



Elektrobit

# EB tresos<sup>®</sup> AutoCore Generic 8 Watchdog Stack documentation

product release 8.8.0





Elektrobit Automotive GmbH  
Am Wolfsmantel 46  
91058 Erlangen, Germany  
Phone: +49 9131 7701 0  
Fax: +49 9131 7701 6333  
Email: [info.automotive@elektrobit.com](mailto:info.automotive@elektrobit.com)

## Technical support

<https://www.elektrobit.com/support>

## Legal disclaimer

Confidential information.

ALL RIGHTS RESERVED. No part of this publication may be copied in any form, by photocopy, microfilm, retrieval system, or by any other means now known or hereafter invented without the prior written permission of Elektrobit Automotive GmbH.

All brand names, trademarks, and registered trademarks are property of their rightful owners and are used only for description.

Copyright 2020, Elektrobit Automotive GmbH.

# Table of Contents

1. Overview of EB tresos AutoCore Generic 8 Watchdog Stack documentation .....	9
2. Supported features .....	10
3. ACG8 Watchdog Stack release notes .....	11
3.1. Overview .....	11
3.2. Scope of the release .....	11
3.2.1. Configuration tool .....	11
3.2.2. AUTOSAR modules .....	11
3.2.3. EB (Elektrobit) modules .....	11
3.2.4. MCAL modules and EB tresos AutoCore OS .....	12
3.3. Module release notes .....	12
3.3.1. WdgIf module release notes .....	12
3.3.1.1. Change log .....	12
3.3.1.2. New features .....	16
3.3.1.3. EB-specific enhancements .....	16
3.3.1.4. Deviations .....	16
3.3.1.5. Limitations .....	17
3.3.1.6. Open-source software .....	17
3.3.2. WdgM module release notes .....	17
3.3.2.1. Change log .....	17
3.3.2.2. New features .....	25
3.3.2.3. EB-specific enhancements .....	25
3.3.2.4. Deviations .....	28
3.3.2.5. Limitations .....	33
3.3.2.6. Open-source software .....	35
4. ACG8 Watchdog Stack user guide .....	36
4.1. Overview .....	36
4.2. Background information .....	36
4.2.1. Stack overview .....	36
4.2.2. Module dependencies .....	38
4.2.2.1. Development Error Detection (Det) .....	38
4.2.2.2. Diagnostic Event Manager (Dem) .....	38
4.2.2.3. Micro-Controller Unit (Mcu) .....	38
4.2.2.4. Runtime Environment (Rte) .....	38
4.2.2.5. Schedule Manager (SchM) .....	38
4.2.2.6. ECU State Manager (EcuM) .....	38
4.2.2.7. BSW Mode Manager (BswM) .....	39
4.2.3. Multi-core support .....	39
4.3. Initializing the watchdog stack .....	40
4.4. Configuring the watchdog stack .....	40

4.5. WdgIf module user guide .....	41
4.5.1. Configuring the WdgIf module .....	41
4.6. WdgM module user guide .....	42
4.6.1. Overview .....	42
4.6.2. Background information .....	42
4.6.2.1. State machines and interfaces .....	42
4.6.2.2. Concepts of supervision .....	43
4.6.2.3. Alive supervision of supervised entities .....	44
4.6.2.3.1. Supervision status state machines .....	44
4.6.2.3.2. Effects of mode switching on alive supervision .....	45
4.6.2.3.3. Example of the alive supervision .....	45
4.6.2.4. Triggering the watchdog .....	46
4.6.2.5. WdgM modes .....	46
4.6.2.6. Development error detection .....	48
4.6.2.7. Run-time error detection .....	50
4.6.2.8. Software component description .....	50
4.6.2.8.1. Data types .....	51
4.6.2.8.1.1. WdgMModeType .....	51
4.6.2.8.2. Ports .....	52
4.6.2.8.2.1. WdgMModePort .....	52
4.6.2.8.2.2. WdgMServicePortSE<nnn> .....	52
4.6.3. Configuring the WdgM module .....	52
4.6.3.1. Referencing all configured watchdogs .....	53
4.6.3.2. Adding all supervised entities .....	53
4.6.3.3. Adding the checkpoints for a supervised entity .....	54
4.6.3.4. Defining the initial and final checkpoints for logical supervision .....	55
4.6.3.5. Defining a configuration set .....	55
4.6.3.6. Configuring a mode .....	57
4.6.3.7. Configuring the WdgM for multi-core .....	58
5. ACG8 Watchdog Stack module references .....	60
5.1. Overview .....	60
5.1.1. Notation in EB module references .....	60
5.1.1.1. Default value of configuration parameters .....	60
5.1.1.2. Range information of configuration parameters .....	60
5.2. WdgIf .....	61
5.2.1. Configuration parameters .....	61
5.2.1.1. CommonPublishedInformation .....	62
5.2.1.2. PublishedInformation .....	65
5.2.1.3. WdgIfDevice .....	65
5.2.1.4. WdgIfGeneral .....	66
5.2.2. Application programming interface (API) .....	67
5.2.2.1. Type definitions .....	67

5.2.2.1.1. WdgIf_ModeType .....	67
5.2.2.2. Macro constants .....	68
5.2.2.2.1. WDGIF_AR_RELEASE_MAJOR_VERSION .....	68
5.2.2.2.2. WDGIF_AR_RELEASE_MINOR_VERSION .....	68
5.2.2.2.3. WDGIF_AR_RELEASE_REVISION_VERSION .....	68
5.2.2.2.4. WDGIF_E_INV_POINTER .....	68
5.2.2.2.5. WDGIF_E_PARAM_DEVICE .....	68
5.2.2.2.6. WDGIF_FAST_MODE .....	68
5.2.2.2.7. WDGIF_MODULE_ID .....	69
5.2.2.2.8. WDGIF_OFF_MODE .....	69
5.2.2.2.9. WDGIF_SID_GETVERSIONINFO .....	69
5.2.2.2.10. WDGIF_SID_SETMODE .....	69
5.2.2.2.11. WDGIF_SID_SETTRIGGERCOND .....	69
5.2.2.2.12. WDGIF_SLOW_MODE .....	69
5.2.2.2.13. WDGIF_SW_MAJOR_VERSION .....	70
5.2.2.2.14. WDGIF_SW_MINOR_VERSION .....	70
5.2.2.2.15. WDGIF_SW_PATCH_VERSION .....	70
5.2.2.2.16. WDGIF_VENDOR_ID .....	70
5.2.2.3. Functions .....	70
5.2.2.3.1. WdgIf_GetVersionInfo .....	70
5.2.2.3.2. WdgIf_SetMode .....	71
5.2.2.3.3. WdgIf_SetTriggerCondition .....	71
5.2.3. Integration notes .....	72
5.2.3.1. Exclusive areas .....	72
5.2.3.2. Production errors .....	72
5.2.3.3. Memory mapping .....	72
5.2.3.4. Integration requirements .....	72
5.3. WdgM .....	73
5.3.1. Configuration parameters .....	73
5.3.1.1. CommonPublishedInformation .....	74
5.3.1.2. PublishedInformation .....	77
5.3.1.3. WdgMDefensiveProgramming .....	77
5.3.1.4. ReportToDem .....	80
5.3.1.5. WdgMConfigSet .....	85
5.3.1.6. WdgMDemEventParameterRefs .....	86
5.3.1.7. WdgMMode .....	89
5.3.1.8. WdgMAliveSupervision .....	91
5.3.1.9. WdgMDeadlineSupervision .....	93
5.3.1.10. WdgMExternalLogicalSupervision .....	94
5.3.1.11. WdgMExternalTransition .....	95
5.3.1.12. WdgMLocalStatusParams .....	95
5.3.1.13. WdgMTrigger .....	96

5.3.1.14. WdgMGeneral .....	97
5.3.1.15. WdgMCallerIds .....	102
5.3.1.16. WdgMSupervisedEntity .....	102
5.3.1.17. WdgMCheckpoint .....	104
5.3.1.18. WdgMInternalTransition .....	105
5.3.1.19. WdgMGeneralMulticore .....	105
5.3.1.20. WdgMServiceAPI .....	107
5.3.1.21. WdgMSupervisorCallouts .....	109
5.3.1.22. WdgMWatchdog .....	114
5.3.2. Application programming interface (API) .....	115
5.3.2.1. Type definitions .....	115
5.3.2.1.1. WdgM_ASR32_SupervisedEntityType .....	115
5.3.2.1.2. WdgM_ASR40_CheckpointIdType .....	115
5.3.2.1.3. WdgM_ASR40_SupervisedEntityType .....	116
5.3.2.1.4. WdgM_CheckpointIdType .....	116
5.3.2.1.5. WdgM_ConfigType .....	116
5.3.2.1.6. WdgM_EB_GlobalInitStatusType .....	116
5.3.2.1.7. WdgM_EB_InitStatusType .....	116
5.3.2.1.8. WdgM_GlobalStatusType .....	117
5.3.2.1.9. WdgM_LocalStatusType .....	117
5.3.2.1.10. WdgM_ModeType .....	117
5.3.2.1.11. WdgM_SupervisedEntityType .....	117
5.3.2.2. Macro constants .....	117
5.3.2.2.1. WDG_M_AR_RELEASE_MAJOR_VERSION .....	117
5.3.2.2.2. WDG_M_AR_RELEASE_MINOR_VERSION .....	118
5.3.2.2.3. WDG_M_AR_RELEASE_REVISION_VERSION .....	118
5.3.2.2.4. WDG_M_EB_E_DEINIT_REQUEST .....	118
5.3.2.2.5. WDG_M_EB_E_INIT_REQUEST .....	118
5.3.2.2.6. WDG_M_EB_E_REENTRANCY .....	118
5.3.2.2.7. WDG_M_EB_E_SETMODE_REQUEST .....	118
5.3.2.2.8. WDG_M_EB_E_SLAVE_FAILED_CHANGEMODE .....	118
5.3.2.2.9. WDG_M_EB_INIT_STATUS_DEINIT .....	119
5.3.2.2.10. WDG_M_EB_INIT_STATUS_INIT .....	119
5.3.2.2.11. WDG_M_EB_PARAM_UNUSED .....	119
5.3.2.2.12. WDG_M_E_AMBIGIOUS .....	119
5.3.2.2.13. WDG_M_E_CPID .....	119
5.3.2.2.14. WDG_M_E_DEPRECATED .....	119
5.3.2.2.15. WDG_M_E_DISABLE_NOT_ALLOWED .....	120
5.3.2.2.16. WDG_M_E_INV_POINTER .....	120
5.3.2.2.17. WDG_M_E_NOT_AUTHORIZED .....	120
5.3.2.2.18. WDG_M_E_NO_INIT .....	120
5.3.2.2.19. WDG_M_E_PARAM_CONFIG .....	120

5.3.2.2.20. WDGM_E_PARAM_MODE .....	120
5.3.2.2.21. WDGM_E_PARAM_SEID .....	120
5.3.2.2.22. WDGM_E_PARAM_WRONG_CORE_ID .....	121
5.3.2.2.23. WDGM_E_SEDEACTIVATED .....	121
5.3.2.2.24. WDGM_GLOBAL_STATUS_DEACTIVATED .....	121
5.3.2.2.25. WDGM_GLOBAL_STATUS_EXPIRED .....	121
5.3.2.2.26. WDGM_GLOBAL_STATUS_FAILED .....	121
5.3.2.2.27. WDGM_GLOBAL_STATUS_OK .....	121
5.3.2.2.28. WDGM_GLOBAL_STATUS_STOPPED .....	122
5.3.2.2.29. WDGM_LOCAL_STATUS_DEACTIVATED .....	122
5.3.2.2.30. WDGM_LOCAL_STATUS_EXPIRED .....	122
5.3.2.2.31. WDGM_LOCAL_STATUS_FAILED .....	122
5.3.2.2.32. WDGM_LOCAL_STATUS_OK .....	122
5.3.2.2.33. WDGM_MODULE_ID .....	123
5.3.2.2.34. WDGM_SID_CHECKPOINT_REACHED .....	123
5.3.2.2.35. WDGM_SID_DEINIT .....	123
5.3.2.2.36. WDGM_SID_GET_FIRST_EXPIRED_SEID .....	123
5.3.2.2.37. WDGM_SID_GET_GLOBAL_STATUS .....	123
5.3.2.2.38. WDGM_SID_GET_LOCAL_STATUS .....	123
5.3.2.2.39. WDGM_SID_GET_MODE .....	123
5.3.2.2.40. WDGM_SID_GET_VERSION_INFO .....	124
5.3.2.2.41. WDGM_SID_INIT .....	124
5.3.2.2.42. WDGM_SID_MAIN_FUNCTION .....	124
5.3.2.2.43. WDGM_SID_PERFORM_RESET .....	124
5.3.2.2.44. WDGM_SID_SET_MODE .....	124
5.3.2.2.45. WDGM_SID_UPDATE_ALIVE_COUNTER .....	124
5.3.2.2.46. WDGM_SW_MAJOR_VERSION .....	124
5.3.2.2.47. WDGM_SW_MINOR_VERSION .....	125
5.3.2.2.48. WDGM_SW_PATCH_VERSION .....	125
5.3.2.2.49. WDGM_VENDOR_ID .....	125
5.3.2.2.50. WdgM_CheckpointReached .....	125
5.3.2.2.51. WdgM_UpdateAliveCounter .....	125
5.3.2.3. Objects .....	125
5.3.2.3.1. WDGM_CONFIG_NAME .....	125
5.3.2.4. Functions .....	126
5.3.2.4.1. Supervisor_WdgM_ASR32_SetModeRedirectionCallout .....	126
5.3.2.4.2. Supervisor_WdgM_ASR40_SetModeRedirectionCallout .....	126
5.3.2.4.3. Supervisor_WdgM_ActivateAliveSupervisionRedirectionCallout .....	127
5.3.2.4.4. Supervisor_WdgM_DeInitRedirectionCallout .....	127
5.3.2.4.5. Supervisor_WdgM_DeactivateAliveSupervisionRedirectionCallout .....	127
5.3.2.4.6. Supervisor_WdgM_DetCallout .....	128
5.3.2.4.7. Supervisor_WdgM_GetCoreIdCallout .....	128

5.3.2.4.8. Supervisor_WdgM_GetExpectedInitStateCallout .....	128
5.3.2.4.9. Supervisor_WdgM_GetExpectedWdgMModeCallout .....	129
5.3.2.4.10. Supervisor_WdgM_GetTimeCallout .....	129
5.3.2.4.11. Supervisor_WdgM_GlobalModeSwitchCallout .....	130
5.3.2.4.12. Supervisor_WdgM_IndividualModeSwitchCallout .....	130
5.3.2.4.13. Supervisor_WdgM_InitRedirectionCallout .....	130
5.3.2.4.14. Supervisor_WdgM_IsPerformResetCallout .....	131
5.3.2.4.15. Supervisor_WdgM_SupervisionExpiredCallout .....	131
5.3.2.4.16. WdgM_ASR32_UpdateAliveCounter .....	131
5.3.2.4.17. WdgM_ASR40_CheckpointReached .....	132
5.3.2.4.18. WdgM_ASR40_UpdateAliveCounter .....	133
5.3.2.4.19. WdgM_DeInit .....	133
5.3.2.4.20. WdgM_GetFirstExpiredSEID .....	133
5.3.2.4.21. WdgM_GetGlobalStatus .....	134
5.3.2.4.22. WdgM_GetLocalStatus .....	134
5.3.2.4.23. WdgM_GetMode .....	135
5.3.2.4.24. WdgM_GetVersionInfo .....	135
5.3.2.4.25. WdgM_Init .....	135
5.3.2.4.26. WdgM_MainFunction .....	136
5.3.2.4.27. WdgM_PerformReset .....	136
5.3.2.4.28. WdgM_SetMode .....	137
5.3.3. Integration notes .....	137
5.3.3.1. Exclusive areas .....	137
5.3.3.2. Production errors .....	137
5.3.3.3. Memory mapping .....	138
5.3.3.4. Integration requirements .....	139
5.3.3.4.1. lim.WdgM.EB_INTREQ_WdgM_0001 .....	139
5.3.3.4.2. lim.WdgM.EB_INTREQ_WdgM_0002 .....	140
5.3.3.4.3. lim.WdgM.EB_INTREQ_WdgM_0003 .....	140
5.3.3.4.4. lim.WdgM.EB_INTREQ_WdgM_0004 .....	140
5.3.3.4.5. lim.WdgM.EB_INTREQ_WdgM_0005 .....	140
5.3.3.4.6. lim.WdgM.EB_INTREQ_WdgM_0006 .....	141
5.3.3.4.7. lim.WdgM.EB_INTREQ_WdgM_0007 .....	141





# 1. Overview of EB tresos AutoCore Generic 8 Watchdog Stack documentation

Welcome to the EB tresos AutoCore Generic 8 Watchdog Stack (ACG8 Watchdog Stack) product documentation.

This document provides:

- ▶ [Chapter 2, “Supported features”](#): list of features supported by the ACG8 Watchdog Stack
- ▶ [Chapter 3, “ACG8 Watchdog Stack release notes”](#): release notes for the ACG8 Watchdog Stack modules
- ▶ [Chapter 4, “ACG8 Watchdog Stack user guide”](#): background information and instructions
- ▶ [Chapter 5, “ACG8 Watchdog Stack module references”](#): information about configuration parameters and the application programming interface

## 2. Supported features

ACG8 Watchdog Stack supports the following features:

- ▶ **Extended logical monitoring on several cores:** Support for multiple supervision graphs per checkpoint to allow monitoring fork/join constructs with truly concurrent execution on different cores.
- ▶ **Logical monitoring on several cores:** Support for monitoring of control flow graphs that are distributed across several cores.
- ▶ **Deadline monitoring on several cores:** Support for deadline monitoring on several cores with start and end checkpoints located on the same core.
- ▶ **Alive supervision on several cores:** Support for alive supervision on several cores with a single main function on a master core.
- ▶ **Alive supervision:** Support for cyclic triggering of a watchdog hardware via the MCAL driver Wdg.

## 3. ACG8 Watchdog Stack release notes

### 3.1. Overview

This chapter provides the ACG8 Watchdog Stack product specific release notes. General release notes that are applicable to all products are provided in the EB tresos AutoCore Generic documentation. Refer to the general release notes in addition to the product release notes documented here.

### 3.2. Scope of the release

#### 3.2.1. Configuration tool

Your release of EB tresos AutoCore is compatible with the release of the EB tresos Studio configuration tool:

- ▶ EB tresos Studio: 27.1.0 b200625-0900

#### 3.2.2. AUTOSAR modules

The following table lists the AUTOSAR modules that are part of this ACG8 Watchdog Stack release.

Module name	AUTOSAR version and revision	SWS version and revision	Module version	Supplier
<a href="#">WdgIf</a>	4.0.3 []	2.5.0 [0000]	6.1.23	Elektrobit Automotive GmbH
<a href="#">WdgM</a>	4.0.3 []	2.2.0 [0000]	6.1.36	Elektrobit Automotive GmbH

Table 3.1. Hardware-Independent Modules specified by the AUTOSAR standard

#### 3.2.3. EB (Elektrobit) modules

The following table lists all modules which are part of this release but are not specified by the AUTOSAR standard. These modules include tooling developed by EB or they may hold files shared by all other modules.

Module name	Module version	Supplier
No EB modules available		

Table 3.2. Modules not specified by the AUTOSAR standard

## 3.2.4. MCAL modules and EB tresos AutoCore OS

For information about MCAL modules and OS, refer to the respective documentation, which is available as PDF at `$TRESOS_BASE/doc/3.0_EB_tresos_AutoCore_OS` and `$TRESOS_BASE/doc/5.0_MCAL_modules`<sup>1</sup>. It is also available in the online help in EB tresos Studio. Browse to the folders `EB tresos AutoCore OS` and `MCAL modules`.

## 3.3. Module release notes

### 3.3.1. WdgIf module release notes

- ▶ AUTOSAR R4.0 Rev 3
- ▶ AUTOSAR SWS document version: 2.5.0
- ▶ Module version: 6.1.23.B337087
- ▶ Supplier: Elektrobit Automotive GmbH

#### 3.3.1.1. Change log

This chapter lists the changes between different versions.

##### Module version 6.1.23

2020-06-19

- ▶ Change all `NO_INIT` memory sections to `CLEARED`.
- ▶ Internal module improvement. This module version update does not affect module functionality.

---

<sup>1</sup>`$TRESOS_BASE` is the location at which you installed EB tresos Studio.



**Module version 6.1.21**

2020-05-04

- ▶ Time 2.1 RFM Release.

**Module version 6.1.20**

2020-02-21

- ▶ ASCWDGIF-321 Fixed known issue: Wrong checks in the configuration for VendorApilInfix will lead to generation fails.

**Module version 6.1.19**

2019-12-17

- ▶ Time 2.0 RFD Release.

**Module version 6.0.19**

2019-10-11

- ▶ Implemented ASIL tags.

**Module version 6.0.18**

2019-06-14

- ▶ Implemented multicore support.
- ▶ Support for BSWMD VendorApilInfix.

**Module version 6.0.17**

2019-04-18

- ▶ Update generator to disable vendor infix for a single referenced driver.

**Module version 6.0.16**

2018-10-25

- ▶ Internal module improvement. This module version update does not affect module functionality.



#### **Module version 6.0.15**

2018-05-25

- ▶ Internal module improvement. This module version update does not affect module functionality.

#### **Module version 6.0.14**

2018-03-02

- ▶ Implemented macro-redefinition guards.
- ▶ Implemented compliance to MISRA-C:2012.
- ▶ Internal module improvement. This module version update does not affect module functionality.

#### **Module version 6.0.13**

2017-09-22

#### **Module version 6.0.12**

2017-04-03

- ▶ Internal module improvement. This module version update does not affect module functionality.

#### **Module version 6.0.11**

2016-11-07

- ▶ Changed WdgIf\_ModeType from enumeration to macro definition.

#### **Module version 6.0.10**

2014-07-11

- ▶ Improved ECUC parameter checks

#### **Module version 6.0.9**

2013-12-13

- ▶ Added non-functional code improvements
- ▶ Updated the Basic Software Module Description to specify all external and internal APIs (Basic Software Module Entities)



#### **Module version 6.0.8**

2013-07-04

- ▶ Added non-functional code improvements

#### **Module version 6.0.7**

2013-06-25

- ▶ Added non-functional code improvements

#### **Module version 6.0.6**

2013-06-14

- ▶ Added non-functional code improvements

#### **Module version 6.0.5**

2013-05-10

- ▶ ASCWDGIF-207 Fixed known issue: WdgIf uses wrong API calls to the Wdg driver if multiple Wdgs are configured

#### **Module version 6.0.4**

2013-03-15

- ▶ Added non-functional code improvements

#### **Module version 6.0.3**

2013-02-08

- ▶ Added specification of memory mappings to Basic Software Module Description

#### **Module version 6.0.2**

2012-10-12

- ▶ Changed the top-level structure of the software-component description in the ARXML files from /AUTOSAR/WdgIf to /AUTOSAR\_WdgIf
- ▶ Added non-functional code improvements

#### **Module version 6.0.1**

2012-03-16

- ▶ Added macro definition for single Wdg driver independent of Det
- ▶ Removed compiler warnings for redefined `WdgIf_SetMode` and `WdgIf_SetTriggerCondition`
- ▶ Added symbolic name values for `WdgIfDeviceIndex`
- ▶ Added generation of BSWMD

#### **Module version 6.0.0**

2012-02-17

- ▶ Initial AUTOSAR 4.0 version

### **3.3.1.2. New features**

- ▶ No new features have been added since the last release.

### **3.3.1.3. EB-specific enhancements**

This chapter lists the enhancements provided by the module.

- ▶ This module provides no EB-specific enhancements.

### **3.3.1.4. Deviations**

This chapter lists the deviations of the module from the AUTOSAR standard.

- ▶ No consistency check between code files and header files

Description:

The inter-module version checks as specified by the `WdgIf` SWS are not implemented.

Rationale:

- ▶ The required compile-time version checks would result in an inflexible, hardly integratable basic software stack.
- ▶ EB tresos AutoCore is an already integrated product.
- ▶ The project handling of EB tresos Studio provides means to enforce that only modules with the same EB tresos AutoCore release version can be added to the project.



Requirements:

WDGIF005

- ▶ No AUTOSAR Debugging support

Description:

WdgIf is not instrumented for the usage with AUTOSAR Debugging.

WDGIF052, WDGIF053, WDGIF054, WDGIF055

### 3.3.1.5. Limitations

This chapter lists the limitations of the module. Refer to the module references chapter *Integration notes*, subsection *Integration requirements* for requirements on integrating this module.

- ▶ For this module no limitations are known.

### 3.3.1.6. Open-source software

WdgIf does not use open-source software.

## 3.3.2. WdgM module release notes

- ▶ AUTOSAR R4.0 Rev 3
- ▶ AUTOSAR SWS document version: 2.2.0
- ▶ Module version: 6.1.36.B337087
- ▶ Supplier: Elektrobit Automotive GmbH

### 3.3.2.1. Change log

This chapter lists the changes between different versions.

**Module version 6.1.36**

2020-06-19



- ▶ Change all NO\_INIT memory sections to CLEARED and restriction on Deadline Supervision in the Limitations.xml file due to multicore use.
- ▶ Internal module improvement. This module version update does not affect module functionality.

#### **Module version 6.1.34**

2020-05-04

- ▶ Time 2.1 RFM Release.

#### **Module version 6.1.33**

2020-04-03

- ▶ ASCWDGM-812 Fixed known issue: WdgM\_Mainfunction not generated for slaves

#### **Module version 6.1.32**

2020-02-21

- ▶ ASCWDGM-803 Fixed known issue: Different memory mapping area for definition and declaration of WdgM\_EB\_GlobalStatus
- ▶ ASCWDGM-795 Fixed known issue: WdgM service component does not work with multi-core distribution
- ▶ ASCWDGM-813 Fixed known issue: "Space" character present in the last line of the file WdgM\_Lcfg.h leads to compiler error

#### **Module version 6.1.31**

2019-12-17

- ▶ TimE 2.0 RFD Release.

#### **Module version 6.0.31**

2019-10-11

- ▶ ASCWDGM-758 Fixed known issue: BSWMD does not generate the BSW implementations of all cores.

#### **Module version 6.0.30**

2019-06-14

- ▶ Implemented multicore support.

#### **Module version 6.0.29**

2018-10-25

- ▶ Internal module improvement. This module version update does not affect module functionality.

#### **Module version 6.0.28**

2018-06-22

- ▶ ASCWDGM-654 Fixed known issue: Variables are not assigned to a memory section.
- ▶ Internal module improvement. This module version update does not affect module functionality
- ▶ Removed AUTOSAR 3.1 support from the module.

#### **Module version 6.0.27**

2018-03-02

- ▶ Internal module improvement. This module version update does not affect module functionality
- ▶ Invert logic for AUTOSAR 4.0.2 and remove AUTOSAR 3.x legacy support for symbolic names
- ▶ ASCWDGM-629 Fixed known issue: Behaviour changed when the same alive supervision is used after switching the mode.
- ▶ Added immediate mode switch when calling WdgM\_SetMode().

#### **Module version 6.0.26**

2017-09-22

- ▶ ASCWDGM-594 Fixed known issue: Dem event in WdgM\_Cfg.h causes compilation error
- ▶ Comply to MISRA-C:2012

#### **Module version 6.0.25**

2017-03-31

- ▶ Internal module improvement. This module version update does not affect module functionality.

#### **Module version 6.0.24**

2017-03-10

- ▶ Internal module improvement. This module version update does not affect module functionality.



#### **Module version 6.0.23**

2016-11-07

- ▶ Extended license checks for logical supervision or deadline supervision to cover new license strings `EB_TIME_CFM` and `EB_TIME_DM` as well.

#### **Module version 6.0.22**

2015-11-06

- ▶ Added non-functional code improvements to fix compiler warnings.

#### **Module version 6.0.21**

2015-10-23

- ▶ Added non-functional code improvements to fix compiler warnings.

#### **Module version 6.0.20**

2015-06-19

- ▶ ASCWDGM-515 Fixed known issue: Support for automatic configuration of DEM events

#### **Module version 6.0.19**

2015-01-07

- ▶ Added non-functional code improvements to fix MISRA violations.

