User Manual

for S32K14X PWM Driver

Document Number: UM2PWMASR4.2 Rev0002 R1.0.1
Rev. 1.0

Contents

Section number **Title** **Page**

Chapter 1
Revision History

Chapter 2
Introduction

2.1 Supported Derivatives....17

2.2 Overview....17

2.3 About this Manual....18

2.4 Acronyms and Definitions....18

2.5 Reference List....19

Chapter 3
Driver

3.1 Requirements....21

3.2 Driver Design Summary....21

3.3 Hardware Resources....21

3.4 Deviation from Requirements....26

3.5 The Phase Shift Functionality....27

3.6 Driver usage and configuration tips....28

3.6.1 Configure PwmChannelConfigSet....28

3.6.2 Configure PwmFtmModule....29

3.6.3 Configure logic channel in PwmChannel....29

3.6.4 Implement errata....29

3.7 Runtime Errors....30

3.8 Software specification....30

3.8.1 Define Reference....30

3.8.1.1 Define PWM_VENDOR_ID....30

3.8.1.2 Define PWM_MODULE_ID....30

3.8.1.3 Define PWM_AR_RELEASE_MAJOR_VERSION....30

3.8.1.4 Define PWM_AR_RELEASE_MINOR_VERSION....31

3.8.1.5 Define PWM_AR_RELEASE_REVISION_VERSION..... 31
3.8.1.6 Define PWM_SW_MAJOR_VERSION..... 31
3.8.1.7 Define PWM_SW_MINOR_VERSION.....31
3.8.1.8 Define PWM_SW_PATCH_VERSION.....32
3.8.1.9 Define PWM_DUTY_CYCLE_100..... 32
3.8.1.10 Define PWM_E_PARAM_CONFIG.....32
3.8.1.11 Define PWM_E_UNINIT..... 32
3.8.1.12 Define PWM_E_PARAM_CHANNEL..... 33
3.8.1.13 Define PWM_E_PERIOD_UNCHANGEABLE..... 33
3.8.1.14 Define PWM_E_ALREADY_INITIALIZED.....34
3.8.1.15 Define PWM_E_PARAM_POINTER.....34
3.8.1.16 Define PWM_E_PARAM_NOTIFICATION..... 35
3.8.1.17 Define PWM_E_PARAM_NOTIFICATION_NULL.....35
3.8.1.18 Define PWM_E_DUTYCYCLE_RANGE.....35
3.8.1.19 Define PWM_E_COUNTERBUS..... 36
3.8.1.20 Define PWM_E_CHANNEL_OFFSET_VALUE.....36
3.8.1.21 Define PWM_E_OPWMB_CHANNEL_OFFSET_DUTYCYCLE_RANGE..... 37
3.8.1.22 Define PWM_E_PARAM_INSTANCE.....37
3.8.1.23 Define PWM_E_OPWMT_CHANNEL_TRIGGER_RANGE.....38
3.8.1.24 Define PWM_E_OUTPUT_STATE.....38
3.8.1.25 Define PWM_E_UNEXPECTED_ISR..... 39
3.8.1.26 Define PWM_E_PARAM_PHASESHIFT_RANGE..... 39
3.8.1.27 Define PWM_E_CHANNEL_PHASE_SHIFT_WITHOUT_COMBINE.....40
3.8.1.28 Define PWM_E_PARAM_SYNCHRONOUS_MODIFIED_COMBINE.....40
3.8.1.29 Define PWM_E_TRIGGER_MASK.....40
3.8.1.30 Define PWM_E_NOT_DISENGAGED.....41
3.8.1.31 Define PWM_E_POWER_STATE_NOT_SUPPORTED..... 41
3.8.1.32 Define PWM_E_TRANSITION_NOT_POSSIBLE..... 42
3.8.1.33 Define PWM_E_PERIPHERAL_NOT_PREPARED.....42

3.8.1.34 Define PWM_INIT_ID....42
3.8.1.35 Define PWM_DEINIT_ID....43
3.8.1.36 Define PWM_SETDUTYCYCLE_ID....43
3.8.1.37 Define PWM_SETPERIODANDDUTY_ID....44
3.8.1.38 Define PWM_SETOUTPUTTOIDLE_ID....44
3.8.1.39 Define PWM_GETOUTPUTSTATE_ID....44
3.8.1.40 Define PWM_DISABLENOTIFICATION_ID....45
3.8.1.41 Define PWM_ENABLENOTIFICATION_ID....45
3.8.1.42 Define PWM_GETVERSIONINFO_ID....45
3.8.1.43 Define PWM_SETPOWERSTATE_ID....46
3.8.1.44 Define PWM_GETCURRENTPOWERSTATE_ID....46
3.8.1.45 Define PWM_GETTARGETPOWERSTATE_ID....46
3.8.1.46 Define PWM_PREPAREPOWERSTATE_ID....47
3.8.1.47 Define PWM_GETCHANNELSTATE_ID....47
3.8.1.48 Define PWM_FORCEOUTPUTTOZERO_ID....47
3.8.1.49 Define PWM_SETCOUNTERBUS_ID....48
3.8.1.50 Define PWM_SETCHANNELOUTPUT_ID....48
3.8.1.51 Define PWM_SETTRIGGERDELAY_ID....49
3.8.1.52 Define PWM_BUFFERTRANSFERENDIS_ID....49
3.8.1.53 Define PWM_SETCLOCKMODE_ID....49
3.8.1.54 Define PWM_SYNCUPDATE_ID....50
3.8.1.55 Define PWM_SETPERIODANDDUTY_NO_UPDATE_ID....50
3.8.1.56 Define PWM_SETDUTYCYCLE_NO_UPDATE_ID....50
3.8.1.57 Define PWM_SELECTCOMMONTIMEBASE_ID....51
3.8.1.58 Define PWM_SETPHASESHIFT_ID....51
3.8.1.59 Define PWM_SETPHASESHIFTNOUPDATE_ID....51
3.8.1.60 Define PWM_ENABLETRIGGER_ID....52
3.8.1.61 Define PWM_DISABLETRIGGER_ID....52
3.8.1.62 Define PWM_RESETCOUNTERENABLE_ID....53

3.8.1.63 Define PWM_RESETCOUNTERDISABLE_ID 53
3.8.1.64 Define PWM_MASKOUTPUT_ID 53
3.8.1.65 Define PWM_UNMASKOUTPUT_ID 54
3.8.1.66 Define PWM_FTM_0_USED 54
3.8.1.67 Define PWM_FTM_1_USED 54
3.8.1.68 Define PWM_FTM_2_USED 54
3.8.1.69 Define PWM_FTM_3_USED 55
3.8.1.70 Define PWM_FTM_4_USED 55
3.8.1.71 Define PWM_FTM_5_USED 55
3.8.1.72 Define PWM_FTM_6_USED 55
3.8.1.73 Define PWM_FTM_7_USED 56
3.8.1.74 Define FTM_0_CH_0_CH_1_ISR 56
3.8.1.75 Define FTM_0_CH_2_CH_3_ISR 56
3.8.1.76 Define FTM_0_CH_4_CH_5_ISR 56
3.8.1.77 Define FTM_0_CH_6_CH_7_ISR 57
3.8.1.78 Define FTM_0_OVF_ISR 57
3.8.1.79 Define FTM_0_FAULT_ISR 57
3.8.1.80 Define FTM_1_CH_0_CH_1_ISR 57
3.8.1.81 Define FTM_1_CH_2_CH_3_ISR 58
3.8.1.82 Define FTM_1_CH_4_CH_5_ISR 58
3.8.1.83 Define FTM_1_CH_6_CH_7_ISR 58
3.8.1.84 Define FTM_1_OVF_ISR 58
3.8.1.85 Define FTM_1_FAULT_ISR 58
3.8.1.86 Define FTM_2_CH_0_CH_1_ISR 59
3.8.1.87 Define FTM_2_CH_2_CH_3_ISR 59
3.8.1.88 Define FTM_2_CH_4_CH_5_ISR 59
3.8.1.89 Define FTM_2_CH_6_CH_7_ISR 59
3.8.1.90 Define FTM_2_OVF_ISR 60
3.8.1.91 Define FTM_2_FAULT_ISR 60

3.8.1.92 Define FTM_3_CH_0_CH_1_ISR ... 60
3.8.1.93 Define FTM_3_CH_2_CH_3_ISR ... 60
3.8.1.94 Define FTM_3_CH_4_CH_5_ISR ... 61
3.8.1.95 Define FTM_3_CH_6_CH_7_ISR ... 61
3.8.1.96 Define FTM_3_OVF_ISR ... 61
3.8.1.97 Define FTM_3_FAULT_ISR ... 61
3.8.1.98 Define FTM_4_CH_0_CH_1_ISR ... 62
3.8.1.99 Define FTM_4_CH_2_CH_3_ISR ... 62
3.8.1.100 Define FTM_4_CH_4_CH_5_ISR ... 62
3.8.1.101 Define FTM_4_CH_6_CH_7_ISR ... 62
3.8.1.102 Define FTM_4_OVF_ISR ... 62
3.8.1.103 Define FTM_4_FAULT_ISR ... 63
3.8.1.104 Define FTM_5_CH_0_CH_1_ISR ... 63
3.8.1.105 Define FTM_5_CH_2_CH_3_ISR ... 63
3.8.1.106 Define FTM_5_CH_4_CH_5_ISR ... 63
3.8.1.107 Define FTM_5_CH_6_CH_7_ISR ... 64
3.8.1.108 Define FTM_5_OVF_ISR ... 64
3.8.1.109 Define FTM_5_FAULT_ISR ... 64
3.8.1.110 Define FTM_6_CH_0_CH_1_ISR ... 64
3.8.1.111 Define FTM_6_CH_2_CH_3_ISR ... 65
3.8.1.112 Define FTM_6_CH_4_CH_5_ISR ... 65
3.8.1.113 Define FTM_6_CH_6_CH_7_ISR ... 65
3.8.1.114 Define FTM_6_OVF_ISR ... 65
3.8.1.115 Define FTM_6_FAULT_ISR ... 66
3.8.1.116 Define FTM_7_CH_0_CH_1_ISR ... 66
3.8.1.117 Define FTM_7_CH_2_CH_3_ISR ... 66
3.8.1.118 Define FTM_7_CH_4_CH_5_ISR ... 66
3.8.1.119 Define FTM_7_CH_6_CH_7_ISR ... 66
3.8.1.120 Define FTM_7_OVF_ISR ... 67

3.8.1.121 Define FTM_7_FAULT_ISR ... 67
3.8.1.122 Define FTM_0_ISR ... 67
3.8.1.123 Define FTM_1_ISR ... 67
3.8.1.124 Define PWM_FTM_CHANNEL ... 68
3.8.1.125 Define PWM_FTM_CHANNELS_NO ... 68
3.8.1.126 Define PWM_FTM_MODULE_NO ... 68
3.8.1.127 Define PWM_HW_CHANNELS_NO_U8 ... 68
3.8.1.128 Define PWM_FTM_MODULE_CHANNELS_NO ... 69
3.8.1.129 Define PWM_FTM_MODULE_FAULT_NO ... 69
3.8.1.130 Define PWM_DEV_ERROR_DETECT ... 69
3.8.1.131 Define PWM_DUTY_PERIOD_UPDATED_ENDPERIOD ... 69
3.8.1.132 Define PWM_DUTYCYCLE_UPDATED_ENDPERIOD ... 70
3.8.1.133 Define PWM_FAULT_SUPPORTED ... 70
3.8.1.134 Define PWM_INDEX ... 70
3.8.1.135 Define PWM_NOTIFICATION_SUPPORTED ... 71
3.8.1.136 Define PWM_PRECOMPILE_SUPPORT ... 71
3.8.1.137 Define PWM_FTM_ENABLE_EXT_TRIGGERS ... 71
3.8.1.138 Define PWM_SET_DUTY_CYCLE_API ... 71
3.8.1.139 Define PWM_SET_OUTPUT_TO_IDLE_API ... 72
3.8.1.140 Define PWM_SET_PERIOD_AND_DUTY_API ... 72
3.8.1.141 Define PWM_SET_CLOCK_MODE_API ... 72
3.8.1.142 Define PWM_SELECT_COMMON_TIMEBASE_API ... 72
3.8.1.143 Define PWM_VERSION_INFO_API ... 73
3.8.1.144 Define PWM_GET_CHANNEL_STATE_API ... 73
3.8.1.145 Define PWM_GET_OUTPUT_STATE_API ... 73
3.8.1.146 Define PWM_FORCE_OUTPUT_TO_ZERO_API ... 73
3.8.1.147 Define PWM_DE_INIT_API ... 74
3.8.1.148 Define PWM_SET_PHASE_SHIFT_API ... 74
3.8.1.149 Define PWM_SET_PHASE_SHIFT_NO_UPDATE_API ... 74

3.8.1.150 Define PWM_ENABLE_TRIGEER_API ... 74
3.8.1.151 Define PWM_DIABLE_TRIGEER_API ... 75
3.8.1.152 Define PWM_RESET_COUNTER_API ... 75
3.8.1.153 Define PWM_ENABLE_MASKING_OPERATIONS ... 75
3.8.1.154 Define PWM_SYNC_UPDATE_API ... 76
3.8.1.155 Define PWM_UPDATE_DUTY_SYNCHRONOUS ... 76
3.8.1.156 Define PWM_SET_DUTY_CYCLE_NO_UPDATE_API ... 76
3.8.1.157 Define PWM_SET_PERIOD_AND_DUTY_NO_UPDATE_API ... 76
3.8.1.158 Define PWM_SELECT_COMMON_TIMEBASE_API ... 72
3.8.1.159 Define PWM_ENABLE_PHASE_SHIFT ... 77
3.8.2 Enum Reference ... 77
3.8.2.1 Enumeration Pwm_ChannelClassType ... 77
3.8.2.2 Enumeration Pwm_OutputStateType ... 78
3.8.2.3 Enumeration Pwm_EdgeNotificationType ... 78
3.8.2.4 Enumeration Pwm_PrescalerType ... 79
3.8.2.5 Enumeration Pwm_GlobalStateType ... 79
3.8.2.6 Enumeration Pwm_AlignmentType ... 79
3.8.2.7 Enumeration Pwm_DataUdateType ... 80
3.8.3 Structs Reference ... 81
3.8.3.1 Structure Pwm_ChannelConfigType ... 81
3.8.3.2 Structure Pwm_ConfigType ... 82
3.8.3.3 Structure Pwm_IpConfigType ... 83
3.8.3.4 Structure Pwm_Ftm_IpConfigType ... 84
3.8.3.5 Structure Pwm_Ftm_ModuleConfigType ... 84
3.8.3.6 Structure Pwm_Ftm_ChannelConfigType ... 86
3.8.4 Types Reference ... 87
3.8.4.1 Typedef Pwm_NotifyType ... 87
3.8.4.2 Typedef Pwm_ChannelType ... 87
3.8.4.3 Typedef Pwm_IpChannelConfigType ... 88

3.8.4.4 Typedef Pwm_ChannelIpType ... 88
3.8.5 Function Reference ... 88
3.8.5.1 Function Pwm_Init ... 88
3.8.5.2 Function Pwm_DeInit ... 90
3.8.5.3 Function Pwm_SetPeriodAndDuty ... 90
3.8.5.4 Function Pwm_SetDutyCycle ... 92
3.8.5.5 Function Pwm_SetOutputToIdle ... 93
3.8.5.6 Function Pwm_DisableNotification ... 94
3.8.5.7 Function Pwm_EnableNotification ... 95
3.8.5.8 Function Pwm_GetChannelState ... 96
3.8.5.9 Function Pwm_GetOutputState ... 97
3.8.5.10 Function Pwm_ForceOutputToZero ... 98
3.8.5.11 Function Pwm_SetClockMode ... 98
3.8.5.12 Function Pwm_SelectCommonTimebase ... 99
3.8.5.13 Function Pwm_GetVersionInfo ... 100
3.8.5.14 Function Pwm_DisableTrigger ... 100
3.8.5.15 Function Pwm_EnableTrigger ... 101
3.8.5.16 Function Pwm_MaskOutputs ... 102
3.8.5.17 Function Pwm_UnMaskOutputs ... 104
3.8.5.18 Function Pwm_ResetCounter ... 105
3.8.5.19 Function Pwm_ResetCounterEnable ... 105
3.8.5.20 Function Pwm_ResetCounterDisable ... 106
3.8.5.21 Function Pwm_SetDutyCycle_NoUpdate ... 106
3.8.5.22 Function Pwm_SetPeriodAndDuty_NoUpdate ... 108
3.8.5.23 Function Pwm_SetPhaseShift ... 109
3.8.5.24 Function Pwm_SetPhaseShift_NoUpdate ... 110
3.8.5.25 Function Pwm_SyncUpdate ... 112
3.8.6 Variables Reference ... 112
3.9 Symbolic Names Disclaimer ... 112

Chapter 4
Tresos Configuration Plug-in

4.1 Configuration elements of PWM 113
4.2 Form IMPLEMENTATION_CONFIG_VARIANT 113
4.3 Form PwmConfigurationOfOptApiServices 114
4.3.1 PwmDeInitApi (PwmConfigurationOfOptApiServices) 114
4.3.2 PwmGetOutputState (PwmConfigurationOfOptApiServices) 114
4.3.3 PwmSetDutyCycle (PwmConfigurationOfOptApiServices) 115
4.3.4 PwmSetOutputToIdle (PwmConfigurationOfOptApiServices) 115
4.3.5 PwmSetPeriodAndDuty (PwmConfigurationOfOptApiServices) 116
4.3.6 PwmVersionInfoApi (PwmConfigurationOfOptApiServices) 116
4.3.7 PwmGetChannelStateApi (PwmConfigurationOfOptApiServices) 116
4.3.8 PwmForceOutputToZeroApi (PwmConfigurationOfOptApiServices) 117
4.3.9 PwmSetDutyCycle_NoUpdate (PwmConfigurationOfOptApiServices) 117
4.3.10 PwmSetPeriodAndDuty_NoUpdate (PwmConfigurationOfOptApiServices) 118
4.3.11 PwmSetPhaseShift (PwmConfigurationOfOptApiServices) 118
4.3.12 PwmSetPhaseShiftNoUpdate (PwmConfigurationOfOptApiServices) 118
4.3.13 PwmEnableTrigger (PwmConfigurationOfOptApiServices) 119
4.3.14 PwmDiableTrigger (PwmConfigurationOfOptApiServices) 119
4.3.15 PwmSwResetCounter (PwmConfigurationOfOptApiServices) 119
4.4 Form PwmGeneral 120
4.4.1 PwmDevErrorDetect (PwmGeneral) 120
4.4.2 PwmDutycycleUpdatedEndperiod (PwmGeneral) 120
4.4.3 PwmNotificationSupported (PwmGeneral) 121
4.4.4 PwmEnableDualClockMode (PwmGeneral) 121
4.4.5 PwmCommonTimebaseSupported (PwmGeneral) 122
4.4.6 PwmPeriodUpdatedEndperiod (PwmGeneral) 122
4.4.7 PwmIndex (PwmGeneral) 122
4.4.8 PwmFaultSupport (PwmGeneral) 123

