User Manual

for MPC574XG PWM Driver

Document Number: UM35PWMASR4.2 Rev0002 R1.0.0
Rev. 5.0.0

Contents

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

Chapter 1
Revision History

Chapter 2
Introduction

2.1 Supported Derivatives....17

2.2 Overview....18

2.3 About this Manual....18

2.4 Acronyms and Definitions....19

2.5 Reference List....19

Chapter 3
Driver

3.1 Requirements....21

3.2 Driver Design Summary....21

3.3 Deviation from Requirements....21

3.4 Runtime Errors....23

3.5 eMIOS specific implementation details....23

3.6 Software specification....28

3.6.1 Define Reference....29

3.6.1.1 Define PWM_VENDOR_ID_C....29

3.6.1.2 Define PWM_MODULE_ID_C....29

3.6.1.3 Define PWM_AR_RELEASE_MAJOR_VERSION_C....29

3.6.1.4 Define PWM_AR_RELEASE_MINOR_VERSION_C....29

3.6.1.5 Define PWM_AR_RELEASE_REVISION_VERSION_C....30

3.6.1.6 Define PWM_SW_MAJOR_VERSION_C....30

3.6.1.7 Define PWM_SW_MINOR_VERSION_C....30

3.6.1.8 Define PWM_SW_PATCH_VERSION_C....30

3.6.1.9 Define PWM_VENDOR_ID....30

3.6.1.10 Define PWM_MODULE_ID....31

3.6.1.11 Define PWM_AR_RELEASE_MAJOR_VERSION.... 31
3.6.1.12 Define PWM_AR_RELEASE_MINOR_VERSION.... 31
3.6.1.13 Define PWM_AR_RELEASE_REVISION_VERSION.... 31
3.6.1.14 Define PWM_SW_MAJOR_VERSION.... 31
3.6.1.15 Define PWM_SW_MINOR_VERSION.... 32
3.6.1.16 Define PWM_SW_PATCH_VERSION.... 32
3.6.1.17 Define PWM_PC_CFG_VENDOR_ID_C.... 32
3.6.1.18 Define PWM_PC_CFG_MODULE_ID_C.... 32
3.6.1.19 Define PWM_PC_CFG_AR_RELEASE_MAJOR_VERSION_C.... 32
3.6.1.20 Define PWM_PC_CFG_AR_RELEASE_MINOR_VERSION_C.... 33
3.6.1.21 Define PWM_PC_CFG_AR_RELEASE_REVISION_VERSION_C.... 33
3.6.1.22 Define PWM_PC_CFG_SW_MAJOR_VERSION_C.... 33
3.6.1.23 Define PWM_PC_CFG_SW_MINOR_VERSION_C.... 33
3.6.1.24 Define PWM_PC_CFG_SW_PATCH_VERSION_C.... 33
3.6.1.25 Define PWM_CFG_VENDOR_ID.... 34
3.6.1.26 Define PWM_CFG_MODULE_ID.... 34
3.6.1.27 Define PWM_CFG_AR_RELEASE_MAJOR_VERSION.... 34
3.6.1.28 Define PWM_CFG_AR_RELEASE_MINOR_VERSION.... 34
3.6.1.29 Define PWM_CFG_AR_RELEASE_REVISION_VERSION.... 34
3.6.1.30 Define PWM_CFG_SW_MAJOR_VERSION.... 35
3.6.1.31 Define PWM_CFG_SW_MINOR_VERSION.... 35
3.6.1.32 Define PWM_CFG_SW_PATCH_VERSION.... 35
3.6.1.33 Define PWM_CFG_ENV_VENDOR_ID.... 35
3.6.1.34 Define PWM_CFG_ENV_MODULE_ID.... 36
3.6.1.35 Define PWM_CFG_ENV_AR_RELEASE_MAJOR_VERSION.... 36
3.6.1.36 Define PWM_CFG_ENV_AR_RELEASE_MINOR_VERSION.... 36
3.6.1.37 Define PWM_CFG_ENV_AR_RELEASE_REVISION_VERSION.... 36
3.6.1.38 Define PWM_CFG_ENV_SW_MAJOR_VERSION.... 36
3.6.1.39 Define PWM_CFG_ENV_SW_MINOR_VERSION.... 37

3.6.1.40 Define PWM_CFG_ENV_SW_PATCH_VERSION....37
3.6.1.41 Define PWM_EMIOS_TYPES_VENDOR_ID....37
3.6.1.42 Define PWM_EMIOS_TYPES_MODULE_ID....37
3.6.1.43 Define PWM_EMIOS_TYPES_AR_RELEASE_MAJOR_VERSION....37
3.6.1.44 Define PWM_EMIOS_TYPES_AR_RELEASE_MINOR_VERSION....38
3.6.1.45 Define PWM_EMIOS_TYPES_AR_RELEASE_REVISION_VERSION....38
3.6.1.46 Define PWM_EMIOS_TYPES_SW_MAJOR_VERSION....38
3.6.1.47 Define PWM_EMIOS_TYPES_SW_MINOR_VERSION....38
3.6.1.48 Define PWM_EMIOS_TYPES_SW_PATCH_VERSION....39
3.6.1.49 Define PWM_IPW_NOTIF_VENDOR_ID....39
3.6.1.50 Define PWM_IPW_NOTIF_MODULE_ID....39
3.6.1.51 Define PWM_IPW_NOTIF_AR_RELEASE_MAJOR_VERSION....39
3.6.1.52 Define PWM_IPW_NOTIF_AR_RELEASE_MINOR_VERSION....40
3.6.1.53 Define PWM_IPW_NOTIF_AR_RELEASE_REVISION_VERSION....40
3.6.1.54 Define PWM_IPW_NOTIF_SW_MAJOR_VERSION....40
3.6.1.55 Define PWM_IPW_NOTIF_SW_MINOR_VERSION....40
3.6.1.56 Define PWM_IPW_NOTIF_SW_PATCH_VERSION....40
3.6.1.57 Define PWM_IPW_TYPES_VENDOR_ID....41
3.6.1.58 Define PWM_IPW_TYPES_MODULE_ID....41
3.6.1.59 Define PWM_IPW_TYPES_AR_RELEASE_MAJOR_VERSION....41
3.6.1.60 Define PWM_IPW_TYPES_AR_RELEASE_MINOR_VERSION....41
3.6.1.61 Define PWM_IPW_TYPES_AR_RELEASE_REVISION_VERSION....42
3.6.1.62 Define PWM_IPW_TYPES_SW_MAJOR_VERSION....42
3.6.1.63 Define PWM_IPW_TYPES_SW_MINOR_VERSION....42
3.6.1.64 Define PWM_IPW_TYPES_SW_PATCH_VERSION....42
3.6.1.65 Define PWM_NOTIF_VENDOR_ID....42
3.6.1.66 Define PWM_NOTIF_MODULE_ID....43
3.6.1.67 Define PWM_NOTIF_AR_RELEASE_MAJOR_VERSION....43
3.6.1.68 Define PWM_NOTIF_AR_RELEASE_MINOR_VERSION....43

3.6.1.69 Define PWM_NOTIF_AR_RELEASE_REVISION_VERSION.... 43
3.6.1.70 Define PWM_NOTIF_SW_MAJOR_VERSION....43
3.6.1.71 Define PWM_NOTIF_SW_MINOR_VERSION....44
3.6.1.72 Define PWM_NOTIF_SW_PATCH_VERSION....44
3.6.1.73 Define PWM_PB_CFG_VENDOR_ID_C....44
3.6.1.74 Define PWM_PB_CFG_MODULE_ID_C....44
3.6.1.75 Define PWM_PB_CFG_AR_RELEASE_MAJOR_VERSION_C....44
3.6.1.76 Define PWM_PB_CFG_AR_RELEASE_MINOR_VERSION_C....45
3.6.1.77 Define PWM_PB_CFG_AR_RELEASE_REVISION_VERSION_C....45
3.6.1.78 Define PWM_PB_CFG_SW_MAJOR_VERSION_C....45
3.6.1.79 Define PWM_PB_CFG_SW_MINOR_VERSION_C.... 45
3.6.1.80 Define PWM_PB_CFG_SW_PATCH_VERSION_C.... 46
3.6.1.81 Define PWM_START_SEC_CODE.... 46
3.6.1.82 Define PWM_START_SEC_CONFIG_DATA_UNSPECIFIED....46
3.6.1.83 Define PWM_START_SEC_VAR_INIT_UNSPECIFIED.... 46
3.6.1.84 Define PWM_STOP_SEC_CODE.... 46
3.6.1.85 Define PWM_STOP_SEC_CONFIG_DATA_UNSPECIFIED....47
3.6.1.86 Define PWM_STOP_SEC_VAR_INIT_UNSPECIFIED.... 47
3.6.1.87 Define PWM_INIT_ID....47
3.6.1.88 Define PWM_DEINIT_ID....47
3.6.1.89 Define PWM_SETDUTYCYCLE_ID....48
3.6.1.90 Define PWM_SETPERIODANDDUTY_ID....48
3.6.1.91 Define PWM_SETOUTPUTTOIDLE_ID....48
3.6.1.92 Define PWM_GETOUTPUTSTATE_ID....49
3.6.1.93 Define PWM_DISABLENOTIFICATION_ID....49
3.6.1.94 Define PWM_ENABLENOTIFICATION_ID....49
3.6.1.95 Define PWM_GETVERSIONINFO_ID....50
3.6.1.96 Define PWM_GETCHANNELSTATE_ID....50
3.6.1.97 Define PWM_SETCOUNTERBUS_ID.... 50

3.6.1.98 Define PWM_SETCHANNELOUTPUT_ID.....51
3.6.1.99 Define PWM_SETTRIGGERDELAY_ID.....51
3.6.1.100 Define PWM_BUFFERTRANSFERENDIS_ID.....51
3.6.1.101 Define PWM_SETCLOCKMODE_ID.....52
3.6.1.102 Define PWM_SYNCUPDATE_ID.....52
3.6.1.103 Define PWM_SETPERIODANDDUTY_NO_UPDATE_ID.....53
3.6.1.104 Define PWM_SETDUTYCYCLE_NO_UPDATE_ID.....53
3.6.1.105 Define PWM_PROCESSNOTIFICATION_ID.....53
3.6.1.106 Define PWM_DUTY_CYCLE_100.....54
3.6.1.107 Define PWM_DE_INIT_API.....54
3.6.1.108 Define PWM_SET_CLOCK_MODE_API.....54
3.6.1.109 Define PWM_GET_CHANNEL_STATE_API.....54
3.6.1.110 Define PWM_GET_OUTPUT_STATE_API.....55
3.6.1.111 Define PWM_SET_DUTY_CYCLE_API.....55
3.6.1.112 Define PWM_SET_OUTPUT_TO_IDLE_API.....55
3.6.1.113 Define PWM_SET_PERIOD_AND_DUTY_API.....55
3.6.1.114 Define PWM_SET_COUNTER_BUS_API.....56
3.6.1.115 Define PWM_SET_CHANNEL_OUTPUT_API.....56
3.6.1.116 Define PWM_SET_TRIGGER_DELAY_API.....56
3.6.1.117 Define PWM_BUFFER_TRANSFER_EN_DIS_API.....56
3.6.1.118 Define PWM_VERSION_INFO_API.....57
3.6.1.119 Define PWM_E_ALREADY_INITIALIZED.....57
3.6.1.120 Define PWM_E_DUTYCYCLE_RANGE.....57
3.6.1.121 Define PWM_E_PARAM_CHANNEL.....58
3.6.1.122 Define PWM_E_INIT_FAILED.....58
3.6.1.123 Define PWM_E_PARAM_NOTIFICATION.....59
3.6.1.124 Define PWM_E_PARAM_NOTIFICATION_NULL.....59
3.6.1.125 Define PWM_E_PARAM_POINTER.....60
3.6.1.126 Define PWM_E_PERIOD_UNCHANGEABLE.....60

3.6.1.127 Define PWM_E_UNEXPECTED_ISR.... 60
3.6.1.128 Define PWM_E_UNINIT.... 61
3.6.1.129 Define PWM_E_PERIODVALUE.... 61
3.6.1.130 Define PWM_E_COUNTERBUS.... 61
3.6.1.131 Define PWM_E_PARAM_INSTANCE.... 62
3.6.1.132 Define PWM_E_CHANNEL_OFFSET_VALUE.... 62
3.6.1.133 Define PWM_E_OPWMB_CHANNEL_OFFSET_DUTYCYCLE_RANGE.... 63
3.6.1.134 Define PWM_E_PERIODVALUE.... 61
3.6.1.135 Define PWM_DEV_ERROR_DETECT.... 63
3.6.1.136 Define PWM_DUTY_PERIOD_UPDATED_ENDPERIOD.... 64
3.6.1.137 Define PWM_DUTYCYCLE_UPDATED_ENDPERIOD.... 64
3.6.1.138 Define PWM_INDEX.... 64
3.6.1.139 Define PWM_PRECOMPILE_SUPPORT.... 65
3.6.1.140 Define PwmChannel_0.... 65
3.6.1.141 Define PwmChannel_1.... 65
3.6.1.142 Define PwmConf_PwmChannelConfigSet_PwmChannel_0.... 65
3.6.1.143 Define PwmConf_PwmChannelConfigSet_PwmChannel_1.... 65
3.6.1.144 Define PWM_EMIOS_M0_C0.... 66
3.6.1.145 Define PWM_FEATURE_DAOC.... 66
3.6.1.146 Define PWM_FEATURE_OPWM.... 66
3.6.1.147 Define PWM_FEATURE_OPWMCB.... 66
3.6.1.148 Define PWM_FEATURE_OPWFM.... 67
3.6.1.149 Define PWM_FEATURE_OPWMT.... 67
3.6.1.150 Define PWM_NOTIFICATON_PLAUSABILITY.... 67
3.6.1.151 Define PWM_OFFSET_PLAUSABILITY.... 67
3.6.1.152 Define PWM_MAX_PERIOD_PLAUSABILITY.... 68
3.6.1.153 Define PWM_EMIOS_16_BITS_VARIANT.... 68
3.6.1.154 Define PWM_EMIOS_HW_MODULES_CFG.... 68
3.6.1.155 Define PWM_EMIOS_HW_CHANNELS.... 68

3.6.1.156 Define PWM_MAX_PERIOD 69
3.6.1.157 Define PWM_DEV_ERROR_DETECT 63
3.6.1.158 Define PWM_DUTY_PERIOD_UPDATED_ENDPERIOD 64
3.6.1.159 Define PWM_DUTYCYCLE_UPDATED_ENDPERIOD 64
3.6.1.160 Define PWM_HW_CHANNELS_NO 70
3.6.1.161 Define PWM_HW_MODULES_CFG 70
3.6.1.162 Define PWM_NOTIFICATION_SUPPORTED 70
3.6.1.163 Define PWM_INIT_CONFIG_PB_DEFINES 70
3.6.1.164 Define PWM_INDEX 64
3.6.1.165 Define PWM_PRECOMPILE_SUPPORT 65
3.6.1.166 Define PWM_ENV_FTE 71
3.6.1.167 Define PWM_ENV_GTE 71
3.6.1.168 Define PWM_ENV_NTE 72
3.6.1.169 Define PWM_ENV_QM 72
3.6.1.170 Define PWM_PARAM_CHECK 72
3.6.1.171 Define PWM_REGISTER_PROTECTION 72
3.6.1.172 Define PWM_VALIDATE_CHANNEL_CONFIG_CALL 72
3.6.1.173 Define PWM_VALIDATE_GLOBAL_CONFIG_CALL 73
3.6.1.174 Define PWM_EMIOS_M0_C0 73
3.6.1.175 Define Pwm_IPW_eMiosChannelNotification 73
3.6.1.176 Define PWM_CONF_CHANNELS_PB_0 73
3.6.1.177 Define PWM_EMIOS_CONF_CHANNELS_PB_0 74
3.6.2 Enum Reference 74
3.6.2.1 Enumeration Pwm_GlobalStateType 74
3.6.2.2 Enumeration Pwm_ChannelClassType 74
3.6.2.3 Enumeration Pwm_OutputStateType 75
3.6.2.4 Enumeration Pwm_PrescalerType 75
3.6.2.5 Enumeration Pwm_EdgeNotificationType 76
3.6.2.6 Enumeration Pwm_StateType 76

Section number	Title	Page

3.6.3 Function Reference 77

3.6.3.1 Function Pwm_DeInit 77

3.6.3.2 Function Pwm_DisableNotification 78

3.6.3.3 Function Pwm_EnableNotification 79

3.6.3.4 Function Pwm_EndValidateChannelConfigCall 80

3.6.3.5 Function Pwm_EndValidateGlobalConfigCall 80

3.6.3.6 Function Pwm_GetChannelState 81

3.6.3.7 Function Pwm_GetOutputState 81

3.6.3.8 Function Pwm_GetVersionInfo 82

3.6.3.9 Function Pwm_Init 83

3.6.3.10 Function Pwm_Notification 84

3.6.3.11 Function Pwm_SetDutyCycle 85

3.6.3.12 Function Pwm_SetOutputToIdle 86

3.6.3.13 Function Pwm_SetPeriodAndDuty 87

3.6.3.14 Function Pwm_SetCounterBus 88

3.6.3.15 Function Pwm_SetChannelOutput 89

3.6.3.16 Function Pwm_BufferTransferEnableDisable 90

3.6.3.17 Function Pwm_SetClockMode 91

3.6.3.18 Function Pwm_ValidateChannelConfigCall 92

3.6.3.19 Function Pwm_ValidateGlobalConfigCall 92

3.6.3.20 Function Pwm_ValidateParamDuty 93

3.6.3.21 Function Pwm_ValidateParamOffsetDuty 94

3.6.3.22 Function Pwm_ValidateParamNotification 94

3.6.3.23 Function Pwm_ValidateParamPtrInit 95

3.6.3.24 Function Pwm_ValidateParamsPeriodDuty 96

3.6.3.25 Function PwmChannel_0_notification 96

3.6.3.26 Function PwmChannel_1_notification 97

3.6.3.27 Function Pwm_GetTargetPowerState 97

3.6.3.28 Function Pwm_GetCurrentPowerState 98

3.6.3.29 Function Pwm_PreparePowerState....99
3.6.3.30 Function Pwm_SetPowerState....99
3.6.4 Structs Reference....100
3.6.4.1 Structure Pwm_ChannelConfigType....100
3.6.4.2 Structure Pwm_ConfigType....101
3.6.4.3 Structure Pwm_eMios_ChannelConfigType....102
3.6.4.4 Structure Pwm_eMios_IpConfigType....103
3.6.4.5 Structure Pwm_IpChannelConfigType....104
3.6.4.6 Structure Pwm_IpConfigType....104
3.6.5 Types Reference....105
3.6.5.1 Typedef Pwm_NotifyType....105
3.6.5.2 Typedef Pwm_ChannelType....105
3.6.5.3 Typedef Pwm_PeriodType....105
3.6.5.4 Typedef Pwm_eMios_ChannelType....106
3.6.5.5 Typedef Pwm_eMios_ControlType....106
3.6.5.6 Typedef Pwm_ChannelIpType....106
3.6.6 Variables Reference....106
3.6.6.1 Variable Pwm_GlobalState....106
3.6.6.2 Variable Pwm_pConfig....106
3.6.6.3 Variable Pwm_Channels_PB_0....107
3.6.6.4 Variable Pwm_eMios_ChannelConfig_PB_0....107
3.6.6.5 Variable Pwm_eMios_IpConfig_PB_0....107
3.6.6.6 Variable Pwm_IpChannelConfig_PB_0....107
3.6.6.7 Variable PwmChannelConfigSet_0....107
3.7 Symbolic Names Disclaimer....108