#### **Module version 6.0.18**

2014-04-25

- ▶ Added non-functional code improvements to improve text of some container descriptions

#### **Module version 6.0.17**

2013-12-13

- ▶ Added support for the integration into an AUTOSAR 3.1 environment
- ▶ ASCWDGM-467 Fixed known issue: Compilation fails on case-sensitive file systems
- ▶ ASCWDGM-493 Fixed known issue: Configuration for reporting of DEM events `WDGM_E_MONITORING` and `WDGM_E_SET_MODE` is not possible if DEM event `WDGM_E_IMPROPER_CALLER` is not used



#### **Module version 6.0.16**

2013-10-11

- ▶ Added non-functional code improvements to update source code documentation for production error reporting
- ▶ ASCWDGM-474 Fixed known issue: The WdgM may not compile if reporting a production error to the Diagnostic Event Manager is configured
- ▶ Added non-functional code improvements to ease the integration of the WdgM into an ASR31 environment
- ▶ TimE Protection license only: Removed obsolete feature for the configuration of DEM callouts

#### **Module version 6.0.15**

2013-08-07

- ▶ Added non-functional code improvements to fix incorrect generation of macro values `0.0U` to `0U`

#### **Module version 6.0.14**

2013-07-24

- ▶ Added non-functional code improvements to deal with tasking compiler bug on XC2361E (V2.5 r1)

#### **Module version 6.0.13**

2013-07-23

- ▶ ASCWDGM-447 Fixed known issue: The WdgM uses incorrect compiler abstraction

#### **Module version 6.0.12**

2013-07-11

- ▶ Fixed minor code issues (non-function changes) for future debugging support

#### **Module version 6.0.11**

2013-06-25

- ▶ Fixed compiler warning on XC2k derivate reported from a tasking compiler
- ▶ Added non-functional code improvements
- ▶ ASCWDGM-412 Fixed known issue: TimE Protection license only: A data corruption in the internal WdgM data may not result in Global Supervision Status `EXPIRED`



### Module version 6.0.10

2013-06-14

- ▶ ASCWDGM-379 Fixed known issue: Inconsistent starting point of the Supervision Reference Cycle for the first evaluation of Alive Supervision
- ▶ ASCWDGM-378 Fixed known issue: TimE Protection license only: The WdgM never recovers from `FAILED` state in case the Error Recovery feature is enabled and only Deadline Supervision or/and Logical Supervision are configured for the current WdgM mode
- ▶ ASCWDGM-383 Fixed known issue: Wrong individual mode switch notification behavior during initialization and de-initialization phases
- ▶ ASCWDGM-384 Fixed known issue: TimE Protection license only: Missing checks allow an inconsistent configuration leading to undefined behavior if Deadline or Logical Supervision is used
- ▶ ASCWDGM-385 Fixed known issue: TimE Protection license only: Deadline Supervision may not be evaluated according to the specification with respect to parameters `WdgMMainFunctionPeriodTolerance`, `WdgMSupervisionCycle`, `WdgMDeadlineMax`, and `WdgMDeadlineMin`
- ▶ Added non-functional code improvements
- ▶ Added simple Dbg instrumentation for tracing of function enter/exit points
- ▶ ASCWDGM-385 Fixed known issue: Unspecified behavior if `WdgIf_SetMode` fails during initialization of the WdgM.
- ▶ ASCWDGM-393 Fixed known issue: TimE Protection license only: Unspecified behavior of Alive Supervision if configured together with Logical or Deadline Supervision and an enabled Error Recovery
- ▶ ASCWDGM-397 Fixed known issue: TimE Protection license only: The WdgM does not compile for some specific Deadline Supervision configurations
- ▶ ASCWDGM-404 Fixed known issue: The WdgM does not compile if more than one CallerId ID is configured for a mode switch request
- ▶ ASCWDGM-406 Fixed known issue: TimE Protection license only: Unspecified behavior if the WdgM switches back to an old mode while some Supervisions are still active

### Module version 6.0.9

2013-04-30

- ▶ Added non-functional code improvements
- ▶ ASCWDGM-359 Fixed known issue: TimE Protection license only: WdgM does not compile if callout API is configured for `GetExpectedWdgMMode`
- ▶ ASCWDGM-361 Fixed known issue: TimE Protection license only: Deadline Supervision is not performed for Checkpoints which are configured to be both Start Checkpoint and End Checkpoint
- ▶ Implemented tracking of a failed Logical Supervision Status for the successor Checkpoints of a Logical Supervision Graph



- ▶ ASCWDGM-363 Fixed known issue: TimE Protection license only: A failed Deadline Supervision may not lead to global state `EXPIRED` if Error Recovery is enabled
- ▶ ASCWDGM-369 Fixed known issue: TimE Protection license only: The individual mode switch callout API contains an incorrect old mode if the Supervised Entity is deactivated
- ▶ ASCWDGM-371 Fixed known issue: The WdgM does not overwrite an active initialization request when required to by a call to `WdgM_DeInit()`

#### **Module version 6.0.8**

2013-03-15

- ▶ Added support for Alive Supervision of multiple Checkpoints of a Supervised Entity
- ▶ TimE Protection license only: Added support for Logical Supervision
- ▶ TimE Protection license only: Added support for Deadline Monitoring
- ▶ TimE Protection license only: Implemented detection of deadline violation of Supervised Entities in the granularity of the main function cycle
- ▶ Implemented smooth error reaction without a reset for individual Supervised Entities
- ▶ TimE Protection license only: Implemented detection of timing violation of schedule main function
- ▶ Added non-functional code improvements

#### **Module version 6.0.7**

2013-02-08

- ▶ Added specification of Memory Mappings to Basic Software Module Description

#### **Module version 6.0.6**

2012-12-21

- ▶ Added non-functional code improvements
- ▶ Implemented usage of AUTOSAR conform type naming for `ModeDeclarationGroups`
- ▶ TimE Protection license only: Added support for integration in safety projects up to ASIL level D

#### **Module version 6.0.5**

2012-10-12

- ▶ Added AUTOSAR 3.2 support of Rte Interface and SWCD

- ▶ Added non-functional code improvements
- ▶ Added support for optional generation of Service APIs according to AUTOSAR 3.2

#### **Module version 6.0.4**

2012-08-17

- ▶ Added definition of Exclusive Area Activation in Basic Software Module Description
- ▶ Fixed minor issues in generated WdgM Service Component Description
- ▶ Added support of symbolic name generation for Checkpoint IDs via Rte

#### **Module version 6.0.3**

2012-07-13

- ▶ Added internal mode switch to a final mode within `WdgM_DeInit` is now optional

#### **Module version 6.0.2**

2012-06-15

- ▶ Updated WdgM to derive value for `WdgMWatchdogName` from container name
- ▶ Fixed compiler warning in case no Wdgs are configured
- ▶ ASCWDGM-248 Fixed known issue: The Watchdog trigger conditions are incorrectly updated in the `MainFunction()` after an enforced reset via `WdgM_PerformReset()`
- ▶ Added switches for optional containers on same tabs as lists
- ▶ ASCWDGM-256 Fixed known issue: Inconsistent storage class specification for `WdgM_EB_UpdateTriggerConditions()` and `WdgM_EB_SetMode()` cause an error or warning

#### **Module version 6.0.1**

2012-03-16

- ▶ Added non-functional code improvements
- ▶ Added generation of BSWMD

#### **Module version 6.0.0**

2012-02-17

- ▶ Initial AUTOSAR 4.0 version



### 3.3.2.2. New features

- ▶ No new features have been added since the last release.

### 3.3.2.3. EB-specific enhancements

This chapter lists the enhancements provided by the module.

- ▶ Optimized usage of WdgM in case no RTE is used

Description:

The parameter `WdgMRteUsage` (see `WDGM.EB.WdgMRteUsage_Conf`) can be used to disable the RTE interface of the Watchdog Manager. This means that the Watchdog Manager module can be used without an RTE if needed.

Rationale:

Disabling the usage of the Rte makes the integration of the WdgM at the beginning easier.

- ▶ Enhanced production error reporting

Description:

An enhanced production error reporting mechanism has been introduced. This allows to configure the following options independently for each Dem event:

- ▶ Report production errors to the Diagnostics Event Manager (Dem).
- ▶ Report production errors to the Development Error Tracer (Det) as development errors.
- ▶ Do not report production errors at all.

If a production error is redirected towards the Det, you may configure the reported Det error-ID.

Rationale:

This enhancement implements the HIS requirements concerning fault operation and error detection: HisGen0007, HisGen0008, and HisGen0009.

- ▶ Watchdog Manager mode switch to a sleep mode

Description:

The configuration parameter `WdgMSleepMode` in the general tab of a `WdgMConfigSet` entry (see `WDGM.EB.WdgM_DeInit.2`) allows the integrator to specify a sleep mode. The `WdgM_DeInit` function then updates the trigger conditions via a Watchdog Manager mode switch to this sleep mode.

Rationale:

The integrator of the WdgM can specify the WdGM sleep mode. The EcuM does not need to explicitly switch the WdgM mode before it calls the `DeInit` function.

► Provision of Checkpoint IDs via WdgM Service Component Description

Description:

The WdgM Service Component Description specifies the name of the configured Checkpoint IDs of each Supervision Entity as a constant within the interface description as follows: `WdgMConf_WdgMCheckpoint_<CheckpointName>`.

Rationale:

A SWC usually includes only the corresponding Rte header file and not the BSW header file of the Watchdog Manager. Therefore, the Rte must have knowledge about the configured Checkpoint IDs in order to generate the symbolic name values for the Checkpoint IDs used in the SWC.

► Support for optional generation of AUTOSAR 3.2 Service Component Description

Description:

Support for the generation of AUTOSAR 3.2, or AUTOSAR 4.0 service APIs and Software Component Description as well as default service APIs and Software Component Description which can be configured to adhere either to AUTOSAR 3.2 or 4.0 schema version.

Rationale:

AUTOSAR 3.2/3.1 application SWCs of Tier-1 shall be deployed in an AUTOSAR 4.0 environment or AUTOSAR 3.2/3.1 environment.

► Time Protection license only: Support for the configuration of callout functions for the integration into projects with ASIL level up to ASIL D

Description:

The Watchdog Manager provides the configuration of callout functions for the following:

- Error indication (instead of DET and DEM reporting).
- Periodically polling the information regarding the Watchdog Manager state (e.g. Initialization, Watchdog Manager Mode, etc.) from an external entity (e.g. Safety Manager) instead of providing the APIs `WdgM_Init`, `WdgM_DeInit`, or `WdgM_SetMode`.

Rationale:

Support for the integration of Time and Execution Protection for automotive projects with ASIL level up to ASIL D.

► Time Protection license only: Detection of deadline violations in the granularity of the main function cycle

Description:

In addition to deadline supervision of AUTOSAR, the Watchdog Manager detects a deadline violation in the granularity of the main function period in case the Stop Checkpoint of an active deadline monitoring is never called.

Rationale:

Required for projects having ASIL level up to ASIL D.

- TimeE Protection license only: Detection of timing violation of scheduled main function

Description:

The Watchdog Manager monitors its own main function period and reports an error in case a timing violation is detected.

Rationale:

Required for projects having ASIL level up to ASIL D.

- Smooth error reaction without a reset for individual Supervised Entities.

Description:

The Watchdog Manager provides the configuration of a smooth error reaction with error recovery for individual Supervised Entities. In this case, the Watchdog Manager reports the status of a failed Supervision without entering the `Expired` state such that a Watchdog reset will not be performed.

Rationale:

Projects may require a different error strategy than provided by the Watchdog Manager (Watchdog reset) in case a Supervised Entity fails.

- Tracking of a failed Logical Supervision Status for the successor Checkpoints of a Logical Supervision Graph

Description:

If the Watchdog Manager detects a failed Logical Supervision for a called Checkpoint participating in an active Supervision Graph, then a call to the API `WdgM_CheckpointReached()` shall return `E_NOT_OK` for all successor Checkpoints of this Supervision Graph.

Rationale:

A Software Component participating in a control loop (e.g. responsible for controlling an actuator) may trust on the return value of the API `WdgM_CheckpointReached()` to decide whether or not the input values are produced in the correct sequence. Thus the input values can be used independent of the schedule period of Watchdog Manager `MainFunction`.

### 3.3.2.4. Deviations

This chapter lists the deviations of the module from the AUTOSAR standard.

- ▶ The supervision mechanism Logical Monitoring is not available if corresponding TimE license is not activated (reference to product description: ASCPD-90)

Description:

The supervision mechanism Logical Monitoring is not supported and therefore, cannot be configured for Supervision Entities. Note: For all requirements which depend on the result of Logical Supervision, the result is assumed to be correct for all Supervision Entities.

Requirements:

WDGM293\_1, WDGM271, WDGM296, WDGM295, WDGM246, WDGM274, WDGM252, WDGM332, WDGM331, WDGM297, WDGM273, WDGM329, WDGM212, WDGM308, WDGM309, WDGM310, WDGM323, WDGM345\_Conf, WDGM351\_Conf, WDGM350\_Conf, WDGM319\_Conf, WDGM324\_Conf, WDGM323\_Conf, WDGM320\_Conf, WDGM322\_Conf, WDGM321\_Conf

- ▶ The supervision mechanism Deadline Monitoring is not available if corresponding TimE license is not activated (reference to product description: ASCPD-90)

Description:

The supervision mechanism Deadline Monitoring is not supported and therefore, cannot be configured for the Supervision Entities.

---

#### NOTE



For all requirements which depend on the result of Deadline Monitoring, the result is assumed to be correct for all Supervision Entities.

---

Requirements:

WDGM293, WDGM298, WDGM228, WDGM229, WDGM294, WDGM299, WDGM313, WDGM314, WDGM322, WDGM314\_Conf, WDGM318\_Conf, WDGM317\_Conf, WDGM315\_Conf, WDGM316\_Conf

- ▶ Restart/shutdown of partitions is not supported (reference to product description: ASCPD-112)

Description:

In contrast to WDGM225 which states that the WdgM shall restart/shutdown the partition of an OS Application which is configured for a Supervised Entity, the WdgM does not restart/shutdown this partition. That is, it does not call `BswM_WdgM_RequestPartitionReset` of the Basic Software Mode Manager module.

Requirements:

WDGM225, WDGM311, WDGM162

- Post-build time configuration

Description:

Only pre-compile time configuration is supported (reference to product description: ASCPD-77)

Requirements:

WDGM127, WDGM004, WDGM042, WDGM010, WDGM029, WDGM266, WDGM255

- The WdgM does not check if a Supervised Entity ID equals some AUTOSAR module ID

Description:

In contrast to AUTOSAR which does not allow the configuration of SWC Supervised Entities with a Supervised Entity ID value that is equal to the module ID of any AUTOSAR BSW module, there is no such constraint on the Supervised Entity ID values. Users/integrators are responsible for the correct configuration of Supervised Entity ID values and the possible consequences.

Rationale:

There may exist use-cases in legacy projects where some Supervised Entities are not associated to SWCs. Therefore these Supervised Entities may require the configuration of the ID of some AUTOSAR BSW module.

Requirements:

WDGM307

- Optional detection of production code errors

Description:

In contrast to AUTOSAR which does not allow to switch off the detection of production code errors for each individual production code error, the following is possible:

- To keep the production code error as specified.
- To report the production code error to the Development Error Tracer (DET) instead.
- To completely switch off the detection of the production code error.

Rationale:

User-specific configuration of production code errors.

Requirements:

WDGM015

- De-initialization of the Watchdog Manager independent of global Supervision Status

Description:

In contrast to AUTOSAR which allows the de-initialization only from global Supervision Status `WDGM_GLOBAL_STATUS_OK`, the Watchdog Manager can be de-initialized independent of the actual global Supervision Status.

Rationale:

The caller of `WdgM_DeInit` must be aware of the actual global Supervision Status.

Requirements:

WDGM286

- Consistent interpretation of parameter `WdgMFailedAliveSupervisionRefCycleTol`

Description:

AUTOSAR generally defines the parameter `WdgMFailedAliveSupervisionRefCycleTol` as the number of allowed failed reference cycles until a Supervised Entity goes into state `WDGM_LOCAL_STATUS_EXPIRED`. However, AUTOSAR specifies a state machine where one additional failed reference cycle is allowed. In contrast to the state machine specified in AUTOSAR, the implementation allows at most `WdgMFailedAliveSupervisionRefCycleTol` failed reference cycles until a Supervised Entity goes into state `WDGM_LOCAL_STATUS_EXPIRED`.

See Bugzilla entry [http://www.autosar.org/bugzilla/show\\_bug.cgi?id=58303](http://www.autosar.org/bugzilla/show_bug.cgi?id=58303)

Requirements:

WDGM206, WDGM204

- Consistent interpretation of parameter `WdgMExpiredSupervisionCycleTol`

Description:

AUTOSAR generally defines the parameter `WdgMExpiredSupervisionCycleTol` as the number of allowed main function cycles in global state `WDGM_GLOBAL_STATUS_EXPIRED` until the `WdgM` enters the global state `WDGM_GLOBAL_STATUS_STOPPED`. However, AUTOSAR specifies a state machine where two additional main function cycles in global state `WDGM_GLOBAL_STATUS_EXPIRED` are allowed. In contrast to the state machine specified in AUTOSAR, the implementation allows at most `WdgMExpiredSupervisionCycleTol` main function cycles in global state `WDGM_GLOBAL_STATUS_EXPIRED` until the `WdgM` enters the global state `WDGM_GLOBAL_STATUS_STOPPED`.

See Bugzilla entry [http://www.autosar.org/bugzilla/show\\_bug.cgi?id=58303](http://www.autosar.org/bugzilla/show_bug.cgi?id=58303)

Requirements:

WDGM077, WDGM219, WDGM220

- Symbolic port names instead of numeric port name numbering

Description:

In contrast to requirements WDGM.ASR40.WDGM147 and WDGM.ASR40.WDGM149, the RTE ports are named by their symbolic short name taken from the configuration.

Rationale:

Symbolic port names do not change when ports are deleted or inserted as it is the case for numeric names because they get renumbered and need to be reconnected. Also the symbolic name can be chosen to reflect the purpose of a port which makes the port connection process easier and less error prone.

Requirements:

WDGM147, WDGM149

- `WdgM_GetVersionInfo` as a function

Description:

In contrast to WDGM262 which suggests to implement the API as a macro in case caller and callee of `WdgM_GetVersionInfo` are available at compile time, the `WdgM` always implements the API as a function.

Requirements:

WDGM262

- No AUTOSAR Debugging support

Description:

`WdgM` is not instrumented for the usage with AUTOSAR Debugging.

Requirements:

WDGM238, WDGM239, WDGM240, WDGM241, WDGM242, WDGM234, WDGM235, WDGM236, WDGM237

- `WdgM` does not check the versions of other modules

Description:

In contrast to WDGM013, the `WdgM` does not check the version numbers of included header files from other modules.

Rationale:

In general, the modules are delivered within a whole AutoCore delivery, in which the versions are consistent and therefore do not have to be checked.

Furthermore, this allows the combination of the module with other AUTOSAR compatible but not fully compliant modules. This might e.g., permit to combine the module with (adapted) modules from different AUTOSAR releases or with non-AUTOSAR modules that simulate the necessary behavior.

Requirements:

WDGM013

- ▶ Mode switch is done synchronously or asynchronously (synchronously to `MainFunction`) if the `WdgMSetModeSynchron` is set respectively not set.

Description:

In contrast to AUTOSAR which specifies that a call to `WdgM_SetMode` immediately switches the `WdgM` mode (WDGM186), the `WdgM_SetMode` request is applied either at the end of the next `MainFunction` call, either like AUTOSAR specifies, depending on how the `WdgMSetModeSynchron` parameter is configured.

See Bugzilla entry [http://www.autosar.org/bugzilla/show\\_bug.cgi?id=57805](http://www.autosar.org/bugzilla/show_bug.cgi?id=57805).

Requirements:

WDGM154

- ▶ (De-)Initialization is done synchronously to `MainFunction`

Description:

In contrast to AUTOSAR which specifies that a call to `WdgM_Init` or `WdgM_DeInit` immediately initializes or de-initializes the Watchdog Manager, the Watchdog Manager is (de-)initialized at the next `MainFunctionCycle`.

Requirements:

WDGM268, WDGM269, WDGM285, WDGM298, WDGM296, WDGM151, WDGM018, WDGM135, WDGM350, WDGM286, WDGM261

- ▶ Non-compliant deviations in vendor-specific module definition file

Description:

The vendor-specific module definition file (VSMD) has non-compliant deviations to the AUTOSAR specification:



Violations against Rule EcucSws\_1014: Additional vendor specific parameter definitions (using ParameterTypes), container definitions and references shall be added to the VSMD according to the alphabetical order.

This affects variables and containers in following StMD-Nodes:

- ▶ /AUTOSAR/WdgM
- ▶ /AUTOSAR/WdgM/WdgMGeneral

Rationale:

A merge of AUTOSAR and vendor specific variables in these containers intentionally results in a different order for a clear arrangement of vendor specific parameters in EB tresos Studio.

### 3.3.2.5. Limitations

This chapter lists the limitations of the module. Refer to the module references chapter *Integration notes*, subsection *Integration requirements* for requirements on integrating this module.

- ▶ Restriction of numbering of Checkpoint IDs

Description:

WDGM306\_Conf does not specify any constraints for the parameter `WdgMCheckpointId` except that the ID must be unique within the Supervised Entity. In contrast to WDGM306\_Conf, all Checkpoint IDs within a Supervised Entity must be zero-based and dense.

Rationale:

This reduces the consumption of RAM and ROM.

Requirements:

WDGM306\_Conf

- ▶ Restriction of number of configurable Checkpoints per Supervision Entity

Description:

The AUTOSAR Specification of the Watchdog Manager specifies that the container `WdgMCheckpoint` as part of `WdgMSupervisedEntity` has a maximum multiplicity of 65535. In contrast to this, at most 256 checkpoints can be configured for a Supervision Entity.

Rationale:

This reduces the consumption of RAM and ROM.

Requirements:

WDGM305\_Conf

- ▶ Restriction of number of configurable Supervision Entities

Description:

The AUTOSAR Specification of the Watchdog Manager specifies that the container `WdgMSupervisedEntity` as part of `WdgMGeneral` has a maximum multiplicity of 65535. In contrast to this, at most 256 Supervised Entities can be configured.

Rationale:

This reduces the consumption of RAM and ROM.

Requirements:

WDGM303\_Conf

- ▶ Restriction on ID range of Checkpoints and Supervised Entities

Description:

AUTOSAR states that the range of valid IDs depends on the number of configured Supervised Entities and on the chosen platform type. In contrast to this, the WdgM always uses the maximum possible range of `uint16` instead of `uint8`.

Requirements:

WDGM038

- ▶ Restriction on dynamic change of the main function period

Description:

In contrast to AUTOSAR which specifies a mode-dependent `WdgM_MainFunction` period, the `MainFunction` period is always defined via the `WdgMSupervisionCycle` parameter of the first configured mode.

- ▶ Restriction on architectural assumption regarding data access

Description:

The CPU of the ECU where WdgM is integrated provides 32-bit data types which can be read and written in an atomic way.

- ▶ Restriction on Deadline Supervision

Description:



Checkpoints used in Deadline Supervision must be called from the same core.

#### **3.3.2.6. Open-source software**

WdgM does not use open-source software.

## 4. ACG8 Watchdog Stack user guide

### 4.1. Overview

This user guide describes the concepts and the configuration of the modules:

- ▶ Watchdog Interface (WdgIf)
- ▶ Watchdog Manager (WdgM)

This user guide is intended for readers who have good knowledge of AUTOSAR and about the purpose of the ACG8 Watchdog Stack modules.

### 4.2. Background information

#### 4.2.1. Stack overview

The watchdog stack of EB tresos AutoCore contains a group of modules that simplify the watchdog task on an automotive ECU. [Figure 4.1, “Watchdog stack overview”](#) illustrates the structure of the watchdog stack.

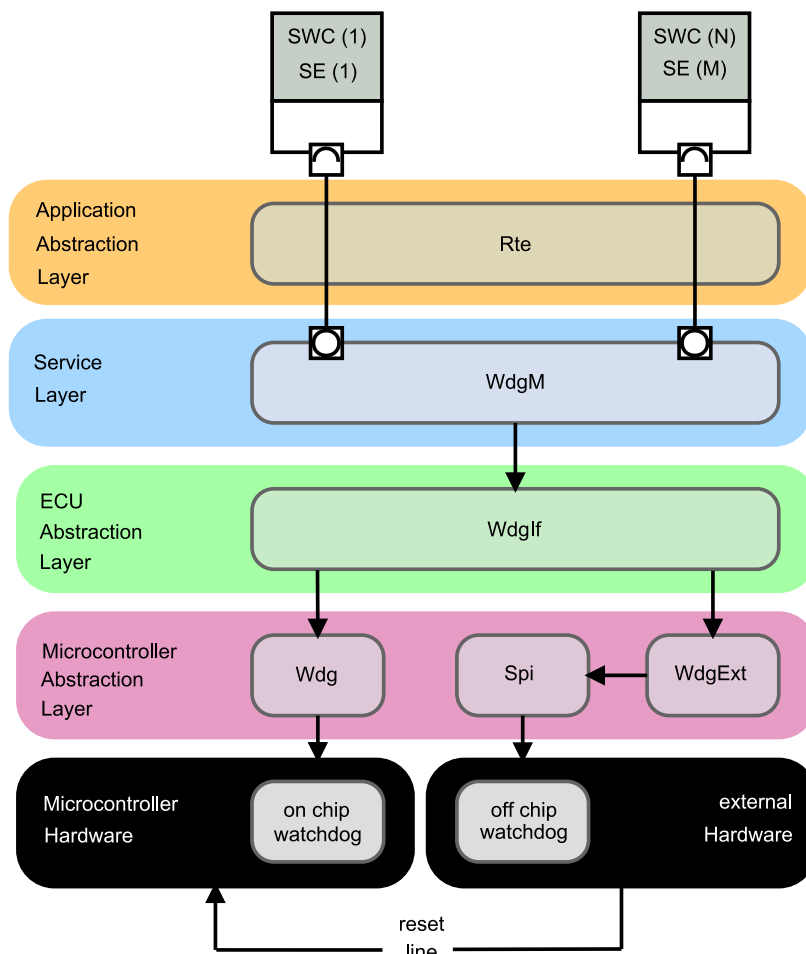


Figure 4.1. Watchdog stack overview

The image shows two software components, the SWC (1) and SWC (N). The `WdgM` module can supervise each of these software components. A software component which is supervised is also called *supervised entity* (SE (1) and SE (M) in [Figure 4.1, “Watchdog stack overview”](#)).

Details about the software component alive supervision are available in [Section 4.6.2.3, “Alive supervision of supervised entities”](#).

The Watchdog Manager (`WdgM`) reports via the Watchdog Interface (`WdgIf`) a triggering condition to the Watchdog Drivers (`Wdg`). The Watchdog Driver then is responsible for triggering the watchdog hardware as long as the triggering condition is true.

For details about triggering the watchdog, see [Section 4.6.2.4, “Triggering the watchdog”](#).

## 4.2.2. Module dependencies

This chapter summarizes the modules that are needed by the watchdog stack to simplify the integration on your ECU.

### 4.2.2.1. Development Error Detection (Det)

The watchdog stack uses the following API function of the `Det` module:

► `Det_ReportError()`

### 4.2.2.2. Diagnostic Event Manager (Dem)

The watchdog stack uses the following API function of the `DEM` module:

► `Dem_ReportErrorStatus()`

### 4.2.2.3. Micro-Controller Unit (Mcu)

The watchdog stack uses the following API function of the `Mcu` module:

► `Mcu_PerformReset()`

### 4.2.2.4. Runtime Environment (Rte)

The `WdgM` uses the following API function of the `Rte` module:

► `Rte_Switch_<port>_<mode declaration group prototype>()`

### 4.2.2.5. Schedule Manager (SchM)

The following main function needs to be called by the `SchM` module:

► `WdgM_MainFunction`

### 4.2.2.6. ECU State Manager (EcuM)

The following (de)init functions need to be called by the `EcuM` module:

- ▶ WdgM\_Init()
- ▶ WdgM\_DeInit()
- ▶ Wdg\_Init()

Note: In sleep modes, the hardware watchdog is triggered by the `ECuM` module.

#### 4.2.2.7. BSW Mode Manager (BswM)

The watchdog stack uses the following API function of the `BswM` module:

- ▶ `BswM_WdgM_RequestPartitionReset()`

### 4.2.3. Multi-core support

The watchdog stack of EB tresos AutoCore supports the use in multi-core systems. It is based on a master-satellite concept, with one master instance and one or more satellite instances. [Figure 4.2, “Multi-core watchdog stack overview”](#) illustrates the structure of the multi-core watchdog stack.