4.4.9 PwmFtmEnableExtTrigger (PwmGeneral)....123
4.4.10 PwmEnablePhaseShift (PwmGeneral)....123
4.4.11 PwmMultiChannelSynch (PwmGeneral)....124
4.4.12 EnableMaskingOperations (PwmGeneral)....124
4.5 Form CommonPublishedInformation....124
4.5.1 ArReleaseMajorVersion (CommonPublishedInformation)....125
4.5.2 ArReleaseMinorVersion (CommonPublishedInformation)....125
4.5.3 ArReleaseRevisionVersion (CommonPublishedInformation)....126
4.5.4 ModuleId (CommonPublishedInformation)....126
4.5.5 SwMajorVersion (CommonPublishedInformation)....127
4.5.6 SwMinorVersion (CommonPublishedInformation)....127
4.5.7 SwPatchVersion (CommonPublishedInformation)....128
4.5.8 VendorApiInfix (CommonPublishedInformation)....128
4.5.9 VendorId (CommonPublishedInformation)....128
4.6 Form PwmChannelConfigSet....129
4.6.1 Form PwmChannel....130
4.6.1.1 PwmChannelId (PwmChannel)....131
4.6.1.2 PwmPeriodInTicks (PwmChannel)....132
4.6.1.3 PwmPeriodDefault (PwmChannel)....132
4.6.1.4 PwmChannelClass (PwmChannel)....133
4.6.1.5 PwmPolarity (PwmChannel)....133
4.6.1.6 PwmDutycycleDefault (PwmChannel)....133
4.6.1.7 PwmIdleState (PwmChannel)....134
4.6.1.8 PwmNotification (PwmChannel)....134
4.6.1.9 PwmFtmChannel (PwmChannel)....135
4.6.2 Form PwmFtmModule....135
4.6.2.1 FtmModule (PwmFtmModule)....136
4.6.2.2 PwmPrescaler (PwmFtmModule)....137
4.6.2.3 PwmPrescaler_Alternate (PwmFtmModule)....137

4.6.2.4 PwmClockRef (PwmFtmModule)..... 138
4.6.2.5 PwmClockSelection (PwmFtmModule)..... 138
4.6.2.6 PwmChanEdgeAlignment (PwmFtmModule)..... 139
4.6.2.7 PwmUpdatedEndPeriod (PwmFtmModule)..... 139
4.6.2.8 PwmUpdatedMiddlePeriod (PwmFtmModule)..... 140
4.6.2.9 PwmReloadFrequency (PwmFtmModule)..... 140
4.6.2.10 PwmDeadTimeCount (PwmFtmModule)..... 141
4.6.2.11 DeadTimePrescaler (PwmFtmModule)..... 142
4.6.2.12 PwmBDMEnable (PwmFtmModule)..... 142
4.6.2.13 PwmFtmFaultFunctionality (PwmFtmModule)..... 143
4.6.2.14 Form PwmFtmChannels..... 144
4.6.2.14.1 PwmFtmChannelId (PwmFtmChannels)..... 145
4.6.2.14.2 ChannelEdgeSetup (PwmFtmChannels)..... 145
4.6.2.14.3 ChannelCombDeadTimeEn (PwmFtmChannels)..... 146
4.6.2.14.4 PwmPhaseShift (PwmFtmChannels)..... 146
4.6.2.15 Form PwmFtmChannelFaultSettings..... 147
4.6.2.15.1 PwmFilterValue (PwmFtmChannelFaultSettings)..... 147
4.6.2.15.2 PwmDisableOutputOnFault0 (PwmFtmChannelFaultSettings)..... 148
4.6.2.15.3 PwmDisableOutputOnFault1 (PwmFtmChannelFaultSettings)..... 148
4.6.2.15.4 PwmDisableOutputOnFault2 (PwmFtmChannelFaultSettings)..... 148
4.6.2.15.5 PwmDisableOutputOnFault3 (PwmFtmChannelFaultSettings)..... 149
4.6.2.15.6 PwmFtmFaultOutputState (PwmFtmModule)..... 149
4.6.2.15.7 Form PwmFtmChannelFault0Settings..... 150
4.6.2.15.7.1 PwmFault0Polarity (PwmFtmChannelFault0Settings)..... 150
4.6.2.15.7.2 PwmEnableFault0Filter (PwmFtmChannelFault0Settings)..... 151
4.6.2.15.7.3 PwmFault0Notification (PwmFtmChannelFault0Settings)..... 151
4.6.2.15.8 Form PwmFtmChannelFault1Settings..... 151
4.6.2.15.8.1 PwmFault1Polarity (PwmFtmChannelFault1Settings)..... 152
4.6.2.15.8.2 PwmEnableFault1Filter (PwmFtmChannelFault1Settings)..... 152

4.6.2.15.8.3 PwmFault1Notification (PwmFtmChannelFault1Settings).....153
4.6.2.15.9 Form PwmFtmChannelFault2Settings.....153
4.6.2.15.9.1 PwmFault2Polarity (PwmFtmChannelFault2Settings)..... 153
4.6.2.15.9.2 PwmEnableFault2Filter (PwmFtmChannelFault2Settings)..... 154
4.6.2.15.9.3 PwmFault2Notification (PwmFtmChannelFault2Settings).....154
4.6.2.15.10 Form PwmFtmChannelFault3Settings.....155
4.6.2.15.10.1 PwmFault3Polarity (PwmFtmChannelFault3Settings)..... 155
4.6.2.15.10.2 PwmEnableFault3Filter (PwmFtmChannelFault3Settings)..... 155
4.6.2.15.10.3 PwmFault3Notification (PwmFtmChannelFault3Settings).....156
4.6.2.16 Form PwmExternalTriggerSettings..... 156
4.6.2.16.1 PwmEnableExternalTriggerChannel0 (PwmExternalTriggerSettings)..... 157
4.6.2.16.2 PwmEnableExternalTriggerChannel1 (PwmExternalTriggerSettings)..... 157
4.6.2.16.3 PwmEnableExternalTriggerChannel2 (PwmExternalTriggerSettings)..... 158
4.6.2.16.4 PwmEnableExternalTriggerChannel3 (PwmExternalTriggerSettings)..... 158
4.6.2.16.5 PwmEnableExternalTriggerChannel4 (PwmExternalTriggerSettings)..... 158
4.6.2.16.6 PwmEnableExternalTriggerChannel5 (PwmExternalTriggerSettings)..... 159
4.6.2.16.7 PwmEnableExternalTriggerChannel6 (PwmExternalTriggerSettings)..... 159
4.6.2.16.8 PwmEnableExternalTriggerChannel7 (PwmExternalTriggerSettings)..... 159
4.6.2.16.9 PwmEnableInitializationTrigger (PwmExternalTriggerSettings)..... 160

Chapter 1
Revision History

Table 1-1. Revision History

Revision	Date	Author	Description
1.0	13/07/2018	NXP MCAL Team	Updated version for ASR 4.2.2S32K14X1.0.1 Release

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 intented to be used with the following microcontroller devices of NXP Semiconductors .

Table 2-1. S32K14X Derivatives

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

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
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
DET	Development Error Tracer
C/CPP	C and C++ Source Code
ECU	Electronic Control Unit

Table continues on the next page...

Table 2-2. Acronyms and Definitions (continued)

Term	Definition
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. 7, 04/2018
3	S32K118 Mask Set Errata for Mask 0N97V (0N97V)	26/02/2018
4	S32K142 Mask Set Errata for Mask 0N33V (0N33V)	30/11/2017
5	S32K144 Mask Set Errata for Mask 0N57U (0N57U)	30/11/2017
6	S32K146 Mask Set Errata for Mask 0N73V (0N73V)	30/11/2017
7	S32K148 Mask Set Errata for Mask 0N20V (0N20V)	30/11/2017

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 implementation of the Pwm driver for this platform uses FlexTimer modules (FTM).

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.

3.3 Hardware Resources

Table 3-1. PWM Hardware channels availability for S32K118 family

Package	FTM module	FTM channel available	FTM fault input available
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

Table continues on the next page...

Table 3-1. PWM Hardware channels availability for S32K118 family (continued)

	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
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_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

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

Package	FTM module	FTM channel available	FTM fault input available
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
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
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

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_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
	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
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
	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

Table continues on the next page...

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

	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_00162	N/S	The Pwm Driver shall support synchronuous and asynchronous power state transitions, depending on the value of the configuration parameter PwmPowerStateAsynchTransitionMode.	eMIOS Hw does not support asynchronous power state transitions
SWS_Pwm_00164	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 The Phase Shift Functionality

PWM phase shift provides a group of channels which have difference phase for each specific channels. Current implementation supports 2 types of phase shift.

The traditional phase shift bases on Modified Combine mode of FTM IP. 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, paramenter PwmGeneral/PwmEnablePhaseShift should be enable first. The reference channel need 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 IP. 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.

3.6 Driver usage and configuration tips

Basically, PWM driver should be configured in 3 steps: Configure PwmChannelConfigSet, configure FTM module in PwmFtmModule and configure logic channel in PwmChannel.

3.6.1 Configure PwmChannelConfigSet

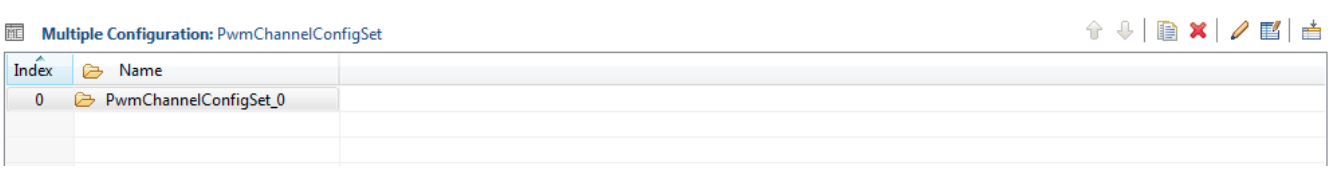


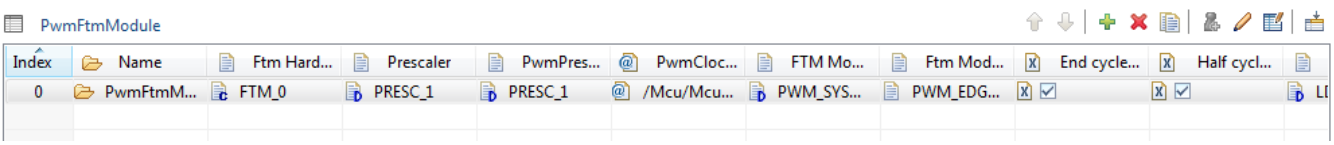
Figure 3-1. Tresos Plugin snapshot for PwmChannelConfigSet form.

Configuring PwmChannelConfigSet can be divided into two parts: configure PwmFtmModule and PwmChannel.

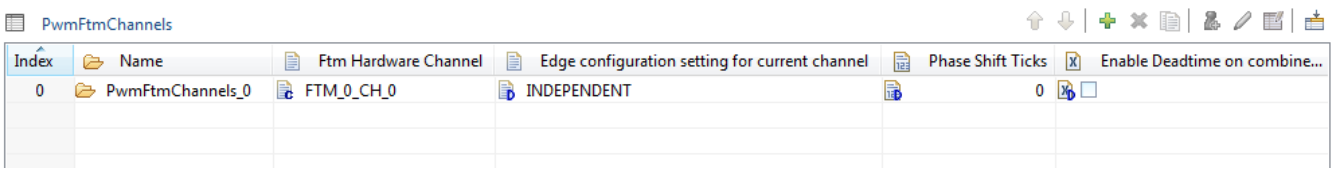
3.6.2 Configure PwmFtmModule

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.

- Create PwmFtmModule Profile(s)

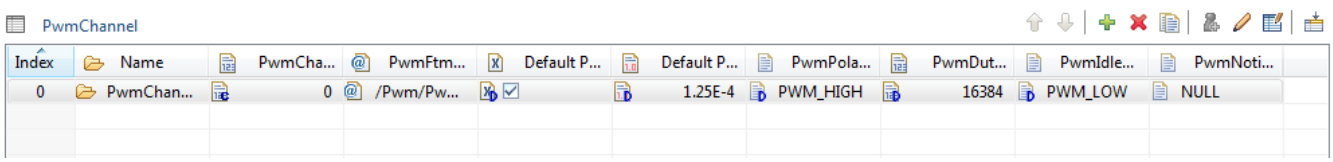


- Create PwmFtmChannel Profile(s) in each PwmFtmModule



3.6.3 Configure logic channel in PwmChannel

This list contains logical channel working as object of PWM APIs. Each logic channel refer to PWM FTM channel which has been configured from Configure PwmFtmModule



3.6.4 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.

3.7 Runtime Errors

None.

3.8 Software specification

The following sections contains driver software specifications.

3.8.1 Define Reference

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

3.8.1.1 Define PWM_VENDOR_ID

Table 3-8. Define PWM_VENDOR_ID Description

Name	PWM_VENDOR_ID
Initializer	43

3.8.1.2 Define PWM_MODULE_ID

Table 3-9. Define PWM_MODULE_ID Description

Name	PWM_MODULE_ID
Initializer	121

3.8.1.3 Define PWM_AR_RELEASE_MAJOR_VERSION

Table 3-10. Define PWM_AR_RELEASE_MAJOR_VERSION Description

Name	PWM_AR_RELEASE_MAJOR_VERSION
Initializer	4

3.8.1.4 Define PWM_AR_RELEASE_MINOR_VERSION

Table 3-11. Define PWM_AR_RELEASE_MINOR_VERSION Description

Name	PWM_AR_RELEASE_MINOR_VERSION
Initializer	2

3.8.1.5 Define PWM_AR_RELEASE_REVISION_VERSION

Table 3-12. Define PWM_AR_RELEASE_REVISION_VERSION Description

Name	PWM_AR_RELEASE_REVISION_VERSION
Initializer	2

3.8.1.6 Define PWM_SW_MAJOR_VERSION

Table 3-13. Define PWM_SW_MAJOR_VERSION Description

Name	PWM_SW_MAJOR_VERSION
Initializer	1

3.8.1.7 Define PWM_SW_MINOR_VERSION

Table 3-14. Define PWM_SW_MINOR_VERSION Description

Name	PWM_SW_MINOR_VERSION
Initializer	0

3.8.1.8 Define PWM_SW_PATCH_VERSION

Table 3-15. Define PWM_SW_PATCH_VERSION Description

Name	PWM_SW_PATCH_VERSION
Initializer	2

3.8.1.9 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

Name	PWM_DUTY_CYCLE_100
Initializer	((uint16) 0x8000U)

3.8.1.10 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

Name	PWM_E_PARAM_CONFIG
Initializer	0x10U

3.8.1.11 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

Name	PWM_E_UNINIT
Initializer	0x11U

3.8.1.12 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

Name	PWM_E_PARAM_CHANNEL
Initializer	0x12U

3.8.1.13 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

Name	PWM_E_PERIOD_UNCHANGEABLE
Initializer	0x13U

3.8.1.14 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

Name	PWM_E_ALREADY_INITIALIZED
Initializer	0x14U

3.8.1.15 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

Name	PWM_E_PARAM_POINTER
Initializer	0x15U

3.8.1.16 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

Name	PWM_E_PARAM_NOTIFICATION
Initializer	0x30U

3.8.1.17 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

Name	PWM_E_PARAM_NOTIFICATION_NULL
Initializer	0x31U

3.8.1.18 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

Name	PWM_E_DUTYCYCLE_RANGE
Initializer	0x32U

3.8.1.19 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

Name	PWM_E_COUNTERBUS
Initializer	0x33U

3.8.1.20 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

Name	PWM_E_CHANNEL_OFFSET_VALUE
Initializer	0x34U

3.8.1.21 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

Name	PWM_E_OPWMB_CHANNEL_OFFSET_DUTYCYCLE_RANGE
Initializer	0x35U

3.8.1.22 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

Name	PWM_E_PARAM_INSTANCE
Initializer	(0x36U)

3.8.1.23 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

Name	PWM_E_OPWMT_CHANNEL_TRIGGER_RANGE
Initializer	0x37U

3.8.1.24 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

Name	PWM_E_OUTPUT_STATE
Initializer	0x38U

3.8.1.25 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

Name	PWM_E_UNEXPECTED_ISR
Initializer	0x39U

3.8.1.26 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

Name	PWM_E_PARAM_PHASESHIFT_RANGE
Initializer	0x3AU

3.8.1.27 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

Name	PWM_E_CHANNEL_PHASE_SHIFT_WITHOUT_COMBINE
Initializer	0x3BU

3.8.1.28 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

Name	PWM_E_PARAM_SYNCHRONOUS_MODIFIED_COMBINE
Initializer	0x3CU

3.8.1.29 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

Name	PWM_E_TRIGGER_MASK
Initializer	0x3DU

3.8.1.30 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

Name	PWM_E_NOT_DISENGAGED
Initializer	0x16U

3.8.1.31 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

Name	PWM_E_POWER_STATE_NOT_SUPPORTED
Initializer	0x17U

3.8.1.32 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

Name	PWM_E_TRANSITION_NOT_POSSIBLE
Initializer	0x18U

3.8.1.33 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

Name	PWM_E_PERIPHERAL_NOT_PREPARED
Initializer	0x19U

3.8.1.34 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

Name	PWM_INIT_ID
Initializer	0x00U

3.8.1.35 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

Name	PWM_DEINIT_ID
Initializer	0x01U

3.8.1.36 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

Name	PWM_SETDUTYCYCLE_ID
Initializer	0x02U

3.8.1.37 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

Name	PWM_SETPERIODANDDUTY_ID
Initializer	0x03U

3.8.1.38 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

Name	PWM_SETOUTPUTTOIDLE_ID
Initializer	0x04U

3.8.1.39 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

Name	PWM_GETOUTPUTSTATE_ID
Initializer	0x05U

3.8.1.40 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

Name	PWM_DISABLENOTIFICATION_ID
Initializer	0x06U

3.8.1.41 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

Name	PWM_ENABLENOTIFICATION_ID
Initializer	0x07U

3.8.1.42 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

Name	PWM_GETVERSIONINFO_ID
Initializer	0x08U

3.8.1.43 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

Name	PWM_SETPOWERSTATE_ID
Initializer	0x09U

3.8.1.44 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

Name	PWM_GETCURRENTPOWERSTATE_ID
Initializer	0x0AU

3.8.1.45 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

Name	PWM_GETTARGETPOWERSTATE_ID
Initializer	0x0BU

3.8.1.46 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

Name	PWM_PREPAREPOWERSTATE_ID
Initializer	0x0CU

3.8.1.47 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

Name	PWM_GETCHANNELSTATE_ID
Initializer	0x20U

3.8.1.48 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

Name	PWM_FORCEOUTPUTTOZERO_ID
Initializer	0x21U

3.8.1.49 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

Name	PWM_SETCOUNTERBUS_ID
Initializer	0x22U

3.8.1.50 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

Name	PWM_SETCHANNELOUTPUT_ID
Initializer	0x23U

3.8.1.51 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

Name	PWM_SETTRIGGERDELAY_ID
Initializer	0x24U

3.8.1.52 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

Name	PWM_BUFFERTRANSFERENDIS_ID
Initializer	0x26U

3.8.1.53 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

Name	PWM_SETCLOCKMODE_ID
Initializer	0x27U

3.8.1.54 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

Name	PWM_SYNCUPDATE_ID
Initializer	0x28U

3.8.1.55 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

Name	PWM_SETPERIODANDDUTY_NO_UPDATE_ID
Initializer	0x29U

3.8.1.56 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

Name	PWM_SETDUTYCYCLE_NO_UPDATE_ID
Initializer	0x2AU

3.8.1.57 Define PWM_SELECTCOMMONTIMEBASE_ID

API service ID of Pwm_SelectCommonTimebase function.

