInformation.....	145
4.5.1	ArReleaseMajorVersion (CommonPublishedInformation).....	145
4.5.2	ArReleaseMinorVersion (CommonPublishedInformation).....	146
4.5.3	ArReleaseRevisionVersion (CommonPublishedInformation).....	146
4.5.4	ModuleId (CommonPublishedInformation).....	147
4.5.5	SwMajorVersion (CommonPublishedInformation).....	147
4.5.6	SwMinorVersion (CommonPublishedInformation).....	147
4.5.7	SwPatchVersion (CommonPublishedInformation).....	148
4.5.8	VendorApiInfix (CommonPublishedInformation).....	148
4.5.9	VendorId (CommonPublishedInformation).....	149
4.6	Form PwmChannelConfigSet.....	149
4.6.1	Form PwmChannel.....	150
4.6.1.1	PwmChannelId (PwmChannel).....	151
4.6.1.2	PwmHwIP (PwmChannel).....	151
4.6.1.3	PwmFtmChannel (PwmChannel).....	151
4.6.1.4	PwmFlexIOChannel (PwmChannel).....	152
4.6.1.5	PwmPeriodInTicks (PwmChannel).....	152
4.6.1.6	PwmPeriodDefault (PwmChannel).....	152
4.6.1.7	PwmChannelClass (PwmChannel).....	153
4.6.1.8	PwmPolarity (PwmChannel).....	153
4.6.1.9	PwmDutycycleDefault (PwmChannel).....	154
4.6.1.10	PwmIdleState (PwmChannel).....	154
4.6.1.11	PwmNotification (PwmChannel).....	155

Section number	Title	Page
4.6.1.12	PwmMcuClockReferencePoint (PwmChannel).....	155
4.6.2	Form PwmFlexIOModule.....	156
4.6.2.1	PwmFlexIOModule (PwmFlexIO).....	156
4.6.2.2	Form PwmFlexIOChannels.....	157
4.6.2.2.1	PwmFlexIOChannelId (PwmFlexIOChannels).....	157
4.6.2.2.2	PwmFlexIOTimer (PwmFlexIOChannels).....	158
4.6.3	Form PwmFtmModule.....	158
4.6.3.1	FtmModule (PwmFtmModule).....	159
4.6.3.2	PwmPrescaler (PwmFtmModule).....	160
4.6.3.3	PwmPrescaler_Alternate (PwmFtmModule).....	160
4.6.3.4	PwmClockRef (PwmFtmModule).....	160
4.6.3.5	PwmClockSelection (PwmFtmModule).....	161
4.6.3.6	PwmChanEdgeAlignment (PwmFtmModule).....	161
4.6.3.7	PwmUpdatedEndPeriod (PwmFtmModule).....	162
4.6.3.8	PwmUpdatedMiddlePeriod (PwmFtmModule).....	162
4.6.3.9	PwmReloadFrequency (PwmFtmModule).....	163
4.6.3.10	PwmDeadTimeCount (PwmFtmModule).....	164
4.6.3.11	DeadTimePrescaler (PwmFtmModule).....	164
4.6.3.12	PwmBDMEnable (PwmFtmModule).....	165
4.6.3.13	PwmFtmFaultFunctionality (PwmFtmModule).....	165
4.6.3.14	Form PwmFtmChannels.....	166
4.6.3.14.1	PwmFtmChannelId (PwmFtmChannels).....	167
4.6.3.14.2	ChannelEdgeSetup (PwmFtmChannels).....	168
4.6.3.14.3	ChannelCombDeadTimeEn (PwmFtmChannels).....	169
4.6.3.14.4	PwmPhaseShift (PwmFtmChannels).....	169
4.6.3.15	Form PwmFtmChannelFaultSettings.....	170
4.6.3.15.1	PwmFilterValue (PwmFtmChannelFaultSettings).....	170
4.6.3.15.2	PwmDisableOutputOnFault0 (PwmFtmChannelFaultSettings).....	170
4.6.3.15.3	PwmDisableOutputOnFault1 (PwmFtmChannelFaultSettings).....	171

Section number	Title	Page
4.6.3.15.4	PwmDisableOutputOnFault2 (PwmFtmChannelFaultSettings).....	171
4.6.3.15.5	PwmDisableOutputOnFault3 (PwmFtmChannelFaultSettings).....	172
4.6.3.15.6	PwmFtmFaultOutputState (PwmFtmModule).....	172
4.6.3.15.7	Form PwmFtmChannelFault0Settings.....	172
4.6.3.15.7.1	PwmFault0Polarity (PwmFtmChannelFault0Settings).....	173
4.6.3.15.7.2	PwmEnableFault0Filter (PwmFtmChannelFault0Settings).....	173
4.6.3.15.7.3	PwmFault0Notification (PwmFtmChannelFault0Settings).....	174
4.6.3.15.8	Form PwmFtmChannelFault1Settings.....	174
4.6.3.15.8.1	PwmFault1Polarity (PwmFtmChannelFault1Settings).....	174
4.6.3.15.8.2	PwmEnableFault1Filter (PwmFtmChannelFault1Settings).....	175
4.6.3.15.8.3	PwmFault1Notification (PwmFtmChannelFault1Settings).....	175
4.6.3.15.9	Form PwmFtmChannelFault2Settings.....	176
4.6.3.15.9.1	PwmFault2Polarity (PwmFtmChannelFault2Settings).....	176
4.6.3.15.9.2	PwmEnableFault2Filter (PwmFtmChannelFault2Settings).....	176
4.6.3.15.9.3	PwmFault2Notification (PwmFtmChannelFault2Settings).....	177
4.6.3.15.10	Form PwmFtmChannelFault3Settings.....	177
4.6.3.15.10.1	PwmFault3Polarity (PwmFtmChannelFault3Settings).....	178
4.6.3.15.10.2	PwmEnableFault3Filter (PwmFtmChannelFault3Settings).....	178
4.6.3.15.10.3	PwmFault3Notification (PwmFtmChannelFault3Settings).....	178
4.6.3.16	Form PwmExternalTriggerSettings.....	179
4.6.3.16.1	PwmEnableExternalTriggerChannel0 (PwmExternalTriggerSettings).....	179
4.6.3.16.2	PwmEnableExternalTriggerChannel1 (PwmExternalTriggerSettings).....	180
4.6.3.16.3	PwmEnableExternalTriggerChannel2 (PwmExternalTriggerSettings).....	180
4.6.3.16.4	PwmEnableExternalTriggerChannel3 (PwmExternalTriggerSettings).....	180
4.6.3.16.5	PwmEnableExternalTriggerChannel4 (PwmExternalTriggerSettings).....	181
4.6.3.16.6	PwmEnableExternalTriggerChannel5 (PwmExternalTriggerSettings).....	181
4.6.3.16.7	PwmEnableExternalTriggerChannel6 (PwmExternalTriggerSettings).....	182
4.6.3.16.8	PwmEnableExternalTriggerChannel7 (PwmExternalTriggerSettings).....	182
4.6.3.16.9	PwmEnableInitializationTrigger (PwmExternalTriggerSettings).....	182





# Chapter 1

## Revision History

**Table 1-1. Revision History**

Revision	Date	Author	Description
1.0	26/04/2019	NXP MCAL Team	Updated version for ASR 4.2.2S32K14XR1.0.2



## Chapter 2

# Introduction

This User Manual describes NXP Semiconductors AUTOSAR Pulse Width Modulation ( PWM ) for S32K14X .

AUTOSAR PWM driver configuration parameters and deviations from the specification are described in PWM 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 .

**Table 2-1. S32K14X Derivatives**

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

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

## 2.2 Overview

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

## AUTOSAR

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

## 2.3 About this Manual

This Technical Reference employs the following typographical conventions:

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

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

Notes and warnings are shown as below:

### Note

This is a note.

## 2.4 Acronyms and Definitions

**Table 2-2. Acronyms and Definitions**

Term	Definition
FlexIO	Flexible I/O
FTM	Flexible Timer
API	Application Programming Interface
AUTOSAR	Automotive Open System Architecture
ASM	Assembler
BSMI	Basic Software Make file Interface
BSW	Basic Software
CAN	Controller Area Network
DEM	Diagnostic Event Manager

*Table continues on the next page...*

**Table 2-2. Acronyms and Definitions (continued)**

Term	Definition
DET	Development Error Tracer
C/CPP	C and C++ Source Code
ECU	Electronic Control Unit
ISR	Interrupt Service Routine
MCU	Micro Controller Unit
N/A	Not Applicable
OS	Operating System
PB Variant	Post Build Variant
PC Variant	Pre Compile Variant
PWM	Pulse Width Modulation
RAM	Random Access Memory
ROM	Read-only Memory
VLE	Variable Length Encoding

## 2.5 Reference List

**Table 2-3. Reference List**

#	Title	Version
1	Specification of PWM Driver	AUTOSAR Release 4.2.2
2	S32K14X Reference Manual	Reference Manual, Rev. 9, 9/2018
3	S32K142 Mask Set Errata for Mask 0N33V (0N33V)	30/11/2017
4	S32K144 Mask Set Errata for Mask 0N57U (0N57U)	30/11/2017
5	S32K146 Mask Set Errata for Mask 0N73V (0N73V)	30/11/2017
6	S32K148 Mask Set Errata for Mask 0N20V (0N20V)	25/10/2018
7	S32K118 Mask Set Errata for Mask 0N97V (0N97V)	07/01/2019



# Chapter 3

## Driver

### 3.1 Requirements

Requirements for this driver are detailed in the AUTOSAR 4.2 Rev0002PWM Driver Software Specification document (See Table [Reference List](#) ).

### 3.2 Driver Design Summary

The AUTOSAR PWM Driver Specification defines the functionality, API and the configuration of the AUTOSAR Basic Software module PWM Driver. Each PWM channel is linked to a hardware channel capable of implementing PWM functionality, which belongs to the microcontroller.

The driver provides functions for initialization and control of the microcontroller 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.

The S32K1XX microcontroller contains FlexIO module and FTM module.

### 3.3 Hardware Resources

**Table 3-1. PWM Hardware channels availability for S32K1XX family**

Derivatives	FlexIO module	FlexIO timer available	FlexIO channel available
All Derivatives	FlexIO_0	Timer_0, Timer_1, Timer_2, Timer_3	Channel_0, Channel_1, Channel_2, Channel_3, Channel_4, Channel_5, Channel_6, Channel_7

**Table 3-2. PWM Hardware channels availability for S32K142 family**

Package	FTM module	FTM channel available	FTM fault input available
LQFP-48	FTM_0	FTM_0_CH_0, FTM_0_CH_1, FTM_0_CH_2, FTM_0_CH_3, FTM_0_CH_4, FTM_0_CH_5, FTM_0_CH_6, FTM_0_CH_7	FTM_0_FLT_2
	FTM_1	FTM_1_CH_0, FTM_1_CH_1, FTM_1_CH_2, FTM_1_CH_3, FTM_1_CH_4, FTM_1_CH_5, FTM_1_CH_6, FTM_1_CH_7	FTM_1_FLT_0, FTM_1_FLT_1, FTM_1_FLT_2
	FTM_2	FTM_2_CH_0, FTM_2_CH_1, FTM_2_CH_2, FTM_2_CH_3	FTM_2_FLT_1
	FTM_3	FTM_3_CH_0, FTM_3_CH_1, FTM_3_CH_2, FTM_3_CH_3, FTM_3_CH_4, FTM_3_CH_5	FTM_3_FLT_1
LQFP-64	FTM_0	FTM_0_CH_0, FTM_0_CH_1, FTM_0_CH_2, FTM_0_CH_3, FTM_0_CH_4, FTM_0_CH_5, FTM_0_CH_6, FTM_0_CH_7	FTM_0_FLT_0, FTM_0_FLT_1, FTM_0_FLT_2, FTM_0_FLT_3
	FTM_1	FTM_1_CH_0, FTM_1_CH_1, FTM_1_CH_2, FTM_1_CH_3, FTM_1_CH_4, FTM_1_CH_5, FTM_1_CH_6, FTM_1_CH_7	FTM_1_FLT_0, FTM_1_FLT_1, FTM_1_FLT_2, FTM_1_FLT_3
	FTM_2	FTM_2_CH_0, FTM_2_CH_1, FTM_2_CH_2, FTM_2_CH_3, FTM_2_CH_4, FTM_2_CH_5	FTM_2_FLT_0, FTM_2_FLT_1, FTM_2_FLT_2, FTM_2_FLT_3
	FTM_3	FTM_3_CH_0, FTM_3_CH_1, FTM_3_CH_2, FTM_3_CH_3, FTM_3_CH_4, FTM_3_CH_5, FTM_3_CH_6, FTM_3_CH_7	FTM_3_FLT_0, FTM_3_FLT_1, FTM_3_FLT_2, FTM_3_FLT_3
LQFP-100	FTM_0	FTM_0_CH_0, FTM_0_CH_1, FTM_0_CH_2, FTM_0_CH_3, FTM_0_CH_4, FTM_0_CH_5, FTM_0_CH_6, FTM_0_CH_7	FTM_0_FLT_0, FTM_0_FLT_1, FTM_0_FLT_2, FTM_0_FLT_3
	FTM_1	FTM_1_CH_0, FTM_1_CH_1, FTM_1_CH_2, FTM_1_CH_3, FTM_1_CH_4, FTM_1_CH_5, FTM_1_CH_6, FTM_1_CH_7	FTM_1_FLT_0, FTM_1_FLT_1, FTM_1_FLT_2, FTM_1_FLT_3
	FTM_2	FTM_2_CH_0, FTM_2_CH_1, FTM_2_CH_2, FTM_2_CH_3, FTM_2_CH_4, FTM_2_CH_5, FTM_2_CH_6, FTM_2_CH_7	FTM_2_FLT_0, FTM_2_FLT_1, FTM_2_FLT_2, FTM_2_FLT_3
	FTM_3	FTM_3_CH_0, FTM_3_CH_1, FTM_3_CH_2, FTM_3_CH_3, FTM_3_CH_4, FTM_3_CH_5, FTM_3_CH_6, FTM_3_CH_7	FTM_3_FLT_0, FTM_3_FLT_1, FTM_3_FLT_2, FTM_3_FLT_3

**Table 3-3. PWM Hardware channels availability for S32K144 family**

Package	FTM module	FTM channel available	FTM fault input available
---------	------------	-----------------------	---------------------------

Table continues on the next page...



**Table 3-3. PWM Hardware channels availability for S32K144 family (continued)**

LQFP-48	FTM_0	FTM_0_CH_0, FTM_0_CH_1, FTM_0_CH_2, FTM_0_CH_3, FTM_0_CH_4, FTM_0_CH_5, FTM_0_CH_6, FTM_0_CH_7	FTM_0_FLT_2
	FTM_1	FTM_1_CH_0, FTM_1_CH_1, FTM_1_CH_2, FTM_1_CH_3, FTM_1_CH_4, FTM_1_CH_5, FTM_1_CH_6, FTM_1_CH_7	FTM_1_FLT_0, FTM_1_FLT_1, FTM_1_FLT_2
	FTM_2	FTM_2_CH_0, FTM_2_CH_1, FTM_2_CH_2, FTM_2_CH_3	FTM_2_FLT_1
	FTM_3	FTM_3_CH_0, FTM_3_CH_1, FTM_3_CH_2, FTM_3_CH_3, FTM_3_CH_4, FTM_3_CH_5	FTM_3_FLT_1
LQFP-64	FTM_0	FTM_0_CH_0, FTM_0_CH_1, FTM_0_CH_2, FTM_0_CH_3, FTM_0_CH_4, FTM_0_CH_5, FTM_0_CH_6, FTM_0_CH_7	FTM_0_FLT_0, FTM_0_FLT_1, FTM_0_FLT_2, FTM_0_FLT_3
	FTM_1	FTM_1_CH_0, FTM_1_CH_1, FTM_1_CH_2, FTM_1_CH_3, FTM_1_CH_4, FTM_1_CH_5, FTM_1_CH_6, FTM_1_CH_7	FTM_1_FLT_0, FTM_1_FLT_1, FTM_1_FLT_2, FTM_1_FLT_3
	FTM_2	FTM_2_CH_0, FTM_2_CH_1, FTM_2_CH_2, FTM_2_CH_3, FTM_2_CH_4, FTM_2_CH_5	FTM_2_FLT_0, FTM_2_FLT_1, FTM_2_FLT_2, FTM_2_FLT_3
	FTM_3	FTM_3_CH_0, FTM_3_CH_1, FTM_3_CH_2, FTM_3_CH_3, FTM_3_CH_4, FTM_3_CH_5, FTM_3_CH_6, FTM_3_CH_7	FTM_3_FLT_0, FTM_3_FLT_1, FTM_3_FLT_2, FTM_3_FLT_3
LQFP-100, MAPBGA-100	FTM_0	FTM_0_CH_0, FTM_0_CH_1, FTM_0_CH_2, FTM_0_CH_3, FTM_0_CH_4, FTM_0_CH_5, FTM_0_CH_6, FTM_0_CH_7	FTM_0_FLT_0, FTM_0_FLT_1, FTM_0_FLT_2, FTM_0_FLT_3
	FTM_1	FTM_1_CH_0, FTM_1_CH_1, FTM_1_CH_2, FTM_1_CH_3, FTM_1_CH_4, FTM_1_CH_5, FTM_1_CH_6, FTM_1_CH_7	FTM_1_FLT_0, FTM_1_FLT_1, FTM_1_FLT_2, FTM_1_FLT_3
	FTM_2	FTM_2_CH_0, FTM_2_CH_1, FTM_2_CH_2, FTM_2_CH_3, FTM_2_CH_4, FTM_2_CH_5, FTM_2_CH_6, FTM_2_CH_7	FTM_2_FLT_0, FTM_2_FLT_1, FTM_2_FLT_2, FTM_2_FLT_3
	FTM_3	FTM_3_CH_0, FTM_3_CH_1, FTM_3_CH_2, FTM_3_CH_3, FTM_3_CH_4, FTM_3_CH_5, FTM_3_CH_6, FTM_3_CH_7	FTM_3_FLT_0, FTM_3_FLT_1, FTM_3_FLT_2, FTM_3_FLT_3

**Table 3-4. PWM Hardware channels availability for S32K146 family**

Package	FTM module	FTM channel available	FTM fault input available
---------	------------	-----------------------	---------------------------

*Table continues on the next page...*

**Table 3-4. PWM Hardware channels availability for S32K146 family (continued)**

LQFP-64	FTM_0	FTM_0_CH_0, FTM_0_CH_1, FTM_0_CH_2, FTM_0_CH_3, FTM_0_CH_4, FTM_0_CH_5, FTM_0_CH_6, FTM_0_CH_7	FTM_0_FLT_0, FTM_0_FLT_1, FTM_0_FLT_2, FTM_0_FLT_3
	FTM_1	FTM_1_CH_0, FTM_1_CH_1, FTM_1_CH_2, FTM_1_CH_3, FTM_1_CH_4, FTM_1_CH_5, FTM_1_CH_6, FTM_1_CH_7	FTM_1_FLT_0, FTM_1_FLT_1, FTM_1_FLT_2, FTM_1_FLT_3
	FTM_2	FTM_2_CH_0, FTM_2_CH_1, FTM_2_CH_2, FTM_2_CH_3, FTM_2_CH_4, FTM_2_CH_5	FTM_2_FLT_0, FTM_2_FLT_1, FTM_2_FLT_2, FTM_2_FLT_3
	FTM_3	FTM_3_CH_0, FTM_3_CH_1, FTM_3_CH_2, FTM_3_CH_3, FTM_3_CH_4, FTM_3_CH_5, FTM_3_CH_6, FTM_3_CH_7	FTM_3_FLT_0, FTM_3_FLT_1, FTM_3_FLT_2, FTM_3_FLT_3
	FTM_4	FTM_4_CH_5, FTM_4_CH_6	N/A
	FTM_5	FTM_5_CH_0, FTM_5_CH_1, FTM_5_CH_3, FTM_5_CH_5	N/A
LQFP-100, MAPBGA-100	FTM_0	FTM_0_CH_0, FTM_0_CH_1, FTM_0_CH_2, FTM_0_CH_3, FTM_0_CH_4, FTM_0_CH_5, FTM_0_CH_6, FTM_0_CH_7	FTM_0_FLT_0, FTM_0_FLT_1, FTM_0_FLT_2, FTM_0_FLT_3
	FTM_1	FTM_1_CH_0, FTM_1_CH_1, FTM_1_CH_2, FTM_1_CH_3, FTM_1_CH_4, FTM_1_CH_5, FTM_1_CH_6, FTM_1_CH_7	FTM_1_FLT_0, FTM_1_FLT_1, FTM_1_FLT_2, FTM_1_FLT_3
	FTM_2	FTM_2_CH_0, FTM_2_CH_1, FTM_2_CH_2, FTM_2_CH_3, FTM_2_CH_4, FTM_2_CH_5, FTM_2_CH_6, FTM_2_CH_7	FTM_2_FLT_0, FTM_2_FLT_1, FTM_2_FLT_2, FTM_2_FLT_3
	FTM_3	FTM_3_CH_0, FTM_3_CH_1, FTM_3_CH_2, FTM_3_CH_3, FTM_3_CH_4, FTM_3_CH_5, FTM_3_CH_6, FTM_3_CH_7	FTM_3_FLT_0, FTM_3_FLT_1, FTM_3_FLT_2, FTM_3_FLT_3
	FTM_4	FTM_4_CH_2, FTM_4_CH_5, FTM_4_CH_6	FTM_4_FLT_0, FTM_4_FLT_1
	FTM_5	FTM_5_CH_0, FTM_5_CH_1, FTM_5_CH_3, FTM_5_CH_5	FTM_5_FLT_0, FTM_5_FLT_1
LQFP-144	FTM_0	FTM_0_CH_0, FTM_0_CH_1, FTM_0_CH_2, FTM_0_CH_3, FTM_0_CH_4, FTM_0_CH_5, FTM_0_CH_6, FTM_0_CH_7	FTM_0_FLT_0, FTM_0_FLT_1, FTM_0_FLT_2, FTM_0_FLT_3
	FTM_1	FTM_1_CH_0, FTM_1_CH_1, FTM_1_CH_2, FTM_1_CH_3, FTM_1_CH_4, FTM_1_CH_5, FTM_1_CH_6, FTM_1_CH_7	FTM_1_FLT_0, FTM_1_FLT_1, FTM_1_FLT_2, FTM_1_FLT_3
	FTM_2	FTM_2_CH_0, FTM_2_CH_1, FTM_2_CH_2, FTM_2_CH_3, FTM_2_CH_4, FTM_2_CH_5, FTM_2_CH_6, FTM_2_CH_7	FTM_2_FLT_0, FTM_2_FLT_1, FTM_2_FLT_2, FTM_2_FLT_3

Table continues on the next page...

**Table 3-4. PWM Hardware channels availability for S32K146 family (continued)**

	FTM_3	FTM_3_CH_0, FTM_3_CH_1, FTM_3_CH_2, FTM_3_CH_3, FTM_3_CH_4, FTM_3_CH_5, FTM_3_CH_6, FTM_3_CH_7	FTM_3_FLT_0, FTM_3_FLT_1, FTM_3_FLT_2, FTM_3_FLT_3
	FTM_4	FTM_4_CH_0, FTM_4_CH_1, FTM_4_CH_2, FTM_4_CH_3, FTM_4_CH_4, FTM_4_CH_5, FTM_4_CH_6, FTM_4_CH_7	FTM_4_FLT_0, FTM_4_FLT_1
	FTM_5	FTM_5_CH_0, FTM_5_CH_1, FTM_5_CH_2, FTM_5_CH_3, FTM_5_CH_4, FTM_5_CH_5, FTM_5_CH_6, FTM_5_CH_7	FTM_5_FLT_0, FTM_5_FLT_1

**Table 3-5. PWM Hardware channels availability for S32K148 family**

Package	FTM module	FTM channel available	FTM fault input available
LQFP-100, MAPBGA-100	FTM_0	FTM_0_CH_0, FTM_0_CH_1, FTM_0_CH_2, FTM_0_CH_3, FTM_0_CH_4, FTM_0_CH_5, FTM_0_CH_6, FTM_0_CH_7	FTM_0_FLT_0, FTM_0_FLT_1, FTM_0_FLT_2, FTM_0_FLT_3
	FTM_1	FTM_1_CH_0, FTM_1_CH_1, FTM_1_CH_2, FTM_1_CH_3, FTM_1_CH_4, FTM_1_CH_5, FTM_1_CH_6, FTM_1_CH_7	FTM_1_FLT_0, FTM_1_FLT_1, FTM_1_FLT_2, FTM_1_FLT_3
	FTM_2	FTM_2_CH_0, FTM_2_CH_1, FTM_2_CH_2, FTM_2_CH_3, FTM_2_CH_4, FTM_2_CH_5, FTM_2_CH_6, FTM_2_CH_7	FTM_2_FLT_0, FTM_2_FLT_1, FTM_2_FLT_2, FTM_2_FLT_3
	FTM_3	FTM_3_CH_0, FTM_3_CH_1, FTM_3_CH_2, FTM_3_CH_3, FTM_3_CH_4, FTM_3_CH_5, FTM_3_CH_6, FTM_3_CH_7	FTM_3_FLT_0, FTM_3_FLT_1, FTM_3_FLT_2, FTM_3_FLT_3
	FTM_4	FTM_4_CH_2, FTM_4_CH_5, FTM_4_CH_6	FTM_4_FLT_0, FTM_4_FLT_1
	FTM_5	FTM_5_CH_0, FTM_5_CH_1, FTM_5_CH_3, FTM_5_CH_5	FTM_5_FLT_0, FTM_5_FLT_1
LQFP-144	FTM_0	FTM_0_CH_0, FTM_0_CH_1, FTM_0_CH_2, FTM_0_CH_3, FTM_0_CH_4, FTM_0_CH_5, FTM_0_CH_6, FTM_0_CH_7	FTM_0_FLT_0, FTM_0_FLT_1, FTM_0_FLT_2, FTM_0_FLT_3
	FTM_1	FTM_1_CH_0, FTM_1_CH_1, FTM_1_CH_2, FTM_1_CH_3, FTM_1_CH_4, FTM_1_CH_5, FTM_1_CH_6, FTM_1_CH_7	FTM_1_FLT_0, FTM_1_FLT_1, FTM_1_FLT_2, FTM_1_FLT_3
	FTM_2	FTM_2_CH_0, FTM_2_CH_1, FTM_2_CH_2, FTM_2_CH_3, FTM_2_CH_4, FTM_2_CH_5, FTM_2_CH_6, FTM_2_CH_7	FTM_2_FLT_0, FTM_2_FLT_1, FTM_2_FLT_2, FTM_2_FLT_3

Table continues on the next page...