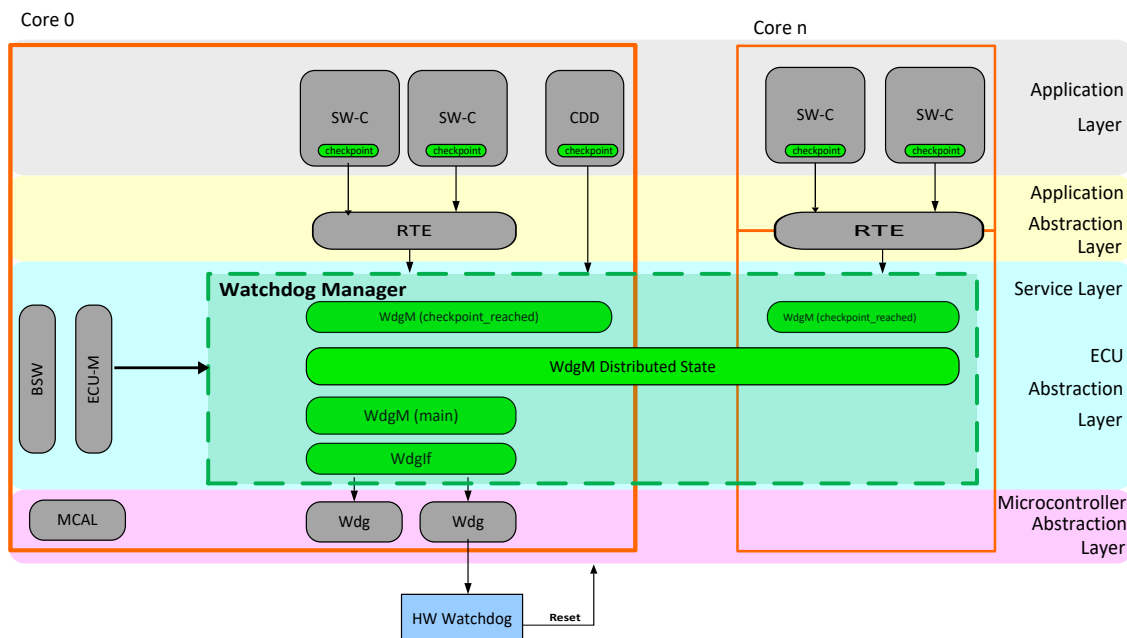


Figure 4.2. Multi-core watchdog stack overview

The following items are processed by the master instance and satellite instances (i.e. all cores in the above picture):

- ▶ Get the local status for each supervised entity: retrieve the status of all checkpoints of a supervised entity mapped to a core
- ▶ Determine the status of the alive indication for each core
- ▶ Determine the local supervision status for each core instance: this is composed of the result of deadline supervision, logical supervision, and alive supervision of that core

The following items are processed only by the master instance (core 0 in the above picture):

- ▶ Determine the global supervision status: set the global supervision status (`WdgM` may observe more than one supervised entity spread on multiple cores)
- ▶ Update the watchdog trigger condition: if the global supervision status is OK

## 4.3. Initializing the watchdog stack

The initialization of the watchdog stack is called by the `EcUM` module. Therefore the `WdgM` and the `Wdg` modules have to be inserted in the start-up sequence *Startup One*. For details on this sequence, see the EB tresos AutoCore Generic Mode Management documentation.

Because the initialization of the `WdgM` is done asynchronously with `WdgM_MainFunction()`, the `WdgM` module is considered to be initialized only after the `WdgM_MainFunction()` is called. If any service of `WdgM` is called before, the module reports the error code `WDGM_E_NO_INIT`.

## 4.4. Configuring the watchdog stack

To simplify the configuration work, we recommend to start in the lowest layer and move your way up. In this way, you can configure the references between the modules without switching from one configuration job to another.

For details about individual configuration parameters, see the module references chapter.

1. Configure the hardware-dependent Watchdog Driver `Wdg`. For configuration instructions, see the folder `plugins/Wdg_<TargetId>/doc/` in your EB tresos Studio installation tree.
2. Configure the Watchdog Interface `WdgIf`. For configuration instructions, see [Section 4.5, “WdgIf module user guide”](#).
3. Configure the Watchdog Manager `WdgM`. For configuration instructions, see [Section 4.6, “WdgM module user guide”](#).



## 4.5. WdgIf module user guide

### 4.5.1. Configuring the WdgIf module

The `WdgIf` module contains a list of references to all `Wdg` modules. The screenshot below shows an example for two watchdog drivers: `Wdg` and `WdgExt`.

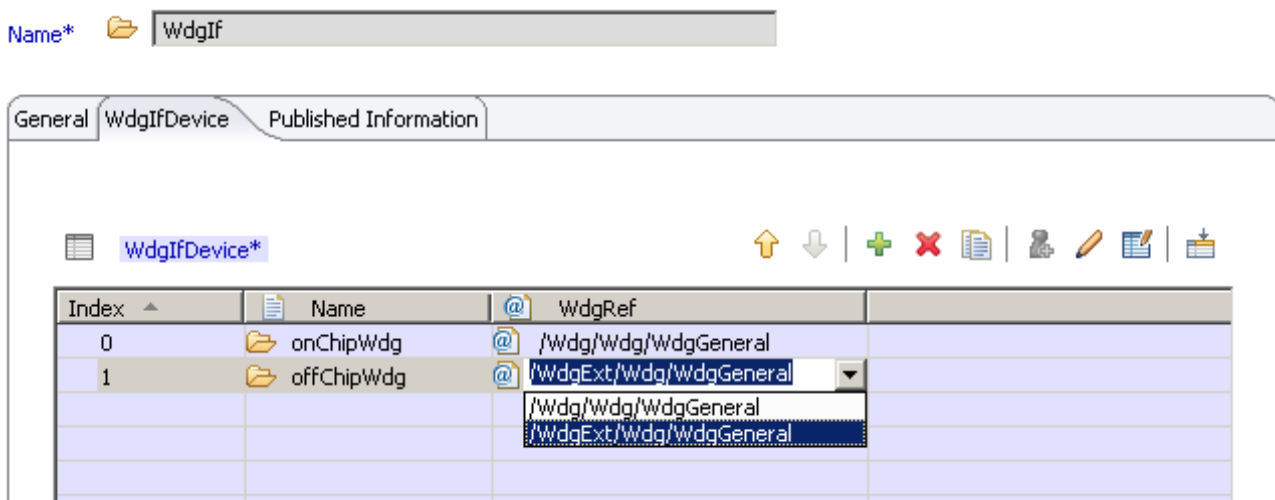


Figure 4.3. Watchdog Interface tab WdgIfDevice



#### Adding a reference to a watchdog

##### Step 1

Open the `WdgIf` editor on the `WdgIfDevice` tab.

##### Step 2

Click the **+** button to add a **WdgIfDevice**.

##### Step 3

Select the desired watchdog driver from the drop-down list box.

##### Step 4

Repeat these steps for each watchdog driver.

For configuration parameter details, see the module references chapter.

## 4.6. WdgM module user guide

### 4.6.1. Overview

This chapter provides you with `WdgM`-specific information:

- ▶ [Section 4.6.2, “Background information”](#) explains the concepts of the `WdgM` module.
- ▶ [Section 4.6.3, “Configuring the WdgM module”](#) provides instructions on how to configure the `WdgM` module.

### 4.6.2. Background information

The Watchdog Manager (`WdgM`) enables you to supervise how reliable an application is executed. It manages the application's periodicity, the logical constraints, and its timing constraints.

The supervision of application timing constraints is decoupled from the timing constraints of any underlying hardware watchdogs. For details about application timing constraints, see [Section 4.6.2.3, “Alive supervision of supervised entities”](#). For details about the hardware watchdogs timing constraints, see [Section 4.6.2.4, “Triggering the watchdog”](#).

The hardware watchdogs are accessed via the homogenous `WdgIf` module. The `Wdg` init function is neither abstracted by the `WdgIf` nor by the `WdgM` module.

#### 4.6.2.1. State machines and interfaces

The following illustration shows the state machines of the `WdgM` module and their interfaces.

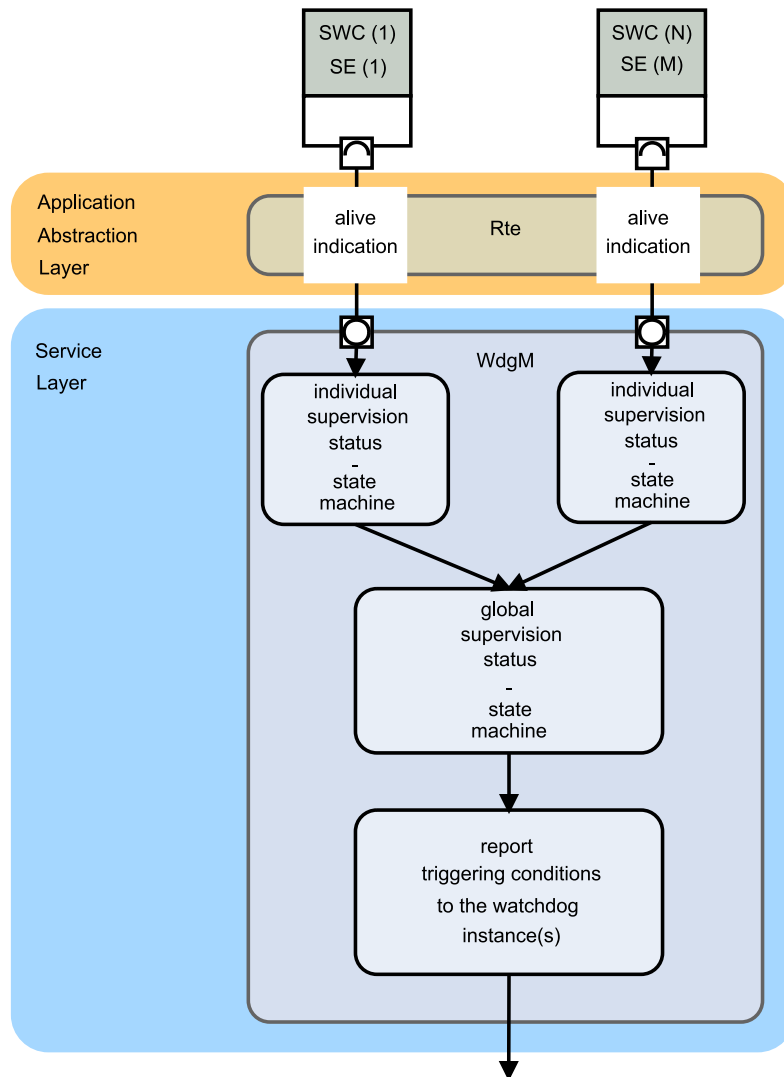


Figure 4.4. Watchdog Manager overview

#### 4.6.2.2. Concepts of supervision

The WdgM provides the following concepts of supervision:

► *Supervision of multiple entities:*

The WdgM supervises many entities for execution reliability. The WdgM provides an individual port for each supervised entity. At each respective port, the supervised entities have to indicate their aliveness for any configured checkpoint.

► *Supervision of a single entity:*

The `WdgM` supervises a single entity for execution reliability. The `WdgM` provides a port for the supervised entity. There, the supervised entity has to indicate its aliveness for the configured checkpoints.

► *No supervision:*

No entity is supervised for execution reliability. The proof of execution reliability follows the cyclic scheduling of the `WdgM`.

### 4.6.2.3. Alive supervision of supervised entities

The alive supervision of supervised entities is optional for the watchdog stack. If there is no supervised entity, the triggering of the watchdog is independent of any software component and simply follows the cyclic scheduling of the `WdgM`.

For details about the watchdog triggering, see [Section 4.6.2.4, “Triggering the watchdog”](#).

If one or more software components are supervised, the triggering of the watchdog follows the state of the *global supervision status* as depicted in [Figure 4.4, “Watchdog Manager overview”](#).

#### 4.6.2.3.1. Supervision status state machines

The `WdgM` handles two different types of state machines:

- Individual supervision status state machine
- Global supervision status state machine

For each supervised entity, there is exactly one *individual supervision status state machine*, which has one of the following states:

► `WDGM_LOCAL_STATUS_OK`

The supervised entity sent its alive indication correctly.

► `WDGM_LOCAL_STATUS_FAILED`

The supervised entity has not sent its alive indication correctly but the number of failed alive supervisions has not yet exceeded a configurable boundary.

► `WDGM_LOCAL_STATUS_EXPIRED`

The supervised entity has not sent its alive indication correctly and the number of failed alive supervisions has been exceeded.

► `WDGM_LOCAL_STATUS_DEACTIVATED`

The alive supervision is not performed on deactivated supervised entities. The global supervision status does not consider deactivated supervised entities.

Besides the individual supervision status state machine(s), there is exactly one *global supervision status state machine*, which depends on the state(s) of the individual supervision status state machine(s). The following states are defined:

- ▶ **WDGM\_GLOBAL\_STATUS\_OK**: All activated individual supervision status state machines are in the state **WDGM\_LOCAL\_STATUS\_OK**.
- ▶ **WDGM\_GLOBAL\_STATUS\_FAILED**: One or more activated individual supervision status state machines are in the state **WDGM\_LOCAL\_STATUS\_FAILED**. There is no individual supervision status state machine that is in the state **WDGM\_LOCAL\_STATUS\_EXPIRED**.
- ▶ **WDGM\_GLOBAL\_STATUS\_EXPIRED**: One or more activated individual supervision status state machines are in the state **WDGM\_LOCAL\_STATUS\_EXPIRED**, but a tolerance window has not yet been exceeded. The tolerance window is configured with the parameter **WdgMExpiredSupervisionCycleTol**. See [Figure 4.13, “Watchdog Manager WdgMConfigSet General tab”](#) for details.
- ▶ **WDGM\_GLOBAL\_STATUS\_STOPPED**: One or more activated individual supervision status state machines are in the state **WDGM\_LOCAL\_STATUS\_EXPIRED** and a tolerance window has been exceeded. The tolerance window is configured with the parameter **WdgMExpiredSupervisionCycleTol**. See [Figure 4.13, “Watchdog Manager WdgMConfigSet General tab”](#) for details.

#### 4.6.2.3.2. Effects of mode switching on alive supervision

If `WdgM_SetMode` is called with a new mode that reuses an alive supervision from the previous mode, the alive indications from the previous mode are ignored. This means that, after changing the mode, alive supervision starts from the beginning.

The mode switch can be done asynchronously at the end of the `WdgM_MainFunction` by setting the parameter `WdgMSetModeSynchron` to `FALSE`. Then, after alive supervision is verified, the following applies:

- ▶ If **WdgMSupervisionReferenceCycle** is 1, the verification of alive supervision is always done.
- ▶ If **WdgMSupervisionReferenceCycle** is greater than 1, the verification depends on when `WdgM_SetMode` is called. If `WdgM_SetMode` is called between the last two **WdgMSupervisionReferenceCycles**, the verification is done. Otherwise, verification is ignored.

#### 4.6.2.3.3. Example of the alive supervision

For two examples of alive supervision, see chapter 7.2.3.1 in the AUTOSAR 4.0.3 Specification of Watchdog Manager, V2.2.0.

#### 4.6.2.4. Triggering the watchdog

The `WdgM` module reports the triggering condition of the watchdog instances. The triggering of the watchdog hardware with the right timing is performed by the Watchdog Drivers as long as the reported trigger condition is true. The triggering condition is a counter value that the `WdgM` sets cyclically. This enables the `WdgM` to supervise the program execution abstracted from the triggering of the hardware watchdog entities.

The correct trigger condition is reported cyclically if the global supervision status equals

- ▶ `WDGM_LOCAL_STATUS_OK`,
- ▶ `WDGM_LOCAL_STATUS_FAILED` or
- ▶ `WDGM_LOCAL_STATUS_EXPIRED`.

The watchdog is *not* triggered if the global supervision status equals `WDGM_GLOBAL_STATUS_STOPPED`.

You may configure timing constraints for each individual watchdog instance within the `WdgM` module. The timing constraints enable:

- ▶ cyclical triggering within a maximum time period
- ▶ triggering within a defined time window

If the `WdgM` module is configured to execute on multiple cores, only the **WdgM Master Instance** triggers the watchdog drivers. If each core has its own watchdog, the **WdgM Master Instance** must have access to each watchdog that has to be triggered.

The `WdgIf` module provides uniform access to services of the underlying watchdog drivers such as mode switching and triggering. Select the appropriate watchdog driver via a device index as described in [Section 4.5, “WdgIf module user guide”](#).

#### 4.6.2.5. WdgM modes

The `WdgM` module allows to configure different modes as shown in [Figure 4.5, “Watchdog Manager modes”](#). This enables you to configure different supervision and trigger settings for different modes of the watchdog stack. For example, you may use another `WdgM` mode during ECU start-up than during normal operation.

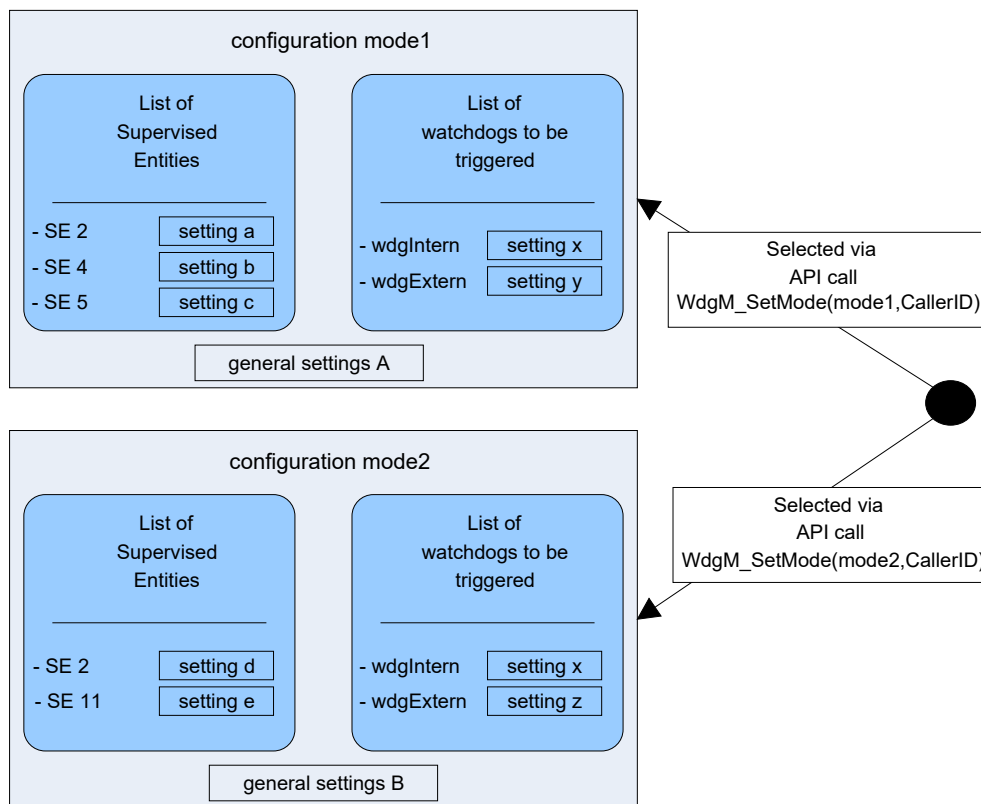


Figure 4.5. Watchdog Manager modes

You can configure a list of supervised entities for each mode. This list of entities is to be supervised for the mode specified. For each supervised entity, it is additionally possible to configure specific settings such as timing parameters or tolerance settings.

Furthermore, you can configure a list of watchdogs that are to be triggered for each mode. Again, you can configure specific settings for each watchdog, such as timing or the mode used.

You can switch between the settings configured during run-time with the API call `WdgM_SetMode()`.

The mode switch can be performed either synchronously with the call or asynchronously at the end of `WdgM_MainFunction`, depending if `WdgMSetModeSynchron` is set to `TRUE`, respectively set to `FALSE`.

If multi-core is configured, the mode switch is done as follows:

1. The `WdgM` master instance does a mode switch.
2. If the mode switch was successful, the master instance prompts all `WdgM` satellite instances to change to the new mode.

Because the instances run in parallel, it may be possible that not all satellite instances change the mode at the same time. During the time the satellites change to the new mode, the `WdgM` master instance computes the global supervision status based on the local statuses of the instances that finished the mode switch.

There is a protection mechanism if the satellite instances do not switch to the new mode. By configuring the `WdgMMasterWaitSlaveModeSwitch` parameter, you can specify how long to wait for the satellites to change to the new mode before an error is given. This mechanism is also used when the `WdgM` instances are initialized.

#### 4.6.2.6. Development error detection

If development error detection is enabled, the driver's services perform regular development error checks. A development error is an error that must be detected and fixed during the development phase. It must not occur in the production code.

All development errors are reported to the `Det` module, a central error hook function within the AUTOSAR environment.

If an error occurs, the error hook routine is called and the error code together with the service-ID and the module-ID are passed on as parameters. See the `Det` module user guide in the EB tresos AutoCore Generic Base documentation for details about the `Det` and its API.

If you enabled development error detection, the following development error checks are performed by the services of the `WdgM` module:

- ▶ Checks the pointer to configuration data when calling the `WdgM_Init()` function.  
  
Reports the error code `WDGM_E_PARAM_CONFIG (0x11)` and service-ID `0x00` if the pointer to configuration data is a NULL pointer.
- ▶ Checks the parameter `mode` for being within the valid range when calling the `WdgM_SetMode()` function.  
  
Reports the error code `WDGM_E_PARAM_MODE (0x12)` and service-ID `0x03` if the `mode` exceeds the valid range.
- ▶ Checks the parameter `SEID` for being a valid Supervised Entity ID when calling the `WdgM_GetLocalStatus()` function.  
  
Reports the error code `WDGM_E_PARAM_SEID (0x13)` and service-ID `0x0c` if the `SEID` is not valid.
- ▶ Checks the parameter `SEID` for being a valid Supervised Entity ID when calling the `WdgM_UpdateAliveCounter()` function.  
  
Reports the error code `WDGM_E_PARAM_SEID (0x13)` and service-ID `0x04` if the `SEID` is not valid.
- ▶ Checks the parameter `SEID` for being a valid Supervised Entity ID when calling the `WdgM_CheckpointReached()` function.  
  
Reports the error code `WDGM_E_PARAM_SEID (0x13)` and service-ID `0x0e` if the `SEID` is not valid.



- ▶ Checks the parameter `CheckpointID` for being a valid Checkpoint ID when calling the `WdgM_CheckpointReached()` function.

Reports the error code `WDGM_E_CPID` (0x16) and service-ID 0x0e if the `CheckpointID` is not valid.

- ▶ Reports the error code `WDGM_E_DEPRECATED` (0x17) and service-ID 0x04 when calling the `WdgM_UpdateAliveCounter()` function.
- ▶ Checks if in the current mode there are more than one Alive Supervisions (`WdgMAliveSupervision`) configured for the Supervised Entity `SEID` when calling the `WdgM_UpdateAliveCounter()` function.

Reports the error code `WDGM_E_AMBIGIOUS` (0x18) and service-ID 0x04 when calling the `WdgM_UpdateAliveCounter()` function if in the current mode there are more than one Alive Supervisions (`WdgMAliveSupervision`) configured.

- ▶ Checks the parameter `SEID` for being an activated Supervised Entity in the current watchdog mode when calling the `WdgM_CheckpointReached()` function.

Reports the error code `WDGM_E_SEDEACTIVATED` (0x19) and service-ID 0x0e if the Supervised Entity with ID `SEID` is deactivated in the current watchdog mode.

- ▶ Checks for unsupported calls of any `WdgM` function except functions `WdgM_GetFirstExpiredSEID()` and `WdgM_Init()`

Reports the error code `WDGM_E_NO_INIT` (0x10), if the `WdgM` function is called in any other than the initialized state.

- ▶ Checks the parameter `Status` to be a valid pointer when calling the `WdgM_GetLocalStatus()` function.

Reports the error code `WDGM_E_INV_POINTER` (0x14) and service-ID 0x0c if the parameter `Status` variable is a NULL pointer.

- ▶ Checks the parameter `Status` to be a valid pointer when calling the `WdgM_GetGlobalStatus()` function.

Reports the error code `WDGM_E_INV_POINTER` (0x14) and service-ID 0x0d if the parameter `Status` variable is a NULL pointer.

- ▶ Checks the parameter `Mode` to be a valid pointer when calling the `WdgM_GetMode()` function.

Reports the error code `WDGM_E_INV_POINTER` (0x14) and service-ID 0x0b if the parameter `Mode` variable is a NULL pointer.

- ▶ Checks the parameter `SEID` to be a valid pointer when calling the `WdgM_GetFirstExpiredSEID()` function.

Reports the error code `WDGM_E_INV_POINTER` (0x14) and service-ID 0x10 if the parameter `SEID` variable is a NULL pointer.

- ▶ Checks the parameter `VersionInfo` to be a valid pointer when calling the `WdgM_GetVersionInfo()` function.

Reports the error code `WDGM_E_INV_POINTER` (0x14) and service-ID 0x02 if the parameter `VersionInfo` variable is a NULL pointer.

- Checks for invalid disabling of the underlying watchdog drivers when calling the `WdgM_SetMode()` function.

Reports the error code `WDGM_E_DISABLE_NOT_ALLOWED` (0x15) and service-ID 0x03, if the parameter `mode` disables the Watchdog Driver (`WDGIF_OFF_MODE`), while the parameter `WdgM_WdgOffModeEnabled` prohibits disabling of underlying watchdog drivers.

- Checks for an invalid core ID calling of a supervised entity checkpoint when calling the `WdgM_CheckpointReached()` function.

Reports the error code `WDGM_E_PARAM_WRONG_CORE_ID` (0x84) and service-ID 0x0e, if the parameter `SEID` is mapped to a different core than the one that calls it.

- Checks if the satellite instances switched to the new mode, in the time specified by parameter `WdgMMasterWaitSlaveModeSwitch` after calling `WdgM_SetMode()` with a new mode.

Reports the error code `WDGM_EB_E_SLAVE_FAILED_CHANGEMODE` (0x85) and service-ID 0x08, if the satellite did not switch to the new mode in the configured time.

#### 4.6.2.7. Run-time error detection

If the global supervision status of the `WdgM` module reaches the state `WDGM_GLOBAL_STATUS_STOPPED` (i.e. alive supervision has failed and a reset occurs), the `WdgM` reports an error event to the Diagnostic Error Manager (`Dem`) module with status `FAILED`. The symbolic name of the event is `WDGM_E_SUPERVISION` and is defined by the `Dem` module.

If a switch mode failure occurs for at least one of the configured watchdog drivers, the `WdgM` reports an error event to the Diagnostic Error Manager (`Dem`) with status `failed`. The symbolic name of the reported error event is `WDGM_E_SET_MODE` and is defined by the `Dem` module.

If an improper caller is detected for the `WdgM_SetMode()` function, the `WdgM` reports the error event `WDGM_E_IMPROPER_CALLER` to the `Dem` with status `FAILED`.

#### 4.6.2.8. Software component description

This chapter describes the data types and interfaces provided by the SWC description of the `WdgM`.

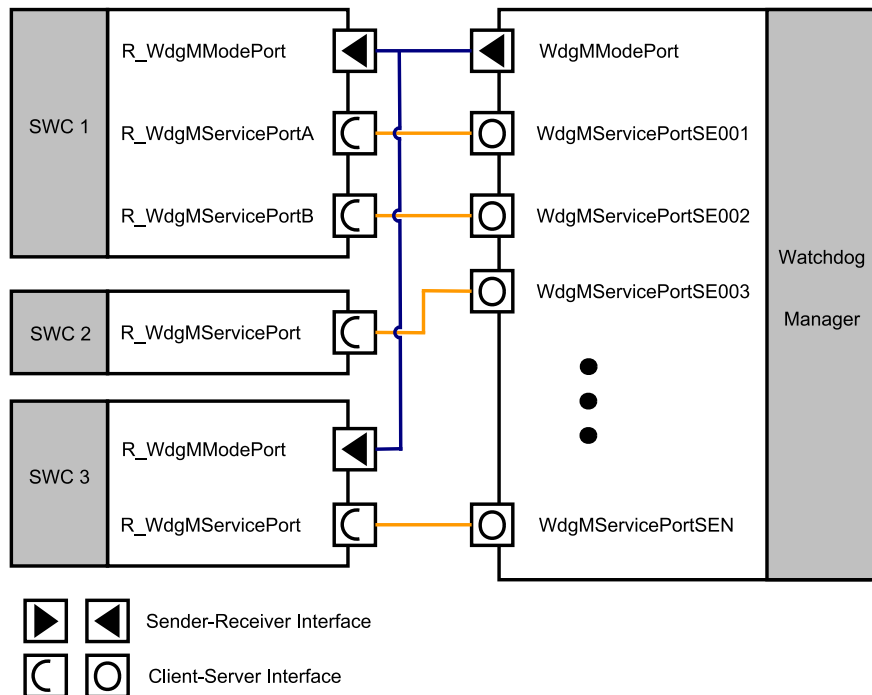


Figure 4.6. Overview of Watchdog Manager ports

#### 4.6.2.8.1. Data types

The following data types are used within the Watchdog Managers software component description.