Details:

Parameters used when raising an error/exception

Table 3-64. Define PWM_SELECTCOMMONTIMEBASE_ID Description

Name	PWM_SELECTCOMMONTIMEBASE_ID
Initializer	0x2BU

3.8.1.58 Define PWM_SETPHASESHIFT_ID

API service ID of Pwm_SetPhaseShift function

Details:

Parameters used when raising an error/exception

Table 3-65. Define PWM_SETPHASESHIFT_ID Description

Name	PWM_SETPHASESHIFT_ID
Initializer	0x2CU

3.8.1.59 Define PWM_SETPHASESHIFTNOUPDATE_ID

API service ID of Pwm_SetPhaseShift_NoUpdate function

Details:

Parameters used when raising an error/exception

Table 3-66. Define PWM_SETPHASESHIFTNOUPDATE_ID Description

Name	PWM_SETPHASESHIFTNOUPDATE_ID
Initializer	0x2DU

3.8.1.60 Define PWM_ENABLETRIGGER_ID

API service ID of Pwm_EnableTrigger function

Details:

Parameters used when raising an error/exception

Table 3-67. Define PWM_ENABLETRIGGER_ID Description

Name	PWM_ENABLETRIGGER_ID
Initializer	0x2EU

3.8.1.61 Define PWM_DISABLETRIGGER_ID

API service ID of Pwm_DisableTrigger function

Details:

Parameters used when raising an error/exception

Table 3-68. Define PWM_DISABLETRIGGER_ID Description

Name	PWM_DISABLETRIGGER_ID
Initializer	0x2FU

3.8.1.62 Define PWM_RESETCOUNTERENABLE_ID

API service ID of Pwm_ResetCounterEnable function.

Details:

Parameters used when raising an error/exception.

Table 3-69. Define PWM_RESETCOUNTERENABLE_ID Description

Name	PWM_RESETCOUNTERENABLE_ID
Initializer	0x30U

3.8.1.63 Define PWM_RESETCOUNTERDISABLE_ID

API service ID of Pwm_ResetCounterDisable function.

Details:

Parameters used when raising an error/exception.

Table 3-70. Define PWM_RESETCOUNTERDISABLE_ID Description

Name	PWM_RESETCOUNTERDISABLE_ID
Initializer	0x31U

3.8.1.64 Define PWM_MASKOUTPUT_ID

API service ID of Pwm_MaskOutputs function.

Details:

Parameters used when raising an error/exception

Table 3-71. Define PWM_MASKOUTPUT_ID Description

Name	PWM_MASKOUTPUT_ID
Initializer	0x32U

3.8.1.65 Define PWM_UNMASKOUTPUT_ID

API service ID of Pwm_UnMaskOutputs function.

Details:

Parameters used when raising an error/exception.

Table 3-72. Define PWM_UNMASKOUTPUT_ID Description

Name	PWM_UNMASKOUTPUT_ID
Initializer	0x33U

3.8.1.66 Define PWM_FTM_0_USED

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

Table 3-73. Define PWM_FTM_0_USED Description

Name	PWM_FTM_0_USED
Initializer	

3.8.1.67 Define PWM_FTM_1_USED

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

Table 3-74. Define PWM_FTM_1_USED Description

Name	PWM_FTM_1_USED
Initializer	

3.8.1.68 Define PWM_FTM_2_USED

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

Table 3-75. Define PWM_FTM_2_USED Description

Name	PWM_FTM_2_USED
Initializer	

3.8.1.69 Define PWM_FTM_3_USED

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

Table 3-76. Define PWM_FTM_3_USED Description

Name	PWM_FTM_3_USED
Initializer	

3.8.1.70 Define PWM_FTM_4_USED

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

Table 3-77. Define PWM_FTM_4_USED Description

Name	PWM_FTM_4_USED
Initializer	

3.8.1.71 Define PWM_FTM_5_USED

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

Table 3-78. Define PWM_FTM_5_USED Description

Name	PWM_FTM_5_USED
Initializer	

3.8.1.72 Define PWM_FTM_6_USED

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

Table 3-79. Define PWM_FTM_6_USED Description

Name	PWM_FTM_6_USED
Initializer	

3.8.1.73 Define PWM_FTM_7_USED

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

Table 3-80. Define PWM_FTM_7_USED Description

Name	PWM_FTM_7_USED
Initializer	

3.8.1.74 Define FTM_0_CH_0_CH_1_ISR

Macros used to enable FTM Notification ISR code.

Table 3-81. Define FTM_0_CH_0_CH_1_ISR Description

Name	FTM_0_CH_0_CH_1_ISR
Initializer	

3.8.1.75 Define FTM_0_CH_2_CH_3_ISR

Macros used to enable FTM Notification ISR code.

Table 3-82. Define FTM_0_CH_2_CH_3_ISR Description

Name	FTM_0_CH_2_CH_3_ISR
Initializer	

3.8.1.76 Define FTM_0_CH_4_CH_5_ISR

Macros used to enable FTM Notification ISR code.

Table 3-83. Define FTM_0_CH_4_CH_5_ISR Description

Name	FTM_0_CH_4_CH_5_ISR

Table continues on the next page...

Table 3-83. Define FTM_0_CH_4_CH_5_ISR Description (continued)

Initializer	

3.8.1.77 Define FTM_0_CH_6_CH_7_ISR

Macros used to enable FTM Notification ISR code.

Table 3-84. Define FTM_0_CH_6_CH_7_ISR Description

Name	FTM_0_CH_6_CH_7_ISR
Initializer	

3.8.1.78 Define FTM_0_OVF_ISR

Macros used to enable FTM Notification ISR code.

Table 3-85. Define FTM_0_OVF_ISR Description

Name	FTM_0_OVF_ISR
Initializer	

3.8.1.79 Define FTM_0_FAULT_ISR

Macros used to enable FTM Notification ISR code.

Table 3-86. Define FTM_0_FAULT_ISR Description

Name	FTM_0_FAULT_ISR
Initializer	

3.8.1.80 Define FTM_1_CH_0_CH_1_ISR

Macros used to enable FTM Notification ISR code.

Table 3-87. Define FTM_1_CH_0_CH_1_ISR Description

Name	FTM_1_CH_0_CH_1_ISR
Initializer	

3.8.1.81 Define FTM_1_CH_2_CH_3_ISR

Macros used to enable FTM Notification ISR code.

Table 3-88. Define FTM_1_CH_2_CH_3_ISR Description

Name	FTM_1_CH_2_CH_3_ISR
Initializer	

3.8.1.82 Define FTM_1_CH_4_CH_5_ISR

Macros used to enable FTM Notification ISR code.

Table 3-89. Define FTM_1_CH_4_CH_5_ISR Description

Name	FTM_1_CH_4_CH_5_ISR
Initializer	

3.8.1.83 Define FTM_1_CH_6_CH_7_ISR

Macros used to enable FTM Notification ISR code.

Table 3-90. Define FTM_1_CH_6_CH_7_ISR Description

Name	FTM_1_CH_6_CH_7_ISR
Initializer	

3.8.1.84 Define FTM_1_OVF_ISR

Macros used to enable FTM Notification ISR code.

Table 3-91. Define FTM_1_OVF_ISR Description

Name	FTM_1_OVF_ISR
Initializer	

3.8.1.85 Define FTM_1_FAULT_ISR

Macros used to enable FTM Notification ISR code.

Table 3-92. Define FTM_1_FAULT_ISR Description

Name	FTM_1_FAULT_ISR
Initializer	

3.8.1.86 Define FTM_2_CH_0_CH_1_ISR

Macros used to enable FTM Notification ISR code.

Table 3-93. Define FTM_2_CH_0_CH_1_ISR Description

Name	FTM_2_CH_0_CH_1_ISR
Initializer	

3.8.1.87 Define FTM_2_CH_2_CH_3_ISR

Macros used to enable FTM Notification ISR code.

Table 3-94. Define FTM_2_CH_2_CH_3_ISR Description

Name	FTM_2_CH_2_CH_3_ISR
Initializer	

3.8.1.88 Define FTM_2_CH_4_CH_5_ISR

Macros used to enable FTM Notification ISR code.

Table 3-95. Define FTM_2_CH_4_CH_5_ISR Description

Name	FTM_2_CH_4_CH_5_ISR
Initializer	

3.8.1.89 Define FTM_2_CH_6_CH_7_ISR

Macros used to enable FTM Notification ISR code.

Table 3-96. Define FTM_2_CH_6_CH_7_ISR Description

Name	FTM_2_CH_6_CH_7_ISR
Initializer	

3.8.1.90 Define FTM_2_OVF_ISR

Macros used to enable FTM Notification ISR code.

Table 3-97. Define FTM_2_OVF_ISR Description

Name	FTM_2_OVF_ISR
Initializer	

3.8.1.91 Define FTM_2_FAULT_ISR

Macros used to enable FTM Notification ISR code.

Table 3-98. Define FTM_2_FAULT_ISR Description

Name	FTM_2_FAULT_ISR
Initializer	

3.8.1.92 Define FTM_3_CH_0_CH_1_ISR

Macros used to enable FTM Notification ISR code.

Table 3-99. Define FTM_3_CH_0_CH_1_ISR Description

Name	FTM_3_CH_0_CH_1_ISR
Initializer	

3.8.1.93 Define FTM_3_CH_2_CH_3_ISR

Macros used to enable FTM Notification ISR code.

Table 3-100. Define FTM_3_CH_2_CH_3_ISR Description

Name	FTM_3_CH_2_CH_3_ISR

Table continues on the next page...

Table 3-100. Define FTM_3_CH_2_CH_3_ISR Description (continued)

Initializer	

3.8.1.94 Define FTM_3_CH_4_CH_5_ISR

Macros used to enable FTM Notification ISR code.

Table 3-101. Define FTM_3_CH_4_CH_5_ISR Description

Name	FTM_3_CH_4_CH_5_ISR
Initializer	

3.8.1.95 Define FTM_3_CH_6_CH_7_ISR

Macros used to enable FTM Notification ISR code.

Table 3-102. Define FTM_3_CH_6_CH_7_ISR Description

Name	FTM_3_CH_6_CH_7_ISR
Initializer	

3.8.1.96 Define FTM_3_OVF_ISR

Macros used to enable FTM Notification ISR code.

Table 3-103. Define FTM_3_OVF_ISR Description

Name	FTM_3_OVF_ISR
Initializer	

3.8.1.97 Define FTM_3_FAULT_ISR

Macros used to enable FTM Notification ISR code.

Table 3-104. Define FTM_3_FAULT_ISR Description

Name	FTM_3_FAULT_ISR
Initializer	

3.8.1.98 Define FTM_4_CH_0_CH_1_ISR

Macros used to enable FTM Notification ISR code.

Table 3-105. Define FTM_4_CH_0_CH_1_ISR Description

Name	FTM_4_CH_0_CH_1_ISR
Initializer	

3.8.1.99 Define FTM_4_CH_2_CH_3_ISR

Macros used to enable FTM Notification ISR code.

Table 3-106. Define FTM_4_CH_2_CH_3_ISR Description

Name	FTM_4_CH_2_CH_3_ISR
Initializer	

3.8.1.100 Define FTM_4_CH_4_CH_5_ISR

Macros used to enable FTM Notification ISR code.

Table 3-107. Define FTM_4_CH_4_CH_5_ISR Description

Name	FTM_4_CH_4_CH_5_ISR
Initializer	

3.8.1.101 Define FTM_4_CH_6_CH_7_ISR

Macros used to enable FTM Notification ISR code.

Table 3-108. Define FTM_4_CH_6_CH_7_ISR Description

Name	FTM_4_CH_6_CH_7_ISR
Initializer	

3.8.1.102 Define FTM_4_OVF_ISR

Macros used to enable FTM Notification ISR code.

Table 3-109. Define FTM_4_OVF_ISR Description

Name	FTM_4_OVF_ISR
Initializer	

3.8.1.103 Define FTM_4_FAULT_ISR

Macros used to enable FTM Notification ISR code.

Table 3-110. Define FTM_4_FAULT_ISR Description

Name	FTM_4_FAULT_ISR
Initializer	

3.8.1.104 Define FTM_5_CH_0_CH_1_ISR

Macros used to enable FTM Notification ISR code.

Table 3-111. Define FTM_5_CH_0_CH_1_ISR Description

Name	FTM_5_CH_0_CH_1_ISR
Initializer	

3.8.1.105 Define FTM_5_CH_2_CH_3_ISR

Macros used to enable FTM Notification ISR code.

Table 3-112. Define FTM_5_CH_2_CH_3_ISR Description

Name	FTM_5_CH_2_CH_3_ISR
Initializer	

3.8.1.106 Define FTM_5_CH_4_CH_5_ISR

Macros used to enable FTM Notification ISR code.

Table 3-113. Define FTM_5_CH_4_CH_5_ISR Description

Name	FTM_5_CH_4_CH_5_ISR
Initializer	

3.8.1.107 Define FTM_5_CH_6_CH_7_ISR

Macros used to enable FTM Notification ISR code.

Table 3-114. Define FTM_5_CH_6_CH_7_ISR Description

Name	FTM_5_CH_6_CH_7_ISR
Initializer	

3.8.1.108 Define FTM_5_OVF_ISR

Macros used to enable FTM Notification ISR code.

Table 3-115. Define FTM_5_OVF_ISR Description

Name	FTM_5_OVF_ISR
Initializer	

3.8.1.109 Define FTM_5_FAULT_ISR

Macros used to enable FTM Notification ISR code.

Table 3-116. Define FTM_5_FAULT_ISR Description

Name	FTM_5_FAULT_ISR
Initializer	

3.8.1.110 Define FTM_6_CH_0_CH_1_ISR

Macros used to enable FTM Notification ISR code.

Table 3-117. Define FTM_6_CH_0_CH_1_ISR Description

Name	FTM_6_CH_0_CH_1_ISR

Table continues on the next page...

Table 3-117. Define FTM_6_CH_0_CH_1_ISR Description (continued)

Initializer	

3.8.1.111 Define FTM_6_CH_2_CH_3_ISR

Macros used to enable FTM Notification ISR code.

Table 3-118. Define FTM_6_CH_2_CH_3_ISR Description

Name	FTM_6_CH_2_CH_3_ISR
Initializer	

3.8.1.112 Define FTM_6_CH_4_CH_5_ISR

Macros used to enable FTM Notification ISR code.

Table 3-119. Define FTM_6_CH_4_CH_5_ISR Description

Name	FTM_6_CH_4_CH_5_ISR
Initializer	

3.8.1.113 Define FTM_6_CH_6_CH_7_ISR

Macros used to enable FTM Notification ISR code.

Table 3-120. Define FTM_6_CH_6_CH_7_ISR Description

Name	FTM_6_CH_6_CH_7_ISR
Initializer	

3.8.1.114 Define FTM_6_OVF_ISR

Macros used to enable FTM Notification ISR code.

Table 3-121. Define FTM_6_OVF_ISR Description

Name	FTM_6_OVF_ISR
Initializer	

3.8.1.115 Define FTM_6_FAULT_ISR

Macros used to enable FTM Notification ISR code.

Table 3-122. Define FTM_6_FAULT_ISR Description

Name	FTM_6_FAULT_ISR
Initializer	

3.8.1.116 Define FTM_7_CH_0_CH_1_ISR

Macros used to enable FTM Notification ISR code.

Table 3-123. Define FTM_7_CH_0_CH_1_ISR Description

Name	FTM_7_CH_0_CH_1_ISR
Initializer	

3.8.1.117 Define FTM_7_CH_2_CH_3_ISR

Macros used to enable FTM Notification ISR code.