Chapter 4
Tresos Configuration Plug-in

4.1 Configuration elements of Pwm....109
4.2 Form IMPLEMENTATION_CONFIG_VARIANT....109

Section number	Title	Page

4.3 Form PwmConfigurationOfOptApiServices 110

4.3.1 PwmDeInitApi (PwmConfigurationOfOptApiServices) 110

4.3.2 PwmGetOutputState (PwmConfigurationOfOptApiServices) 111

4.3.3 PwmSetDutyCycle (PwmConfigurationOfOptApiServices) 111

4.3.4 PwmSetOutputToIdle (PwmConfigurationOfOptApiServices) 111

4.3.5 PwmSetPeriodAndDuty (PwmConfigurationOfOptApiServices) 112

4.3.6 PwmVersionInfoApi (PwmConfigurationOfOptApiServices) 112

4.3.7 PwmGetChannelStateApi (PwmConfigurationOfOptApiServices) 112

4.3.8 PwmSetCounterBusApi (PwmConfigurationOfOptApiServices) 113

4.3.9 MultiPwmChannelSynch (PwmConfigurationOfOptApiServices) 113

4.3.10 PwmSetChannelOutputApi (PwmConfigurationOfOptApiServices) 114

4.3.11 PwmSetTriggerDelayApi (PwmConfigurationOfOptApiServices) 114

4.4 Form PwmGeneral 114

4.4.1 PwmDevErorDetect (PwmGeneral) 115

4.4.2 PwmDutycycleUpdatedEndperiod (PwmGeneral) 115

4.4.3 PwmNotificationSupported (PwmGeneral) 115

4.4.4 PwmPeriodUpdatedEndperiod (PwmGeneral) 116

4.4.5 PwmEnableDualClockMode (PwmGeneral) 116

4.4.6 PwmChangeRegisterA (PwmGeneral) 117

4.4.7 PwmIndex (PwmGeneral) 117

4.4.8 PwmEnableUserModeSupport(PwmGeneral) 118

4.4.9 PwmTimerPrecision (PwmGeneral) 118

4.5 Form CommonPublishedInformation 118

4.5.1 ArReleaseMajorVersion (CommonPublishedInformation) 119

4.5.2 ArReleaseMinorVersion (CommonPublishedInformation) 119

4.5.3 ArReleaseRevisionVersion (CommonPublishedInformation) 120

4.5.4 ModuleId (CommonPublishedInformation) 120

4.5.5 SwMajorVersion (CommonPublishedInformation) 121

4.5.6 SwMinorVersion (CommonPublished Information) 121

4.5.7 SwPatchVersion (CommonPublishedInformation) 121
4.5.8 VendorApiInfix (CommonPublishedInformation) 122
4.5.9 VendorId (CommonPublishedInformation) 122
4.6 Form PwmChannelConfigSet 123
4.6.1 Form PwmChannel 123
4.6.1.1 PwmChannelId (PwmChannel) 123
4.6.1.2 PwmHwIP (PwmChannel) 124
4.6.1.3 PwmeMiosChannel (PwmChannel) 124
4.6.1.4 PwmPeriodInTicks (PwmChannel) 125
4.6.1.5 PwmPeriodDefault (PwmChannel) 125
4.6.1.6 PwmChannelClass (PwmChannel) 125
4.6.1.7 PwmPolarity (PwmChannel) 126
4.6.1.8 PwmDutycycleDefault (PwmChannel) 126
4.6.1.9 PwmIdleState (PwmChannel) 126
4.6.1.10 PwmNotification (PwmChannel) 127
4.6.1.11 PwmMcuClockReferencePoint (PwmChannel) 127
4.6.2 Form PwmeMios 127
4.6.2.1 PwmeMiosModule (PwmeMios) 128
4.6.2.2 Form PwmeMiosChannels 128
4.6.2.2.1 PwmeMiosChannel (PwmeMiosChannels) 129
4.6.2.2.2 PwmeMiosClock (PwmeMiosChannels) 130
4.6.2.2.3 PwmeMiosClock_Alternate (PwmeMiosChannels) 130
4.6.2.2.4 PwmModeSelect (PwmeMiosChannels) 131
4.6.2.2.5 EmiosUnifiedChannelBusSelect (PwmeMiosChannels) 131
4.6.2.2.6 PwmOffset (PwmeMiosChannels) 131
4.6.2.2.7 PwmTriggerDelay (PwmeMiosChannels) 132
4.6.2.2.8 Pwm_Deadtime (PwmeMiosChannels) 132
4.6.2.2.9 OffsetDelayAdjust (PwmeMiosChannels) 133
4.6.2.3 Form PwmeMiosMasterBus 133

4.6.2.3.1 PwmeMiosMasterBus (PwmeMiosMasterBus)....133
4.6.2.3.2 MasterBusPrescaler (PwmeMiosMasterBus).... 134
4.6.2.3.3 MasterBusPrescaler_Alternate (PwmeMiosMasterBus).... 134
4.6.2.3.4 MasterModeSelect (PwmeMiosMasterBus).... 135
4.6.2.3.5 PwmPeriodInTicks (PwmeMiosMasterBus).... 135
4.6.2.3.6 MasterBusPeriodDefault (PwmeMiosMasterBus)....135
4.6.3 Form PwmHwIntrruptConfigList.... 136
4.6.3.1 PwmInterruptName(PwmHwIntrruptConfigList)....137
4.6.3.2 PwmInterruptEnable(PwmHwIntrruptConfigList).... 139
4.6.3.3 PwmChannelIsUsed (PwmHwIntrruptConfigList)....139

Chapter 1
Revision History

Table 1-1. Revision History

Revision	Date	Author	Description
1.0.0	22/08/2014	Do The Viet	Pwm User Manual for MPC574XG - 0.9.0 Release
2.0.0	24/04/2015	Do The VIet	Pwm User Manual for MPC574XG - RTM1.0.0 Release
3.0.0	10/07/2015	Do The Viet	Pwm User Manual for MPC574XG - RTM1.0.1 Release
4.0.0	12/08/2016	Lam Nguyen	Pwm User Manual for MPC574XG - RTM1.0.2 Release
5.0.0	17/02/2017	An Do Xuan	Pwm User Manual for MPC574XG AUTOSAR 4.2.2- RTM1.0.0 Release

Chapter 2
Introduction

This User Manual describes NXP Semiconductor AUTOSAR Pulse Width Modulation (PWM) for MPC574XG .

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 Semiconductor .

Table 2-1. MPC574XG Derivatives

NXP Semiconductor	MPC5748G_LQFP176, MPC5748G_MAPBGA256, MPC5748G_MAPBGA324, MPC5747G_LQFP176, MPC5747G_MAPBGA256, MPC5747G_MAPBGA324, MPC5746G_LQFP176, MPC5746G_MAPBGA256, MPC5746G_MAPBGA324, MPC5748C_LQFP176, MPC5748C_MAPBGA256, MPC5748C_MAPBGA324, MPC5747C_LQFP176, MPC5747C_MAPBGA256, MPC5747C_MAPBGA324, MPC5746C_LQFP176, MPC5746C_MAPBGA256, MPC5746C_MAPBGA324, MPC5746C_MAPBGA100, MPC5745C_LQFP176, MPC5745C_MAPBGA256, MPC5745C_MAPBGA100, MPC5744C_LQFP176, MPC5744C_MAPBGA256,

Table 2-1. MPC574XG Derivatives

	MPC5744C_MAPBGA100, MPC5746B_LQFP176, MPC5746B_MAPBGA256, MPC5746B_MAPBGA100, MPC5744B_LQFP176, MPC5744B_MAPBGA256, MPC5744B_MAPBGA100, MPC5745B_LQFP176, MPC5745B_MAPBGA256, MPC5745B_MAPBGA100

All of the above microcontroller devices are collectively named as MPC574XG .

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
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
ISR	Interrupt Service Routine
MCU	Microcontroller Unit
PWM	Pulse Width Modulation
RAM	Random Access Memory
ROM	Read-only Memory
OS	Operating System
PB Variant	Post Build Variant
PC Variant	Pre Compile Variant
VLE	Variable Length Encoding
N/A	Not Applicable
MCU	Micro Controller Unit
EMIOS	Configurable Enhanced Modular IO Subsystem

2.5 Reference List

Table 2-3. Reference List

#	Title	Version
1	AUTOSAR 4.2 Rev0002PWM Driver Software Specification Document.	v2.5.0
2	MPC5748G Reference Manual	Rev. 5, 12/2016

Table continues on the next page...

Table 2-3. Reference List (continued)

#	Title	Version
3	MPC5746C Reference Manual	Rev. 4, 12/2016
4	MPC5748G_1N81M_Rev.2 (official document) (1N81M)	Jun-16
5	MPC5748G_1N81M_0N78S_Comparison_Summary_v2_0 (internal document) (1N81M, 0N78S)	31.10.2016
6	MPC5746C_1N06M_Rev.4 (official document) (1N06M)	Jul-16
7	MPC5746C_cut1.1_cut2.0_cut2.1_comparison_v0 (internal document) (1N06M, 0N84S, 1N84S)	14-Sep-16
8	C3M_cut2.1_new_errata_20170113 (internal document) (1N84S)	13-Jan-17

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 EMIOS modules.

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 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.

1. Additional Requirement 1
2. Additional Requirement 2

3. Additional Requirement ...
4. Additional Requirement N

Table 3-1. 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 3-2. Driver Deviations Table

Requirement	Status	Description	Notes
PWM005	N/S	Production errors shall be reported to Diagnostic Event Manager.	PWM does not report any errors to DEM.
PWM066a	N/S	The PWM Driver module shall optionally include the Dem.h file if any production error will be issued by the implementation.	PWM does not report any errors to DEM.
PWM066b	N/S	The name of the Event Id symbols are implementation specific.	This is not a requirement.
PWM075d	N/S	Pwm_Lcfg.c shall include Pwm.h and Memmap.h	Link time configuration not supported.
PWM075g	N/S	Pwm_Irq.c shall include 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
PWM094	N/F	This chapter defines all interfaces which are required to fulfill an optional functionality of the module: Dem_ReportErrorStatus - Queues the reported events from ...	PWM does not report any errors to DEM.
PWM104	N/F	This chapter lists all types included from other modules: Dem - Dem_EventIdType, Dem_EventStatusType...	PWM does not report any errors to DEM.
PWM145	N/S	Each variable that shall be accessible by AUTOSAR Debugging, shall be defined as global variable.	Pwm driver does not implement AUTOSAR Debugging concept
PWM147	N/S	All type definitions of variables which shall be debugged shall be accessible by the header file PWM.h	Pwm driver does not implement AUTOSAR Debugging concept
PWM148	N/S	The declaration of variables in the header file shall be such that it is possible to calculate the size of the variables by C-sizeof.	Pwm driver does not implement AUTOSAR Debugging concept
PWM149	N/S	Variables available for debugging shall be described in the respective Basic Software Module Description.	Pwm driver does not implement AUTOSAR Debugging concept

3.4 Runtime Errors

None.

3.5 eMIOS specific implementation details

Operation modes:

Pwm driver can support following operation modes :

- OPWFMB

- OPWMB

- OPWMCB

- OPWMT

- DAOC

eMIOS counter:

Please note that the eMIOS internal hardware counter will always start from 0x1. This is true for all channels types in the current implementation: OPWFMB, OPWMB, OPWMT, DAOC and OPWMCB

The configured default period value is incremented in the code by 1 at Pwm_Init(). Pwm_SetPeriodAndDuty() does not perform any other operation on the period value parameter and the value is written in the hardware registers.

There are 4 Counter buses can be used by unified channel

PWM_BUS_A: used for DAOC, OPWMCB, OPWMB,OPWMT. Bus A is available on all channels and is driven by eMios channel 23.

PWM_BUS_F: used for DAOC, OPWMCB, OPWMB,OPWMT. Bus F is available on all channels and is driven by eMios channel 22.

PWM_BUS_DIVERSE: used for DAOC, OPWMCB, OPWMB, OPWMT. Counter bus B, C, D or E specific to each unified channel. Bus B is controlled by unified channel 0 and is available for unified channel 0 to channel 7. Bus C is controlled by unified channel 8 and is available for unified channel 8 to channel 15. Bus D is controlled by unified channel 16 and is available for unified channel 16 to channel 23. Bus E is controlled by unified channel 24 and is available for unified channel 24 to channel 31.

PWM_BUS_INTERNAL_COUNTER: used only for OPWFMB. Bus internal is available on channels: 0,1,2,3,4,5,6,7,8,16,22,23,24 from emios_0 and 0, 8, 16, 22,23,24 from emios_1.

Configuring period value on PWM channels using a Master eMIOS channel

Initial Assumptions:

- Pwm channel is expected to run at 100 Hz and is configured on emios_0_channel_9 which operates in OPWMT mode and with the bus set to: BUS_A.
- Clock source for emios_0 was, previously, configured in MCU at 1 MHz. Emios_0_channel_23 prescaler is set to Div_1. For this emios_0_channel_23 needs to be configured and, in order to obtain a Pwm output of 100 Hz, MasterBusPeriodDefault has to be written with the value of 0.010. (MasterBusPeriodInTicks is set to false).

NOTE

When a master bus channel is configured the value written for PwmPeriodDefault is ignored. All PWM channels that use BUS_A will run at 100Hz no matter what operating mode they are using.

eMIOS module sharing and dependency between PWM, Icu and Gpt

eMIOS IP is shared between all timer-based AUTOSAR modules which means that PWM, ICU and GPT may configure different functionalities of the eMIOS hardware. In order to prevent one ASR module overwriting the eMIOS configuration of another ASR module, the user must assure that the following restrictions are met simultaneously: each ASR module should use a different unified channel and each ASR module should configure its own master bus channel. For example if PWM module uses eMIOS0_CH1 in "OPWMT mode" and requires a bus type: DIVERSE (in this case BUS B which is driven by uses eMIOS0_CH0 - see Table 3-3 for details), then neither ICU nor GPT should configure eMIOS0_CH0 or eMIOS0_CH1 either directly or indirectly. Furthermore it is recommended that if BUS DIVERSE is to be used by one ASR module than that bus shall be used only by the said module.

Notifications:

Please note that not all combinations of signal polarity and notifications are supported by the EMIOS implementation:

Unsupported combination for all modes:

1. Polarity PWM_HIGH & Notification PWM_RISING_EDGE

2. Pwm_Polarity PWM_LOW & Notification PWM_FALLING_EDGE

In this case the following DET error will be generated:

Det_ReportError(PWM_MODULE_ID, 0, PWM_ENABLENOTIFICATION_ID, PWM_E_PARAM_NOTIFICATION);

Signal Period for the channels in OPWMB

When using OPWMB mode, please make sure that the master bus must be configured with Up Buffered Counter mode. In this case the period of this OPWMB channel is determined by the period of master bus. Please note that this master bus channel shall not be reconfigured or used for any other purpose.

Signal Period for the channels in OPWMCB

When using OPWMCB mode, please make sure that the master bus must be configured with Up/Down Buffered Counter mode . In this case the period of this OPWMCB channel is determined by the period of master bus. Please note that this master bus channel shall not be reconfigured or used for any other purpose.

Limitations of using DAOC mode

- DAOC mode can be used with external (BUS_A or BUS_DIVERSE) counter bus. In this case, the period of this channel have to less than the period counter of master bus.
- DAOC mode do not support the changing duty at the end of period boundary.
- In order to function, the DAOC mode requires that the channel interrupts are enabled at all time. To avoid interrupt flooding caused by frequent compare match interrupts on a DAOC channel it is recommended to avoid high PWM frequencies (more than 1000 Hz) or extreme values for duty (less than 5% or greater than 95%) even if PWM frequency is no larger than 100 Hz.

NOTE

These numbers are given for a CPU running at 40Mhz and with the compiler configurations as described in *PWM Integration Manual*, *chapter 3.1: Build Options*.

Channel class and operation modes

User can select channel class, include :

- PWM_VARIABLE_PERIOD. This class is supported for OPWFMB and DAOC modes.

- PWM_FIXED_PERIOD. All modes can be support for this class. Please note that when OPWMB is used, in this case the Offset parameter (phase shift) must be set with 0 value.

- PWM_FIXED_PERIOD_SHIFTED, This class is supported for OPWMB modes.

Notifications and OPWMT:

In this mode the trigger event is not generated on the rising / falling edge of the Pwm signal. Instead the PwmTriggerDelay parameter will determine the location of the trigger within the signal period. Please make sure the location of the OPWMT trigger is located within the period of the reference channel. If this condition is not met the trigger will not occur.

By default OPWMT generated flags are routed to CTU.

Calling Pwm_EnableNotification() for OPWMT channels will use the generated issue interrupts to the CPU. In this case the parameter used to specify the notification edge is not used. Calling Pwm_DisableNotification() will route the flags events back to the CTU.

Update Period and Update Duty at the end of period with operation modes

Please note that in current implementation, all modes only support update Period and Dutycycle at end period

Operation modes supported by hardware channel

Table 3-3. Operation modes supported by hardware channel

	DAOC	OPWFMB	OPWMCB	OPWMB	OPWMT
EMIOS_0_CH_0		YES		YES	YES
EMIOS_0_CH_1	YES	YES	YES	YES	YES
EMIOS_0_CH_2	YES	YES	YES	YES	YES
EMIOS_0_CH_3	YES	YES	YES	YES	YES
EMIOS_0_CH_4	YES	YES	YES	YES	YES
EMIOS_0_CH_5	YES	YES	YES	YES	YES
EMIOS_0_CH_6	YES	YES	YES	YES	YES
EMIOS_0_CH_7	YES	YES	YES	YES	YES
EMIOS_0_CH_8		YES		YES	YES
EMIOS_0_CH_9	YES			YES	YES
EMIOS_0_CH_10	YES			YES	YES
EMIOS_0_CH_11	YES			YES	YES
EMIOS_0_CH_12	YES			YES	YES
EMIOS_0_CH_13	YES			YES	YES
EMIOS_0_CH_14	YES			YES	YES
EMIOS_0_CH_15	YES			YES	YES
EMIOS_0_CH_16		YES		YES	YES
EMIOS_0_CH_17				YES	YES
EMIOS_0_CH_18				YES	YES
EMIOS_0_CH_19				YES	YES

Table continues on the next page...

Table 3-3. Operation modes supported by hardware channel (continued)

	DAOC	OPWFMB	OPWMCB	OPWMB	OPWMT
EMIOS_0_CH_20				YES	YES
EMIOS_0_CH_22		YES		YES	YES
EMIOS_0_CH_23		YES		YES	YES
EMIOS_0_CH_24		YES		YES	YES
EMIOS_0_CH_25				YES	YES
EMIOS_0_CH_26				YES	YES
EMIOS_0_CH_27				YES	YES
EMIOS_0_CH_28				YES	YES
EMIOS_0_CH_29				YES	YES
EMIOS_0_CH_30				YES	YES
EMIOS_0_CH_31				YES	YES
EMIOS_1_CH_0		YES		YES	YES
EMIOS_1_CH_1	YES			YES	YES
EMIOS_1_CH_2	YES			YES	YES
EMIOS_1_CH_3	YES			YES	YES
EMIOS_1_CH_4	YES			YES	YES
EMIOS_1_CH_5	YES			YES	YES
EMIOS_1_CH_6	YES			YES	YES
EMIOS_1_CH_7	YES			YES	YES
EMIOS_1_CH_8		YES		YES	YES
EMIOS_1_CH_9	YES			YES	YES
EMIOS_1_CH_10	YES			YES	YES
EMIOS_1_CH_11	YES			YES	YES
EMIOS_1_CH_12	YES			YES	YES
EMIOS_1_CH_13	YES			YES	YES
EMIOS_1_CH_14	YES			YES	YES
EMIOS_1_CH_15	YES			YES	YES
EMIOS_1_CH_16		YES		YES	YES
EMIOS_1_CH_17				YES	YES
EMIOS_1_CH_18				YES	YES
EMIOS_1_CH_19				YES	YES
EMIOS_1_CH_20				YES	YES
EMIOS_1_CH_21				YES	YES
EMIOS_1_CH_22		YES		YES	YES
EMIOS_1_CH_23		YES		YES	YES
EMIOS_1_CH_24		YES		YES	YES
EMIOS_1_CH_25				YES	YES
EMIOS_1_CH_26				YES	YES
EMIOS_1_CH_27				YES	YES

Table continues on the next page...