##### 4.6.2.8.1.1. WdgMModeType

<b>Element:</b>	<b>ModeDeclarationGroup</b>
<b>Modes:</b>	SUPERVISION_OK, SUPERVISION_FAILED, SUPERVISION_EXPIRED, SUPERVISION_STOPPED, SUPERVISION_DEACTIVATED
<b>Initial mode:</b>	SUPERVISION_OK
<b>Description:</b>	These modes represent the global Watchdog Manager alive supervision status which is reported to the RTE.

Table 4.1. TS\_WDGM\_70764

#### 4.6.2.8.2. Ports

##### 4.6.2.8.2.1. WdgMModePort

<b>Interface type:</b>	<b>ModeSwitchInterface</b>
<b>Interface name:</b>	WdgM_IndividualMode
<b>Mode declarations group prototypes:</b>	currentMode (data type WdgMModeType)
<b>Description:</b>	This mode port reports the current Local Supervision Status of a single Supervised Entity.

<b>Interface type:</b>	<b>ModeSwitchInterface</b>
<b>Interface name:</b>	WdgM_GlobalMode
<b>Mode declarations group prototypes:</b>	currentMode (data type WdgMModeType)
<b>Description:</b>	This mode port reports the current mode of the Watchdog Manager which represents the Global Supervision Status that is combined from all individual Supervised Entities.

##### 4.6.2.8.2.2. WdgMServicePortSE<nnn>

<b>Interface type:</b>	<b>ClientServerInterface</b>
<b>Interface name:</b>	WdgM_AliveSupervision
<b>Operation prototypes:</b>	UpdateAliveCounter(ERR/{E_NOT_OK/})  CheckpointReached(IN WdgM_CheckpointIdType, ERR/{E_NOT_OK/})
<b>Description:</b>	This port allows each supervised entity (SE) to trigger its alive counter (CheckpointReached).

### 4.6.3. Configuring the WdgM module

For information about the `WdgM` module initialization, see [Section 4.3, “Initializing the watchdog stack”](#).

### 4.6.3.1. Referencing all configured watchdogs



Referencing all watchdogs configured in WdgIf

#### Step 1

Open the `WdgM` editor on the **WdgMWatchdog** tab.

#### Step 2

Click the **+** button to add a **WdgMWatchdog** list entry.

#### Step 3

Select the desired watchdog interface from the drop-down list box. See the following image for a configuration example.

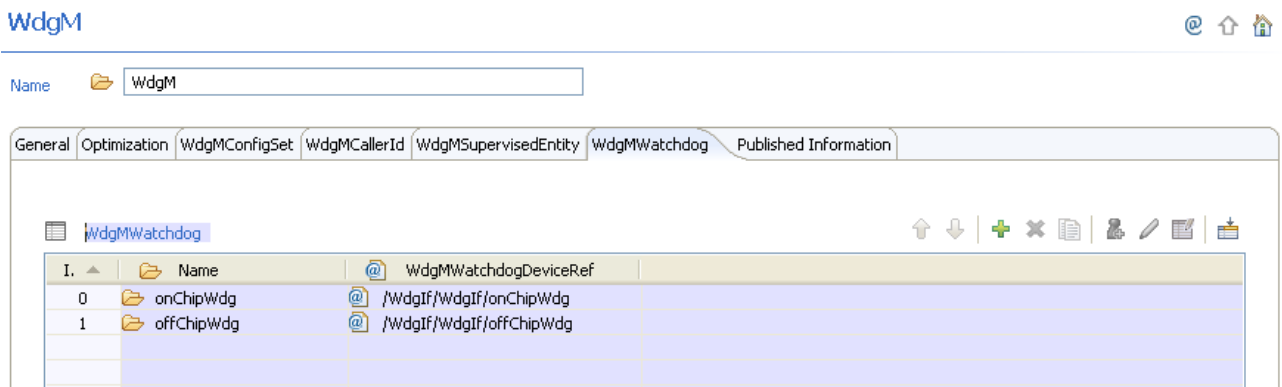


Figure 4.7. Watchdog manager tab WdgMWatchdog

### 4.6.3.2. Adding all supervised entities



Adding a supervised entity

#### Step 1

On the **WdgMSupervisedEntity** tab, click the **+** button to add a **WdgMSupervisedEntity** list entry.

The **WdgMSupervisedEntityId** is set automatically when you add a new element.

#### Step 2

For a multi-core distribution of `WdgM`, configure the **WdgMSupervisedEntityCoreId** with the core ID on which the supervised entity resides. See the following image for a configuration example.

**WdgM** @ ↑ ↓ 🏠

Name 📁 WdgM


General EB Published Information Optimization WdgMConfigSet WdgMCallerId WdgMSupervisedEntity WdgMWatchdog Published Information

📄 WdgMSupervisedEntity ↑ ↓ + × 📄 👤 ✎ 📄 📅

Index	Name	WdgMSupervisedEn...	WdgMInt...	WdgMSupervisedEntityCoreId	WdgMErr...
0	WdgMSup...	0	@ /WdgM/W...	0	<input type="checkbox"/>
1	WdgMSup...	1	@ /WdgM/W...	1	<input type="checkbox"/>
2	WdgMSup...	2	@ /WdgM/W...	2	<input type="checkbox"/>

Figure 4.8. Watchdog manager tab WdgMSupervisedEntity

### 4.6.3.3. Adding the checkpoints for a supervised entity



Adding a checkpoint for a supervised entity

- Step 1  
On the **WdgMSupervisedEntity** tab, open an entry for editing.
- Step 2  
On the **WdgMCheckpoint** tab, click the + button to add a **WdgMCheckpoint** list entry.
- Step 3  
Set the **WdgMCheckpointId** to a unique ID in the scope of a supervised entity. See the following image for a configuration example.

**WdgMSupervisedEntity** @ ↑ ↓ 🏠

Name 📁 WdgMSupervisedEntity\_0

General WdgMInternalCheckpointFinalRef WdgMCheckpoint WdgMInternalTransition

📄 WdgMCheckpoint ↑ ↓ + × 📄 👤 ✎ 📄 📅

I.	Name	WdgMCheckpointId
0	CP0_1	0

Figure 4.9. Watchdog manager tab WdgMCheckpoint

- Step 4  
Add more checkpoints as required.

#### 4.6.3.4. Defining the initial and final checkpoints for logical supervision



Adding the initial and final checkpoints

##### Step 1

Within the **WdgMSupervisedEntity**, go to the **WdgMInternalCheckpointFinalRef** tab.

##### Step 2

Click the **+** button to add a **WdgMInternalCheckpointFinalRef** list entry.

##### Step 3

Select the reference to the final checkpoint from the drop-down list box. See the following image for a configuration example.

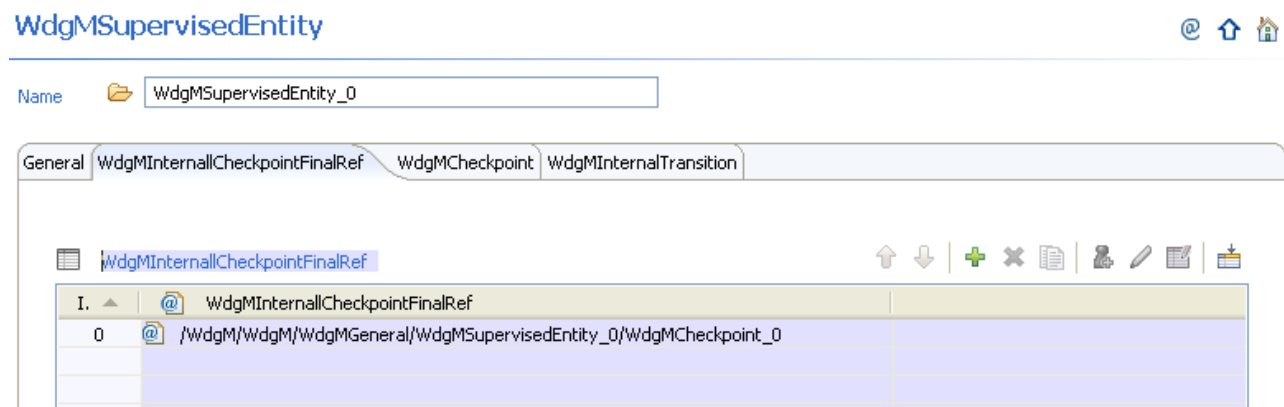


Figure 4.10. Watchdog manager tab WdgMInternalCheckpointFinalRef

##### Step 4

Switch to the **General** tab.

##### Step 5

In **WdgMInternalCheckpointInitialRef**, select the initial checkpoint from the drop-down list box.

#### 4.6.3.5. Defining a configuration set

You can define different configuration sets for different modes. For background information, see [Section 4.6.2.5, “WdgM modes”](#). You configure the configuration set in the **WdgMConfigSet** container. Only one multiple configuration container is allowed. To configure the multiple configuration container, double-click it.

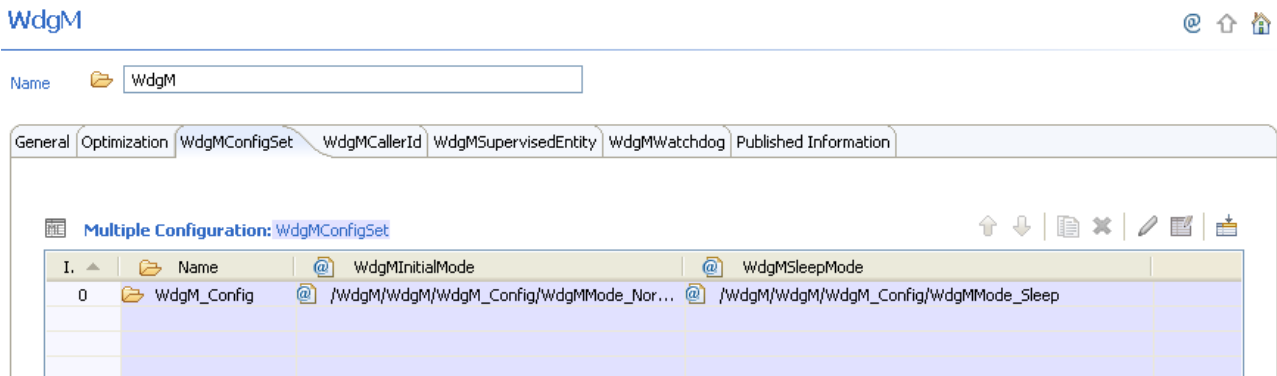


Figure 4.11. Watchdog Manager WdgMConfigSet



### Defining a configuration set

**Step 1**

Open the **WdgMConfigSet** entry for editing.

**Step 2**

Go to the **WdgMMode** tab.

**Step 3**

Click on **+** to add at least one mode.

**Step 4**

Set the **WdgMModeId** to a unique ID. See the following image for a configuration example.

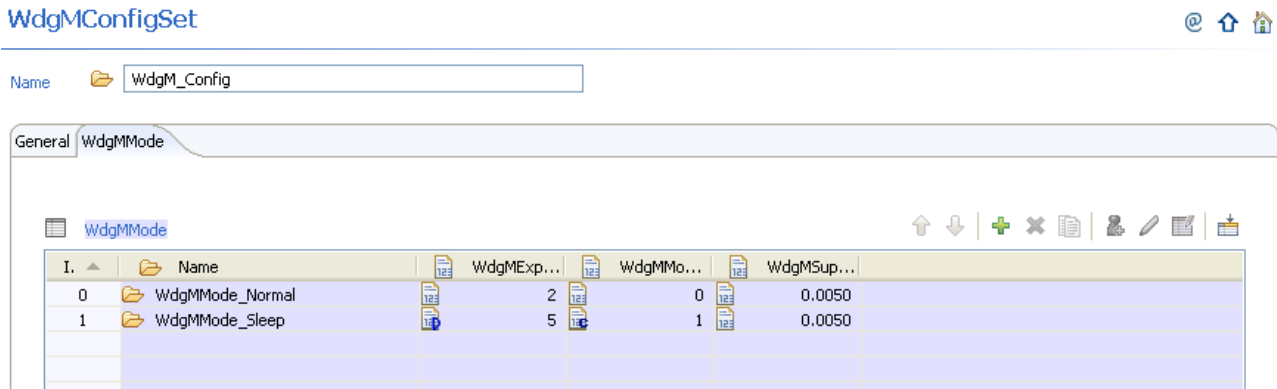


Figure 4.12. Watchdog Manager modes

**Step 5**

Go to the General tab

**Step 6**

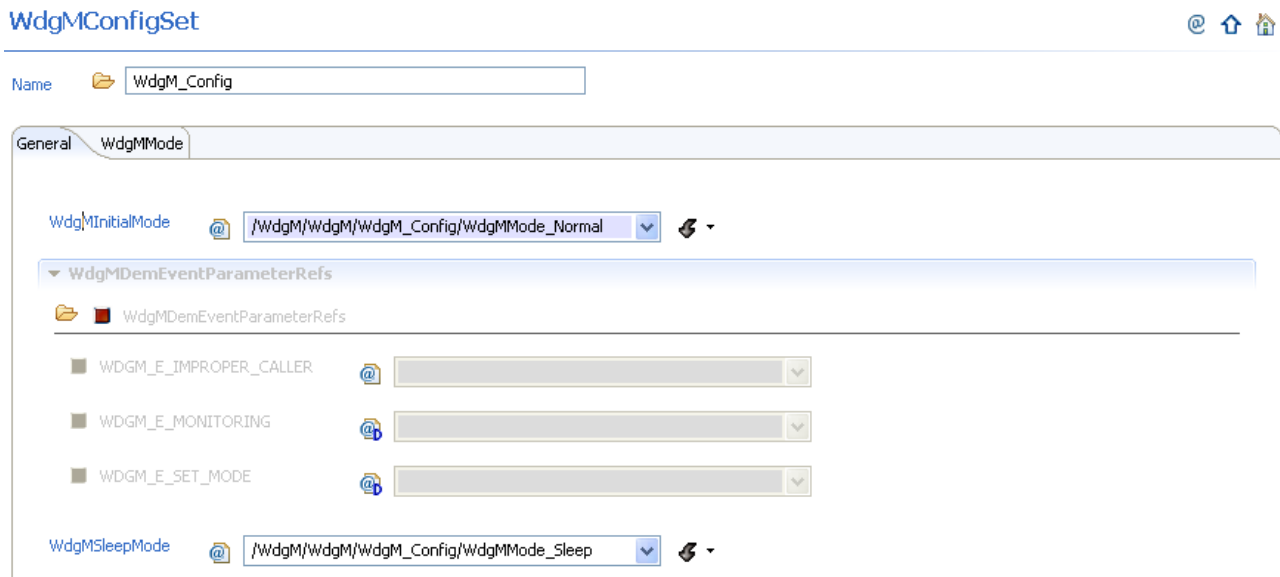
Set the **WdgMInitialMode** to a configured WdgMMode that the Watchdog Manager is in after it has been initialized.



#### Step 7

Set the **WdgMSleepMode** to a configured WdgMMode that the Watchdog Manager is in after `WdgM_DeInit()` has been called. See the following image for a configuration example.

### WdgMConfigSet



The screenshot shows the 'WdgMConfigSet' configuration window with the 'General' tab selected. At the top, there is a 'Name' field containing 'WdgM\_Config'. Below this, the 'WdgMMode' tab is active. It features a 'WdgMInitialMode' dropdown menu set to '/WdgM/WdgM/WdgM\_Config/WdgMMode\_Normal'. Underneath, a section titled 'WdgMDemEventParameterRefs' contains three entries: 'WDGM\_E\_IMPROPER\_CALLER', 'WDGM\_E\_MONITORING', and 'WDGM\_E\_SET\_MODE', each with a corresponding dropdown menu. At the bottom, the 'WdgMSleepMode' dropdown menu is set to '/WdgM/WdgM/WdgM\_Config/WdgMMode\_Sleep'.

Figure 4.13. Watchdog Manager WdgMConfigSet General tab

#### 4.6.3.6. Configuring a mode

The concept of the different modes is explained in [Section 4.6.2.5, “WdgM modes”](#). For details about each configuration parameter, see the module references chapter.



#### Configuring a mode


##### Step 1

On the **WdgMMode** tab, double-click a mode entry.

##### Step 2



On the **General** tab, configure the general settings for the mode. See the following image for a configuration example.



**WdgMMode** @ ↑ 🏠

Name  WdgMMode\_Normal

---

General **WdgMAliveSupervision** WdgMDeadlineSupervision WdgMExternalLogicalSupervision WdgMLocalStatusParams WdgMTrigger

WdgMExpiredSupervisionCycleTol (0 -> 65535)   

WdgMModeId (0 -> 255) (0 -> 255)   



WdgMSupervisionCycle (0 -> 65535)   

Figure 4.14. Watchdog Manager mode General

#### Step 3

On the **WdgMAliveSupervision** tab, add all software components that require an alive supervision in this mode.

#### Step 4

On the **WdgMDeadlineSupervision** tab, add all software components that require a deadline supervision in this mode.


#### Step 5

On the **WdgMExternalLogicalSupervision** tab, add all software components that require an external logical supervision in this mode.

#### Step 6

On the **WdgMTrigger** tab, add all watchdog drivers that shall be triggered in this mode.

#### Step 7

On the **WdgMLocalStatusParams**, click the  button to add a **WdgMLocalStatusParams** list entry for each configured supervised entity in this mode, and set the **WdgMLocalStatusSupervisedEntityRef** to this supervised entity.

#### Step 8

Double-click each **WdgMLocalStatusParams** list entry and set the **WdgMFailedAliveSupervisionRefCycleTol** to configure the acceptable amount of reference cycles with incorrect/failed alive supervisions for the referenced supervised entity.

### 4.6.3.7. Configuring the WdgM for multi-core

The **General Multicore Configuration Parameters** container is an optional container for the general configuration of `WdgMGeneralMulticore`. This shall be configured only if `WdgM` is used on multiple cores.



The screenshot shows a configuration window titled "General Multicore Configuration Parameters". It features a "Name" field with the value "WdgMGeneralMulticore". Below this, there are three rows of configuration parameters, each with a name, a value field, and a refresh icon:

Parameter Name	Value
WdgM Number Of Cores	3
WdgM Master Core Id	0
WdgMMasterWaitSlaveModeSwitch	0

Figure 4.15. General Multi-core Configuration Parameters tab



### Configuring the WdgM for multi-core

#### Step 1

Set the **WdgMNumberOfCores** to a value that specifies on how many cores **WdgM** shall execute.

#### Step 2

Set the **WdgMMasterCoreId** to a value that corresponds to the core ID on which the master instance shall execute.

#### Step 3

Set the **WdgMMasterWaitSlaveModeSwitch** to a value that specifies how many main functions the master instance shall wait for mode switch synchronization between cores.

## 5. ACG8 Watchdog Stack module references

### 5.1. Overview

This chapter provides module references for the ACG8 Watchdog Stack product modules. These include a detailed description of all configuration parameters. Furthermore this chapter lists the application programming interface with all data types, constants and functions.

The content of the sections is sorted alphabetically according the EB tresos AutoCore Generic module names.

For further information on the functional behavior of these modules, refer to the chapter ACG8 Watchdog Stack user's guide.

#### 5.1.1. Notation in EB module references

EB notation may differ from the AUTOSAR standard notation in the software specification documents (SWS). This section describes the notation of *default value* and *range* fields in the EB module references.

##### 5.1.1.1. Default value of configuration parameters

If there is no default value specified for a parameter, the default value field is omitted to prevent ambiguity with parameters that have -- as default values.

Example: The parameter `BswMCompuConstText` of the `BswM` module of EB tresos AutoCore Generic 8 Mode Management has no default value field, therefore it is omitted.

##### 5.1.1.2. Range information of configuration parameters

The range of a configuration parameter contains an upper and a lower boundary. However, in special cases the range of allowed values can be computed by means of an XPath function that is evaluated at configuration time. An XPath function can either be a standard `xpath:<function>()` or a custom `cxpath:<function>()` function. The range of a configuration parameter may be computed based on other configuration parameters

that are referenced from the XPath function. For more information on custom XPath functions, see section *Custom XPath Functions API* of the EB tresos Studio developer's guide.

Example: The parameter `BswMCompuConstText` of the `BswM` module of EB tresos AutoCore Generic 8 Mode Management has the custom XPath function `cxpath:getCompuMethodsVT()` in the range field which provides the allowed values.

## 5.2. WdgIf

### 5.2.1. Configuration parameters

Containers included		
Container name	Multiplicity	Description
<a href="#">CommonPublishedInformation</a>	1..1	<b>Label:</b> Common Published Information Common container, aggregated by all modules. It contains published information about vendor and versions.
<a href="#">PublishedInformation</a>	1..1	<b>Label:</b> EB Published Information Additional published parameters not covered by Common-PublishedInformation container.
<a href="#">WdgIfDevice</a>	1..255	It contains the information for the selection of a particular Watchdog device in case multiple Watchdog drivers are connected.
<a href="#">WdgIfGeneral</a>	1..1	This container collects all generic watchdog interface parameters.

Parameters included	
Parameter name	Multiplicity
<a href="#">IMPLEMENTATION_CONFIG_VARIANT</a>	1..1

Parameter Name	IMPLEMENTATION_CONFIG_VARIANT
Label	Config Variant
Multiplicity	1..1
Type	ENUMERATION
Default value	VariantPreCompile

<b>Range</b>	VariantPreCompile
--------------	-------------------

### 5.2.1.1. CommonPublishedInformation

Parameters included	
Parameter name	Multiplicity
<a href="#">ArMajorVersion</a>	1..1
<a href="#">ArMinorVersion</a>	1..1
<a href="#">ArPatchVersion</a>	1..1
<a href="#">SwMajorVersion</a>	1..1
<a href="#">SwMinorVersion</a>	1..1
<a href="#">SwPatchVersion</a>	1..1
<a href="#">ModuleId</a>	1..1
<a href="#">VendorId</a>	1..1
<a href="#">Release</a>	1..1

Parameter Name	ArMajorVersion	
<b>Label</b>	AUTOSAR Major Version	
<b>Description</b>	Major version number of AUTOSAR specification on which the appropriate implementation is based on.	
<b>Multiplicity</b>	1..1	
<b>Type</b>	INTEGER_LABEL	
<b>Default value</b>	2	
<b>Configuration class</b>	<b>PublishedInformation:</b>	
<b>Origin</b>	Elektrobit Automotive GmbH	

Parameter Name	ArMinorVersion	
<b>Label</b>	AUTOSAR Minor Version	
<b>Description</b>	Minor version number of AUTOSAR specification on which the appropriate implementation is based on.	
<b>Multiplicity</b>	1..1	
<b>Type</b>	INTEGER_LABEL	
<b>Default value</b>	5	

<b>Configuration class</b>	<b>PublishedInformation:</b>	
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>ArPatchVersion</b>	
<b>Label</b>	AUTOSAR Patch Version	
<b>Description</b>	Patch level version number of AUTOSAR specification on which the appropriate implementation is based on.	
<b>Multiplicity</b>	1..1	
<b>Type</b>	INTEGER_LABEL	
<b>Default value</b>	0	
<b>Configuration class</b>	<b>PublishedInformation:</b>	
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>SwMajorVersion</b>	
<b>Label</b>	Software Major Version	
<b>Description</b>	Major version number of the vendor specific implementation of the module.	
<b>Multiplicity</b>	1..1	
<b>Type</b>	INTEGER_LABEL	
<b>Default value</b>	6	
<b>Configuration class</b>	<b>PublishedInformation:</b>	
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>SwMinorVersion</b>	
<b>Label</b>	Software Minor Version	
<b>Description</b>	Minor version number of the vendor specific implementation of the module. The numbering is vendor specific.	
<b>Multiplicity</b>	1..1	
<b>Type</b>	INTEGER_LABEL	
<b>Default value</b>	1	
<b>Configuration class</b>	<b>PublishedInformation:</b>	
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>SwPatchVersion</b>	
<b>Label</b>	Software Patch Version	

<b>Description</b>	Patch level version number of the vendor specific implementation of the module. The numbering is vendor specific.	
<b>Multiplicity</b>	1..1	
<b>Type</b>	INTEGER_LABEL	
<b>Default value</b>	23	
<b>Configuration class</b>	<b>PublishedInformation:</b>	
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>ModuleId</b>	
<b>Label</b>	Numeric Module ID	
<b>Description</b>	Module ID of this module from Module List	
<b>Multiplicity</b>	1..1	
<b>Type</b>	INTEGER_LABEL	
<b>Default value</b>	43	
<b>Configuration class</b>	<b>PublishedInformation:</b>	
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>VendorId</b>	
<b>Label</b>	Vendor ID	
<b>Description</b>	Vendor ID of the dedicated implementation of this module according to the AUTOSAR vendor list	
<b>Multiplicity</b>	1..1	
<b>Type</b>	INTEGER_LABEL	
<b>Default value</b>	1	
<b>Configuration class</b>	<b>PublishedInformation:</b>	
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>Release</b>	
<b>Label</b>	Release Information	
<b>Multiplicity</b>	1..1	
<b>Type</b>	STRING_LABEL	
<b>Default value</b>		
<b>Configuration class</b>	<b>PublishedInformation:</b>	
<b>Origin</b>	Elektrobit Automotive GmbH	



### 5.2.1.2. PublishedInformation

Parameters included	
Parameter name	Multiplicity
<a href="#">PbcfgMSupport</a>	1..1

Parameter Name	PbcfgMSupport	
Label	PbcfgM support	
Description	Specifies whether or not the WdgIf can use the PbcfgM module for post-build support.	
Multiplicity	1..1	
Type	BOOLEAN	
Default value	false	
Configuration class	PublishedInformation:	
Origin	Elektrobit Automotive GmbH	

### 5.2.1.3. WdgIfDevice

Parameters included	
Parameter name	Multiplicity
<a href="#">WdgIfDeviceIndex</a>	1..1
<a href="#">WdgIfDriverRef</a>	1..1
<a href="#">WdgIfDrvBswImplementationRef</a>	0..1

Parameter Name	WdgIfDeviceIndex	
Description	Represents the watchdog interface ID so that it can be referenced by the watchdog manager.	
Multiplicity	1..1	
Type	INTEGER	
Configuration class	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	WdgIfDriverRef
Description	Reference to the watchdog drivers that are controlled by the watchdog interface.