**Table 3-5. PWM Hardware channels availability for S32K148 family (continued)**

	FTM_3	FTM_3_CH_0, FTM_3_CH_1, FTM_3_CH_2, FTM_3_CH_3, FTM_3_CH_4, FTM_3_CH_5, FTM_3_CH_6, FTM_3_CH_7	FTM_3_FLT_0, FTM_3_FLT_1, FTM_3_FLT_2, FTM_3_FLT_3
	FTM_4	FTM_4_CH_0, FTM_4_CH_1, FTM_4_CH_2, FTM_4_CH_3, FTM_4_CH_4, FTM_4_CH_5, FTM_4_CH_6, FTM_4_CH_7	FTM_4_FLT_0, FTM_4_FLT_1
	FTM_5	FTM_5_CH_0, FTM_5_CH_1, FTM_5_CH_2, FTM_5_CH_3, FTM_5_CH_4, FTM_5_CH_5, FTM_5_CH_6, FTM_5_CH_7	FTM_5_FLT_0, FTM_5_FLT_1
	FTM_6	FTM_6_CH_0, FTM_6_CH_1, FTM_6_CH_2, FTM_6_CH_3, FTM_6_CH_4, FTM_6_CH_5, FTM_6_CH_7	FTM_6_FLT_0, FTM_6_FLT_1
	FTM_7	FTM_7_CH_0, FTM_7_CH_1, FTM_7_CH_2, FTM_7_CH_3, FTM_7_CH_5, FTM_7_CH_7	FTM_7_FLT_0, FTM_7_FLT_1
LQFP-176	FTM_0	FTM_0_CH_0, FTM_0_CH_1, FTM_0_CH_2, FTM_0_CH_3, FTM_0_CH_4, FTM_0_CH_5, FTM_0_CH_6, FTM_0_CH_7	FTM_0_FLT_0, FTM_0_FLT_1, FTM_0_FLT_2, FTM_0_FLT_3
	FTM_1	FTM_1_CH_0, FTM_1_CH_1, FTM_1_CH_2, FTM_1_CH_3, FTM_1_CH_4, FTM_1_CH_5, FTM_1_CH_6, FTM_1_CH_7	FTM_1_FLT_0, FTM_1_FLT_1, FTM_1_FLT_2, FTM_1_FLT_3
	FTM_2	FTM_2_CH_0, FTM_2_CH_1, FTM_2_CH_2, FTM_2_CH_3, FTM_2_CH_4, FTM_2_CH_5, FTM_2_CH_6, FTM_2_CH_7	FTM_2_FLT_0, FTM_2_FLT_1, FTM_2_FLT_2, FTM_2_FLT_3
	FTM_3	FTM_3_CH_0, FTM_3_CH_1, FTM_3_CH_2, FTM_3_CH_3, FTM_3_CH_4, FTM_3_CH_5, FTM_3_CH_6, FTM_3_CH_7	FTM_3_FLT_0, FTM_3_FLT_1, FTM_3_FLT_2, FTM_3_FLT_3
	FTM_4	FTM_4_CH_0, FTM_4_CH_1, FTM_4_CH_2, FTM_4_CH_3, FTM_4_CH_4, FTM_4_CH_5, FTM_4_CH_6, FTM_4_CH_7	FTM_4_FLT_0, FTM_4_FLT_1
	FTM_5	FTM_5_CH_0, FTM_5_CH_1, FTM_5_CH_2, FTM_5_CH_3, FTM_5_CH_4, FTM_5_CH_5, FTM_5_CH_6, FTM_5_CH_7	FTM_5_FLT_0, FTM_5_FLT_1
	FTM_6	FTM_6_CH_0, FTM_6_CH_1, FTM_6_CH_2, FTM_6_CH_3, FTM_6_CH_4, FTM_6_CH_5, FTM_6_CH_6, FTM_6_CH_7	FTM_6_FLT_0, FTM_6_FLT_1
	FTM_7	FTM_7_CH_0, FTM_7_CH_1, FTM_7_CH_2, FTM_7_CH_3, FTM_7_CH_4, FTM_7_CH_5, FTM_7_CH_6, FTM_7_CH_7	FTM_7_FLT_0, FTM_7_FLT_1

### 3.4 Deviation from Requirements

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

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

**Table 3-6. Deviations Status Column Description**

Term	Definition
N/A	Not available
N/T	Not testable
N/S	Out of scope
N/I	Not implemented
N/F	Not fully implemented

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

**Table 3-7. Driver Deviations Table**

Requirement	Status	Description	Notes
PWM065	N/A	The Pwm SWS shall not define the code file structure.	N/A
PWM075c	N/I	Pwm_Lcfg.c shall include Pwm.h and Pwm_Memmap.h.	Link time configuration not supported.
PWM075e	N/I	Pwm.c shall include Pwm.h, Pwm_MemMap.h, Det.h and SchM_Pwm.h.	SchM_Pwm.h is included by lower layer of the driver.
PWM075g	N/A	Pwm_Irq.c shall include Pwm_MemMap.h and Pwm.h.	Due to layer separation, implementation of Pwm driver does not include a Pwm_Irq.c file. The interrupts are handled in distinct files by each separate hardware module used
PWM093	N/S	The users of the Pwm module shall not call the function Pwm_Init during a running operation.	N/A
PWM116	N/I	The Pwm module's environment shall not call any function of the Pwm module before having called Pwm_Init.	N/A
PWM120a	N/I	For pre-compile and link time configuration variants, a NULL pointer shall be passed to the initialization routine.	Applicable if only one variant is used.
SWS_Pwm_001 62	N/S	The Pwm Driver shall support synchronous and asynchronous power state transitions, depending on the value of the configuration	eMIOS Hw does not support asynchronous power state transitions

*Table continues on the next page...*

**Table 3-7. Driver Deviations Table (continued)**

Requirement	Status	Description	Notes
		parameter PwmPowerStateAsynchTransitionMode.	
SWS_Pwm_001 64	N/S	In case the configuration parameter PwmPowerStateAsynchTransitionMode 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.	eMIOS Hw does not support asynchronous power state transitions

## 3.5 FlexIO specific implementation details

### FlexIO for supporting multiple modules

In this current implementation, FlexIO is supported for list of modules below:

- PWM
- SPI
- I2C
- LIN
- ...

All modules are integrated configuration in MCL, so users must configure MCL module to use PWM functionality. Refer to section [Configure PwmFlexIOModule](#) for details.

### Function calling:

User should follow these guides to run if configured FlexIO for PWM functionality:

- Include MCL generated configuration file (e.g. CDD\_Mcl\_Cfg.h, CDD\_Mcl\_PBCfg.c) along with PWM configuration file destination.
- Add '#include "Mcl.h"' in code.
- In main function, call Mcl\_Init() function first.
- Call Pwm\_Init() function after Mcl\_Init().

## 3.6 FTM specific implementation details

### The phase shift functionality

PWM phase shift provides a group of channels which have difference phase for each specific channels. Current implementation supports two types of phase shift.

- The traditional phase shift bases on Modified Combine mode of FTM. This mode implements PWM Class `PWM_FIXED_PERIOD_SHIFTED`, which the period and phase shift value is fixed, the duty cycle can be variable. In order to configure this mode, parameter `PwmGeneral/PwmEnablePhaseShift` should be enable first. The reference channel needs to be configured with `ChannelEdgeSetup` as `PHASE_SHIFTED_SYNCED` or `PHASE_SHIFTED_COMPLEMENTARY`.
- The second type of phase shift bases on Combine mode of FTM. This mode supports full-bridge phase shift controller. Other than normal PWM channels, duty cycle in this mode is always fixed to 50% while period and phase shift value can be variable. To have channels operate in this mode, `PwmSetPhaseShift` or/and `PwmSetPhaseShiftNoUpdate` should be enabled first. The `ChannelEdgeSetup` of reference channel need to be `COMBINED_SYNCED` or `COMBINED_COMPLEMENTARY`.

### NOTE

The phase shift mode bases on Combine mode does not support notification. Therefore when `PwmSetPhaseShift` or/and `PwmSetPhaseShiftNoUpdate` is enabled, channel reference to `COMBINED_SYNCED` or `COMBINED_COMPLEMENTARY` should not be assigned any notification.

## 3.7 Driver limitations

In this current implementation, there are some limitations for FlexIO driver due to hardware supports:

- FlexIO compare registers are not buffered so they are written directly, therefore the update of duty cycle and period will be immediate. Option `PwmDutycycleUpdatedEndperiod` and `PwmPeriodUpdatedEndperiod` is not supported for FlexIO.
- FlexIO notification only support for rising edge. Other cases are not supported, including falling edges and both edges notification.
- FlexIO does not support functionality of `Pwm_SetOutputToIdle`. With function `Pwm_GetOutputState`, calling this function for FlexIO PWM channel will always return `PWM_LOW`.
- FlexIO does not support for Duty cycle : (0)0% , (0x8000)100% and Period :0U, When user call this function with parameter 0% or 100% duty cycle, 0 preiod ,it will generate a spike pulse.

### 3.8 Driver usage and configuration tips

Basically, PWM driver should be configured in 3 steps:

- Choose PWM module/channel to be used and its interrupt in PwmHwConfiguration.
- Configure PWM module/channel in PwmModule tab.
- Configure logic channel in PwmChannel.

#### 3.8.1 Configure PwmHwConfiguration

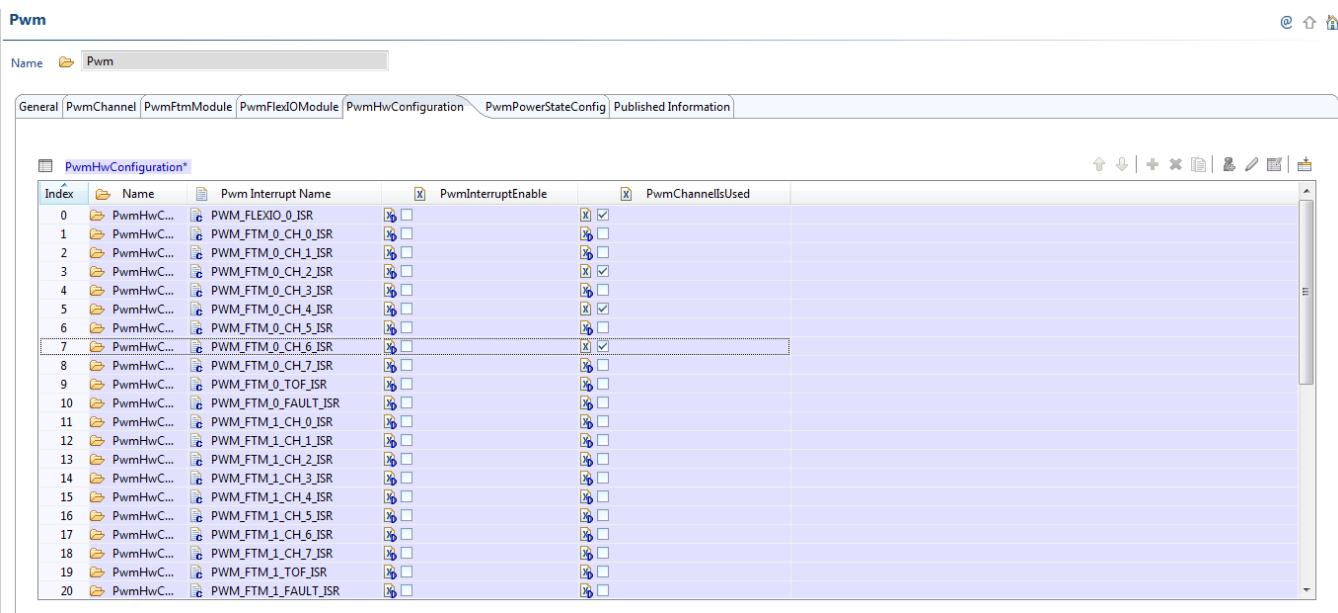


Figure 3-1. Tresos Plugin snapshot for PwmHwConfiguration form.

PwmHwConfiguration form allows user to choose modules or channels to be used in configuration with PwmIsUsed selection, also provides options to use interrupt service for each module/channel in PwmInterruptEnable selection.

#### 3.8.2 Configure PwmModule

This part contains guide to configure 2 modules available for S32K1XX: FlexIO and FTM.



### 3.8.2.1 Configure PwmFtmModule

Configuration IP hardware specific shall be done with the flow similar to hardware structure. For FTM, hardware structure is PWM FTM module -> PWM FTM channel.

- Configure FTM module in PwmFtmModule form

The screenshot displays the 'PwmFtmChannels' configuration tab in the Tresos Plugin. The configuration options are as follows:

- Ftm Hardware Module\***: FTM\_0
- Prescaler\***: PRESC\_1
- PwmPrescaler\_Alternate\***: PRESC\_1
- FTM Module clock selection\***: PWM\_SYSTEM\_CLOCK
- Ftm Module's Channels Alignment\***: PWM\_EDGE\_ALIGNED
- End cycle reload\***: ☒ **Half cycle reload\***: ☐
- Reload Frequency\***: LDFQ\_EACH1
- Deadtime Counter (0 -> 1023)\***: 0
- DeadTime Counter Prescaler\***: PRESC\_1
- Pwm Background Debug Mode configuration\***: CNT\_STOPPED\_FLAG\_SET
- Pwm Fault Funtionality and Clear Mode\***: FLTCTRL\_DISABLED


**Fault Settings**

- Name\***: PwmFtmChannelFaultSettings
- Pwm Fault Filter Value (0 -> 15)\***: 0
- Disable Channel Output On Fault 0\***: ☐ **Disable Channel Output On Fault 1\***: ☐

**Figure 3-2. Tresos Plugin snapshot for PwmFtmModule form.**



- Configure FTM channel in PwmFtmChannel form



## PwmFtmChannels


Name\*  PwmFtmChannels\_0

---

General

Ftm Hardware Channel\*  FTM\_0\_CH\_1 

Edge configuration setting for current channel\*  INDEPENDENT 

Phase Shift Ticks (0 -> 65533)\*  0




Enable Deadtime on combined channels\*   

Figure 3-3. Tresos Plugin snapshot for PwmFtmChannel form.

### 3.8.2.2 Configure PwmFlexIOModule

Configuration FlexIO hardware specific shall be done with the following guides:

**In MCL modules:**

- In MclGeneral section, Enable node :Mcl FlexIO Support.

**Mcl General Configuration**

Name MclGeneral

Mcl Disable Production Error Reporting	<input type="checkbox"/>	Mcl Development Error Detect	<input checked="" type="checkbox"/>
Mcl Dma Notification Supported	<input checked="" type="checkbox"/>	Mcl Dma Error Notification Supported	<input checked="" type="checkbox"/>
Mcl_VersionInfoApi	<input checked="" type="checkbox"/>	Mcl_DmaGetChannelErrorStatusApi	<input type="checkbox"/>
Mcl_DmaGetGlobalErrorStatusApi	<input type="checkbox"/>	Mcl_DeInitApi	<input type="checkbox"/>
Mcl_CommonTimebaseSupported	<input type="checkbox"/>	Mcl DMA Supported	<input checked="" type="checkbox"/>
Mcl Trig Mux supported	<input type="checkbox"/>	Mcl Enable User Mode Support	<input type="checkbox"/>
<b>Mcl FlexIO Support</b>	<input checked="" type="checkbox"/>		
Mcl Error Notification for Dma Instance 0	NULL_PTR		

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

- In MclConfigSet section, user can choose DOZEN or DBGE enable bit option

**FlexioConfig**

Name\* FlexioConfig

---

**FlexioGeneral**

Name\* FlexioGeneral

---

Dozen Enable\* ☒

DBGE\* ☒

**Figure 3-5. Tresos Plugin snapshot for MclConfigSet/FlexioGeneral form.**

- Then, user need to call Mcl\_Init() function before Pwm\_Init() When using FlexIO module

### In PWM modules:

- In PwmFlexIO module.

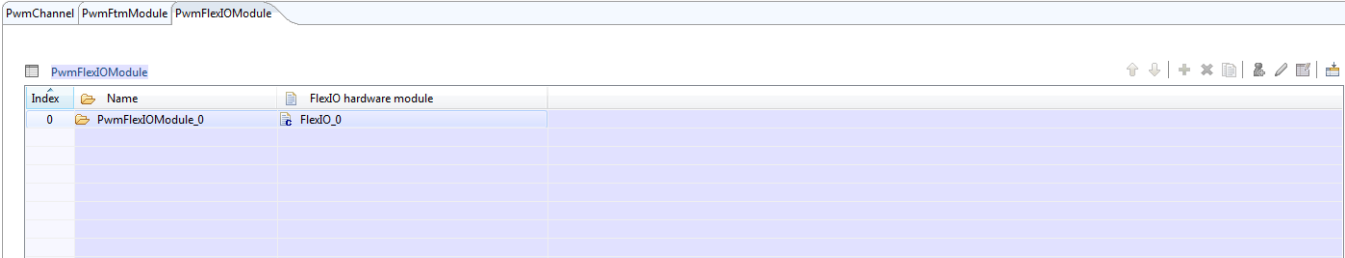


Figure 3-6. Tresos Plugin snapshot for PwmFlexIO form.

- In PwmFlexIOChannels section, user must choose FlexIO channel, FlexIO timer for PWM channel

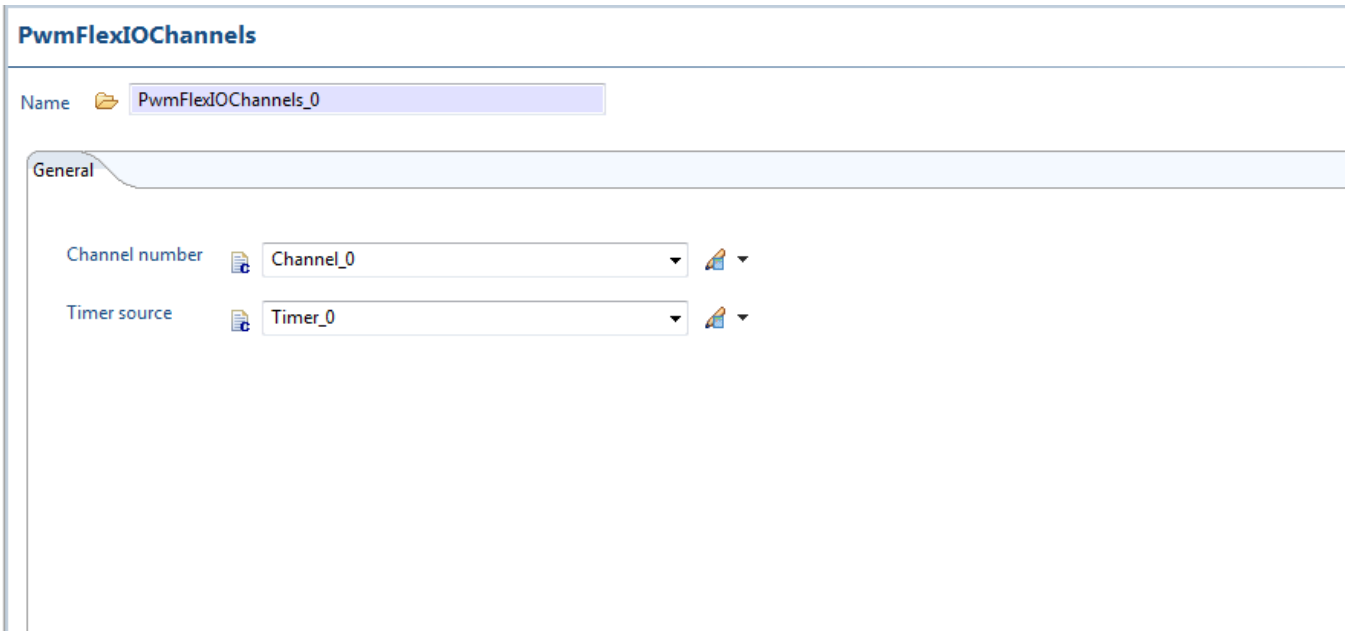


Figure 3-7. Tresos Plugin snapshot for PwmFlexIOChannels form.

### 3.8.3 Configure logic channel in PwmChannel

This list contains logical channel working as object of PWM APIs. Each logic channel refers to PWM hardware channel which has been configured from PWM modules through above steps.

## PwmChannel



Name\* PwmChannel\_0

General

PwmChannelId (0 -> 4294967295)*	0
PWM Hardware IP*	FTM
PwmFtmChannel*	/Pwm/Pwm/PwmChannelConfigSet/PwmFtmModule_0/PwmFtmChannels_0
PwmFlexIOChannel*	
Default Period In Ticks*	<input checked="" type="checkbox"/>
Default Period (0 -> 65534)*	1.25E-4
PwmChannelClass*	PWM_FIXED_PERIOD
PwmPolarity*	PWM_HIGH
PwmDutyCycleDefault (0 -> 32768)*	16384
PwmIdleState*	PWM_LOW
PwmNotification*	NULL
PwmMcuClockReferencePoint*	/Mcu/Mcu/McuModuleConfiguration_0/McuClockSettingConfig_0/McuClockReferencePoint_0

**Figure 3-8. Tresos Plugin snapshot for PwmChannel form.**

Select the hardware channel that needs to be referenced to, the period, the duty cycle...

### 3.8.4 Implement errata

For errata e10856 implementation, PWM module should be configured as follow to handle errata successfully:

- In PwmGeneral, option PwmNotificationSupported and Fault Support Enable (PwmFaultSupport) should be set both to enable TOF and fault notification.
- In PwmFtmModule selected to implement errata, the option Pwm Fault Funtionality and Clear Mode (PwmFtmFaultFunctionality) should be chosen between AUTOMATIC\_FOR\_EVEN\_CHANNELS and AUTOMATIC\_FOR\_ALL\_CHANNELS. TOF\_ISR and FAULT\_ISR of selected FTM module will be generated automatically on, and functions of both ISRs will be available.

## 3.9 Runtime Errors

None.

## 3.10 Software specification

The following sections contains driver software specifications.

### 3.10.1 Define Reference

Constants supported by the driver are as per AUTOSAR PWM Driver software specification Version 4.2 Rev0002 .

#### 3.10.1.1 Define PWM\_VENDOR\_ID

Table 3-8. Define PWM\_VENDOR\_ID Description

Name	PWM_VENDOR_ID
Initializer	43

#### 3.10.1.2 Define PWM\_MODULE\_ID

Table 3-9. Define PWM\_MODULE\_ID Description

Name	PWM_MODULE_ID
Initializer	121

#### 3.10.1.3 Define PWM\_AR\_RELEASE\_MAJOR\_VERSION

Table 3-10. Define PWM\_AR\_RELEASE\_MAJOR\_VERSION Description

Name	PWM_AR_RELEASE_MAJOR_VERSION
Initializer	4

#### 3.10.1.4 Define PWM\_AR\_RELEASE\_MINOR\_VERSION

Table 3-11. Define PWM\_AR\_RELEASE\_MINOR\_VERSION Description

Name	PWM_AR_RELEASE_MINOR_VERSION
Initializer	2

### 3.10.1.5 Define PWM\_AR\_RELEASE\_REVISION\_VERSION

**Table 3-12. Define PWM\_AR\_RELEASE\_REVISION\_VERSION  
Description**

<b>Name</b>	PWM_AR_RELEASE_REVISION_VERSION
<b>Initializer</b>	2

### 3.10.1.6 Define PWM\_SW\_MAJOR\_VERSION

**Table 3-13. Define PWM\_SW\_MAJOR\_VERSION  
Description**

<b>Name</b>	PWM_SW_MAJOR_VERSION
<b>Initializer</b>	1

### 3.10.1.7 Define PWM\_SW\_MINOR\_VERSION

**Table 3-14. Define PWM\_SW\_MINOR\_VERSION  
Description**

<b>Name</b>	PWM_SW_MINOR_VERSION
<b>Initializer</b>	0

### 3.10.1.8 Define PWM\_SW\_PATCH\_VERSION

**Table 3-15. Define PWM\_SW\_PATCH\_VERSION  
Description**

<b>Name</b>	PWM_SW_PATCH_VERSION
<b>Initializer</b>	4

### 3.10.1.9 Define PWM\_DUTY\_CYCLE\_100

100% duty cycle.

#### **Details:**

Value used to map a duty cycle of 100% on a 16 bit variable.

**Table 3-16. Define PWM\_DUTY\_CYCLE\_100 description**

<b>Name</b>	PWM_DUTY_CYCLE_100
<b>Initializer</b>	((uint16) 0x8000U)

### 3.10.1.10 Define PWM\_E\_PARAM\_CONFIG

Error signaling that Pwm\_Init service was called with wrong parameter.

#### Details:

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

**Implements:** Pwm\_ErrorIds\_define AUTOSAR

**Table 3-17. Define PWM\_E\_PARAM\_CONFIG description**

<b>Name</b>	PWM_E_PARAM_CONFIG
<b>Initializer</b>	0x10U

### 3.10.1.11 Define PWM\_E\_UNINIT

Error signaling that an API service was called before module initialization.

#### Details:

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

**Implements:** Pwm\_ErrorIds\_define AUTOSAR

**Table 3-18. Define PWM\_E\_UNINIT Description**

<b>Name</b>	PWM_E_UNINIT
<b>Initializer</b>	0x11U



### 3.10.1.12 Define PWM\_E\_PARAM\_CHANNEL

Error signaling that an API service was called with an invalid channel identifier.

#### **Details:**

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

**Implements:** Pwm\_ErrorIds\_define AUTOSAR

**Table 3-19. Define PWM\_E\_PARAM\_CHANNEL Description**

<b>Name</b>	PWM_E_PARAM_CHANNEL
<b>Initializer</b>	0x12U

### 3.10.1.13 Define PWM\_E\_PERIOD\_UNCHANGEABLE

Error signaling incorrect usage of PWM channel service on a channel configured with fixed period.

#### **Details:**

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

**Implements:** Pwm\_ErrorIds\_define AUTOSAR

**Table 3-20. Define PWM\_E\_PERIOD\_UNCHANGEABLE Description**

<b>Name</b>	PWM_E_PERIOD_UNCHANGEABLE
<b>Initializer</b>	0x13U

### 3.10.1.14 Define PWM\_E\_ALREADY\_INITIALIZED

Error signaling that Pwm\_Init service was called while the PWM driver was already initialised.

#### **Details:**

Will be reported to DET when Pwm\_Init service was called while the PWM driver was already initialised.

**Implements:** Pwm\_ErrorIds\_define AUTOSAR

**Table 3-21. Define PWM\_E\_ALREADY\_INITIALIZED Description**

<b>Name</b>	PWM_E_ALREADY_INITIALIZED
<b>Initializer</b>	0x14U

### 3.10.1.15 Define PWM\_E\_PARAM\_POINTER

Error signaling that an invalid parameter pointer is passed to Pwm\_GetVersionInfo function.

#### **Details:**

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

**Implements:** Pwm\_ErrorIds\_define AUTOSAR

**Table 3-22. Define PWM\_E\_PARAM\_POINTER Description**

<b>Name</b>	PWM_E_PARAM_POINTER
<b>Initializer</b>	0x15U

### 3.10.1.16 Define PWM\_E\_PARAM\_NOTIFICATION

Invalid polarity selected for edge notification.

#### **Details:**

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.

**Implements:** Pwm\_ErrorIds\_define AUTOSAR

**Table 3-23. Define PWM\_E\_PARAM\_NOTIFICATION Description**

<b>Name</b>	PWM_E_PARAM_NOTIFICATION
<b>Initializer</b>	0x30U

### 3.10.1.17 Define PWM\_E\_PARAM\_NOTIFICATION\_NULL

Error signaling that a NULL function is configured as notification callback.

#### Details:

Will be generated when a NULL function is configured as notification callback for one PWM channel and Pwm\_EnableNotification is called for that channel.

**Implements:** Pwm\_ErrorIds\_define AUTOSAR

**Table 3-24. Define PWM\_E\_PARAM\_NOTIFICATION\_NULL Description**

<b>Name</b>	PWM_E_PARAM_NOTIFICATION_NULL
<b>Initializer</b>	0x31U

### 3.10.1.18 Define PWM\_E\_DUTYCYCLE\_RANGE

Error signaling that Pwm\_SetDutyCycle or Pwm\_SetPeriodAndDuty service was called with invalid duty cycle range.

#### Details:

Generated when Pwm\_SetDutyCycle or Pwm\_SetPeriodAndDuty are called with a value for duty cycle out of valid range [0x0000, 0x8000].

**Implements:** Pwm\_ErrorIds\_define Non-AUTOSAR

**Table 3-25. Define PWM\_E\_DUTYCYCLE\_RANGE Description**

<b>Name</b>	PWM_E_DUTYCYCLE_RANGE
<b>Initializer</b>	0x32U

### 3.10.1.19 Define PWM\_E\_COUNTERBUS

Generated when Pwm\_SetCounterBus is called with an invalid bus.

#### Details:

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

**Implements:** Pwm\_ErrorIds\_define Non-AUTOSAR

**Note:** In current implementation, this definition is not used.

**Table 3-26. Define PWM\_E\_COUNTERBUS Description**

<b>Name</b>	PWM_E_COUNTERBUS
<b>Initializer</b>	0x33U

### 3.10.1.20 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.

#### Details:

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

**Implements:** Pwm\_ErrorIds\_define Non-AUTOSAR

**Table 3-27. Define PWM\_E\_CHANNEL\_OFFSET\_VALUE  
Description**

<b>Name</b>	PWM_E_CHANNEL_OFFSET_VALUE
<b>Initializer</b>	0x34U

### 3.10.1.21 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.

#### **Details:**

**Implements:** Pwm\_ErrorIds\_define Non-AUTOSAR

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

**Note:** In current implementation, this definition is not used.

**Table 3-28. Define PWM\_E\_OPWMB\_CHANNEL\_OFFSET\_DUTYCYCLE\_RANGE Description**

<b>Name</b>	PWM_E_OPWMB_CHANNEL_OFFSET_DUTYCYCLE_RANGE
<b>Initializer</b>	0x35U

### 3.10.1.22 Define PWM\_E\_PARAM\_INSTANCE

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

#### **Details:**

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

**Implements:** Pwm\_ErrorIds\_define Non-AUTOSAR

**Table 3-29. Define PWM\_E\_PARAM\_INSTANCE Description**

<b>Name</b>	PWM_E_PARAM_INSTANCE
<b>Initializer</b>	(0x36U)

### 3.10.1.23 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.

#### Details:

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

**Implements:** Pwm\_ErrorIds\_define Non-AUTOSAR

**Note:** In current implementation, this definition is not used.

**Table 3-30. Define PWM\_E\_OPWMT\_CHANNEL\_TRIGGER\_RANGE Description**

<b>Name</b>	PWM_E_OPWMT_CHANNEL_TRIGGER_RANGE
<b>Initializer</b>	0x37U

### 3.10.1.24 Define PWM\_E\_OUTPUT\_STATE

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

#### Details:

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

**Implements:** Pwm\_ErrorIds\_define Non-AUTOSAR

**Note:** In current implementation, this definition is not used.

**Table 3-31. Define PWM\_E\_OUTPUT\_STATE Description**

<b>Name</b>	PWM_E_OUTPUT_STATE
<b>Initializer</b>	0x38U

### 3.10.1.25 Define PWM\_E\_UNEXPECTED\_ISR

Error signaling that an unexpected PWM interrupt has been triggered:

1. When the driver is not initialized.
2. For a hardware channel that is not used by any logic channel.
3. For a logic channel that has no notification configured.

