Document Title	Specification of Watchdog Interface
Document Owner	AUTOSAR GbR
Document Responsibility	AUTOSAR GbR
Document Identification No	041
Document Classification	Standard

Document Version	2.2.1
Document Status	Final
Part of Release	3.1
Revision	0001

Document Change History

Date	Version	Changed by	Change Description
23.06.2008	2.2.1	AUTOSAR Administration	Legal disclaimer revised
07.12.2007	2.2.0	AUTOSAR Administration	• Tables of chapter 8 has been replaced with Contents generated from AUTOSAR BSW model • Document meta information extended • Small layout adaptations made
31.01.2007	2.1.0	AUTOSAR Administration	• In chapter 5.1.2 the file include structure has been changed to comply with the SPAL general include structure. • Legal disclaimer revised • Release Notes added • “Advice for users” revised • “Revision Information” added
20.03.2006	2.0.0	AUTOSAR Administration	Document structure adapted to common Release 2.0 SWS Template
23.06.2005	1.0.0	AUTOSAR Administration	Initial release

Page left intentionally blank

Disclaimer

This document of a specification as released by the AUTOSAR Development Partnership is intended **for the purpose of information only**. The commercial exploitation of material contained in this specification requires membership of the AUTOSAR Development Partnership or an agreement with the AUTOSAR Development Partnership. The AUTOSAR Development Partnership will not be liable for any use of this specification. Following the completion of the development of the AUTOSAR specifications commercial exploitation licenses will be made available to end users by way of written License Agreement only.

No part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from the publisher." The word AUTOSAR and the AUTOSAR logo are registered trademarks.

Copyright © 2004-2008 AUTOSAR Development Partnership. All rights reserved.

Advice to users of AUTOSAR Specification Documents:

AUTOSAR Specification Documents may contain exemplary items (exemplary reference models, "use cases", and/or references to exemplary technical solutions, devices, processes or software).
Any such exemplary items are contained in the Specification Documents for illustration purposes only, and they themselves are not part of the AUTOSAR Standard. Neither their presence in such Specification Documents, nor any later documentation of AUTOSAR conformance of products actually implementing such exemplary items, imply that intellectual property rights covering such exemplary items are licensed under the same rules as applicable to the AUTOSAR Standard.

Table of Contents

1 Introduction and functional overview

This specification describes the functionality, API and the configuration of the AUTOSAR Basic Software module Watchdog Driver Interface.

In case of more than one watchdog device and watchdog driver (e.g. both an internal software watchdog and an external hardware watchdog) being used on an ECU, this module allows the watchdog manager to select the correct watchdog driver - and thus the watchdog device - while retaining the API and functionality of the underlying driver.

WDGIF026: The Watchdog Driver Interface provides uniform access to services of the underlying watchdog drivers like mode switching and triggering. The appropriate watchdog driver is selected by a device index. The behaviour (synchronous / asynchronous / timing) of the services of the watchdog drivers is preserved.

2 Acronyms and abbreviations

Note: For this module there are no local acronyms and abbreviations. All used acronyms and abbreviations should be contained in the AUTOSAR glossary.

3 Related documentation

3.1 Input documents

[1] Layered Software Architecture
https:/svn2.autosar.org/repos2/22_Releases
AUTOSAR_LayeredSoftwareArchitecture.pdf

[2] General Requirements on Basic Software Modules
https:/svn2.autosar.org/repos2/22_Releases
AUTOSAR_SRS_General.pdf

[3] General Requirements on SPAL
https:/svn2.autosar.org/repos2/22_Releases
AUTOSAR_SRS_SPAL_General.pdf

[4] Requirements on Memory Hardware Abstraction Layer
https:/svn2.autosar.org/repos2/22_Releases
AUTOSAR_SRS_MemHw_AbstractionLayer.pdf

[5] Specification of Watchdog Driver
https:/svn2.autosar.org/repos2/22_Releases
AUTOSAR_SWS_WatchdogDriver.pdf

[6] Specification of Development Error Tracer
https:/svn2.autosar.org/repos2/22_Releases
AUTOSAR_SWS_DET.pdf

[7] AUTOSAR Basic Software Module Description Template,
https://svn2.autosar.org/repos2/22_Releases/
AUTOSAR_BSW_Module_Description.pdf

3.2 Related standards and norms

None

4 Constraints and assumptions

4.1 Limitations

No limitations.