<b>Multiplicity</b>	1..1
<b>Type</b>	SYMBOLIC-NAME-REFERENCE
<b>Configuration class</b>	<b>VariantPreCompile:</b> VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC

<b>Parameter Name</b>	<b>WdgIfDrvBswImplementationRef</b>
<b>Description</b>	Reference to the BswImplementation of the underlying driver which contains the vendorId and vendorApiInfix.
<b>Multiplicity</b>	0..1
<b>Type</b>	FOREIGN-REFERENCE
<b>Configuration class</b>	<b>PreCompile:</b> VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH

#### 5.2.1.4. WdgIfGeneral

Parameters included	
Parameter name	Multiplicity
<a href="#">WdgIfDevErrorDetect</a>	1..1
<a href="#">WdgIfVersionInfoApi</a>	1..1
<a href="#">WdgIfDriverAPIInfixEnable</a>	1..1

<b>Parameter Name</b>	<b>WdgIfDevErrorDetect</b>
<b>Description</b>	Pre-processor switch for enabling the development error detection and reporting. true: Development error detection enabled false: Development error detection disabled
<b>Multiplicity</b>	1..1
<b>Type</b>	BOOLEAN
<b>Default value</b>	true
<b>Configuration class</b>	<b>VariantPreCompile:</b> VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC

<b>Parameter Name</b>	<b>WdgIfVersionInfoApi</b>
<b>Description</b>	Pre-processor switch to enable / disable the service returning the version information.

	true: Version information service enabled false: Version information service disabled	
<b>Multiplicity</b>	1..1	
<b>Type</b>	BOOLEAN	
<b>Default value</b>	false	
<b>Configuration class</b>	<b>VariantPreCompile:</b>	VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC	

<b>Parameter Name</b>	<b>WdgIfDriverAPIInfixEnable</b>	
<b>Description</b>	<p>This parameter defines if WdgIf shall use the Vendor Id and the API Infix for accessing the Wdg Driver module in case a single Wdg driver is configured.</p> <p>true: WdgIf uses the Vendor Id and the API Infix of the Wdg Driver for accessing the Driver API (e.g. Wdg_1_driver) in case only a single Wdg driver is used. In addition this name mangling is also used for including the Wdg Driver header file (e.g. Wdg_1_driver.h)</p> <p>false: WdgIf does not use the Vendor Id and the API Infix of the Wdg Driver in case only a single Wdg driver is used.</p> <p>Note: If more than one Wdg driver is configured, name mangling must be used.</p>	
<b>Multiplicity</b>	1..1	
<b>Type</b>	BOOLEAN	
<b>Default value</b>	true	
<b>Configuration class</b>	<b>VariantPreCompile:</b>	VariantPreCompile

## 5.2.2. Application programming interface (API)

### 5.2.2.1. Type definitions

#### 5.2.2.1.1. WdgIf\_ModeType

<b>Purpose</b>	Mode type of the WdgIf module.
<b>Type</b>	uint8

<b>Description</b>	This uint8 type holds the mode types that are passed as parameters to Wdg_Set-Mode().
--------------------	---

## 5.2.2.2. Macro constants

### 5.2.2.2.1. WDGIF\_AR\_RELEASE\_MAJOR\_VERSION

<b>Purpose</b>	AUTOSAR release major version.
<b>Value</b>	4U

### 5.2.2.2.2. WDGIF\_AR\_RELEASE\_MINOR\_VERSION

<b>Purpose</b>	AUTOSAR release minor version.
<b>Value</b>	0U

### 5.2.2.2.3. WDGIF\_AR\_RELEASE\_REVISION\_VERSION

<b>Purpose</b>	AUTOSAR release revision version.
<b>Value</b>	3U

### 5.2.2.2.4. WDGIF\_E\_INV\_POINTER

<b>Purpose</b>	DET: API service called with null pointer parameter.
<b>Value</b>	0x02U

### 5.2.2.2.5. WDGIF\_E\_PARAM\_DEVICE

<b>Purpose</b>	DET: API service called with wrong device index parameter.
<b>Value</b>	0x01U

### 5.2.2.2.6. WDGIF\_FAST\_MODE

<b>Purpose</b>	In this mode, the watchdog driver is set up for a short timeout period (fast triggering).
----------------	---

<b>Value</b>	2u
--------------	----

#### 5.2.2.2.7. WDGIF\_MODULE\_ID

<b>Purpose</b>	AUTOSAR module identification.
<b>Value</b>	43U

#### 5.2.2.2.8. WDGIF\_OFF\_MODE

<b>Purpose</b>	In this mode, the watchdog driver is disabled (switched off).
<b>Value</b>	0u

#### 5.2.2.2.9. WDGIF\_SID\_GETVERSIONINFO

<b>Purpose</b>	API service id for <a href="#">WdgIf_GetVersionInfo()</a> .
<b>Value</b>	0x03U

#### 5.2.2.2.10. WDGIF\_SID\_SETMODE

<b>Purpose</b>	API service id for <a href="#">WdgIf_SetMode()</a> .
<b>Value</b>	0x01U

#### 5.2.2.2.11. WDGIF\_SID\_SETTRIGGERCOND

<b>Purpose</b>	API service id for <a href="#">WdgIf_SetTriggerCondition()</a> .
<b>Value</b>	0x02U

#### 5.2.2.2.12. WDGIF\_SLOW\_MODE

<b>Purpose</b>	In this mode, the watchdog driver is set up for a long timeout period (slow triggering).
----------------	--

<b>Value</b>	1u
--------------	----

#### 5.2.2.2.13. WDGIF\_SW\_MAJOR\_VERSION

<b>Purpose</b>	AUTOSAR module major version.
<b>Value</b>	6U

#### 5.2.2.2.14. WDGIF\_SW\_MINOR\_VERSION

<b>Purpose</b>	AUTOSAR module minor version.
<b>Value</b>	1U

#### 5.2.2.2.15. WDGIF\_SW\_PATCH\_VERSION

<b>Purpose</b>	AUTOSAR module patch version.
<b>Value</b>	23U

#### 5.2.2.2.16. WDGIF\_VENDOR\_ID

<b>Purpose</b>	AUTOSAR vendor identification: Elektrobit Automotive GmbH.
<b>Value</b>	1U

### 5.2.2.3. Functions

#### 5.2.2.3.1. WdgIf\_GetVersionInfo

<b>Purpose</b>	Get version information of the Watchdog Interface.
<b>Synopsis</b>	<pre>void <b>WdgIf_GetVersionInfo</b> ( Std_VersionInfoType *const VersionInfoPtr );</pre>
<b>Service ID</b>	<a href="#">WDGIF_SID_GETVERSIONINFO</a>
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Reentrant

<b>Parameters (out)</b>	VersionInfoPtr	Pointer to where to store the version information of this module
<b>Description</b>	This service returns the version information of this module.	

#### 5.2.2.3.2. WdgIf\_SetMode

<b>Purpose</b>	Set mode of the Watchdog driver.	
<b>Synopsis</b>	<pre>Std_ReturnType WdgIf_SetMode ( uint8 DeviceIndex , WdgIf_ModeType WdgMode );</pre>	
<b>Service ID</b>	<a href="#">WDGIF_SID_SETMODE</a>	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Non reentrant	
<b>Parameters (in)</b>	DeviceIndex	index of requested watchdog driver
	WdgMode	requested mode of watchdog driver
<b>Parameters (in,out)</b>	DeviceIndex	index of requested watchdog driver
	WdgMode	requested mode of watchdog driver
<b>Return Value</b>	Std_ReturnType	
<b>Description</b>	<p>This function provides access to the mode switching services of the underlying watchdog drivers. The function is mapped to the service Wdg_SetMode().</p> <p>This function is implemented as macro if development error detection is turned off.</p>	

#### 5.2.2.3.3. WdgIf\_SetTriggerCondition

<b>Purpose</b>	Trigger the Watchdog driver.	
<b>Synopsis</b>	<pre>void WdgIf_SetTriggerCondition ( uint8 DeviceIndex , uint16 Timeout );</pre>	
<b>Service ID</b>	<a href="#">WDGIF_SID_SETTRIGGERCOND</a>	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Non reentrant	
<b>Parameters (in)</b>	DeviceIndex	Identifies the Watchdog Driver instance.
	Timeout	Timeout value (milliseconds) for setting the trigger counter.

<b>Description</b>	<p>This service maps the service <code>WdgIf_SetTriggerCondition</code> to the service <code>Wdg_SetTriggerCondition</code> of the corresponding Watchdog Driver.</p> <p>This function is implemented as macro if development error detection is turned off.</p>
--------------------	--

## 5.2.3. Integration notes

### 5.2.3.1. Exclusive areas

Exclusive areas are not used by the `WdgIf` module.

### 5.2.3.2. Production errors

Production errors are not reported by the `WdgIf` module.

### 5.2.3.3. Memory mapping

General information about memory mapping is provided in the EB tresos AutoCore Generic documentation. Refer to the section `Memory mapping and compiler abstraction in the Integration notes` section for details.

The following table provides the list of sections that may be mapped for this module:

Memory section
<code>CODE_ASIL_D</code>
<code>CONST_ASIL_D_UNSPECIFIED</code>

### 5.2.3.4. Integration requirements

**WARNING**



**Integration requirements list is not exhaustive**

The following list of integration requirements helps you to integrate your product. However, this list is not exhaustive. You also require information from the user guide, release notes, and EB tresos AutoCore known issues to successfully integrate your product.

Integration requirements are not listed for the `WdgIf` module.



## 5.3. WdgM

### 5.3.1. Configuration parameters

Containers included		
Container name	Multiplicity	Description
<a href="#">CommonPublishedInformation</a>	1..1	<b>Label:</b> Common Published Information Common container, aggregated by all modules. It contains published information about vendor and versions.
<a href="#">PublishedInformation</a>	1..1	<b>Label:</b> EB Published Information Additional published parameters not covered by Common-PublishedInformation container.
<a href="#">WdgMDefensiveProgramming</a>	1..1	<b>Label:</b> Defensive Programming Options Parameters for defensive programming
<a href="#">ReportToDem</a>	1..1	<b>Label:</b> Production error handling Production error handling
<a href="#">WdgMConfigSet</a>	1..n	This container describes one of multiple configuration sets of WdgM. This is a MultipleConfigurationContainer, i.e. this container and its sub-containers exist once per configuration set.
<a href="#">WdgMGeneral</a>	1..1	<b>Label:</b> General Configuration Parameters Container defines all general configuration parameters of the Watchdog Manager.

Parameters included	
Parameter name	Multiplicity
<a href="#">IMPLEMENTATION_CONFIG_VARIANT</a>	1..1

Parameter Name	IMPLEMENTATION_CONFIG_VARIANT
Label	Config Variant
Multiplicity	1..1
Type	ENUMERATION
Default value	VariantPreCompile

Range	VariantPreCompile
-------	-------------------

### 5.3.1.1. CommonPublishedInformation

Parameters included	
Parameter name	Multiplicity
<a href="#">ArMajorVersion</a>	1..1
<a href="#">ArMinorVersion</a>	1..1
<a href="#">ArPatchVersion</a>	1..1
<a href="#">SwMajorVersion</a>	1..1
<a href="#">SwMinorVersion</a>	1..1
<a href="#">SwPatchVersion</a>	1..1
<a href="#">ModuleId</a>	1..1
<a href="#">VendorId</a>	1..1
<a href="#">Release</a>	1..1

Parameter Name	ArMajorVersion	
Label	AUTOSAR Major Version	
Description	Major version number of AUTOSAR specification on which the appropriate implementation is based on.	
Multiplicity	1..1	
Type	INTEGER_LABEL	
Default value	2	
Configuration class	PublishedInformation:	
Origin	Elektrobit Automotive GmbH	

Parameter Name	ArMinorVersion	
Label	AUTOSAR Minor Version	
Description	Minor version number of AUTOSAR specification on which the appropriate implementation is based on.	
Multiplicity	1..1	
Type	INTEGER_LABEL	
Default value	2	

<b>Configuration class</b>	<b>PublishedInformation:</b>	
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>ArPatchVersion</b>	
<b>Label</b>	AUTOSAR Patch Version	
<b>Description</b>	Patch level version number of AUTOSAR specification on which the appropriate implementation is based on.	
<b>Multiplicity</b>	1..1	
<b>Type</b>	INTEGER_LABEL	
<b>Default value</b>	0	
<b>Configuration class</b>	<b>PublishedInformation:</b>	
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>SwMajorVersion</b>	
<b>Label</b>	Software Major Version	
<b>Description</b>	Major version number of the vendor specific implementation of the module.	
<b>Multiplicity</b>	1..1	
<b>Type</b>	INTEGER_LABEL	
<b>Default value</b>	6	
<b>Configuration class</b>	<b>PublishedInformation:</b>	
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>SwMinorVersion</b>	
<b>Label</b>	Software Minor Version	
<b>Description</b>	Minor version number of the vendor specific implementation of the module. The numbering is vendor specific.	
<b>Multiplicity</b>	1..1	
<b>Type</b>	INTEGER_LABEL	
<b>Default value</b>	1	
<b>Configuration class</b>	<b>PublishedInformation:</b>	
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>SwPatchVersion</b>	
<b>Label</b>	Software Patch Version	

<b>Description</b>	Patch level version number of the vendor specific implementation of the module. The numbering is vendor specific.	
<b>Multiplicity</b>	1..1	
<b>Type</b>	INTEGER_LABEL	
<b>Default value</b>	36	
<b>Configuration class</b>	<b>PublishedInformation:</b>	
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>ModuleId</b>	
<b>Label</b>	Numeric Module ID	
<b>Description</b>	Module ID of this module from Module List	
<b>Multiplicity</b>	1..1	
<b>Type</b>	INTEGER_LABEL	
<b>Default value</b>	13	
<b>Configuration class</b>	<b>PublishedInformation:</b>	
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>VendorId</b>	
<b>Label</b>	Vendor ID	
<b>Description</b>	Vendor ID of the dedicated implementation of this module according to the AUTOSAR vendor list	
<b>Multiplicity</b>	1..1	
<b>Type</b>	INTEGER_LABEL	
<b>Default value</b>	1	
<b>Configuration class</b>	<b>PublishedInformation:</b>	
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>Release</b>	
<b>Label</b>	Release Information	
<b>Multiplicity</b>	1..1	
<b>Type</b>	STRING_LABEL	
<b>Default value</b>		
<b>Configuration class</b>	<b>PublishedInformation:</b>	

<b>Origin</b>	Elektrobit Automotive GmbH
---------------	----------------------------

### 5.3.1.2. PublishedInformation

Parameters included	
Parameter name	Multiplicity
<a href="#">PbcfgMSupport</a>	1..1

<b>Parameter Name</b>	<b>PbcfgMSupport</b>	
<b>Label</b>	PbcfgM support	
<b>Description</b>	Specifies whether or not the WdgM can use the PbcfgM module for post-build support.	
<b>Multiplicity</b>	1..1	
<b>Type</b>	BOOLEAN	
<b>Default value</b>	false	
<b>Configuration class</b>	<b>PublishedInformation:</b>	
<b>Origin</b>	Elektrobit Automotive GmbH	

### 5.3.1.3. WdgMDefensiveProgramming

Parameters included	
Parameter name	Multiplicity
<a href="#">WdgMDefProgEnabled</a>	1..1
<a href="#">WdgMPrecondAssertEnabled</a>	1..1
<a href="#">WdgMPostcondAssertEnabled</a>	1..1
<a href="#">WdgMStaticAssertEnabled</a>	1..1
<a href="#">WdgMUnreachAssertEnabled</a>	1..1
<a href="#">WdgMInvariantAssertEnabled</a>	1..1

<b>Parameter Name</b>	<b>WdgMDefProgEnabled</b>	
<b>Label</b>	Enable Defensive Programming	
<b>Description</b>	<p>Enables or disables the defensive programming feature for the module WdgM.</p> <p>Note: This feature is dependent on the use of the development error detection module. To use the defensive programming feature, proceed as follows:</p>	

	<ol style="list-style-type: none"> <li>1. Enable development error detection</li> <li>2. Enable defensive programming</li> <li>3. Enable assertions as required</li> </ol>	
<b>Multiplicity</b>	1..1	
<b>Type</b>	BOOLEAN	
<b>Default value</b>	false	
<b>Configuration class</b>	<b>VariantPreCompile:</b>	VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>WdgMPrecondAssertEnabled</b>	
<b>Label</b>	Enable Precondition Assertions	
<b>Description</b>	<p>Enables handling of precondition assertion checks reported from the module WdgM.</p> <p>Dependency on parameter(s):</p> <ul style="list-style-type: none"> <li>▶ Enable Development Error Detection (WdgMDevErrorDetect): must be enabled</li> <li>▶ Enable Defensive Programming (WdgMDefProgEnabled): must be enabled</li> </ul>	
<b>Multiplicity</b>	1..1	
<b>Type</b>	BOOLEAN	
<b>Default value</b>	false	
<b>Configuration class</b>	<b>VariantPreCompile:</b>	VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>WdgMPostcondAssertEnabled</b>	
<b>Label</b>	Enable Postcondition Assertions	
<b>Description</b>	<p>Enables handling of postcondition assertion checks reported from the module WdgM.</p> <p>Dependency on parameter(s):</p> <ul style="list-style-type: none"> <li>▶ Enable Development Error Detection (WdgMDevErrorDetect): must be enabled</li> <li>▶ Enable Defensive Programming (WdgMDefProgEnabled): must be enabled</li> </ul>	

<b>Multiplicity</b>	1..1
<b>Type</b>	BOOLEAN
<b>Default value</b>	false
<b>Configuration class</b>	<b>VariantPreCompile:</b> VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH

<b>Parameter Name</b>	<b>WdgMStaticAssertEnabled</b>
<b>Label</b>	Enable Static Assertions
<b>Description</b>	<p>Enables handling of static assertion checks reported from the module WdgM.</p> <p>Dependency on parameter(s):</p> <ul style="list-style-type: none"> <li>▶ Enable Development Error Detection (WdgMDevErrorDetect): must be enabled</li> <li>▶ Enable Defensive Programming (WdgMDefProgEnabled): must be enabled</li> </ul>
<b>Multiplicity</b>	1..1
<b>Type</b>	BOOLEAN
<b>Default value</b>	false
<b>Configuration class</b>	<b>VariantPreCompile:</b> VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH

<b>Parameter Name</b>	<b>WdgMUnreachAssertEnabled</b>
<b>Label</b>	Enable Unreachable Code Assertions
<b>Description</b>	<p>Enables handling of unreachable code assertion checks reported from the module WdgM.</p> <p>Dependency on parameter(s):</p> <ul style="list-style-type: none"> <li>▶ Enable Development Error Detection (WdgMDevErrorDetect): must be enabled</li> <li>▶ Enable Defensive Programming (WdgMDefProgEnabled): must be enabled</li> </ul>
<b>Multiplicity</b>	1..1
<b>Type</b>	BOOLEAN
<b>Default value</b>	false
<b>Configuration class</b>	<b>VariantPreCompile:</b> VariantPreCompile

<b>Origin</b>	Elektrobit Automotive GmbH	
<b>Parameter Name</b>	<b>WdgMInvariantAssertEnabled</b>	
<b>Label</b>	Enable Invariant Assertions	
<b>Description</b>	<p>Enables handling of invariant assertion checks reported from functions of the module WdgM.</p> <p>Dependency on parameter(s):</p> <ul style="list-style-type: none"> <li>▶ Enable Development Error Detection (WdgMDevErrorDetect): must be enabled</li> <li>▶ Enable Defensive Programming (WdgMDefProgEnabled): must be enabled</li> </ul>	
<b>Multiplicity</b>	1..1	
<b>Type</b>	BOOLEAN	
<b>Default value</b>	false	
<b>Configuration class</b>	<b>VariantPreCompile:</b>	VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH	

### 5.3.1.4. ReportToDem

Parameters included		
Parameter name	Multiplicity	
<a href="#">WdgMSupervisionReportToDem</a>	1..1	
<a href="#">WdgMSupervisionDemDetErrId</a>	1..1	
<a href="#">WdgMSetModeReportToDem</a>	1..1	
<a href="#">WdgMSetModeDemDetErrId</a>	1..1	
<a href="#">WdgMImproperCallerReportToDem</a>	1..1	
<a href="#">WdgMImproperCallerDemDetErrId</a>	1..1	
<a href="#">WdgMMFTimingViolationReportToDem</a>	1..1	
<a href="#">WdgMMFTimingViolationDemDetErrId</a>	1..1	
<a href="#">WdgMDataCorruptionReportToDem</a>	1..1	
<a href="#">WdgMDataCorruptionDemDetErrId</a>	1..1	

<b>Parameter Name</b>	<b>WdgMSupervisionReportToDem</b>	
<b>Label</b>	WdgM Supervision Failure	



<b>Description</b>	Selects the handling of the production error: <i>WDGM_E_MONITORING</i> (formerly <i>WDGM_E_SUPERVISION</i> ) <ul style="list-style-type: none"> <li>▶ <b>DEM:</b> The production error <i>WDGM_E_MONITORING</i> is reported to the Diagnostics Event Manager (Dem).</li> <li>▶ <b>DET:</b> The production error <i>WDGM_E_MONITORING</i> is reported to the Development Error Tracer (Det) if enabled.</li> <li>▶ <b>DISABLE:</b> The production error <i>WDGM_E_MONITORING</i> is not reported at all.</li> </ul>	
<b>Multiplicity</b>	1..1	
<b>Type</b>	ENUMERATION	
<b>Default value</b>	DEM	
<b>Range</b>	DEM	
	DET	
	DISABLE	
<b>Configuration class</b>	<b>PreCompile:</b>	VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>WdgMSupervisionDemDetErrId</b>	
<b>Label</b>	WdgM Supervision Failure DemToDet ErrorId	
<b>Description</b>	If a production error is reported towards the Det, this parameter defines the error id which is reported towards the Det.  The preprocessor define <i>WDGM_E_DEMTODET_E_SUPERVISION</i> is generated holding the value of <i>WdgMSupervisionDemDetErrId</i> .	
<b>Multiplicity</b>	1..1	
<b>Type</b>	INTEGER	
<b>Default value</b>	30	
<b>Range</b>	<=255	
	>=30	
<b>Configuration class</b>	<b>PreCompile:</b>	VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>WdgMSetModeReportToDem</b>	
<b>Label</b>	WdgM SetMode Failure	
<b>Description</b>	Selects the handling of the production error: <i>WDGM_E_SET_MODE</i>	

	<ul style="list-style-type: none"> <li>► <b>DEM:</b> The production error <code>WDGM_E_SET_MODE</code> is reported to the Diagnostics Event Manager (Dem).</li> <li>► <b>DET:</b> The production error <code>WDGM_E_SET_MODE</code> is reported to the Development Error Tracer (Det) if enabled.</li> <li>► <b>DISABLE:</b> The production error <code>WDGM_E_SET_MODE</code> is not reported at all.</li> </ul>	
<b>Multiplicity</b>	1..1	
<b>Type</b>	ENUMERATION	
<b>Default value</b>	DEM	
<b>Range</b>	DEM	
	DET	
	DISABLE	
<b>Configuration class</b>	<b>VariantPreCompile:</b>	VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>WdgMSetModeDemDetErrId</b>	
<b>Label</b>	WdgM SetMode Failure DemToDet ErrorId	
<b>Description</b>	<p>If a production error is reported towards the Det, this parameter defines the error id which is reported towards the Det.</p> <p>The preprocessor define <code>WDGM_EB_E_DEMTODET_SET_MODE</code> is generated holding the value of <code>WdgMSetModeDemDetErrId</code>.</p>	
<b>Multiplicity</b>	1..1	
<b>Type</b>	INTEGER	
<b>Default value</b>	30	
<b>Range</b>	<=255	
	>=30	
<b>Configuration class</b>	<b>PreCompile:</b>	VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>WdgMImproperCallerReportToDem</b>	
<b>Label</b>	WdgM ImproperCaller Failure	
<b>Description</b>	<p>Selects the handling of the production error: <code>WDGM_E_IMPROPER_CALLER</code></p> <ul style="list-style-type: none"> <li>► <b>DEM:</b> The production error <code>WDGM_E_IMPROPER_CALLER</code> is reported to the Diagnostics Event Manager (Dem).</li> </ul>	

	<ul style="list-style-type: none"> <li>▶ <b>DET:</b> The production error <code>WDGM_E_IMPROPER_CALLER</code> is reported to the Development Error Tracer (Det) if enabled.</li> <li>▶ <b>DISABLE:</b> The production error <code>WDGM_E_IMPROPER_CALLER</code> is not reported at all.</li> </ul>	
<b>Multiplicity</b>	1..1	
<b>Type</b>	ENUMERATION	
<b>Default value</b>	DEM	
<b>Range</b>	DEM	
	DET	
	DISABLE	
<b>Configuration class</b>	<b>PreCompile:</b>	VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>WdgMImproperCallerDemDetErrId</b>	
<b>Label</b>	WdgM ImproperCaller Failure DemToDet ErrorId	
<b>Description</b>	<p>If a production error is reported towards the Det, this parameter defines the error id which is reported towards the Det.</p> <p>The preprocessor define <code>WDGM_EB_E_DEMTODET_IMPROPER_CALLER</code> is generated holding the value of <code>WdgMImproperCallerDemDetErrId</code>.</p>	
<b>Multiplicity</b>	1..1	
<b>Type</b>	INTEGER	
<b>Default value</b>	30	
<b>Range</b>	<=255	
	>=30	
<b>Configuration class</b>	<b>PreCompile:</b>	VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>WdgMMFTimingViolationReportToDem</b>	
<b>Label</b>	WdgM MainFunction Timing Failure	
<b>Description</b>	<p>Selects the handling of the production error: <code>WDGM_E_MF_TIMINGVIOLATION</code></p> <ul style="list-style-type: none"> <li>▶ <b>DEM:</b> The production error <code>WDGM_E_MF_TIMINGVIOLATION</code> is reported to the Diagnostics Event Manager (Dem).</li> <li>▶ <b>DET:</b> The production error <code>WDGM_E_MF_TIMINGVIOLATION</code> is reported to the Development Error Tracer (Det) if enabled.</li> </ul>	

	▶ <b>DISABLE:</b> The production error <code>WDGM_E_MF_TIMINGVIOLATION</code> is not reported at all.	
<b>Multiplicity</b>	1..1	
<b>Type</b>	ENUMERATION	
<b>Default value</b>	DEM	
<b>Range</b>	DEM	
	DET	
	DISABLE	
<b>Configuration class</b>	<b>PreCompile:</b>	VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>WdgMMFTimingViolationDemDetErrId</b>	
<b>Label</b>	WdgM MainFunction Timing Failure DemToDet ErrorId	
<b>Description</b>	<p>If a production error is reported towards the Det, this parameter defines the error id which is reported towards the Det.</p> <p>The preprocessor define <code>WDGM_E_DEMTODET_E_MF_TIMINGVIOLATION</code> is generated holding the value of <code>WdgMMFTimingViolationDemDetErrId</code>.</p>	
<b>Multiplicity</b>	1..1	
<b>Type</b>	INTEGER	
<b>Default value</b>	30	
<b>Range</b>	<=255	
	>=30	
<b>Configuration class</b>	<b>PreCompile:</b>	VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>WdgMDataCorruptionReportToDem</b>	
<b>Label</b>	WdgM Data Corruption Failure	
<b>Description</b>	<p>Selects the handling of the production error: <code>WDGM_E_DATA_CORRUPTION</code></p> <ul style="list-style-type: none"> <li>▶ <b>DEM:</b> The production error <code>WDGM_E_DATA_CORRUPTION</code> is reported to the Diagnostics Event Manager (Dem).</li> <li>▶ <b>DET:</b> The production error <code>WDGM_E_DATA_CORRUPTION</code> is reported to the Development Error Tracer (Det) if enabled.</li> <li>▶ <b>DISABLE:</b> The production error <code>WDGM_E_DATA_CORRUPTION</code> is not reported at all.</li> </ul>	

<b>Multiplicity</b>	1..1	
<b>Type</b>	ENUMERATION	
<b>Default value</b>	DEM	
<b>Range</b>	DEM	
	DET	
	DISABLE	
<b>Configuration class</b>	<b>VariantPreCompile:</b>	VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>WdgMDataCorruptionDemDetErrId</b>	
<b>Label</b>	WdgM Data Corruption Failure DemToDet ErrorId	
<b>Description</b>	<p>If a production error is reported towards the Det, this parameter defines the error id which is reported towards the Det.</p> <p>The preprocessor define <code>WDGM_E_DEMTODET_E_DATA_CORRUPTION</code> is generated holding the value of <code>WdgMDataCorruptionDemDetErrId</code>.</p>	
<b>Multiplicity</b>	1..1	
<b>Type</b>	INTEGER	
<b>Default value</b>	30	
<b>Range</b>	<=255	
	>=30	
<b>Configuration class</b>	<b>PreCompile:</b>	VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH	

### 5.3.1.5. WdgMConfigSet

Containers included		
Container name	Multiplicity	Description
<a href="#">WdgMDemEventParameter-Refs</a>	1..1	Container for the references to DemEventParameter elements which shall be invoked using the API <code>Dem_ReportErrorStatus</code> API in case the corresponding error occurs. The EventId is taken from the referenced DemEventParameter's DemEventId value. The standardized errors are provided in the container and can be extended by vendor specific error references.

Containers included		
<a href="#">WdgMMode</a>	1..255	The container describes one of several modes of the Watchdog Manager.