**Details:**

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

**Implements:** Pwm\_ErrorIds\_define Non-AUTOSAR

**Table 3-32. Define PWM\_E\_UNEXPECTED\_ISR  
Description**

<b>Name</b>	PWM_E_UNEXPECTED_ISR
<b>Initializer</b>	0x39U

### 3.10.1.26 Define PWM\_E\_PARAM\_PHASESHIFT\_RANGE

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

**Details:**

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

**Implements:** Pwm\_ErrorIds\_define Non-AUTOSAR

**Table 3-33. Define PWM\_E\_PARAM\_PHASESHIFT\_RANGE  
Description**

<b>Name</b>	PWM_E_PARAM_PHASESHIFT_RANGE
<b>Initializer</b>	0x3AU

### 3.10.1.27 Define PWM\_E\_CHANNEL\_PHASE\_SHIFT\_WITHOUT\_COMBINE

Generated when given channel is other than combine channel edge setup (COMBINED\_SYNCED and COMBINED\_COMPLEMENTARY).

**Details:**

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

**Implements:** Pwm\_ErrorIds\_define Non-AUTOSAR

**Table 3-34. Define PWM\_E\_CHANNEL\_PHASE\_SHIFT\_WITHOUT\_COMBINE Description**

<b>Name</b>	PWM_E_CHANNEL_PHASE_SHIFT_WITHOUT_COMBINE
<b>Initializer</b>	0x3BU

### 3.10.1.28 Define PWM\_E\_PARAM\_SYNCHRONOUS\_MODIFIED\_COMBINE

Generated when given channel is Modified Combine channel edge setup (PHASE\_SHIFTED\_SYNCED and PHASE\_SHIFTED\_COMPLEMENTARY).

#### **Details:**

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

**Implements:** Pwm\_ErrorIds\_define Non-AUTOSAR

**Table 3-35. Define PWM\_E\_PARAM\_SYNCHRONOUS\_MODIFIED\_COMBINE Description**

<b>Name</b>	PWM_E_PARAM_SYNCHRONOUS_MODIFIED_COMBINE
<b>Initializer</b>	0x3CU

### 3.10.1.29 Define PWM\_E\_TRIGGER\_MASK

Generated when bit mask is not compatible with hardware register.

#### **Details:**

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

**Implements:** Pwm\_ErrorIds\_define AUTOSAR

**Table 3-36. Define PWM\_E\_TRIGGER\_MASK Description**

<b>Name</b>	PWM_E_TRIGGER_MASK
<b>Initializer</b>	0x3DU



### 3.10.1.30 Define PWM\_E\_NOT\_DISENGAGED

Generated when Pwm\_SetPowerState is called while the PWM module is still in use.

#### Details:

Errors and exceptions that will be detected by the PWM driver

**Implements:** Pwm\_ErrorIds\_define AUTOSAR

**Table 3-37. Define PWM\_E\_NOT\_DISENGAGED Description**

<b>Name</b>	PWM_E_NOT_DISENGAGED
<b>Initializer</b>	0x16U

### 3.10.1.31 Define PWM\_E\_POWER\_STATE\_NOT\_SUPPORTED

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

#### Details:

Errors and exceptions that will be detected by the PWM driver

**Implements:** Pwm\_ErrorIds\_define AUTOSAR

**Table 3-38. Define PWM\_E\_POWER\_STATE\_NOT\_SUPPORTED Description**

<b>Name</b>	PWM_E_POWER_STATE_NOT_SUPPORTED
<b>Initializer</b>	0x17U

### 3.10.1.32 Define PWM\_E\_TRANSITION\_NOT\_POSSIBLE

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

#### Details:

Errors and exceptions that will be detected by the PWM driver

**Implements:** Pwm\_ErrorIds\_define AUTOSAR

**Table 3-39. Define PWM\_E\_TRANSITION\_NOT\_POSSIBLE  
Description**

<b>Name</b>	PWM_E_TRANSITION_NOT_POSSIBLE
<b>Initializer</b>	0x18U

### 3.10.1.33 Define PWM\_E\_PERIPHERAL\_NOT\_PREPARED

Generated when Pwm\_SetPowerState has been called without having called the API.

**Details:**

Errors and exceptions that will be detected by the PWM driver

**Implements:** Pwm\_ErrorIds\_define AUTOSAR

**Table 3-40. Define PWM\_E\_PERIPHERAL\_NOT\_PREPARED  
Description**

<b>Name</b>	PWM_E_PERIPHERAL_NOT_PREPARED
<b>Initializer</b>	0x19U

### 3.10.1.34 Define PWM\_INIT\_ID

API service ID of Pwm\_Init function.

**Details:**

Parameter used to identify the service when reporting and error to DET.

**Table 3-41. Define PWM\_INIT\_ID Description**

<b>Name</b>	PWM_INIT_ID
<b>Initializer</b>	0x00U

### 3.10.1.35 Define PWM\_DEINIT\_ID

API service ID of Pwm\_DeInit function.

#### Details:

Parameter used to identify the service when reporting and error to DET.

**Table 3-42. Define PWM\_DEINIT\_ID Description**

<b>Name</b>	PWM_DEINIT_ID
<b>Initializer</b>	0x01U

### 3.10.1.36 Define PWM\_SETDUTYCYCLE\_ID

API service ID of Pwm\_SetDutyCycle function.

#### Details:

Parameter used to identify the service when reporting and error to DET.

**Table 3-43. Define PWM\_SETDUTYCYCLE\_ID Description**

<b>Name</b>	PWM_SETDUTYCYCLE_ID
<b>Initializer</b>	0x02U

### 3.10.1.37 Define PWM\_SETPERIODANDDUTY\_ID

API service ID of Pwm\_SetPeriodAndDuty function.

#### Details:

Parameter used to identify the service when reporting and error to DET.

**Table 3-44. Define PWM\_SETPERIODANDDUTY\_ID Description**

<b>Name</b>	PWM_SETPERIODANDDUTY_ID
<b>Initializer</b>	0x03U

### 3.10.1.38 Define PWM\_SETOUTPUTTOIDLE\_ID

API service ID of Pwm\_SetOutputToIdle function.

#### Details:

Parameter used to identify the service when reporting and error to DET.

**Table 3-45. Define PWM\_SETOUTPUTTOIDLE\_ID Description**

<b>Name</b>	PWM_SETOUTPUTTOIDLE_ID
<b>Initializer</b>	0x04U

### 3.10.1.39 Define PWM\_GETOUTPUTSTATE\_ID

API service ID of Pwm\_GetOutputState function.

#### Details:

Parameter used to identify the service when reporting and error to DET.

**Note:** In the current implementation this API does NOT reflect the state of the PWM output signal and the returned value is always PWM\_LOW.

**Table 3-46. Define PWM\_GETOUTPUTSTATE\_ID Description**

<b>Name</b>	PWM_GETOUTPUTSTATE_ID
<b>Initializer</b>	0x05U

### 3.10.1.40 Define PWM\_DISABLENOTIFICATION\_ID

API service ID of Pwm\_DisableNotification function.

#### Details:

Parameter used to identify the service when reporting and error to DET.

**Table 3-47. Define PWM\_DISABLENOTIFICATION\_ID Description**

<b>Name</b>	PWM_DISABLENOTIFICATION_ID
<b>Initializer</b>	0x06U

### 3.10.1.41 Define PWM\_ENABLENOTIFICATION\_ID

API service ID of Pwm\_EnableNotification function.

#### Details:

Parameter used to identify the service when reporting and error to DET.

**Table 3-48. Define PWM\_ENABLENOTIFICATION\_ID Description**

<b>Name</b>	PWM_ENABLENOTIFICATION_ID
<b>Initializer</b>	0x07U

### 3.10.1.42 Define PWM\_GETVERSIONINFO\_ID

API service ID of Pwm\_GetVersionInfo function.

#### Details:

Parameter used to identify the service when reporting and error to DET.

**Table 3-49. Define PWM\_GETVERSIONINFO\_ID Description**

<b>Name</b>	PWM_GETVERSIONINFO_ID
<b>Initializer</b>	0x08U

### 3.10.1.43 Define PWM\_SETPOWERSTATE\_ID

API service ID of Pwm\_SetPowerState function.

#### Details:

Parameters used when raising an error/exception

**Table 3-50. Define PWM\_SETPOWERSTATE\_ID  
Description**

<b>Name</b>	PWM_SETPOWERSTATE_ID
<b>Initializer</b>	0x09U

### 3.10.1.44 Define PWM\_GETCURRENTPOWERSTATE\_ID

API service ID of Pwm\_GetCurrentPowerState function.

#### Details:

Parameters used when raising an error/exception

**Table 3-51. Define PWM\_GETCURRENTPOWERSTATE\_ID  
Description**

<b>Name</b>	PWM_GETCURRENTPOWERSTATE_ID
<b>Initializer</b>	0x0AU

### 3.10.1.45 Define PWM\_GETTARGETPOWERSTATE\_ID

API service ID of Pwm\_GetTargetPowerState function.

#### Details:

Parameters used when raising an error/exception

**Table 3-52. Define PWM\_GETTARGETPOWERSTATE\_ID  
Description**

<b>Name</b>	PWM_GETTARGETPOWERSTATE_ID
<b>Initializer</b>	0x0BU

### 3.10.1.46 Define PWM\_PREPAREPOWERSTATE\_ID

API service ID of Pwm\_PreparePowerState function.

**Details:**

Parameters used when raising an error/exception

**Table 3-53. Define PWM\_PREPAREPOWERSTATE\_ID Description**

<b>Name</b>	PWM_PREPAREPOWERSTATE_ID
<b>Initializer</b>	0x0CU

### 3.10.1.47 Define PWM\_GETCHANNELSTATE\_ID

API service ID of Pwm\_GetChannelState function.

**Details:**

Parameters used when raising an error/exception

**Table 3-54. Define PWM\_GETCHANNELSTATE\_ID Description**

<b>Name</b>	PWM_GETCHANNELSTATE_ID
<b>Initializer</b>	0x20U

### 3.10.1.48 Define PWM\_FORCEOUTPUTTOZERO\_ID

API service ID of Pwm\_ForceOutputToZero function.

**Details:**

Parameters used when raising an error/exception

**Table 3-55. Define PWM\_FORCEOUTPUTTOZERO\_ID Description**

<b>Name</b>	PWM_FORCEOUTPUTTOZERO_ID
<b>Initializer</b>	0x21U

### 3.10.1.49 Define PWM\_SETCOUNTERBUS\_ID

API service ID of Pwm\_SetCounterBus function.

#### Details:

Parameter used to identify the service when reporting and error to DET.

**Table 3-56. Define PWM\_SETCOUNTERBUS\_ID Description**

<b>Name</b>	PWM_SETCOUNTERBUS_ID
<b>Initializer</b>	0x22U

### 3.10.1.50 Define PWM\_SETCHANNELOUTPUT\_ID

API service ID of Pwm\_SetChannelOutput function.

#### Details:

Parameter used to identify the service when reporting and error to DET.

**Table 3-57. Define PWM\_SETCHANNELOUTPUT\_ID Description**

<b>Name</b>	PWM_SETCHANNELOUTPUT_ID
<b>Initializer</b>	0x23U

### 3.10.1.51 Define PWM\_SETTRIGGERDELAY\_ID

API service ID of Pwm\_SetTriggerDelay function.

#### Details:

Parameter used to identify the service when reporting and error to DET.

**Table 3-58. Define PWM\_SETTRIGGERDELAY\_ID Description**

<b>Name</b>	PWM_SETTRIGGERDELAY_ID
<b>Initializer</b>	0x24U



### 3.10.1.52 Define PWM\_BUFFERTRANSFERENDIS\_ID

API service ID of Pwm\_BufferTransferEnableDisable function.

#### Details:

Parameter used to identify the service when reporting and error to DET.

**Table 3-59. Define PWM\_BUFFERTRANSFERENDIS\_ID Description**

<b>Name</b>	PWM_BUFFERTRANSFERENDIS_ID
<b>Initializer</b>	0x26U

### 3.10.1.53 Define PWM\_SETCLOCKMODE\_ID

API service ID of Pwm\_SetClockMode function.

#### Details:

Parameters used when raising an error/exception

**Table 3-60. Define PWM\_SETCLOCKMODE\_ID Description**

<b>Name</b>	PWM_SETCLOCKMODE_ID
<b>Initializer</b>	0x27U

### 3.10.1.54 Define PWM\_SYNCUPDATE\_ID

API service ID of Pwm\_SyncUpdate function.

#### Details:

Parameter used to identify the service when reporting and error to DET.

**Table 3-61. Define PWM\_SYNCUPDATE\_ID Description**

<b>Name</b>	PWM_SYNCUPDATE_ID
<b>Initializer</b>	0x28U

### 3.10.1.55 Define PWM\_SETPERIODANDDUTY\_NO\_UPDATE\_ID

API service ID of Pwm\_SetPeriodAndDuty\_NoUpdate function.

#### Details:

Parameter used to identify the service when reporting and error to DET.

**Table 3-62. Define PWM\_SETPERIODANDDUTY\_NO\_UPDATE\_ID Description**

<b>Name</b>	PWM_SETPERIODANDDUTY_NO_UPDATE_ID
<b>Initializer</b>	0x29U

### 3.10.1.56 Define PWM\_SETDUTYCYCLE\_NO\_UPDATE\_ID

API service ID of Pwm\_SetDutyCycle\_NoUpdate function.

#### Details:

Parameter used to identify the service when reporting and error to DET.

**Table 3-63. Define PWM\_SETDUTYCYCLE\_NO\_UPDATE\_ID Description**

<b>Name</b>	PWM_SETDUTYCYCLE_NO_UPDATE_ID
<b>Initializer</b>	0x2AU

### 3.10.1.57 Define PWM\_SETPHASESHIFT\_ID

API service ID of Pwm\_SetPhaseShift function

**Details:**

Parameters used when raising an error/exception

**Table 3-64. Define PWM\_SETPHASESHIFT\_ID  
Description**

<b>Name</b>	PWM_SETPHASESHIFT_ID
<b>Initializer</b>	0x2CU

### 3.10.1.58 Define PWM\_SETPHASESHIFTNOUPDATE\_ID

API service ID of Pwm\_SetPhaseShift\_NoUpdate function

**Details:**

Parameters used when raising an error/exception

**Table 3-65. Define PWM\_SETPHASESHIFTNOUPDATE\_ID  
Description**

<b>Name</b>	PWM_SETPHASESHIFTNOUPDATE_ID
<b>Initializer</b>	0x2DU

### 3.10.1.59 Define PWM\_ENABLETRIGGER\_ID

API service ID of Pwm\_EnableTrigger function

**Details:**

Parameters used when raising an error/exception

**Table 3-66. Define PWM\_ENABLETRIGGER\_ID  
Description**

<b>Name</b>	PWM_ENABLETRIGGER_ID
<b>Initializer</b>	0x2EU

### 3.10.1.60 Define PWM\_DISABLETRIGGER\_ID

API service ID of Pwm\_DisableTrigger function

#### Details:

Parameters used when raising an error/exception

**Table 3-67. Define PWM\_DISABLETRIGGER\_ID  
Description**

<b>Name</b>	PWM_DISABLETRIGGER_ID
<b>Initializer</b>	0x2FU

### 3.10.1.61 Define PWM\_RESETCOUNTERENABLE\_ID

API service ID of Pwm\_ResetCounterEnable function.

#### Details:

Parameters used when raising an error/exception.

**Table 3-68. Define PWM\_RESETCOUNTERENABLE\_ID  
Description**

<b>Name</b>	PWM_RESETCOUNTERENABLE_ID
<b>Initializer</b>	0x30U

### 3.10.1.62 Define PWM\_RESETCOUNTERDISABLE\_ID

API service ID of Pwm\_ResetCounterDisable function.

#### Details:

Parameters used when raising an error/exception.

**Table 3-69. Define PWM\_RESETCOUNTERDISABLE\_ID Description**

<b>Name</b>	PWM_RESETCOUNTERDISABLE_ID
<b>Initializer</b>	0x31U

### 3.10.1.63 Define PWM\_MASKOUTPUT\_ID

API service ID of Pwm\_MaskOutputs function.

#### Details:

Parameters used when raising an error/exception

**Table 3-70. Define PWM\_MASKOUTPUT\_ID Description**

<b>Name</b>	PWM_MASKOUTPUT_ID
<b>Initializer</b>	0x32U

### 3.10.1.64 Define PWM\_UNMASKOUTPUT\_ID

API service ID of Pwm\_UnMaskOutputs function.

#### Details:

Parameters used when raising an error/exception.

**Table 3-71. Define PWM\_UNMASKOUTPUT\_ID Description**

<b>Name</b>	PWM_UNMASKOUTPUT_ID
<b>Initializer</b>	0x33U

### 3.10.1.65 Define PWM\_DISABLERELOADNOTIF\_ID

API service ID of Pwm\_DisableReloadNotification function.

#### Details:

Parameters used when raising an error/exception

**Table 3-72. Define PWM\_DISABLERELOADNOTIF\_ID**  
**Description**

<b>Name</b>	PWM_DISABLERELOADNOTIF_ID
<b>Initializer</b>	0x34U

### 3.10.1.66 Define PWM\_ENABLERELOADNOTIF\_ID

API service ID of Pwm\_EnableReloadNotification function.

#### Details:

Parameters used when raising an error/exception

**Table 3-73. Define PWM\_ENABLERELOADNOTIF\_ID**  
**Description**

<b>Name</b>	PWM_ENABLERELOADNOTIF_ID
<b>Initializer</b>	0x35U

### 3.10.1.67 Define PWM\_FLEXIO\_USED

Macros used to indicate that FlexIO modules is used in current configuration.

**Table 3-74. Define PWM\_FLEXIO\_USED Description**

<b>Name</b>	PWM_FLEXIO_USED
<b>Initializer</b>	(STD_OFF)

### 3.10.1.68 Define PWM\_FTM\_USED

Macros used to indicate that Ftm modules is used in current configuration.

**Table 3-75. Define PWM\_FTM\_USED Description**

<b>Name</b>	PWM_FTM_USED
<b>Initializer</b>	(STD_OFF)

### 3.10.1.69 Define PWM\_FLEXIO\_0\_CH\_0\_1\_USED

Macros used to lock for PWM mode for FlexIO modules in current configuration.

**Table 3-76. Define PWM\_FLEXIO\_0\_CH\_0\_1\_USED Description**

<b>Name</b>	PWM_FLEXIO_0_CH_0_1_USED
<b>Initializer</b>	(0U)

### 3.10.1.70 Define PWM\_FLEXIO\_0\_CH\_2\_3\_USED

Macros used to lock for PWM mode for FlexIO modules in current configuration.

**Table 3-77. Define PWM\_FLEXIO\_0\_CH\_2\_3\_USED Description**

<b>Name</b>	PWM_FLEXIO_0_CH_2_3_USED
<b>Initializer</b>	(0U)

### 3.10.1.71 Define PWM\_FLEXIO\_0\_CH\_0\_1\_ISR\_USED

Macros used to enable ISR for FlexIO module 0.

**Table 3-78. Define PWM\_FLEXIO\_0\_CH\_0\_1\_ISR\_USED Description**

<b>Name</b>	PWM_FLEXIO_0_CH_0_1_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.72 Define PWM\_FLEXIO\_0\_CH\_2\_3\_ISR\_USED

Macros used to enable ISR for FlexIO module 0.

**Table 3-79. Define PWM\_FLEXIO\_0\_CH\_2\_3\_ISR\_USED Description**

<b>Name</b>	PWM_FLEXIO_0_CH_2_3_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.73 Define PWM\_FLEXIO\_0\_PIN\_0\_USED

Macros used to check conflict Pinout for FlexIO module 0.

**Table 3-80. Define PWM\_FLEXIO\_0\_PIN\_0\_USED**  
**Description**

<b>Name</b>	PWM_FLEXIO_0_PIN_0_USED
<b>Initializer</b>	(0U)

### 3.10.1.74 Define PWM\_FLEXIO\_0\_PIN\_1\_USED

Macros used to check conflict Pinout for FlexIO module 0.

**Table 3-81. Define PWM\_FLEXIO\_0\_PIN\_1\_USED**  
**Description**

<b>Name</b>	PWM_FLEXIO_0_PIN_1_USED
<b>Initializer</b>	(0U)

### 3.10.1.75 Define PWM\_FLEXIO\_0\_PIN\_2\_USED

Macros used to check conflict Pinout for FlexIO module 0.

**Table 3-82. Define PWM\_FLEXIO\_0\_PIN\_2\_USED**  
**Description**

<b>Name</b>	PWM_FLEXIO_0_PIN_2_USED
<b>Initializer</b>	(0U)

### 3.10.1.76 Define PWM\_FLEXIO\_0\_PIN\_3\_USED

Macros used to check conflict Pinout for FlexIO module 0.

**Table 3-83. Define PWM\_FLEXIO\_0\_PIN\_3\_USED**  
**Description**

<b>Name</b>	PWM_FLEXIO_0_PIN_3_USED
<b>Initializer</b>	(0U)



### 3.10.1.77 Define PWM\_FLEXIO\_0\_PIN\_4\_USED

Macros used to check conflict Pinout for FlexIO module 0.

**Table 3-84. Define PWM\_FLEXIO\_0\_PIN\_4\_USED**  
**Description**

<b>Name</b>	PWM_FLEXIO_0_PIN_4_USED
<b>Initializer</b>	(0U)

### 3.10.1.78 Define PWM\_FLEXIO\_0\_PIN\_5\_USED

Macros used to check conflict Pinout for FlexIO module 0.

**Table 3-85. Define PWM\_FLEXIO\_0\_PIN\_5\_USED**  
**Description**

<b>Name</b>	PWM_FLEXIO_0_PIN_5_USED
<b>Initializer</b>	(0U)

### 3.10.1.79 Define PWM\_FLEXIO\_0\_PIN\_6\_USED

Macros used to check conflict Pinout for FlexIO module 0.

**Table 3-86. Define PWM\_FLEXIO\_0\_PIN\_6\_USED**  
**Description**

<b>Name</b>	PWM_FLEXIO_0_PIN_6_USED
<b>Initializer</b>	(0U)

### 3.10.1.80 Define PWM\_FLEXIO\_0\_PIN\_7\_USED

Macros used to check conflict Pinout for FlexIO module 0.

**Table 3-87. Define PWM\_FLEXIO\_0\_PIN\_7\_USED**  
**Description**

<b>Name</b>	PWM_FLEXIO_0_PIN_7_USED
<b>Initializer</b>	(0U)

### 3.10.1.81 Define PWM\_FTM\_0\_USED

Macros used to lock for PWM mode FTM modules in current configuration.

**Table 3-88. Define PWM\_FTM\_0\_USED Description**

<b>Name</b>	PWM_FTM_0_USED
<b>Initializer</b>	(0U)

### 3.10.1.82 Define PWM\_FTM\_1\_USED

Macros used to lock for PWM mode FTM modules in current configuration.

**Table 3-89. Define PWM\_FTM\_1\_USED Description**

<b>Name</b>	PWM_FTM_1_USED
<b>Initializer</b>	(0U)

### 3.10.1.83 Define PWM\_FTM\_2\_USED

Macros used to lock for PWM mode FTM modules in current configuration.

**Table 3-90. Define PWM\_FTM\_2\_USED Description**

<b>Name</b>	PWM_FTM_2_USED
<b>Initializer</b>	(0U)

### 3.10.1.84 Define PWM\_FTM\_3\_USED

Macros used to lock for PWM mode FTM modules in current configuration.

**Table 3-91. Define PWM\_FTM\_3\_USED Description**

<b>Name</b>	PWM_FTM_3_USED
<b>Initializer</b>	(0U)

### 3.10.1.85 Define PWM\_FTM\_4\_USED

Macros used to lock for PWM mode FTM modules in current configuration.

**Table 3-92. Define PWM\_FTM\_4\_USED Description**

<b>Name</b>	PWM_FTM_4_USED
<b>Initializer</b>	(0U)

### 3.10.1.86 Define PWM\_FTM\_5\_USED

Macros used to lock for PWM mode FTM modules in current configuration.

**Table 3-93. Define PWM\_FTM\_5\_USED Description**

<b>Name</b>	PWM_FTM_5_USED
<b>Initializer</b>	(0U)

### 3.10.1.87 Define PWM\_FTM\_6\_USED

Macros used to lock for PWM mode FTM modules in current configuration.

**Table 3-94. Define PWM\_FTM\_6\_USED Description**

<b>Name</b>	PWM_FTM_6_USED
<b>Initializer</b>	(0U)

### 3.10.1.88 Define PWM\_FTM\_7\_USED

Macros used to lock for PWM mode FTM modules in current configuration.

**Table 3-95. Define PWM\_FTM\_7\_USED Description**

<b>Name</b>	PWM_FTM_7_USED
<b>Initializer</b>	(0U)

### 3.10.1.89 Define PWM\_FTM\_0\_CH\_0\_CH\_1\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-96. Define PWM\_FTM\_0\_CH\_0\_CH\_1\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_0_CH_0_CH_1_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.90 Define PWM\_FTM\_0\_CH\_2\_CH\_3\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-97. Define PWM\_FTM\_0\_CH\_2\_CH\_3\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_0_CH_2_CH_3_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.91 Define PWM\_FTM\_0\_CH\_4\_CH\_5\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-98. Define PWM\_FTM\_0\_CH\_4\_CH\_5\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_0_CH_4_CH_5_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.92 Define PWM\_FTM\_0\_CH\_6\_CH\_7\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-99. Define PWM\_FTM\_0\_CH\_6\_CH\_7\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_0_CH_6_CH_7_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.93 Define PWM\_FTM\_0\_OVF\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-100. Define PWM\_FTM\_0\_OVF\_ISR\_USED**  
Description

<b>Name</b>	PWM_FTM_0_OVF_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.94 Define PWM\_FTM\_0\_FAULT\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-101. Define PWM\_FTM\_0\_FAULT\_ISR\_USED**  
Description

<b>Name</b>	PWM_FTM_0_FAULT_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.95 Define PWM\_FTM\_1\_CH\_0\_CH\_1\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-102. Define PWM\_FTM\_1\_CH\_0\_CH\_1\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_1_CH_0_CH_1_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.96 Define PWM\_FTM\_1\_CH\_2\_CH\_3\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-103. Define PWM\_FTM\_1\_CH\_2\_CH\_3\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_1_CH_2_CH_3_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.97 Define PWM\_FTM\_1\_CH\_4\_CH\_5\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-104. Define PWM\_FTM\_1\_CH\_4\_CH\_5\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_1_CH_4_CH_5_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.98 Define PWM\_FTM\_1\_CH\_6\_CH\_7\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-105. Define PWM\_FTM\_1\_CH\_6\_CH\_7\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_1_CH_6_CH_7_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.99 Define PWM\_FTM\_1\_OVF\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-106. Define PWM\_FTM\_1\_OVF\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_1_OVF_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.100 Define PWM\_FTM\_1\_FAULT\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-107. Define PWM\_FTM\_1\_FAULT\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_1_FAULT_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.101 Define PWM\_FTM\_2\_CH\_0\_CH\_1\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-108. Define PWM\_FTM\_2\_CH\_0\_CH\_1\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_2_CH_0_CH_1_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.102 Define PWM\_FTM\_2\_CH\_2\_CH\_3\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-109. Define PWM\_FTM\_2\_CH\_2\_CH\_3\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_2_CH_2_CH_3_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.103 Define PWM\_FTM\_2\_CH\_4\_CH\_5\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-110. Define PWM\_FTM\_2\_CH\_4\_CH\_5\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_2_CH_4_CH_5_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.104 Define PWM\_FTM\_2\_CH\_6\_CH\_7\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-111. Define PWM\_FTM\_2\_CH\_6\_CH\_7\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_2_CH_6_CH_7_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.105 Define PWM\_FTM\_2\_OVF\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-112. Define PWM\_FTM\_2\_OVF\_ISR\_USED**  
Description

<b>Name</b>	PWM_FTM_2_OVF_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.106 Define PWM\_FTM\_2\_FAULT\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-113. Define PWM\_FTM\_2\_FAULT\_ISR\_USED**  
Description

<b>Name</b>	PWM_FTM_2_FAULT_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.107 Define PWM\_FTM\_3\_CH\_0\_CH\_1\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-114. Define PWM\_FTM\_3\_CH\_0\_CH\_1\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_3_CH_0_CH_1_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.108 Define PWM\_FTM\_3\_CH\_2\_CH\_3\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-115. Define PWM\_FTM\_3\_CH\_2\_CH\_3\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_3_CH_2_CH_3_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.109 Define PWM\_FTM\_3\_CH\_4\_CH\_5\_ISR\_USED

Macros used to enable FTM Notification ISR code.