4.2 Applicability to car domains

No restrictions.

5 Dependencies to other modules

The Watchdog Driver Interface is part of the ECU Abstraction Layer. It allows the upper layer, i.e. the watchdog manager, to uniformly access one or more watchdog drivers. The implementation of the Watchdog Driver Interface therefore depends on the number of watchdog drivers below.

5.1 File structure

5.1.1 Code file structure

WDGIF037: The code file structure shall not be defined within this specification.

5.1.2 Header file structure

WDGIF001: The Watchdog Driver Interface shall consist of the following parts:

- An API header file “`WdgIf.h`” for accessing the underlying watchdog drivers
- A type header file “`WdgIf_Types.h`” providing standard types for both the watchdog drivers and the watchdog manager
- A configuration header file “`WdgIf_Cfg.h`” providing platform and device specific types for both the watchdog drivers and the watchdog manager
- If required, an implementation source file `WdgIf.c` (e.g. for tables of function pointers)

WDGIF002: The file include structure shall be as follows:

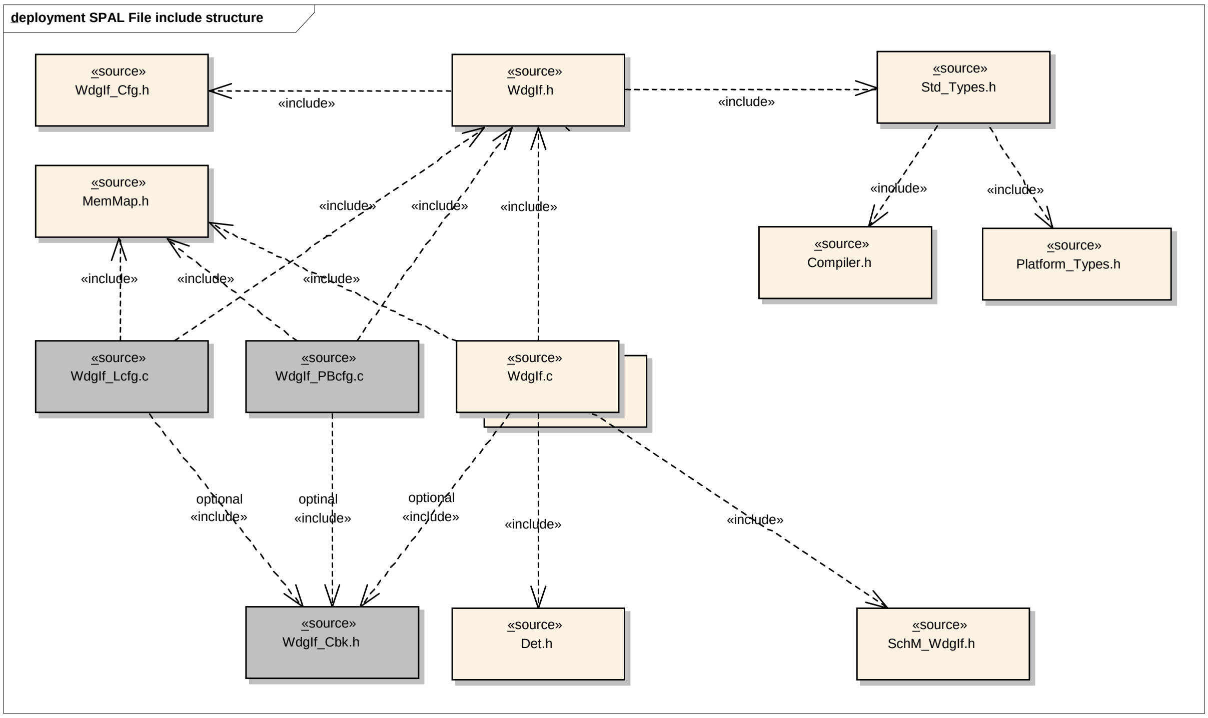

Figure 1: File include structure of the Watchdog Interface

- WdgIf_Types.h shall include the standard, platform and compiler specific header files (not shown).
- WdgIf_Types.h shall be included in the header files of all underlying watchdog drivers
- WdgIf_Cfg.h shall include the header files of all underlying watchdog drivers
- WdgIf.h shall include WdgIf_Cfg.h
- If imeplemented, WdgIf.c shall include WdgIf.h
- Only WdgIf.h shall be included by the upper layer (not shown)

6 Requirements traceability

Document: General Requirements on Basic Software Modules

Requirement	***Satisfied by***
[[BSW00344] Reference to link-time configuration	Not applicable (this module only provides pre-compile time parameters)
BSW00404] Reference to post build time configuration	Not applicable (this module only provides pre-compile time parameters)
[BSW00405] Reference to multiple configuration sets	Not applicable (this module does not provide an initialization routine)