Parameters included	
Parameter name	Multiplicity
<a href="#">WdgMInitialMode</a>	1..1
<a href="#">WdgMSleepMode</a>	0..1

Parameter Name	WdgMInitialMode	
Description	The mode that the Watchdog Manager is in after it has been initialized.	
Multiplicity	1..1	
Type	SYMBOLIC-NAME-REFERENCE	
Configuration class	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	WdgMSleepMode	
Description	The Watchdog Manager switches to this WdgM mode at the beginning of WdgM_DeInit and executes the configured trigger conditions before finally deactivating the Watchdog Manager. This final mode switch usually has no Supervision Entities configured and may be necessary for a relaxed Watchdog triggering during the shutdown or sleep phase. If parameter is disabled, this final mode switch has to be performed externally via the WdgM_SetMode() API and at least one main function has to be executed afterwards such that the Watchdog Manager calls the required trigger conditions before finally executing WdgM_DeInit.	
Multiplicity	0..1	
Type	SYMBOLIC-NAME-REFERENCE	
Configuration class	PreCompile:	VariantPreCompile
Origin	Elektrobit Automotive GmbH	

### 5.3.1.6. WdgMDemEventParameterRefs

Parameters included	
Parameter name	Multiplicity
<a href="#">WDGM_E_DATA_CORRUPTION</a>	1..1
<a href="#">WDGM_E_IMPROPER_CALLER</a>	1..1

Parameters included	
<a href="#">WDGM_E_MF_TIMINGVIOLATION</a>	1..1
<a href="#">WDGM_E_MONITORING</a>	1..1
<a href="#">WDGM_E_SET_MODE</a>	1..1

Parameter Name	WDGM_E_DATA_CORRUPTION	
Description	<p>Reference to the DemEventParameter that shall be issued when data corruption is detected in the internal WdgM data, which is stored double inverse.</p> <p>Dependency on parameter(s):</p> <ul style="list-style-type: none"> <li>▶ <code>WdgMDataCorruptionReportToDem</code>: Select DEM to enable the reporting of <code>WDGM_E_DATA_CORRUPTION</code>.</li> </ul> <p>Further notes:</p> <ul style="list-style-type: none"> <li>▶ Activation: Thrown, if data corruption is detected in the internal WdgM data.</li> <li>▶ Healing: None. The error resides in memory until it is deleted.</li> <li>▶ Trigger debounce: None. The error is reported on first occurrence.</li> <li>▶ Rate of diagnostic checks: Checked on every call of the service that reports this error.</li> </ul>	
Multiplicity	1..1	
Type	SYMBOLIC-NAME-REFERENCE	
Configuration class	PreCompile:	VariantPreCompile
Origin	Elektrobit Automotive GmbH	

Parameter Name	WDGM_E_IMPROPER_CALLER	
Description	<p>Reference to the DemEventParameter that shall be issued when the defensive behavior checks detect an improper caller.</p> <p>Dependency on parameter(s):</p> <ul style="list-style-type: none"> <li>▶ <code>WdgMImproperCallerReportToDem</code>: Select DEM to enable the reporting of <code>WDGM_E_IMPROPER_CALLER</code>.</li> </ul> <p>Further notes:</p> <ul style="list-style-type: none"> <li>▶ Activation: Thrown, if the passed CallerID is not in the list of the configured list of allowed CallerIDs.</li> <li>▶ Healing: None. The error resides in memory until it is deleted.</li> <li>▶ Trigger debounce: None. The error is reported on first occurrence.</li> </ul>	

	► Rate of diagnostic checks: Checked on every call of the service that reports this error.
<b>Multiplicity</b>	1..1
<b>Type</b>	SYMBOLIC-NAME-REFERENCE
<b>Configuration class</b>	<b>VariantPreCompile:</b> VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC

<b>Parameter Name</b>	<b>WDGM_E_MF_TIMINGVIOLATION</b>
<b>Description</b>	<p>Reference to the DemEventParameter that shall be issued when the actual Watchdog Manager main function period deviates from the configured mode-dependent schedule period (<code>WdgMSupervisionCycle</code>).</p> <p>Dependency on parameter(s):</p> <ul style="list-style-type: none"> <li>► <code>WdgMMFTimingViolationReportToDem</code>: Select DEM to enable the reporting of <code>WDGM_E_MF_TIMINGVIOLATION</code>.</li> </ul> <p>Further notes:</p> <ul style="list-style-type: none"> <li>► Activation: Thrown, if the <code>WdgM_MainFunction()</code> period deviates from the configured mode-dependent schedule period (<code>WdgMSupervisionCycle</code>).</li> <li>► Healing: None. The error resides in memory until it is deleted.</li> <li>► Trigger debounce: None. The error is reported on first occurrence.</li> <li>► Rate of diagnostic checks: Checked cyclically within <code>WdgM_MainFunction()</code>.</li> </ul>
<b>Multiplicity</b>	1..1
<b>Type</b>	SYMBOLIC-NAME-REFERENCE
<b>Configuration class</b>	<b>PreCompile:</b> VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH

<b>Parameter Name</b>	<b>WDGM_E_MONITORING</b>
<b>Description</b>	<p>Reference to the DemEventParameter that shall be issued when the following error occurs: <i>Monitoring has failed and a watchdog reset will occur.</i></p> <p>Dependency on parameter(s):</p> <ul style="list-style-type: none"> <li>► <code>WdgMSupervisionReportToDem</code>: Select DEM to enable the reporting of <code>WDGM_E_MONITORING</code>.</li> </ul> <p>Further notes:</p>



	<ul style="list-style-type: none"> <li>▶ Activation: Thrown, if supervision fails for a supervised entity.</li> <li>▶ Healing: None. The error resides in memory until it is deleted.</li> <li>▶ Trigger debounce: None. The error is reported on first occurrence.</li> <li>▶ Rate of diagnostic checks: Checked cyclically within <code>WdgM_MainFunction()</code>.</li> </ul>	
<b>Multiplicity</b>	1..1	
<b>Type</b>	SYMBOLIC-NAME-REFERENCE	
<b>Configuration class</b>	<b>VariantPreCompile:</b>	VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC	

<b>Parameter Name</b>	<b>WDGM_E_SET_MODE</b>	
<b>Description</b>	<p>Reference to the DemEventParameter that shall be issued when the following error occurs: <i>Watchdog drivers' mode switch has failed</i>.</p> <p>Dependency on parameter(s):</p> <ul style="list-style-type: none"> <li>▶ <code>WdgMSetModeReportToDem</code>: Select DEM to enable the reporting of <code>WDGM_E_SET_MODE</code>.</li> </ul> <p>Further notes:</p> <ul style="list-style-type: none"> <li>▶ Activation: Thrown, if watchdog drivers' mode switch has failed.</li> <li>▶ Healing: None. The error resides in memory until it is deleted.</li> <li>▶ Trigger debounce: None. The error is reported on first occurrence.</li> <li>▶ Rate of diagnostic checks: Checked cyclically within <code>WdgM_MainFunction()</code>.</li> </ul>	
<b>Multiplicity</b>	1..1	
<b>Type</b>	SYMBOLIC-NAME-REFERENCE	
<b>Configuration class</b>	<b>VariantPreCompile:</b>	VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC	

### 5.3.1.7. WdgMMode

Containers included		
Container name	Multiplicity	Description
<a href="#">WdgMAliveSupervision</a>	0..65535	This container collects all configuration parameters of Alive-Supervision of one Checkpoint. Note that each Checkpoint

Containers included		
		may have different parameters. For example, it may have different min and max margin.
<a href="#">WdgMDeadlineSupervision</a>	0..65535	This container collects all configuration parameters for Deadline Supervision for a Supervised Entity.
<a href="#">WdgMExternalLogicalSupervision</a>	0..65535	This container collects all configuration parameters for Logical Supervision for one external graph.
<a href="#">WdgMLocalStatusParams</a>	0..65535	This container collects all configuration parameters for the Local Status of a Supervised Entity.
<a href="#">WdgMTrigger</a>	0..255	This container collects all configuration parameters for the triggering of hardware watchdogs.

Parameters included	
Parameter name	Multiplicity
<a href="#">WdgMExpiredSupervisionCycleTol</a>	1..1
<a href="#">WdgMModelId</a>	1..1
<a href="#">WdgMSupervisionCycle</a>	1..1

Parameter Name	WdgMExpiredSupervisionCycleTol	
Description	This parameter shall be used to define a value that fixes the amount of expired supervision cycles for how long the blocking of watchdog triggering shall be postponed, AFTER THE GLOBAL SUPERVISION STATUS HAS REACHED THE STATE EXPIRED.	
Multiplicity	1..1	
Type	INTEGER	
Default value	5	
Range	<div>&lt;=65535</div> <div>&gt;=0</div>	
Configuration class	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	WdgMModelId
Label	WdgMModelId (0 -> 255)
Description	This parameter fixes the identifier for the mode. This identifier is for instance passed as a parameter to the WdgM_SetMode service.
Multiplicity	1..1

<b>Type</b>	INTEGER	
<b>Configuration class</b>	<b>VariantPreCompile:</b>	VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC	

<b>Parameter Name</b>	<b>WdgMSupervisionCycle</b>	
<b>Description</b>	This parameter defines the schedule period of the main function WdgM_Main-Function. Unit: [s]	
<b>Multiplicity</b>	1..1	
<b>Type</b>	FLOAT	
<b>Default value</b>	0.01	
<b>Configuration class</b>	<b>VariantPreCompile:</b>	VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC	

### 5.3.1.8. WdgMAliveSupervision

Parameters included	
Parameter name	Multiplicity
<a href="#">WdgMExpectedAliveIndications</a>	1..1
<a href="#">WdgMMaxMargin</a>	1..1
<a href="#">WdgMMinMargin</a>	1..1
<a href="#">WdgMSupervisionReferenceCycle</a>	1..1
<a href="#">WdgMAliveSupervisionCheckpointRef</a>	1..1

<b>Parameter Name</b>	<b>WdgMExpectedAliveIndications</b>	
<b>Description</b>	This parameter contains the amount of expected alive indications of the Checkpoint within the referenced amount of defined supervision cycles according to corresponding SE.	
<b>Multiplicity</b>	1..1	
<b>Type</b>	INTEGER	
<b>Default value</b>	1	
<b>Range</b>	<=65535	
	>=0	
<b>Configuration class</b>	<b>VariantPreCompile:</b>	VariantPreCompile

<b>Origin</b>	AUTOSAR_ECUC
---------------	--------------

<b>Parameter Name</b>	<b>WdgMMaxMargin</b>	
<b>Label</b>	WdgMMaxMargin (0 -> 255)	
<b>Description</b>	This parameter contains the amount of alive indications of the Checkpoint that are acceptable to be additional to the expected alive indications within the corresponding supervision reference cycle.	
<b>Multiplicity</b>	1..1	
<b>Type</b>	INTEGER	
<b>Default value</b>	0	
<b>Configuration class</b>	<b>VariantPreCompile:</b>	VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC	

<b>Parameter Name</b>	<b>WdgMMinMargin</b>	
<b>Label</b>	WdgMMinMargin (0 -> 255)	
<b>Description</b>	This parameter contains the amount of alive indications of the Checkpoint that are acceptable to be missed from the expected alive indications within the corresponding supervision reference cycle.	
<b>Multiplicity</b>	1..1	
<b>Type</b>	INTEGER	
<b>Default value</b>	0	
<b>Configuration class</b>	<b>VariantPreCompile:</b>	VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC	

<b>Parameter Name</b>	<b>WdgMSupervisionReferenceCycle</b>	
<b>Description</b>	This parameter shall contain the amount of supervision cycles to be used as reference by the alive-supervision mechanism to perform the checkup with counted alive indications according to corresponding SE.	
<b>Multiplicity</b>	1..1	
<b>Type</b>	INTEGER	
<b>Default value</b>	1	
<b>Range</b>	<=65535	
	>=1	
<b>Configuration class</b>	<b>VariantPreCompile:</b>	VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC	

Parameter Name	WdgMAliveSupervisionCheckpointRef	
Description	Reference to Checkpoint within a Supervised Entity that shall be supervised.	
Multiplicity	1..1	
Type	SYMBOLIC-NAME-REFERENCE	
Configuration class	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

### 5.3.1.9. WdgMDeadlineSupervision

Parameters included	
Parameter name	Multiplicity
<a href="#">WdgMDeadlineMax</a>	1..1
<a href="#">WdgMDeadlineMin</a>	1..1
<a href="#">WdgMDeadlineStartRef</a>	1..1
<a href="#">WdgMDeadlineStopRef</a>	1..1

Parameter Name	WdgMDeadlineMax	
Description	This parameter contains the longest time span after which the deadline is considered to be met. Unit: [s]	
Multiplicity	1..1	
Type	FLOAT	
Configuration class	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	WdgMDeadlineMin	
Description	This parameter contains the shortest time span after which the deadline is considered to be met. Unit: [s]	
Multiplicity	1..1	
Type	FLOAT	
Configuration class	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	WdgMDeadlineStartRef	
Description	This is the reference to the start Checkpoint for Deadline Supervision.	

<b>Multiplicity</b>	1..1
<b>Type</b>	SYMBOLIC-NAME-REFERENCE
<b>Configuration class</b>	<b>VariantPreCompile:</b> VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC

<b>Parameter Name</b>	<b>WdgMDeadlineStopRef</b>
<b>Description</b>	This is the reference to the stop Checkpoint for Deadline Supervision.
<b>Multiplicity</b>	1..1
<b>Type</b>	SYMBOLIC-NAME-REFERENCE
<b>Configuration class</b>	<b>VariantPreCompile:</b> VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC

### 5.3.1.10. WdgMExternalLogicalSupervision

Containers included		
Container name	Multiplicity	Description
<a href="#">WdgMExternalTransition</a>	0..65535	This container collects the Checkpoints for an External Transition across Supervised Entities.

Parameters included	
Parameter name	Multiplicity
<a href="#">WdgMExternalCheckpointFinalRef</a>	1..65535
<a href="#">WdgMExternalCheckpointInitialRef</a>	1..65535

<b>Parameter Name</b>	<b>WdgMExternalCheckpointFinalRef</b>
<b>Description</b>	This is the reference to the final Checkpoint(s) for this External Graph.
<b>Multiplicity</b>	1..65535
<b>Type</b>	SYMBOLIC-NAME-REFERENCE
<b>Configuration class</b>	<b>PreCompile:</b> VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC

<b>Parameter Name</b>	<b>WdgMExternalCheckpointInitialRef</b>
<b>Description</b>	This is the reference to the initial Checkpoint(s) for this External Graph.
<b>Multiplicity</b>	1..65535

Type	SYMBOLIC-NAME-REFERENCE	
Configuration class	PreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

### 5.3.1.11. WdgMExternalTransition

Parameters included	
Parameter name	Multiplicity
<a href="#">WdgMExternalTransitionDestRef</a>	1..1
<a href="#">WdgMExternalTransitionSourceRef</a>	1..1

Parameter Name	WdgMExternalTransitionDestRef	
Description	This is the reference to the destination Checkpoint of an External Transition.	
Multiplicity	1..1	
Type	SYMBOLIC-NAME-REFERENCE	
Configuration class	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	WdgMExternalTransitionSourceRef	
Description	This is the reference to the source Checkpoint of an External Transition.	
Multiplicity	1..1	
Type	SYMBOLIC-NAME-REFERENCE	
Configuration class	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

### 5.3.1.12. WdgMLocalStatusParams

Parameters included	
Parameter name	Multiplicity
<a href="#">WdgMFailedAliveSupervisionRefCycleTol</a>	1..1
<a href="#">WdgMLocalStatusSupervisedEntityRef</a>	1..1

Parameter Name	WdgMFailedAliveSupervisionRefCycleTol
----------------	---------------------------------------

<b>Description</b>	This parameter shall contain the acceptable amount of reference cycles with incorrect/failed alive supervisions for this Supervised Entity.	
<b>Multiplicity</b>	1..1	
<b>Type</b>	INTEGER	
<b>Range</b>	<=255	
	>=0	
<b>Configuration class</b>	<b>VariantPreCompile:</b>	VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC	

<b>Parameter Name</b>	<b>WdgMLocalStatusSupervisedEntityRef</b>	
<b>Description</b>	This is the reference to the Supervised Entity for which the Local Status parameters are specified.	
<b>Multiplicity</b>	1..1	
<b>Type</b>	SYMBOLIC-NAME-REFERENCE	
<b>Configuration class</b>	<b>VariantPreCompile:</b>	VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC	

### 5.3.1.13. WdgMTrigger

Parameters included	
Parameter name	Multiplicity
<a href="#">WdgMTriggerConditionValue</a>	1..1
<a href="#">WdgMWatchdogMode</a>	1..1
<a href="#">WdgMTriggerWatchdogRef</a>	1..1

<b>Parameter Name</b>	<b>WdgMTriggerConditionValue</b>	
<b>Description</b>	This parameter shall contain the value that is passed to WdgIf_SetTriggerCondition for this watchdog.	
<b>Multiplicity</b>	1..1	
<b>Type</b>	INTEGER	
<b>Range</b>	<=65535	
	>=1	
<b>Configuration class</b>	<b>VariantPreCompile:</b>	VariantPreCompile



<b>Origin</b>	AUTOSAR_ECUC	
<b>Parameter Name</b>	<b>WdgMWatchdogMode</b>	
<b>Description</b>	This parameter contains the watchdog mode that shall be used for the referenced watchdog in this Watchdog Manager mode. Implementation Type: WdgIf_ModeType	
<b>Multiplicity</b>	1..1	
<b>Type</b>	ENUMERATION	
<b>Default value</b>	WDGIF_FAST_MODE	
<b>Range</b>	WDGIF_FAST_MODE	
	WDGIF_OFF_MODE	
	WDGIF_SLOW_MODE	
<b>Configuration class</b>	<b>VariantPreCompile:</b>	VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC	

<b>Parameter Name</b>	<b>WdgMTriggerWatchdogRef</b>	
<b>Description</b>	This parameter is a reference to the configured watchdog.	
<b>Multiplicity</b>	1..1	
<b>Type</b>	SYMBOLIC-NAME-REFERENCE	
<b>Configuration class</b>	<b>VariantPreCompile:</b>	VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC	

#### 5.3.1.14. WdgMGeneral

Containers included		
Container name	Multiplicity	Description
<a href="#">WdgMCallerIds</a>	0..1	Contains the definition of valid CallerIds for the callers who have permission to call the function WdgM_SetMode.
<a href="#">WdgMSupervisedEntity</a>	1..65535	This container collects all common (mode-independent) parameters of a Supervised Entity to be supervised by the Watchdog Manager.
<a href="#">WdgMGeneralMulticore</a>	1..1	<b>Label:</b> General Multicore Configuration Parameters
<a href="#">WdgMServiceAPI</a>	1..1	<b>Label:</b> Service API Parameters  Container for configuration of the service API of WdgM.

Containers included		
<a href="#">WdgMSupervisorCallouts</a>	0..1	Enables the configuration of callouts for WdgM APIs and integration of a trusted component (e.g. &Supervisor;).
<a href="#">WdgMWatchdog</a>	0..255	This container collects all common (mode-independent) parameters of a Watchdog to be triggered by the Watchdog Manager.

Parameters included	
Parameter name	Multiplicity
<a href="#">WdgMDefensiveBehavior</a>	1..1
<a href="#">WdgMDemStoppedSupervisionReport</a>	1..1
<a href="#">WdgMDevErrorDetect</a>	1..1
<a href="#">WdgMImmediateReset</a>	1..1
<a href="#">WdgMOffModeEnabled</a>	1..1
<a href="#">WdgMVersionInfoApi</a>	1..1
<a href="#">WdgMSetModeSynchron</a>	1..1
<a href="#">WdgMRteUsage</a>	1..1
<a href="#">WdgMBSWCompatibilityMode</a>	1..1
<a href="#">WdgMPartitioningEnabled</a>	1..1

Parameter Name	WdgMDefensiveBehavior	
Description	Preprocessor switch to enable/disable the defensive behavior of the Watchdog Manager module.	
Multiplicity	1..1	
Type	BOOLEAN	
Default value	true	
Configuration class	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	WdgMDemStoppedSupervisionReport
Description	<p>Parameter to enable/disable the error reporting to DEM.</p> <ul style="list-style-type: none"> <li>▶ true: A notification to DEM is sent if the Watchdog Manager reaches the state WDG_M_GLOBAL_STATUS_STOPPED.</li> <li>▶ false: The notification is disabled.</li> </ul>
Multiplicity	1..1

<b>Type</b>	BOOLEAN
<b>Default value</b>	true
<b>Configuration class</b>	<b>VariantPreCompile:</b> VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC

<b>Parameter Name</b>	<b>WdgMDevErrorDetect</b>
<b>Description</b>	<p>Preprocessor switch to enable/disable development error detection and reporting.</p> <p>Shall be used to remove unneeded code segments regarding DET features</p> <ul style="list-style-type: none"> <li>▶ true: Development error detection is enabled</li> <li>▶ false: Development error detection is disabled</li> </ul>
<b>Multiplicity</b>	1..1
<b>Type</b>	BOOLEAN
<b>Default value</b>	true
<b>Configuration class</b>	<b>VariantPreCompile:</b> VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC

<b>Parameter Name</b>	<b>WdgMImmediateReset</b>
<b>Description</b>	<p>This parameter enables/disables the immediate reset feature in case of alive-supervision failure.</p> <ul style="list-style-type: none"> <li>▶ true: Immediate reset is enabled</li> <li>▶ false: Immediate reset is disabled</li> </ul>
<b>Multiplicity</b>	1..1
<b>Type</b>	BOOLEAN
<b>Default value</b>	false
<b>Configuration class</b>	<b>VariantPreCompile:</b> VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC

<b>Parameter Name</b>	<b>WdgMOffModeEnabled</b>
<b>Description</b>	<p>This parameter enables/disables the selection of the "OffMode" of the watchdog driver.</p> <ul style="list-style-type: none"> <li>▶ true: "OffMode" selection is allowed</li> <li>▶ false: "OffMode" selection is disallowed</li> </ul>

<b>Multiplicity</b>	1..1
<b>Type</b>	BOOLEAN
<b>Default value</b>	true
<b>Configuration class</b>	<b>VariantPreCompile:</b> VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC

<b>Parameter Name</b>	<b>WdgMVersionInfoApi</b>
<b>Description</b>	Preprocessor switch to enable/disable the existence of the API WdgM_GetVersionInfo. Shall be used to remove unneeded code segments.  <ul style="list-style-type: none"> <li>▶ true: API is enabled</li> <li>▶ false: API is disabled</li> </ul>
<b>Multiplicity</b>	1..1
<b>Type</b>	BOOLEAN
<b>Default value</b>	false
<b>Configuration class</b>	<b>VariantPreCompile:</b> VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC

<b>Parameter Name</b>	<b>WdgMSetModeSynchron</b>
<b>Description</b>	This parameter enable WdgM_SetMode synchronously switch to the new mode.  This behavior is available for the case when the Supervisor Proxy is not used and multicore is disabled.
<b>Multiplicity</b>	1..1
<b>Type</b>	BOOLEAN
<b>Default value</b>	false
<b>Configuration class</b>	<b>VariantPreCompile:</b> VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH

<b>Parameter Name</b>	<b>WdgMRteUsage</b>
<b>Description</b>	This parameter enables the usage of the RTE for this module.  For an easy integration it is recommended to disable the usage of the RTE at the beginning of the integration work.
<b>Multiplicity</b>	1..1
<b>Type</b>	BOOLEAN

<b>Default value</b>	false
<b>Configuration class</b>	<b>VariantPreCompile:</b> VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH

<b>Parameter Name</b>	<b>WdgMBSWCompatibilityMode</b>
<b>Description</b>	<p>Configures whether the AUTOSAR BSW APIs shall be compatible to AUTOSAR 3.2, or AUTOSAR 4.0.</p> <ul style="list-style-type: none"> <li>▶ <code>AUTOSAR_32</code> = AUTOSAR 3.2 BSW APIs are provided (e.g. WdgM_Set-Mode without CallerId)</li> <li>▶ <code>AUTOSAR_40</code> = AUTOSAR 4.0 BSW APIs are provided (e.g. WdgM_Set-Mode with CallerId)</li> </ul>
<b>Multiplicity</b>	1..1
<b>Type</b>	ENUMERATION
<b>Default value</b>	AUTOSAR_40
<b>Range</b>	<div>AUTOSAR_32</div> <div>AUTOSAR_40</div>
<b>Configuration class</b>	<b>VariantPreCompile:</b> VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH

<b>Parameter Name</b>	<b>WdgMPartitioningEnabled</b>
<b>Description</b>	<p>This parameter defines whether the environment makes use of partitioning. In this case, the WdgM provides a separate non-AUTOSAR conform memory abstraction which must be configured by the integrator to be read/write accessible from the execution context of all other partitions / Software Components.</p> <ul style="list-style-type: none"> <li>▶ <b>Enabled:</b> WdgM runtime data which must be written from different callers outside the execution context of the WdgM_Mainfunction (e.g. within the execution context of a Software Component due to the call to WdgM_CheckpointReached) is allocated to a separate non-AUTOSAR conform memory abstraction identifier.</li> <li>▶ <b>Disabled:</b> Complete WdgM runtime data are allocated to AUTOSAR conform memory abstraction identifiers.</li> </ul>
<b>Multiplicity</b>	1..1
<b>Type</b>	BOOLEAN
<b>Default value</b>	false
<b>Configuration class</b>	<b>VariantPreCompile:</b> VariantPreCompile

<b>Origin</b>	Elektrobit Automotive GmbH
---------------	----------------------------

### 5.3.1.15. WdgMCallerIds

Parameters included		
Parameter name	Multiplicity	
<a href="#">WdgMCallerId</a>	0..255	

Parameter Name	WdgMCallerId	
Description	This parameter defines one valid CallerId for the callers who have permission to call the function WdgM_SetMode.	
Multiplicity	0..255	
Type	INTEGER	
Configuration class	<b>PreCompile:</b>	VariantPreCompile
Origin	AUTOSAR_ECUC	

### 5.3.1.16. WdgMSupervisedEntity

Containers included		
Container name	Multiplicity	Description
<a href="#">WdgMCheckpoint</a>	1..65535	This container collects all Checkpoints of this Supervised Entity. Each Supervised Entity has at least one Checkpoint.
<a href="#">WdgMInternalTransition</a>	0..65535	This container defines the graph of Internal Transitions within this Supervised Entity.