Table 3-3. Operation modes supported by hardware channel (continued)

	DAOC	OPWFMB	OPWMCB	OPWMB	OPWMT
EMIOS_1_CH_28				YES	YES
EMIOS_1_CH_29				YES	YES
EMIOS_1_CH_30				YES	YES
EMIOS_1_CH_31				YES	YES
EMIOS_2_CH_0		YES		YES	YES
EMIOS_2_CH_1				YES	YES
EMIOS_2_CH_2				YES	YES
EMIOS_2_CH_3				YES	YES
EMIOS_2_CH_4				YES	YES
EMIOS_2_CH_5				YES	YES
EMIOS_2_CH_6				YES	YES
EMIOS_2_CH_7				YES	YES
EMIOS_2_CH_8		YES		YES	YES
EMIOS_2_CH_9				YES	YES
EMIOS_2_CH_10				YES	YES
EMIOS_2_CH_11				YES	YES
EMIOS_2_CH_12				YES	YES
EMIOS_2_CH_13				YES	YES
EMIOS_2_CH_14				YES	YES
EMIOS_2_CH_15				YES	YES
EMIOS_2_CH_16		YES		YES	YES
EMIOS_2_CH_17				YES	YES
EMIOS_2_CH_18				YES	YES
EMIOS_2_CH_19				YES	YES
EMIOS_2_CH_20				YES	YES
EMIOS_2_CH_21				YES	YES
EMIOS_2_CH_22		YES		YES	YES
EMIOS_2_CH_23		YES		YES	YES
EMIOS_2_CH_24		YES		YES	YES
EMIOS_2_CH_25				YES	YES
EMIOS_2_CH_26				YES	YES
EMIOS_2_CH_27				YES	YES
EMIOS_2_CH_28				YES	YES
EMIOS_2_CH_29				YES	YES
EMIOS_2_CH_30				YES	YES
EMIOS_2_CH_31				YES	YES

3.6 Software specification

The following sections contains driver software specifications.