[BSW00345] Pre-compile-time configuration	WDGIF033
[BSW159] Tool-based configuration	WDGIF033
[BSW167] Static configuration checking	WDGIF005
[BSW171] Configurability of optional functionality	WDGIF033, WDGIF040
[BSW170] Data for reconfiguration of AUTOSAR SW-components	Not applicable (this module does not depend on faults, signals, …)
[BSW00380] Separate C-File for configuration parameters	Not applicable (this module only provides pre-compile time parameters)
[BSW00419] Separate C-Files for pre-compile time configuration parameters	Not applicable (this module does only provide #define's as pre-compile time configuration parameters)
BSW00381] Separate configuration header file for pre-compile time parameters	WDGIF001, WDGIF002
[BSW00412] Separate H-File for configuration parameters	Not applicable (this module only provides pre-compile time parameters)
[BSW00382] Not-used configuration elements need to be listed	Not applicable (there are no not-used configuration elements for this module)
[BSW00383] List dependencies of configuration files	Not applicable (this module does not use configuration files from other modules)
[BSW00384] List dependencies to other modules	Chapter 5
[BSW00387] Specify the configuration class of callback function	Not applicable (this module does not provide any callback functions)
[BSW00388] Introduce containers	Chapter 10.2
[BSW00389] Containers shall have names	Chapter 10.2.2
[BSW00390] Parameter content shall be unique within the module	Chapter 10.2.2
[BSW00391] Parameter shall have unique names	Chapter 10.2.2
[BSW00392] Parameters shall have a type	Chapter 10.2.2
[BSW00393] Parameters shall have a range	Chapter 10.2.2
[BSW00394] Specify the scope of the parameters	Chapter 10.2.2
[BSW00395] List the required parameters (per parameter)	Chapter 10.2.2
[BSW00396] Configuration classes	Chapter 10.2.2
[BSW00397] Pre-compile-time parameters	Chapter 10.2.2
[BSW00398] Link-time parameters	Not applicable (this module does not provide any link-time parameters)
[BSW00399] Loadable Post-build time	Not applicable