Table 3-124. Define FTM_7_CH_2_CH_3_ISR Description

Name	FTM_7_CH_2_CH_3_ISR
Initializer	

3.8.1.118 Define FTM_7_CH_4_CH_5_ISR

Macros used to enable FTM Notification ISR code.

Table 3-125. Define FTM_7_CH_4_CH_5_ISR Description

Name	FTM_7_CH_4_CH_5_ISR
Initializer	

3.8.1.119 Define FTM_7_CH_6_CH_7_ISR

Macros used to enable FTM Notification ISR code.

Table 3-126. Define FTM_7_CH_6_CH_7_ISR Description

Name	FTM_7_CH_6_CH_7_ISR
Initializer	

3.8.1.120 Define FTM_7_OVF_ISR

Macros used to enable FTM Notification ISR code.

Table 3-127. Define FTM_7_OVF_ISR Description

Name	FTM_7_OVF_ISR
Initializer	

3.8.1.121 Define FTM_7_FAULT_ISR

Macros used to enable FTM Notification ISR code.

Table 3-128. Define FTM_7_FAULT_ISR Description

Name	FTM_7_FAULT_ISR
Initializer	

3.8.1.122 Define FTM_0_ISR

Macros used to enable FTM Notification ISR code for S32K118.

Table 3-129. Define FTM_0_ISR Description

Name	FTM_0_ISR
Initializer	

3.8.1.123 Define FTM_1_ISR

Macros used to enable FTM Notification ISR code for S32K118.

Table 3-130. Define FTM_1_ISR Description

Name	FTM_1_ISR
Initializer	

3.8.1.124 Define PWM_FTM_CHANNEL

Symbolic name for FTM channels.

Table 3-131. Define PWM_FTM_CHANNEL Description

Name	PWM_FTM_CHANNEL
Initializer	((`Pwm_ChannelIpType`) 0)

3.8.1.125 Define PWM_FTM_CHANNELS_NO

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

Table 3-132. Define PWM_FTM_CHANNELS_NO Description

Name	PWM_FTM_CHANNELS_NO
Initializer	(32U)

3.8.1.126 Define PWM_FTM_MODULE_NO

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

Table 3-133. Define PWM_FTM_MODULE_NO Description

Name	PWM_FTM_MODULE_NO
Initializer	(4U)

3.8.1.127 Define PWM_HW_CHANNELS_NO_U8

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

Table 3-134. Define PWM_HW_CHANNELS_NO Description

Name	PWM_HW_CHANNELS_NO_U8

Table continues on the next page...

Table 3-134. Define PWM_HW_CHANNELS_NO Description (continued)

Initializer	(32U)

3.8.1.128 Define PWM_FTM_MODULE_CHANNELS_NO

Define specifies the number of channels per each module.

Table 3-135. Define PWM_FTM_MODULE_CHANNELS_NO Description

Name	PWM_FTM_MODULE_CHANNELS_NO
Initializer	(8U)

3.8.1.129 Define PWM_FTM_MODULE_FAULT_NO

This define specifies the number of fault channels per module.

Table 3-136. Define PWM_FTM_MODULE_FAULT_NO Description

Name	PWM_FTM_MODULE_FAULT_NO
Initializer	(4U)

3.8.1.130 Define PWM_DEV_ERROR_DETECT

Switch for enabling the development error detection.

Table 3-137. Define PWM_DEV_ERROR_DETECT Description

Name	PWM_DEV_ERROR_DETECT
Initializer	(STD_ON)

3.8.1.131 Define PWM_DUTY_PERIOD_UPDATED_ENDPERIOD

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

Table 3-138. Define PWM_DUTY_PERIOD_UPDATED_ENDPERIOD Description

Name	PWM_DUTY_PERIOD_UPDATED_ENDPERIOD
Initializer	(STD_ON)

3.8.1.132 Define PWM_DUTYCYCLE_UPDATED_ENDPERIOD

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

Table 3-139. Define PWM_DUTYCYCLE_UPDATED_ENDPERIOD Description

Name	PWM_DUTYCYCLE_UPDATED_ENDPERIOD
Initializer	(STD_ON)

3.8.1.133 Define PWM_FAULT_SUPPORTED

Switch for enabling the fault functionality.

Table 3-140. Define PWM_FAULT_SUPPORTED Description

Name	PWM_FAULT_SUPPORTED
Initializer	(STD_OFF)

3.8.1.134 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 3-141. Define PWM_INDEX Description

Name	PWM_INDEX
Initializer	(0U)

3.8.1.135 Define PWM_NOTIFICATION_SUPPORTED

Switch to indicate that the notifications are supported.

Table 3-142. Define PWM_NOTIFICATION_SUPPORTED Description

Name	PWM_NOTIFICATION_SUPPORTED
Initializer	(STD_ON)

3.8.1.136 Define PWM_PRECOMPILE_SUPPORT

PWM pre-compile switch.

Table 3-143. Define PWM_PRECOMPILE_SUPPORT Description

Name	PWM_PRECOMPILE_SUPPORT
Initializer	(STD_OFF)

3.8.1.137 Define PWM_FTM_ENABLE_EXT_TRIGGERS

Switch to indicate that PWM_FTM_ENABLE_EXT_TRIGGERS is supported.

Table 3-144. Define PWM_FTM_ENABLE_EXT_TRIGGERS Description

Name	PWM_FTM_ENABLE_EXT_TRIGGERS
Initializer	(STD_OFF)

3.8.1.138 Define PWM_SET_DUTY_CYCLE_API

Switch to indicate that Pwm_SetDutyCycle API is supported.

Table 3-145. Define PWM_SET_DUTY_CYCLE_API Description

Name	PWM_SET_DUTY_CYCLE_API
Initializer	(STD_ON)

3.8.1.139 Define PWM_SET_OUTPUT_TO_IDLE_API

Switch to indicate that Pwm_SetOutputToIdle API is supported.

Table 3-146. Define PWM_SET_OUTPUT_TO_IDLE_API Description

Name	PWM_SET_OUTPUT_TO_IDLE_API
Initializer	(STD_ON)

3.8.1.140 Define PWM_SET_PERIOD_AND_DUTY_API

Switch to indicate that Pwm_SetPeriodAndDuty API is supported.

Table 3-147. Define PWM_SET_PERIOD_AND_DUTY_API Description

Name	PWM_SET_PERIOD_AND_DUTY_API
Initializer	(STD_ON)

3.8.1.141 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 3-148. Define PWM_SET_CLOCK_MODE_API Description

Name	PWM_SET_CLOCK_MODE_API
Initializer	(STD_ON)

3.8.1.142 Define PWM_SELECT_COMMON_TIMEBASE_API

Switch to indicate that PWM_SELECT_COMMON_TIMEBASE_API is supported.

Table 3-149. Define PWM_SELECT_COMMON_TIMEBASE_API Description

Name	PWM_SELECT_COMMON_TIMEBASE_API
Initializer	(STD_OFF)

3.8.1.143 Define PWM_VERSION_INFO_API

Switch to indicate that Pwm_GetVersionInfo API is supported.

Table 3-150. Define PWM_VERSION_INFO_API Description

Name	PWM_VERSION_INFO_API
Initializer	(STD_ON)

3.8.1.144 Define PWM_GET_CHANNEL_STATE_API

Switch to indicate that Pwm_GetChannelState API is supported.

Table 3-151. Define PWM_GET_CHANNEL_STATE_API Description

Name	PWM_GET_CHANNEL_STATE_API
Initializer	(STD_OFF)

3.8.1.145 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 3-152. Define PWM_GET_OUTPUT_STATE_API Description

Name	PWM_GET_OUTPUT_STATE_API
Initializer	(STD_OFF)

3.8.1.146 Define PWM_FORCE_OUTPUT_TO_ZERO_API

Switch to indicate that Pwm_ForceOutputToZero API is supported.

Table 3-153. Define PWM_FORCE_OUTPUT_TO_ZERO_API Description

Name	PWM_FORCE_OUTPUT_TO_ZERO_API
Initializer	(STD_OFF)

3.8.1.147 Define PWM_DE_INIT_API

Switch to indicate that Pwm_DeInit API is supported.

Table 3-154. Define PWM_DE_INIT_API Description

Name	PWM_DE_INIT_API
Initializer	(STD_ON)

3.8.1.148 Define PWM_SET_PHASE_SHIFT_API

Switch for enabling Pwm_SetPhaseShift API.

Table 3-155. Define PWM_SET_PHASE_SHIFT_API Description

Name	PWM_SET_PHASE_SHIFT_API
Initializer	(STD_ON)

3.8.1.149 Define PWM_SET_PHASE_SHIFT_NO_UPDATE_API

Switch for enabling Pwm_SetPhaseShift_NoUpdate API.

Table 3-156. Define PWM_SET_PHASE_SHIFT_NO_UPDATE_API Description

Name	PWM_SET_PHASE_SHIFT_NO_UPDATE_API
Initializer	(STD_ON)

3.8.1.150 Define PWM_ENABLE_TRIGEER_API

Switch for enabling Pwm_EnableTrigger API.

Table 3-157. Define PWM_ENABLE_TRIGEER_API Description

Name	PWM_ENABLE_TRIGEER_API
Initializer	(STD_ON)

3.8.1.151 Define PWM_DIABLE_TRIGEER_API

Switch for enabling Pwm_DisableTrigger API.

Table 3-158. Define PWM_DIABLE_TRIGEER_API Description

Name	PWM_DIABLE_TRIGEER_API
Initializer	(STD_ON)

3.8.1.152 Define PWM_RESET_COUNTER_API

Switch to indicate that Pwm_SwResetCounter API is supported.

Table 3-159. Define PWM_RESET_COUNTER_API Description

Name	PWM_RESET_COUNTER_API
Initializer	(STD_ON)

3.8.1.153 Define PWM_ENABLE_MASKING_OPERATIONS

Switch for enabling MaskOutput API.

Table 3-160. Define PWM_ENABLE_MASKING_OPERATIONS Description

Name	PWM_ENABLE_MASKING_OPERATIONS
Initializer	(STD_ON)

3.8.1.154 Define PWM_SYNC_UPDATE_API

Switch to indicate that Pwm_SyncUpdate API is supported.

Table 3-161. Define PWM_SYNC_UPDATE_API Description

Name	PWM_SYNC_UPDATE_API
Initializer	(STD_ON)

3.8.1.155 Define PWM_UPDATE_DUTY_SYNCHRONOUS

Switch to indicate that the notifications are supported.

Table 3-162. Define PWM_UPDATE_DUTY_SYNCHRONOUS Description

Name	PWM_UPDATE_DUTY_SYNCHRONOUS
Initializer	(STD_ON)

3.8.1.156 Define PWM_SET_DUTY_CYCLE_NO_UPDATE_API

Switch to indicate that PwmSetDutyCycle_NoUpdate API is supported.

Table 3-163. Define PWM_SET_DUTY_CYCLE_NO_UPDATE_API Description

Name	PWM_SET_DUTY_CYCLE_NO_UPDATE_API
Initializer	(STD_ON)

3.8.1.157 Define PWM_SET_PERIOD_AND_DUTY_NO_UPDATE_API

Switch to indicate that PwmSetPeriodAndDuty_NoUpdate API is supported.

Table 3-164. Define PWM_SET_PERIOD_AND_DUTY_NO_UPDATE_API Description

Name	PWM_SET_PERIOD_AND_DUTY_NO_UPDATE_API
Initializer	(STD_ON)

3.8.1.158 Define PWM_SELECT_COMMON_TIMEBASE_API

Switch to indicate that PWM_SELECT_COMMON_TIMEBASE_API is supported.

Table 3-165. Define PWM_SELECT_COMMON_TIMEBASE_API Description

Name	PWM_SELECT_COMMON_TIMEBASE_API
Initializer	(STD_OFF)

3.8.1.159 Define PWM_ENABLE_PHASE_SHIFT

Switch for enabling phase shift feature.

Table 3-166. Define PWM_ENABLE_PHASE_SHIFT Description

Name	PWM_ENABLE_PHASE_SHIFT
Initializer	(STD_OFF)

3.8.2 Enum Reference

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

3.8.2.1 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 3-167. Enumeration Pwm_ChannelClassType Values

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

3.8.2.2 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 3-168. Enumeration Pwm_OutputStateType Values

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

3.8.2.3 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 3-169. 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.

3.8.2.4 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 3-170. Enumeration Pwm_PrescalerType Values

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

3.8.2.5 Enumeration Pwm_GlobalStateType

Enum containing the possible states of the PWM driver.

Table 3-171. Enumeration Pwm_GlobalStateType Values

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

3.8.2.6 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 3-172. 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		One signal is generated that is aligned around the middle of the PWM period.
PWM_COMBINE_SYNCED		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.
PWM_COMBINE_COMPL		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		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		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		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.

3.8.2.7 Enumeration Pwm_DataUdateType

Details:

This enumeration specifies updating signal immediately or using synchronization.

Implements: Pwm_DataUdateType_enumeration

Table 3-173. Enumeration Pwm_DataUdateType Values

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

3.8.3 Structs Reference

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

3.8.3.1 Structure Pwm_ChannelConfigType

PWM channel high level configuration structure.

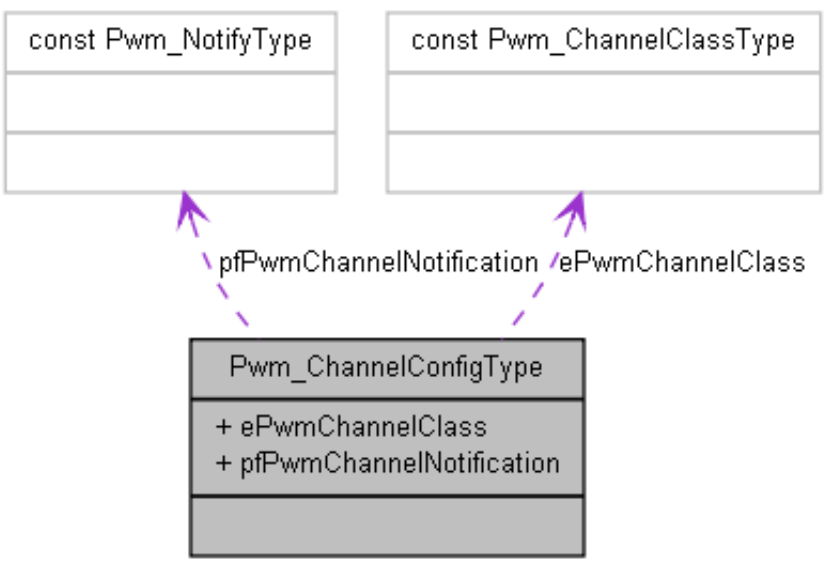


Figure 3-2. Struct Pwm_ChannelConfigType

Implements: Pwm_ChannelConfigType_struct

Declaration:

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

Table 3-174. Structure Pwm_ChannelConfigType member description

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

3.8.3.2 Structure Pwm_ConfigType

PWM high level configuration structure.

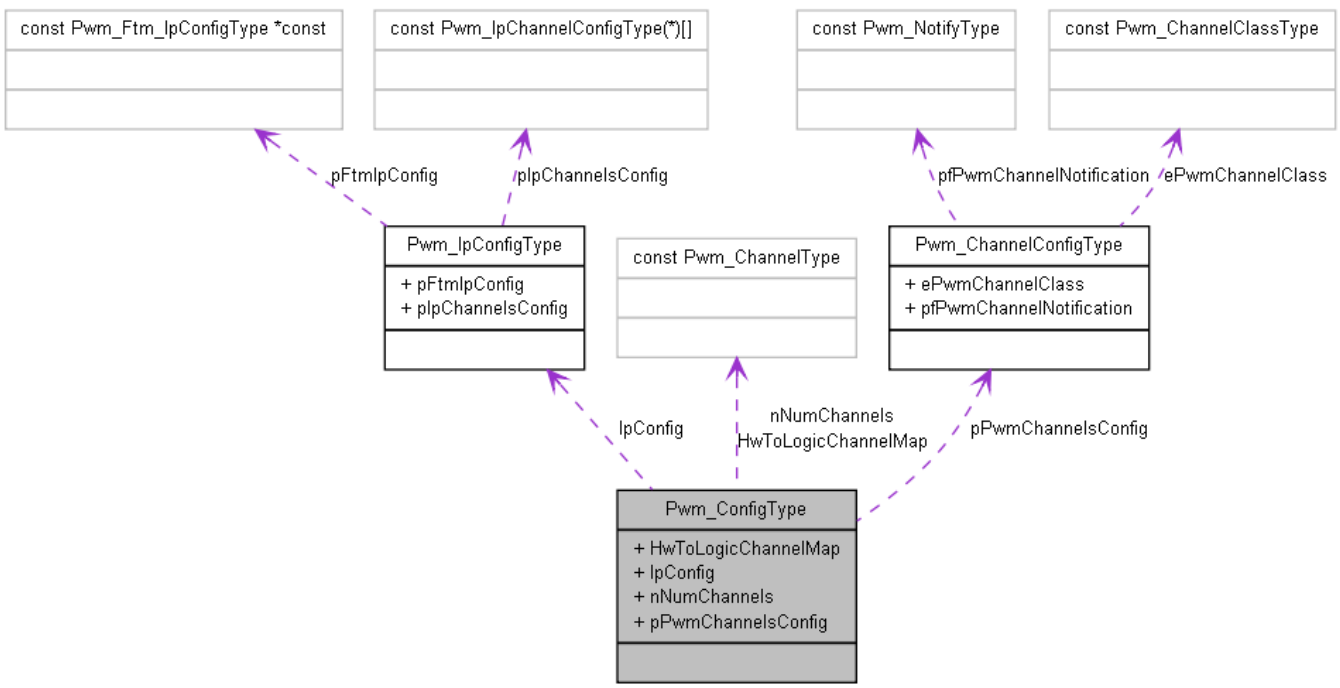


Figure 3-3. Struct Pwm_ConfigType

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 3-175. 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.

3.8.3.3 Structure Pwm_IpConfigType

Combined IP specific configuration structure.