3.6.1 Define Reference

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

3.6.1.1 Define PWM_VENDOR_ID_C

Table 3-4. Define PWM_VENDOR_ID_C Description

Name	PWM_VENDOR_ID_C
Initializer	43

3.6.1.2 Define PWM_MODULE_ID_C

Table 3-5. Define PWM_MODULE_ID_C Description

Name	PWM_MODULE_ID_C
Initializer	121

3.6.1.3 Define PWM_AR_RELEASE_MAJOR_VERSION_C

Table 3-6. Define PWM_AR_RELEASE_MAJOR_VERSION_C Description

Name	PWM_AR_RELEASE_MAJOR_VERSION_C
Initializer	4

3.6.1.4 Define PWM_AR_RELEASE_MINOR_VERSION_C

Table 3-7. Define PWM_AR_RELEASE_MINOR_VERSION_C Description

Name	PWM_AR_RELEASE_MINOR_VERSION_C
Initializer	0

3.6.1.5 Define PWM_AR_RELEASE_REVISION_VERSION_C

Table 3-8. Define PWM_AR_RELEASE_REVISION_VERSION_C Description

Name	PWM_AR_RELEASE_REVISION_VERSION_C
Initializer	3

3.6.1.6 Define PWM_SW_MAJOR_VERSION_C

Table 3-9. Define PWM_SW_MAJOR_VERSION_C Description

Name	PWM_SW_MAJOR_VERSION_C
Initializer	1

3.6.1.7 Define PWM_SW_MINOR_VERSION_C

Table 3-10. Define PWM_SW_MINOR_VERSION_C Description

Name	PWM_SW_MINOR_VERSION_C
Initializer	0

3.6.1.8 Define PWM_SW_PATCH_VERSION_C

Table 3-11. Define PWM_SW_PATCH_VERSION_C Description

Name	PWM_SW_PATCH_VERSION_C
Initializer	2

3.6.1.9 Define PWM_VENDOR_ID

Table 3-12. Define PWM_VENDOR_ID Description

Name	PWM_VENDOR_ID
Initializer	43

3.6.1.10 Define PWM_MODULE_ID

Table 3-13. Define PWM_MODULE_ID Description

Name	PWM_MODULE_ID
Initializer	121

3.6.1.11 Define PWM_AR_RELEASE_MAJOR_VERSION

Table 3-14. Define PWM_AR_RELEASE_MAJOR_VERSION Description

Name	PWM_AR_RELEASE_MAJOR_VERSION
Initializer	4

3.6.1.12 Define PWM_AR_RELEASE_MINOR_VERSION

Table 3-15. Define PWM_AR_RELEASE_MINOR_VERSION Description

Name	PWM_AR_RELEASE_MINOR_VERSION
Initializer	0

3.6.1.13 Define PWM_AR_RELEASE_REVISION_VERSION

Table 3-16. Define PWM_AR_RELEASE_REVISION_VERSION Description

Name	PWM_AR_RELEASE_REVISION_VERSION
Initializer	3

3.6.1.14 Define PWM_SW_MAJOR_VERSION

Table 3-17. Define PWM_SW_MAJOR_VERSION Description

Name	PWM_SW_MAJOR_VERSION
Initializer	1

3.6.1.15 Define PWM_SW_MINOR_VERSION

Table 3-18. Define PWM_SW_MINOR_VERSION Description

Name	PWM_SW_MINOR_VERSION
Initializer	0

3.6.1.16 Define PWM_SW_PATCH_VERSION

Table 3-19. Define PWM_SW_PATCH_VERSION Description

Name	PWM_SW_PATCH_VERSION
Initializer	2

3.6.1.17 Define PWM_PC_CFG_VENDOR_ID_C

Table 3-20. Define PWM_PC_CFG_VENDOR_ID_C Description

Name	PWM_PC_CFG_VENDOR_ID_C
Initializer	43

3.6.1.18 Define PWM_PC_CFG_MODULE_ID_C

Table 3-21. Define PWM_PC_CFG_MODULE_ID_C Description

Name	PWM_PC_CFG_MODULE_ID_C
Initializer	121

3.6.1.19 Define PWM_PC_CFG_AR_RELEASE_MAJOR_VERSION_C

Table 3-22. Define PWM_PC_CFG_AR_RELEASE_MAJOR_VERSION_C Description

Name	PWM_PC_CFG_AR_RELEASE_MAJOR_VERSION_C
Initializer	4

3.6.1.20 Define PWM_PC_CFG_AR_RELEASE_MINOR_VERSION_C

Table 3-23. Define PWM_PC_CFG_AR_RELEASE_MINOR_VERSION_C Description

Name	PWM_PC_CFG_AR_RELEASE_MINOR_VERSION_C
Initializer	0

3.6.1.21 Define PWM_PC_CFG_AR_RELEASE_REVISION_VERSION_C

Table 3-24. Define PWM_PC_CFG_AR_RELEASE_REVISION_VERSION_C Description

Name	PWM_PC_CFG_AR_RELEASE_REVISION_VERSION_C
Initializer	3

3.6.1.22 Define PWM_PC_CFG_SW_MAJOR_VERSION_C

Table 3-25. Define PWM_PC_CFG_SW_MAJOR_VERSION_C Description

Name	PWM_PC_CFG_SW_MAJOR_VERSION_C
Initializer	1

3.6.1.23 Define PWM_PC_CFG_SW_MINOR_VERSION_C

Table 3-26. Define PWM_PC_CFG_SW_MINOR_VERSION_C Description

Name	PWM_PC_CFG_SW_MINOR_VERSION_C
Initializer	0

3.6.1.24 Define PWM_PC_CFG_SW_PATCH_VERSION_C

Table 3-27. Define PWM_PC_CFG_SW_PATCH_VERSION_C Description

Name	PWM_PC_CFG_SW_PATCH_VERSION_C
Initializer	2

3.6.1.25 Define PWM_CFG_VENDOR_ID

Table 3-28. Define PWM_CFG_VENDOR_ID Description

Name	PWM_CFG_VENDOR_ID
Initializer	43

3.6.1.26 Define PWM_CFG_MODULE_ID

Table 3-29. Define PWM_CFG_MODULE_ID Description

Name	PWM_CFG_MODULE_ID
Initializer	121

3.6.1.27 Define PWM_CFG_AR_RELEASE_MAJOR_VERSION

Table 3-30. Define PWM_CFG_AR_RELEASE_MAJOR_VERSION Description

Name	PWM_CFG_AR_RELEASE_MAJOR_VERSION
Initializer	4

3.6.1.28 Define PWM_CFG_AR_RELEASE_MINOR_VERSION

Table 3-31. Define PWM_CFG_AR_RELEASE_MINOR_VERSION Description

Name	PWM_CFG_AR_RELEASE_MINOR_VERSION
Initializer	0

3.6.1.29 Define PWM_CFG_AR_RELEASE_REVISION_VERSION

Table 3-32. Define PWM_CFG_AR_RELEASE_REVISION_VERSION Description

Name	PWM_CFG_AR_RELEASE_REVISION_VERSION
Initializer	3

3.6.1.30 Define PWM_CFG_SW_MAJOR_VERSION

Table 3-33. Define PWM_CFG_SW_MAJOR_VERSION Description

Name	PWM_CFG_SW_MAJOR_VERSION
Initializer	1

3.6.1.31 Define PWM_CFG_SW_MINOR_VERSION

Table 3-34. Define PWM_CFG_SW_MINOR_VERSION Description

Name	PWM_CFG_SW_MINOR_VERSION
Initializer	0

3.6.1.32 Define PWM_CFG_SW_PATCH_VERSION

Table 3-35. Define PWM_CFG_SW_PATCH_VERSION Description

Name	PWM_CFG_SW_PATCH_VERSION
Initializer	2

3.6.1.33 Define PWM_CFG_ENV_VENDOR_ID

Table 3-36. Define PWM_CFG_ENV_VENDOR_ID Description

Name	PWM_CFG_ENV_VENDOR_ID
Initializer	43

3.6.1.34 Define PWM_CFG_ENV_MODULE_ID

Table 3-37. Define PWM_CFG_ENV_MODULE_ID Description

Name	PWM_CFG_ENV_MODULE_ID
Initializer	121

3.6.1.35 Define PWM_CFG_ENV_AR_RELEASE_MAJOR_VERSION

Table 3-38. Define PWM_CFG_ENV_AR_RELEASE_MAJOR_VERSION Description

Name	PWM_CFG_ENV_AR_RELEASE_MAJOR_VERSION
Initializer	4

3.6.1.36 Define PWM_CFG_ENV_AR_RELEASE_MINOR_VERSION

Table 3-39. Define PWM_CFG_ENV_AR_RELEASE_MINOR_VERSION Description

Name	PWM_CFG_ENV_AR_RELEASE_MINOR_VERSION
Initializer	0

3.6.1.37 Define PWM_CFG_ENV_AR_RELEASE_REVISION_VERSION

Table 3-40. Define PWM_CFG_ENV_AR_RELEASE_REVISION_VERSION Description

Name	PWM_CFG_ENV_AR_RELEASE_REVISION_VERSION
Initializer	3

3.6.1.38 Define PWM_CFG_ENV_SW_MAJOR_VERSION

Table 3-41. Define PWM_CFG_ENV_SW_MAJOR_VERSION Description

Name	PWM_CFG_ENV_SW_MAJOR_VERSION
Initializer	1

3.6.1.39 Define PWM_CFG_ENV_SW_MINOR_VERSION

Table 3-42. Define PWM_CFG_ENV_SW_MINOR_VERSION Description

Name	PWM_CFG_ENV_SW_MINOR_VERSION
Initializer	0

3.6.1.40 Define PWM_CFG_ENV_SW_PATCH_VERSION

Table 3-43. Define PWM_CFG_ENV_SW_PATCH_VERSION Description

Name	PWM_CFG_ENV_SW_PATCH_VERSION
Initializer	2

3.6.1.41 Define PWM_EMIOS_TYPES_VENDOR_ID

Table 3-44. Define PWM_EMIOS_TYPES_VENDOR_ID Description

Name	PWM_EMIOS_TYPES_VENDOR_ID
Initializer	43

3.6.1.42 Define PWM_EMIOS_TYPES_MODULE_ID

Table 3-45. Define PWM_EMIOS_TYPES_MODULE_ID Description

Name	PWM_EMIOS_TYPES_MODULE_ID
Initializer	121

3.6.1.43 Define PWM_EMIOS_TYPES_AR_RELEASE_MAJOR_VERSION

Table 3-46. Define PWM_EMIOS_TYPES_AR_RELEASE_MAJOR_VERSION Description

Name	PWM_EMIOS_TYPES_AR_RELEASE_MAJOR_VERSION
Initializer	4

3.6.1.44 Define PWM_EMIOS_TYPES_AR_RELEASE_MINOR_VERSION

Table 3-47. Define PWM_EMIOS_TYPES_AR_RELEASE_MINOR_VERSION Description

Name	PWM_EMIOS_TYPES_AR_RELEASE_MINOR_VERSION
Initializer	0

3.6.1.45 Define PWM_EMIOS_TYPES_AR_RELEASE_REVISION_VERSION

Table 3-48. Define PWM_EMIOS_TYPES_AR_RELEASE_REVISION_VERSION Description

Name	PWM_EMIOS_TYPES_AR_RELEASE_REVISION_VERSION
Initializer	3

3.6.1.46 Define PWM_EMIOS_TYPES_SW_MAJOR_VERSION

Table 3-49. Define PWM_EMIOS_TYPES_SW_MAJOR_VERSION Description

Name	PWM_EMIOS_TYPES_SW_MAJOR_VERSION
Initializer	1

3.6.1.47 Define PWM_EMIOS_TYPES_SW_MINOR_VERSION

Table 3-50. Define PWM_EMIOS_TYPES_SW_MINOR_VERSION Description

Name	PWM_EMIOS_TYPES_SW_MINOR_VERSION
Initializer	0

3.6.1.48 Define PWM_EMIOS_TYPES_SW_PATCH_VERSION

Table 3-51. Define PWM_EMIOS_TYPES_SW_PATCH_VERSION Description

Name	PWM_EMIOS_TYPES_SW_PATCH_VERSION
Initializer	2

3.6.1.49 Define PWM_IPW_NOTIF_VENDOR_ID

Table 3-52. Define PWM_IPW_NOTIF_VENDOR_ID Description

Name	PWM_IPW_NOTIF_VENDOR_ID
Initializer	43

3.6.1.50 Define PWM_IPW_NOTIF_MODULE_ID

Table 3-53. Define PWM_IPW_NOTIF_MODULE_ID Description

Name	PWM_IPW_NOTIF_MODULE_ID
Initializer	121

3.6.1.51 Define PWM_IPW_NOTIF_AR_RELEASE_MAJOR_VERSION

Table 3-54. Define PWM_IPW_NOTIF_AR_RELEASE_MAJOR_VERSION Description

Name	PWM_IPW_NOTIF_AR_RELEASE_MAJOR_VERSION
Initializer	4

3.6.1.52 Define PWM_IPW_NOTIF_AR_RELEASE_MINOR_VERSION

Table 3-55. Define PWM_IPW_NOTIF_AR_RELEASE_MINOR_VERSION Description

Name	PWM_IPW_NOTIF_AR_RELEASE_MINOR_VERSION
Initializer	0

3.6.1.53 Define PWM_IPW_NOTIF_AR_RELEASE_REVISION_VERSION

Table 3-56. Define PWM_IPW_NOTIF_AR_RELEASE_REVISION_VERSION Description

Name	PWM_IPW_NOTIF_AR_RELEASE_REVISION_VERSION
Initializer	3

3.6.1.54 Define PWM_IPW_NOTIF_SW_MAJOR_VERSION

Table 3-57. Define PWM_IPW_NOTIF_SW_MAJOR_VERSION Description

Name	PWM_IPW_NOTIF_SW_MAJOR_VERSION
Initializer	1

3.6.1.55 Define PWM_IPW_NOTIF_SW_MINOR_VERSION

Table 3-58. Define PWM_IPW_NOTIF_SW_MINOR_VERSION Description

Name	PWM_IPW_NOTIF_SW_MINOR_VERSION
Initializer	0

3.6.1.56 Define PWM_IPW_NOTIF_SW_PATCH_VERSION

Table 3-59. Define PWM_IPW_NOTIF_SW_PATCH_VERSION Description

Name	PWM_IPW_NOTIF_SW_PATCH_VERSION
Initializer	2

3.6.1.57 Define PWM_IPW_TYPES_VENDOR_ID

Table 3-60. Define PWM_IPW_TYPES_VENDOR_ID Description

Name	PWM_IPW_TYPES_VENDOR_ID
Initializer	43

3.6.1.58 Define PWM_IPW_TYPES_MODULE_ID

Table 3-61. Define PWM_IPW_TYPES_MODULE_ID Description

Name	PWM_IPW_TYPES_MODULE_ID
Initializer	121

3.6.1.59 Define PWM_IPW_TYPES_AR_RELEASE_MAJOR_VERSION

Table 3-62. Define PWM_IPW_TYPES_AR_RELEASE_MAJOR_VERSION Description

Name	PWM_IPW_TYPES_AR_RELEASE_MAJOR_VERSION
Initializer	4

3.6.1.60 Define PWM_IPW_TYPES_AR_RELEASE_MINOR_VERSION

Table 3-63. Define PWM_IPW_TYPES_AR_RELEASE_MINOR_VERSION Description

Name	PWM_IPW_TYPES_AR_RELEASE_MINOR_VERSION
Initializer	0

3.6.1.61 Define PWM_IPW_TYPES_AR_RELEASE_REVISION_VERSION

Table 3-64. Define PWM_IPW_TYPES_AR_RELEASE_REVISION_VERSION Description

Name	PWM_IPW_TYPES_AR_RELEASE_REVISION_VERSION
Initializer	3

3.6.1.62 Define PWM_IPW_TYPES_SW_MAJOR_VERSION

Table 3-65. Define PWM_IPW_TYPES_SW_MAJOR_VERSION Description

Name	PWM_IPW_TYPES_SW_MAJOR_VERSION
Initializer	1

3.6.1.63 Define PWM_IPW_TYPES_SW_MINOR_VERSION

Table 3-66. Define PWM_IPW_TYPES_SW_MINOR_VERSION Description

Name	PWM_IPW_TYPES_SW_MINOR_VERSION
Initializer	0

3.6.1.64 Define PWM_IPW_TYPES_SW_PATCH_VERSION

Table 3-67. Define PWM_IPW_TYPES_SW_PATCH_VERSION Description

Name	PWM_IPW_TYPES_SW_PATCH_VERSION
Initializer	2

3.6.1.65 Define PWM_NOTIF_VENDOR_ID

Table 3-68. Define PWM_NOTIF_VENDOR_ID Description

Name	PWM_NOTIF_VENDOR_ID
Initializer	43

3.6.1.66 Define PWM_NOTIF_MODULE_ID

Table 3-69. Define PWM_NOTIF_MODULE_ID Description

Name	PWM_NOTIF_MODULE_ID
Initializer	121

3.6.1.67 Define PWM_NOTIF_AR_RELEASE_MAJOR_VERSION

Table 3-70. Define PWM_NOTIF_AR_RELEASE_MAJOR_VERSION Description

Name	PWM_NOTIF_AR_RELEASE_MAJOR_VERSION
Initializer	4

3.6.1.68 Define PWM_NOTIF_AR_RELEASE_MINOR_VERSION

Table 3-71. Define PWM_NOTIF_AR_RELEASE_MINOR_VERSION Description

Name	PWM_NOTIF_AR_RELEASE_MINOR_VERSION
Initializer	0

3.6.1.69 Define PWM_NOTIF_AR_RELEASE_REVISION_VERSION

Table 3-72. Define PWM_NOTIF_AR_RELEASE_REVISION_VERSION Description

Name	PWM_NOTIF_AR_RELEASE_REVISION_VERSION
Initializer	3

3.6.1.70 Define PWM_NOTIF_SW_MAJOR_VERSION

Table 3-73. Define PWM_NOTIF_SW_MAJOR_VERSION Description

Name	PWM_NOTIF_SW_MAJOR_VERSION
Initializer	1

3.6.1.71 Define PWM_NOTIF_SW_MINOR_VERSION

Table 3-74. Define PWM_NOTIF_SW_MINOR_VERSION Description

Name	PWM_NOTIF_SW_MINOR_VERSION
Initializer	0

3.6.1.72 Define PWM_NOTIF_SW_PATCH_VERSION

Table 3-75. Define PWM_NOTIF_SW_PATCH_VERSION Description

Name	PWM_NOTIF_SW_PATCH_VERSION
Initializer	1

3.6.1.73 Define PWM_PB_CFG_VENDOR_ID_C

Table 3-76. Define PWM_PB_CFG_VENDOR_ID_C Description

Name	PWM_PB_CFG_VENDOR_ID_C
Initializer	43

3.6.1.74 Define PWM_PB_CFG_MODULE_ID_C

Table 3-77. Define PWM_PB_CFG_MODULE_ID_C Description

Name	PWM_PB_CFG_MODULE_ID_C
Initializer	121

3.6.1.75 Define PWM_PB_CFG_AR_RELEASE_MAJOR_VERSION_C

Table 3-78. Define PWM_PB_CFG_AR_RELEASE_MAJOR_VERSION_C Description

Name	PWM_PB_CFG_AR_RELEASE_MAJOR_VERSION_C
Initializer	4

3.6.1.76 Define PWM_PB_CFG_AR_RELEASE_MINOR_VERSION_C

Table 3-79. Define PWM_PB_CFG_AR_RELEASE_MINOR_VERSION_C Description

Name	PWM_PB_CFG_AR_RELEASE_MINOR_VERSION_C
Initializer	0

3.6.1.77 Define PWM_PB_CFG_AR_RELEASE_REVISION_VERSION_C

Table 3-80. Define PWM_PB_CFG_AR_RELEASE_REVISION_VERSION_C Description

Name	PWM_PB_CFG_AR_RELEASE_REVISION_VERSION_C
Initializer	3

3.6.1.78 Define PWM_PB_CFG_SW_MAJOR_VERSION_C

Table 3-81. Define PWM_PB_CFG_SW_MAJOR_VERSION_C Description

Name	PWM_PB_CFG_SW_MAJOR_VERSION_C
Initializer	1

3.6.1.79 Define PWM_PB_CFG_SW_MINOR_VERSION_C

Table 3-82. Define PWM_PB_CFG_SW_MINOR_VERSION_C Description

Name	PWM_PB_CFG_SW_MINOR_VERSION_C
Initializer	0

3.6.1.80 Define PWM_PB_CFG_SW_PATCH_VERSION_C

Table 3-83. Define PWM_PB_CFG_SW_PATCH_VERSION_C Description

Name	PWM_PB_CFG_SW_PATCH_VERSION_C
Initializer	2

3.6.1.81 Define PWM_START_SEC_CODE

Table 3-84. Define PWM_START_SEC_CODE Description

Name	PWM_START_SEC_CODE
Initializer	

3.6.1.82 Define PWM_START_SEC_CONFIG_DATA_UNSPECIFIED

Table 3-85. Define PWM_START_SEC_CONFIG_DATA_UNSPECIFIED Description

Name	PWM_START_SEC_CONFIG_DATA_UNSPECIFIED
Initializer	

3.6.1.83 Define PWM_START_SEC_VAR_INIT_UNSPECIFIED

Table 3-86. Define PWM_START_SEC_VAR_INIT_UNSPECIFIED Description

Name	PWM_START_SEC_VAR_INIT_UNSPECIFIED
Initializer	

3.6.1.84 Define PWM_STOP_SEC_CODE

Table 3-87. Define PWM_STOP_SEC_CODE Description

Name	PWM_STOP_SEC_CODE
Initializer	

3.6.1.85 Define PWM_STOP_SEC_CONFIG_DATA_UNSPECIFIED

Table 3-88. Define PWM_STOP_SEC_CONFIG_DATA_UNSPECIFIED Description

Name	PWM_STOP_SEC_CONFIG_DATA_UNSPECIFIED
Initializer	

3.6.1.86 Define PWM_STOP_SEC_VAR_INIT_UNSPECIFIED

Table 3-89. Define PWM_STOP_SEC_VAR_INIT_UNSPECIFIED Description

Name	PWM_STOP_SEC_VAR_INIT_UNSPECIFIED
Initializer	

3.6.1.87 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 3-90. Define PWM_INIT_ID Description

Name	PWM_INIT_ID
Initializer	0x00U

3.6.1.88 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 3-91. Define PWM_DEINIT_ID Description

Name	PWM_DEINIT_ID
Initializer	0x01U

3.6.1.89 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 3-92. Define PWM_SETDUTYCYCLE_ID Description

Name	PWM_SETDUTYCYCLE_ID
Initializer	0x02U

3.6.1.90 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 3-93. Define PWM_SETPERIODANDDUTY_ID Description

Name	PWM_SETPERIODANDDUTY_ID
Initializer	0x03U

3.6.1.91 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 3-94. Define PWM_SETOUTPUTTOIDLE_ID Description

Name	PWM_SETOUTPUTTOIDLE_ID
Initializer	0x04U

3.6.1.92 Define PWM_GETOUTPUTSTATE_ID

API service ID of Pwm_GetOutputState function.

Details:

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

Table 3-95. Define PWM_GETOUTPUTSTATE_ID Description

Name	PWM_GETOUTPUTSTATE_ID
Initializer	0x05U

3.6.1.93 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 3-96. Define PWM_DISABLENOTIFICATION_ID Description

Name	PWM_DISABLENOTIFICATION_ID
Initializer	0x06U

3.6.1.94 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 3-97. Define PWM_ENABLENOTIFICATION_ID Description

Name	PWM_ENABLENOTIFICATION_ID
Initializer	0x07U

3.6.1.95 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 3-98. Define PWM_GETVERSIONINFO_ID Description

Name	PWM_GETVERSIONINFO_ID
Initializer	0x08U

3.6.1.96 Define PWM_GETCHANNELSTATE_ID

API service ID of Pwm_GetChannelState function.

Details:

Parameters used when raising an error/exception

Table 3-99. Define PWM_GETCHANNELSTATE_ID Description

Name	PWM_GETCHANNELSTATE_ID
Initializer	0x20U

3.6.1.97 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 3-100. Define PWM_SETCOUNTERBUS_ID Description

Name	PWM_SETCOUNTERBUS_ID
Initializer	0x22U

3.6.1.98 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 3-101. Define PWM_SETCHANNELOUTPUT_ID Description

Name	PWM_SETCHANNELOUTPUT_ID
Initializer	0x23U

3.6.1.99 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 3-102. Define PWM_SETTRIGGERDELAY_ID Description

Name	PWM_SETTRIGGERDELAY_ID
Initializer	0x24U

3.6.1.100 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 3-103. Define PWM_BUFFERTRANSFERENDIS_ID Description

Name	PWM_BUFFERTRANSFERENDIS_ID
Initializer	0x26U

3.6.1.101 Define PWM_SETCLOCKMODE_ID

API service ID of Pwm_SetOutputToIdle function.

Details:

Parameters used when raising an error/exception

Table 3-104. Define PWM_SETCLOCKMODE_ID Description

Name	PWM_SETCLOCKMODE_ID
Initializer	0x27U

3.6.1.102 Define PWM_SYNCUPDATE_ID

API service ID of Pwm_SetPeriodAndDuty_NoUpdate function.

Details:

Parameters used when raising an error/exception

Table 3-105. Define PWM_SYNCUPDATE_ID

Name	PWM_SYNCUPDATE_ID
Initializer	0x28U

3.6.1.103 Define PWM_SETPERIODANDDUTY_NO_UPDATE_ID

API service ID of Pwm_SetPeriodAndDuty_NoUpdate function.

Details:

Parameters used when raising an error/exception

Table 3-106. Define PWM_SETPERIODANDDUTY_NO_UPDATE_ID Description

Name	PWM_SETPERIODANDDUTY_NO_UPDATE_ID
Initializer	0x29U

3.6.1.104 Define PWM_SETDUTYCYCLE_NO_UPDATE_ID

API service ID of Pwm_SetDutyCycle_NoUpdate function

Details:

Parameters used when raising an error/exception

Table 3-107. Define PWM_SETDUTYCYCLE_NO_UPDATE_ID Description

Name	PWM_SETDUTYCYCLE_NO_UPDATE_ID
Initializer	0x2AU

3.6.1.105 Define PWM_PROCESSNOTIFICATION_ID

API service ID.

Table 3-108. Define PWM_PROCESSNOTIFICATION_ID Description

Name	PWM_PROCESSNOTIFICATION_ID
Initializer	(0x0AU)

3.6.1.106 Define PWM_DUTY_CYCLE_100

100% dutycycle.

Details:

Value used to map a DutyCycle of 100% on a 16 bit variable

Table 3-109. Define PWM_DUTY_CYCLE_100 Description

Name	PWM_DUTY_CYCLE_100
Initializer	((uint16)0x8000U)

3.6.1.107 Define PWM_DE_INIT_API

Switch to indicate that Pwm_DeInit API is supported.

Table 3-110. Define PWM_DE_INIT_API Description

Name	PWM_DE_INIT_API
Initializer	(STD_ON)

3.6.1.108 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 3-111. Define PWM_SET_CLOCK_MODE_API Description

Name	PWM_SET_CLOCK_MODE_API
Initializer	(STD_ON)

3.6.1.109 Define PWM_GET_CHANNEL_STATE_API

Switch to indicate that Pwm_GetChannelState API is supported.

Table 3-112. Define PWM_GET_CHANNEL_STATE_API Description

Name	PWM_GET_CHANNEL_STATE_API
Initializer	(STD_ON)

3.6.1.110 Define PWM_GET_OUTPUT_STATE_API

Switch to indicate that Pwm_GetOutputState API is supported.

Table 3-113. Define PWM_GET_OUTPUT_STATE_API Description

Name	PWM_GET_OUTPUT_STATE_API
Initializer	(STD_ON)

3.6.1.111 Define PWM_SET_DUTY_CYCLE_API

Switch to indicate that Pwm_SetDutyCycle API is supported.

Table 3-114. Define PWM_SET_DUTY_CYCLE_API Description

Name	PWM_SET_DUTY_CYCLE_API
Initializer	(STD_ON)

3.6.1.112 Define PWM_SET_OUTPUT_TO_IDLE_API

Switch to indicate that Pwm_SetOutputToIdle API is supported.

Table 3-115. Define PWM_SET_OUTPUT_TO_IDLE_API Description

Name	PWM_SET_OUTPUT_TO_IDLE_API
Initializer	(STD_ON)

3.6.1.113 Define PWM_SET_PERIOD_AND_DUTY_API

Switch to indicate that Pwm_SetPeriodAndDuty API is supported.

Table 3-116. Define PWM_SET_PERIOD_AND_DUTY_API Description

Name	PWM_SET_PERIOD_AND_DUTY_API
Initializer	(STD_ON)

3.6.1.114 Define PWM_SET_COUNTER_BUS_API

Switch to indicate that Pwm_SetCounterBus API is supported.

Table 3-117. Define PWM_SET_COUNTER_BUS_API Description

Name	PWM_SET_COUNTER_BUS_API
Initializer	(STD_ON)

3.6.1.115 Define PWM_SET_CHANNEL_OUTPUT_API

Switch to indicate that Pwm_SetChannelOutput API is supported.

Table 3-118. Define PWM_SET_CHANNEL_OUTPUT_API Description

Name	PWM_SET_CHANNEL_OUTPUT_API
Initializer	(STD_ON)

3.6.1.116 Define PWM_SET_TRIGGER_DELAY_API

Switch to indicate that Pwm_SetTriggerDelay API is supported.

Table 3-119. Define PWM_SET_TRIGGER_DELAY_API Description

Name	PWM_SET_TRIGGER_DELAY_API
Initializer	(STD_ON)

3.6.1.117 Define PWM_BUFFER_TRANSFER_EN_DIS_API

Switch to indicate that Pwm_BufferTransferEnableDisable API is supported.

Table 3-120. Define PWM_BUFFER_TRANSFER_EN_DIS_API Description

Name	PWM_BUFFER_TRANSFER_EN_DIS_API
Initializer	(STD_ON)

3.6.1.118 Define PWM_VERSION_INFO_API

Switch to indicate that Pwm_GetVersionInfo API is supported.

Table 3-121. Define PWM_VERSION_INFO_API Description

Name	PWM_VERSION_INFO_API
Initializer	(STD_ON)

3.6.1.119 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

Table 3-122. Define PWM_E_ALREADY_INITIALIZED Description

Name	PWM_E_ALREADY_INITIALIZED
Initializer	0x14U

3.6.1.120 Define PWM_E_DUTYCYCLE_RANGE

Error signaling that Pwm_SetDutyCycle or Pwm_SetPeriodAndDuty service was called with invalid dutycycle range.

Details:

Will be reported to DET when Pwm_SetDutyCycle or Pwm_SetPeriodAndDuty service is called with a value for dutycyle out of valid range [0x0000, 0x8000].

Implements: Pwm_ErrorIds_define

Table 3-123. Define PWM_E_DUTYCYCLE_RANGE Description

Name	PWM_E_DUTYCYCLE_RANGE
Initializer	0x17U

3.6.1.121 Define PWM_E_PARAM_CHANNEL

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

Details:

Will be reported to DET when any of the Pwm channel services except Pwm_GetOutputState is called with an invalid channel identifier.

Implements: Pwm_ErrorIds_define

Table 3-124. Define PWM_E_PARAM_CHANNEL Description

Name	PWM_E_PARAM_CHANNEL
Initializer	0x12U

3.6.1.122 Define PWM_E_INIT_FAILED

API Pwm_Init service called with wrong parameter.

Details:

Errors and exceptions that will be detected by the PWM driver

Implements: Pwm_ErrorIds_define AUTOSAR

Table 3-125. Define PWM_E_INIT_FAILED Description

Name	PWM_E_INIT_FAILED
Initializer	0x10U

3.6.1.123 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 3-126. Define PWM_E_PARAM_NOTIFICATION Description

Name	PWM_E_PARAM_NOTIFICATION
Initializer	0x18U

3.6.1.124 Define PWM_E_PARAM_NOTIFICATION_NULL

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

Details:

Will be reported to DET when a NULL function is configured as notification callback for a pwm channel and Pwm_EnableNotification service is called for that particular channel.

Implements: Pwm_ErrorIds_define

Table 3-127. Define PWM_E_PARAM_NOTIFICATION_NULL Description

Name	PWM_E_PARAM_NOTIFICATION_NULL
Initializer	0x16U

3.6.1.125 Define PWM_E_PARAM_POINTER

Error signaling that an invalid parameter pointer is passed to Pwm_GetVersionInfo function.

Details:

Will be reported to DET when Pwm_GetVersionInfo function is called with NULL pointer parameter.

Implements: Pwm_ErrorIds_define

Table 3-128. Define PWM_E_PARAM_POINTER Description

Name	PWM_E_PARAM_POINTER
Initializer	0x15U

3.6.1.126 Define PWM_E_PERIOD_UNCHANGEABLE

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

Details:

Will be reported to DET when Pwm_SetPeriodAndDuty service is called for a channel configured with fixed period.

Implements: Pwm_ErrorIds_define

Table 3-129. Define PWM_E_PERIOD_UNCHANGEABLE Description

Name	PWM_E_PERIOD_UNCHANGEABLE
Initializer	0x13U

3.6.1.127 Define PWM_E_UNEXPECTED_ISR

Error signaling that an unexpected Pwm interrupt has been triggered.

Details:

Will be reported to DET when one of the IPs used by the Pwm driver generates an interrupt while the Pwm driver is in un-initialized state.

Implements: Pwm_ErrorIds_define Non-AUTOSAR

Table 3-130. Define PWM_E_UNEXPECTED_ISR Description

Name	PWM_E_UNEXPECTED_ISR
Initializer	0x30U

3.6.1.128 Define PWM_E_UNINIT

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

Details:

Will be reported to DET when any of the Pwm services except Pwm_Init and Pwm_GetOutputState is called before module initialization.

Implements: Pwm_ErrorIds_define

Table 3-131. Define PWM_E_UNINIT Description

Name	PWM_E_UNINIT
Initializer	0x11U

3.6.1.129 Define PWM_E_PERIODVALUE

Symbolic Name for period det error.

Table 3-132. Define PWM_E_PERIODVALUE Description

Name	PWM_E_PERIODVALUE
Initializer	(0x1AU)

3.6.1.130 Define PWM_E_COUNTERBUS

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

Details:

Will be reported to DET when Pwm_SetCounterBus is called with an invalid Bus.

Implements: Pwm_ErrorIds_define

Table 3-133. Define PWM_E_COUNTERBUS Description

Name	PWM_E_COUNTERBUS
Initializer	0x19U

3.6.1.131 Define PWM_E_PARAM_INSTANCE

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

Details:

Will be reported to DET when Pwm_BufferTransferEnableDisable is called with module id is more than the number of module that supported by this platform.

Implements: Pwm_ErrorIds_define

Table 3-134. Define PWM_E_PARAM_INSTANCE Description

Name	PWM_E_PARAM_INSTANCE
Initializer	0x1DU

3.6.1.132 Define PWM_E_CHANNEL_OFFSET_VALUE

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

Details:

Pwm_Init and Pwm_SetDutyCycle will report to DET when the configured offset for the OPWMB channel is more than the period of the associated MCB channel

Implements: Pwm_ErrorIds_define

Table 3-135. Define PWM_E_CHANNEL_OFFSET_VALUE Description

Name	PWM_E_CHANNEL_OFFSET_VALUE
Initializer	0x1EU

3.6.1.133 Define PWM_E_OPWMB_CHANNEL_OFFSET_DUTYCYCLE_RANGE

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

Details:

Will be reported to DET when the requested offset value plus the current requested dutycycle leads to programming event B over the Period value leading to unexpected behavior of the pwm signal.

Implements: Pwm_ErrorIds_define

Table 3-136. Define PWM_E_OPWMB_CHANNEL_OFFSET_DUTYCYCLE_RANGE Description

Name	PWM_E_OPWMB_CHANNEL_OFFSET_DUTYCYCLE_RANGE
Initializer	0x1BU

3.6.1.134 Define PWM_E_PERIODVALUE

Symbolic Name for period det error.

Table 3-137. Define PWM_E_PERIODVALUE Description

Name	PWM_E_PERIODVALUE
Initializer	(0x1AU)

3.6.1.135 Define PWM_DEV_ERROR_DETECT

Switch for enabling the development error detection.

Table 3-138. Define PWM_DEV_ERROR_DETECT Description

Name	PWM_DEV_ERROR_DETECT
Initializer	(STD_ON)

3.6.1.136 Define PWM_DUTY_PERIOD_UPDATED_ENDPERIOD

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

Table 3-139. Define PWM_DUTY_PERIOD_UPDATED_ENDPERIOD Description

Name	PWM_DUTY_PERIOD_UPDATED_ENDPERIOD
Initializer	(STD_ON)

3.6.1.137 Define PWM_DUTYCYCLE_UPDATED_ENDPERIOD

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

Table 3-140. Define PWM_DUTYCYCLE_UPDATED_ENDPERIOD Description

Name	PWM_DUTYCYCLE_UPDATED_ENDPERIOD
Initializer	(STD_ON)

3.6.1.138 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)

3.6.1.139 Define PWM_PRECOMPILE_SUPPORT

Pwm Pre Compile Switch.

Table 3-142. Define PWM_PRECOMPILE_SUPPORT Description

Name	PWM_PRECOMPILE_SUPPORT
Initializer	(STD_ON)

3.6.1.140 Define PwmChannel_0

Symbolic Names for configured channels.