parameters	(this module does not provide any post build parameters)
[BSW00400] Selectable Post-build time parameters	Not applicable (this module does not provide any post build parameters)
[BSW00402] Published information	Chapter 10.3
[BSW00375] Notification of wake-up reason	Not applicable (this module does not wake up the ECU / MCU)
[BSW101] Initialization interface	Not applicable (the module does not need to be initialized)
[BSW00416] Sequence of Initialization	Not applicable (requirement on system integration, not on a single module)
[BSW00406] Check module initialization	Not applicable (the module does not need to be initialized)
[BSW168] Diagnostic Interface of SW components	Not applicable (the module does not support a special diagnostic interface)
[BSW00407] Function to read out published parameters	Chapter 8.3.3
[BSW00423] Usage of SW-C template to describe BSW modules with AUTOSAR Interfaces	Not applicable (this module does not provide an AUTOSAR interface)
[BSW00424] BSW main processing function task allocation	Not applicable (this module does not provide a main function)
[BSW00425] Trigger conditions for schedulable objects	Not applicable (this module does not provide any scheduled objects)
[BSW00426] Exclusive areas in BSW modules	Not applicable (this module does not have any exclusive areas)
[BSW00427] ISR description for BSW modules	Not applicable (this module does not implement any ISRs)
[BSW00428] Execution order dependencies of main processing functions	Not applicable (this module does not provide a main function)
[BSW00429] Restricted BSW OS functionality access	Not applicable (this module does not use any OS functions or objects)
[BSW00431] The BSW Scheduler module implements task bodies	Not applicable (requirement on the BSW task scheduler)
[BSW00432] Modules should have separate main processing functions for read/receive and write/transmit data path	Not applicable (this module does not provide a main function, much less two)
[BSW00433] Calling of main processing functions	Not applicable (requirement on the BSW task scheduler)
[BSW00434] The Schedule Module shall provide an API for exclusive areas	Not applicable (requirement on the BSW task scheduler)
[BSW00336] Shutdown interface	Not applicable (the module does not need to be shut down)
[BSW00337] Classification of errors	WDGIF006, WDGIF009
[BSW00338] Detection and Reporting of development errors	WDGIF007
[BSW00369] Do not return development error codes via API	WDGIF007
[BSW00339] Reporting of production relevant error status	Not applicable (no production relevant errors)
[BSW00421] Reporting of production relevant error events	Not applicable (no production relevant errors)
[BSW00422] Debouncing of production relevant error status	Not applicable (requirement for DEM, not a general

	requirement)
[BSW00420] Production relevant error event rate detection	Not applicable (requirement for DEM, not a general requirement)
[BSW00417] Reporting of Error Events by Non-Basic Software	Not applicable (this is a BSW module)
[BSW00323] API parameter checking	WDGIF028
[BSW004] Version check	WDGIF005
[BSW00409] Header files for production code error IDs	WDGIF009
[BSW00385] List possible error notificatons	WDGIF006
[BSW00386] Configuration for detecting an error	WDGIF006, WDGIF007, WDGIF031, WDGIF033
[BSW161] Microcontroller abstraction	Not applicable (requirement on AUTOSAR architecture, not a single module)
[BSW162] ECU layout abstraction	Not applicable (requirement on AUTOSAR architecture, not a single module)
[BSW00324] Do not use HIS I/O Library	Not applicable (architecture decision)
[BSW005] No hard coded horizontal interfaces within MCAL	Not applicable (requirement on AUTOSAR architecture, not a single module)
[BSW00415] User dependent include files	Not applicable (only one user for this module)
[BSW164] Implementation of interrupt service routines	Not applicable (this module does not implement any ISRs)
[BSW00325] Runtime of interrupt service routines	Not applicable (this module does not implement any ISRs)
[BSW00326] Transition from ISRs to OS tasks	Not applicable (this module does not implement any ISRs)
[BSW00342] Usage of source code and object code	Not applicable (requirement on AUTOSAR architecture, not a single module)
[BSW00343] Specification and configuration of time	Not applicable (no configurable timings)
[BSW160] Human-readable configuration data	Not applicable (requirement on documentation, not on specification)
[BSW007] HIS MISRA C	Not applicable (requirement on implementation, not on specification)
[BSW00300] Module naming convention	Not applicable (requirement on implementation, not on specification)
[BSW00413] Accessing instances of BSW modules	Not applicable (this is not a driver)
[BSW00347] Naming separation of different instances of BSW drivers	Not applicable (this is not a driver)
[BSW00305] Self-defined data types naming convention	Chapter 8.2
[BSW00307] Global variables naming convention	Not applicable (requirement on the implementation, not on the specification)
[BSW00310] API naming convention	Chapters 8.3.1, 8.3.2, 8.3.3
[BSW00373] Main processing function naming convention	Not applicable (this module does not provide a main processing function)
[BSW00327] Error values naming convention	WDGIF006