**Table 3-116. Define PWM\_FTM\_3\_CH\_4\_CH\_5\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_3_CH_4_CH_5_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.110 Define PWM\_FTM\_3\_CH\_6\_CH\_7\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-117. Define PWM\_FTM\_3\_CH\_6\_CH\_7\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_3_CH_6_CH_7_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.111 Define PWM\_FTM\_3\_OVF\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-118. Define PWM\_FTM\_3\_OVF\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_3_OVF_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.112 Define PWM\_FTM\_3\_FAULT\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-119. Define PWM\_FTM\_3\_FAULT\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_3_FAULT_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.113 Define PWM\_FTM\_4\_CH\_0\_CH\_1\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-120. Define PWM\_FTM\_4\_CH\_0\_CH\_1\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_4_CH_0_CH_1_ISR_USED
<b>Initializer</b>	(0U)

**3.10.1.114 Define PWM\_FTM\_4\_CH\_2\_CH\_3\_ISR\_USED**

Macros used to enable FTM Notification ISR code.

**Table 3-121. Define PWM\_FTM\_4\_CH\_2\_CH\_3\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_4_CH_2_CH_3_ISR_USED
<b>Initializer</b>	(0U)

**3.10.1.115 Define PWM\_FTM\_4\_CH\_4\_CH\_5\_ISR\_USED**

Macros used to enable FTM Notification ISR code.

**Table 3-122. Define PWM\_FTM\_4\_CH\_4\_CH\_5\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_4_CH_4_CH_5_ISR_USED
<b>Initializer</b>	(0U)

**3.10.1.116 Define PWM\_FTM\_4\_CH\_6\_CH\_7\_ISR\_USED**

Macros used to enable FTM Notification ISR code.

**Table 3-123. Define PWM\_FTM\_4\_CH\_6\_CH\_7\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_4_CH_6_CH_7_ISR_USED
<b>Initializer</b>	(0U)

**3.10.1.117 Define PWM\_FTM\_4\_OVF\_ISR\_USED**

Macros used to enable FTM Notification ISR code.

**Table 3-124. Define PWM\_FTM\_4\_OVF\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_4_OVF_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.118 Define PWM\_FTM\_4\_FAULT\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-125. Define PWM\_FTM\_4\_FAULT\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_4_FAULT_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.119 Define PWM\_FTM\_5\_CH\_0\_CH\_1\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-126. Define PWM\_FTM\_5\_CH\_0\_CH\_1\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_5_CH_0_CH_1_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.120 Define PWM\_FTM\_5\_CH\_2\_CH\_3\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-127. Define PWM\_FTM\_5\_CH\_2\_CH\_3\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_5_CH_2_CH_3_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.121 Define PWM\_FTM\_5\_CH\_4\_CH\_5\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-128. Define PWM\_FTM\_5\_CH\_4\_CH\_5\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_5_CH_4_CH_5_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.122 Define PWM\_FTM\_5\_CH\_6\_CH\_7\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-129. Define PWM\_FTM\_5\_CH\_6\_CH\_7\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_5_CH_6_CH_7_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.123 Define PWM\_FTM\_5\_OVF\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-130. Define PWM\_FTM\_5\_OVF\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_5_OVF_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.124 Define PWM\_FTM\_5\_FAULT\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-131. Define PWM\_FTM\_5\_FAULT\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_5_FAULT_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.125 Define PWM\_FTM\_6\_CH\_0\_CH\_1\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-132. Define PWM\_FTM\_6\_CH\_0\_CH\_1\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_6_CH_0_CH_1_ISR_USED
<b>Initializer</b>	(0U)

**3.10.1.126 Define PWM\_FTM\_6\_CH\_2\_CH\_3\_ISR\_USED**

Macros used to enable FTM Notification ISR code.

**Table 3-133. Define PWM\_FTM\_6\_CH\_2\_CH\_3\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_6_CH_2_CH_3_ISR_USED
<b>Initializer</b>	(0U)

**3.10.1.127 Define PWM\_FTM\_6\_CH\_4\_CH\_5\_ISR\_USED**

Macros used to enable FTM Notification ISR code.

**Table 3-134. Define PWM\_FTM\_6\_CH\_4\_CH\_5\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_6_CH_4_CH_5_ISR_USED
<b>Initializer</b>	(0U)

**3.10.1.128 Define PWM\_FTM\_6\_CH\_6\_CH\_7\_ISR\_USED**

Macros used to enable FTM Notification ISR code.

**Table 3-135. Define PWM\_FTM\_6\_CH\_6\_CH\_7\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_6_CH_6_CH_7_ISR_USED
<b>Initializer</b>	(0U)

**3.10.1.129 Define PWM\_FTM\_6\_OVF\_ISR\_USED**

Macros used to enable FTM Notification ISR code.

**Table 3-136. Define PWM\_FTM\_6\_OVF\_ISR\_USED**  
Description

<b>Name</b>	PWM_FTM_6_OVF_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.130 Define PWM\_FTM\_6\_FAULT\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-137. Define PWM\_FTM\_6\_FAULT\_ISR\_USED**  
Description

<b>Name</b>	PWM_FTM_6_FAULT_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.131 Define PWM\_FTM\_7\_CH\_0\_CH\_1\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-138. Define PWM\_FTM\_7\_CH\_0\_CH\_1\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_7_CH_0_CH_1_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.132 Define PWM\_FTM\_7\_CH\_2\_CH\_3\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-139. Define PWM\_FTM\_7\_CH\_2\_CH\_3\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_7_CH_2_CH_3_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.133 Define PWM\_FTM\_7\_CH\_4\_CH\_5\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-140. Define PWM\_FTM\_7\_CH\_4\_CH\_5\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_7_CH_4_CH_5_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.134 Define PWM\_FTM\_7\_CH\_6\_CH\_7\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-141. Define PWM\_FTM\_7\_CH\_6\_CH\_7\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_7_CH_6_CH_7_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.135 Define PWM\_FTM\_7\_OVF\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-142. Define PWM\_FTM\_7\_OVF\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_7_OVF_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.136 Define PWM\_FTM\_7\_FAULT\_ISR\_USED

Macros used to enable FTM Notification ISR code.

**Table 3-143. Define PWM\_FTM\_7\_FAULT\_ISR\_USED Description**

<b>Name</b>	PWM_FTM_7_FAULT_ISR_USED
<b>Initializer</b>	(0U)

### 3.10.1.137 Define PWM\_FTM\_CHANNEL

Symbolic name for FTM channels.

**Table 3-144. Define PWM\_FTM\_CHANNEL Description**

<b>Name</b>	PWM_FTM_CHANNEL
<b>Initializer</b>	((Pwm_ChannelIpType) 0)

### 3.10.1.138 Define PWM\_FLEXIO\_CHANNEL

Symbolic name for FTM channels.

**Table 3-145. Define PWM\_FLEXIO\_CHANNEL Description**

<b>Name</b>	PWM_FLEXIO_CHANNEL
<b>Initializer</b>	((Pwm_ChannelIpType) 1)

### 3.10.1.139 Define PWM\_FTM\_CHANNELS\_NO

Defines the maximum number of FTM channels in all existing configurations.

**Table 3-146. Define PWM\_FTM\_CHANNELS\_NO Description**

<b>Name</b>	PWM_FTM_CHANNELS_NO
<b>Initializer</b>	Dependent on derivative

### 3.10.1.140 Define PWM\_FTM\_CHANNELS\_MAX\_U8

Defines the number of FTM channels used in all existing configurations.

**Table 3-147. Define PWM\_FTM\_CHANNELS\_MAX\_U8 Description**

<b>Name</b>	PWM_FTM_CHANNELS_MAX_U8
<b>Initializer</b>	(8U)

### 3.10.1.141 Define PWM\_FTM\_MODULE\_NO

This define specifies the number of FTM Modules available in all existing configuration.

**Table 3-148. Define PWM\_FTM\_MODULE\_NO Description**

<b>Name</b>	PWM_FTM_MODULE_NO
-------------	-------------------

*Table continues on the next page...*



**Table 3-148. Define PWM\_FTM\_MODULE\_NO Description  
(continued)**

<b>Initializer</b>	Dependent on derivative
--------------------	-------------------------

### 3.10.1.142 Define PWM\_FLEXIO\_MODULE\_NO

This define specifies the number of FlexIO modules available in all existing configuration.

**Table 3-149. Define PWM\_FLEXIO\_MODULE\_NO  
Description**

<b>Name</b>	PWM_FLEXIO_MODULE_NO
<b>Initializer</b>	(1U)

### 3.10.1.143 Define PWM\_FLEXIO\_CHANNEL\_NO

Defines the maximum number of channels available to be configured in all existing FlexIO modules.

**Table 3-150. Define PWM\_FLEXIO\_CHANNEL\_NO Description**

<b>Name</b>	PWM_FLEXIO_CHANNEL_NO
<b>Initializer</b>	(0U)

### 3.10.1.144 Define PWM\_FLEXIO\_CHANNELS\_MAX\_U8

Defines the maximum number of channels available to be configured in all existing FlexIO modules.

**Table 3-151. Define PWM\_FLEXIO\_CHANNELS\_MAX\_U8  
Description**

<b>Name</b>	PWM_FLEXIO_CHANNELS_MAX_U8
<b>Initializer</b>	(8U)

### 3.10.1.145 Define PWM\_HW\_CHANNELS\_NO\_U8

Defines the maximum number of hardware channels configurable on this platform.

**Table 3-152. Define PWM\_HW\_CHANNELS\_NO Description**

<b>Name</b>	PWM_HW_CHANNELS_NO_U8
<b>Initializer</b>	(32U)

### 3.10.1.146 Define PWM\_FTM\_MODULE\_CHANNELS\_NO

Define specifies the number of channels per each module.

**Table 3-153. Define PWM\_FTM\_MODULE\_CHANNELS\_NO Description**

<b>Name</b>	PWM_FTM_MODULE_CHANNELS_NO
<b>Initializer</b>	(8U)

### 3.10.1.147 Define PWM\_FTM\_MODULE\_FAULT\_NO

This define specifies the number of fault channels per module.

**Table 3-154. Define PWM\_FTM\_MODULE\_FAULT\_NO Description**

<b>Name</b>	PWM_FTM_MODULE_FAULT_NO
<b>Initializer</b>	(4U)

### 3.10.1.148 Define PWM\_DEV\_ERROR\_DETECT

Switch for enabling the development error detection.

**Table 3-155. Define PWM\_DEV\_ERROR\_DETECT Description**

<b>Name</b>	PWM_DEV_ERROR_DETECT
<b>Initializer</b>	(STD_ON)

### 3.10.1.149 Define PWM\_DUTY\_PERIOD\_UPDATED\_ENDPERIOD

Switch for enabling the update of the period parameter at the end of the current period.

**Table 3-156. Define PWM\_DUTY\_PERIOD\_UPDATED\_ENDPERIOD**  
**Description**

<b>Name</b>	PWM_DUTY_PERIOD_UPDATED_ENDPERIOD
<b>Initializer</b>	(STD_ON)

### 3.10.1.150 Define PWM\_DUTYCYCLE\_UPDATED\_ENDPERIOD

Switch for enabling the update of the duty cycle parameter at the end of the current period.

**Table 3-157. Define PWM\_DUTYCYCLE\_UPDATED\_ENDPERIOD**  
**Description**

<b>Name</b>	PWM_DUTYCYCLE_UPDATED_ENDPERIOD
<b>Initializer</b>	(STD_ON)

### 3.10.1.151 Define PWM\_FAULT\_SUPPORTED

Switch for enabling the fault functionality.

**Table 3-158. Define PWM\_FAULT\_SUPPORTED**  
**Description**

<b>Name</b>	PWM_FAULT_SUPPORTED
<b>Initializer</b>	(STD_OFF)

### 3.10.1.152 Define PWM\_INDEX

Specifies the InstanceId of this module instance.

#### Details:

Specifies the InstanceId of this module instance. If only one instance is present it shall have the Id 0. Not used in the current implementation

**Table 3-159. Define PWM\_INDEX Description**

<b>Name</b>	PWM_INDEX
<b>Initializer</b>	(0U)

### 3.10.1.153 Define PWM\_NOTIFICATION\_SUPPORTED

Switch to indicate that the notifications are supported.

**Table 3-160. Define PWM\_NOTIFICATION\_SUPPORTED Description**

<b>Name</b>	PWM_NOTIFICATION_SUPPORTED
<b>Initializer</b>	(STD_ON)

### 3.10.1.154 Define PWM\_PRECOMPILE\_SUPPORT

PWM pre-compile switch.

**Table 3-161. Define PWM\_PRECOMPILE\_SUPPORT Description**

<b>Name</b>	PWM_PRECOMPILE_SUPPORT
<b>Initializer</b>	(STD_OFF)

### 3.10.1.155 Define PWM\_FTM\_ENABLE\_EXT\_TRIGGERS

Switch to indicate that PWM\_FTM\_ENABLE\_EXT\_TRIGGERS is supported.

**Table 3-162. Define PWM\_FTM\_ENABLE\_EXT\_TRIGGERS Description**

<b>Name</b>	PWM_FTM_ENABLE_EXT_TRIGGERS
<b>Initializer</b>	(STD_OFF)

### 3.10.1.156 Define PWM\_SET\_DUTY\_CYCLE\_API

Switch to indicate that Pwm\_SetDutyCycle API is supported.

**Table 3-163. Define PWM\_SET\_DUTY\_CYCLE\_API**  
Description

<b>Name</b>	PWM_SET_DUTY_CYCLE_API
<b>Initializer</b>	(STD_ON)

### 3.10.1.157 Define PWM\_SET\_OUTPUT\_TO\_IDLE\_API

Switch to indicate that Pwm\_SetOutputToIdle API is supported.

**Table 3-164. Define PWM\_SET\_OUTPUT\_TO\_IDLE\_API**  
Description

<b>Name</b>	PWM_SET_OUTPUT_TO_IDLE_API
<b>Initializer</b>	(STD_ON)

### 3.10.1.158 Define PWM\_SET\_PERIOD\_AND\_DUTY\_API

Switch to indicate that Pwm\_SetPeriodAndDuty API is supported.

**Table 3-165. Define PWM\_SET\_PERIOD\_AND\_DUTY\_API**  
Description

<b>Name</b>	PWM_SET_PERIOD_AND_DUTY_API
<b>Initializer</b>	(STD_ON)

### 3.10.1.159 Define PWM\_SET\_CLOCK\_MODE\_API

Switch to indicate that Pwm\_SetClockMode API is supported. This API will allow selection of the prescaler from two configured values.

**Table 3-166. Define PWM\_SET\_CLOCK\_MODE\_API**  
Description

<b>Name</b>	PWM_SET_CLOCK_MODE_API
<b>Initializer</b>	(STD_ON)

### 3.10.1.160 Define PWM\_VERSION\_INFO\_API

Switch to indicate that Pwm\_GetVersionInfo API is supported.

**Table 3-167. Define PWM\_VERSION\_INFO\_API Description**

<b>Name</b>	PWM_VERSION_INFO_API
<b>Initializer</b>	(STD_ON)

### 3.10.1.161 Define PWM\_GET\_CHANNEL\_STATE\_API

Switch to indicate that Pwm\_GetChannelState API is supported.

**Table 3-168. Define PWM\_GET\_CHANNEL\_STATE\_API Description**

<b>Name</b>	PWM_GET_CHANNEL_STATE_API
<b>Initializer</b>	(STD_OFF)

### 3.10.1.162 Define PWM\_GET\_OUTPUT\_STATE\_API

Switch to indicate that Pwm\_GetOutputState API is supported.

#### Note

Due to hardware restrictions this service is disabled at compile time.

**Table 3-169. Define PWM\_GET\_OUTPUT\_STATE\_API Description**

<b>Name</b>	PWM_GET_OUTPUT_STATE_API
<b>Initializer</b>	(STD_OFF)

### 3.10.1.163 Define PWM\_FORCE\_OUTPUT\_TO\_ZERO\_API

Switch to indicate that Pwm\_ForceOutputToZero API is supported.

**Table 3-170. Define PWM\_FORCE\_OUTPUT\_TO\_ZERO\_API Description**

<b>Name</b>	PWM_FORCE_OUTPUT_TO_ZERO_API
<b>Initializer</b>	(STD_OFF)

### 3.10.1.164 Define PWM\_DE\_INIT\_API

Switch to indicate that Pwm\_DeInit API is supported.

**Table 3-171. Define PWM\_DE\_INIT\_API Description**

<b>Name</b>	PWM_DE_INIT_API
<b>Initializer</b>	(STD_ON)

### 3.10.1.165 Define PWM\_SET\_PHASE\_SHIFT\_API

Switch for enabling Pwm\_SetPhaseShift API.

**Table 3-172. Define PWM\_SET\_PHASE\_SHIFT\_API Description**

<b>Name</b>	PWM_SET_PHASE_SHIFT_API
<b>Initializer</b>	(STD_ON)

### 3.10.1.166 Define PWM\_SET\_PHASE\_SHIFT\_NO\_UPDATE\_API

Switch for enabling Pwm\_SetPhaseShift\_NoUpdate API.

**Table 3-173. Define PWM\_SET\_PHASE\_SHIFT\_NO\_UPDATE\_API Description**

<b>Name</b>	PWM_SET_PHASE_SHIFT_NO_UPDATE_API
<b>Initializer</b>	(STD_ON)

### 3.10.1.167 Define PWM\_ENABLE\_TRIGEER\_API

Switch for enabling Pwm\_EnableTrigger API.

**Table 3-174. Define PWM\_ENABLE\_TRIGEER\_API Description**

<b>Name</b>	PWM_ENABLE_TRIGEER_API
<b>Initializer</b>	(STD_ON)

### 3.10.1.168 Define PWM\_DIABLE\_TRIGEER\_API

Switch for enabling Pwm\_DisableTrigger API.

**Table 3-175. Define PWM\_DIABLE\_TRIGEER\_API  
Description**

<b>Name</b>	PWM_DIABLE_TRIGEER_API
<b>Initializer</b>	(STD_ON)

### 3.10.1.169 Define PWM\_RESET\_COUNTER\_API

Switch to indicate that Pwm\_SwResetCounter API is supported.

**Table 3-176. Define PWM\_RESET\_COUNTER\_API  
Description**

<b>Name</b>	PWM_RESET_COUNTER_API
<b>Initializer</b>	(STD_ON)

### 3.10.1.170 Define PWM\_ENABLE\_MASKING\_OPERATIONS

Switch for enabling MaskOutput API.

**Table 3-177. Define PWM\_ENABLE\_MASKING\_OPERATIONS  
Description**

<b>Name</b>	PWM_ENABLE_MASKING_OPERATIONS
<b>Initializer</b>	(STD_ON)

### 3.10.1.171 Define PWM\_SYNC\_UPDATE\_API

Switch to indicate that Pwm\_SyncUpdate API is supported.

**Table 3-178. Define PWM\_SYNC\_UPDATE\_API  
Description**

<b>Name</b>	PWM_SYNC_UPDATE_API
<b>Initializer</b>	(STD_ON)



### 3.10.1.172 Define PWM\_UPDATE\_DUTY\_SYNCHRONOUS

Switch to indicate that the notifications are supported.

**Table 3-179. Define PWM\_UPDATE\_DUTY\_SYNCHRONOUS  
Description**

<b>Name</b>	PWM_UPDATE_DUTY_SYNCHRONOUS
<b>Initializer</b>	(STD_ON)

### 3.10.1.173 Define PWM\_SET\_DUTY\_CYCLE\_NO\_UPDATE\_API

Switch to indicate that PwmSetDutyCycle\_NoUpdate API is supported.

**Table 3-180. Define PWM\_SET\_DUTY\_CYCLE\_NO\_UPDATE\_API  
Description**

<b>Name</b>	PWM_SET_DUTY_CYCLE_NO_UPDATE_API
<b>Initializer</b>	(STD_ON)

### 3.10.1.174 Define PWM\_SET\_PERIOD\_AND\_DUTY\_NO\_UPDATE\_API

Switch to indicate that PwmSetPeriodAndDuty\_NoUpdate API is supported.

**Table 3-181. Define PWM\_SET\_PERIOD\_AND\_DUTY\_NO\_UPDATE\_API  
Description**

<b>Name</b>	PWM_SET_PERIOD_AND_DUTY_NO_UPDATE_API
<b>Initializer</b>	(STD_ON)

### 3.10.1.175 Define PWM\_ENABLE\_PHASE\_SHIFT

Switch for enabling phase shift feature.

**Table 3-182. Define PWM\_ENABLE\_PHASE\_SHIFT  
Description**

<b>Name</b>	PWM_ENABLE_PHASE_SHIFT
<b>Initializer</b>	(STD_OFF)

## 3.10.2 Enum Reference

Enumeration of all constants supported by the driver are as per AUTOSAR PWM Driver software specification Version 4.2 Rev0002 .

### 3.10.2.1 Enumeration Pwm\_ChannelClassType

PWM channel type.

#### Details:

Parameter used to identify the service when reporting and error to DET. PWM channel type.

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

**Implements:** Pwm\_ChannelClassType\_enumeration

**Table 3-183. Enumeration Pwm\_ChannelClassType Values**

Name	Initializer	Description
PWM_VARIABLE_PERIOD	0	The period and duty cycle can be altered.
PWM_FIXED_PERIOD	1	Only the duty cycle can be altered.
PWM_FIXED_PERIOD_SHIFTED	2	Only the duty cycle can be altered.

### 3.10.2.2 Enumeration Pwm\_OutputStateType

Output signal level.

#### Details:

This enumeration specifies the return type of Pwm\_GetOutputState.

#### **Note**

Due to a hardware limitation, calls to Pwm\_GetOutputState will always return PWM\_LOW.

**Implements:** Pwm\_OutputStateType\_enumeration

**Table 3-184. Enumeration Pwm\_OutputStateType Values**

Name	Initializer	Description
PWM_LOW	0	PWM level is logic low.
PWM_HIGH	1	PWM level is logic high.

### 3.10.2.3 Enumeration Pwm\_EdgeNotificationType

Edge notification type.

**Details:**

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

**Implements:** Pwm\_EdgeNotificationType\_enumeration

**Table 3-185. Enumeration Pwm\_EdgeNotificationType Values**

Name	Initializer	Description
PWM_RISING_EDGE	1	A notification will be generated on the rising edge.
PWM_FALLING_EDGE	2	A notification will be generated on the falling edge.
PWM_BOTH_EDGES	3	A notification will be generated on any state transition.

### 3.10.2.4 Enumeration Pwm\_PrescalerType

Pre-scaler type.

**Details:**

This enumeration specifies the input parameter of Pwm\_SetClockMode to configure base-clock timers.

**Note**

Only two types of pre-scalers can be used.

**Implements:** Pwm\_PrescalerType\_enumeration

**Table 3-186. Enumeration Pwm\_PrescalerType Values**

Name	Initializer	Description
PWM_PRIMARY_PRESCALER	0	Primary (default) pre-scaler value.
PWM_ALTERNATIVE_PRESCALER	1	Alternative value of the pre-scaler.

### 3.10.2.5 Enumeration Pwm\_GlobalStateType

Enum containing the possible states of the PWM driver.

**Table 3-187. Enumeration Pwm\_GlobalStateType Values**

Name	Initializer	Description
PWM_STATE_UNINIT	0	Return state Uninitialize of PWM driver
PWM_STATE_IDLE	1	Return state Idle of PWM driver

### 3.10.2.6 Enumeration Pwm\_AlignmentType

PWM signal alignment. This parameter is applied at a PWM submodule level and is available only for FTM.

#### Details:

This field will vary the alignment of the output signals for the specified channel.

- If the selected mode is PWM\_EDGE\_ALIGNED the signals will be aligned at the starting edge of the PWM period.
- If the selected mode is PWM\_CENTER\_ALIGNED the signals will be aligned around the middle of the PWM period.

**Implements:** Pwm\_AlignmentType\_enumeration

**Table 3-188. Enumeration Pwm\_AlignmentType Values**

Name	Initializer	Description
PWM_EDGE_ALIGNED	0	One signal is generated that is aligned at the starting edge of the PWM period. This is the default generation mode
PWM_CENTER_ALIGNED	1	One signal is generated that is aligned around the middle of the PWM period.
PWM_COMBINE_SYNCED	2	Two channels are combined into two signals with the same polarity but can be phase shifted from the master channel by a configurable deadtime; first channel defines the leading edge of the signal, second channel defines the trailing edge of the signal.

*Table continues on the next page...*

**Table 3-188. Enumeration Pwm\_AlignmentType Values (continued)**

Name	Initializer	Description
PWM_COMBINE_COMPL	3	Two channels are combined into two signals with the same polarity which can be phase shifted from the master channel by a configurable deadtime; first channel defines the leading edge of the signal, second channel defines the trailing edge of the signal.
PWM_PHASE_SHIFT_SINGLE	4	Two channels are paired into one output signal; first channel defines the leading edge of the signal, second channel defines the trailing edge of the signal.
PWM_PHASE_SHIFTED_SYNCED	5	Two channels are paired into two-synced output signals; for both signals the first channel defines the leading edge of the signal, second channel defines the trailing edge of the signal.
PWM_PHASE_SHIFTED_COMPLEMENTARY	6	Two channels are paired into two-complementary output signals; for both signals the first channel defines the leading edge of the signal, second channel defines the trailing edge of the signal.

### 3.10.2.7 Enumeration Pwm\_PowerStateRequestResultType

Output signal level.

#### **Details:**

Result of the requests related to power state transitions

**Implements:** Pwm\_PowerStateRequestResultType\_enumeration

**Table 3-189. Enumeration Pwm\_PowerStateRequestResultType Values**

Name	Initializer	Description
PWM_SERVICE_ACCEPTED	0	Power state change executed.
PWM_NOT_INIT	1	Module not initialized.
PWM_SEQUENCE_ERROR	2	Wrong API call sequence.
PWM_HW_FAILURE	3	The HW module has a failure which prevents it to enter the required power state.
PWM_POWER_STATE_NOT_SUPP	4	Module does not support the requested power state.
PWM_TRANS_NOT_POSSIBLE	5	Module cannot transition directly from the current power state to the requested power state.

### 3.10.2.8 Enumeration Pwm\_PowerStateType

Output signal level.

#### Details:

This enum specifies Power state currently active or set as target power state

**Implements:** Pwm\_PowerStateType\_enumeration

**Table 3-190. Enumeration Pwm\_PowerStateType Values**

Name	Initializer	Description
PWM_FULL_POWER	0	Pwm full power mode.
PWM_LOW_POWER	1	Pwm low power mode.
PWM_NODEFINE_POWER	2	Pwm no define power mode.

### 3.10.2.9 Enumeration Pwm\_DataUdateType

#### Details:

This enumeration specifies updating signal immediately or using synchronization.

**Implements:** Pwm\_DataUdateType\_enumeration

**Table 3-191. Enumeration Pwm\_DataUdateType Values**

Name	Initializer	Description
PWM_UPDATE_SYNCHRONOUS	0	The signals are updated synchronous.
PWM_UPDATE_ASYNCHRONOUS	1	The signals are updated asynchronous.

### 3.10.3 Structs Reference

Data structures supported by the driver are as per AUTOSAR PWM Driver software specification Version 4.2 Rev0002 .

### 3.10.3.1 Structure Pwm\_FlexIO\_ChannelConfigType

FlexIO IP specific channel configuration structure type.