Table 3-143. Define PwmChannel_0 Description

Name	PwmChannel_0
Initializer	(`Pwm_ChannelType`)0U

3.6.1.141 Define PwmChannel_1

Table 3-144. Define PwmChannel_1 Description

Name	PwmChannel_1
Initializer	(`Pwm_ChannelType`)1U

3.6.1.142 Define PwmConf_PwmChannelConfigSet_PwmChannel_0

Symbolic Names for configured channels. (ecuc 2108 compliant)

Table 3-145. Define PwmConf_PwmChannelConfigSet_PwmChannel_0 Description

Name	PwmConf_PwmChannelConfigSet_PwmChannel_0
Initializer	(`Pwm_ChannelType`)0U

3.6.1.143 Define PwmConf_PwmChannelConfigSet_PwmChannel_1

Symbolic Names for configured channels. (ecuc 2108 compliant)

Table 3-146. Define PwmConf_PwmChannelConfigSet_PwmChannel_1 Description

Name	PwmConf_PwmChannelConfigSet_PwmChannel_1
Initializer	(`Pwm_ChannelType`)1U

3.6.1.144 Define PWM_EMIOS_M0_C0

eMios module and channel macros for module 0

Table 3-147. Define PWM_EMIOS_M0_C0 Description

Name	PWM_EMIOS_M0_C0
Initializer	(0U)

3.6.1.145 Define PWM_FEATURE_DAOC

Macro used to enable or disable DAOC mode

Table 3-148. Define PWM_FEATURE_DAOC Description

Name	PWM_FEATURE_DAOC
Initializer	(STD_OFF)

3.6.1.146 Define PWM_FEATURE_OPWM

Macro used to enable or disable OPWM mode. In current implementation, this define alway STD_OFF

Table 3-149. Define PWM_FEATURE_OPWM Description

Name	PWM_FEATURE_OPWM
Initializer	(STD_OFF)

3.6.1.147 Define PWM_FEATURE_OPWMCB

Macro used to enable or disable OPWMCB mode

Table 3-150. Define PWM_FEATURE_OPWMCB Description

Name	PWM_FEATURE_OPWMCB
Initializer	(STD_OFF)

3.6.1.148 Define PWM_FEATURE_OPWFM

Macro used to enable or disable OPWFM mode. In current implementation, this define alway STD_OFF

Table 3-151. Define PWM_FEATURE_OPWFM Description

Name	PWM_FEATURE_OPWFM
Initializer	(STD_OFF)

3.6.1.149 Define PWM_FEATURE_OPWMT

Macro used to enable or disable OPWMT mode

Table 3-152. Define PWM_FEATURE_OPWMT Description

Name	PWM_FEATURE_OPWMT
Initializer	(STD_OFF)

3.6.1.150 Define PWM_NOTIFICATON_PLAUSABILITY

Table 3-153. Define PWM_NOTIFICATON_PLAUSABILITY Description

Name	PWM_NOTIFICATON_PLAUSABILITY
Initializer	(STD_ON)

3.6.1.151 Define PWM_OFFSET_PLAUSABILITY

Table 3-154. Define PWM_OFFSET_PLAUSABILITY Description

Name	PWM_OFFSET_PLAUSABILITY
Initializer	(STD_ON)

3.6.1.152 Define PWM_MAX_PERIOD_PLAUSABILITY

Table 3-155. Define PWM_MAX_PERIOD_PLAUSABILITY Description

Name	PWM_MAX_PERIOD_PLAUSABILITY
Initializer	(STD_ON)

3.6.1.153 Define PWM_EMIOS_16_BITS_VARIANT

Define the modulos of eMIOS counter is 16 bits

Table 3-156. Define PWM_EMIOS_16_BITS_VARIANT Description

Name	PWM_EMIOS_16_BITS_VARIANT
Initializer	(STD_ON)

3.6.1.154 Define PWM_EMIOS_HW_MODULES_CFG

This define specifies the number of eMios Modules available on the current platform.

Table 3-157. Define PWM_EMIOS_HW_MODULES_CFG Description

Name	PWM_EMIOS_HW_MODULES_CFG
Initializer	(3U)

3.6.1.155 Define PWM_EMIOS_HW_CHANNELS

This define specifies the number of eMios channels available on the current platform.

Table 3-158. Define PWM_EMIOS_HW_CHANNELS Description

Name	PWM_EMIOS_HW_CHANNELS
Initializer	(96U)

3.6.1.156 Define PWM_MAX_PERIOD

This macro define the max value of period that supported by this platform

Table 3-159. Define PWM_MAX_PERIOD Description

Name	PWM_MAX_PERIOD
Initializer	(0xFFFFU)

3.6.1.157 Define PWM_DEV_ERROR_DETECT

Switch for enabling the development error detection.

Table 3-160. Define PWM_DEV_ERROR_DETECT Description

Name	PWM_DEV_ERROR_DETECT
Initializer	(STD_ON)

3.6.1.158 Define PWM_DUTY_PERIOD_UPDATED_ENDPERIOD

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

Table 3-161. Define PWM_DUTY_PERIOD_UPDATED_ENDPERIOD Description

Name	PWM_DUTY_PERIOD_UPDATED_ENDPERIOD
Initializer	(STD_ON)

3.6.1.159 Define PWM_DUTYCYCLE_UPDATED_ENDPERIOD

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

Table 3-162. Define PWM_DUTYCYCLE_UPDATED_ENDPERIOD Description

Name	PWM_DUTYCYCLE_UPDATED_ENDPERIOD
Initializer	(STD_ON)

3.6.1.160 Define PWM_HW_CHANNELS_NO

This define specifies the number of Pwm channels available on the current platform.

Table 3-163. Define PWM_HW_CHANNELS_NO Description

Name	PWM_HW_CHANNELS_NO
Initializer	(96U)

3.6.1.161 Define PWM_HW_MODULES_CFG

This define specifies the number of eMios Modules available on the current platform.

Table 3-164. Define PWM_HW_MODULES_CFG Description

Name	PWM_HW_MODULES_CFG
Initializer	(3U)

3.6.1.162 Define PWM_NOTIFICATION_SUPPORTED

Switch to indicate that the notifications are supported.

Table 3-165. Define PWM_NOTIFICATION_SUPPORTED Description

Name	PWM_NOTIFICATION_SUPPORTED
Initializer	(STD_ON)

3.6.1.163 Define PWM_INIT_CONFIG_PB_DEFINES

Declaration of config sets for PB configuration.

Table 3-166. Define PWM_INIT_CONFIG_PB_DEFINES Description

Name	PWM_INIT_CONFIG_PB_DEFINES
Initializer	extern CONST(`Pwm_ConfigType`, PWM_CONST)`PwmChannelConfigSet_0`;

3.6.1.164 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 3-167. Define PWM_INDEX Description

Name	PWM_INDEX
Initializer	(0U)

3.6.1.165 Define PWM_PRECOMPILE_SUPPORT

Pwm Pre Compile Switch.

Table 3-168. Define PWM_PRECOMPILE_SUPPORT Description

Name	PWM_PRECOMPILE_SUPPORT
Initializer	(STD_ON)

3.6.1.166 Define PWM_ENV_FTE

Table 3-169. Define PWM_ENV_FTE Description

Name	PWM_ENV_FTE
Initializer	STD_ON

3.6.1.167 Define PWM_ENV_GTE

Table 3-170. Define PWM_ENV_GTE Description

Name	PWM_ENV_GTE
Initializer	STD_OFF

3.6.1.168 Define PWM_ENV_NTE

Table 3-171. Define PWM_ENV_NTE Description

Name	PWM_ENV_NTE
Initializer	STD_OFF

3.6.1.169 Define PWM_ENV_QM

Table 3-172. Define PWM_ENV_QM Description

Name	PWM_ENV_QM
Initializer	STD_OFF

3.6.1.170 Define PWM_PARAM_CHECK

Table 3-173. Define PWM_PARAM_CHECK Description

Name	PWM_PARAM_CHECK
Initializer	PWM_DEV_ERROR_DETECT

3.6.1.171 Define PWM_REGISTER_PROTECTION

Table 3-174. Define PWM_REGISTER_PROTECTION Description

Name	PWM_REGISTER_PROTECTION
Initializer	STD_OFF

3.6.1.172 Define PWM_VALIDATE_CHANNEL_CONFIG_CALL

Table 3-175. Define PWM_VALIDATE_CHANNEL_CONFIG_CALL Description

Name	PWM_VALIDATE_CHANNEL_CONFIG_CALL
Initializer	PWM_DEV_ERROR_DETECT

3.6.1.173 Define PWM_VALIDATE_GLOBAL_CONFIG_CALL

Table 3-176. Define PWM_VALIDATE_GLOBAL_CONFIG_CALL Description

Name	PWM_VALIDATE_GLOBAL_CONFIG_CALL
Initializer	PWM_DEV_ERROR_DETECT

3.6.1.174 Define PWM_EMIOS_M0_C0

eMios module and channel macros for module 0

"x" is the id of eMios module, from 0 to 2

"y" is the id of eMios channel, from 0 to 31

Table 3-177. Define PWM_EMIOS_M0_C0 Description

Name	PWM_EMIOS_M0_C0
Initializer	(0U)

3.6.1.175 Define Pwm_IPW_eMiosChannelNotification

Table 3-178. Define Pwm_IPW_eMiosChannelNotification Description

Name	Pwm_IPW_eMiosChannelNotification
Initializer	Pwm_Notification((`Pwm_ChannelType`)Channel)

3.6.1.176 Define PWM_CONF_CHANNELS_PB_0

Number of configured Pwm channels.

Table 3-179. Define PWM_CONF_CHANNELS_PB_0 Description

Name	PWM_CONF_CHANNELS_PB_0
Initializer	2

3.6.1.177 Define PWM_EMIOS_CONF_CHANNELS_PB_0

Number of configured eMios channels.

Table 3-180. Define PWM_EMIOS_CONF_CHANNELS_PB_0 Description

Name	PWM_EMIOS_CONF_CHANNELS_PB_0
Initializer	0

3.6.2 Enum Reference

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

3.6.2.1 Enumeration Pwm_GlobalStateType

Enum containing the possible states of the Pwm driver.

Table 3-181. Enumeration Pwm_GlobalStateType Values

Name	Initializer	Description
PWM_STATE_UNINIT	0x00	
PWM_STATE_IDLE	0x01	

3.6.2.2 Enumeration Pwm_ChannelClassType

Details:

This enum used to identify the channel class type

Implements: Pwm_ChannelClassType_enumeration

Table 3-182. Enumeration Pwm_ChannelClassType Values

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

3.6.2.3 Enumeration Pwm_OutputStateType

Output signal level.

Details:

This enum 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 3-183. Enumeration Pwm_OutputStateType Values

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

3.6.2.4 Enumeration Pwm_PrescalerType

Output signal level.

Details:

This enum specifies the input parameter of Pwm_SetClockMode

Note

Only two types of prescalers can be used

Implements: Pwm_PrescalerType_enumeration

Table 3-184. Enumeration Pwm_PrescalerType Values

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

3.6.2.5 Enumeration Pwm_EdgeNotificationType

Edge notification type.

Details:

This enum defines the type of edge tranzition that can generate a notification

Implements: Pwm_EdgeNotificationType_enumeration

Table 3-185. Enumeration Pwm_EdgeNotificationType Values

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

3.6.2.6 Enumeration Pwm_StateType

Details:

Parameter used to Defines state of PWM channel

Implements: Pwm_StateType_enumeration

Table 3-186. Enumeration Pwm_StateType Values

Name	Initializer	Description
PWM_ACTIVE	0	The PWM pin will be in the same state of configured polarity.

Table continues on the next page...

Table 3-186. Enumeration Pwm_StateType Values (continued)

Name	Initializer	Description
PWM_INACTIVE	1	The PWM pin will be in the opposite state of configured polarity.

3.6.3 Function Reference

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

3.6.3.1 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);`

3.6.3.2 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 3-187. Pwm_DisableNotification Arguments

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

3.6.3.3 Function Pwm_EnableNotification

This function enables the user notifications.

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 3-188. Pwm_EnableNotification Arguments

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

3.6.3.4 Function Pwm_EndValidateChannelConfigCall

Completes the execution of a function impacting the configuration of a driver channel.

Details:

Performs actions in order to ensure that after it's caller has completed the execution the driver will remain in a state allowing execution of other functions updating the configuration of the entire driver or of a single channel.

Return: void.

Note

.

Prototype: `LOCAL_INLINE void Pwm_EndValidateChannelConfigCall(Pwm_ChannelType ChannelNumber);`

3.6.3.5 Function Pwm_EndValidateGlobalConfigCall

Completes the execution of a function impacting the configuration of the entire driver.

Details:

Performs actions in order to ensure that after it's caller has completed the execution the driver will remain in a state allowing execution of other functions updating the configuration of the entire driver or of a single channel

Return: void.

Note

.

Prototype: `LOCAL_INLINE void Pwm_EndValidateGlobalConfigCall(Std_ReturnType ValidCall, uint8 ServiceId);`

Table 3-189. Pwm_EndValidateGlobalConfigCall Arguments

Type	Name	Direction	Description
uint8	ServiceId	**input**	The service id of the caller function.
Std_ReturnType	ValidCall	**input**	The function call was previously validated.

3.6.3.6 Function Pwm_GetChannelState

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

Details:

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

Return: uint16 - DutyCycle of the requested channel.

Implements: Pwm_GetChannelState_Activity

Prototype: `uint16 Pwm_GetChannelState(Pwm_ChannelType ChannelNumber);`

Table 3-190. Pwm_GetChannelState Arguments

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

3.6.3.7 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

Note

Due to hardware limitation this function will always return PWM_LOW

Prototype: `Pwm_OutputStateType Pwm_GetOutputState(Pwm_ChannelType ChannelNumber);`

Table 3-191. Pwm_GetOutputState Arguments

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

Table 3-192. 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.

3.6.3.8 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 *versioninfo);`

Table 3-193. Pwm_GetVersionInfo Arguments

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

3.6.3.9 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.

For pre-compile and link time configuration variants, a NULL pointer shall be passed to the initialization routine. In this case the check for this NULL pointer has to be omitted.

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

Return: void.

Implements: Pwm_Init_Activity

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

Table 3-194. Pwm_Init Arguments

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

3.6.3.10 Function Pwm_Notification

Pwm_Notification.

Details:

This function is called from Pwm_IPW.c file in order to forward a channel notification call from the IP configuration.

Return: Void.

Prototype: `void Pwm_Notification(Pwm_ChannelType Channel);`

Table 3-195. Pwm_Notification Arguments

Type	Name	Direction	Description
`Pwm_ChannelType`	Channel		- Hw channel for which notification should be called.

3.6.3.11 Function Pwm_SetDutyCycle

This function sets the dutycycle 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 DutyCycle);`

Table 3-196. Pwm_SetDutyCycle Arguments

Type	Name	Direction	Description
`Pwm_ChannelType`	ChannelNumber	**input**	Pwm channel id.
uint16	DutyCycle	**input**	Pwm dutycycle value 0x0000 for 0% ... 0x8000 for 100%.

3.6.3.12 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 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 3-197. Pwm_SetOutputToIdle Arguments

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

3.6.3.13 Function Pwm_SetPeriodAndDuty

This function sets the period and the dutycycle 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 DutyCycle);`

Table 3-198. Pwm_SetPeriodAndDuty Arguments

Type	Name	Direction	Description
`Pwm_ChannelType`	ChannelNumber	**input**	- pwm channel id.
`Pwm_PeriodType`	Period	**input**	- pwm signal period value.
uint16	DutyCycle	**input**	- pwm dutycycle value 0x0000 for 0% ... 0x8000 for 100%.

3.6.3.14 Function Pwm_SetCounterBus

This function will change the bus of pwm channels running.

Details:

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

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_COUNTERBUS if the u32Bus is not correct.

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.

- 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).

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_SetCounterBus_Activity

Prototype: `void Pwm_SetCounterBus(Pwm_ChannelType ChannelNumber, uint32 Bus);`

Table 3-199. Pwm_SetCounterBus Arguments

Type	Name	Direction	Description
`Pwm_ChannelType`	ChannelNumber	**input**	- pwm channel id.
`uint32`	u32Bus	**input**	- pwm bus counter.

3.6.3.15 Function Pwm_SetChannelOutput

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

Details:

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

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).

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_SetChannelOutput(Pwm_ChannelType ChannelNumber, Pwm_StateType nState);`

Table 3-200. Pwm_SetChannelOutput Arguments

Type	Name	Direction	Description
`Pwm_ChannelType`	ChannelNumber	**input**	- pwm channel id.
`Pwm_StateType`	nState	**input**	- Active/Inactive state of the channel.

3.6.3.16 Function Pwm_BufferTransferEnableDisable

This function enable or diable the buffer transfer register to synchronize multiple PWM channels.

Details:

Implementation specific function to enable/disable the buffer transfer..

If development error detection for the Pwm module is enabled, the PWM functions shall check the module id and raise development error PWM_E_PARAM_INSTANCE if the module id is more than the number of module that supported by this platform.

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).

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_BufferTransferEnableDisable_Activity

Prototype: `void Pwm_BufferTransferEnableDisable(uint8 u8ModuleIndex, uint32 u32ChannelMasks);`

Table 3-201. Pwm_BufferTransferEnableDisable Arguments

Type	Name	Direction	Description
`uint8`	u8ModuleIndex	**input**	- module id.
`uint32`	u32ChannelMasks	**input**	- channel mask.

3.6.3.17 Function Pwm_SetClockMode

Pwm_SetClockMode.

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 3-202. Pwm_SetClockMode Arguments

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

3.6.3.18 Function Pwm_ValidateChannelConfigCall

Validate the call of a function impacting the configuration of one of the driver's.

Details:

channels. Before executing, a function which changes the configuration of a channel shall check if: 1. It is not preempting itself 2. It is not preempting a function that changes the configuration of the entire driver In any of the above cases, the function will report an error to Det or Serr, depending on the environment the driver is run in.

Return: Std_ReturnType Call is valid or not.

Note

.

Prototype: `Std_ReturnType Pwm_ValidateChannelConfigCall(Pwm_ChannelType ChannelNumber, uint8 ServiceId);`

Table 3-203. Pwm_ValidateChannelConfigCall Arguments

Type	Name	Direction	Description
	ChannelId	**input**	Id of the channel the caller tries to update.
uint8	ServiceId	**input**	Id of the service calling this function.

Table 3-204. Pwm_ValidateChannelConfigCall Return Values

Name	Description
E_OK	Caller of the function can continue its execution.
E_NOT_OK	Caller of the function should drop execution.

3.6.3.19 Function Pwm_ValidateGlobalConfigCall

Validate the call of a function impacting the configuration of the entire driver.

Details:

Before executing, a function which changes the configuration of the entire driver shall check if: 1. It is not preempting itself 2. It is not preempting another function that changes the configuration of the entire driver 3. It is not preempting a function that changes the configuration of one of the driver's channels In any of the above cases, the function will report an error to Det or Serr, depending on the environment the driver is run in.

Return: Std_ReturnType Call is valid or not.

Note

.

Prototype: `Std_ReturnType Pwm_ValidateGlobalConfigCall(uint8 ServiceId);`

Table 3-205. Pwm_ValidateGlobalConfigCall Arguments

Type	Name	Direction	Description
uint8	ServiceId	**input**	Id of the service calling this function.

Table 3-206. Pwm_ValidateGlobalConfigCall Return Values

Name	Description
E_OK	Caller of the function can continue its execution.
E_NOT_OK	Caller of the function should drop execution.

3.6.3.20 Function Pwm_ValidateParamDuty

Validate the DutyCycle parameter of the Pwm_SetDutyCycle API.In case an error is detected, the function will report it to Det or Serr, depending on the environment the driver is run in.

Return: Std_ReturnType Validity of the DutyCycle parameter.

Note

.

Prototype: `LOCAL_INLINE uint8 Pwm_ValidateParamDuty(uint16 DutyCycle);`

Table 3-207. Pwm_ValidateParamDuty Arguments

Type	Name	Direction	Description
uint16	DutyCycle	**input**	DutyCycle value to be validated.