[BSW00335] Status values naming convention	Not applicable (this module does not provide an internal status variable)
[BSW00350] Development error detection keyword	WDGIF007, WDGIF031, WDGIF033
[BSW00408] Configuration parameter naming convention	Chapter 10.2.2
[BSW00410] Compiler switches shall have defined values	Chapter 10.2.2
[BSW00411] Get version info keyword	Chapter 10.2.2
[BSW00346] Basic set of module files	WDGIF001
[BSW158] Separation of configuration from implementation	WDGIF001
[BSW00314] Separation of interrupt frames and service routines	Not applicable (this module does not implement any ISRs)
[BSW00370] Separation of callback interface from API	Not applicable (this module does not provide any callback routines)
[BSW00348] Standard type header	WDGIF001
[BSW00353] Platform specific type header	WDGIF002
[BSW00361] Compiler specific language extension header	WDGIF002
[BSW00301] Limit imported information	WDGIF001
[BSW00302] Limit exported information	Not applicable (requirement on the implementation, not on the specification)
[BSW00328] Avoid duplication of code	Not applicable (requirement on the implementation, not on the specification)
[BSW00312] Shared code shall be reentrant	Not applicable (requirement on the implementation, not on the specification)
[BSW006] Platform independency	Not applicable (this is a module of the microcontroller abstraction layer)
[BSW00357] Standard API return type	Chapter 8.3.1
[BSW00377] Module specific API return types	Not applicable (no module specific return types)
[BSW00304] AUTOSAR integer data types	Not applicable (requirement on implementation, not for specification)
[BSW00355] Do not redefine AUTOSAR integer data types	Not applicable (requirement on implementation, not for specification)
[BSW00378] AUTOSAR oolean type	Not applicable (requirement on implementation, not for specification)
[BSW00306] Avoid direct use of compiler and platform specific keywords	Not applicable (requirement on implementation, not for specification)
[BSW00308] Definition of global data	Not applicable (requirement on implementation, not for specification)
[BSW00309] Global data with read-only constraint	Not applicable (requirement on implementation, not for specification)
[BSW00371] Do not pass function pointers via API	Not applicable (no function pointers in this specification)
[BSW00358] Return type of init() functions	Not applicable

	(this module does not need to be initiaized)
[BSW00376] Return type and parameters of main processing functions	Not applicable (this module does not provide a main processing function)
[BSW00359] Return type of callback functions	Not applicable (this module does not provide any callback routines)
[BSW00360] Parameters of callback functions	Not applicable (this module does not provide any callback routines)
[BSW00329] Avoidance of generic interfaces	Chapters 8.3.1, 8.3.2, 8.3.3 (explicit interfaces defined)
[BSW00330] Usage of macros / inline functions instead of functions	Not applicable (requirement on implementation, not for specification)
[BSW00331] Separation of error and status values	Not applicable (this module does not provide any internal status variable)
[BSW009] Module User Documentation	Not applicable (requirement on documentation, not on specification)
[BSW00401] Documentation of multiple instances of configuration parameters	Not applicable (this module does not need to be initiaized)
[BSW172] Compatibility and documentation of scheduling strategy	Not applicable (no internal scheduling policy)
[BSW010] Memory resource documentation	Not applicable (requirement on documentation, not on specification)
[BSW00333] Documentation of callback function context	Not applicable (this module does not provide any callback routines)
[BSW00374] Module vendor identification	WDGIF034
[BSW00379] Module identification	WDGIF034
[BSW003] Version identification	WDGIF034
[BSW00318] Format of module version numbers	WDGIF034
[BSW00321] Enumeration of module version numbers	Not applicable (requirement on implementation, not for specification)
[BSW00341] Microcontroller compatibility documentation	Not applicable (requirement on documentation, not on specification)
[BSW00334] Provision of XML file	Not applicable (requirement on documentation, not on specification)
[BSW00435] Module Header File Structure for the Basic Software Scheduler	Chapter 5.1.2
[BSW00436] Module Header File Structure for the Basic Software Memory Mapping	Chapter 5.1.2