**Implements:** Pwm\_FlexIO\_ChannelConfigType\_struct

#### Declaration:

```
typedef struct
{
    Pwm_PeriodType          nPwmDefaultPeriod,
    uint16                  u16PwmDefaultDutyCycle,
    Pwm_FlexIO_ChannelType  nHwChannelId,
    Pwm_FlexIO_TimerType    nHwTimerId,
} Pwm_FlexIO_ChannelConfigType;
```

**Table 3-192. Structure Pwm\_FlexIO\_ChannelConfigType member description**

Member	Description
nPwmDefaultPeriod	PWM default period: [0-255] in ticks.
u16PwmDefaultDutyCycle	Default value for duty cycle: [0-0x8000] (0-100%).
nHwChannelId	FlexIO channel ID.
nHwTimerId	FlexIO timer ID.

### 3.10.3.2 Structure Pwm\_FlexIO\_IpConfigType

Structure that combine number of FlexIO channels in the PWM configuration.

**Implements:** Pwm\_FlexIO\_IpConfigType\_struct

#### Declaration:

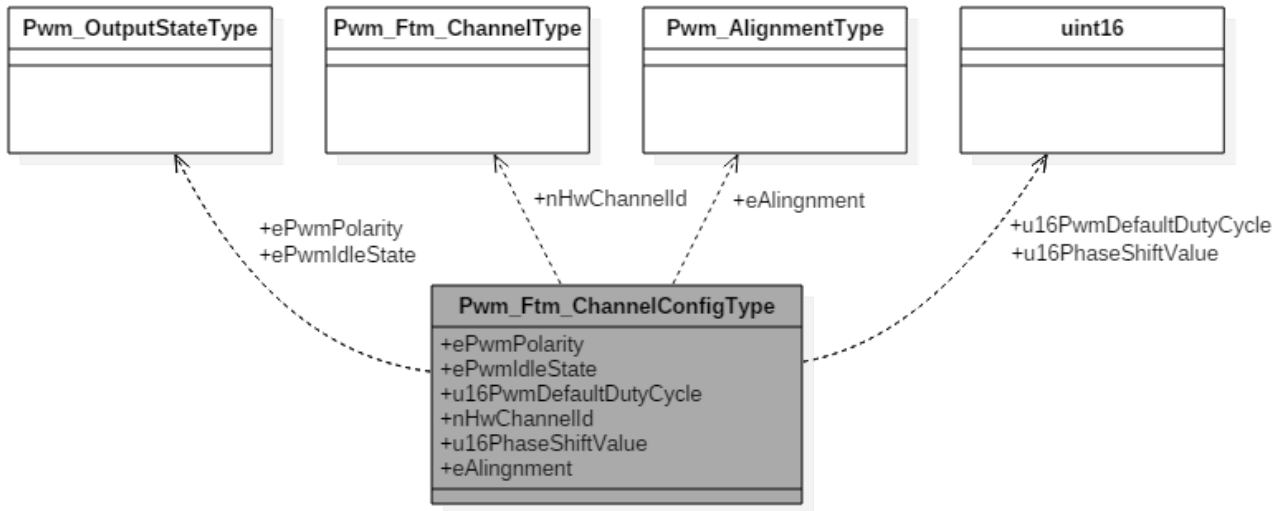
```
typedef struct
{
    uint8                  u8ChannelNumber,
    Pwm_FlexIO_ChannelConfigType (*pChannelsConfig) [],
} Pwm_FlexIO_IpConfigType;
```

**Table 3-193. Structure Pwm\_FlexIO\_IpConfigType member description**

Member	Description
u8ChannelNumber	Number of FlexIO channels in the PWM configuration.
pChannelsConfig	Pointer to the array of configured channels for FlexIO.

### 3.10.3.3 Structure Pwm\_Ftm\_ChannelConfigType

Ftm IP specific channel configuration structure type.



**Figure 3-9. Struct Pwm\_Ftm\_ChannelConfigType**

**Implements:** Pwm\_Ftm\_ChannelConfigType\_struct

#### Declaration:

```

typedef struct
{
    Pwm_OutputStateType ePwmPolarity,
    Pwm_OutputStateType ePwmIdleState,
    uint16              u16PwmDefaultDutyCycle,
    Pwm_Ftm_ChannelType nHwChannelId,
    uint16              u16PhaseShiftValue,
    Pwm_AlignmentType   eAlignment
} Pwm_Ftm_ChannelConfigType;
  
```

**Table 3-194. Structure Pwm\_Ftm\_ChannelConfigType member description**

Member	Description
ePwmPolarity	PWM signal polarity: High or low.
ePwmIdleState	PWM signal idle state: High or low.
u16PwmDefaultDutyCycle	Default value for duty cycle: [0-0x8000] (0-100%).
nHwChannelId	FTM channel ID.
u16PhaseShiftValue	Default value for phase shift: [0-0xFFFE].
eAlignment	Channel alignment type.



### 3.10.3.4 Structure Pwm\_Ftm\_ModuleConfigType

Structure that combine number of FTM channels in the PWM configuration.

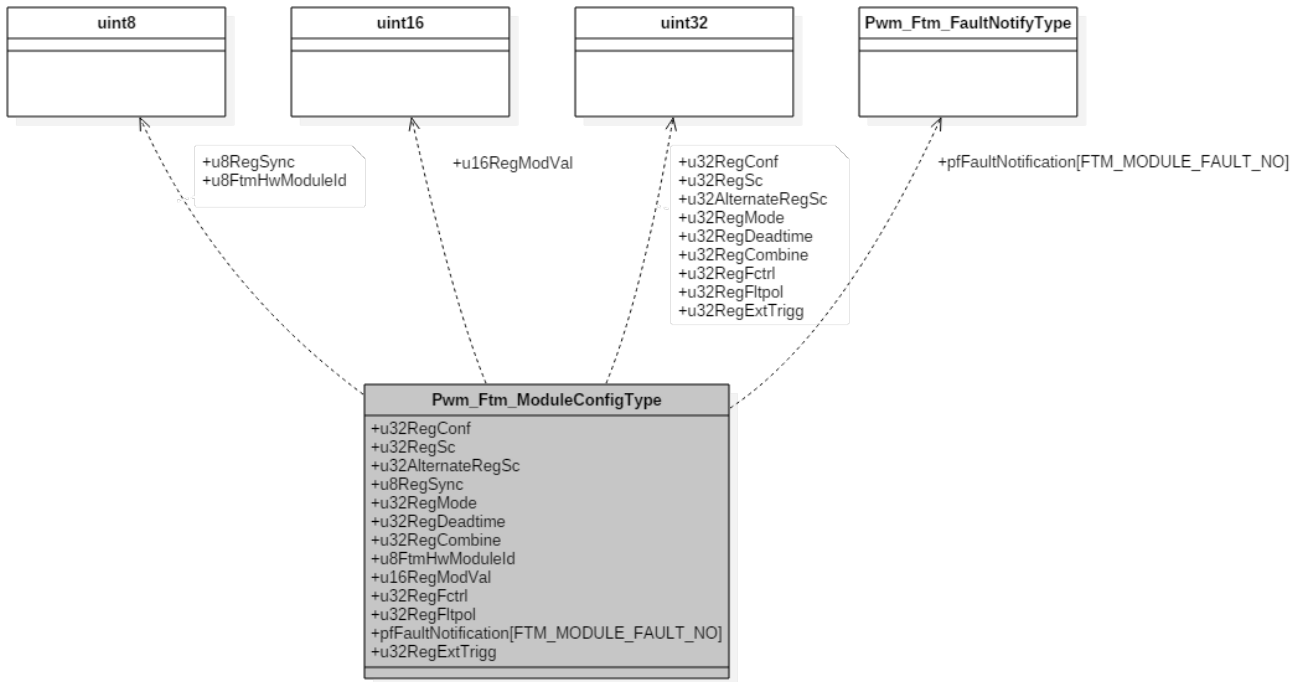


Figure 3-10. Struct Pwm\_Ftm\_ModuleConfigType

**Implements:** Pwm\_Ftm\_ModuleConfigType\_struct

**Declaration:**

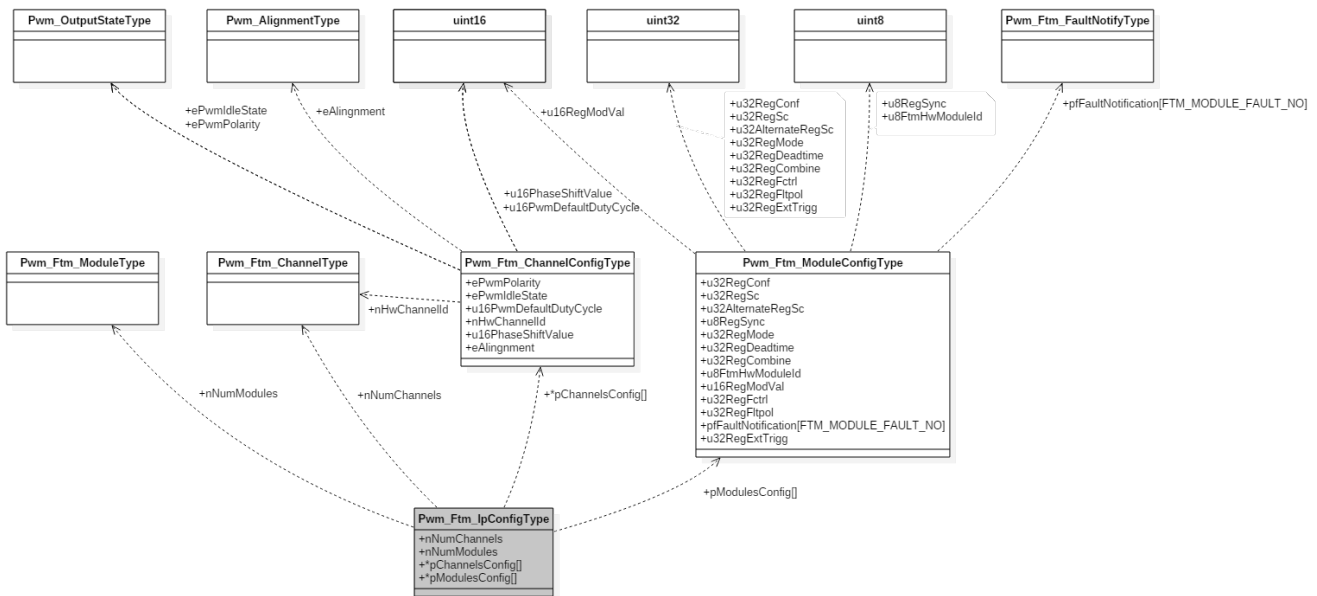
```
typedef struct
{
    uint32 u32RegConf,
    uint32 u32RegSc,
    uint32 u32AlternateRegSc,
    uint8 u8RegSync,
    uint32 u32RegMode,
    uint32 u32RegDeadtime,
    uint32 u32RegCombine,
    uint8 u8FtmHwModuleId,
    uint16 u16RegModVal,
    uint32 u32RegFctrl,
    uint32 u32RegFltpol,
    Pwm_Ftm_FaultNotifyType pfFaultNotification[FTM_MODULE_FAULT_NO],
    uint32 u32RegExtTrigg
} Pwm_Ftm_ModuleConfigType
```

**Table 3-195. Structure Pwm\_Ftm\_ModuleConfigType member description**

Member	Description
u32RegConf	Configuration register value.
u32RegSc	Status and control register value.
u32AlternateRegSc	Alternate status and control register value.
u8RegSync	Synchronize register value.
u32RegMode	Mode selection register value.
u32RegDeadtime	Deadtime register value.
u32RegCombine	Channel combine register value.
u16RegModVal	Default period (MOD) register value.
u32RegFctrl	Fault control register value.
u32RegFltpol	Fault polarity register value.
u8FtmHwModuleId	Id value of the configured FTM module.
pfFaultNotification	Pointer to fault notification function.
u32RegExtTrigg	External trigger register value.

### 3.10.3.5 Structure Pwm\_Ftm\_IpConfigType

Structure that combine number of FTM channels in the PWM configuration.

**Figure 3-11. Struct Pwm\_Ftm\_IpConfigType**

**Implements:** `Pwm_Ftm_IpConfigType_struct`

**Declaration:**

```
typedef struct
{
    Pwm_Ftm_ChannelType      nNumChannels,
    Pwm_Ftm_ModuleType       nNumModules,
    Pwm_Ftm_ChannelConfigType (*pChannelsConfig) [],
    Pwm_Ftm_ModuleConfigType  (*pModulesConfig) []
} Pwm_Ftm_IpConfigType;
```

**Table 3-196. Structure Pwm\_Ftm\_IpConfigType member description**

Member	Description
nNumChannels	Number of FTM channels in each module of the current the PWM configuration.
nNumModules	Number of FTM modules in the PWM configuration.
pChannelsConfig	Pointer to the configured channels for FTM.
pModulesConfig	Pointer to the configured modules for FTM.

**3.10.3.6 Structure Pwm\_IpConfigType**

Combined IP specific configuration structure.

**Implements:** Pwm\_IpConfigType\_struct

**Declaration:**

```
typedef struct
{
    Pwm_FlexIO_IpConfigType      *pFlexIOIpConfig,
    Pwm_Ftm_IpConfigType         *pFtmIpConfig,
    Pwm_IpChannelConfigType      (*pIpChannelsConfig) []
} Pwm_IpConfigType;
```

**Table 3-197. Structure Pwm\_IpConfigType member description**

Member	Description
pFlexIOIpConfig	Pointer to FlexIO IPV configuration.
pFtmIpConfig	Pointer to FTM IPV configuration.
pIpChannelsConfig	Pointer to Array containing IP type and index in the IP configuration table for each PWM channel.

**3.10.3.7 Structure Pwm\_IpChannelConfigType**

PWM channel high level configuration structure.

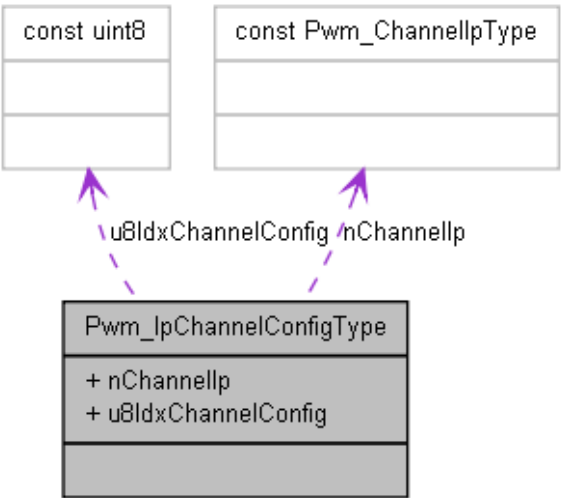


Figure 3-12. Structure Pwm\_IpChannelConfigType

Declaration:

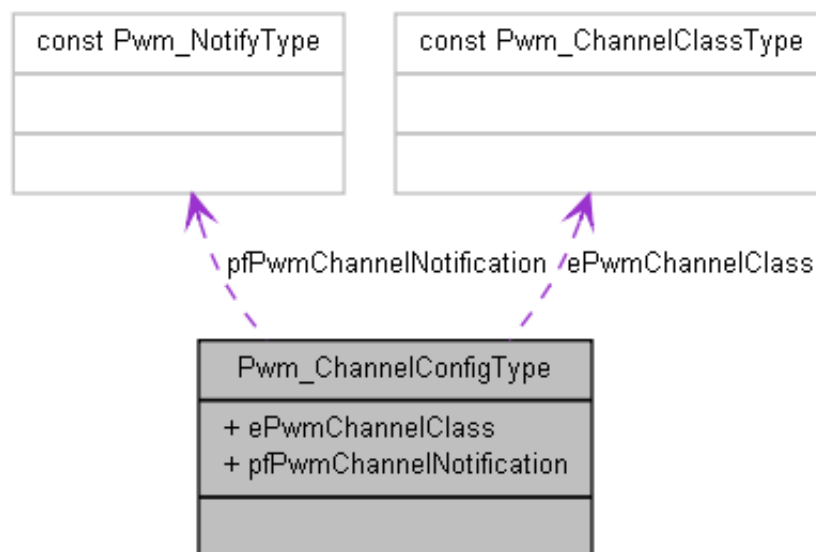
```
typedef struct
{
    Pwm_ChannelIpType    nChannelIp,
    uint8                u8IdxChannelConfig
} Pwm_IpChannelConfigType;
```

Table 3-198. Structure Pwm\_IpChannelConfigType member description

Member	Description
nChannelIp	The IP used to implement this specific PWM channel.
u8IdxChannelConfig	Index in the IP specific configuration table.

3.10.3.8 Structure Pwm\_ChannelConfigType

PWM channel high level configuration structure.



**Figure 3-13. Struct Pwm\_ChannelConfigType**

**Implements:** `Pwm_ChannelConfigType_struct`

#### Declaration:

```
typedef struct
{
    const Pwm_ChannelClassType ePwmChannelClass,
    const Pwm_NotifyType pfPwmChannelNotification
} Pwm_ChannelConfigType;
```

**Table 3-199. Structure Pwm\_ChannelConfigType member description**

Member	Description
ePwmChannelClass	Channel class type: Variable/Fixed period.
pfPwmChannelNotification	Pointer to notification function.

### 3.10.3.9 Structure Pwm\_ConfigType

PWM high level configuration structure.

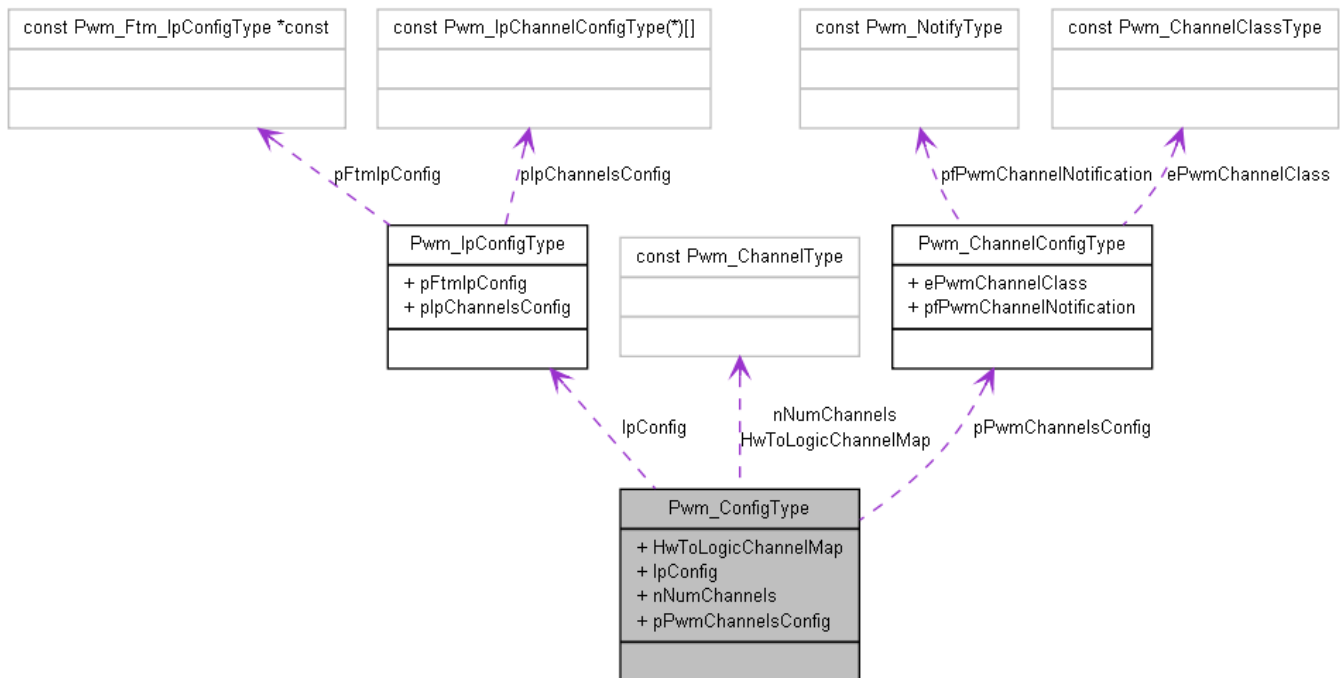


Figure 3-14. Struct Pwm\_ConfigType

**Implements:** Pwm\_ConfigType\_struct

**Declaration:**

```
typedef struct
{
    const Pwm_ChannelType HwToLogicChannelMap[PWM_HW_CHANNELS_NO_U8],
    const Pwm_IpConfigType IpConfig,
    const Pwm_ChannelType nNumChannels,
    const Pwm_ChannelConfigType(* pPwmChannelsConfig) []
} Pwm_ConfigType;
```

Table 3-200. Structure Pwm\_ConfigType member description

Member	Description
HwToLogicChannelMap	Index table to translate HW channels to logical used to process interrupts for notifications.
IpConfig	Combined IP specific configuration structure.
nNumChannels	Number of PWM configured channels.
pPwmChannelsConfig	Pointer to the list of PWM configured channels.

### 3.10.4 Types Reference

Types supported by the driver are as per AUTOSAR PWM Driver software specification Version 4.2 Rev0002 .

### 3.10.4.1 Typedef Pwm\_FlexIO\_ChannelType

FlexIO hardware channel type.

**Type:** uint8

### 3.10.4.2 Typedef Pwm\_FlexIO\_TimerType

FlexIO hardware timer type.

**Type:** uint8

### 3.10.4.3 Typedef Pwm\_NotifyType

Channel notification typedef.

**Details:**

Pointer to notification handler

**Type:** void\*

### 3.10.4.4 Typedef Pwm\_ChannelType

PWM channel type.

**Implements:** Pwm\_ChannelType\_typedef

**Type:** uint8

### 3.10.4.5 Typedef Pwm\_PeriodType

Channel period typedef.

**Implements:** Pwm\_PeriodType\_typedef

**Type:** uint16

### 3.10.4.6 Typedef Pwm\_ChannelIpType

IP type used to implement a PWM channel.

**Implements:** Pwm\_ChannelIpType\_typedef

**Type:** uint8

## 3.10.5 Function Reference

Functions of all functions supported by the driver are as per AUTOSAR PWM Driver software specification Version 4.2 Rev0002 .

### 3.10.5.1 Function Pwm\_Init

This function initializes the PWM driver.

**Details:**

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 `PwmDevErrorDetect` 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.

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`.

**Return:** void.

**Implements:** `Pwm_Init_Activity`

**Prototype:** `void Pwm_Init(const Pwm_ConfigType *ConfigPtr);`

**Table 3-201. Pwm\_Init Arguments**

Type	Name	Direction	Description
<code>constPwm_ConfigType*</code>	<code>ConfigPtr</code>	input	Pointer to PWM top configuration structure.

### 3.10.5.2 Function Pwm\_DeInit

This function deinitializes the PWM driver.

#### Details:

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.

**Return:** void.

**Implements:** Pwm\_DeInit\_Activity

**Prototype:** void Pwm\_DeInit(void);

### 3.10.5.3 Function Pwm\_SetPeriodAndDuty

This function sets the period and the duty cycle for the specified PWM channel.

#### Details:

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 `PwmDevErrorDetect` 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`.

**Return:** void.

**Implements:** `Pwm_SetPeriodAndDuty_Activity`

**Prototype:** `void Pwm_SetPeriodAndDuty(Pwm_ChannelType ChannelNumber, Pwm_PeriodType Period, uint16 u16DutyCycle);`

**Table 3-202. Pwm\_SetPeriodAndDuty Arguments**

Type	Name	Direction	Description
<code>Pwm_ChannelType</code>	<code>ChannelNumber</code>	input	- PWM channel id.
<code>Pwm_PeriodType</code>	<code>Period</code>	input	- PWM signal period value.

*Table continues on the next page...*

**Table 3-202. Pwm\_SetPeriodAndDuty Arguments (continued)**

Type	Name	Direction	Description
uint16	u16DutyCycle	input	- PWM duty cycle value 0x0000 for 0% ... 0x8000 for 100%.

### 3.10.5.4 Function Pwm\_SetDutyCycle

This function sets the duty cycle for the specified PWM channel.

#### Details:

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 PwmDevErrorDetect 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).

**Return:** void.

**Implements:** Pwm\_SetDutyCycle\_Activity

**Prototype:** void Pwm\_SetDutyCycle(Pwm\_ChannelType ChannelNumber, uint16 u16DutyCycle);

**Table 3-203. Pwm\_SetDutyCycle Arguments**

Type	Name	Direction	Description
Pwm_ChannelType	ChannelNumber	input	PWM channel id.
uint16	u16DutyCycle	input	PWM duty cycle value 0x0000 for 0% ... 0x8000 for 100%.

### 3.10.5.5 Function Pwm\_SetOutputToIdle

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

**Details:**

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 PwmDevErrorDetect 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 `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`.

**Return:** void.

**Implements:** `Pwm_SetOutputToIdle_Activity`

**Prototype:** `void Pwm_SetOutputToIdle(Pwm_ChannelType ChannelNumber);`

**Table 3-204. Pwm\_SetOutputToIdle Arguments**

Type	Name	Direction	Description
<code>Pwm_ChannelType</code>	<code>ChannelNumber</code>	input	- pwm channel id.

### 3.10.5.6 Function `Pwm_DisableNotification`

This function disables the user notifications.

**Details:**

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 `PwmDevErrorDetect` 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.

**Return:** void.

**Implements:** Pwm\_DisableNotification\_Activity

**Prototype:** void Pwm\_DisableNotification(Pwm\_ChannelType ChannelNumber);

**Table 3-205. Pwm\_DisableNotification Arguments**

Type	Name	Direction	Description
Pwm_ChannelType	ChannelNumber	input	PWM channel id.

### 3.10.5.7 Function Pwm\_EnableNotification

This function enables the user notifications.

#### NOTE

Notification is not supported for Center Aligned channels. If reference channel is configured with PWM\_CENTER\_ALIGNED as PwmChanEdgeAlignment then notification should not use with this channel and PwmNotification callback function should be NULL as well. Pwm\_EnableNotification should not be called with this channel.

#### **Details:**

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\_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 PwmDevErrorDetect 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.

**Return:** void.

**Implements:** Pwm\_EnableNotification\_Activity

**Prototype:** void Pwm\_EnableNotification(Pwm\_ChannelType ChannelNumber,  
Pwm\_EdgeNotificationType Notification);

**Table 3-206. Pwm\_EnableNotification Arguments**

Type	Name	Direction	Description
Pwm_ChannelType	ChannelNumber	input	PWM channel id.
Pwm_EdgeNotificationType	Notification	input	Notification type to be enabled.

### 3.10.5.8 Function Pwm\_GetChannelState

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

**Details:**

The function Pwm\_GetChannelState shall return the duty cycle of the channel. In case the channel is idle, the returned value will be zero.

**Return:** uint16 - duty cycle of the requested channel.

**Implements:** Pwm\_GetChannelState\_Activity

**Prototype:** uint16 Pwm\_GetChannelState(Pwm\_ChannelType ChannelNumber);



**Table 3-207. Pwm\_GetChannelState Arguments**

Type	Name	Direction	Description
Pwm_ChannelType	ChannelNumber	input	PWM channel id.

### 3.10.5.9 Function Pwm\_GetOutputState

This function returns the signal output state.

#### **Details:**

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 PwmDevErrorDetect 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.

**Return:** Pwm\_OutputStateType PWM signal output logic value.

**Implements:** Pwm\_GetOutputState\_Activity

**Prototype:** Pwm\_OutputStateType Pwm\_GetOutputState(Pwm\_ChannelType ChannelNumber);

**Table 3-208. Pwm\_GetOutputState Arguments**

Type	Name	Direction	Description
Pwm_ChannelType	ChannelNumber	input	PWM channel id.

**Table 3-209. Pwm\_GetOutputState Return Values**

Name	Description
PWM_LOW	- The output state of PWM channel is low.
PWM_HIGH	- The output state of PWM channel is high.

### 3.10.5.10 Function Pwm\_ForceOutputToZero

This function forces of the output of a given FTM channel to logic 0.

#### Details:

This API sets output state of a FTM channel depending on the value of bForce parameter and the configured polarity of the given channel.

**Return:** void.

**Implements:** Pwm\_ForceOutputToZero\_Activity

**Prototype:** void Pwm\_ForceOutputToZero(Pwm\_ChannelType ChannelNumber, boolean bForce);

**Table 3-210. Pwm\_ForceOutputToZero Arguments**

Type	Name	Direction	Description
Pwm_ChannelType	ChannelNumber	input	PWM channel id.
boolean	bForce	input	Boolean value of the expected new state of the output pin (channel).

### 3.10.5.11 Function Pwm\_SetClockMode

Implementation specific function to change the peripheral clock frequency..