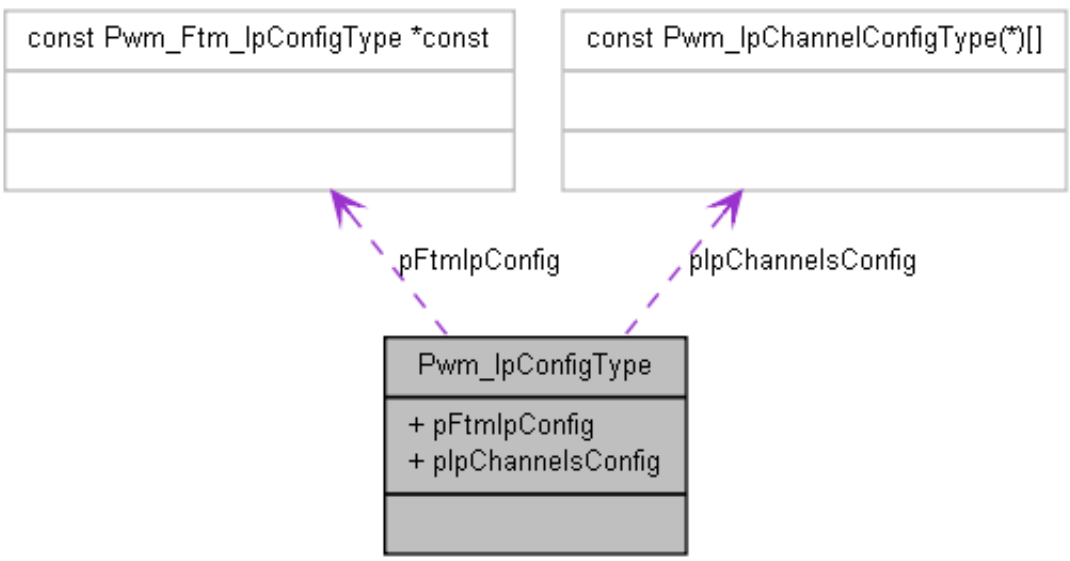


Figure 3-4. Struct Pwm_IpConfigType

Declaration:

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

Table 3-176. Structure Pwm_IpConfigType member description

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

3.8.3.4 Structure Pwm_Ftm_IpConfigType

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

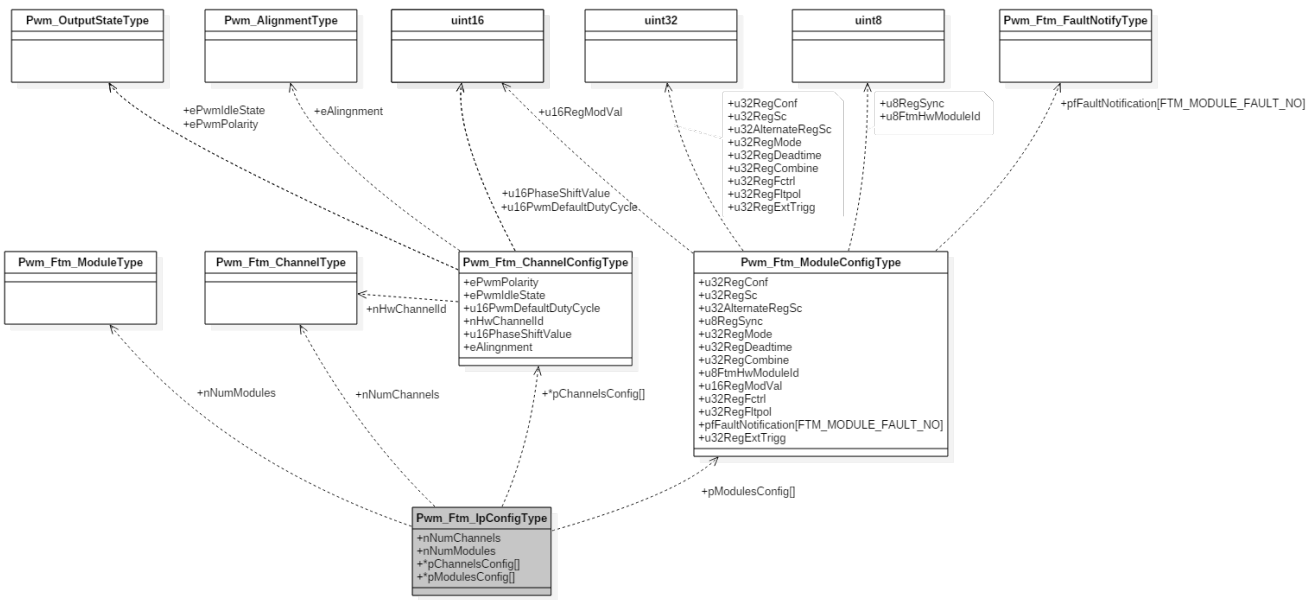


Figure 3-5. Struct Pwm_Ftm_IpConfigType

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 3-177. 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.

3.8.3.5 Structure Pwm_Ftm_ModuleConfigType

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

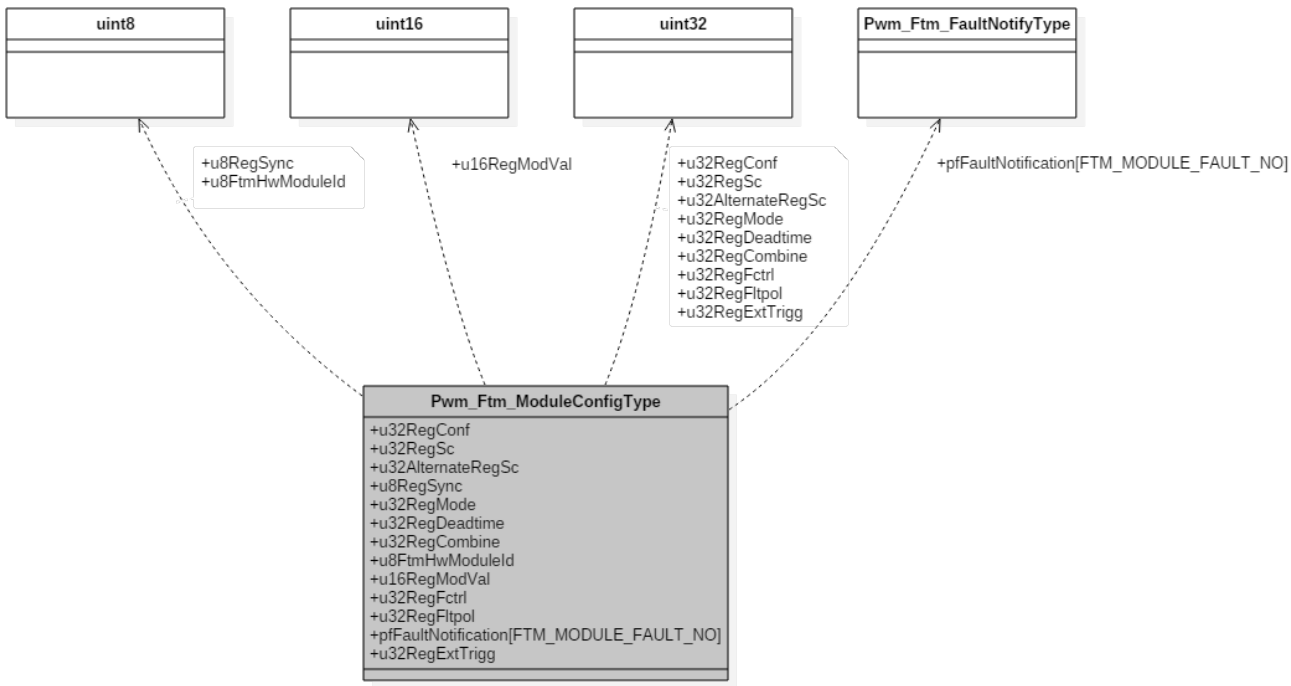


Figure 3-6. Struct Pwm_Ftm_ModuleConfigType

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 3-178. Structure Pwm_Ftm_ModuleConfigType member description

Member	Description
u32RegConf	Configuration register value.

Table continues on the next page...

Table 3-178. Structure Pwm_Ftm_ModuleConfigType member description (continued)

Member	Description
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.

3.8.3.6 Structure Pwm_Ftm_ChannelConfigType

Ftm IP specific channel configuration structure type.

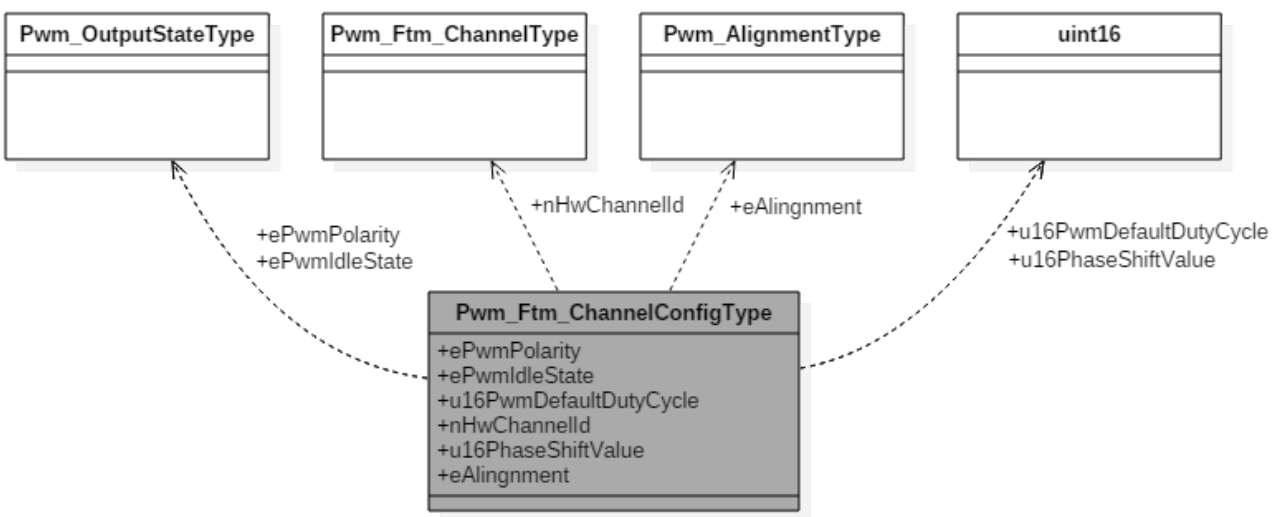


Figure 3-7. Struct Pwm_Ftm_ChannelConfigType

Implements: Pwm_Ftm_ChannelConfigType_struct

Declaration:

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

Table 3-179. 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].
eAlingnment	Channel alignment type.

3.8.4 Types Reference

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

3.8.4.1 Typedef Pwm_NotifyType

Channel notification typedef.

Details:

Pointer to notification handler

Type: void*

3.8.4.2 Typedef Pwm_ChannelType

PWM channel type.

Implements: Pwm_ChannelType_typedef

Type: uint8

3.8.4.3 Typedef Pwm_IpChannelConfigType

PWM channel high level configuration structure.

Implements: Pwm_IpChannelConfigType_typedef

Type: uint8

3.8.4.4 Typedef Pwm_ChannelIpType

IP type used to implement a PWM channel.

Implements: Pwm_ChannelIpType_typedef

Type: uint8

3.8.5 Function Reference

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

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

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

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 3-180. Pwm_Init Arguments

Type	Name	Direction	Description
const`Pwm_ConfigType`*	ConfigPtr	**input**	Pointer to PWM top configuration structure.

3.8.5.2 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);`

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

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

Return: void.

Implements: Pwm_SetPeriodAndDuty_Activity

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

Table 3-181. Pwm_SetPeriodAndDuty 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%.

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

Return: void.

Implements: Pwm_SetDutyCycle_Activity

Prototype: `void Pwm_SetDutyCycle(Pwm_ChannelType ChannelNumber, uint16 u16DutyCycle);`

Table 3-182. 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%.

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

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

After the call of the function Pwm_SetOutputToIdle, fixed period type channels shall be reactivated using only the API `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 3-183. Pwm_SetOutputToIdle Arguments

Type	Name	Direction	Description
`Pwm_ChannelType`	ChannelNumber	**input**	- pwm channel id.

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

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

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

Return: void.

Implements: Pwm_DisableNotification_Activity

Prototype: `void Pwm_DisableNotification(Pwm_ChannelType ChannelNumber);`

Table 3-184. Pwm_DisableNotification Arguments

Type	Name	Direction	Description
`Pwm_ChannelType`	ChannelNumber	**input**	PWM channel id.

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

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

Return: void.

Implements: Pwm_EnableNotification_Activity

Prototype: `void Pwm_EnableNotification(Pwm_ChannelType ChannelNumber, Pwm_EdgeNotificationType Notification);`

Table 3-185. Pwm_EnableNotification Arguments

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

3.8.5.8 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 3-186. Pwm_GetChannelState Arguments

Type	Name	Direction	Description
`Pwm_ChannelType`	ChannelNumber	**input**	PWM channel id.

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

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

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

Return: Pwm_OutputStateType PWM signal output logic value.

Implements: Pwm_GetOutputState_Activity

Prototype: `Pwm_OutputStateType Pwm_GetOutputState(Pwm_ChannelType ChannelNumber);`

Table 3-187. Pwm_GetOutputState Arguments

Type	Name	Direction	Description
`Pwm_ChannelType`	ChannelNumber	**input**	PWM channel id.

Table 3-188. 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.

3.8.5.10 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 3-189. 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).

3.8.5.11 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 3-190. Pwm_SetClockMode Arguments

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

3.8.5.12 Function Pwm_SelectCommonTimebase

Implementation specific function to updates the Global Timebase bits of configured modules.

Details:

This function is used to set the global timebase bits for modules that support the global timebase feature. The function selects the module that gives the common timebase and the modules that are use this timebase (as bits in u16ElementSyncList). Then it synchronizes the modules. For example:

- u16ElementSyncList is 0x0003 - modules 0 and 1 use the timebase given by ModuleId
- u16ElementSyncList is 0x0005 - modules 0 and 2 use the timebase given by ModuleId

Return: void.

Implements: Pwm_SelectCommonTimebase_Activity

Prototype: `void Pwm_SelectCommonTimebase(uint8 ModuleId, uint16 u16ElementSyncList);`

Table 3-191. Pwm_SelectCommonTimebase Arguments

Type	Name	Direction	Description
ModuleId	uint8	Input	PWM module id
u16ElementSyncList	uint16	Input	PWM module mask value

3.8.5.13 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 3-192. Pwm_GetVersionInfo Arguments

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

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

Return: void.

Implements: Pwm_DisableTrigger_Activity

Prototype: `void Pwm_DisableTrigger(uint8 u8TriggerHostId, uint16 u16TriggerMask);`

Table 3-193. Pwm_DisableTrigger Arguments

Type	Name	Direction	Description
uint8	u8TriggerHostId	**input**	Hardware module
uint16	u16TriggerMask	**input**	bit mask will be set to enable trigger with corresponding sources.

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

Return: void.

Implements: Pwm_EnableTrigger_Activity

Prototype: `void Pwm_EnableTrigger(uint8 u8TriggerHostId, uint16 u16TriggerMask);`

Table 3-194. 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.

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

Return: void.

Implements: Pwm_MaskOutputs_Activity

Prototype: `void Pwm_MaskOutputs(uint8 u8ModuleId, uint8 u8ChannelMask);`

Table 3-195. 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.

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

Return: void.

Implements: Pwm_UnMaskOutputs_Activity

Prototype: `void Pwm_UnMaskOutputs(uint8 u8ModuleId, uint8 u8ChannelMask);`

Table 3-196. 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.

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

Return: void.

Implements: Pwm_ResetCounter_Activity

Prototype: `void Pwm_ResetCounter(uint8 u8ModuleId);`

Table 3-197. Pwm_ResetCounter Arguments

Type	Name	Direction	Description
uint8	u8ModuleId	**input**	Hardware module

3.8.5.19 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 3-198. Pwm_ResetCounterEnable Arguments

Type	Name	Direction	Description
uint8	u8ModuleId	**Input**	Hardware module.

3.8.5.20 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 3-199. Pwm_ResetCounterDisable Arguments

Type	Name	Direction	Description
uint8	u8ModuleId	**Input**	Hardware module.

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

Return: void.

Implements: Pwm_SetDutyCycle_NoUpdate_Activity

Prototype: `void Pwm_SetDutyCycle_NoUpdate(Pwm_ChannelType ChannelNumber, uint16 u16DutyCycle);`

Table 3-200. 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%.

3.8.5.22 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: AbsoluteDutyCycle = ((uint32)AbsolutePeriodTime * RelativeDutyCycle) >> 15;

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

Return: void.

Implements: Pwm_SetPeriodAndDuty_NoUpdate_Activity

Prototype: `void Pwm_SetPeriodAndDuty_NoUpdate(Pwm_ChannelType ChannelNumber, PeriodType Period, uint16 u16DutyCycle);`

Table 3-201. 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%.

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

Return: void.

Implements: Pwm_SetPhaseShift_Activity

Prototype: `void Pwm_SetPhaseShift(Pwm_ChannelType u8ChannelNumber, Pwm_PeriodType nPeriod, uint16 u16PhaseShift);`

Table 3-202. 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

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

Return: void.

Implements: Pwm_SetPhaseShift_NoUpdate_Activity

Prototype: `void Pwm_SetPhaseShift_NoUpdate(Pwm_ChannelType u8ChannelNumber, Pwm_PeriodType nPeriod, uint16 u16PhaseShift);`

Table 3-203. 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

3.8.5.25 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 3-204. Pwm_SyncUpdate Arguments

Type	Name	Direction	Description
uint8	ModuleId	Input	FTM hardware module ID
uint16	u16SubmoduleMask	Input	PWM submodule mask value

3.8.6 Variables Reference

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

3.9 Symbolic Names Disclaimer

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

#define <Container_Short_Name> <Container_ID>

For this reason it is forbidden to duplicate the name of such containers across the MCAL configuration, or to use names that may trigger other compile issues (e.g. match existing #ifdefs arguments).