Document: General Requirements on SPAL

Requirement	***Satisfied by***
[BSW12263] Object code compatible configuration concept	Not applicable (the module is not configurable at runtime)
[BSW12056] Configuration of notification mechanisms	Not applicable (the module does not support any notification mechanism)
[BSW12267] Configuration of wake-up sources	Not applicable (the module does not wake up the ECU / MCU)

[BSW12057] Driver module initialization	Not applicable (the module does not support initialization)
[BSW12125] Initialization of hardware resources	Not applicable (the module does not support initialization)
[BSW12163] Driver module de-initialization	Not applicable (the module does not support initialization)
[BSW12058] Individual initialization of overall registers	Not applicable (the module does not support initialization)
[BSW12059] General initialization of overall registers	Not applicable (the module does not support initialization)
[BSW12060] General initialization of one-time writable registers	Not applicable (the module does not support initialization)
[BSW12062] Selection of static configuration sets	Not applicable (the module is not configurable at runtime)
[BSW12461] Responsibility for register initialization	Not applicable (the module does not support initialization)
[BSW12462] Provide settings for register initialization	Not applicable (requirement on implementation, not on specification)
[BSW12463] Combine and forward settings for register initialization	Not applicable (requirement on configuration, not on specification)
[BSW12062] Selection of static configuration sets	Not applicable (the module does not support initialization)
[BSW12068] MCAL initialization sequence	Not applicable (not a requirement for a SW module but for system integration)
[BSW12069] Wake-up notification of ECU State Manager	Not applicable (the module does not wake up the ECU / MCU)
[BSW157] Notification mechanisms of drivers and handlers	Not applicable (the module does not support any notification mechanism)
[BSW12155] Prototypes of callback functions	Not applicable (the module does not provide any callback functions)
[BSW12169] Control of operation mode	Not applicable (the module does not support different operating modes)
[BSW12063] Raw value mode	Not applicable (the module does not provide any data to the user)
[BSW12075] Use of application buffers	Not applicable (the module does not operate on buffers)
[BSW12129] Resetting of interrupt flags	Not applicable (the module does not implement any interrupt service routines)
[BSW12171] Support of synchronous and asynchronous SPI interface	Not applicable (that is a requirement for an SPI driver)
[BSW12064] Change of operation mode during running operation	Not applicable (the module does not support different operating modes)
[BSW12448] Behavior after development error detection	WDGIF028
[BSW12067] Setting of wake-up conditions	Not applicable (the module does not wake up the ECU / MCU)
[BSW12077] Non-blocking implementation	Not applicable (no long term loops)
[BSW12078] Runtime and memory efficiency	Not applicable (requirement for implementation, not for

	specification)
[BSW12092] Access to drivers	Not applicable (only interface to watchdog drivers)
[BSW12265] Configuration data shall be kept constant	Not applicable (no configuration data)
[BSW12264] Specification of configuration items	WDGIF033
[BSW12081] Use HIS requirements as input	Not applicable (no requirements, only specification available)