Table 3-208. Pwm_ValidateParamDuty Return Values

Name	Description
E_OK	DutyCycle is valid.
E_NOT_OK	DutyCycle is invalid.

3.6.3.21 Function Pwm_ValidateParamOffsetDuty

Validate the offset (phase-shift) value parameter of the Pwm_SetDutyCycle API. The offset should be less then the channel period. In case an error is detected, the function will report it to Det or Serr, depending on the environment the driver is run in.

Return: Std_ReturnType Validity of the Offset parameter.

Note

.

Prototype: `LOCAL_INLINE uint8 Pwm_ValidateParamOffsetDuty(Pwm_ChannelType ChannelNumber, uint16 DutyCycle, const Pwm_IpConfigType * pIpConfig);`

Table 3-209. Pwm_ValidateParamOffsetDuty Arguments

Type	Name	Direction	Description
Pwm_ChannelType	ChannelNumber	**input**	Index of the given Pwm Channel.
uint16	DutyCycle	**input**	DutyCycle value to be validated.
Pwm_IpConfigType *	pIpConfig	**input**	pointer to configuration structure of the low-level driver.

Table 3-210. Pwm_ValidateParamDuty Return Values

Name	Description
E_OK	DutyCycle is valid.
E_NOT_OK	DutyCycle is invalid.

3.6.3.22 Function Pwm_ValidateParamNotification

Validate the notification handler of a Pwm channel. In order to be valid, the notification handler should not be NULL. In case it is NULL, the function will report an error to Det or Serr,depending on the environment the driver is run in.

Return: Std_ReturnType Validity of notification handler.

Note

.

Prototype: `LOCAL_INLINE uint8 Pwm_ValidateParamNotification(Pwm_NotifyType pPwmChannelNotification);`

Table 3-211. Pwm_ValidateParamNotification Arguments

Type	Name	Direction	Description
`Pwm_NotifyType`	pPwmChannelNotification	**input**	Notification of the channel to be validated.

Table 3-212. Pwm_ValidateParamNotification Return Values

Name	Description
E_OK	Notification handler is valid.
E_NOT_OK	Notification handler is not valid.

3.6.3.23 Function Pwm_ValidateParamPtrInit

Validate the configuration parameter of the Pwm_Init API. The check is required only in variant Post-Build, where the pointer should not be NULL.In case an error is detected, the function will report an error to Det or Serr, depending on the environment the driver is run in.

Return: Std_ReturnType Validity of the pointer.

Note

.

Prototype: `LOCAL_INLINE uint8 Pwm_ValidateParamPtrInit(const Pwm_ConfigType *ConfigPtr);`

Table 3-213. Pwm_ValidateParamPtrInit Arguments

Type	Name	Direction	Description
const`Pwm_ConfigType`*	ConfigPtr	**input**	Pointer to the configuration the driver is to be init with.

Table 3-214. Pwm_ValidateParamPtrInit Return Values

Name	Description
E_OK	Pointer is valid.
E_NOT_OK	Pointer is invalid.

3.6.3.24 Function Pwm_ValidateParamsPeriodDuty

Validate the Period and DutyCycle parameters of the Pwm_SetPeriodAndDuty API.The Period is validated in the sense that it can be updated only for channels having Variable Period class. In case any of the parameters is invalid, the function will report an error to Det or Serr,depending on the environment the driver is run in.

Return: Std_ReturnType Validity of Channel Class and DutyCycle parameter.

Note

.

Prototype: `LOCAL_INLINE uint8 Pwm_ValidateParamsPeriodDuty(Pwm_ChannelClassType ChannelClass, uint16 DutyCycle);`

Table 3-215. Pwm_ValidateParamsPeriodDuty Arguments

Type	Name	Direction	Description
`Pwm_ChannelClassType`	ChannelClass	**input**	Class of the channel to be validated.
uint16	DutyCycle	**input**	DutyCycle value to be validated.

Table 3-216. Pwm_ValidateParamsPeriodDuty Return Values

Name	Description
E_OK	Channel Class and Duty are both valid.
E_NOT_OK	One of or both Channel Class and DutytCycle are invalid.

3.6.3.25 Function PwmChannel_0_notification

Prototypes of Pwm channels User Notifications.

Prototype: `void PwmChannel_0_notification(void);`

3.6.3.26 Function PwmChannel_1_notification

Prototype: `void PwmChannel_1_notification(void);`

3.6.3.27 Function Pwm_GetTargetPowerState

Get the target power state of the Pwm HW unit.

Details:

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

If development error detection for the Pwm module is enabled, when a development error occurs, the corresponding PWM function shall:

- Report the error to the Development Error Tracer.
- Skip the desired functionality in order to avoid any corruptions of data or hardware registers (this means leave the function without any actions).

If the 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: Std_ReturnType

Implements: Pwm_GetTargetPowerState_Activity

Prototype: Std_ReturnType Pwm_GetTargetPowerState (Pwm_PowerStateType* pTargetPowerState, Pwm_PowerStateRequestResultType* pResult)

Table 3-217. Pwm_SetPowerState Arguments

Type	Name	Direction	Description
`Pwm_PowerStateType*`	pTargetPowerState	**output**	- The Target power mode of the Pwm HW Unit is returned in this parameter.
`Pwm_PowerStateRequestResultType*`	pResult	**output**	- Pointer to a variable to store the result of this function

3.6.3.28 Function Pwm_GetCurrentPowerState

Get the current power state of the Pwm HW unit.

Details:

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

If development error detection for the Pwm module is enabled, when a development error occurs, the corresponding PWM function shall:

- Report the error to the Development Error Tracer.
- Skip the desired functionality in order to avoid any corruptions of data or hardware registers (this means leave the function without any actions).

If the 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: Std_ReturnType.

Implements: Pwm_GetCurrentPowerState_Activity

Prototype: Std_ReturnType Pwm_GetCurrentPowerState (Pwm_PowerStateType* pCurrentPowerState, Pwm_PowerStateRequestResultType* pResult)

Table 3-218. Pwm_SetPowerState Arguments

Type	Name	Direction	Description
Pwm_PowerStateType*	pCurrentPowerState,	**ouput**	- The current power mode of the Pwm HW Unit is returned in this parameter
Pwm_PowerStateRequestResultType*	pResult	**output**	- Pointer to a variable to store the result of this function

3.6.3.29 Function Pwm_PreparePowerState

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

Details:

This API starts the needed process to allow the Pwm HW module to enter the requested power state..

If development error detection for the Pwm module is enabled, when a development error occurs, the corresponding PWM function shall:

- Report the error to the Development Error Tracer.
- Skip the desired functionality in order to avoid any corruptions of data or hardware registers (this means leave the function without any actions).

If the 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: Std_ReturnType.

Implements: Pwm_PreparePowerState_Activity

Prototype: Std_ReturnType Pwm_PreparePowerState (Pwm_PowerStateType nPowerState, Pwm_PowerStateRequestResultType* pResult)

Table 3-219. Pwm_SetPowerState Arguments

Type	Name	Direction	Description
`Pwm_PowerStateType`	nPowerState	**input**	- The target power state intended to be attained.
`Pwm_PowerStateRequestResultType*`	pResult	**output**	- Pointer to a variable to store the result of this function

3.6.3.30 Function Pwm_SetPowerState

Function to enters the already prepared power state.

Details:

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

If development error detection for the Pwm module is enabled, when a development error occurs, the corresponding PWM function shall:

- Report the error to the Development Error Tracer.
- Skip the desired functionality in order to avoid any corruptions of data or hardware registers (this means leave the function without any actions).

If the 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: Std_ReturnType.

Implements: Pwm_SetPowerState_Activity

Prototype: Std_ReturnType Pwm_SetPowerState (Pwm_PowerStateRequestResultType* pResult)

Table 3-220. Pwm_SetPowerState Arguments

Type	Name	Direction	Description
`Pwm_PowerStateRequestResultType*`	pResult	**output**	- Pointer to a variable to store the result of this function

3.6.4 Structs Reference

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

3.6.4.1 Structure Pwm_ChannelConfigType

Pwm channel high level configuration structure.

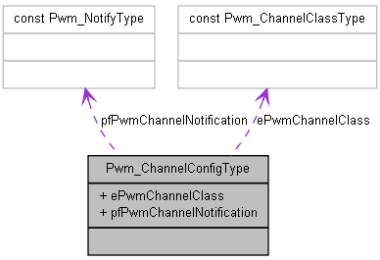

Figure 3-1. Struct Pwm_ChannelConfigType

Implements: Pwm_ChannelConfigType_struct

Declaration:

```
typedef struct
                          {
                            constPwm_ChannelClassType ePwmChannelClass
                          } Pwm_ChannelConfigType;
```

Table 3-221. Structure Pwm_ChannelConfigType member description

Member	Description
ePwmChannelClass	Channel class type: Variable/Fixed period.

3.6.4.2 Structure Pwm_ConfigType

Pwm high level configuration structure.

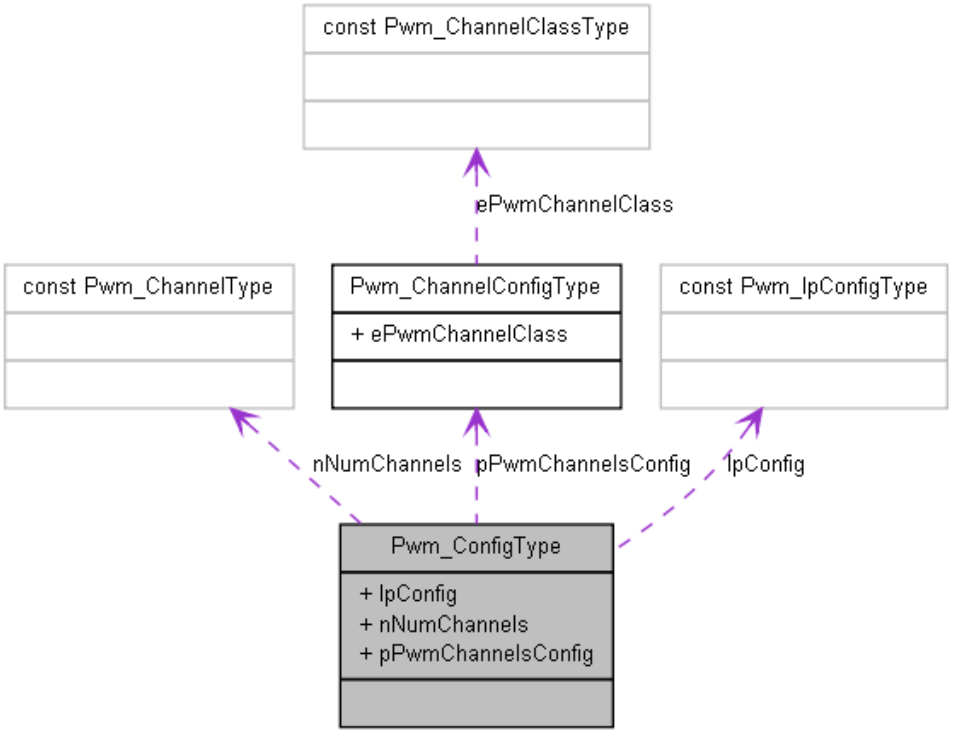

Figure 3-2. Struct Pwm_ConfigType

Implements: Pwm_ConfigType_struct

Declaration:

```
typedef struct
                        {
                          const Pwm_IpConfigType IpConfig,
                                  constPwm_ChannelType nNumChannels,
                                  constPwm_ChannelConfigType(* pPwmChannelsConfig)[]
                        } Pwm_ConfigType;
```

Table 3-222. Structure Pwm_ConfigType member description

Member	Description
IpConfig	Combined IP specific configuration structure.
nNumChannels	Number of Pwm configured channels.
pPwmChannelsConfig	Pointer to the list of Pwm configured channels.

3.6.4.3 Structure Pwm_eMios_ChannelConfigType

eMios IP channel specific configuration structure for the PWM functionality

Declaration:

```
typedef struct
                    {
                         constPwm_OutputStateType ePwmIdleState,
                        constPwm_OutputStateType ePwmPolarity,
                        constPwm_eMios_ControlType u32ControlValue,
                        constPwm_eMios_ChannelType u8HwChannel,
                        constPwm_PeriodType nPwmDefaultPeriod,
                        constuint16 u16PwmDefaultDutyCycle,
                         constPwm_PeriodType nPwmOffset,
                         constuint8 u8MasterMode,
                         constPwm_PeriodType nPwmDaocModulo,
                         constPwm_PeriodType nPwmTriggerDelay,
                         constboolean bPwmOffsetTriggerDelay,
                         constPwm_PeriodType nPwmDeadTime
                    } Pwm_eMios_ChannelConfigType;
```

Table 3-223. Structure Pwm_eMios_ChannelConfigType member description

Member	Description
ePwmIdleState	Pwm signal idle state: High or low.
ePwmPolarity	Pwm signal polarity: High or low.
u32ControlValue	eMios channel parameters
u8HwChannel	Assigned eMios channel id.
nPwmDefaultPeriod	Default value for period.
u16PwmDefaultDutyCycle	Default value for duty cycle: [0-0x8000] (0-100%).
nPwmOffset	Default value for pwm offset.
u8MasterMode	Default mode of Master bus.
nPwmDaocModulo	Default modulos of channel in DAOC mode.
nPwmTriggerDelay	Default trigger delay in OPWMT mode.
bPwmOffsetTriggerDelay	Default offset trigger delay.
nPwmDeadTime	Default dead time in OPWMCB mode.

3.6.4.4 Structure Pwm_eMios_IpConfigType

eMios IP specific configuration structure type

Declaration:

```
typedef struct
                         {
                           constPwm_eMios_ChannelType nNumChannels,
                                   constPwm_eMios_ChannelConfigType(* pChannelsConfig)[]
                         } Pwm_eMios_IpConfigType;
```

Table 3-224. Structure Pwm_eMios_IpConfigType member description

Member	Description
nNumChannels	Number of eMios channels in the Pwm configuration.
pChannelsConfig	Pointer to the configured channels for eMios.

3.6.4.5 Structure Pwm_IpChannelConfigType

Pwm channel high level configuration structure.

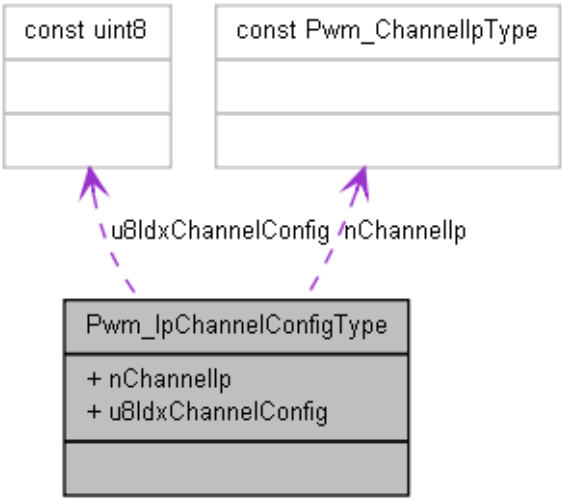

Figure 3-3. Struct Pwm_IpChannelConfigType

Declaration:

```
typedef struct
                        {
                          constPwm_ChannelIpType nChannelIp,
                                  const uint8 u8IdxChannelConfig
                        } Pwm_IpChannelConfigType;
```

Table 3-225. Structure Pwm_IpChannelConfigType member description

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

3.6.4.6 Structure Pwm_IpConfigType

Combined IP specific configuration structure.

Declaration:

```
typedef struct
                        {
                          constPwm_eMios_IpConfigType*const peMiosIpConfig,
                                  constPwm_IpChannelConfigType(* pIpChannelsConfig)[]
                        } Pwm_IpConfigType;
```

Table 3-226. Structure Pwm_IpConfigType member description

Member	Description
peMiosIpConfig	Pointer to the structure containing eMios configuration.
pIpChannelsConfig	Pointer to Array containing IP type and index in the IP configuration table for each Pwm channel.

3.6.5 Types Reference

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

3.6.5.1 Typedef Pwm_NotifyType

Channel notification typedef.

Details:

Pointer to notification handler

Type: void(*

3.6.5.2 Typedef Pwm_ChannelType

Pwm channel type.

Implements: Pwm_ChannelType_typedef

Type: uint8

3.6.5.3 Typedef Pwm_PeriodType

Channel period typedef.

Implements: Pwm_PeriodType_typedef

Type: uint16

3.6.5.4 Typedef Pwm_eMios_ChannelType

eMios HW module/channel id type

Type: uint8

3.6.5.5 Typedef Pwm_eMios_ControlType

eMios unified channel control register value

Type: uint32

3.6.5.6 Typedef Pwm_ChannelIpType

IP type used to implement a Pwm channel.

Type: uint8

3.6.6 Variables Reference

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

3.6.6.1 Variable Pwm_GlobalState

Variable storing the current state of the Pwm driver.

Declaration:

```
Pwm_GlobalStateType Pwm_GlobalState
```

3.6.6.2 Variable Pwm_pConfig

Pointer to the top level configuration structure - valid only when the driver is in the initialized state.

Declaration:

```
const Pwm_ConfigType* Pwm_pConfig
```

3.6.6.3 Variable Pwm_Channels_PB_0

Array of configured Pwm channels.

Declaration:

```
const Pwm_ChannelConfigType Pwm_Channels_PB_0[PWM_CONF_CHANNELS_PB_0]
```

3.6.6.4 Variable Pwm_eMios_ChannelConfig_PB_0

Details:

Configurations for Pwm channels using eMios

Declaration:

```
const Pwm_eMios_ChannelConfigType Pwm_eMios_ChannelConfig_PB_0[PWM_EMIOS_CONF_CHANNELS_PB_]
```

3.6.6.5 Variable Pwm_eMios_IpConfig_PB_0

eMios IP configuration.

Declaration:

```
const Pwm_eMios_IpConfigType Pwm_eMios_IpConfig_PB_0
```

3.6.6.6 Variable Pwm_IpChannelConfig_PB_0

Pwm channels IP related configuration array.

Declaration:

```
const Pwm_IpChannelConfigType Pwm_IpChannelConfig_PB_0[PWM_CONF_CHANNELS_PB_0]
```

3.6.6.7 Variable PwmChannelConfigSet_0

Pwm high level configuration structure.

Declaration:

```
const Pwm_ConfigType PwmChannelConfigSet_0
```

3.7 Symbolic Names Disclaimer

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

#define <Container_Short_Name> <Container_ID>

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

Chapter 4
Tresos Configuration Plug-in

This chapter describes the Tresos configuration plug-in for the PWM Driver. The most of the parameters are described below.

4.1 Configuration elements of Pwm

Included forms :

- IMPLEMENTATION_CONFIG_VARIANT
- PwmConfigurationOfOptApiServices
- PwmGeneral
- CommonPublishedInformation
- PwmChannelConfigSet

4.2 Form IMPLEMENTATION_CONFIG_VARIANT

- **VariantPreCompile**: Only precompile time configuration parameters.
- **VariantPostBuild**: Mix of precompile and postbuild time configuration parameters.

If Config Variant is set to

VariantPreCompile

, the files Pwm_Cfg.h and Pwm_Cfg.c should be used.

If Config Variant is set to

VariantPostBuild

, the files Pwm_Cfg.h and Pwm_PBcfg.c should be used.

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

Table 4-1. Attribute IMPLEMENTATION_CONFIG_VARIANT detailed description

Property	Value
Label	Config Variant
Default	VariantPreCompile
Range	VariantPreCompile VariantPostBuild

4.3 Form PwmConfigurationOfOptApiServices

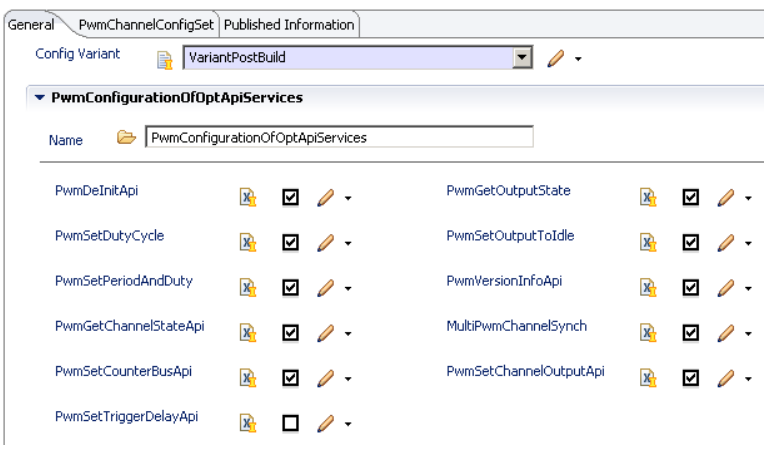

Figure 4-2. Tresos Plugin snapshot for PwmConfigurationOfOptApiServices form.