Chapter 4
Tresos Configuration Plug-in

This chapter describes the Tresos configuration plug-in for the 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

Table 4-1. Revision table

Revision	Date
1.0.2	2017-10-19

4.2 Form IMPLEMENTATION_CONFIG_VARIANT

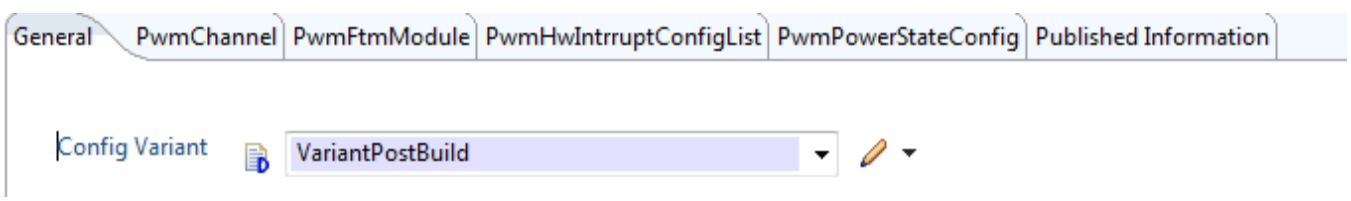


Figure 4-1. Tresos Plugin snapshot for IMPLEMENTATION_CONFIG_VARIANT form.

Table 4-2. Attribute IMPLEMENTATION_CONFIG_VARIANT detailed description

Property	Value
Label	Config Variant
Default	VariantPreCompile
Range	VariantPreCompile VariantPostBuild

4.3 Form PwmConfigurationOfOptApiServices

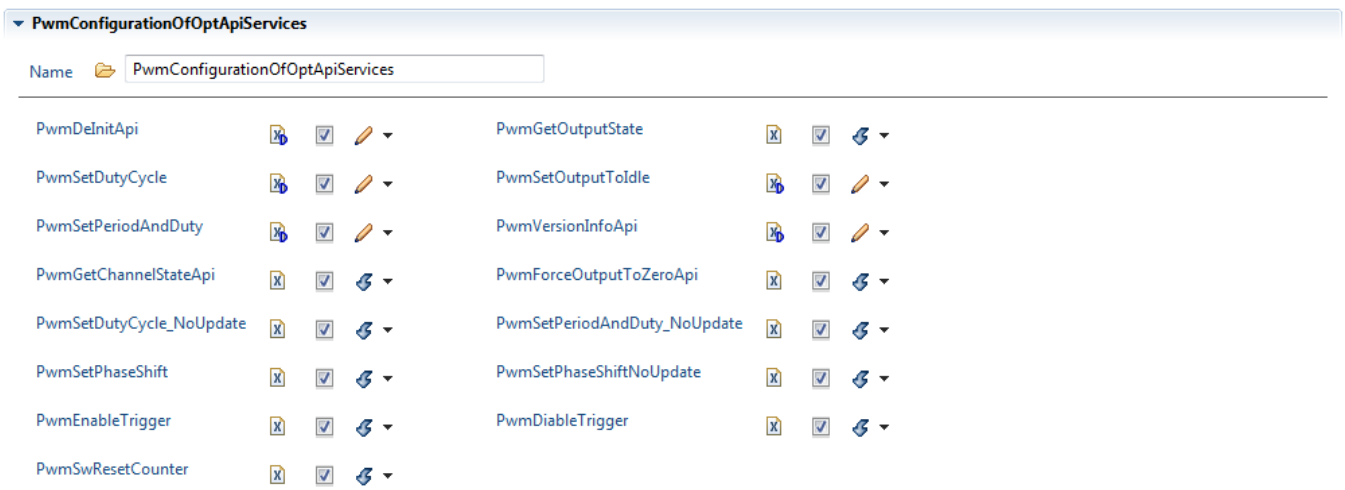


Figure 4-2. Tresos Plugin snapshot for PwmConfigurationOfOptApiServices form.

4.3.1 PwmDeInitApi (PwmConfigurationOfOptApiServices)

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

Table 4-3. 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 4-4. 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 4-5. 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 4-6. Attribute PwmSetOutputToIdle (PwmConfigurationOfOptApiServices) detailed description

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
SCOPE	LOCAL

Table continues on the next page...

Table 4-6. Attribute PwmSetOutputToIdle (PwmConfigurationOfOptApiServices) detailed description (continued)

Property	Value
Symbolic Name	false
Default	true

4.3.5 PwmSetPeriodAndDuty (PwmConfigurationOfOptApiServices)

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

Table 4-7. 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 4-8. 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 4-9. Attribute PwmGetChannelStateApi (PwmConfigurationOfOptApiServices) detailed description

Property	Value
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

4.3.8 PwmForceOutputToZeroApi (PwmConfigurationOfOptApiServices)

Switch to indicate that the Pwm_ForceOutputToZero API is supported

Table 4-10. 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 4-11. 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 4-12. 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 4-13. 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 4-14. Attribute PwmSetPhaseShiftNoUpdate (PwmConfigurationOfOptApiServices) detailed description

Property	Value
Type	BOOLEAN
Origin	Custom

Table continues on the next page...

Table 4-14. Attribute PwmSetPhaseShiftNoUpdate (PwmConfigurationOfOptApiServices) detailed description (continued)

Property	Value
Symbolic Name	false
Default	false

4.3.13 PwmEnableTrigger (PwmConfigurationOfOptApiServices)

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

Table 4-15. 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 4-16. 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 4-17. Attribute PwmSwResetCounter (PwmConfigurationOfOptApiServices) detailed description

Property	Value
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

4.4 Form PwmGeneral

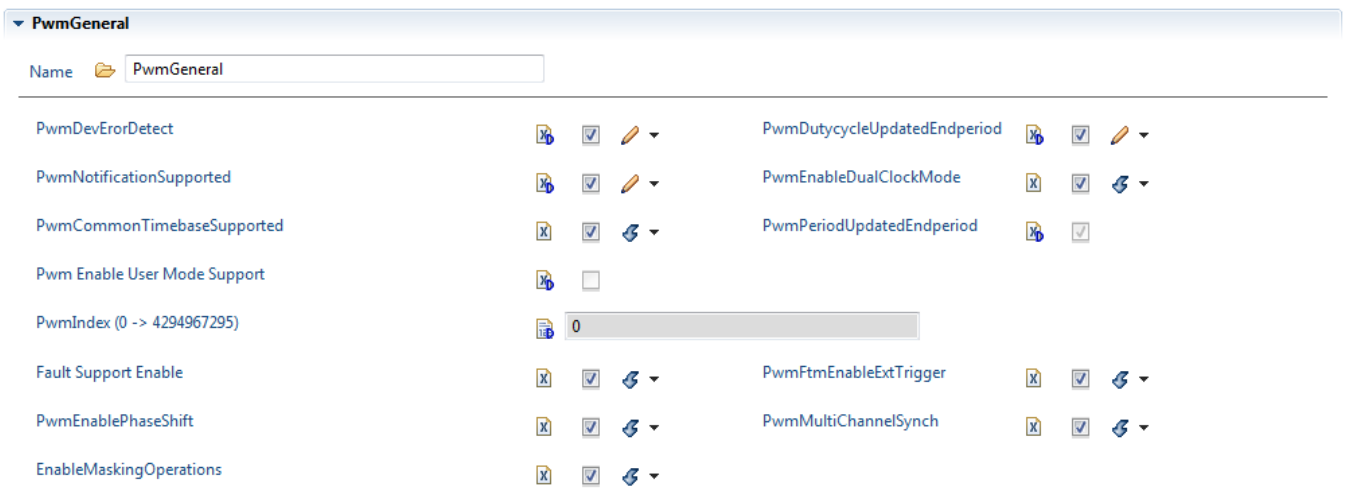


Figure 4-3. Tresos Plugin snapshot for PwmGeneral form.

4.4.1 PwmDevErrorDetect (PwmGeneral)

Switch to enable/disable the development error detection.

Table 4-18. 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 4-19. 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 4-20. 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 4-21. Attribute PwmEnableDualClockMode (PwmGeneral) detailed description

Property	Value
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	true

4.4.5 PwmCommonTimebaseSupported (PwmGeneral)

Enable/Disable the option to set the a common timebase for multiple modules.

LIMITATION

This is an Implementation Specific Parameter. Enabling this feature will allow the use of the Pwm_SelectCommonTimebase API

Table 4-22. Attribute PwmCommonTimebaseSupported (PwmGeneral) detailed description

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	false

4.4.6 PwmPeriodUpdatedEndperiod (PwmGeneral)

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

LIMITATION

Curent implementation supports the update of the period parameter only at the end of the current period.

Table 4-23. Attribute PwmPeriodUpdatedEndperiod (PwmGeneral) detailed description

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	true
READONLY	true

4.4.7 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 4-24. Attribute PwmIndex (PwmGeneral) detailed description

Property	Value
Type	INTEGER
Origin	AUTOSAR_ECUC
SCOPE	LOCAL
Symbolic Name	false
Default	0
Invalid	Range <=4294967295 >=0

4.4.8 PwmFaultSupport (PwmGeneral)

This enables the fault functionality.

Table 4-25. Attribute PwmFaultSupport (PwmGeneral) detailed description

Property	Value
Label	Fault Support Enable
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

4.4.9 PwmFtmEnableExtTrigger (PwmGeneral)

This enables external trigger generation.

Table 4-26. Attribute PwmFtmEnableExtTrigger (PwmGeneral) detailed description

Property	Value
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

4.4.10 PwmEnablePhaseShift (PwmGeneral)

This enables Phase Shift feature.

Table 4-27. Attribute PwmEnablePhaseShift (PwmGeneral) detailed description

Property	Value
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

4.4.11 PwmMultiChannelSynch (PwmGeneral)

Enable/Disable the option to set the duty cycle synchronous for several channels

Table 4-28. Attribute PwmEnablePhaseShift (PwmGeneral) detailed description

Property	Value
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

4.4.12 EnableMaskingOperations (PwmGeneral)

Add/remove the service Pwm_MaskOutputs and Pwm_UnMaskOutputs from the code.

Table 4-29. Attribute EnableMaskingOperations (PwmGeneral) detailed description

Property	Value
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

4.5 Form CommonPublishedInformation

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

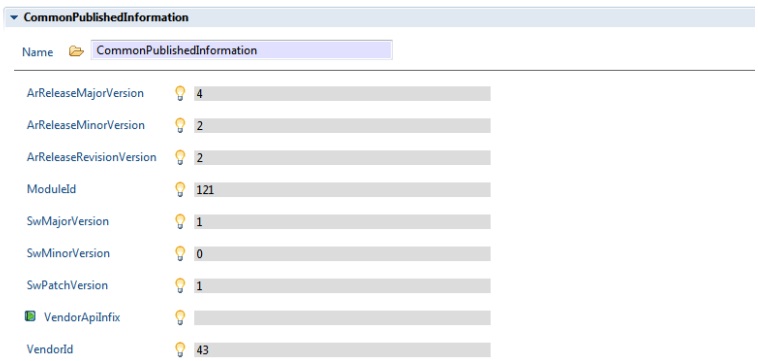


Figure 4-4. Tresos Plugin snapshot for CommonPublishedInformation form.

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
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
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

Table continues on the next page...

Table 4-33. Attribute ModuleId (CommonPublishedInformation) detailed description (continued)

Property	Value
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 >=0 <=0

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	1
Invalid	Range >=1 <=1

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 PwmFtmModule

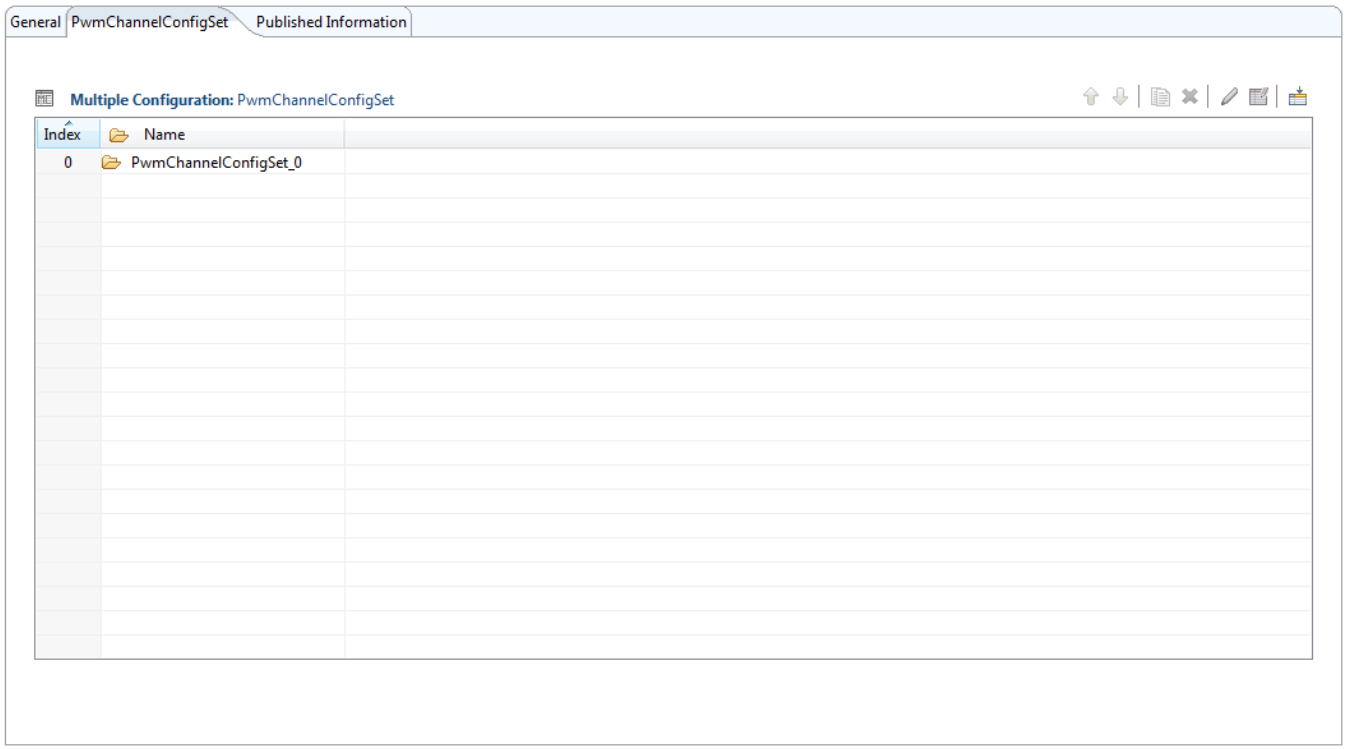
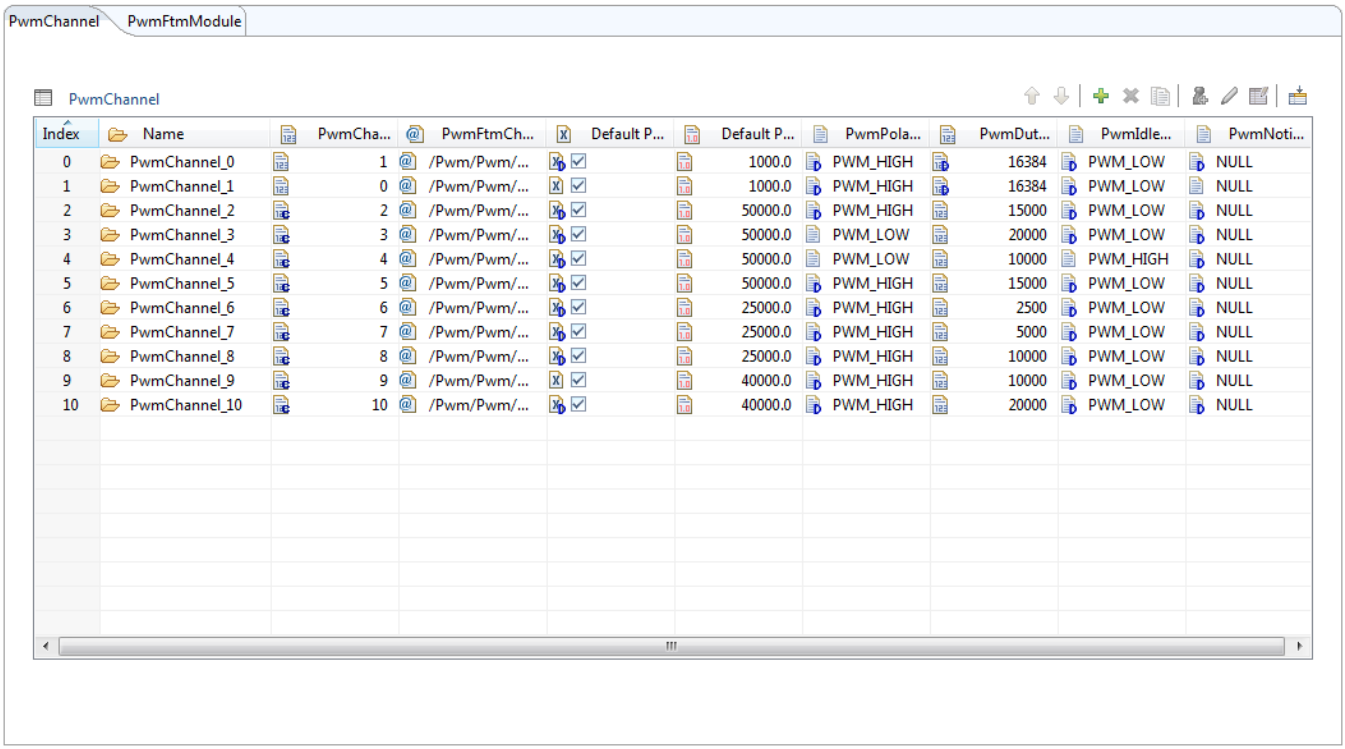


Figure 4-5. Tresos Plugin snapshot for PwmChannelConfigSet form.

4.6.1 Form PwmChannel

Configuration of an individual PWM channel.