#### Details:

This function is called to select one of the two possible prescalers:  
PWM\_PRIMARY\_PRESCALER or PWM\_ALTERNATIVE\_PRESCALER

**Return:** None.

**Prototype:** `void Pwm_SetClockMode(Pwm_PrescalerType ePrescaler);`

**Table 3-211. Pwm\_SetClockMode Arguments**

Type	Name	Direction	Description
Pwm_PrescalerType	ePrescaler	Input	One of the two possible prescalers: PWM_PRIMARY_PRESCALER or PWM_ALTERNATIVE_PRESCALER

### 3.10.5.12 Function Pwm\_GetVersionInfo

This function returns PWM driver version details.

**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.

**Return:** void.

**Implements:** Pwm\_GetVersionInfo\_Activity

**Prototype:** `void Pwm_GetVersionInfo(Std_VersionInfoType *pVersioninfo);`

**Table 3-212. Pwm\_GetVersionInfo Arguments**

Type	Name	Direction	Description
Std_VersionInfoType *	pVersioninfo	input, output	Pointer to Std_VersionInfoType output variable.

### 3.10.5.13 Function Pwm\_DisableTrigger

This function disable trigger generation for specific source

**Details:**

Corresponding bits with trigger source as bellow:

- Bit 0 Channel 2 Trigger Enable
- Bit 1 Channel 3 Trigger Enable
- Bit 2 Channel 4 Trigger Enable
- Bit 3 Channel 5 Trigger Enable
- Bit 4 Channel 0 Trigger Enable
- Bit 5 Channel 1 Trigger Enable
- Bit 6 Initialization Trigger Enable
- Bit 8 Channel 6 Trigger Enable
- Bit 9 Channel 7 Trigger Enable

If development error detection for the PWM module is enabled, the PWM functions shall check `u32TriggerMask` to make sure it has to be compatible with hardware register. Development error `PWM_E_TRIGGER_MASK` will be raised if this condition is not passed

If development error detection for the PWM module is enabled, the PWM functions shall check the parameter `u8TriggerHostId` and raise development error `PWM_E_PARAM_INSTANCE` if the parameter `u8TriggerHostId` 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 `PwmDevErrorDetect` 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`).

**Return:** void.

**Implements:** `Pwm_DisableTrigger_Activity`

**Prototype:** `void Pwm_DisableTrigger(uint8 u8TriggerHostId, uint16 u16TriggerMask);`

**Table 3-213. Pwm\_DisableTrigger Arguments**

Type	Name	Direction	Description
uint8	<code>u8TriggerHostId</code>	input	Hardware module
uint16	<code>u16TriggerMask</code>	input	bit mask will be set to enable trigger with corresponding sources.

### 3.10.5.14 Function Pwm\_EnableTrigger

This function enable trigger generation for specific source

#### **Details:**

Corresponding bits with trigger source as bellow:

- Bit 0 Channel 2 Trigger Enable
- Bit 1 Channel 3 Trigger Enable
- Bit 2 Channel 4 Trigger Enable
- Bit 3 Channel 5 Trigger Enable
- Bit 4 Channel 0 Trigger Enable
- Bit 5 Channel 1 Trigger Enable
- Bit 6 Initialization Trigger Enable
- Bit 8 Channel 6 Trigger Enable
- Bit 9 Channel 7 Trigger Enable

If development error detection for the PWM module is enabled, the PWM functions shall check u32TriggerMask to make sure it has to be compatible with hardware register. Development error PWM\_E\_TRIGGER\_MASK will be raised if this condition is not passed

If development error detection for the PWM module is enabled, the PWM functions shall check the parameter u8TriggerHostId and raise development error PWM\_E\_PARAM\_INSTANCE if the parameter u8TriggerHostId 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 PwmDevErrorDetect 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).

**Return:** void.

**Implements:** Pwm\_EnableTrigger\_Activity

**Prototype:** `void Pwm_EnableTrigger(uint8 u8TriggerHostId, uint16 u16TriggerMask);`

**Table 3-214. Pwm\_EnableTrigger Arguments**

Type	Name	Direction	Description
uint8	u8TriggerHostId	input	Hardware module
uint16	u16TriggerMask	input	bit mask will be set to enable trigger with corresponding sources.

### 3.10.5.15 Function Pwm\_MaskOutputs

This function force channels output to their inactive state

**Details:**

Corresponding bits with channel will be masked:

- Bit 0 Channel 0 Output Mask
- Bit 1 Channel 1 Output Mask
- Bit 2 Channel 2 Output Mask
- Bit 3 Channel 3 Output Mask
- Bit 4 Channel 4 Output Mask
- Bit 5 Channel 5 Output Mask
- Bit 6 Channel 6 Output Mask
- Bit 7 Channel 7 Output Mask

.

If development error detection for the PWM module is enabled, the PWM functions shall check the parameter u8ModuleId and raise development error PWM\_E\_PARAM\_INSTANCE if the parameter u8ModuleId 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 PwmDevErrorDetect 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).

**Return:** void.

**Implements:** Pwm\_MaskOutputs\_Activity

**Prototype:** void Pwm\_MaskOutputs(uint8 u8ModuleId, uint8 u8ChannelMask);

**Table 3-215. Pwm\_MaskOutputs Arguments**

Type	Name	Direction	Description
uint8	u8ModuleId	input	Hardware module
uint8	u8ChannelMask	input	Bit mask will be set to enable trigger with corresponding channel.

### 3.10.5.16 Function Pwm\_UnMaskOutputs

This function puts channels output to normal operation state

**Details:**

Corresponding bits with channel will be masked:

- Bit 0 Channel 0 Output Mask
- Bit 1 Channel 1 Output Mask
- Bit 2 Channel 2 Output Mask
- Bit 3 Channel 3 Output Mask
- Bit 4 Channel 4 Output Mask
- Bit 5 Channel 5 Output Mask
- Bit 6 Channel 6 Output Mask
- Bit 7 Channel 7 Output Mask

If development error detection for the PWM module is enabled, the PWM functions shall check the parameter u8ModuleId and raise development error PWM\_E\_PARAM\_INSTANCE if the parameter u8ModuleId 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 PwmDevErrorDetect 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).

**Return:** void.

**Implements:** Pwm\_UnMaskOutputs\_Activity

**Prototype:** void Pwm\_UnMaskOutputs(uint8 u8ModuleId, uint8 u8ChannelMask);

**Table 3-216. Pwm\_UnMaskOutputs Arguments**

Type	Name	Direction	Description
uint8	u8ModuleId	input	Hardware module
uint8	u8ChannelMask	input	Bit mask will be set to enable trigger with corresponding channel.

### 3.10.5.17 Function Pwm\_ResetCounter

This function shall reset the HW counter of the PWM timer

If development error detection for the Pwm module is enabled, the PWM functions shall check the parameter u8ModuleId and raise development error PWM\_E\_PARAM\_INSTANCE if the parameter u8ModuleId 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 PwmDevErrorDetect 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).

**Return:** void.

**Implements:** Pwm\_ResetCounter\_Activity

**Prototype:** void Pwm\_ResetCounter(uint8 u8ModuleId);



**Table 3-217. Pwm\_ResetCounter Arguments**

Type	Name	Direction	Description
uint8	u8ModuleId	input	Hardware module

### 3.10.5.18 Function Pwm\_ResetCounterEnable

This function shall enable the PWM timer HW counter reset by Pwm\_SyncUpdate() function.

**Details:**

**Return:** void.

**Implements:** Pwm\_ResetCounterEnable\_Activity

**Prototype:** void Pwm\_ResetCounterEnable(uint8 u8ModuleId);

**Table 3-218. Pwm\_ResetCounterEnable Arguments**

Type	Name	Direction	Description
uint8	u8ModuleId	Input	Hardware module.

### 3.10.5.19 Function Pwm\_ResetCounterDisable

This function shall disable the PWM timer HW counter reset by Pwm\_SyncUpdate() function.

**Details:**

**Return:** void.

**Implements:** Pwm\_ResetCounterDisable\_Activity

**Prototype:** void Pwm\_ResetCounterDisable(uint8 u8ModuleId);

**Table 3-219. Pwm\_ResetCounterDisable Arguments**

Type	Name	Direction	Description
uint8	u8ModuleId	Input	Hardware module.

### 3.10.5.20 Function Pwm\_SetDutyCycle\_NoUpdate

This function sets the values of duty cycle for the specified PWM channel but without updating the PWM output.

#### Details:

The function Pwm\_SetDutyCycle\_NoUpdate shall set the duty cycle of the PWM channel to the corresponding 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 synchronous 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 hardware channel operation mode, if given channel is Modified Combine channel edge setup, (PHASE\_SHIFTED\_SYNCED and PHASE\_SHIFTED\_COMPLEMENTARY), development error PWM\_E\_PARAM\_SYNCHRONOUS\_MODIFIED\_COMBINE will be raised.

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 PwmDevErrorDetect 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).

**Return:** void.

**Implements:** Pwm\_SetDutyCycle\_NoUpdate\_Activity

**Prototype:** void Pwm\_SetDutyCycle\_NoUpdate(Pwm\_ChannelType ChannelNumber, uint16 u16DutyCycle);

**Table 3-220. Pwm\_SetDutyCycle\_NoUpdate Arguments**

Type	Name	Direction	Description
Pwm_ChannelType	ChannelNumber	input	PWM channel id.
uint16	u16DutyCycle	input	PWM duty cycle value 0x0000 for 0% ... 0x8000 for 100%.

### 3.10.5.21 Function Pwm\_SetPeriodAndDuty\_NoUpdate

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

**Details:**

The function Pwm\_SetPeriodAndDuty\_NoUpdate shall set the period and duty cycle of the PWM channel to the corresponding 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 synchronous 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 hardware channel operation mode, if given channel is Modified Combine channel edge setup, (PHASE\_SHIFTED\_SYNCED and PHASE\_SHIFTED\_COMPLEMENTARY), development error PWM\_E\_PARAM\_SYNCHRONOUS\_MODIFIED\_COMBINE will be raised.

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:  $\text{AbsoluteDutyCycle} = ((\text{uint32})\text{AbsolutePeriodTime} * \text{RelativeDutyCycle}) \gg 15;$

If the PwmDevErrorDetect 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).

**Return:** void.

**Implements:** Pwm\_SetPeriodAndDuty\_NoUpdate\_Activity

**Prototype:** void Pwm\_SetPeriodAndDuty\_NoUpdate(Pwm\_ChannelType ChannelNumber, PeriodType Period, uint16 u16DutyCycle);

**Table 3-221. Pwm\_SetPeriodAndDuty\_NoUpdate Arguments**

Type	Name	Direction	Description
Pwm_ChannelType	ChannelNumber	input	PWM channel id.
Pwm_PeriodType	Period	input	PWM signal period value
uint16	u16DutyCycle	input	PWM duty cycle value 0x0000 for 0% ... 0x8000 for 100%.

### 3.10.5.22 Function Pwm\_SetPhaseShift

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

**Details:**

In order to have Phase-Shifted Full-Bridge controller, Pwm\_SetPhaseShift is introduced. This function bases on FTM Combine mode with Cn and C(n+1) combine to generate leading edge and trailing edge. Pwm\_SetPhaseShift allows to set both phase shift value and period, the duty value is fixed to 50%.

If development error detection for the PWM module is enabled, the PWM functions shall check phase shift parameter to make sure it is not greater than 50% duty (0x4000). Development error PWM\_E\_PARAM\_PHASESHIFT\_RANGE will be raised if above condition is not passed.

If development error detection for the PWM module is enabled, development error PWM\_E\_CHANNEL\_PHASE\_SHIFT\_WITHOUT\_COMBINE will be raised when given channel other than combine channel edge setup (COMBINED\_SYNCED and COMBINED\_COMPLEMENTARY)

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 PwmDevErrorDetect 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).

**Return:** void.

**Implements:** Pwm\_SetPhaseShift\_Activity

**Prototype:** void Pwm\_SetPhaseShift(Pwm\_ChannelType u8ChannelNumber, Pwm\_PeriodType nPeriod, uint16 u16PhaseShift);

**Table 3-222. Pwm\_SetPhaseShift Arguments**

Type	Name	Direction	Description
Pwm_ChannelType	u8ChannelNumber	input	PWM channel id.
Pwm_PeriodType	nPeriod	input	PWM signal period value
uint16	u16PhaseShift	input	Phase shift value

### 3.10.5.23 Function Pwm\_SetPhaseShift\_NoUpdate

This function set phase shift value and also force duty cycle to 50%. The output will take effect after Pwm\_SyncUpdate be called.

#### Details:

In order to have Phase-Shifted Full-Bridge controller, Pwm\_SetPhaseShift\_NoUpdate is introduced. This function bases on FTM Combine mode with Cn and C(n+1) combine to generate leading edge and trailing edge. Pwm\_SetPhaseShift\_NoUpdate allows to set both phase shift value and period, the duty value is fixed to 50%.

If development error detection for the PWM module is enabled, the PWM functions shall check phase shift parameter to make sure it is not greater than 50% duty (0x4000). Development error PWM\_E\_PARAM\_PHASESHIFT\_RANGE will be raised if above condition is not passed.

If development error detection for the PWM module is enabled, development error PWM\_E\_CHANNEL\_PHASE\_SHIFT\_WITHOUT\_COMBINE will be raised when given channel other than combine channel edge setup (COMBINED\_SYNCED and COMBINED\_COMPLEMENTARY)

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 PwmDevErrorDetect 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).

**Return:** void.

**Implements:** Pwm\_SetPhaseShift\_NoUpdate\_Activity

**Prototype:** void Pwm\_SetPhaseShift\_NoUpdate(Pwm\_ChannelType u8ChannelNumber, Pwm\_PeriodType nPeriod, uint16 u16PhaseShift);

**Table 3-223. Pwm\_SetPhaseShift\_NoUpdate Arguments**

Type	Name	Direction	Description
Pwm_ChannelType	u8ChannelNumber	input	PWM channel id.
Pwm_PeriodType	nPeriod	input	PWM signal period value
uint16	u16PhaseShift	input	Phase shift value

### 3.10.5.24 Function Pwm\_SyncUpdate

#### Details:

This function is called to update pending data from buffer.

**Return:** None.

**Prototype:** void Pwm\_SyncUpdate(uint8 ModuleId, uint16 u16SubmoduleMask);

**Table 3-224. Pwm\_SyncUpdate Arguments**

Type	Name	Direction	Description
uint8	ModuleId	Input	FTM hardware module ID
uint16	u16SubmoduleMask	Input	PWM submodule mask value

### 3.10.5.25 Function Pwm\_SetPowerState

Function to enters the already prepared power state.

#### Details:

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

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 PwmDevErrorDetect 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.

**Return:** Std\_ReturnType.

**Implements:** Pwm\_SetPowerState\_Activity

**Prototype:** Std\_ReturnType Pwm\_SetPowerState  
(Pwm\_PowerStateRequestResultType\* pResult )

**Table 3-225. Pwm\_SetPowerState Arguments**

Type	Name	Direction	Description
Pwm_PowerStateRequestResultType*	pResult	output	- Pointer to a variable to store the result of this function

### 3.10.5.26 Function Pwm\_GetCurrentPowerState

Get the current power state of the Pwm HW unit.

#### Details:

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

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 PwmDevErrorDetect 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.

**Return:** Std\_ReturnType.



**Implements:** Pwm\_GetCurrentPowerState\_Activity

**Prototype:** Std\_ReturnType Pwm\_GetCurrentPowerState (Pwm\_PowerStateType\* pCurrentPowerState, Pwm\_PowerStateRequestResultType\* pResult )

**Table 3-226. Pwm\_SetPowerState Arguments**

Type	Name	Direction	Description
Pwm_PowerStateType*	pCurrentPowerState,	output	- The current power mode of the Pwm HW Unit is returned in this parameter
Pwm_PowerStateRequestResultType*	pResult	output	- Pointer to a variable to store the result of this function

### 3.10.5.27 Function Pwm\_GetTargetPowerState

Get the target power state of the Pwm HW unit.

#### Details:

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

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 PwmDevErrorDetect 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.

**Return:** Std\_ReturnType

**Implements:** Pwm\_GetTargetPowerState\_Activity

**Prototype:** Std\_ReturnType Pwm\_GetTargetPowerState (Pwm\_PowerStateType\* pTargetPowerState, Pwm\_PowerStateRequestResultType\* pResult )

**Table 3-227. Pwm\_SetPowerState Arguments**

Type	Name	Direction	Description
Pwm_PowerStateType*	pTargetPowerState	output	- The Target power mode of the Pwm HW Unit is returned in this parameter.
Pwm_PowerStateRequestResultType*	pResult	output	- Pointer to a variable to store the result of this function

### 3.10.5.28 Function Pwm\_PreparePowerState

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

#### **Details:**

This API starts the needed process to allow the Pwm HW module to enter the requested power state..

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 PwmDevErrorDetect 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.

**Return:** Std\_ReturnType.

**Implements:** Pwm\_PreparePowerState\_Activity

**Prototype:** Std\_ReturnType Pwm\_PreparePowerState ( Pwm\_PowerStateType nPowerState, Pwm\_PowerStateRequestResultType\* pResult )

**Table 3-228. Pwm\_SetPowerState Arguments**

Type	Name	Direction	Description
Pwm_PowerStateType	nPowerState	input	- The target power state intended to be attained.
Pwm_PowerStateRequestResultType*	pResult	output	- Pointer to a variable to store the result of this function

### 3.10.6 Variables Reference

Variables supported by the driver are as per AUTOSAR PWM Driver software specification Version 4.2 Rev0002 .

## 3.11 Symbolic Names Disclaimer

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

```
#define <Container_Short_Name> <Container_ID>
```

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



## Chapter 4

# Tresos Configuration Plug-in

This chapter describes the Tresos configuration plug-in for the PWM Driver. The most of the parameters are described below.

### 4.1 Configuration elements of PWM

Included forms:

- IMPLEMENTATION\_CONFIG\_VARIANT
- PwmConfigurationOfOptApiServices
- PwmGeneral
- CommonPublishedInformation
- PwmChannelConfigSet

### 4.2 Form IMPLEMENTATION\_CONFIG\_VARIANT

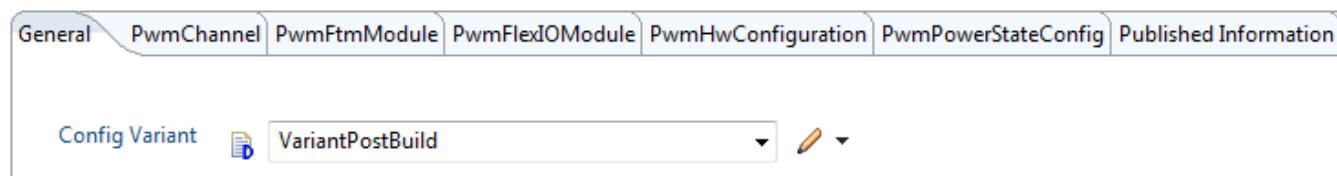
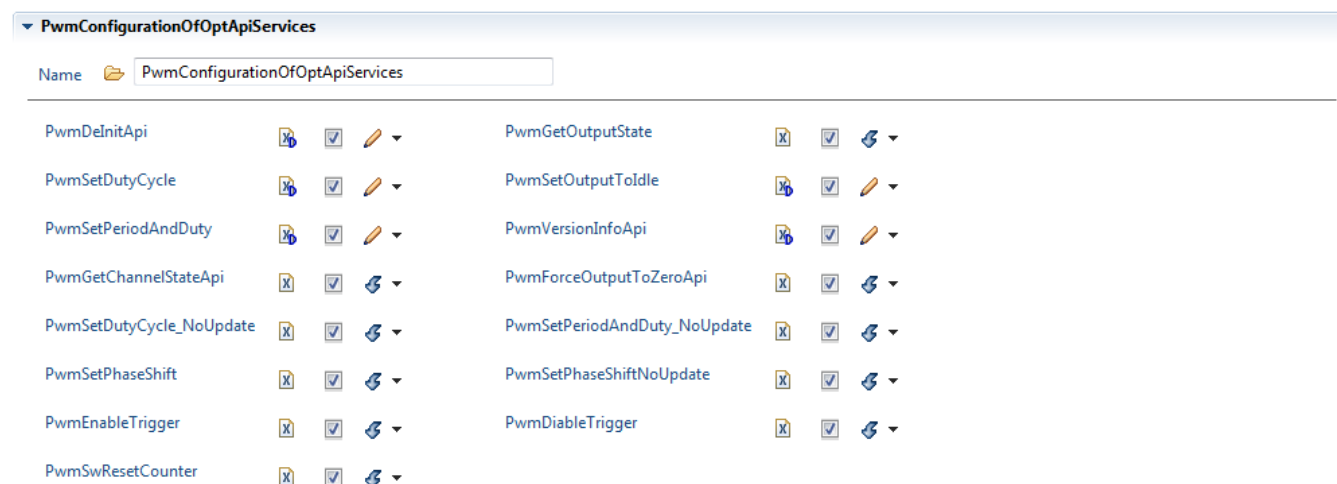


Figure 4-1. Tresos Plugin snapshot for IMPLEMENTATION\_CONFIG\_VARIANT form.

Table 4-1. Attribute IMPLEMENTATION\_CONFIG\_VARIANT detailed description

Property	Value
Label	Config Variant
Default	VariantPreCompile
Range	VariantPreCompile VariantPostBuild

## 4.3 Form PwmConfigurationOfOptApiServices



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

### 4.3.1 PwmDeInitApi (PwmConfigurationOfOptApiServices)

Add/remove the service Pwm\_DeInit() from the code.

**Table 4-2. Attribute PwmDeInitApi (PwmConfigurationOfOptApiServices) detailed description**

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
SCOPE	LOCAL
Symbolic Name	false
Default	true

### 4.3.2 PwmGetOutputState (PwmConfigurationOfOptApiServices)

Add/remove the service Pwm\_GetOutputState() from the code.

**Table 4-3. Attribute PwmGetOutputState (PwmConfigurationOfOptApiServices) detailed description**

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
SCOPE	LOCAL
Symbolic Name	false
READONLY	false

### 4.3.3 PwmSetDutyCycle (PwmConfigurationOfOptApiServices)

Add/remove the service Pwm\_SetDutyCycle() from the code.

**Table 4-4. Attribute PwmSetDutyCycle (PwmConfigurationOfOptApiServices) detailed description**

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
SCOPE	LOCAL
Symbolic Name	false
Default	true

### 4.3.4 PwmSetOutputToIdle (PwmConfigurationOfOptApiServices)

Add/remove the service Pwm\_SetOutputToIdle() from the code.

**Table 4-5. Attribute PwmSetOutputToIdle (PwmConfigurationOfOptApiServices) detailed description**

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
SCOPE	LOCAL
Symbolic Name	false
Default	true

### 4.3.5 PwmSetPeriodAndDuty (PwmConfigurationOfOptApiServices)

Add/remove the service Pwm\_SetPeriodAndDuty () from the code.

**Table 4-6. Attribute PwmSetPeriodAndDuty (PwmConfigurationOfOptApiServices) detailed description**

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
SCOPE	LOCAL
Symbolic Name	false
Default	true

### 4.3.6 PwmVersionInfoApi (PwmConfigurationOfOptApiServices)

Switch to indicate that the Pwm\_GetVersionInfo is supported

**Table 4-7. Attribute PwmVersionInfoApi (PwmConfigurationOfOptApiServices) detailed description**

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
SCOPE	LOCAL
Symbolic Name	false
Default	true

### 4.3.7 PwmGetChannelStateApi (PwmConfigurationOfOptApiServices)

Switch to indicate that the Pwm\_GetChannelState is supported

**Table 4-8. Attribute PwmGetChannelStateApi (PwmConfigurationOfOptApiServices) detailed description**

Property	Value
Type	BOOLEAN
Origin	Custom

*Table continues on the next page...*



**Table 4-8. Attribute PwmGetChannelStateApi (PwmConfigurationOfOptApiServices) detailed description (continued)**

Property	Value
Symbolic Name	false
Default	false

### 4.3.8 PwmForceOutputToZeroApi (PwmConfigurationOfOptApiServices)

Switch to indicate that the Pwm\_ForceOutputToZero API is supported

**Table 4-9. Attribute PwmForceOutputToZeroApi (PwmConfigurationOfOptApiServices) detailed description**

Property	Value
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

### 4.3.9 PwmSetDutyCycle\_NoUpdate (PwmConfigurationOfOptApiServices)

Add/remove the service Pwm\_SetDutyCycle\_NoUpdate API is supported

**Table 4-10. Attribute PwmSetDutyCycle\_NoUpdate (PwmConfigurationOfOptApiServices) detailed description**

Property	Value
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

### 4.3.10 PwmSetPeriodAndDuty\_NoUpdate (PwmConfigurationOfOptApiServices)

Add/remove the service Pwm\_SetPeriodAndDuty\_NoUpdate() from the code.

**Table 4-11. Attribute PwmSetPeriodAndDuty\_NoUpdate  
(PwmConfigurationOfOptApiServices) detailed description**

Property	Value
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

### 4.3.11 PwmSetPhaseShift (PwmConfigurationOfOptApiServices)

Add/remove the service Pwm\_SetPhaseShift() from the code.

**Table 4-12. Attribute PwmSetPhaseShift (PwmConfigurationOfOptApiServices) detailed description**

Property	Value
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

### 4.3.12 PwmSetPhaseShiftNoUpdate (PwmConfigurationOfOptApiServices)

Add/remove the service Pwm\_SetPhaseShift\_NoUpdate() from the code.

**Table 4-13. Attribute PwmSetPhaseShiftNoUpdate (PwmConfigurationOfOptApiServices) detailed description**

Property	Value
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

### 4.3.13 PwmEnableTrigger (PwmConfigurationOfOptApiServices)

Add/remove the service Pwm\_EnableTrigger() from the code.

**Table 4-14. Attribute Pwm\_EnableTrigger (PwmConfigurationOfOptApiServices) detailed description**

Property	Value
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

### 4.3.14 PwmDiableTrigger (PwmConfigurationOfOptApiServices)

Add/remove the service Pwm\_DisableTrigger() from the code.

**Table 4-15. Attribute PwmDiableTrigger (PwmConfigurationOfOptApiServices) detailed description**

Property	Value
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

### 4.3.15 PwmSwResetCounter (PwmConfigurationOfOptApiServices)

Add/remove the service Pwm\_ResetCounterEnable and Pwm\_ResetCounterDisable from the code.

**Table 4-16. Attribute PwmSwResetCounter (PwmConfigurationOfOptApiServices) detailed description**

Property	Value
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

## 4.4 Form PwmGeneral

**PwmGeneral**

Name

PwmDevErrorDetect	<input checked="" type="checkbox"/>		PwmDutycycleUpdatedEndperiod	<input checked="" type="checkbox"/>	
PwmNotificationSupported	<input checked="" type="checkbox"/>		PwmEnableDualClockMode	<input checked="" type="checkbox"/>	
PwmPeriodUpdatedEndperiod	<input checked="" type="checkbox"/>		Pwm Enable User Mode Support	<input type="checkbox"/>	
PwmIndex (0 -> 4294967295)	<input type="text" value="0"/>				
Fault Support Enable*	<input checked="" type="checkbox"/>		PwmFtmEnableExtTrigger*	<input checked="" type="checkbox"/>	
PwmEnablePhaseShift	<input checked="" type="checkbox"/>		PwmMultiChannelSynch*	<input checked="" type="checkbox"/>	
EnableMaskingOperations*	<input checked="" type="checkbox"/>				

Figure 4-3. Tresos Plugin snapshot for PwmGeneral form.

### 4.4.1 PwmDevErrorDetect (PwmGeneral)

Switch to enable/disable the development error detection.

Table 4-17. Attribute PwmDevErrorDetect (PwmGeneral) detailed description

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
SCOPE	LOCAL
Symbolic Name	false
Default	true

### 4.4.2 PwmDutycycleUpdatedEndperiod (PwmGeneral)

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

#### LIMITATION

Curent implementation supports the update of the duty cycle parameter only at the end of the current period.

**Table 4-18. Attribute PwmDutycycleUpdatedEndperiod (PwmGeneral) detailed description**

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
SCOPE	LOCAL
Symbolic Name	false
Default	true

### 4.4.3 PwmNotificationSupported (PwmGeneral)

Switch to indicate that the notifications are supported.

**Table 4-19. Attribute PwmNotificationSupported (PwmGeneral) detailed description**

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
SCOPE	LOCAL
Symbolic Name	false
Default	true

### 4.4.4 PwmEnableDualClockMode (PwmGeneral)

Switch to enable dual clock support.

**Table 4-20. Attribute PwmEnableDualClockMode (PwmGeneral) detailed description**

Property	Value
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	true

### 4.4.5 PwmPeriodUpdatedEndperiod (PwmGeneral)

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

## LIMITATION

Current implementation supports the update of the period parameter only at the end of the current period.