4.3.1 PwmDeInitApi (PwmConfigurationOfOptApiServices)

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

Table 4-2. Attribute PwmDeInitApi (PwmConfigurationOfOptApiServices) detailed description

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	true

4.3.2 PwmGetOutputState (PwmConfigurationOfOptApiServices)

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

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 4-3. Attribute PwmGetOutputState (PwmConfigurationOfOptApiServices) detailed description

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	false

4.3.3 PwmSetDutyCycle (PwmConfigurationOfOptApiServices)

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

Table 4-4. Attribute PwmSetDutyCycle (PwmConfigurationOfOptApiServices) detailed description

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	true

4.3.4 PwmSetOutputToIdle (PwmConfigurationOfOptApiServices)

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

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

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	true

4.3.5 PwmSetPeriodAndDuty (PwmConfigurationOfOptApiServices)

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

Table 4-6. Attribute PwmSetPeriodAndDuty (PwmConfigurationOfOptApiServices) detailed description

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	true

4.3.6 PwmVersionInfoApi (PwmConfigurationOfOptApiServices)

Switch to indicate that the Pwm_GetVersionInfo is supported

Table 4-7. Attribute PwmVersionInfoApi (PwmConfigurationOfOptApiServices) detailed description

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	true

4.3.7 PwmGetChannelStateApi (PwmConfigurationOfOptApiServices)

Switch to indicate that the PwmGetChannelStateApi is supported

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

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	false

4.3.8 PwmSetCounterBusApi (PwmConfigurationOfOptApiServices)

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

Table 4-9. Attribute PwmSetCounterBusApi (PwmConfigurationOfOptApiServices) detailed description

Property	Value
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	true

4.3.9 MultiPwmChannelSynch (PwmConfigurationOfOptApiServices)

This parameter is used to synchronize multiple pwm channels.

Table 4-10. Attribute MultiPwmChannelSynch (PwmConfigurationOfOptApiServices) detailed description

Property	Value
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

4.3.10 PwmSetChannelOutputApi (PwmConfigurationOfOptApiServices)

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

Table 4-11. Attribute PwmSetChannelOutputApi (PwmConfigurationOfOptApiServices) detailed description

Property	Value
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

4.3.11 PwmSetTriggerDelayApi (PwmConfigurationOfOptApiServices)

Adds / removes the services PwmSetTriggerDelay() from the code.This function is called when the prescaler value needs to be change to maintain same period at different frequency.

Table 4-12. Attribute PwmSetTriggerDelayApi (PwmConfigurationOfOptApiServices) detailed description

Property	Value
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

4.4 Form PwmGeneral

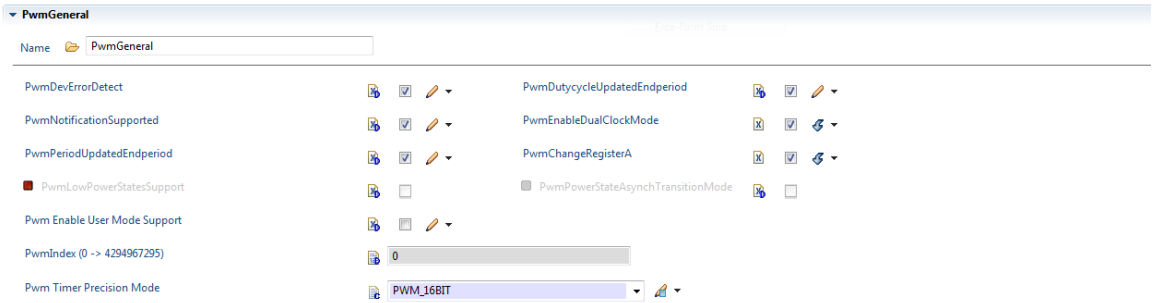

Figure 4-3. Tresos Plugin snapshot for PwmGeneral form.

4.4.1 PwmDevErorDetect (PwmGeneral)

Switch for enabling / disabling the development error detection.

Table 4-13. Attribute PwmDevErorDetect (PwmGeneral) detailed description

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	true

4.4.2 PwmDutycycleUpdatedEndperiod (PwmGeneral)

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

LIMITATION

In current implementation, there aren't any modes support update dutycycle immediate, this parameter should be TRUE

Table 4-14. Attribute PwmDutycycleUpdatedEndperiod (PwmGeneral) detailed description

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	true

4.4.3 PwmNotificationSupported (PwmGeneral)

Switch to indicate that the notifications are supported.

Table 4-15. Attribute PwmNotificationSupported (PwmGeneral) detailed description

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	true

4.4.4 PwmPeriodUpdatedEndperiod (PwmGeneral)

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

LIMITATION

In current implementation, there aren't any modes support update period immediate, this parameter should be TRUE

Table 4-16. Attribute PwmPeriodUpdatedEndperiod (PwmGeneral) detailed description

Property	Value
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	true

4.4.5 PwmEnableDualClockMode (PwmGeneral)

Switch for enabling the update of the dual clock mode.

LIMITATION

Curent implementation supports the dual clock mode.

Table 4-17. Attribute PwmEnableDualClockMode (PwmGeneral) detailed description

Property	Value
Type	BOOLEAN

Table continues on the next page...

Table 4-17. Attribute PwmEnableDualClockMode (PwmGeneral) detailed description (continued)

Property	Value
Origin	Custom
Symbolic Name	false
Default	false

4.4.6 PwmChangeRegisterA (PwmGeneral)

EN: If enabled for all OPWMT channels at dutycycle change register A will be updated instead of B. For OPWMT mode this means that the CTU trigger will maintain a fixed position relative to register B position. Note: This is an eMios implementation specific API.

LIMITATION

Table 4-18. Attribute PwmChangeRegisterA (PwmGeneral) detailed description

Property	Value
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Editable	false

4.4.7 PwmIndex (PwmGeneral)

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

LIMITATION

In the current implementation this parameter is not used.

Table 4-19. Attribute PwmIndex (PwmGeneral) detailed description

Property	Value
Type	INTEGER
Origin	AUTOSAR_ECUC
Symbolic Name	false

Table continues on the next page...

Table 4-19. Attribute PwmIndex (PwmGeneral) detailed description (continued)

Property	Value
Default	0
Invalid	Range <=4294967295 >=0

4.4.8 PwmEnableUserModeSupport(PwmGeneral)

Switch for enabling the user mode suport

LIMITATION

Table 4-20. Attribute PwmEnableUserModeSupport (PwmGeneral) detailed description

Property	Value
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

4.4.9 PwmTimerPrecision (PwmGeneral)

Select type of timer is 16bits or 24bits

Table 4-21. Attribute PwmTimerPrecision (PwmGeneral) detailed description

Property	Value
Type	String(Range)
ENUMERATION	PWM_16BIT

4.5 Form CommonPublishedInformation

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

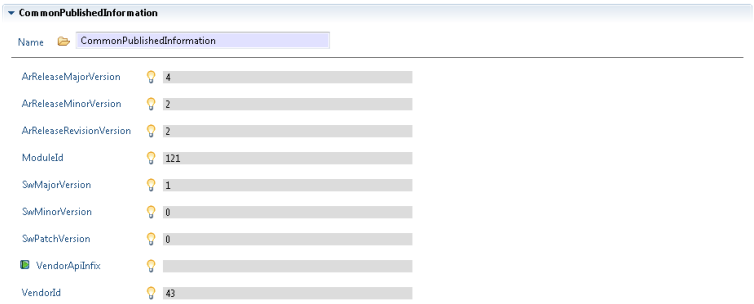

Figure 4-4. Tresos Plugin snapshot for CommonPublishedInformation form.

4.5.1 ArReleaseMajorVersion (CommonPublishedInformation)

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

Table 4-22. 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 4-23. Attribute ArReleaseMinorVersion (CommonPublishedInformation) detailed description

Property	Value
Label	AUTOSAR Minor Version
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false

Table continues on the next page...

Table 4-23. Attribute ArReleaseMinorVersion (CommonPublishedInformation) detailed description (continued)

Property	Value
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 4-24. 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 4-25. Attribute ModuleId (CommonPublishedInformation) detailed description

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

4.5.5 SwMajorVersion (CommonPublishedInformation)

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

Table 4-26. 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 4-27. 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 4-28. Attribute SwPatchVersion (CommonPublishedInformation) detailed description

Property	Value
Label	Software Patch Version
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false
Default	0
Invalid	Range >=0 <=0

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 4-29. 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 4-30. 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.

4.6.1 Form PwmChannel

Configuration of an individual PWM channel.

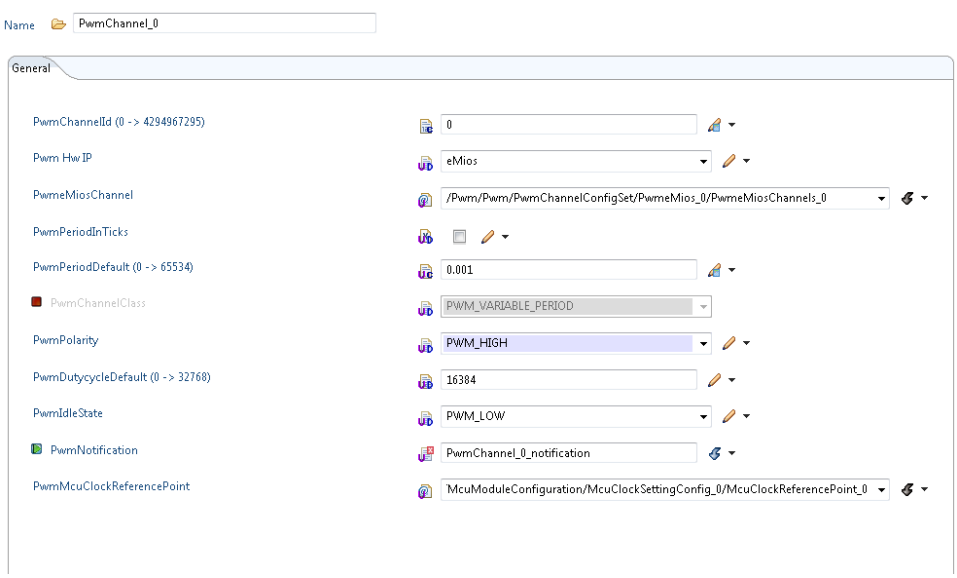

Figure 4-5. Tresos Plugin snapshot for PwmChannel form.

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 4-31. Attribute PwmChannelId (PwmChannel) detailed description

Property	Value
Type	INTEGER
Origin	AUTOSAR_ECUC
Symbolic Name	true
Invalid	Range <=4294967295 >=0

4.6.1.2 PwmHwIP (PwmChannel)

Hardware IP to be used for current PWM channel eMios - eMios Hardware IP will be used. Please select an eMios configured channel from eMiosChannel combo below

Table 4-32. Attribute PwmHwIP (PwmChannel) detailed description

Property	Value
Label	Pwm Hw IP
Type	ENUMERATION
Origin	Custom
Symbolic Name	false
Default	eMios
Range	eMios

4.6.1.3 PwmeMiosChannel (PwmChannel)

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

Table 4-33. Attribute PwmeMiosChannel (PwmChannel) detailed description

Property	Value
Type	REFERENCE
Origin	Custom

4.6.1.4 PwmPeriodInTicks (PwmChannel)

PwmPeriodInTicks By default Period unit is measured in Seconds. Enable this check to set Default Period unit in Ticks.

Table 4-34. Attribute PwmPeriodInTicks (PwmChannel) detailed description

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

4.6.1.5 PwmPeriodDefault (PwmChannel)

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

Table 4-35. Attribute PwmPeriodDefault (PwmChannel) detailed description

Property	Value
Label	Default Period
Type	FLOAT
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	0.0010
Invalid	Range <=65534 >=0

4.6.1.6 PwmChannelClass (PwmChannel)

Table 4-36. Attribute PwmChannelClass (PwmChannel) detailed description

Property	Value
Type	ENUMERATION
Origin	AUTOSAR_ECUC
Symbolic Name	false

Table continues on the next page...

Table 4-36. Attribute PwmChannelClass (PwmChannel) detailed description (continued)

Property	Value
Default	PWM_FIXED_PERIOD
Range	PWM_FIXED_PERIOD PWM_FIXED_PERIOD_SHIFTED PWM_VARIABLE_PERIOD

4.6.1.7 PwmPolarity (PwmChannel)

Defines the starting polarity of each PWM channel.

Table 4-37. Attribute PwmPolarity (PwmChannel) detailed description

Property	Value
Type	ENUMERATION
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	PWM_HIGH
Range	PWM_HIGH PWM_LOW

4.6.1.8 PwmDutycycleDefault (PwmChannel)

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

Table 4-38. Attribute PwmDutycycleDefault (PwmChannel) detailed description

Property	Value
Type	INTEGER
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	16384
Invalid	Range <=32768 >=0

4.6.1.9 PwmIdleState (PwmChannel)

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

Table 4-39. Attribute PwmIdleState (PwmChannel) detailed description

Property	Value
Type	ENUMERATION
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	PWM_LOW
Range	PWM_HIGH PWM_LOW

4.6.1.10 PwmNotification (PwmChannel)

User callback function NOTE: please use NULL or NULL_PTR w/o any quotes. If the used string is different from NULL or NULL_PTR it will be used as the configured function name.

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

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

4.6.1.11 PwmMcuClockReferencePoint (PwmChannel)

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

Motor Control Clock configuration in AuxClock0En tab is the reference clock for eMios and CTU

Table 4-41. Attribute PwmMcuClockReferencePoint (PwmChannel) detailed description

Property	Value
Type	REFERENCE
Origin	AUTOSAR_ECUC

4.6.2 Form PwmeMios

Configuration of a eMios module available on the platfom.

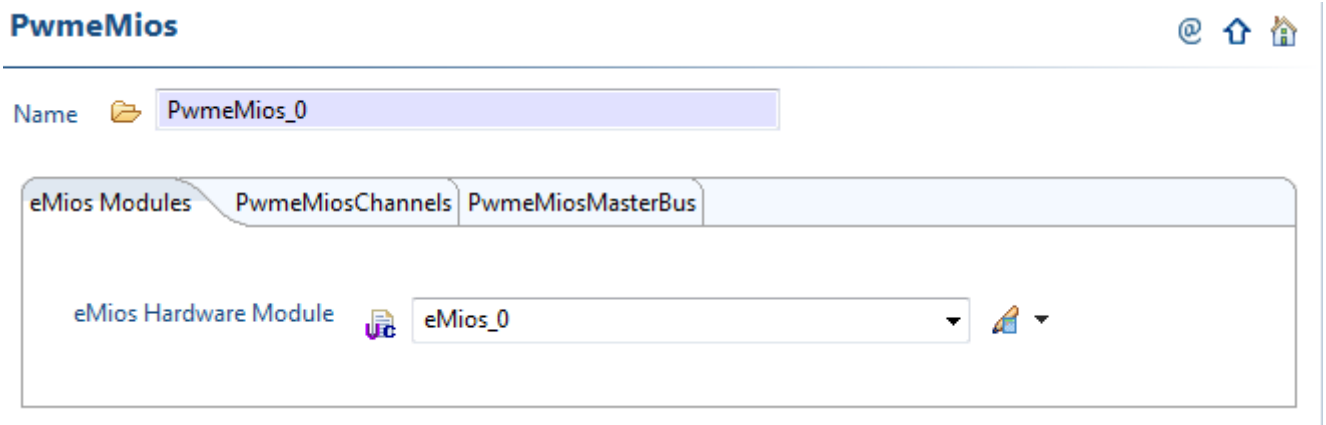

Figure 4-6. Tresos Plugin snapshot for PwmeMios form.

4.6.2.1 PwmeMiosModule (PwmeMios)

Select the physical eMios Module.

Table 4-42. Attribute PwmeMiosModule (PwmeMios) detailed description

Property	Value
Label	eMios Hardware Module
Type	ENUMERATION
Origin	Custom
Symbolic Name	false

4.6.2.2 Form PwmeMiosChannels

List of eMios channels available on the platform.

Is included by form : Form PwmeMios

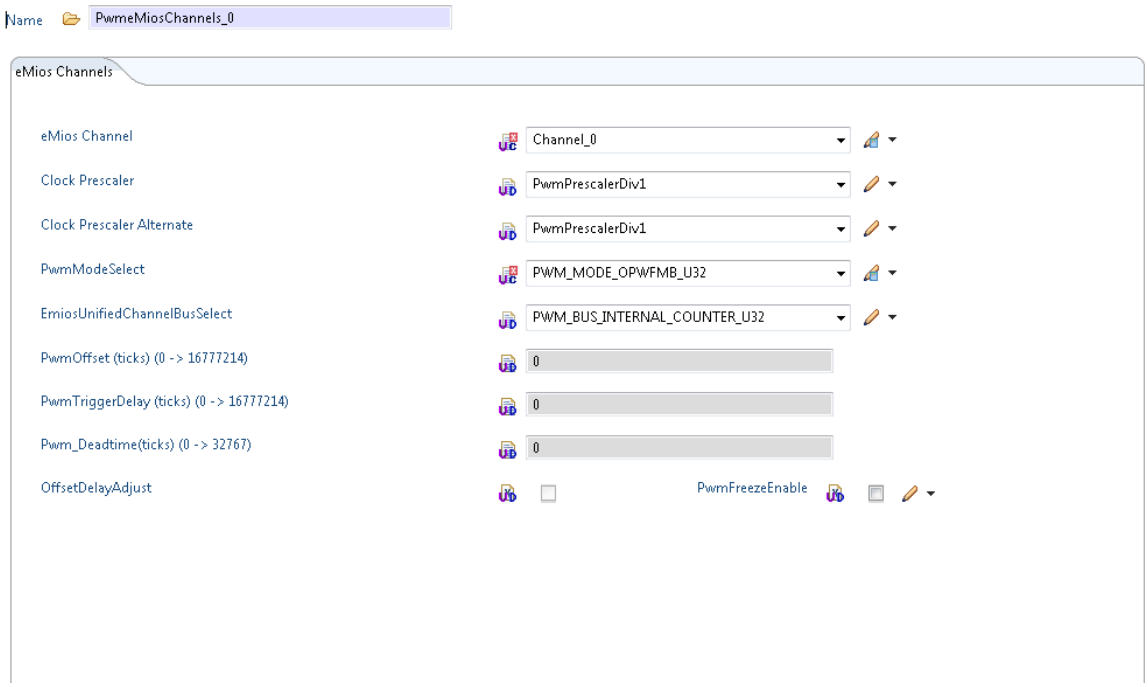

Figure 4-7. Tresos Plugin snapshot for PwmeMiosChannels form.

4.6.2.2.1 PwmeMiosChannel (PwmeMiosChannels)

Selects one of the eMios channels available on the platform.