Is included by form: Form PwmChannelConfigSet



Index	Name	PwmCha...	PwmFtmCh...	Default P...	Default P...	PwmPola...	PwmDut...	PwmIdle...	PwmNoti...
0	PwmChannel_0	1	/Pwm/Pwm/...		1000.0	PWM_HIGH	16384	PWM_LOW	NULL
1	PwmChannel_1	0	/Pwm/Pwm/...		1000.0	PWM_HIGH	16384	PWM_LOW	NULL
2	PwmChannel_2	2	/Pwm/Pwm/...		50000.0	PWM_HIGH	15000	PWM_LOW	NULL
3	PwmChannel_3	3	/Pwm/Pwm/...		50000.0	PWM_LOW	20000	PWM_LOW	NULL
4	PwmChannel_4	4	/Pwm/Pwm/...		50000.0	PWM_LOW	10000	PWM_HIGH	NULL
5	PwmChannel_5	5	/Pwm/Pwm/...		50000.0	PWM_HIGH	15000	PWM_LOW	NULL
6	PwmChannel_6	6	/Pwm/Pwm/...		25000.0	PWM_HIGH	2500	PWM_LOW	NULL
7	PwmChannel_7	7	/Pwm/Pwm/...		25000.0	PWM_HIGH	5000	PWM_LOW	NULL
8	PwmChannel_8	8	/Pwm/Pwm/...		25000.0	PWM_HIGH	10000	PWM_LOW	NULL
9	PwmChannel_9	9	/Pwm/Pwm/...		40000.0	PWM_HIGH	10000	PWM_LOW	NULL
10	PwmChannel_10	10	/Pwm/Pwm/...		40000.0	PWM_HIGH	20000	PWM_LOW	NULL

Figure 4-6. Tresos Plugin snapshot for PwmChannel form.

Below is a detailed description of the configurable parameters of each PWM channel.

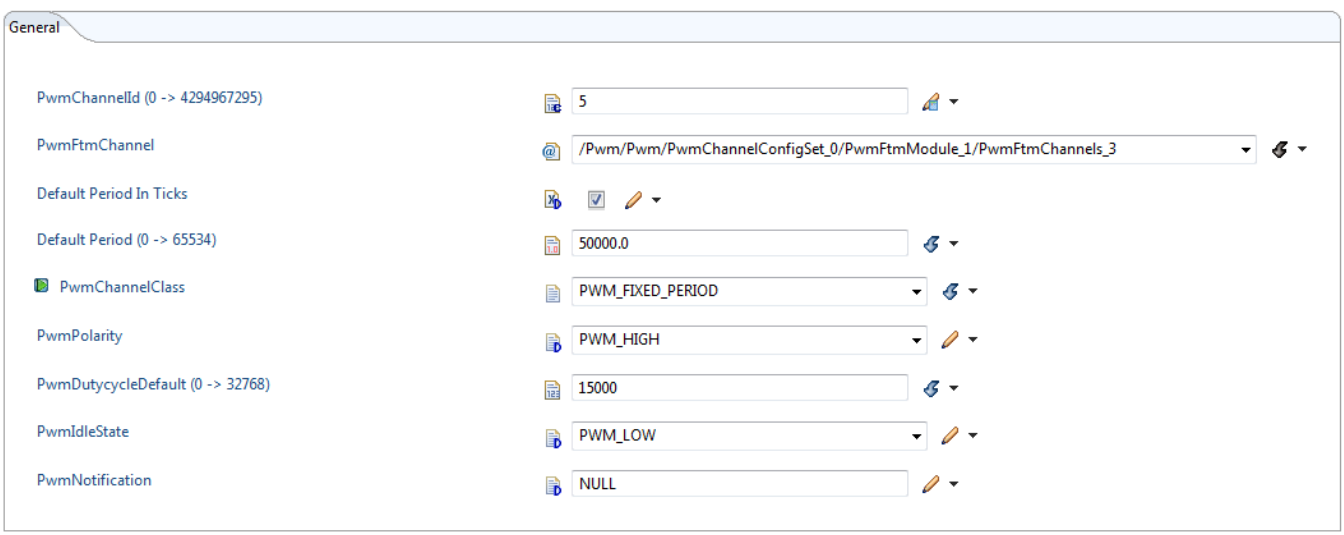


Figure 4-7. Tresos Plugin snapshot for PwmChannel form with parameter details.

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 PwmPeriodInTicks (PwmChannel)

By default period unit is measured in seconds. Enable this check to set default period unit in ticks.

Table 4-40. Attribute PwmPeriodInTicks (PwmChannel) detailed description

Property	Value
Label	Default Period In Ticks
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	true

4.6.1.3 PwmPeriodDefault (PwmChannel)

Default period value used at initialization. The measure unit are in ticks. Valid values [0, 0xFFFE = 65534]

Table 4-41. 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 <=65534 >=0

4.6.1.4 PwmChannelClass (PwmChannel)

Class of PWM Channel.

Table 4-42. 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	PWM_FIXED_PERIOD PWM_VARIABLE_PERIOD PWM_FIXED_PERIOD_SHIFTED

4.6.1.5 PwmPolarity (PwmChannel)

Define the starting polarity of each PWM channel.

Table 4-43. 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

4.6.1.6 PwmDutycycleDefault (PwmChannel)

Default value for duty cycle used for Initialization 0, represents 0% 0x4000, represents 50% 0x8000, represents 100%.

Table 4-44. 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

4.6.1.7 PwmIdleState (PwmChannel)

This parameter represents the state of the Pin when the output is set to Idle.

Table 4-45. 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

4.6.1.8 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 4-46. Attribute PwmNotification (PwmChannel) detailed description

Property	Value
Type	FUNCTION-NAME
Origin	AUTOSAR_ECUC
SCOPE	LOCAL
POSTBUILDVARIANTVALUE	true
OPTIONAL	true
Symbolic Name	false
Default	NULL

4.6.1.9 PwmFtmChannel (PwmChannel)

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

Table 4-47. Attribute PwmFtmChannel (PwmChannel) detailed description

Property	Value
Type	REFERENCE
Origin	Custom
POSTBUILDVARIANTVALUE	true

4.6.2 Form PwmFtmModule

Configuration of a FTM module available on the platform.

Is included by form : Form PwmChannelConfigSet

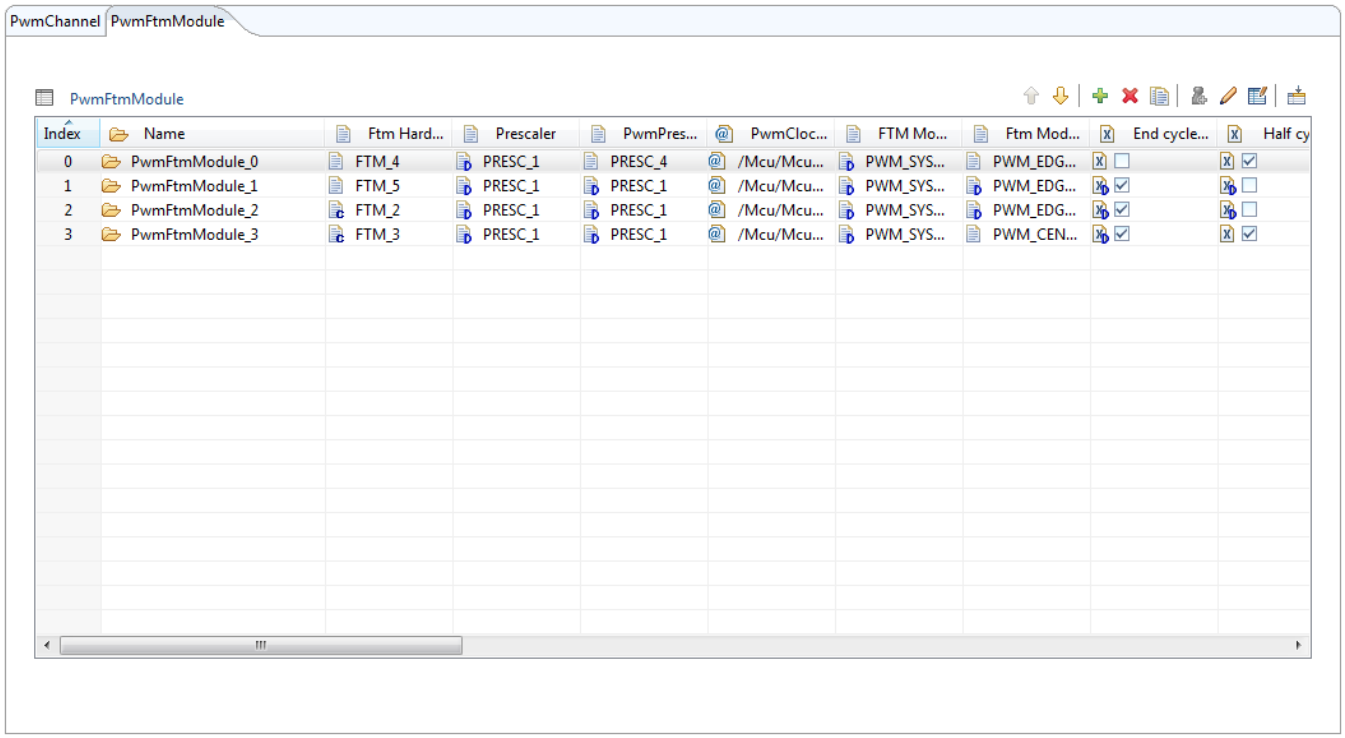


Figure 4-8. Tresos Plugin snapshot for PwmFtmModule form.

Below is a detailed description of the configurable parameters of each FTM module.

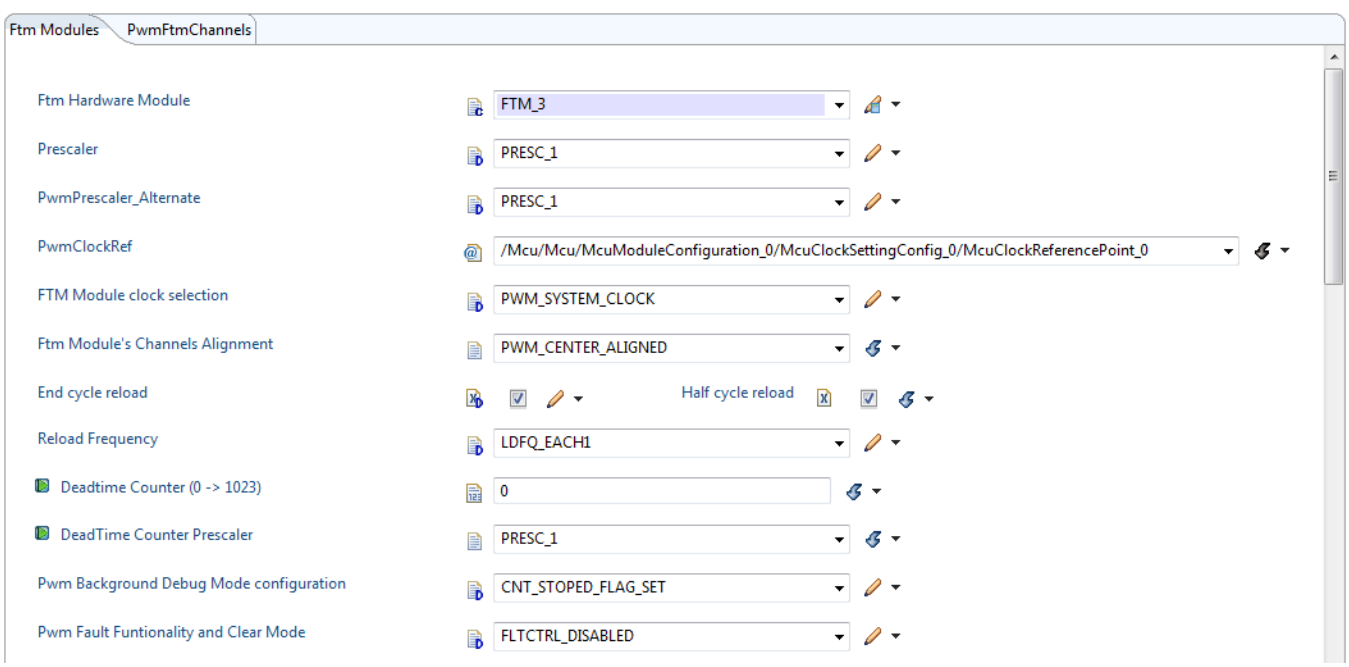


Figure 4-9. Tresos Plugin snapshot for PwmFtmModule form with parameter details.

4.6.2.1 FtmModule (PwmFtmModule)

Select one of the 8 sub-modules available on the current FTM module.

Table 4-48. Attribute FtmModule (PwmFtmModule) detailed description

Property	Value
Label	Ftm Hardware Module
Type	ENUMERATION
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false

4.6.2.2 PwmPrescaler (PwmFtmModule)

PwmPrescaler. AUTOSAR Requirements: not specified.

Table 4-49. 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

4.6.2.3 PwmPrescaler_Alternate (PwmFtmModule)

PwmPrescaler_Alternate. AUTOSAR Requirements: not specified.

Table 4-50. 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

4.6.2.4 PwmClockRef (PwmFtmModule)

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

Table 4-51. Attribute PwmClockRef (PwmFtmModule) detailed description

Property	Value
Type	REFERENCE
Origin	Custom

4.6.2.5 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 4-52. 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

4.6.2.6 PwmChanEdgeAlignment (PwmFtmModule)

This list will change the PWM generation mode for this FTM sub-module. This in a Non-AUTOSAR feature.

Table 4-53. 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

4.6.2.7 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 4-54. Attribute PwmUpdatedEndPeriod (PwmFtmModule) detailed description

Property	Value
Type	BOOLEAN

Table continues on the next page...

Table 4-54. Attribute PwmUpdatedEndPeriod (PwmFtmModule) detailed description (continued)

Property	Value
Origin	AUTOSAR_ECUC
SCOPE	LOCAL
Symbolic Name	false
Default	true

4.6.2.8 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 4-55. Attribute PwmUpdatedMiddlePeriod (PwmFtmModule) detailed description

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
SCOPE	LOCAL
Symbolic Name	false
Default	false

4.6.2.9 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 4-56. Attribute PwmReloadFrequency (PwmFtmModule) detailed description

Property	Value
Label	Prescaler
Type	ENUMERATION
Origin	Custom

Table continues on the next page...

Table 4-56. Attribute PwmReloadFrequency (PwmFtmModule) detailed description (continued)

Property	Value
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 LDFQ_EACH21 LDFQ_EACH22 LDFQ_EACH23 LDFQ_EACH24 LDFQ_EACH25 LDFQ_EACH26 LDFQ_EACH27 LDFQ_EACH28 LDFQ_EACH29 LDFQ_EACH30 LDFQ_EACH31 LDFQ_EACH32

4.6.2.10 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 4-57. 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

4.6.2.11 DeadTimePrescaler (PwmFtmModule)

DeadTime counter prescaler.

Table 4-58. 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

4.6.2.12 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 4-59. 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

4.6.2.13 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 4-60. Attribute PwmFtmFaultFunctionality (PwmFtmModule) detailed description

Property	Value
Label	Pwm Fault Funtionality and Clear Mode
Type	ENUMERATION
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false

Table continues on the next page...

Table 4-60. Attribute PwmFtmFaultFunctionality (PwmFtmModule) detailed description (continued)

Property	Value
Default	FLTCTRL_DISABLED
Range	FLTCTRL_DISABLED AUTOMATIC_FOR_EVEN_CHANNELS AUTOMATIC_FOR_ALL_CHANNELS

4.6.2.14 Form PwmFtmChannels

Configuration of an individual PWM FTM channel.

Is included by form: Form PwmFtmModule

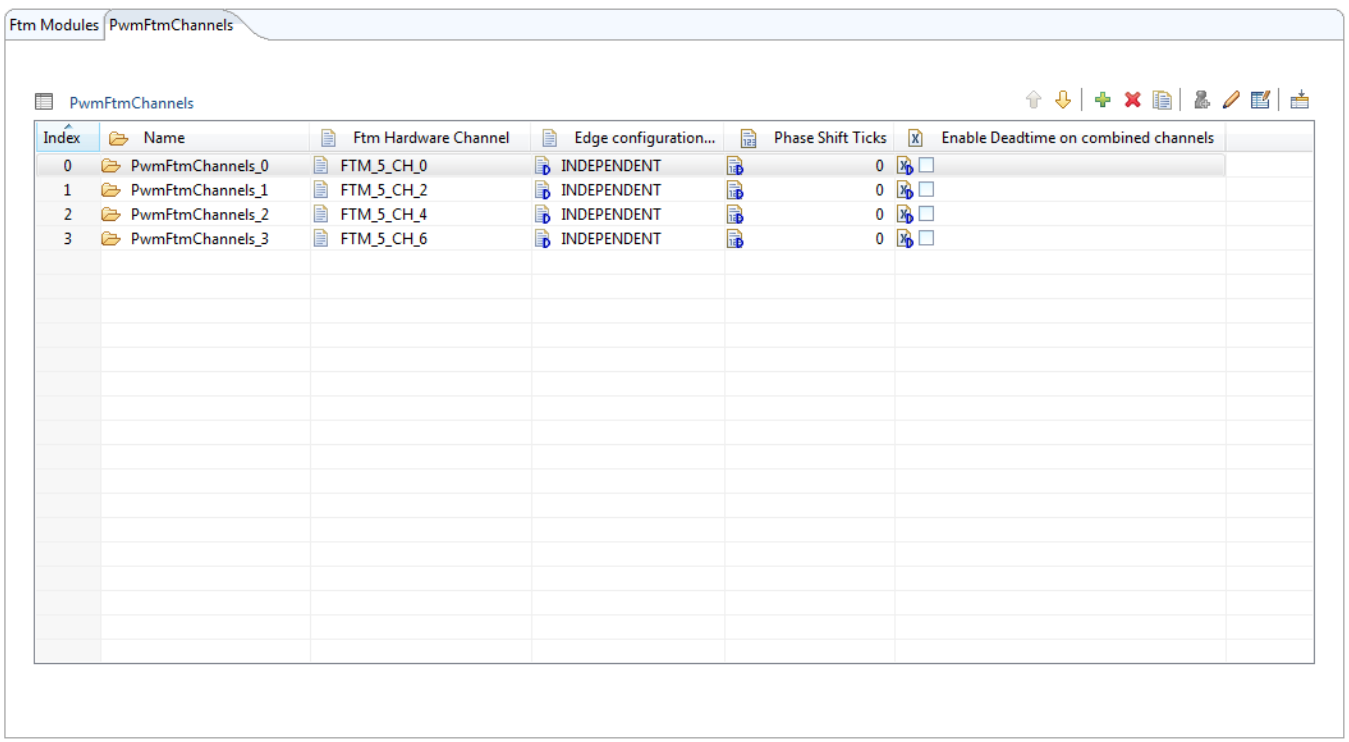


Figure 4-10. Tresos Plugin snapshot for PwmFtmChannels form.