Parameters included	
Parameter name	Multiplicity
<a href="#">WdgMSupervisedEntityId</a>	1..1
<a href="#">WdgMEcucPartitionRef</a>	0..1
<a href="#">WdgMOsApplicationRef</a>	0..1
<a href="#">WdgMInternalCheckpointInitialRef</a>	1..1
<a href="#">WdgMInternalCheckpointFinalRef</a>	1..65535
<a href="#">WdgMSupervisedEntityCoreId</a>	1..1
<a href="#">WdgMErrorRecoveryEnabled</a>	1..1

Parameter Name	WdgMSupervisedEntityId	
Label	WdgMSupervisedEntityId (0 -> 65535)	
Description	This parameter shall contain the unique identifier of the supervised entity.	
Multiplicity	1..1	
Type	INTEGER	
Configuration class	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	WdgMEcucPartitionRef	
Description	Denotes in which "EcucPartition" the supervised entity is executed. When the partition is stopped, the supervised entity shall be de-activated in the WdgM to avoid an ECU reset.	
Multiplicity	0..1	
Type	REFERENCE	
Configuration class	PreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	WdgMOsApplicationRef	
Description	Optional reference to an OS Application. Beware, the Watchdog Manager module will trigger a partition restart of this OS Application when the corresponding Supervised Entity reaches WDGM_LOCAL_STATUS_FAILED.	
Multiplicity	0..1	
Type	REFERENCE	
Configuration class	PreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	WdgMInternalCheckpointInitialRef	
Description	This is the reference to the initial Checkpoint for this Supervised Entity.	
Multiplicity	1..1	
Type	SYMBOLIC-NAME-REFERENCE	
Configuration class	VariantPreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	WdgMInternalCheckpointFinalRef	
----------------	--------------------------------	--

<b>Description</b>	This is the reference to the final Checkpoint(s) for this Supervised Entity.	
<b>Multiplicity</b>	1..65535	
<b>Type</b>	SYMBOLIC-NAME-REFERENCE	
<b>Configuration class</b>	<b>PreCompile:</b>	VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC	

<b>Parameter Name</b>	<b>WdgMSupervisedEntityCoreId</b>	
<b>Label</b>	WdgMSupervisedEntityCoreId	
<b>Description</b>	This parameter maps to which core the SupervisedEntity belongs to.	
<b>Multiplicity</b>	1..1	
<b>Type</b>	INTEGER	
<b>Default value</b>	0	
<b>Configuration class</b>	<b>VariantPreCompile:</b>	VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC	

<b>Parameter Name</b>	<b>WdgMErrorRecoveryEnabled</b>	
<b>Description</b>	If set to true, then the Supervised Entity never enters the local Supervision Status <code>WDGM_LOCAL_STATUS_EXPIRED</code> (i.e. independent of the configured kind of supervision, in case of a detected incorrect supervision, it remains in status <code>WDGM_LOCAL_STATUS_FAILED</code> until all supervisions succeed again). If set to false, then the Supervised Entity behaves according to AUTOSAR.	
<b>Multiplicity</b>	1..1	
<b>Type</b>	BOOLEAN	
<b>Default value</b>	false	
<b>Configuration class</b>	<b>VariantPreCompile:</b>	VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH	

### 5.3.1.17. WdgMCheckpoint

Parameters included	
Parameter name	Multiplicity
<a href="#">WdgMCheckpointId</a>	1..1

<b>Parameter Name</b>	<b>WdgMCheckpointId</b>
-----------------------	-------------------------



<b>Description</b>	This parameter shall contain the unique identifier of Checkpoint.	
<b>Multiplicity</b>	1..1	
<b>Type</b>	INTEGER	
<b>Configuration class</b>	<b>VariantPreCompile:</b>	VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC	

### 5.3.1.18. WdgMInternalTransition

Parameters included	
Parameter name	Multiplicity
<a href="#">WdgMInternalTransitionDestRef</a>	1..1
<a href="#">WdgMInternalTransitionSourceRef</a>	1..1

<b>Parameter Name</b>	<b>WdgMInternalTransitionDestRef</b>
<b>Description</b>	This is the reference to the destination Checkpoint of a Internal Transition within this Supervised Entity.
<b>Multiplicity</b>	1..1
<b>Type</b>	SYMBOLIC-NAME-REFERENCE
<b>Configuration class</b>	<b>VariantPreCompile:</b> VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC

<b>Parameter Name</b>	<b>WdgMInternalTransitionSourceRef</b>
<b>Description</b>	This is the reference to the source Checkpoint of a Internal Transition within this Supervised Entity.
<b>Multiplicity</b>	1..1
<b>Type</b>	SYMBOLIC-NAME-REFERENCE
<b>Configuration class</b>	<b>VariantPreCompile:</b> VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC

### 5.3.1.19. WdgMGeneralMulticore

Parameters included	
Parameter name	Multiplicity

Parameters included	
<a href="#">WdgMNumberOfCores</a>	0..1
<a href="#">WdgMMasterCoreId</a>	0..1
<a href="#">WdgMMasterWaitSlaveModeSwitch</a>	0..1

Parameter Name	WdgMNumberOfCores	
Label	WdgM Number Of Cores	
Description	This parameter defines the maximum number of cores on which WdgM is distributed.	
Multiplicity	0..1	
Type	INTEGER	
Default value	1	
Configuration class	PreCompile:	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	WdgMMasterCoreId	
Label	WdgM Master Core Id	
Description	This parameter maps the master instance of WdgM to a specific Os Core ID.	
Multiplicity	0..1	
Type	INTEGER	
Default value	0	
Configuration class	PreCompile:	VariantPreCompile
Origin	Elektrobit Automotive GmbH	

Parameter Name	WdgMMasterWaitSlaveModeSwitch	
Description	This parameter defines the amount of time WdgM Master Instance shall wait for WdgM Satellite Instances to finish the mode switch, until it will check to see if the mode switch was successfully from the time the master changed to the new mode( initialization is considered a particular mode switch operation to Wdg-MInitialMode ). If this parameter is set to zero, the check is disabled. In case the mode switch was not done successfully WDG_M_EB_E_SLAVE_FAILED_-CHANGEMODE runtime error will be reported. Unit: Number of WdgM Master Instance main functions.	
Multiplicity	0..1	
Type	INTEGER	
Default value	0	

<b>Configuration class</b>	<b>PreCompile:</b>	VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH	

### 5.3.1.20. WdgMServiceAPI

Parameters included	
Parameter name	Multiplicity
<a href="#">WdgMEnableASR32ServiceAPI</a>	1..1
<a href="#">WdgMEnableASR40ServiceAPI</a>	1..1
<a href="#">WdgMDefaultASRServiceAPI</a>	1..1
<a href="#">WdgMEnableASR32ActivateAliveSupervisionAPI</a>	0..1
<a href="#">WdgMEnableASR32DeactivateAliveSupervisionAPI</a>	0..1

Parameter Name	WdgMEnableASR32ServiceAPI	
Label	Enable AUTOSAR 3.2 service API	
Description	Configures whether the AUTOSAR 3.2 service API shall be provided.  ▶ TRUE = Enables AUTOSAR 3.2 service API.  ▶ FALSE = Disables AUTOSAR 3.2 service API.	
Multiplicity	1..1	
Type	BOOLEAN	
Default value	false	
Configuration class	PreCompile:	VariantPreCompile
Origin	Elektrobit Automotive GmbH	

Parameter Name	WdgMEnableASR40ServiceAPI
<b>Label</b>	Enable AUTOSAR 4.0 service API
<b>Description</b>	Configures whether the AUTOSAR 4.0 service API shall be provided.  <ul style="list-style-type: none"> <li>▶ TRUE = Enables AUTOSAR 4.0 service API.</li> <li>▶ FALSE = Disables AUTOSAR 4.0 service API.</li> </ul>
<b>Multiplicity</b>	1..1
<b>Type</b>	BOOLEAN
<b>Default value</b>	false

<b>Configuration class</b>	<b>PreCompile:</b>	VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>WdgMDefaultASRServiceAPI</b>	
<b>Label</b>	Default AUTOSAR service API	
<b>Description</b>	<p>Defines the default AUTOSAR service API.</p> <ul style="list-style-type: none"> <li>▶ AUTOSAR_32 = AUTOSAR 3.2 service API is the default one.</li> <li>▶ AUTOSAR_40 = AUTOSAR 4.0 service API is the default one.</li> <li>▶ NONE = No default AUTOSAR service API is provided.</li> </ul>	
<b>Multiplicity</b>	1..1	
<b>Type</b>	ENUMERATION	
<b>Default value</b>	AUTOSAR_40	
<b>Range</b>	AUTOSAR_32 AUTOSAR_40 NONE	
<b>Configuration class</b>	<b>PreCompile:</b>	VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>WdgMEnableASR32ActivateAliveSupervisionAPI</b>	
<b>Label</b>	Enable AUTOSAR 3.2 service API ActivateAliveSupervision	
<b>Description</b>	<p>Configures whether the AUTOSAR 3.2 service API ActivateAliveSupervision shall be provided for sake of compatibility with existing ASR32 Software Components.</p> <ul style="list-style-type: none"> <li>▶ Enabled = The WdgM re-directs the call to WdgM_ActivateAliveSupervision to an externally implemented API with the API name specified within this parameter. (Note: makes only sense if either parameter Default Service API is set to AUTOSAR_32 or generation of ASR32 Service API is enabled.)</li> <li>▶ Disabled = The WdgM Service Component does not support / provide this service API.</li> </ul>	
<b>Multiplicity</b>	0..1	
<b>Type</b>	STRING	
<b>Default value</b>		
<b>Configuration class</b>	<b>PreCompile:</b>	VariantPreCompile

<b>Origin</b>	Elektrobit Automotive GmbH	
<b>Parameter Name</b>	<b>WdgMEnableASR32DeactivateAliveSupervisionAPI</b>	
<b>Label</b>	Enable AUTOSAR 3.2 service API DeactivateAliveSupervision	
<b>Description</b>	<p>Configures whether the AUTOSAR 3.2 service API DeactivateAliveSupervision shall be provided for sake of compatibility with existing ASR32 Software Components.</p> <ul style="list-style-type: none"> <li>▶ <b>Enabled</b> = The WdgM re-directs the call to WdgM_DeactivateAliveSupervision to an externally implemented API with the API name specified within this parameter. (Note: makes only sense if either parameter Default Service API is set to AUTOSAR_32 or generation of ASR32 Service API is enabled.)</li> <li>▶ <b>Disabled</b> = The WdgM Service Component does not support / provide this service API.</li> </ul>	
<b>Multiplicity</b>	0..1	
<b>Type</b>	STRING	
<b>Default value</b>		
<b>Configuration class</b>	<b>PreCompile:</b>	VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH	

### 5.3.1.21. WdgMSupervisorCallouts

Parameters included	
Parameter name	Multiplicity
<a href="#">WdgMGetExpectedInitStateCallout</a>	0..1
<a href="#">WdgMInitRedirectionCallout</a>	1..1
<a href="#">WdgMDeInitRedirectionCallout</a>	1..1
<a href="#">WdgMGetExpectedWdgMModeCallout</a>	0..1
<a href="#">WdgMSetModeRedirectionCallout</a>	1..1
<a href="#">WdgMGetElapsedTimeCallout</a>	0..1
<a href="#">WdgMTimeGranularity</a>	1..1
<a href="#">WdgMMainFunctionPeriodTolerance</a>	1..1
<a href="#">WdgMIsPerformResetCallout</a>	0..1
<a href="#">WdgMSupervisionExpiredCallout</a>	0..1

Parameters included	
<a href="#">WdgMIndividualModeSwitchCallout</a>	0..1
<a href="#">WdgMGlobalModeSwitchCallout</a>	0..1
<a href="#">WdgMDetCallout</a>	0..1
<a href="#">WdgMGetCoreIdCallout</a>	0..1

Parameter Name	WdgMGetExpectedInitStateCallout	
Description	<p>Defines the implemented API name for polling a desired initialization state. This API is usually implemented in the Supervisor and shall have the following syntax:</p> <p>Std_ReturnType [ConfiguredAPIName](WdgM_EB_InitStatusType* InitStatus)</p>	
Multiplicity	0..1	
Type	STRING	
Default value		
Configuration class	PreCompile:	VariantPreCompile
Origin	Elektrobit Automotive GmbH	

Parameter Name	WdgMInitRedirectionCallout	
Description	<p>This parameter is only used if parameter WdgMGetExpectedInitStateCallout is enabled. The WdgM redirects each call to BSW API WdgM_Init to the configured callout API. If this parameter is left empty, then the BSW API WdgM_Init is not provided and no redirection is performed.</p>	
Multiplicity	1..1	
Type	STRING	
Default value		
Configuration class	PreCompile:	VariantPreCompile
Origin	Elektrobit Automotive GmbH	

Parameter Name	WdgMDeInitRedirectionCallout	
Description	<p>This parameter is only used if parameter WdgMGetExpectedInitStateCallout is enabled. The WdgM redirects each call to BSW API WdgM_DeInit to the configured callout API. If this parameter is left empty, then the BSW API WdgM_DeInit is not provided and no redirection is performed.</p>	
Multiplicity	1..1	
Type	STRING	

<b>Default value</b>	
<b>Configuration class</b>	<b>PreCompile:</b> VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH

<b>Parameter Name</b>	<b>WdgMGetExpectedWdgMModeCallout</b>
<b>Description</b>	<p>Description: Defines the implemented API name for polling the expected WdgM-Mode. This API is usually implemented in the Supervisor and shall have the following syntax:</p> <p>Std_ReturnType [ConfiguredAPIName](WdgM_ModeType* Mode)</p>
<b>Multiplicity</b>	0..1
<b>Type</b>	STRING
<b>Default value</b>	
<b>Configuration class</b>	<b>PreCompile:</b> VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH

<b>Parameter Name</b>	<b>WdgMSetModeRedirectionCallout</b>
<b>Description</b>	<p>This parameter is only used if parameter WdgMGetExpectedWdgMModeCallout is enabled. The WdgM redirects each call to BSW API WdgM_SetMode to the configured callout API. If this parameter is left empty, then the BSW API WdgM_SetMode is not provided and no redirection is performed.</p>
<b>Multiplicity</b>	1..1
<b>Type</b>	STRING
<b>Default value</b>	
<b>Configuration class</b>	<b>PreCompile:</b> VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH

<b>Parameter Name</b>	<b>WdgMGetElapsedTimeCallout</b>
<b>Description</b>	<p>Defines the implemented API name for getting the information regarding the elapsed time. The time units of the parameters are expected to be in real-time with the granularity provided in parameter WdgMTimeGranularity. This API is usually implemented in a safe OS and shall have the following syntax:</p> <p>void [ConfiguredAPIName](uint32* PreviousTime, uint32* ElapsedTime)</p>
<b>Multiplicity</b>	0..1
<b>Type</b>	STRING
<b>Default value</b>	

<b>Configuration class</b>	<b>PreCompile:</b>	VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>WdgMTimeGranularity</b>	
<b>Description</b>	This parameter defines the granularity in real-time between two consecutive units of the time parameters used in the GetElapsedTime API specified via parameter WdgMGetElapsedTimeCallout. Unit: [s]	
<b>Multiplicity</b>	1..1	
<b>Type</b>	FLOAT	
<b>Default value</b>	0.0001	
<b>Configuration class</b>	<b>PreCompile:</b>	VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>WdgMMainFunctionPeriodTolerance</b>	
<b>Description</b>	This parameter defines the allowed tolerance (plus / minus) in real-time between two main function calls with respect to the configured schedule time of parameter WdgMSupervisionCycle. Unit: [s]	
<b>Multiplicity</b>	1..1	
<b>Type</b>	FLOAT	
<b>Default value</b>	0.0001	
<b>Configuration class</b>	<b>PreCompile:</b>	VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>WdgMIsPerformResetCallout</b>	
<b>Description</b>	Defines the implemented API name for getting the authorization of a requested Watchdog reset. This API is usually implemented in the Supervisor and shall have the following syntax:  Std_ReturnType [ConfiguredAPIName](void)	
<b>Multiplicity</b>	0..1	
<b>Type</b>	STRING	
<b>Default value</b>		
<b>Configuration class</b>	<b>PreCompile:</b>	VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>WdgMSupervisionExpiredCallout</b>	
-----------------------	--------------------------------------	--



<b>Description</b>	<p>Defines the implemented API name for indicating the Supervisor about a Supervision failure and getting the information for a possible re-initialization. This API is usually implemented in the Supervisor and shall have the following syntax:</p> <pre>void [ConfiguredAPIName](WdgM_SupervisedEntityType FirstExpiredSEID)</pre>	
<b>Multiplicity</b>	0..1	
<b>Type</b>	STRING	
<b>Default value</b>		
<b>Configuration class</b>	<b>PreCompile:</b>	VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>WdgMIndividualModeSwitchCallout</b>	
<b>Description</b>	<p>Defines the implemented API name for indicating the Supervisor about a mode switch of a Supervised Entity (e.g. from WDGM_LOCAL_STATUS_OK to WDGM_LOCAL_STATUS_FAILED, or from WDGM_LOCAL_STATUS_FAILED to WDGM_LOCAL_STATUS_DEACTIVATED, etc.). This API is usually implemented in the Supervisor and shall have the following syntax:</p> <pre>void [ConfiguredAPIName](WdgM_SupervisedEntityType SEID, WdgM_ModeType OldMode, WdgM_ModeType NewMode)</pre>	
<b>Multiplicity</b>	0..1	
<b>Type</b>	STRING	
<b>Default value</b>		
<b>Configuration class</b>	<b>PreCompile:</b>	VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>WdgMGlobalModeSwitchCallout</b>	
<b>Description</b>	<p>Defines the implemented API name for indicating the Supervisor about a global mode switch of the WdgM (e.g. from WDGM_GLOBAL_STATUS_OK to WDGM_GLOBAL_STATUS_FAILED, or from WDGM_GLOBAL_STATUS_FAILED to WDGM_GLOBAL_STATUS_EXPIRED, etc.). This API is usually implemented in the Supervisor and shall have the following syntax:</p> <pre>void [ConfiguredAPIName](WdgM_ModeType OldMode, WdgM_ModeType NewMode)</pre>	
<b>Multiplicity</b>	0..1	
<b>Type</b>	STRING	
<b>Default value</b>		

<b>Configuration class</b>	<b>PreCompile:</b>	VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>WdgMDetCallout</b>	
<b>Description</b>	Defines the implemented API name for indicating the Supervisor about an internal DET error. This API shall have the following syntax:  void [ConfiguredAPIName](uint8 ApiID, uint8 ErrorID)	
<b>Multiplicity</b>	0..1	
<b>Type</b>	STRING	
<b>Default value</b>		
<b>Configuration class</b>	<b>PreCompile:</b>	VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH	

<b>Parameter Name</b>	<b>WdgMGetCoreIdCallout</b>	
<b>Description</b>	Defines the implemented API name for retrieving the core id information required for Temporal Program Flow Monitoring and Logical Program Flow Monitoring. This API shall have the following syntax:  uint16 [ConfiguredAPIName](void)	
<b>Multiplicity</b>	0..1	
<b>Type</b>	STRING	
<b>Default value</b>		
<b>Configuration class</b>	<b>PreCompile:</b>	VariantPreCompile
<b>Origin</b>	Elektrobit Automotive GmbH	

### 5.3.1.22. WdgMWatchdog

Parameters included	
Parameter name	Multiplicity
<a href="#">WdgMWatchdogName</a>	1..1
<a href="#">WdgMWatchdogDeviceRef</a>	1..1

<b>Parameter Name</b>	<b>WdgMWatchdogName</b>
<b>Description</b>	This parameter shall contain the symbolic name of the watchdog instance.

<b>Multiplicity</b>	1..1
<b>Type</b>	STRING
<b>Configuration class</b>	<b>VariantPreCompile:</b> VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC

<b>Parameter Name</b>	<b>WdgMWatchdogDeviceRef</b>
<b>Description</b>	Reference to one device container of Watchdog Interface. In the referenced container WdgIfDevice, the parameter WdgIfDeviceIndex contains the Index parameter that WdgM has to use for WdgIf_SetTriggerCondition calls for that watchdog instance.
<b>Multiplicity</b>	1..1
<b>Type</b>	SYMBOLIC-NAME-REFERENCE
<b>Configuration class</b>	<b>VariantPreCompile:</b> VariantPreCompile
<b>Origin</b>	AUTOSAR_ECUC

## 5.3.2. Application programming interface (API)

### 5.3.2.1. Type definitions

#### 5.3.2.1.1. WdgM\_ASR32\_SupervisedEntityType

<b>Purpose</b>	Supervised entity id.
<b>Type</b>	uint16
<b>Description</b>	This type identifies an individual Supervised Entity for the Watchdog Manager.

#### 5.3.2.1.2. WdgM\_ASR40\_CheckpointIdType

<b>Purpose</b>	Checkpoint entity id.
<b>Type</b>	uint16
<b>Description</b>	This type identifies an individual checkpoint entity of a supervised entity for the Watchdog Manager.

#### 5.3.2.1.3. WdgM\_ASR40\_SupervisedEntityType

<b>Purpose</b>	Supervised entity id.
<b>Type</b>	uint16
<b>Description</b>	This type identifies an individual Supervised Entity for the Watchdog Manager.

#### 5.3.2.1.4. WdgM\_CheckpointIdType

<b>Purpose</b>	Checkpoint Id Type.
<b>Type</b>	<a href="#">WdgM_ASR40_CheckpointIdType</a>
<b>Description</b>	Provide AUTOSAR 4.0 CheckpointIdType as default to other BSW modules

#### 5.3.2.1.5. WdgM\_ConfigType

<b>Purpose</b>	configuration of WdgM
<b>Type</b>	uint8
<b>Description</b>	This is a dummy structure to hold configuration parameters. As the WdgM is not able to be link or post build time configurable this structure is not used. It is defined for compatibility only. A pointer to this structure is passed to the Watchdog Manager initialization function for configuration.

#### 5.3.2.1.6. WdgM\_EB\_GlobalInitStatusType

<b>Purpose</b>	Global Init status.
<b>Type</b>	uint32
<b>Description</b>	This type is used to represent the global initialization status of the Watchdog Manager.

#### 5.3.2.1.7. WdgM\_EB\_InitStatusType

<b>Purpose</b>	Init status.
<b>Type</b>	uint8
<b>Description</b>	This type is used to define the expected / actual initialization status The two possible states are: WDGM_EB_INIT_STATUS_INIT ..... WdgM is / shall be initialized WDGM_EB_INIT_STATUS_DEINIT ... WdgM is / shall be de-initialized

#### 5.3.2.1.8. WdgM\_GlobalStatusType

<b>Purpose</b>	Global status.
<b>Type</b>	uint8
<b>Description</b>	This type is used to represent the global supervision status of the Watchdog Manager.

#### 5.3.2.1.9. WdgM\_LocalStatusType

<b>Purpose</b>	Local status.
<b>Type</b>	uint8
<b>Description</b>	This type is used to represent the local supervision status of the individual supervised entities.

#### 5.3.2.1.10. WdgM\_ModeType

<b>Purpose</b>	Mode id.
<b>Type</b>	uint8
<b>Description</b>	This type distinguishes the different modes that were configured for the Watchdog Manager.

#### 5.3.2.1.11. WdgM\_SupervisedEntityType

<b>Purpose</b>	Supervised Entity Id Type.
<b>Type</b>	<a href="#">WdgM_ASR40_SupervisedEntityType</a>
<b>Description</b>	Provide AUTOSAR 4.0 SupervisedEntityType as default to other BSW modules

### 5.3.2.2. Macro constants

#### 5.3.2.2.1. WDG\_M\_AR\_RELEASE\_MAJOR\_VERSION

<b>Purpose</b>	AUTOSAR release major version.
<b>Value</b>	4U

#### 5.3.2.2.2. WDGM\_AR\_RELEASE\_MINOR\_VERSION

<b>Purpose</b>	AUTOSAR release minor version.
<b>Value</b>	0U

#### 5.3.2.2.3. WDGM\_AR\_RELEASE\_REVISION\_VERSION

<b>Purpose</b>	AUTOSAR release revision version.
<b>Value</b>	3U

#### 5.3.2.2.4. WDGM\_EB\_E\_DEINIT\_REQUEST

<b>Purpose</b>	Development error: WdgM de-initialization request failed.
<b>Value</b>	0x81U

#### 5.3.2.2.5. WDGM\_EB\_E\_INIT\_REQUEST

<b>Purpose</b>	Development error: WdgM initialization request failed.
<b>Value</b>	0x80U

#### 5.3.2.2.6. WDGM\_EB\_E\_REENTRANCY

<b>Purpose</b>	Development error: non-reentrant WdgM main function is executed in parallel.
<b>Value</b>	0x83U

#### 5.3.2.2.7. WDGM\_EB\_E\_SETMODE\_REQUEST

<b>Purpose</b>	Development error: WdgM mode change request failed.
<b>Value</b>	0x82U

#### 5.3.2.2.8. WDGM\_EB\_E\_SLAVE\_FAILED\_CHANGEMODE

<b>Purpose</b>	Development error: Satellite instance failed to change to the new mode in the time given by parameter WdgMMasterWaitSlaveModeSwitch. Initialization is considered a particular mode switch operation to WdgMInitialMode.
----------------	--

<b>Value</b>	0x85U
--------------	-------

#### 5.3.2.2.9. WdGM\_EB\_INIT\_STATUS\_DEINIT

<b>Purpose</b>	WdgM is / shall be de-initialized.
<b>Value</b>	0xFFU

#### 5.3.2.2.10. WdGM\_EB\_INIT\_STATUS\_INIT

<b>Purpose</b>	WdgM is / shall be initialized.
<b>Value</b>	0U

#### 5.3.2.2.11. WdGM\_EB\_PARAM\_UNUSED

<b>Purpose</b>	This macro can be used to avoid compiler warnings.
<b>Value</b>	((void) (x))

#### 5.3.2.2.12. WdGM\_E\_AMBIGIOUS

<b>Purpose</b>	Development error: Function WdgM_UpdateAliveIndication cannot determine the Checkpoint, because there are more than one alive supervisions configured in the current mode for the given Supervised Entity.
<b>Value</b>	0x18U

#### 5.3.2.2.13. WdGM\_E\_CPID

<b>Purpose</b>	Development error: API service used with an invalid CheckpointId.
<b>Value</b>	0x16U

#### 5.3.2.2.14. WdGM\_E\_DEPRECATED

<b>Purpose</b>	Development error: Deprecated API service was used.
<b>Value</b>	0x17U

#### 5.3.2.2.15. WDGGM\_E\_DISABLE\_NOT\_ALLOWED

<b>Purpose</b>	Development error: Disabling of watchdog not allowed.
<b>Value</b>	0x15U

#### 5.3.2.2.16. WDGGM\_E\_INV\_POINTER

<b>Purpose</b>	Development error: API service called with NULL pointer.
<b>Value</b>	0x14U

#### 5.3.2.2.17. WDGGM\_E\_NOT\_AUTHORIZED

<b>Purpose</b>	Development error: API service is not authorized to be executed (e.g. in case of a call to WdgM_PerformReset).
<b>Value</b>	0xF1U

#### 5.3.2.2.18. WDGGM\_E\_NO\_INIT

<b>Purpose</b>	Development error: WdgM not initialized.
<b>Value</b>	0x10U

#### 5.3.2.2.19. WDGGM\_E\_PARAM\_CONFIG

<b>Purpose</b>	Development error: API service Wdg_Init was called with an erroneous configuration set.
<b>Value</b>	0x11U

#### 5.3.2.2.20. WDGGM\_E\_PARAM\_MODE

<b>Purpose</b>	Development error: API service called with invalid mode.
<b>Value</b>	0x12U

#### 5.3.2.2.21. WDGGM\_E\_PARAM\_SEID

<b>Purpose</b>	Development error: API service called with invalid supervised entity.
----------------	---



<b>Value</b>	0x13U
--------------	-------

#### 5.3.2.2.22. WDGM\_E\_PARAM\_WRONG\_CORE\_ID

<b>Purpose</b>	Development error: API service called with wrong core id.
<b>Value</b>	0x84U

#### 5.3.2.2.23. WDGM\_E\_SEDEACTIVATED

<b>Purpose</b>	Development error: API service used with a checkpoint of a Supervised Entity that is deactivated in the current Watchdog Manager mode.
<b>Value</b>	0x19U

#### 5.3.2.2.24. WDGM\_GLOBAL\_STATUS\_DEACTIVATED

<b>Purpose</b>	Supervision deactivated.
<b>Value</b>	4U

#### 5.3.2.2.25. WDGM\_GLOBAL\_STATUS\_EXPIRED

<b>Purpose</b>	At least one Supervised Entity has Local Supervision Status WDGM_LOCAL_STATUS_EXPIRED and the time limit to postpone the blocking of watchdog triggering is not yet exceeded.
<b>Value</b>	2U

#### 5.3.2.2.26. WDGM\_GLOBAL\_STATUS\_FAILED

<b>Purpose</b>	At least one Supervised Entity has Local Supervision Status WDGM_LOCAL_STATUS_FAILED.
<b>Value</b>	1U

#### 5.3.2.2.27. WDGM\_GLOBAL\_STATUS\_OK

<b>Purpose</b>	Alive / Deadline / Logical Supervision of all active Supervised Entities fulfilled.
----------------	---

<b>Value</b>	0U
--------------	----

#### 5.3.2.2.28. WDGM\_GLOBAL\_STATUS\_STOPPED

<b>Purpose</b>	At least one Supervised Entity has Local Supervision Status WDGM_LOCAL_STATUS_EXPIRED and the time limit to postpone the blocking of watchdog triggering is exceeded.
<b>Value</b>	3U

#### 5.3.2.2.29. WDGM\_LOCAL\_STATUS\_DEACTIVATED

<b>Purpose</b>	Supervision deactivated.
<b>Value</b>	4U
<b>Description</b>	Alive, Deadline, and Logical Supervision is disabled for this Supervised Entity.

#### 5.3.2.2.30. WDGM\_LOCAL\_STATUS\_EXPIRED

<b>Purpose</b>	One of Alive, Deadline, or Logical Supervision is not fulfilled.
<b>Value</b>	2U
<b>Description</b>	Timing constraints for Alive Supervision have been violated including the margins for more often than the acceptable amount of failed supervision reference cycles.

#### 5.3.2.2.31. WDGM\_LOCAL\_STATUS\_FAILED

<b>Purpose</b>	Alive Supervision not fulfilled, but Deadline and Logical Supervision are fulfilled.
<b>Value</b>	1U
<b>Description</b>	Timing constraints for Alive Supervision have been violated including the margins, but the amount of failed supervision reference cycles has not been exceeded.

#### 5.3.2.2.32. WDGM\_LOCAL\_STATUS\_OK

<b>Purpose</b>	Alive / Deadline / Logical Supervision fulfilled.
<b>Value</b>	0U
<b>Description</b>	Timing constraints for Alive Supervision are fulfilled within the configured margins.