Table 4-43. Attribute PwmeMiosChannel (PwmeMiosChannels) detailed description

Property	Value
Label	eMios Channel
Type	ENUMERATION
Origin	Custom
Symbolic Name	false
Default	Channel_0
Range	Channel_0 Channel_1 Channel_2 Channel_3 Channel_4 Channel_5 Channel_6 Channel_7 Channel_8 Channel_9 Channel_10 Channel_11 Channel_12 Channel_13 Channel_14 Channel_15

Table 4-43. Attribute PwmeMiosChannel (PwmeMiosChannels) detailed description

Property	Value
	Channel_16 Channel_17 Channel_18 Channel_19 Channel_20 Channel_21 Channel_22 Channel_23 Channel_24 Channel_25 Channel_26 Channel_27 Channel_28 Channel_29 Channel_30 Channel_31

4.6.2.2.2 PwmeMiosClock (PwmeMiosChannels)

Select input count source (clock) used for this eMios channel

Table 4-44. Attribute PwmeMiosClock (PwmeMiosChannels) detailed description

Property	Value
Label	Clock Source and Prescaler
Type	ENUMERATION
Origin	Custom
Default	PwmPrescalerDiv1_U32
Range	PwmPrescalerDiv1_U32 PwmPrescalerDiv2_U32 PwmPrescalerDiv3_U32 PwmPrescalerDiv4_U32

4.6.2.2.3 PwmeMiosClock_Alternate (PwmeMiosChannels)

Alternative prescaler and clock source used when dual clock mode is enabled.

Table 4-45. Attribute PwmeMiosClock_Alternate (PwmeMiosChannels) detailed description

Property	Value
Label	Clock Source and Prescaler
Type	ENUMERATION
Origin	Custom
Default	PwmPrescalerDiv1_U32

Table continues on the next page...

Table 4-45. Attribute PwmeMiosClock_Alternate (PwmeMiosChannels) detailed description (continued)

Property	Value
Range	PwmPrescalerDiv1_U32 PwmPrescalerDiv2_U32 PwmPrescalerDiv3_U32 PwmPrescalerDiv4_U32

4.6.2.2.4 PwmModeSelect (PwmeMiosChannels)

Selects one of the operation mode available on the platform.

Table 4-46. Attribute PwmModeSelect (PwmeMiosChannels) detailed description

Property	Value
Label	Mode selection
Type	ENUMERATION
Origin	Custom
Default	PWM_MODE_OPWFMB_U32
Range	PWM_MODE_OPWFMB_U32 PWM_MODE_OPWMB_U32 PWM_MODE_OPWMT_U32 PWM_MODE_OPWMCB_LEAD_DEADTIME_U32 PWM_MODE_OPWMCB_TRAIL_DEADTIME_U32 PWM_MODE_DAOC_U32

4.6.2.2.5 EmiosUnifiedChannelBusSelect (PwmeMiosChannels)

Selects one of the counter bus available on the platform.

Table 4-47. Attribute EmiosUnifiedChannelBusSelect (PwmeMiosChannels) detailed description

Property	Value
Label	Bus selection
Type	ENUMERATION
Origin	Custom
Default	PWM_BUS_INTERNAL_COUNTER_U32
Range	PWM_BUS_A_U32 PWM_BUS_F_U32 PWM_BUS_DIVERSE_U32 PWM_BUS_INTERNAL_COUNTER_U32

4.6.2.2.6 PwmOffset (PwmeMiosChannels)

Default value for Pwm offset.

Table 4-48. Attribute PwmOffset (PwmeMiosChannels) detailed description

Property	Value
Label	Pwm offset selection
Type	INTEGER
Origin	Custom
Default	0
Invalid	Range <=65534 >=0

4.6.2.2.7 PwmTriggerDelay (PwmeMiosChannels)

Default value for Pwm trigger delay.

Table 4-49. Attribute PwmTriggerDelay (PwmeMiosChannels) detailed description

Property	Value
Label	Pwm TriggerDelay selection
Type	INTEGER
Origin	Custom
Default	0
Invalid	Range <=65534 >=0

4.6.2.2.8 Pwm_Deadtime (PwmeMiosChannels)

Default value for Pwm dead time.

Table 4-50. Attribute Pwm_Deadtime (PwmeMiosChannels) detailed description

Property	Value
Label	Pwm dead time selection
Type	INTEGER
Origin	Custom
Default	0
Invalid	Range <=65534 >=0

4.6.2.2.9 OffsetDelayAdjust (PwmeMiosChannels)

Enable/Disable offset delay

Table 4-51. Attribute OffsetDelayAdjust (PwmeMiosChannels) detailed description

Property	Value
Label	Enable/Disable offset delay
Type	BOOLEAN
Origin	Custom
Default	false

4.6.2.3 Form PwmeMiosMasterBus

List of eMios master bus available on the platform.

Is included by form : Form PwmeMios

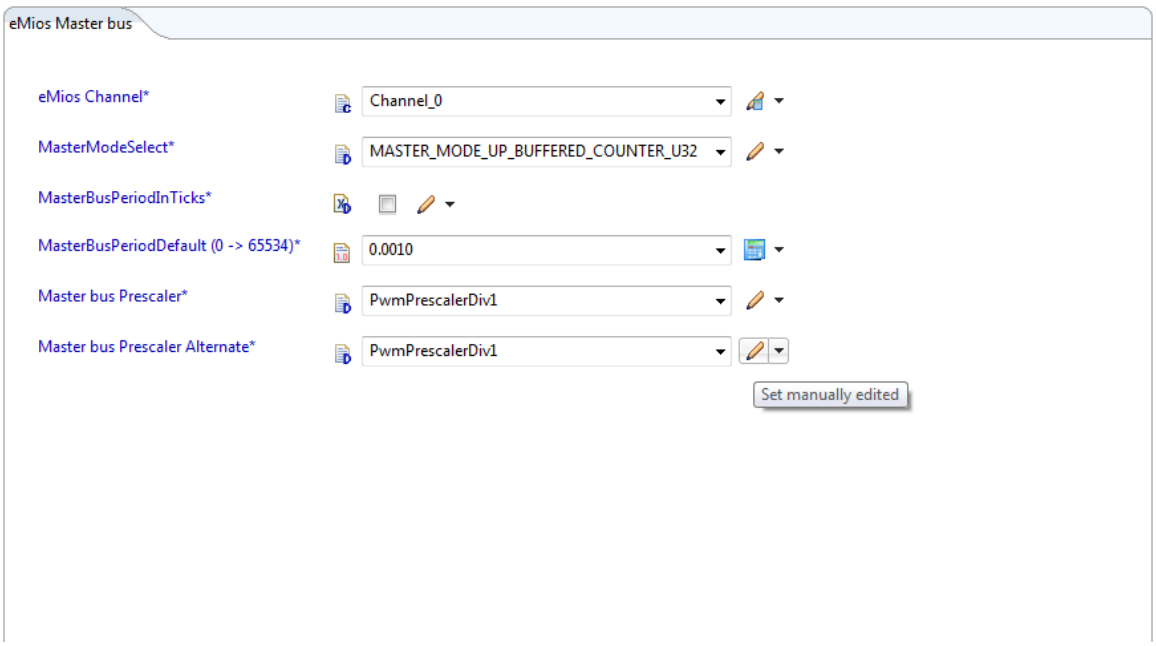

Figure 4-8. Tresos Plugin snapshot for PwmeMiosMasterBus form.

4.6.2.3.1 PwmeMiosMasterBus (PwmeMiosMasterBus)

Selects one of the eMios master bus available on the platform.

Table 4-52. Attribute PwmeMiosMasterBus (PwmeMiosMasterBus) detailed description

Property	Value
Label	eMios MasterBus
Type	ENUMERATION
Origin	Custom
Symbolic Name	false
Default	Channel_0
Range	Channel_0 Channel_8 Channel_16 Channel_22 Channel_23 Channel_24

4.6.2.3.2 MasterBusPrescaler (PwmeMiosMasterBus)

Select input count source (clock) used for this master bus channel

Table 4-53. Attribute MasterBusPrescaler (PwmeMiosMasterBus) detailed description

Property	Value
Label	Clock Source and Prescaler
Type	ENUMERATION
Origin	Custom
Default	PwmPrescalerDiv1_U32
Range	PwmPrescalerDiv1_U32 PwmPrescalerDiv2_U32 PwmPrescalerDiv3_U32 PwmPrescalerDiv4_U32

4.6.2.3.3 MasterBusPrescaler_Alternate (PwmeMiosMasterBus)

Alternative prescaler and clock source used for this master bus when dual clock mode is enabled.

Table 4-54. Attribute MasterBusPrescaler_Alternate (PwmeMiosMasterBus) detailed description

Property	Value
Label	Clock Source and Prescaler
Type	ENUMERATION
Origin	Custom
Default	PwmPrescalerDiv1_U32

Table continues on the next page...

Table 4-54. Attribute MasterBusPrescaler_Alternate (PwmeMiosMasterBus) detailed description (continued)

Property	Value
Range	PwmPrescalerDiv1_U32 PwmPrescalerDiv2_U32 PwmPrescalerDiv3_U32 PwmPrescalerDiv4_U32

4.6.2.3.4 MasterModeSelect (PwmeMiosMasterBus)

Selects one of the operation mode for master bus available on the platform.

Table 4-55. Attribute MasterModeSelect (PwmeMiosMasterBus) detailed description

Property	Value
Label	Mode selection
Type	ENUMERATION
Origin	Custom
Default	MASTER_MODE_UP_BUFFERED_COUNTER_U32
Range	MASTER_MODE_UP_BUFFERED_COUNTER_U32 MASTER_MODE_UP_DOWN_BUFFERED_COUNTER_U32

4.6.2.3.5 PwmPeriodInTicks (PwmeMiosMasterBus)

PwmPeriodInTicks By default Period unit is measured in Seconds. Enable this check to set Default Period unit in Ticks.

Table 4-56. Attribute PwmPeriodInTicks (PwmChannel) detailed description

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

4.6.2.3.6 MasterBusPeriodDefault (PwmeMiosMasterBus)

Default value for master bus period

Table 4-57. Attribute MasterBusPeriodDefault (PwmeMiosMasterBus) detailed description

Property	Value
Label	Default value for master bus period
Type	INTEGER
Origin	Custom
Default	0
Invalid	Range <=65534 >=0

4.6.3 Form PwmHwIntrruptConfigList

List of HW interrupts available for the entire platform.

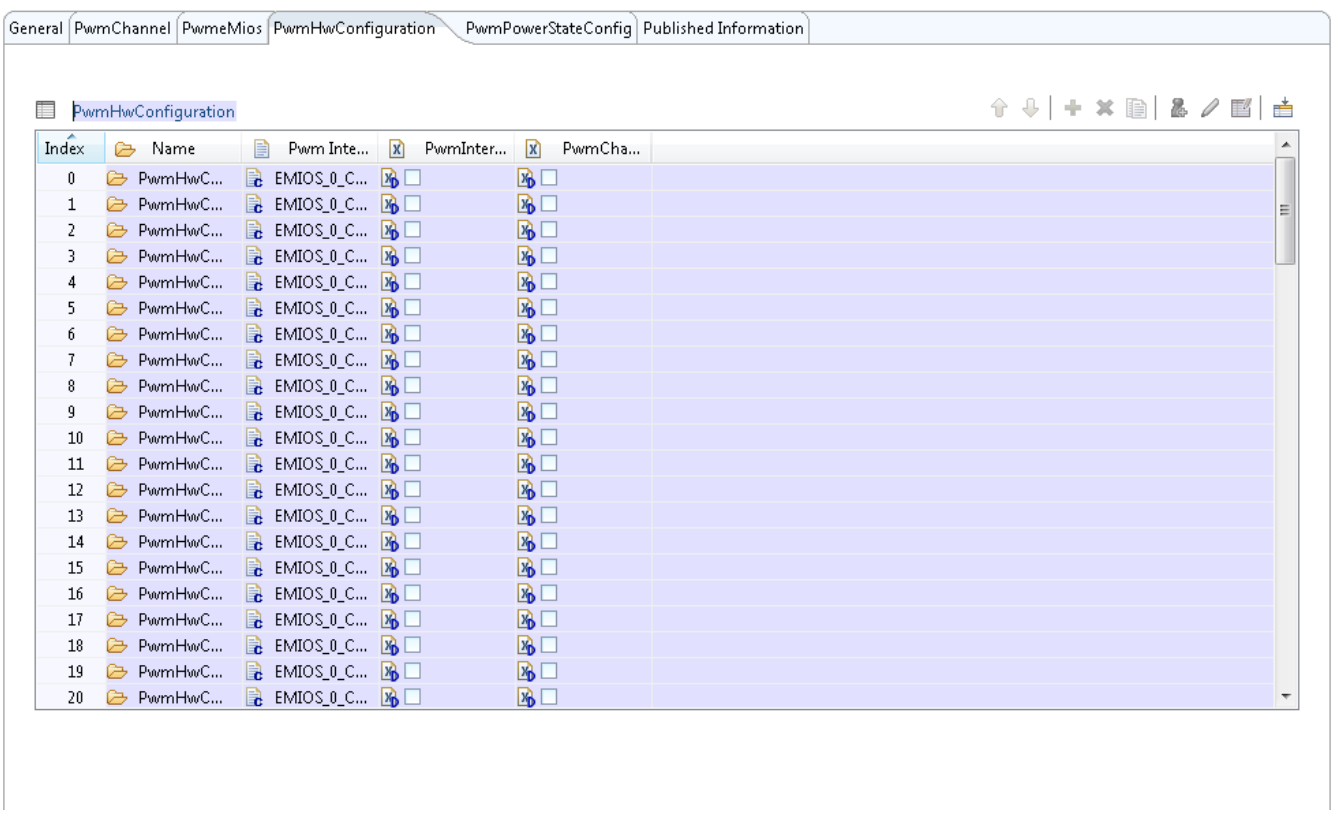

Figure 4-9. Tresos Plugin snapshot for PwmHwIntrruptConfigList form.

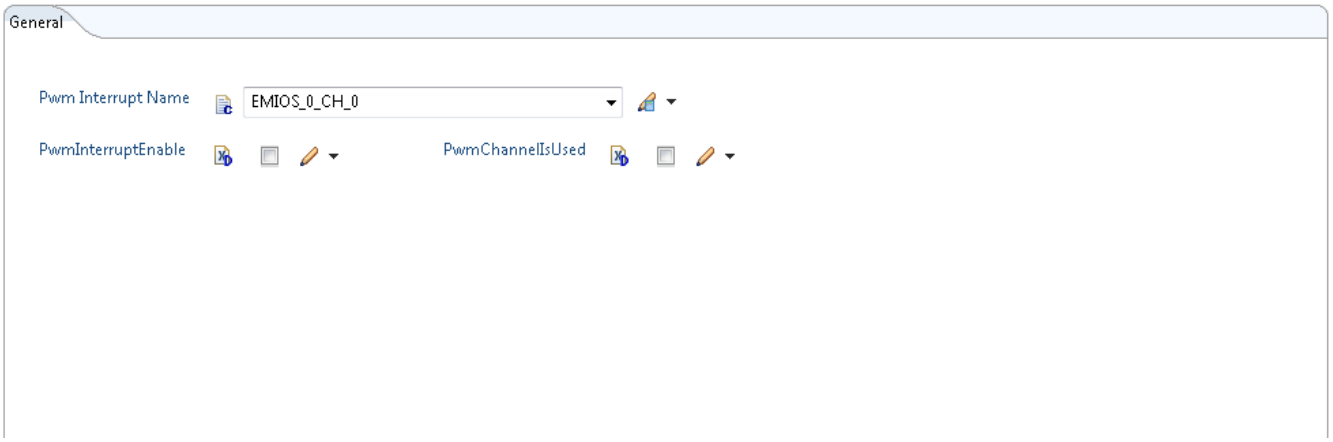

Figure 4-10. Tresos Plugin snapshot for PwmHwIntrruptConfigList_0 container.

4.6.3.1 PwmInterruptName(PwmHwIntrruptConfigList)

ID of HW interrupt resources.

Table 4-58. Attribute PwmInterruptName (PwmHwIntrruptConfigList) detailed description

Property	Value
Label	PWM Peripheral ISR Name
Type	String(Range)
Origin	Custom
Symbolic Name	false
Default	EMIOS_0_CH_0
Range	EMIOS_0_CH_0 EMIOS_0_CH_1 EMIOS_0_CH_2 EMIOS_0_CH_3 EMIOS_0_CH_4 EMIOS_0_CH_5 EMIOS_0_CH_6 EMIOS_0_CH_7 EMIOS_0_CH_8 EMIOS_0_CH_9 EMIOS_0_CH_10 EMIOS_0_CH_11 EMIOS_0_CH_12 EMIOS_0_CH_13 EMIOS_0_CH_14 EMIOS_0_CH_15 EMIOS_0_CH_16 EMIOS_0_CH_17 EMIOS_0_CH_18 EMIOS_0_CH_19 EMIOS_0_CH_20 EMIOS_0_CH_21 EMIOS_0_CH_22

Table 4-58. Attribute PwmInterruptName (PwmHwIntrruptConfigList) detailed description

Property	Value
	EMIOS_0_CH_23 EMIOS_0_CH_24 EMIOS_0_CH_25 EMIOS_0_CH_26 EMIOS_0_CH_27 EMIOS_0_CH_28 EMIOS_0_CH_29 EMIOS_0_CH_30 EMIOS_0_CH_31 EMIOS_1_CH_0 EMIOS_1_CH_1 EMIOS_1_CH_2 EMIOS_1_CH_3 EMIOS_1_CH_4 EMIOS_1_CH_5 EMIOS_1_CH_6 EMIOS_1_CH_7 EMIOS_1_CH_8 EMIOS_1_CH_9 EMIOS_1_CH_10 EMIOS_1_CH_11 EMIOS_1_CH_12 EMIOS_1_CH_13 EMIOS_1_CH_14 EMIOS_1_CH_15 EMIOS_1_CH_16 EMIOS_1_CH_17 EMIOS_1_CH_18 EMIOS_1_CH_19 EMIOS_1_CH_20 EMIOS_1_CH_21 EMIOS_1_CH_22 EMIOS_1_CH_23 EMIOS_1_CH_24 EMIOS_1_CH_25 EMIOS_1_CH_26 EMIOS_1_CH_27 EMIOS_1_CH_28 EMIOS_1_CH_29 EMIOS_1_CH_30 EMIOS_1_CH_31 EMIOS_2_CH_0 EMIOS_2_CH_1 EMIOS_2_CH_2 EMIOS_2_CH_3 EMIOS_2_CH_4 EMIOS_2_CH_5 EMIOS_2_CH_6 EMIOS_2_CH_7 EMIOS_2_CH_8 EMIOS_2_CH_9 EMIOS_2_CH_10 EMIOS_2_CH_11 EMIOS_2_CH_12 EMIOS_2_CH_13 EMIOS_2_CH_14

Table 4-58. Attribute PwmInterruptName (PwmHwIntrruptConfigList) detailed description

Property	Value
	EMIOS_2_CH_15 EMIOS_2_CH_16 EMIOS_2_CH_17 EMIOS_2_CH_18 EMIOS_2_CH_19 EMIOS_2_CH_20 EMIOS_2_CH_21 EMIOS_2_CH_22 EMIOS_2_CH_23 EMIOS_2_CH_24 EMIOS_2_CH_25 EMIOS_2_CH_26 EMIOS_2_CH_27 EMIOS_2_CH_28 EMIOS_2_CH_29 EMIOS_2_CH_30 EMIOS_2_CH_31

4.6.3.2 PwmInterruptEnable(PwmHwIntrruptConfigList)

Enable/Disable HW channels' Interrupt Sources.

Table 4-59. Attribute PwmInterruptEnable(PwmHwIntrruptConfigList) detailed description

Property	Value
Label	PwmInterruptEnable
Type	Boolean(Range)
Origin	Custom
Default	false
Range	true false

4.6.3.3 PwmChannelIsUsed (PwmHwIntrruptConfigList)

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

Table 4-60. Attribute PwmChannelIsUsed (PwmHwIntrruptConfigList) detailed description

Property	Value
Label	PwmChannelIsUsed
Type	Boolean(Range)
Origin	Custom

Table continues on the next page...

Table 4-60. Attribute PwmChannelIsUsed (PwmHwIntrruptConfigList) detailed description (continued)

Property	Value
Default	false
Range	true false

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Document Number UM35PWMASR4.2 Rev0002 R1.0.0
Revision 5.0.0