Below is a detailed description of the configurable parameters of each FTM channel.

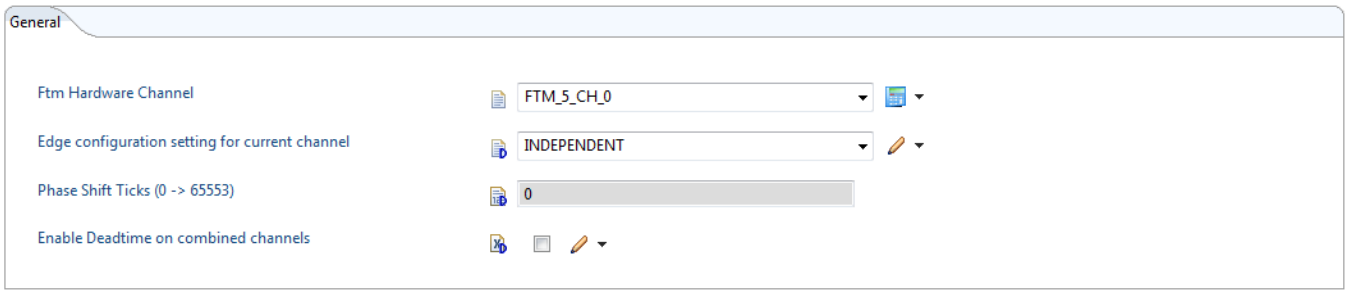


Figure 4-11. Tresos Plugin snapshot for PwmFtmChannels form with parameter details.

4.6.2.14.1 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 4-61. Attribute PwmFtmChannelId (PwmFtmChannels) detailed description

Property	Value
Label	Ftm Hardware Channel
Type	ENUMERATION
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false

4.6.2.14.2 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 4-62. Attribute ChannelEdgeSetup (PwmFtmChannels) detailed description

Property	Value
Label	Edge configuration setting for current channel
Type	ENUMERATION
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	INDEPENDENT
Range	INDEPENDENT PHASE_SHIFTED_SINGLE COMBINED_SYNCED COMBINED_COMPLEMENTARY PHASE_SHIFTED_SYNCED PHASE_SHIFTED_COMPLEMENTARY

4.6.2.14.3 ChannelCombDeadTimeEn (PwmFtmChannels)

Switch to enable/disable Deadtime for combine channels.

Table 4-63. Attribute ChannelCombDeadTimeEn (PwmFtmChannels) detailed description

Property	Value
Label	Enable Deadtime on combined channels
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

4.6.2.14.4 PwmPhaseShift (PwmFtmChannels)

Phase shift value (in tick) for channel in PHASE_SHIFTED_SINGLE/ PHASE_SHIFTED_SYNCED/PHASE_SHIFTED_COMPLEMENTARY

Table 4-64. 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

4.6.2.15 Form PwmFtmChannelFaultSettings

Global Fault configuration. AUTOSAR Requirements: not specified.

Is included by form: Form PwmFtmModule

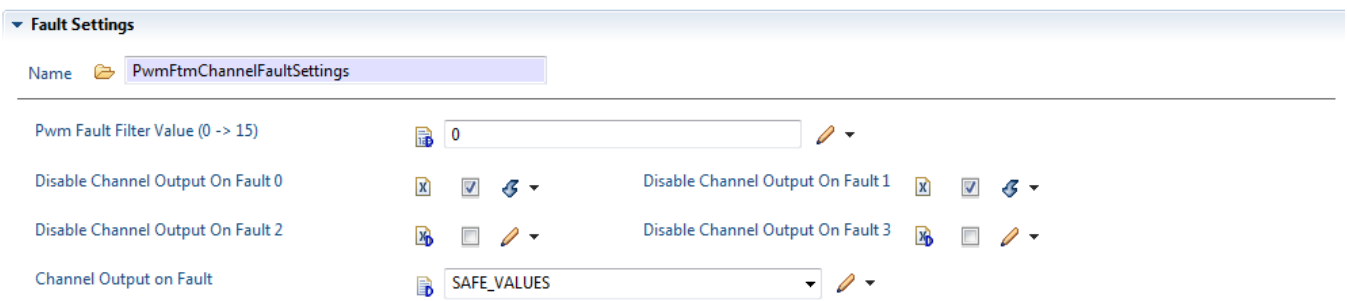


Figure 4-12. Tresos Plugin snapshot for PwmFtmChannelFaultSettings form.

4.6.2.15.1 PwmFilterValue (PwmFtmChannelFaultSettings)

Value used for debouncing Fault inputs.

Table 4-65. Attribute PwmFilterValue (PwmFtmChannelFaultSettings) detailed description

Property	Value
Label	Pwm Fault Filter Value
Type	INTEGER
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	0

Table continues on the next page...

Table 4-65. Attribute PwmFilterValue (PwmFtmChannelFaultSettings) detailed description (continued)

Property	Value
Invalid	Range <=15 >=0

4.6.2.15.2 PwmDisableOutputOnFault0 (PwmFtmChannelFaultSettings)

Disable the PWM output on detection of fault on fault channel 0. AUTOSAR Requirements: not specified.

Table 4-66. 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

4.6.2.15.3 PwmDisableOutputOnFault1 (PwmFtmChannelFaultSettings)

Disable the PWM output on detection of fault on fault channel 1. AUTOSAR Requirements: not specified.

Table 4-67. 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

4.6.2.15.4 PwmDisableOutputOnFault2 (PwmFtmChannelFaultSettings)

Disable the PWM output on detection of fault on fault channel 2. AUTOSAR Requirements: not specified.

Table 4-68. 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

4.6.2.15.5 PwmDisableOutputOnFault3 (PwmFtmChannelFaultSettings)

Disable the PWM output on detection of fault on fault channel 3. AUTOSAR Requirements: not specified.

Table 4-69. 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

4.6.2.15.6 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 4-70. 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

4.6.2.15.7 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.

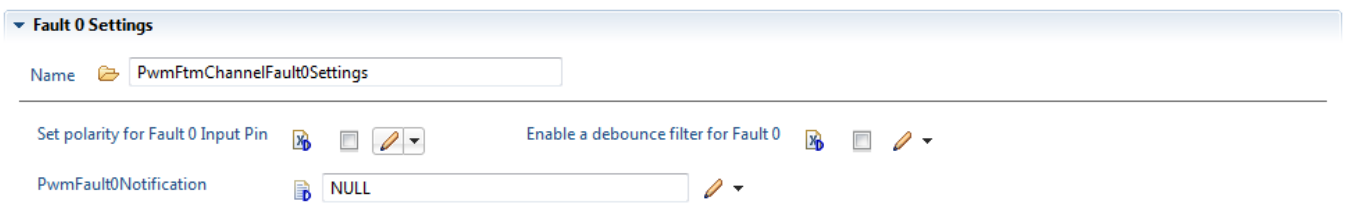


Figure 4-13. Tresos Plugin snapshot for PwmFtmChannelFault0Settings form.

4.6.2.15.7.1 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 4-71. Attribute PwmFault0Polarity (PwmFtmChannelFault0Settings) detailed description

Property	Value
Label	Set polarity for Fault 0 Input Pin
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false

Table continues on the next page...

Table 4-71. Attribute PwmFault0Polarity (PwmFtmChannelFault0Settings) detailed description (continued)

Property	Value
Default	false

4.6.2.15.7.2 PwmEnableFault0Filter (PwmFtmChannelFault0Settings)

Enable a debounce filter for Fault 0 Input pin. AUTOSAR Requirements: not specified.

Table 4-72. 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

4.6.2.15.7.3 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 4-73. Attribute PwmFault0Notification (PwmFtmChannelFault0Settings) detailed description

Property	Value
Type	FUNCTION-NAME
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	NULL

4.6.2.15.8 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.

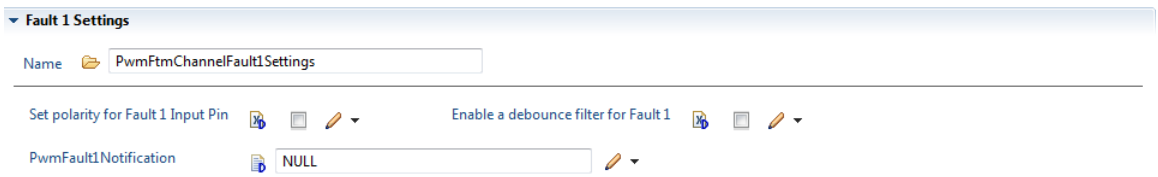


Figure 4-14. Tresos Plugin snapshot for PwmFtmChannelFault1Settings form.

4.6.2.15.8.1 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 4-74. 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

4.6.2.15.8.2 PwmEnableFault1Filter (PwmFtmChannelFault1Settings)

Enable a debounce filter for Fault 1 Input pin. AUTOSAR Requirements: not specified.

Table 4-75. 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

4.6.2.15.8.3 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 4-76. Attribute PwmFault1Notification (PwmFtmChannelFault1Settings) detailed description

Property	Value
Type	FUNCTION-NAME
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	NULL

4.6.2.15.9 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.

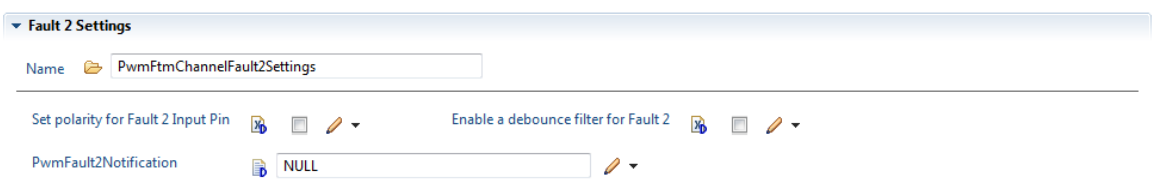


Figure 4-15. Tresos Plugin snapshot for PwmFtmChannelFault2Settings form.

4.6.2.15.9.1 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 4-77. 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

4.6.2.15.9.2 PwmEnableFault2Filter (PwmFtmChannelFault2Settings)

Enable a debounce filter for Fault 2 Input pin. AUTOSAR Requirements: not specified.

Table 4-78. 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

4.6.2.15.9.3 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 4-79. Attribute PwmFault2Notification (PwmFtmChannelFault2Settings) detailed description

Property	Value
Type	FUNCTION-NAME
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false

Table continues on the next page...

Table 4-79. Attribute PwmFault2Notification (PwmFtmChannelFault2Settings) detailed description (continued)

Property	Value
Default	NULL

4.6.2.15.10 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.

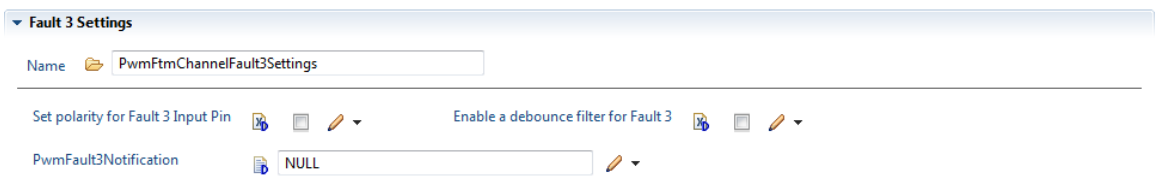


Figure 4-16. Tresos Plugin snapshot for PwmFtmChannelFault3Settings form.

4.6.2.15.10.1 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 4-80. 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

4.6.2.15.10.2 PwmEnableFault3Filter (PwmFtmChannelFault3Settings)

Enable a debounce filter for Fault 3 Input pin. AUTOSAR Requirements: not specified.

Table 4-81. 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

4.6.2.15.10.3 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 4-82. Attribute PwmFault3Notification (PwmFtmChannelFault3Settings) detailed description

Property	Value
Type	FUNCTION-NAME
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	NULL

4.6.2.16 Form PwmExternalTriggerSettings

Global Trigger configuration.

Is included by form: Form PwmFtmModule

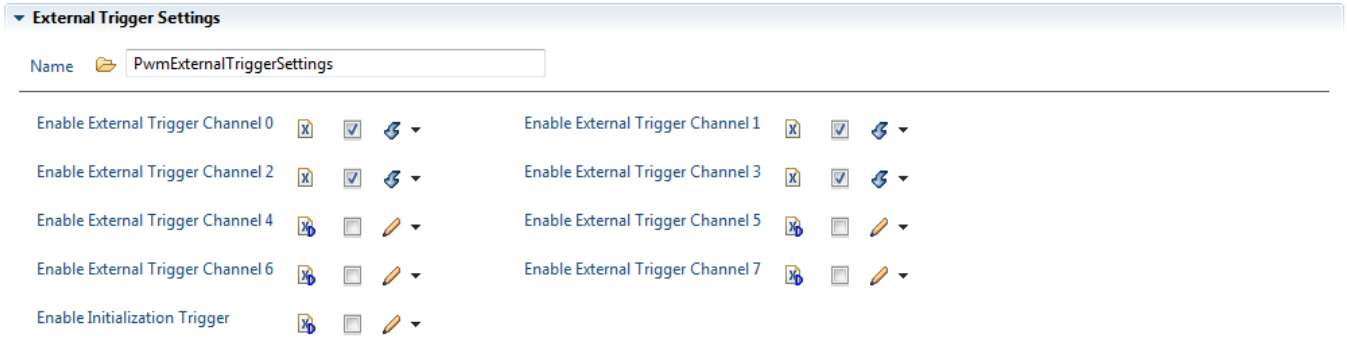


Figure 4-17. Tresos Plugin snapshot for PwmExternalTriggerSettings form.

4.6.2.16.1 PwmEnableExternalTriggerChannel0 (PwmExternalTriggerSettings)

Enable the generation of the channel trigger on Trigger channel 0.

Table 4-83. Attribute PwmEnableExternalTriggerChannel0 (PwmExternalTriggerSettings) detailed description

Property	Value
Label	Enable External Trigger Channel 0
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

4.6.2.16.2 PwmEnableExternalTriggerChannel1 (PwmExternalTriggerSettings)

Enable the generation of the channel trigger on Trigger channel 1.

Table 4-84. Attribute PwmEnableExternalTriggerChannel1 (PwmExternalTriggerSettings) detailed description

Property	Value
Label	Enable External Trigger Channel 1
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

4.6.2.16.3 PwmEnableExternalTriggerChannel2 (PwmExternalTriggerSettings)

Enable the generation of the channel trigger on Trigger channel 2.

Table 4-85. Attribute PwmEnableExternalTriggerChannel2 (PwmExternalTriggerSettings) detailed description

Property	Value
Label	Enable External Trigger Channel 2
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

4.6.2.16.4 PwmEnableExternalTriggerChannel3 (PwmExternalTriggerSettings)

Enable the generation of the channel trigger on Trigger channel 3.

Table 4-86. Attribute PwmEnableExternalTriggerChannel3 (PwmExternalTriggerSettings) detailed description

Property	Value
Label	Enable External Trigger Channel 3
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

4.6.2.16.5 PwmEnableExternalTriggerChannel4 (PwmExternalTriggerSettings)

Enable the generation of the channel trigger on Trigger channel 4.

Table 4-87. Attribute PwmEnableExternalTriggerChannel4 (PwmExternalTriggerSettings) detailed description

Property	Value
Label	Enable External Trigger Channel 4

Table continues on the next page...

Table 4-87. Attribute PwmEnableExternalTriggerChannel4 (PwmExternalTriggerSettings) detailed description (continued)

Property	Value
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

4.6.2.16.6 PwmEnableExternalTriggerChannel5 (PwmExternalTriggerSettings)

Enable the generation of the channel trigger on Trigger channel 5.

Table 4-88. Attribute PwmEnableExternalTriggerChannel5 (PwmExternalTriggerSettings) detailed description

Property	Value
Label	Enable External Trigger Channel 5
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

4.6.2.16.7 PwmEnableExternalTriggerChannel6 (PwmExternalTriggerSettings)

Enable the generation of the channel trigger on Trigger channel 6.

Table 4-89. Attribute PwmEnableExternalTriggerChannel6 (PwmExternalTriggerSettings) detailed description

Property	Value
Label	Enable External Trigger Channel 6
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

4.6.2.16.8 PwmEnableExternalTriggerChannel7 (PwmExternalTriggerSettings)

Enable the generation of the channel trigger on Trigger channel 7.

Table 4-90. Attribute PwmEnableExternalTriggerChannel7 (PwmExternalTriggerSettings) detailed description

Property	Value
Label	Enable External Trigger Channel 7
Type	BOOLEAN
Origin	Custom
POSTBUILDVARIANTVALUE	true
Symbolic Name	false
Default	false

4.6.2.16.9 PwmEnableInitializationTrigger (PwmExternalTriggerSettings)

Enable the generation of the trigger when the FTM counter is equal to the CNTIN register.

Table 4-91. 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

Web Support:
nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer’s technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

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. Arm, AMBA, Artisan, Cortex, Jazelle, Keil, SecurCore, Thumb, TrustZone, and µVision are registered trademarks of Arm Limited (or its subsidiaries) in the EU and/or elsewhere. Arm7, Arm9, Arm11, big.LITTLE, CoreLink, CoreSight, DesignStart, Mali, Mbed, NEON, POP, Sensinode, Socrates, ULINK and Versatile are trademarks of Arm Limited (or its subsidiaries) in the EU and/or elsewhere. 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.

© 2018 NXP B.V.

Document Number UM2PWMASR4.2 Rev0002 R1.0.1
Revision 1.0