#### 5.3.2.2.33. WDGM\_MODULE\_ID

<b>Purpose</b>	AUTOSAR module identification.
<b>Value</b>	13U

#### 5.3.2.2.34. WDGM\_SID\_CHECKPOINT\_REACHED

<b>Purpose</b>	Service id of <a href="#">WdgM_CheckpointReached()</a> .
<b>Value</b>	0x0eU

#### 5.3.2.2.35. WDGM\_SID\_DEINIT

<b>Purpose</b>	Service id of <a href="#">WdgM_DeInit()</a> .
<b>Value</b>	0x01U

#### 5.3.2.2.36. WDGM\_SID\_GET\_FIRST\_EXPIRED\_SEID

<b>Purpose</b>	Service id of <a href="#">WdgM_GetFirstExpiredSEID()</a> .
<b>Value</b>	0x10U

#### 5.3.2.2.37. WDGM\_SID\_GET\_GLOBAL\_STATUS

<b>Purpose</b>	Service id of <a href="#">WdgM_GetGlobalStatus()</a> .
<b>Value</b>	0x0dU

#### 5.3.2.2.38. WDGM\_SID\_GET\_LOCAL\_STATUS

<b>Purpose</b>	Service id of <a href="#">WdgM_GetLocalStatus()</a> .
<b>Value</b>	0x0cU

#### 5.3.2.2.39. WDGM\_SID\_GET\_MODE

<b>Purpose</b>	Service id of <a href="#">WdgM_GetMode()</a> .
<b>Value</b>	0x0bU

#### 5.3.2.2.40. WDMG\_SID\_GET\_VERSION\_INFO

<b>Purpose</b>	Service id of <a href="#">WdgM_GetVersionInfo()</a> .
<b>Value</b>	0x02U

#### 5.3.2.2.41. WDMG\_SID\_INIT

<b>Purpose</b>	Service id of <a href="#">WdgM_Init()</a> .
<b>Value</b>	0x00U

#### 5.3.2.2.42. WDMG\_SID\_MAIN\_FUNCTION

<b>Purpose</b>	Service id of <a href="#">WdgM_MainFunction()</a> .
<b>Value</b>	0x08U

#### 5.3.2.2.43. WDMG\_SID\_PERFORM\_RESET

<b>Purpose</b>	Service id of <a href="#">WdgM_PerformReset()</a> .
<b>Value</b>	0x0fU

#### 5.3.2.2.44. WDMG\_SID\_SET\_MODE

<b>Purpose</b>	Service id of <a href="#">WdgM_SetMode()</a> .
<b>Value</b>	0x03U

#### 5.3.2.2.45. WDMG\_SID\_UPDATE\_ALIVE\_COUNTER

<b>Purpose</b>	Service id of <a href="#">WdgM_UpdateAliveCounter()</a> .
<b>Value</b>	0x04U

#### 5.3.2.2.46. WDMG\_SW\_MAJOR\_VERSION

<b>Purpose</b>	AUTOSAR module major version.
<b>Value</b>	6U

#### 5.3.2.2.47. WDGM\_SW\_MINOR\_VERSION

<b>Purpose</b>	AUTOSAR module minor version.
<b>Value</b>	1U

#### 5.3.2.2.48. WDGM\_SW\_PATCH\_VERSION

<b>Purpose</b>	AUTOSAR module patch version.
<b>Value</b>	36U

#### 5.3.2.2.49. WDGM\_VENDOR\_ID

<b>Purpose</b>	AUTOSAR vendor identification: Elektrobit Automotive GmbH.
<b>Value</b>	1U

#### 5.3.2.2.50. WdgM\_CheckpointReached

<b>Purpose</b>	Wrapping macro for WdgM_CheckpointReached to provide AUTOSAR 4.0 API as default to other BSW modules.
<b>Value</b>	WdgM_ASR40_CheckpointReached

#### 5.3.2.2.51. WdgM\_UpdateAliveCounter

<b>Purpose</b>	Wrapping macro for WdgM_UpdateAliveCounter to provide AUTOSAR 4.0 API as default to other BSW modules.
<b>Value</b>	WdgM_ASR40_UpdateAliveCounter

### 5.3.2.3. Objects

#### 5.3.2.3.1. WDGM\_CONFIG\_NAME

<b>Purpose</b>	Data structure storing configuration data.
<b>Type</b>	const <a href="#">WdgM_ConfigType</a>

<b>Description</b>	A pointer to this structure must be given to <a href="#">WdgM_Init()</a> . The name of this structure is defined by the container name of the first entry of the WdgM configuration set list.
--------------------	---

### 5.3.2.4. Functions

#### 5.3.2.4.1. Supervisor\_WdgM\_ASR32\_SetModeRedirectionCallout

<b>Purpose</b>	Redirected callout API for WdgM_SetMode.	
<b>Synopsis</b>	<pre>Std_ReturnType Supervisor_WdgM_ASR32_SetModeRedirectionCallout ( WdgM_ModeType Mode );</pre>	
<b>Service ID</b>	0x03	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Non reentrant	
<b>Parameters (in)</b>	Mode	One of the configured Watchdog Manager modes
<b>Return Value</b>	Success of operation	
	E_OK	Successfully changed to the new mode
	E_NOT_OK	Changing to the new mode failed

#### 5.3.2.4.2. Supervisor\_WdgM\_ASR40\_SetModeRedirectionCallout

<b>Purpose</b>	Redirected callout API for WdgM_SetMode.	
<b>Synopsis</b>	<pre>Std_ReturnType Supervisor_WdgM_ASR40_SetModeRedirectionCallout ( WdgM_ModeType Mode , uint16 CallerID );</pre>	
<b>Service ID</b>	0x03	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Non reentrant	
<b>Parameters (in)</b>	Mode	One of the configured Watchdog Manager modes
	CallerID	Module ID of the calling module
<b>Return Value</b>	Success of operation	
	E_OK	Successfully changed to the new mode
	E_NOT_OK	Changing to the new mode failed

#### 5.3.2.4.3. Supervisor\_WdgM\_ActivateAliveSupervisionRedirectionCallout

<b>Purpose</b>	Activate alive supervision of a considered entity via a pre-configured callout API.	
<b>Synopsis</b>	<pre>Std_ReturnType Supervisor_WdgM_ActivateAliveSupervisionRedirectionCallout ( WdgM_ASR40_SupervisedEntityType SEId );</pre>	
<b>Parameters (in)</b>	SEId	Id of supervised entity whose alive supervision should be activated
<b>Return Value</b>	Success of operation	
	E_OK	Operation successful
	E_NOT_OK	Operation failed
<b>Description</b>	<p>This function exists for ASR31 compatibility and simply redirects the call to a pre-configured API.</p> <p>Note: In ASR31, this API activated the alive supervision of a supervised entity of the active alive supervision configuration set.</p>	

#### 5.3.2.4.4. Supervisor\_WdgM\_DeInitRedirectionCallout

<b>Purpose</b>	Redirected callout API for WdgM_DeInit.	
<b>Synopsis</b>	<pre>void Supervisor_WdgM_DeInitRedirectionCallout ( void );</pre>	
<b>Service ID</b>	0x01	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Non reentrant	

#### 5.3.2.4.5. Supervisor\_WdgM\_DeactivateAliveSupervisionRedirectionCallout

<b>Purpose</b>	Deactivate alive supervision of a considered entity via a pre-configured callout API.	
<b>Synopsis</b>	<pre>Std_ReturnType Supervisor_WdgM_DeactivateAliveSupervisionRedirectionCallout ( WdgM_ASR40_SupervisedEntityType SEId );</pre>	
<b>Parameters (in)</b>	SEId	Id of supervised entity whose alive supervision should be deactivated
<b>Return Value</b>	Success of operation	
	E_OK	Operation successful
	E_NOT_OK	Operation failed
<b>Description</b>	<p>This function exists for ASR31 compatibility and simply redirects the call to a pre-configured API.</p>	

Note: In ASR31, this API deactivated the alive supervision of a supervised entity of the active alive supervision configuration set.

#### 5.3.2.4.6. Supervisor\_WdgM\_DetCallout

<b>Purpose</b>	Indicate an internal error.	
<b>Synopsis</b>	<code>void Supervisor_WdgM_DetCallout ( uint8 SID , uint8 ErrorID );</code>	
<b>Parameters (in)</b>	SID	ID of the API where an internal error was detected
	ErrorID	Internal Error Type

#### 5.3.2.4.7. Supervisor\_WdgM\_GetCoreIdCallout

<b>Purpose</b>	The function returns a unique core identifier.	
<b>Synopsis</b>	<code>WdgM_EB_CoreIdType Supervisor_WdgM_GetCoreIdCallout ( void );</code>	
<b>Parameters (in)</b>	None	
<b>Return Value</b>	The return value is the unique ID of the core.	

#### 5.3.2.4.8. Supervisor\_WdgM\_GetExpectedInitStateCallout

<b>Purpose</b>	Provide actual / get expected initialization state.	
<b>Synopsis</b>	<code>Std_ReturnType Supervisor_WdgM_GetExpectedInitState- Callout ( WdgM_EB_InitStatusType * InitStatus );</code>	
<b>Parameters (in)</b>	inout]	InitStatus The caller of this API (WdgM) provides the current initial status, the expected initial status for the WdgM shall be returned in case E_OK is returned.
<b>Parameters (in,out)</b>	inout]	InitStatus The caller of this API (WdgM) provides the current initial status, the expected initial status for the WdgM shall be returned in case E_OK is returned.
<b>Return Value</b>	Success of operation	
	E_OK	The returned value in InitStatus is valid. The WdgM changes to the expected state.



	E_NOT_OK	The returned value is not valid and will be ignored. WdgM continues normal operation.
<b>Description</b>	The callout is invoked at the beginning of WdgM_MainFunction cycle. Possible values for InitStatus: WDGM_EB_INIT_STATUS_INIT The WdgM shall be initialized, respectively stay initialized. WDGM_EB_INIT_STATUS_DEINIT The WdgM shall be de-initialized, respectively stay de-initialized.	

#### 5.3.2.4.9. Supervisor\_WdgM\_GetExpectedWdgMModeCallout

<b>Purpose</b>	Provide actual / get expected WdgM Mode.	
<b>Synopsis</b>	<pre>Std_ReturnType Supervisor_WdgM_GetExpected- WdgMModeCallout ( WdgM_ModeType * WdgMMode );</pre>	
<b>Parameters (in)</b>	inout]	WdgMMode The caller of this API (WdgM) provides the current mode, the expected WdgM mode shall be returned in case E_OK is returned.
<b>Parameters (in,out)</b>	inout]	WdgMMode The caller of this API (WdgM) provides the current mode, the expected WdgM mode shall be returned in case E_OK is returned.
<b>Return Value</b>	Success of operation	
	E_OK	The WdgM shall perform a mode switch to the mode stored in the argument WdgM-Mode.
	E_NOT_OK	The returned value is not valid and will be ignored. WdgM continues normal operation.
<b>Description</b>	The callout is invoked at the beginning of WdgM_MainFunction cycle if WdgM is initialized and the prior call to WdgMGetExpectedInitStateCallout also returned WDGM_EB_INIT_STATUS_INIT.	

#### 5.3.2.4.10. Supervisor\_WdgM\_GetTimeCallout

<b>Purpose</b>	Get elapsed time.
----------------	-------------------

<b>Synopsis</b>	<pre>void <b>Supervisor_WdgM_GetTimeCallout</b> ( uint32                                      * PreviousTime , uint32 * ElapsedTime );</pre>	
<b>Parameters (in)</b>	inout]	PreviousTime The old time is passed in order to calculate the difference with respect to the actual time. The actual time is returned via this variable.
<b>Parameters (in,out)</b>	inout]	PreviousTime The old time is passed in order to calculate the difference with respect to the actual time. The actual time is returned via this variable.
<b>Parameters (out)</b>	ElapsedTime	The elapsed time with respect to the time passed via parameter PreviousTime.

#### 5.3.2.4.11. Supervisor\_WdgM\_GlobalModeSwitchCallout

<b>Purpose</b>	Indicate a state transition of the global Supervision State.	
<b>Synopsis</b>	<pre>void <b>Supervisor_WdgM_GlobalModeSwitchCallout</b> ( WdgM_GlobalStatusType OldMode , WdgM_GlobalStatusType NewMode );</pre>	
<b>Parameters (in)</b>	OldMode	Old global WdgMMode
	NewMode	New global WdgMMode

#### 5.3.2.4.12. Supervisor\_WdgM\_IndividualModeSwitchCallout

<b>Purpose</b>	Indicate a state transition of a Supervised Entity.	
<b>Synopsis</b>	<pre>void <b>Supervisor_WdgM_IndividualModeSwitchCallout</b> ( WdgM_AS40_SupervisedEntityIdType SEID , WdgM_LocalStatusType OldMode , WdgM_LocalStatusType NewMode );</pre>	
<b>Parameters (in)</b>	SEID	Supervised Entity ID that performed a mode switch
	OldMode	Old WdgMMode of the SEID
	NewMode	New WdgMMode of the SEID

#### 5.3.2.4.13. Supervisor\_WdgM\_InitRedirectionCallout

<b>Purpose</b>	Redirected callout API for WdgM_Init.
----------------	---------------------------------------

<b>Synopsis</b>	<pre>void <b>Supervisor_WdgM_InitRedirectionCall-</b> <b>out</b> ( const WdgM_ConfigType * ConfigPtr );</pre>	
<b>Service ID</b>	0x00	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Non reentrant	
<b>Parameters (in)</b>	ConfigPtr	Pointer to configuration data, this parameter is ignored in the current implementation.

#### 5.3.2.4.14. Supervisor\_WdgM\_IsPerformResetCallout

<b>Purpose</b>	Get authorization for direct reset.	
<b>Synopsis</b>	<pre>Std_ReturnType <b>Supervisor_WdgM_IsPerformResetCallout</b> ( void );</pre>	
<b>Return Value</b>	Returns if the caller of the WdgM Perform Reset API is authorized.	
	E_OK	Caller of Perform Reset is authorized and therefore reset request will be executed by setting all triggering conditions to 0.
	E_NOT_OK	Caller of Perform Reset is not authorized and therefore reset request will be ignored. WdgM continues normal operation.

#### 5.3.2.4.15. Supervisor\_WdgM\_SupervisionExpiredCallout

<b>Purpose</b>	Indicate an expired Supervised Entity.	
<b>Synopsis</b>	<pre>void <b>Supervisor_WdgM_SupervisionExpiredCallout</b> ( WdgM_AS40_SupervisedEntityIdType ExpiredSEID );</pre>	
<b>Parameters (in)</b>	ExpiredSEID	Supervised Entity ID that expired (one of Deadline Supervision, Logical Supervision, or Alive Supervision failed).

#### 5.3.2.4.16. WdgM\_AS32\_UpdateAliveCounter

<b>Purpose</b>	Give alive indications to the Watchdog Manager via AUTOSAR 3.2 API.	
<b>Synopsis</b>	<pre>Std_ReturnType <b>WdgM_AS32_UpdateAliveCounter</b> ( WdgM_AS32_SupervisedEntityIdType SEID );</pre>	

<b>Service ID</b>	0x04	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Reentrant	
<b>Production Errors</b>	► <a href="#">WDGM_E_DATA_CORRUPTION</a> : thrown, if data corruption is detected in the internal WdgM data	
<b>Parameters (in)</b>	SEID	ID of supervised entity whose alive counter is updated
<b>Return Value</b>	Success of operation	
	E_OK	Operation successful
	E_NOT_OK	Operation failed
<b>Description</b>	This function updates the alive counter of a requested supervised entity of the current mode.	

#### 5.3.2.4.17. WdgM\_ASR40\_CheckpointReached

<b>Purpose</b>	Give alive indications to the Watchdog Manager.	
<b>Synopsis</b>	<pre>Std_ReturnType WdgM_ASR40_CheckpointReached ( WdgM_ASR40_SupervisedEntityType SEID ,   WdgM_ASR40_CheckpointIdType CheckpointID );</pre>	
<b>Service ID</b>	0x0e	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Reentrant	
<b>Production Errors</b>	► <a href="#">WDGM_E_DATA_CORRUPTION</a> : thrown, if data corruption is detected in the internal WdgM data	
<b>Parameters (in)</b>	SEID	ID of supervised entity whose alive counter is updated
	CheckpointID	Identifier of the Checkpoint within a Supervised Entity that has been reached.
<b>Return Value</b>	Success of operation	
	E_OK	Operation successful
	E_NOT_OK	Operation failed
<b>Description</b>	This function indicates to the Watchdog Manager that a checkpoint within a supervised entity has been reached.	

#### 5.3.2.4.18. WdgM\_ASR40\_UpdateAliveCounter

<b>Purpose</b>	Give alive indications to the Watchdog Manager via AUTOSAR 4.0 API (deprecated).	
<b>Synopsis</b>	<pre>Std_ReturnType WdgM_ASR40_UpdateAliveCounter ( WdgM_ASR40_SupervisedEntityType SEID );</pre>	
<b>Service ID</b>	0x04	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Reentrant	
<b>Production Errors</b>	► <a href="#">WDGM_E_DATA_CORRUPTION</a> : thrown, if data corruption is detected in the internal WdgM data	
<b>Parameters (in)</b>	SEID	ID of supervised entity whose alive counter is updated
<b>Return Value</b>	Success of operation	
	E_OK	Operation successful
	E_NOT_OK	Operation failed
<b>Description</b>	This function updates the alive counter of a requested supervised entity of the current mode.	

#### 5.3.2.4.19. WdgM\_DeInit

<b>Purpose</b>	De-initialize the Watchdog Manager.
<b>Synopsis</b>	<pre>void WdgM_DeInit ( void );</pre>
<b>Service ID</b>	0x01
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non reentrant

#### 5.3.2.4.20. WdgM\_GetFirstExpiredSEID

<b>Purpose</b>	Get SEID that first reached WDGM_LOCAL_STATUS_EXPIRED.
<b>Synopsis</b>	<pre>Std_ReturnType WdgM_GetFirstExpiredSEID ( WdgM_ASR40_SupervisedEntityType * SEID );</pre>
<b>Service ID</b>	0x10
<b>Sync/Async</b>	Synchronous

<b>Reentrancy</b>	Reentrant	
<b>Parameters (out)</b>	SEID	Supervised entity ID that first reached WDG_M_LOCAL_STATUS_EXPIRED
<b>Return Value</b>	Success of operation	
	E_OK	Operation successful
	E_NOT_OK	Operation failed
<b>Description</b>	Returns SEID that first reached the state WDG_M_LOCAL_STATUS_EXPIRED.	

#### 5.3.2.4.21. WdgM\_GetGlobalStatus

<b>Purpose</b>	Get global supervision status of the Watchdog Manager.	
<b>Synopsis</b>	<pre>Std_ReturnType WdgM_GetGlobalStatus ( WdgM_GlobalStatusType * Status );</pre>	
<b>Service ID</b>	0x0d	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Reentrant	
<b>Parameters (out)</b>	Status	Global supervision status
<b>Return Value</b>	Success of operation	
	E_OK	Operation successful
	E_NOT_OK	Operation failed

#### 5.3.2.4.22. WdgM\_GetLocalStatus

<b>Purpose</b>	Get supervision status of a specific entity.	
<b>Synopsis</b>	<pre>Std_ReturnType WdgM_GetLocalStatus ( WdgM_ASR40_SupervisedEntityIdType SEID , WdgM_LocalStatusType * Status );</pre>	
<b>Service ID</b>	0x0c	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Reentrant	
<b>Parameters (in)</b>	SEID	ID of supervised entity whose status shall be returned
<b>Parameters (out)</b>	Status	Status of the given supervised entity
<b>Return Value</b>	Success of operation	

	E_OK	Operation successful
	E_NOT_OK	Operation failed

#### 5.3.2.4.23. WdgM\_GetMode

<b>Purpose</b>	Returns the current mode of the Watchdog Manager.	
<b>Synopsis</b>	<code>Std_ReturnType WdgM_GetMode ( WdgM_ModeType * Mode );</code>	
<b>Service ID</b>	0x0b	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Reentrant	
<b>Parameters (out)</b>	Mode	Current WdgM mode
<b>Return Value</b>	Success of operation	
	E_OK	Current mode successfully returned
	E_NOT_OK	Returning current mode failed

#### 5.3.2.4.24. WdgM\_GetVersionInfo

<b>Purpose</b>	Get version information of the Watchdog Manager.	
<b>Synopsis</b>	<code>void WdgM_GetVersionInfo ( Std_VersionInfoType * VersionInfo );</code>	
<b>Service ID</b>	0x02	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Non reentrant	
<b>Parameters (out)</b>	VersionInfo	Version information
<b>Description</b>	<p>This service returns the version information of this module. The version information includes</p> <ul style="list-style-type: none"> <li>▶ Module Id</li> <li>▶ Vendor Id</li> <li>▶ Vendor specific version numbers</li> </ul>	

#### 5.3.2.4.25. WdgM\_Init

<b>Purpose</b>	Initialize the Watchdog Manager.
----------------	----------------------------------

<b>Synopsis</b>	<code>void <b>WdgM_Init</b> ( const WdgM_ConfigType * ConfigPtr );</code>	
<b>Service ID</b>	0x00	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Non reentrant	
<b>Parameters (in)</b>	ConfigPtr	Pointer to configuration data, this parameter is ignored in the current implementation.

#### 5.3.2.4.26. WdgM\_MainFunction

<b>Purpose</b>	Cyclic main function for WdgM processing.	
<b>Synopsis</b>	<code>void <b>WdgM_MainFunction</b> ( void );</code>	
<b>Service ID</b>	0x08	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Non reentrant	
<b>Production Errors</b>	<ul style="list-style-type: none"> <li>▶ <a href="#">WDGM_E_MONITORING</a>: thrown, if supervision has failed for a supervised entity</li> <li>▶ <a href="#">WDGM_E_SET_MODE</a>: thrown, if watchdog drivers' mode switch has failed</li> <li>▶ <a href="#">WDGM_E_MF_TIMINGVIOLATION</a>: thrown, if WdgM_MainFunction period deviates from the configured mode-dependent schedule period (WdgMSupervision-Cycle)</li> <li>▶ <a href="#">WDGM_E_DATA_CORRUPTION</a>: thrown, if data corruption is detected in the internal WdgM data</li> </ul>	
<b>Description</b>	Performs the processing of the cyclic Watchdog Manager jobs.	

#### 5.3.2.4.27. WdgM\_PerformReset

<b>Purpose</b>	Force a watchdog reset.	
<b>Synopsis</b>	<code>void <b>WdgM_PerformReset</b> ( void );</code>	
<b>Service ID</b>	0x0f	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Non reentrant	
<b>Description</b>	Instructs the Watchdog Manager to cause a watchdog reset.	



#### 5.3.2.4.28. WdgM\_SetMode

<b>Purpose</b>	Set the current mode of the Watchdog Manager.	
<b>Synopsis</b>	<pre>Std_ReturnType WdgM_SetMode ( WdgM_                                ModeType Mode , uint16 CallerID );</pre>	
<b>Service ID</b>	0x03	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Non reentrant	
<b>Production Errors</b>	► <a href="#">WDGM_E_IMPROPER_CALLER</a> : thrown, if the passed CallerID is not in the list of the configured list of allowed CallerIDs	
<b>Parameters (in)</b>	Mode	One of the configured Watchdog Manager modes
	CallerID	Module ID of the calling module
<b>Return Value</b>	Success of operation	
	E_OK	Successfully changed to the new mode
	E_NOT_OK	Changing to the new mode failed

### 5.3.3. Integration notes

#### 5.3.3.1. Exclusive areas

Exclusive areas are not used by the WdgM module.

#### 5.3.3.2. Production errors

<a href="#">WDGM_E_DATA_CORRUPTION</a>	<ul style="list-style-type: none"> <li>► <a href="#">WdgM_ASR32_UpdateAliveCounter</a></li> <li>► <a href="#">WdgM_ASR40_CheckpointReached</a></li> <li>► <a href="#">WdgM_ASR40_UpdateAliveCounter</a></li> <li>► <a href="#">WdgM_MainFunction</a></li> </ul>
<a href="#">WDGM_E_IMPROPER_CALLER</a>	► <a href="#">WdgM_SetMode</a>

<a href="#">WDGM_E_MF_TIMINGVIOLATION</a>	▶ <a href="#">WdgM_MainFunction</a>
<a href="#">WDGM_E_MONITORING</a>	▶ <a href="#">WdgM_MainFunction</a>
<a href="#">WDGM_E_SET_MODE</a>	▶ <a href="#">WdgM_MainFunction</a>

### 5.3.3.3. Memory mapping

General information about memory mapping is provided in the EB tresos AutoCore Generic documentation. Refer to the section [Memory mapping and compiler abstraction](#) in the [Integration notes](#) section for details.

The following table provides the list of sections that may be mapped for this module:

Memory section
CODE
CODE_ASIL_D
CONST_8
CONST_ASIL_D_8
CONST_16
CONST_ASIL_D_16
CONST_ASIL_D_UNSPECIFIED
VAR_CLEARED_ASIL_D_UNSPECIFIED
VAR_INIT_ASIL_D_UNSPECIFIED
VAR_CLEARED_16
VAR_CLEARED_ASIL_D_16
VAR_CLEARED_ASIL_D_32
VAR_INIT_ASIL_D_8
VAR_INIT_8
VAR_INIT_16
VAR_CLEARED_UNSPECIFIED
VAR_CLEARED_ASIL_D_8
VAR_INIT_ASIL_D_LOCAL_8
VAR_INIT_UNSPECIFIED
VAR_INIT_GLOBAL_32
VAR_POWER_ON_CLEARED_ASIL_D_UNSPECIFIED

VAR_GLOBAL_32
VAR_CLEARED_GLOBAL_UNSPECIFIED
SHARED_VAR_CLEARED_GLOBAL_UNSPECIFIED
VAR_CLEARED_32
VAR_POWER_ON_INIT_ASIL_D_16
VAR_INIT_ASIL_D_32
VAR_POWER_ON_INIT_ASIL_D_UNSPECIFIED
SHARED_VAR_CLEARED_UNSPECIFIED
SHARED_VAR_INIT_16
SHARED_VAR_INIT_UNSPECIFIED
VAR_INIT_8BIT
CONST_8BIT
VAR_8BIT
VAR_CLEARED_8BIT
CONST_UNSPECIFIED
VAR_32BIT
VAR_CLEARED_32BIT
CONST_16BIT
VAR_CLEARED_16BIT

#### 5.3.3.4. Integration requirements

##### WARNING



##### Integration requirements list is not exhaustive

The following list of integration requirements helps you to integrate your product. However, this list is not exhaustive. You also require information from the user's guide, release notes, and EB tresos AutoCore known issues to successfully integrate your product.

##### 5.3.3.4.1. lim.WdgM.EB\_INTREQ\_WdgM\_0001

<b>Description</b>	If the WdgM_PerformReset API can interrupt the WdgM_MainFunction, the Watchdog Manager may postpone the update of the Watchdog triggering conditions to 0 by one main function cycle due to race conditions.
<b>Rationale</b>	Existing race condition in case WdgM_PerformReset interrupt has higher priority than WdgM_MainFunction.

#### 5.3.3.4.2. lim.WdgM.EB\_INTREQ\_WdgM\_0002

<b>Description</b>	If the WdgM is integrated in an environment where the individual applications are allocated to different partitions, then the general parameter WdgMPartitioningEnabled must be enabled. The WdgM then makes use of the memory section WDGM_START/STOP_SEC_SHARED_VAR_CLEARED_UNSPECIFIED which is required for the access to the shared graph data (Logical Supervision) in the context of different applications / Supervised Entities. The integrity of the shared data is ensured by holding all values double inverse. Therefore, the integrator must configure this section to be globally accessible (write and readably) from all partitions.
<b>Rationale</b>	This version of the WdgM module is tailored for the use in a single core environment where the shared memory approach is more efficient than splitting the memory among the different partitions.

#### 5.3.3.4.3. lim.WdgM.EB\_INTREQ\_WdgM\_0003

<b>Description</b>	If mode switch is done synchronous (WdgMSetModeSynchron is set to TRUE) then the API WdgM_SetMode should not be preempt by WdgM_DeInit.
<b>Rationale</b>	Both APIs are setting the watchdog to a specific mode. The preemption of WdgM_SetMode by WdgM_DeInit may lead to inconsistency in the watchdog mode.

#### 5.3.3.4.4. lim.WdgM.EB\_INTREQ\_WdgM\_0004

<b>Description</b>	If mode switch is done synchronous (WdgMSetModeSynchron is set to TRUE) and the WdgM_SetMode API can interrupt the WdgM_CheckpointReached, the Watchdog Manager may postpone the evaluation of the Supervised Entity results if it is not used in the new mode until it is activated again in a new mode, due to race conditions.
<b>Rationale</b>	Existing race condition in case WdgM_SetMode interrupt has higher priority than WdgM_CheckpointReached.

#### 5.3.3.4.5. lim.WdgM.EB\_INTREQ\_WdgM\_0005

<b>Description</b>	If mode switch is done synchronous (WdgMSetModeSynchron is set to TRUE) and the WdgM_MainFunction API can interrupt the WdgM_SetMode, the Watchdog Manager may evaluate results of the Supervised Entities from the old mode and the new mode, until the next main function, due to race conditions.
<b>Rationale</b>	Existing race condition in case WdgM_MainFunction interrupt has higher priority than WdgM_CheckpointReached.

#### 5.3.3.4.6. lim.WdgM.EB\_INTREQ\_WdgM\_0006

<b>Description</b>	If mode switch is done synchronous (WdgMSetModeSynchron is set to TRUE) then WdgM_SetMode should not interrupt the WdgM_MainFunction.
<b>Rationale</b>	WdgMFailedAliveSupervisionRefCycle of a Supervised Entity which is not used in the new mode when WdgM_SetMode is called may have a value of 1 instead of 0 when the Supervised Entity is reactivated in a new mode.

#### 5.3.3.4.7. lim.WdgM.EB\_INTREQ\_WdgM\_0007

<b>Description</b>	If legacy symbolic names (from AUTOSAR versions 3.x or lower than 4.0.3 ) are used then the macro WDGM_PROVIDE_LEGACY_SYMBOLIC_NAMES shall be defined before including the WdgM header file.
<b>Rationale</b>	The usage of legacy symbolic names is discouraged, but for backwards compatibility the usage of these symbolic names is provided. Recommendation is to go for AUTOSAR 4.0.3 symbolic names (See TPS_ECUC_02108 from AUTOSAR_TPS_ECU-Configuration.pdf) and not enable the macro.