**Table 4-21. Attribute PwmPeriodUpdatedEndperiod (PwmGeneral) detailed description**

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	true
READONLY	true

#### 4.4.6 PwmIndex (PwmGeneral)

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

#### LIMITATION

In the current implementation this parameter is not used.

**Table 4-22. Attribute PwmIndex (PwmGeneral) detailed description**

Property	Value
Type	INTEGER
Origin	AUTOSAR_ECUC
SCOPE	LOCAL
Symbolic Name	false
Default	0
Invalid	Range <=4294967295 >=0

#### 4.4.7 PwmFaultSupport (PwmGeneral)

This enables the fault functionality.

**Table 4-23. Attribute PwmFaultSupport (PwmGeneral) detailed description**

Property	Value
Label	Fault Support Enable
Type	BOOLEAN

*Table continues on the next page...*

**Table 4-23. Attribute PwmFaultSupport (PwmGeneral) detailed description (continued)**

Property	Value
Origin	Custom
Symbolic Name	false
Default	false

#### 4.4.8 PwmFtmEnableExtTrigger (PwmGeneral)

This enables external trigger generation.

**Table 4-24. Attribute PwmFtmEnableExtTrigger (PwmGeneral) detailed description**

Property	Value
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

#### 4.4.9 PwmEnablePhaseShift (PwmGeneral)

This enables Phase Shift feature.

**Table 4-25. Attribute PwmEnablePhaseShift (PwmGeneral) detailed description**

Property	Value
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

#### 4.4.10 PwmMultiChannelSynch (PwmGeneral)

Enable/Disable the option to set the duty cycle synchronous for several channels

**Table 4-26. Attribute PwmEnablePhaseShift (PwmGeneral) detailed description**

Property	Value
Type	BOOLEAN

*Table continues on the next page...*

**Table 4-26. Attribute PwmEnablePhaseShift (PwmGeneral) detailed description (continued)**

Property	Value
Origin	Custom
Symbolic Name	false
Default	false

#### 4.4.11 EnableMaskingOperations (PwmGeneral)

Add/remove the service Pwm\_MaskOutputs and Pwm\_UnMaskOutputs from the code.

**Table 4-27. Attribute EnableMaskingOperations (PwmGeneral) detailed description**

Property	Value
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

#### 4.4.12 PwmLowPowerStatesSupport (PwmGeneral)

If enables, 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.

**Table 4-28. Attribute PwmLowPowerStatesSupport (PwmGeneral) detailed description**

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	false

#### 4.4.13 PwmPowerStateAsynchTransitionMode (PwmGeneral)

Enable/disable support of the PWM Driver to the asynchronous power state transition.



**Table 4-29. Attribute PwmPowerStateAsynchTransitionMode (PwmGeneral) detailed description**

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	false

## 4.5 Form CommonPublishedInformation

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

Property	Value
ArReleaseMajorVersion	4
ArReleaseMinorVersion	2
ArReleaseRevisionVersion	2
ModuleId	121
SwMajorVersion	1
SwMinorVersion	0
SwPatchVersion	2
VendorApiInfix	
VendorId	43

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

### 4.5.1 ArReleaseMajorVersion (CommonPublishedInformation)

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

**Table 4-30. Attribute ArReleaseMajorVersion (CommonPublishedInformation) detailed description**

Property	Value
Label	AUTOSAR Major Version
Type	INTEGER_LABEL
Origin	Custom

*Table continues on the next page...*

**Table 4-30. Attribute ArReleaseMajorVersion (CommonPublishedInformation) detailed description (continued)**

Property	Value
Symbolic Name	false
Default	4
Invalid	Range >=4 <=4

### 4.5.2 ArReleaseMinorVersion (CommonPublishedInformation)

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

**Table 4-31. Attribute ArReleaseMinorVersion (CommonPublishedInformation) detailed description**

Property	Value
Label	AUTOSAR Minor Version
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false
Default	2
Invalid	Range >=2 <=2

### 4.5.3 ArReleaseRevisionVersion (CommonPublishedInformation)

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

**Table 4-32. Attribute ArReleaseRevisionVersion (CommonPublishedInformation) detailed description**

Property	Value
Label	AUTOSAR Release Revision Version
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false
Default	2

*Table continues on the next page...*

**Table 4-32. Attribute ArReleaseRevisionVersion (CommonPublishedInformation) detailed description (continued)**

Property	Value
Invalid	Range >=2 <=2

#### 4.5.4 ModuleId (CommonPublishedInformation)

Module ID of this module from Module List.

**Table 4-33. Attribute ModuleId (CommonPublishedInformation) detailed description**

Property	Value
Label	Module Id
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false
Default	121
Invalid	Range >=121 <=121

#### 4.5.5 SwMajorVersion (CommonPublishedInformation)

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

**Table 4-34. Attribute SwMajorVersion (CommonPublishedInformation) detailed description**

Property	Value
Label	Software Major Version
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false
Default	1
Invalid	Range >=1 <=1

### 4.5.6 SwMinorVersion (CommonPublishedInformation)

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

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

Property	Value
Label	Software Minor Version
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false
Default	0
Invalid	Range <div>&gt;=0</div> <div>&lt;=0</div>

### 4.5.7 SwPatchVersion (CommonPublishedInformation)

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

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

Property	Value
Label	Software Patch Version
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false
Default	2
Invalid	Range <div>&gt;=2</div> <div>&lt;=2</div>

### 4.5.8 VendorApiInfix (CommonPublishedInformation)

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

<ModuleName>\_>VendorId>\_<VendorApiInfix><Api name from SWS>. E.g. assuming

that the VendorId of the implementor is 123 and the implementer chose a VendorApiInfix of "v11r456" a api name Can\_Write defined in the SWS will translate to Can\_123\_v11r456Write. This parameter is mandatory for all modules with upper multiplicity > 1. It shall not be used for modules with upper multiplicity =1.

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

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

### 4.5.9 VendorId (CommonPublishedInformation)

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

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

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

## 4.6 Form PwmChannelConfigSet

This container contains the Channel configuration parameter of the PWM driver.

### Included forms:

- [Form PwmChannel](#)
- [Form PwmFlexIOModule](#)
- [Form PwmFtmModule](#)

Form PwmChannelConfigSet

PwmChannelConfigSet

Name

PwmChannelConfigSet

Figure 4-5. Tresos Plugin snapshot for PwmChannelConfigSet form.

4.6.1 Form PwmChannel

Configuration of an individual PWM channel.

Is included by form: [Form PwmChannelConfigSet](#)

General   PwmChannel   PwmFtmModule   PwmFlexIOModule   PwmHwConfiguration   PwmPowerStateConfig   Published Information												
PwmChannel*												
Index	Name	PwmCha...	PWM Har...	Default P...	Default P...	PwmPola...	PwmDut...	PwmIdle...	PwmMcu...			
0	PwmChan...	0	FlexIO		1.25E-4	PWM_HIGH	16384	PWM_LOW	/Mcu/Mcu...			
1	PwmChan...	1	FTM		1.25E-4	PWM_HIGH	16384	PWM_LOW	/Mcu/Mcu...			
2	PwmChan...	2	FTM		1.25E-4	PWM_HIGH	16384	PWM_LOW	/Mcu/Mcu...			
3	PwmChan...	3	FlexIO		1.25E-4	PWM_HIGH	16384	PWM_LOW	/Mcu/Mcu...			
4	PwmChan...	4	FTM		1.25E-4	PWM_HIGH	16384	PWM_LOW	/Mcu/Mcu...			
5	PwmChan...	5	FTM		1.25E-4	PWM_HIGH	16384	PWM_LOW	/Mcu/Mcu...			
6	PwmChan...	6	FlexIO		1.25E-4	PWM_HIGH	16384	PWM_LOW	/Mcu/Mcu...			
7	PwmChan...	7	FTM		1.25E-4	PWM_HIGH	16384	PWM_LOW	/Mcu/Mcu...			
8	PwmChan...	9	FTM		1.25E-4	PWM_HIGH	16384	PWM_LOW	/Mcu/Mcu...			
9	PwmChan...	10	FlexIO		1.25E-4	PWM_HIGH	16384	PWM_LOW	/Mcu/Mcu...			

Figure 4-6. Tresos Plugin snapshot for PwmChannel form.

Below is a detailed description of the configurable parameters of each PWM channel.

PwmChannel

Name

PwmChannel\_0

General

PwmChannelId (0 -> 4294967295)\*

0

PWM Hardware IP\*

FTM

PwmFtmChannel\*

/Pwm/Pwm/PwmChannelConfigSet/PwmFtmModule\_0/PwmFtmChannels\_0

PwmFlexIOChannel\*

Default Period In Ticks\*

☒

Default Period (0 -> 65534)\*

1.25E-4

PwmChannelClass\*

PWM\_FIXED\_PERIOD

PwmPolarity\*

PWM\_HIGH

PwmDutycycleDefault (0 -> 32768)\*

16384

PwmIdleState\*

PWM\_LOW

PwmNotification\*

PwmMcuClockReferencePoint\*

/Mcu/Mcu/McuModuleConfiguration\_0/McuClockSettingConfig\_0/McuClockReferencePoint\_0

Figure 4-7. Tresos Plugin snapshot for PwmChannel form with parameter details.

#### 4.6.1.1 PwmChannelId (PwmChannel)

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

**Table 4-39. Attribute PwmChannelId (PwmChannel) detailed description**

Property	Value
Type	INTEGER
Origin	AUTOSAR_ECUC
SCOPE	LOCAL
Symbolic Name	true

#### 4.6.1.2 PwmHwIP (PwmChannel)

Hardware IP to be used for current PWM channel.

- FlexIO - FlexIO Hardware IP will be used. Please select a FlexIO configured channel from PwmFlexIOChannel combo below.
- FTM - FTM Hardware IP will be used. Please select an eMios configured channel from PwmFTMChannel combo below.

**Table 4-40. Attribute PwmHwIP (PwmChannel) detailed description**

Property	Value
Label	PWM Hardware IP
Type	ENUMERATION
Origin	Custom
Symbolic Name	false
Default	FTM
Range	FlexIO FTM

#### 4.6.1.3 PwmFtmChannel (PwmChannel)

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

**Table 4-41. Attribute PwmFtmChannel (PwmChannel) detailed description**

Property	Value
Type	REFERENCE
Origin	Custom
POSTBUILDVARIANTVALUE	true

#### 4.6.1.4 PwmFlexIOChannel (PwmChannel)

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

**Table 4-42. Attribute PwmFlexIOChannel (PwmChannel) detailed description**

Property	Value
Type	REFERENCE
Origin	Custom
POSTBUILDVARIANTVALUE	true

#### 4.6.1.5 PwmPeriodInTicks (PwmChannel)

By default period unit is measured in seconds. Enable this check to set default period unit in ticks.

**Table 4-43. Attribute PwmPeriodInTicks (PwmChannel) detailed description**

Property	Value
Label	Default Period In Ticks
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	true

#### 4.6.1.6 PwmPeriodDefault (PwmChannel)

Default period value used at initialization. The measure unit are in ticks. Valid values [0, 0xFFFFE = 65534]



**Table 4-44. Attribute PwmPeriodDefault (PwmChannel) detailed description**

Property	Value
Label	Default Period
Type	FLOAT
Origin	AUTOSAR_ECUC
SCOPE	LOCAL
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	1.25e-4
Invalid	Range <div> <div>&lt;=65534</div> <div>&gt;=0</div> </div>

#### 4.6.1.7 PwmChannelClass (PwmChannel)

Class of PWM Channel.

**Table 4-45. Attribute PwmChannelClass (PwmChannel) detailed description**

Property	Value
Type	ENUMERATION
Origin	AUTOSAR_ECUC
Symbolic Name	false
SCOPE	LOCAL
POSTBUILDVARIANTVALUE	true
OPTIONAL	true
Default	PWM_FIXED_PERIOD
Range	<div> <div>PWM_FIXED_PERIOD</div> <div>PWM_VARIABLE_PERIOD</div> <div>PWM_FIXED_PERIOD_SHIFTED</div> </div>

#### 4.6.1.8 PwmPolarity (PwmChannel)

Define the starting polarity of each PWM channel.

##### NOTE

- If user use FlexIO module ,the node "PwmPolarity" will be disable.

**Table 4-46. Attribute PwmPolarity (PwmChannel) detailed description**

Property	Value
Type	ENUMERATION
Origin	AUTOSAR_ECUC
SCOPE	LOCAL
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	PWM_HIGH
Range	PWM_HIGH PWM_LOW

#### 4.6.1.9 PwmDutycycleDefault (PwmChannel)

Default value for duty cycle used for Initialization 0, represents 0% 0x4000, represents 50% 0x8000, represents 100%.

**Table 4-47. Attribute PwmDutycycleDefault (PwmChannel) detailed description**

Property	Value
Type	INTEGER
Origin	AUTOSAR_ECUC
SCOPE	LOCAL
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	16384
Invalid	Range ≤32768 ≥0

#### 4.6.1.10 PwmIdleState (PwmChannel)

This parameter represents the state of the Pin when the output is set to Idle.

#### NOTE

- If user use FlexIO module ,the node "PwmIdleState" will be disable.

**Table 4-48. Attribute PwmIdleState (PwmChannel) detailed description**

Property	Value
Type	ENUMERATION
Origin	AUTOSAR_ECUC
SCOPE	LOCAL
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	PWM_LOW
Range	PWM_HIGH PWM_LOW

#### 4.6.1.11 PwmNotification (PwmChannel)

User callback function 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.

#### NOTE

Notification is not supported for Center Aligned channels. If reference channel is configured with PWM\_CENTER\_ALIGNED as PwmChanEdgeAlignment then notification should not use with this channel and PwmNotification callback function should be NULL as well.

**Table 4-49. Attribute PwmNotification (PwmChannel) detailed description**

Property	Value
Type	FUNCTION-NAME
Origin	AUTOSAR_ECUC
SCOPE	LOCAL
POSTBUILDVARIANTVALUE	true
OPTIONAL	true
Symbolic Name	false
Default	NULL

#### 4.6.1.12 PwmMcuClockReferencePoint (PwmChannel)

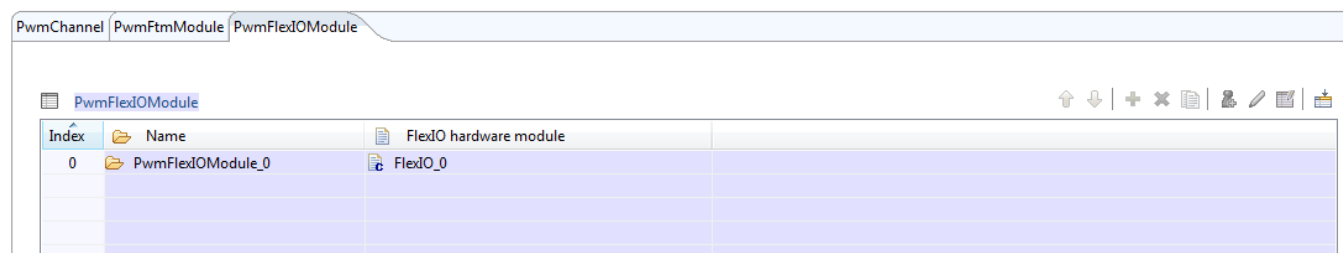
Reference to the PWM clock source configuration, which is set in the MCU driver configuration.

**Table 4-50. Attribute PwmMcuClockReferencePoint (PwmChannel) detailed description**

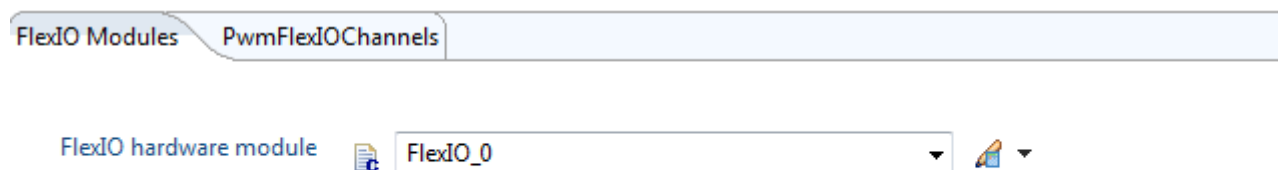
Property	Value
Type	REFERENCE
Origin	AUTOSAR_ECUC

## 4.6.2 Form PwmFlexIOModule

Configuration of a FlexIO module available on the platform.

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

Below is a detailed description of the configurable parameters of each FlexIO module.

**Figure 4-9. Tresos Plugin snapshot for PwmFlexIO form with parameter details.**

### 4.6.2.1 PwmFlexIOModule (PwmFlexIO)

Selects one of the FlexIO modules available on the platform.

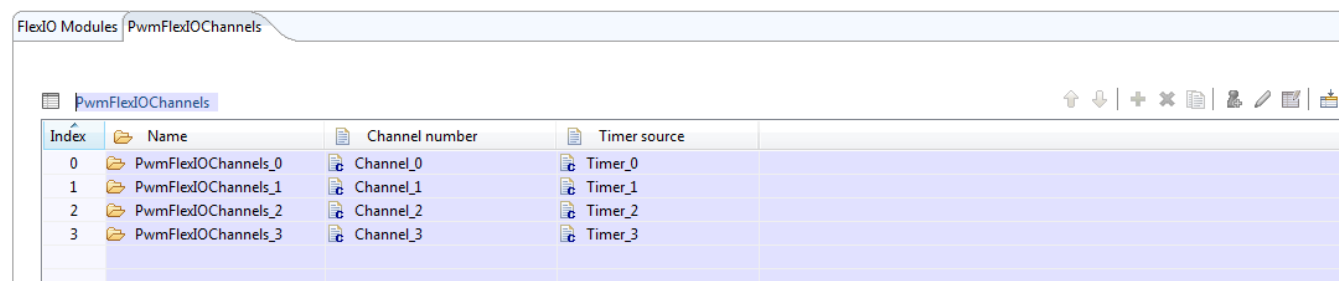
**Table 4-51. Attribute PwmFlexIOModule (PwmFlexIO) detailed description**

Property	Value
Label	FlexIO Hardware Module
Type	ENUMERATION
Origin	Custom
Symbolic Name	false

### 4.6.2.2 Form PwmFlexIOChannels

Configuration of an individual PWM FlexIO channel.

Is included by form: [Form PwmFlexIOModule](#)

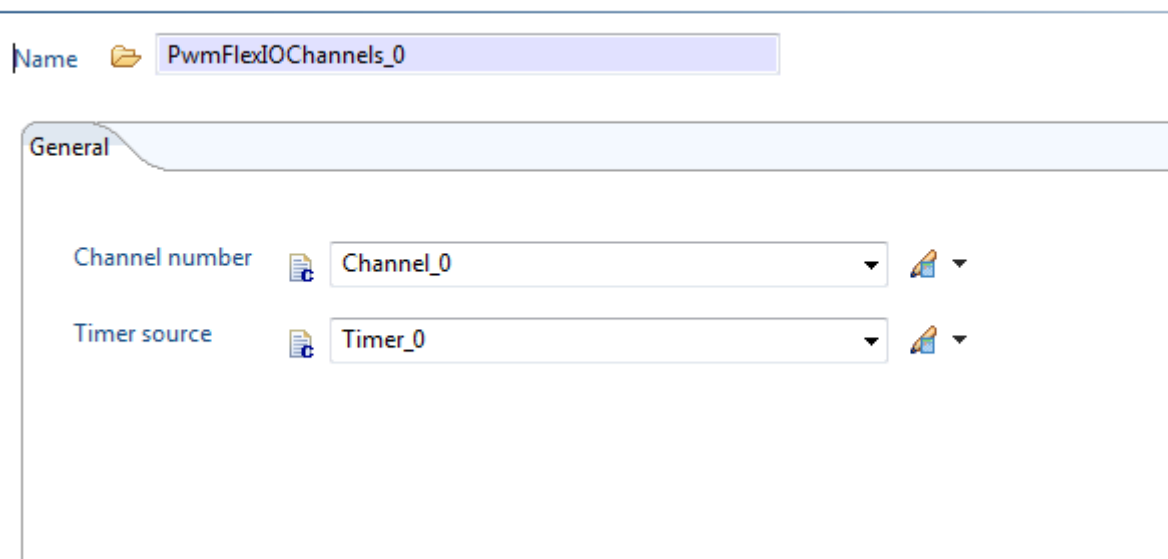



Index	Name	Channel number	Timer source
0	PwmFlexIOChannels_0	Channel_0	Timer_0
1	PwmFlexIOChannels_1	Channel_1	Timer_1
2	PwmFlexIOChannels_2	Channel_2	Timer_2
3	PwmFlexIOChannels_3	Channel_3	Timer_3

**Figure 4-10. Tresos Plugin snapshot for PwmFlexIOChannels form.**




Below is a detailed description of the configurable parameters of each FlexIO channel.




#### PwmFlexIOChannels



Name  PwmFlexIOChannels\_0

General

Channel number  Channel\_0  

Timer source  Timer\_0  

**Figure 4-11. Tresos Plugin snapshot for PwmFlexIOChannels form with parameter details.**

#### 4.6.2.2.1 PwmFlexIOChannelId (PwmFlexIOChannels)

Select one of the FlexIO channels available on the platform.

**Table 4-52. Attribute PwmFlexIOChannelId (PwmFlexIOChannels) detailed description**

Property	Value
Label	Channel number
Type	ENUMERATION
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Range	Channel_0 Channel_1 Channel_2 Channel_3 Channel_4 Channel_5 Channel_6 Channel_7

#### 4.6.2.2.2 PwmFlexIOTimer (PwmFlexIOChannels)

Select one of the FlexIO timer sources available on the platform.

**Table 4-53. Attribute PwmFlexIOTimer (PwmFlexIOChannels) detailed description**

Property	Value
Label	Timer source
Type	ENUMERATION
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Range	Timer_0 Timer_1 Timer_2 Timer_3

### 4.6.3 Form PwmFtmModule

Configuration of a FTM module available on the platform.

Is included by form : [Form PwmChannelConfigSet](#)

PwmFtmModule													
Index	Name	Ftm Hardware Module	Prescaler	PwmPres...	FTM Mo...	Ftm Mod...	End cycle...	Half cycl...	Reload Fr...	Pwm Bac...	Pwm Fau...	Pv	
0	PwmFtmModule_0	FTM_0	PRESC_1	PRESC_1	PWM_SYS...	PWM_EDG...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	LDFQ_EAC...	CNT_ON...	FLTCTRL...		
1	PwmFtmModule_1	FTM_1	PRESC_1	PRESC_1	PWM_SYS...	PWM_EDG...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	LDFQ_EAC...	CNT_ON...	FLTCTRL...		
2	PwmFtmModule_2	FTM_2	PRESC_1	PRESC_1	PWM_SYS...	PWM_EDG...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	LDFQ_EAC...	CNT_ON...	FLTCTRL...		
3	PwmFtmModule_3	FTM_3	PRESC_1	PRESC_1	PWM_SYS...	PWM_EDG...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	LDFQ_EAC...	CNT_ON...	FLTCTRL...		

**Figure 4-12. Tresos Plugin snapshot for PwmFtmModule form.**

Below is a detailed description of the configurable parameters of each FTM module.

Ftm Modules	
PwmFtmChannels	
Ftm Hardware Module	FTM_0
Prescaler	PRESC_1
PwmPrescaler_Alternate	PRESC_1
FTM Module clock selection	PWM_SYSTEM_CLOCK
Ftm Module's Channels Alignment	PWM_EDGE_ALIGNED
End cycle reload	<input checked="" type="checkbox"/> Half cycle reload <input type="checkbox"/>
Reload Frequency	LDFQ_EACH1
Deadtime Counter (0 -> 1023)	0
DeadTime Counter Prescaler	PRESC_1
Pwm Background Debug Mode configuration	CNT_ON_OUTPUTS_ON
Pwm Fault Functionality and Clear Mode	FLTCTRL_DISABLED

**Figure 4-13. Tresos Plugin snapshot for PwmFtmModule form with parameter details.**

#### 4.6.3.1 FtmModule (PwmFtmModule)

Select one of the 8 sub-modules available on the current FTM module.

**Table 4-54. Attribute FtmModule (PwmFtmModule) detailed description**

Property	Value
Label	Ftm Hardware Module
Type	ENUMERATION
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false

### 4.6.3.2 PwmPrescaler (PwmFtmModule)

PwmPrescaler. AUTOSAR Requirements: not specified.

**Table 4-55. Attribute PwmPrescaler (PwmFtmModule) detailed description**

Property	Value
Label	Prescaler
Type	ENUMERATION
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	PRESC_1
Range	PRESC_1 PRESC_2 PRESC_4 PRESC_8 PRESC_16 PRESC_32 PRESC_64 PRESC_128

### 4.6.3.3 PwmPrescaler\_Alternate (PwmFtmModule)

PwmPrescaler\_Alternate. AUTOSAR Requirements: not specified.

**Table 4-56. Attribute PwmPrescaler\_Alternate (PwmFtmModule) detailed description**

Property	Value
Label	Prescaler
Type	ENUMERATION
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	PRESC_1
Range	PRESC_1 PRESC_2 PRESC_4 PRESC_8 PRESC_16 PRESC_32 PRESC_64 PRESC_128



#### 4.6.3.4 PwmClockRef (PwmFtmModule)

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

**Table 4-57. Attribute PwmClockRef (PwmFtmModule) detailed description**

Property	Value
Type	REFERENCE
Origin	Custom

#### 4.6.3.5 PwmClockSelection (PwmFtmModule)

FTM Module clock selection.

- PWM\_NO\_CLOCK: No input clock is given to FTM. FTM count is not operational.
- PWM\_SYSTEM\_CLOCK: System clock is given to FTM.
- PWM\_EXTERNAL\_CLOCK: An fixed frequency clock is given to FTM.
- PWM\_FIXED\_FREQ\_CLOCK: An external clock is given to FTM.

#### NOTE

Clock selection and timer frequency are dependent on the value of the clock that is configured in MCU and referenced by / PwmClockRef. Make sure that the correct reference is used for the configured clock.

**Table 4-58. Attribute PwmClockSelection (PwmFtmModule) detailed description**

Property	Value
Label	FTM Module clock selection
Type	ENUMERATION
Origin	Custom
Symbolic Name	false
Default	PWM_SYSTEM_CLOCK
Range	PWM_SYSTEM_CLOCK PWM_FIXED_FREQ_CLOCK PWM_EXTERNAL_CLOCK PWM_NO_CLOCK

#### 4.6.3.6 PwmChanEdgeAlignment (PwmFtmModule)

This list will change the PWM generation mode for this FTM sub-module. This in a Non-AUTOSAR feature.

**Table 4-59. Attribute PwmChanEdgeAlignment (PwmFtmModule) detailed description**

Property	Value
Label	Ftm Module's Channels Alignment
Type	ENUMERATION
POSTBUILDVARIANTVALUE	true
Origin	Custom
Symbolic Name	false
Default	PWM_EDGE_ALIGNED
Range	PWM_EDGE_ALIGNED PWM_CENTER_ALIGNED

#### 4.6.3.7 PwmUpdatedEndPeriod (PwmFtmModule)

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

##### NOTE

This function enabled if PwmDutycycleUpdatedEndperiod is set and PWM\_CENTER\_ALIGNED is set.

**Table 4-60. Attribute PwmUpdatedEndPeriod (PwmFtmModule) detailed description**

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
SCOPE	LOCAL
Symbolic Name	false
Default	true

#### 4.6.3.8 PwmUpdatedMiddlePeriod (PwmFtmModule)

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

##### NOTE

This function enabled if PwmDutycycleUpdatedEndperiod is set and PWM\_CENTER\_ALIGNED is set.

**Table 4-61. Attribute PwmUpdatedMiddlePeriod (PwmFtmModule) detailed description**

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
SCOPE	LOCAL
Symbolic Name	false
Default	false

#### 4.6.3.9 PwmReloadFrequency (PwmFtmModule)

Define the number of enabled reload opportunities should happen until an enabled reload opportunity becomes a reload point.

#### NOTE

This function enabled if PwmDutycycleUpdatedEndperiod is set.

**Table 4-62. Attribute PwmReloadFrequency (PwmFtmModule) detailed description**

Property	Value
Label	Prescaler
Type	ENUMERATION
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	LDFQ_EACH1
Range	LDFQ_EACH1 LDFQ_EACH2 LDFQ_EACH3 LDFQ_EACH4 LDFQ_EACH5 LDFQ_EACH6 LDFQ_EACH7 LDFQ_EACH8 LDFQ_EACH9 LDFQ_EACH10 LDFQ_EACH11 LDFQ_EACH12 LDFQ_EACH13 LDFQ_EACH14 LDFQ_EACH15 LDFQ_EACH16 LDFQ_EACH17 LDFQ_EACH18 LDFQ_EACH19 LDFQ_EACH20

**Table 4-62. Attribute PwmReloadFrequency (PwmFtmModule) detailed description**

Property	Value
	LDFQ_EACH21 LDFQ_EACH22 LDFQ_EACH23 LDFQ_EACH24 LDFQ_EACH25 LDFQ_EACH26 LDFQ_EACH27 LDFQ_EACH28 LDFQ_EACH29 LDFQ_EACH30 LDFQ_EACH31 LDFQ_EACH32

#### 4.6.3.10 PwmDeadTimeCount (PwmFtmModule)

DeadTime Counter used to create a phase offset between the output of two consecutive FTM channels used complementary modes. Complementary channels behave as a single channel with two outputs which may have a deadtime between them. For channels which are configured as independent this value has no effect.

#### NOTE

When PwmDeadTimeCount is enable and difference than zero, all channels should be configured with ChannelEdgeSetup as COMBINED\_COMPLEMENTARY or PHASE\_SHIFTED\_COMPLEMENTARY. Deadtime insertion with COMBINED\_SYNCED or PHASE\_SHIFTED\_SYNCED may cause to odd channel (n+1) not work.

**Table 4-63. Attribute PwmDeadTimeCount (PwmFtmModule) detailed description**

Property	Value
Label	Deadtime Counter
Type	INTEGER
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
OPTIONAL	true
Default	0
Invalid	Range <=63 >=0

### 4.6.3.11 DeadTimePrescaler (PwmFtmModule)

DeadTime counter prescaler.

**Table 4-64. Attribute DeadTimePrescaler (PwmFtmModule) detailed description**

Property	Value
Label	DeadTime Counter Prescaler
Type	ENUMERATION
POSTBUILDVARIANTVALUE	true
OPTIONAL	true
Origin	Custom
Symbolic Name	false
Default	PRESC_1
Range	PRESC_1 PRESC_4 PRESC_16

### 4.6.3.12 PwmBDMEnable (PwmFtmModule)

Selects the Debug mode.

#### Note

On FTM, PIT, STM base platforms. If the chip is in BDM mode and A5 core configure is set, then counter is stopped due to hardware restriction.

**Table 4-65. Attribute PwmBDMEnable (PwmFtmModule) detailed description**

Property	Value
Label	Pwm Background Debug Mode configuration
Type	ENUMERATION
Origin	Custom
Symbolic Name	false
POSTBUILDVARIANTVALUE	true
Default	CNT_STOPED_FLAG_SET
Range	CNT_STOPED_FLAG_SET CNT_STOPED_OUTPUTS_SAFE CNT_STOPED_OUTPUTS_FROZEN CNT_ON_OUTPUTS_ON

### 4.6.3.13 PwmFtmFaultFunctionality (PwmFtmModule)

Selects the fault control mode.

- **FLTCTRL\_DISABLED**: Fault control disabled for all channels.
- **AUTOMATIC\_FOR\_EVEN\_CHANNELS**: Fault control enabled for even channels (0, 2, 4..) with automatic fault clear.
- **AUTOMATIC\_FOR\_ALL\_CHANNELS**: Fault control enabled for all channels with automatic fault clear.

#### NOTE

For any automatic fault clear modes, a Fault Notification function should be defined. In automatic modes, the hardware shall be configured into manual fault clearing and automatic fault clearing shall be implemented by software.

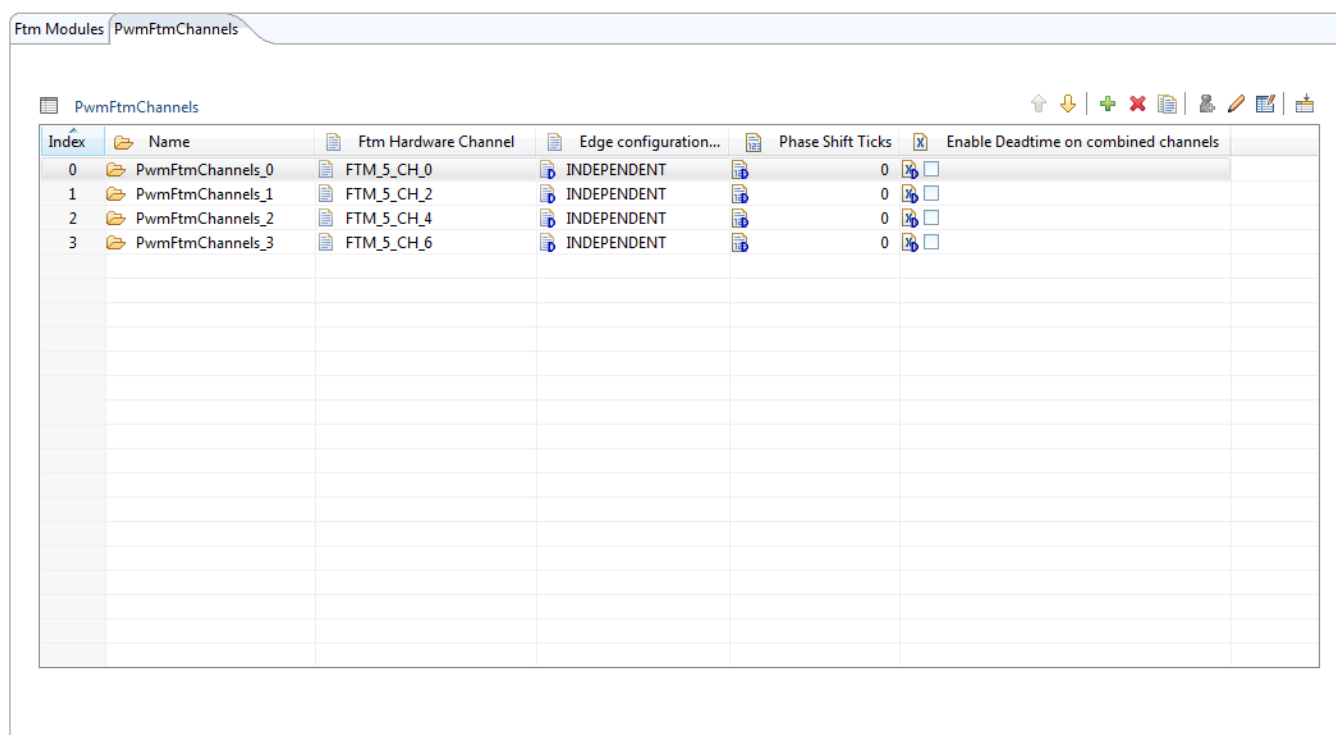
**Table 4-66. Attribute PwmFtmFaultFunctionality (PwmFtmModule) detailed description**

Property	Value
Label	Pwm Fault Funtionality and Clear Mode
Type	ENUMERATION
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	FLTCTRL_DISABLED
Range	FLTCTRL_DISABLED AUTOMATIC_FOR_EVEN_CHANNELS AUTOMATIC_FOR_ALL_CHANNELS

### 4.6.3.14 Form PwmFtmChannels

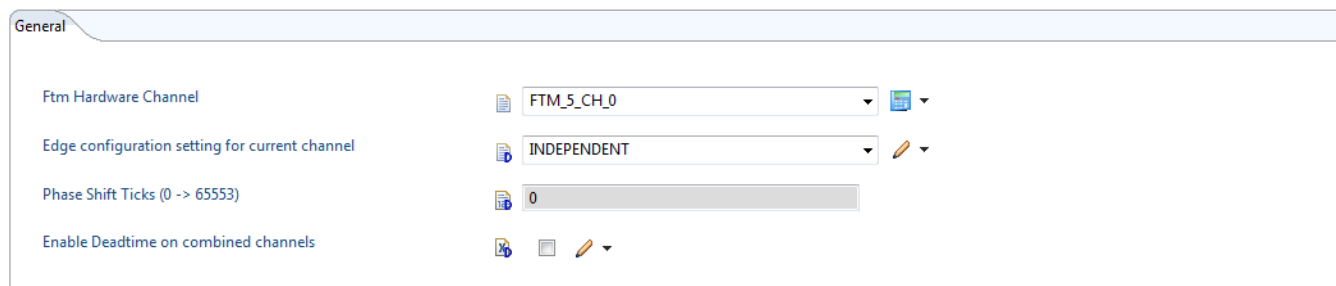
Configuration of an individual PWM FTM channel.

**Is included by form:** [Form PwmFtmModule](#)



**Figure 4-14. Treso Plugin snapshot for PwmFtmChannels form.**

Below is a detailed description of the configurable parameters of each FTM channel.



**Figure 4-15. Treso Plugin snapshot for PwmFtmChannels form with parameter details.**

#### 4.6.3.14.1 PwmFtmChannelId (PwmFtmChannels)

Channel Id of the FTM channel of the current. This value will be assigned to the symbolic name derived of the PwmChannel container short name.

**Table 4-67. Attribute PwmFtmChannelId (PwmFtmChannels) detailed description**

Property	Value
Label	Ftm Hardware Channel
Type	ENUMERATION
Origin	Custom

*Table continues on the next page...*

**Table 4-67. Attribute PwmFtmChannelId (PwmFtmChannels) detailed description (continued)**

Property	Value
POSTBUILDVARIANTVALUE	true
Symbolic Name	false

**4.6.3.14.2 ChannelEdgeSetup (PwmFtmChannels)**

- **INDEPENDENT**: Single PWM output will be generated using only one FTM HW Channel.
- **PHASE\_SHIFTED\_SINGLE**: Single phase shifted PWM output will be generated using two FTM HW Channels.
- **COMBINED\_SYNCED**: Two phase-synced PWM outputs will be generated using two FTM HW Channels.
- **COMBINED\_COMPLEMENTARY**: Two phase-complementary PWM outputs will be generated using two FTM HW Channels; The deadtime insertion is applicable to ensures that no two complementary signals (channels (n)and (n+1)) drive the active state at the same time
- **PHASE\_SHIFTED\_SYNCED**: Two phase-shifted synchronous PWM outputs will be generated using two FTM HW Channels
- **PHASE\_SHIFTED\_COMPLEMENTARY**: Two phase-shifted complementary PWM outputs will be generated using two FTM HW Channels. The deadtime insertion is applicable to ensures that no two complementary signals (channels (n)and (n+1)) drive the active state at the same time.

**NOTE**

Due to limitation of hardware, if a channel is configured with **PHASE\_SHIFTED\_SINGLE**, the port pin associate to odd channel (n+1) is configured as PWM output, then PWM pulse will generate at this pin, that is not as expectation of **PHASE\_SHIFTED\_SINGLE** behavior . Therefore, the port pin associate to odd channel should not be configured as PWM output.

**Table 4-68. Attribute ChannelEdgeSetup (PwmFtmChannels) detailed description**

Property	Value
Label	Edge configuration setting for current channel
Type	ENUMERATION
Origin	Custom
POSTBUILDVARIANTVALUE	true

*Table continues on the next page...*



**Table 4-68. Attribute ChannelEdgeSetup (PwmFtmChannels) detailed description (continued)**

Property	Value
Symbolic Name	false
Default	INDEPENDENT
Range	INDEPENDENT PHASE_SHIFTED_SINGLE COMBINED_SYNCED COMBINED_COMPLEMENTARY PHASE_SHIFTED_SYNCED PHASE_SHIFTED_COMPLEMENTARY

#### 4.6.3.14.3 ChannelCombDeadTimeEn (PwmFtmChannels)

Switch to enable/disable Deadtime for combine channels.

**Table 4-69. Attribute ChannelCombDeadTimeEn (PwmFtmChannels) detailed description**

Property	Value
Label	Enable Deadtime on combined channels
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

#### 4.6.3.14.4 PwmPhaseShift (PwmFtmChannels)

Phase shift value (in tick) for channel in PHASE\_SHIFTED\_SINGLE/  
PHASE\_SHIFTED\_SYNCED/PHASE\_SHIFTED\_COMPLEMENTARY

**Table 4-70. Attribute PwmPhaseShift (PwmFtmChannels) detailed description**

Property	Value
Label	Phase Shift Ticks
Type	INTEGER
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	0
Invalid	Range 65553 >=0

### 4.6.3.15 Form PwmFtmChannelFaultSettings

Global Fault configuration. AUTOSAR Requirements: not specified.

Is included by form: [Form PwmFtmModule](#)

Figure 4-16. Tresos Plugin snapshot for PwmFtmChannelFaultSettings form.

#### 4.6.3.15.1 PwmFilterValue (PwmFtmChannelFaultSettings)

Value used for debouncing Fault inputs.

Table 4-71. Attribute PwmFilterValue (PwmFtmChannelFaultSettings) detailed description

Property	Value
Label	Pwm Fault Filter Value
Type	INTEGER
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	0
Invalid	Range <=15 >=0

#### 4.6.3.15.2 PwmDisableOutputOnFault0 (PwmFtmChannelFaultSettings)

Disable the PWM output on detection of fault on fault channel 0. AUTOSAR Requirements: not specified.

**Table 4-72. Attribute PwmDisableOutputOnFault0 (PwmFtmChannelFaultSettings) detailed description**

Property	Value
Label	Disable Channel Output On Fault 0
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

#### 4.6.3.15.3 PwmDisableOutputOnFault1 (PwmFtmChannelFaultSettings)

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

Requirements: not specified.

**Table 4-73. Attribute PwmDisableOutputOnFault1 (PwmFtmChannelFaultSettings) detailed description**

Property	Value
Label	Disable Channel Output On Fault 1
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

#### 4.6.3.15.4 PwmDisableOutputOnFault2 (PwmFtmChannelFaultSettings)

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

Requirements: not specified.

**Table 4-74. Attribute PwmDisableOutputOnFault2 (PwmFtmChannelFaultSettings) detailed description**

Property	Value
Label	Disable Channel Output On Fault 2
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

#### 4.6.3.15.5 PwmDisableOutputOnFault3 (PwmFtmChannelFaultSettings)

Disable the PWM output on detection of fault on fault channel 3. AUTOSAR Requirements: not specified.

**Table 4-75. Attribute PwmDisableOutputOnFault3 (PwmFtmChannelFaultSettings) detailed description**

Property	Value
Label	Disable Channel Output On Fault 3
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

#### 4.6.3.15.6 PwmFtmFaultOutputState (PwmFtmModule)

Specify the fault state for the PWM channel output during fault conditions.

- **SAFE\_VALUES**: outputs will be placed into safe values when fault events in ongoing (defined by polarity).
- **TRISTATE**: outputs will be tri-stated when fault event is ongoing.

AUTOSAR Requirements: not specified.

**Table 4-76. Attribute PwmFtmFaultOutputState (PwmFtmModule) detailed description**

Property	Value
Label	Channel Output on Fault
Type	ENUMERATION
Origin	Custom
Symbolic Name	false
Default	SAFE_VALUES
Range	SAFE_VALUES TRISTATE

#### 4.6.3.15.7 Form PwmFtmChannelFault0Settings

Global Configuration for Fault 0.

**Is included by form :** [Form PwmFtmChannelFaultSettings](#)

**Note**

Ftm Fault0 Settings can only be activated by checking "Disable Channel Output On Fault 0" in [Form PwmFtmChannelFaultSettings](#) tab.

**Figure 4-17. Tressos Plugin snapshot for PwmFtmChannelFault0Settings form.**

#### 4.6.3.15.7.1 PwmFault0Polarity (PwmFtmChannelFault0Settings)

Set polarity for Fault 0 Input Pin (Checking the box means fault will be active LOW, else fault will be active HIGH). AUTOSAR Requirements: not specified.

**Table 4-77. Attribute PwmFault0Polarity (PwmFtmChannelFault0Settings) detailed description**

Property	Value
Label	Set polarity for Fault 0 Input Pin
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

#### 4.6.3.15.7.2 PwmEnableFault0Filter (PwmFtmChannelFault0Settings)

Enable a debounce filter for Fault 0 Input pin. AUTOSAR Requirements: not specified.

**Table 4-78. Attribute PwmEnableFault0Filter (PwmFtmChannelFault0Settings) detailed description**

Property	Value
Label	Enable a debounce filter for Fault 0
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

4.6.3.15.7.3 PwmFault0Notification (PwmFtmChannelFault0Settings)

User callback function.

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.

Table 4-79. Attribute PwmFault0Notification (PwmFtmChannelFault0Settings) detailed description

Property	Value
Type	FUNCTION-NAME
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	NULL

4.6.3.15.8 Form PwmFtmChannelFault1Settings

Global Configuration for Fault 1.

Is included by form: [Form PwmFtmChannelFaultSettings](#)

Note

Ftm Fault1 Settings can only be activated by checking "Disable Channel Output On Fault 1" in [Form PwmFtmChannelFaultSettings](#) tab.

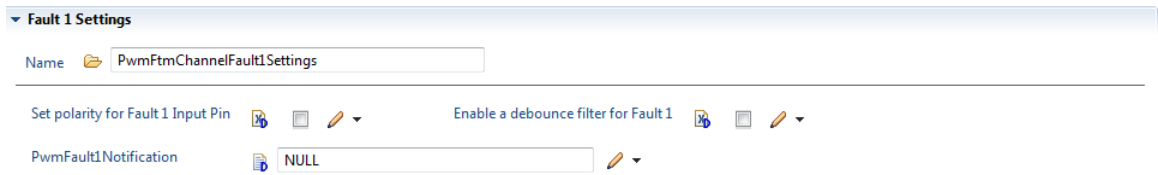


Figure 4-18. Tresos Plugin snapshot for PwmFtmChannelFault1Settings form.

4.6.3.15.8.1 PwmFault1Polarity (PwmFtmChannelFault1Settings)

Set polarity for Fault 1 Input Pin (Checking the box means fault will be active LOW, else fault will be active HIGH). AUTOSAR Requirements: not specified.

**Table 4-80. Attribute PwmFault1Polarity (PwmFtmChannelFault1Settings) detailed description**

Property	Value
Label	Set polarity for Fault 1 Input Pin
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

#### 4.6.3.15.8.2 PwmEnableFault1Filter (PwmFtmChannelFault1Settings)

Enable a debounce filter for Fault 1 Input pin. AUTOSAR Requirements: not specified.

**Table 4-81. Attribute PwmEnableFault1Filter (PwmFtmChannelFault1Settings) detailed description**

Property	Value
Label	Enable a debounce filter for Fault 1
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

#### 4.6.3.15.8.3 PwmFault1Notification (PwmFtmChannelFault1Settings)

User callback function.

### 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.

**Table 4-82. Attribute PwmFault1Notification (PwmFtmChannelFault1Settings) detailed description**

Property	Value
Type	FUNCTION-NAME
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false

*Table continues on the next page...*

**Table 4-82. Attribute PwmFault1Notification (PwmFtmChannelFault1Settings) detailed description (continued)**

Property	Value
Default	NULL

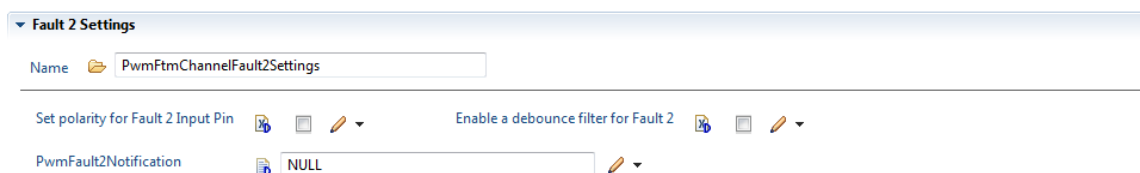
#### 4.6.3.15.9 Form PwmFtmChannelFault2Settings

Global Configuration for Fault 2.

Is included by form : [Form PwmFtmChannelFaultSettings](#)

##### Note

Ftm Fault2 Settings can only be activated by checking "Disable Channel Output On Fault 2" in [Form PwmFtmChannelFaultSettings](#) tab.



**Figure 4-19. Tresos Plugin snapshot for PwmFtmChannelFault2Settings form.**

#### 4.6.3.15.9.1 PwmFault2Polarity (PwmFtmChannelFault2Settings)

Set polarity for Fault 2 Input Pin (Checking the box means fault will be active LOW, else fault will be active HIGH). AUTOSAR Requirements: not specified.

**Table 4-83. Attribute PwmFault2Polarity (PwmFtmChannelFault2Settings) detailed description**

Property	Value
Label	Set polarity for Fault 2 Input Pin
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

#### 4.6.3.15.9.2 PwmEnableFault2Filter (PwmFtmChannelFault2Settings)

Enable a debounce filter for Fault 2 Input pin. AUTOSAR Requirements: not specified.



**Table 4-84. Attribute PwmEnableFault2Filter (PwmFtmChannelFault2Settings) detailed description**

Property	Value
Label	Enable a debounce filter for Fault 2
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

#### 4.6.3.15.9.3 PwmFault2Notification (PwmFtmChannelFault2Settings)

User callback function.

#### 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.

**Table 4-85. Attribute PwmFault2Notification (PwmFtmChannelFault2Settings) detailed description**

Property	Value
Type	FUNCTION-NAME
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	NULL

#### 4.6.3.15.10 Form PwmFtmChannelFault3Settings

Global Configuration for Fault 3.

Is included by form : [Form PwmFtmChannelFaultSettings](#)

#### Note

Ftm Fault3 Settings can only be activated by checking "Disable Channel Output On Fault 3" in [Form PwmFtmChannelFaultSettings](#) tab.

**Figure 4-20. Tresos Plugin snapshot for PwmFtmChannelFault3Settings form.**

#### 4.6.3.15.10.1 PwmFault3Polarity (PwmFtmChannelFault3Settings)

Set polarity for Fault 3 Input Pin (Checking the box means fault will be active LOW, else fault will be active HIGH). AUTOSAR Requirements: not specified.

**Table 4-86. Attribute PwmFault3Polarity (PwmFtmChannelFault3Settings) detailed description**

Property	Value
Label	Set polarity for Fault 3 Input Pin
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

#### 4.6.3.15.10.2 PwmEnableFault3Filter (PwmFtmChannelFault3Settings)

Enable a debounce filter for Fault 3 Input pin. AUTOSAR Requirements: not specified.

**Table 4-87. Attribute PwmEnableFault3Filter (PwmFtmChannelFault3Settings) detailed description**

Property	Value
Label	Enable a debounce filter for Fault 3
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

#### 4.6.3.15.10.3 PwmFault3Notification (PwmFtmChannelFault3Settings)

User callback function.

**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.

**Table 4-88. Attribute PwmFault3Notification (PwmFtmChannelFault3Settings) detailed description**

Property	Value
Type	FUNCTION-NAME
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	NULL

### 4.6.3.16 Form PwmExternalTriggerSettings

Global Trigger configuration.

Is included by form: [Form PwmFtmModule](#)

**Figure 4-21. Tresos Plugin snapshot for PwmExternalTriggerSettings form.**

#### 4.6.3.16.1 PwmEnableExternalTriggerChannel0 (PwmExternalTriggerSettings)

Enable the generation of the channel trigger on Trigger channel 0.

**Table 4-89. Attribute PwmEnableExternalTriggerChannel0 (PwmExternalTriggerSettings) detailed description**

Property	Value
Label	Enable External Trigger Channel 0

*Table continues on the next page...*

**Table 4-89. Attribute PwmEnableExternalTriggerChannel0 (PwmExternalTriggerSettings) detailed description (continued)**

Property	Value
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

#### 4.6.3.16.2 PwmEnableExternalTriggerChannel1 (PwmExternalTriggerSettings)

Enable the generation of the channel trigger on Trigger channel 1.

**Table 4-90. Attribute PwmEnableExternalTriggerChannel1 (PwmExternalTriggerSettings) detailed description**

Property	Value
Label	Enable External Trigger Channel 1
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

#### 4.6.3.16.3 PwmEnableExternalTriggerChannel2 (PwmExternalTriggerSettings)

Enable the generation of the channel trigger on Trigger channel 2.

**Table 4-91. Attribute PwmEnableExternalTriggerChannel2 (PwmExternalTriggerSettings) detailed description**

Property	Value
Label	Enable External Trigger Channel 2
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

#### 4.6.3.16.4 PwmEnableExternalTriggerChannel3 (PwmExternalTriggerSettings)

Enable the generation of the channel trigger on Trigger channel 3.

**Table 4-92. Attribute PwmEnableExternalTriggerChannel3 (PwmExternalTriggerSettings) detailed description**

Property	Value
Label	Enable External Trigger Channel 3
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

#### 4.6.3.16.5 PwmEnableExternalTriggerChannel4 (PwmExternalTriggerSettings)

Enable the generation of the channel trigger on Trigger channel 4.

**Table 4-93. Attribute PwmEnableExternalTriggerChannel4 (PwmExternalTriggerSettings) detailed description**

Property	Value
Label	Enable External Trigger Channel 4
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

#### 4.6.3.16.6 PwmEnableExternalTriggerChannel5 (PwmExternalTriggerSettings)

Enable the generation of the channel trigger on Trigger channel 5.

**Table 4-94. Attribute PwmEnableExternalTriggerChannel5 (PwmExternalTriggerSettings) detailed description**

Property	Value
Label	Enable External Trigger Channel 5
Type	BOOLEAN

*Table continues on the next page...*

**Table 4-94. Attribute PwmEnableExternalTriggerChannel5 (PwmExternalTriggerSettings) detailed description (continued)**

Property	Value
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

#### 4.6.3.16.7 PwmEnableExternalTriggerChannel6 (PwmExternalTriggerSettings)

Enable the generation of the channel trigger on Trigger channel 6.

**Table 4-95. Attribute PwmEnableExternalTriggerChannel6 (PwmExternalTriggerSettings) detailed description**

Property	Value
Label	Enable External Trigger Channel 6
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

#### 4.6.3.16.8 PwmEnableExternalTriggerChannel7 (PwmExternalTriggerSettings)

Enable the generation of the channel trigger on Trigger channel 7.

**Table 4-96. Attribute PwmEnableExternalTriggerChannel7 (PwmExternalTriggerSettings) detailed description**

Property	Value
Label	Enable External Trigger Channel 7
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

#### 4.6.3.16.9 PwmEnableInitializationTrigger (PwmExternalTriggerSettings)

Enable the generation of the trigger when the FTM counter is equal to the CNTIN register.

**Table 4-97. Attribute PwmEnableInitializationTrigger (PwmExternalTriggerSettings) detailed description**

Property	Value
Label	Enable Initialization Trigger
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false





**How to Reach Us:****Home Page:**[nxp.com](http://nxp.com)**Web Support:**[nxp.com/support](http://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](http://nxp.com/SalesTermsandConditions).

While NXP has implemented advanced security features, all products may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on customer's applications and products, and NXP accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP, the NXP logo, NXP SECURE CONNECTIONS FOR A SMARTER WORLD, COOLFLUX, EMBRACE, GREENCHIP, HITAG, I2C BUS, ICODE, JCOP, LIFE VIBES, MIFARE, MIFARE CLASSIC, MIFARE DESFire, MIFARE PLUS, MIFARE FLEX, MANTIS, MIFARE ULTRALIGHT, MIFARE4MOBILE, MIGLO, NTAG, ROADLINK, SMARTLX, SMARTMX, STARPLUG, TOPFET, TRENCHMOS, UCODE, Freescale, the Freescale logo, AltiVec, C-5, CodeTEST, CodeWarrior, ColdFire, ColdFire+, C-Ware, the Energy Efficient Solutions logo, Kinetis, Layerscape, MagniV, mobileGT, PEG, PowerQUICC, Processor Expert, QorIQ, QorIQ Qonverge, Ready Play, SafeAssure, the SafeAssure logo, StarCore, Symphony, VortiQa, Vybrid, Airfast, BeeKit, BeeStack, CoreNet, Flexis, MXC, Platform in a Package, QUICC Engine, SMARTMOS, Tower, TurboLink, and UMEMS are trademarks of NXP B.V. All other product or service names are the property of their respective owners. AMBA, Arm, Arm7, Arm7TDMI, Arm9, Arm11, Artisan, big.LITTLE, Cordio, CoreLink, CoreSight, Cortex, DesignStart, DynamIQ, Jazelle, Keil, Mali, Mbed, Mbed Enabled, NEON, POP, RealView, SecurCore, Socrates, Thumb, TrustZone, ULINK, ULINK2, ULINK-ME, ULINK-PLUS, ULINKpro, µVision, Versatile are trademarks or registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. The related technology may be protected by any or all of patents, copyrights, designs and trade secrets. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. The Power Architecture and Power.org word marks and the Power and Power.org logos and related marks are trademarks and service marks licensed by Power.org.

© 2019 NXP B.V.

Document Number UM2PWMASR4.2 Rev0002 R1.0.2  
Revision 1.0