Document: Requirements on Watchdog Driver

Requirement	***Satisfied by***
[BSW12015] Configuration of watchdog modes	WDGIF016
[BSW12105] Watchdog initialization	Not applicable (the module does not support initialization)
[BSW12106] Prohibit disabling of watchdog	Not applicable (the module does not support initialization)
[BSW12018] Watchdog mode selection service	Not applicable (the module does not support different operating modes)
[BSW12019] Watchdog trigger service	WDGIF017
[BSW12165] Functional scope	WDGIF017, WDGIF026
[BSW12166] SPI channel configuration	Not applicable (the module is not configurable at runtime)
[BSW12167] Common Watchdog API	WDGIF017
[BSW12168] Microcontroller independency	Not applicable (requirement for implementation, not for specification)

Document: Requirements on Memory Hardware Abstraction Layer

Requirement	***Satisfied by***
BSW14019 Provide uniform access to underlying memory abstraction modules	WDGIF017, WDGIF026
BSW14020 Selection of underlying memory abstraction modules	WDGIF018
BSW14021 Number of underlying memory abstraction modules	WDGIF019, WDGIF020, WDGIF033
BSW14022 Preserving of functionality	WDGIF003, WDGIF004
BSW14023 Parameter checking	WDGIF005, WDGIF028
BSW14024 Preserving of timing behavior	WDGIF003, WDGIF004
BSW14025 Efficient implementation	WDGIF019, WDGIF020

7 Functional specification

7.1 General behavior

WDGIF003: The Watchdog Driver Interface shall not add functionality to the watchdog drivers. Also the Watchdog Driver Interface does not abstract from watchdog properties like toggle or window mode, timeout periods etc. that is it does not hide any features of the underlying watchdog driver and watchdog hardware.

WDGIF004: The Watchdog Driver Interface shall not change the behavior of the services of the underlying watchdog drivers.

WDGIF005: The configuration parameters shall be checked statically (at least during compile time) for correctness. The version information in the module header and source files shall be validated and consistent (e.g. by comparing the version information in the module header and source files with a pre-processor macro).

7.2 Error classification

WDGIF006: The following errors and exceptions shall be detectable by the Watchdog Driver Interface depending on its configuration (development / production).

Type or error	***Relevance***	***Related error code***	***Value [hex]***
API service called with wrong device index parameter	Development	`WDGIF_E_PARAM_DEVICE`	`0x01`

WDGIF029: Values for production code Event Ids are assigned externally by the configuration of the Dem. They are published in the file Dem_IntErrId.h and included via Dem.h.

WDGIF030: Development error values are of type uint8.

7.3 Error detection

WDGIF007: The detection of development errors is configurable (*ON* / *OFF*) at pre-compile time. The switch *`WDGIF_DEV_ERROR_DETECT`* (see chapter 10) shall activate or deactivate the detection of all development errors.

WDGIF031: If the *`WDGIF_DEV_ERROR_DETECT`* switch is enabled API parameter checking is enabled. The detailed description of the detected errors can be found in chapter 7.2and chapter 1.

7.4 Error notification

WDGIF032: Detected development errors shall be reported to the Development Error Tracer (DET) if the pre-processor switch *WDGIF_DEV_ERROR_DETECT* is set (see chapter 10).

WDGIF009: A detection of errors not listed in the table above [WDGIF006] shall not be implemented.

7.5 API parameter checking

WDGIF028: If more than one watchdog driver is configured and the development error detection is enabled for this module, the parameter DeviceIndex shall be checked for being an existing device within the module's services. Detected errors shall be reported to the Development Error Tracer (DET) with the error code WDGIF_E_PARAM_DEVICE and the called service shall not be executed, if the called function has a return value this value shall be set E_NOT_OK.

8 API specification

8.1 Imported types

In this chapter all types included from the following files are listed:

WDGIF041:

Header file	***Imported Type***
Std_Types.h	Std_ReturnType
	Std_VersionInfoType

WDGIF010: The types specified in this chapter shall be located in the file WdgIf_Types.h.

WDGIF011: The types specified in this chapter shall not be changed or extended for a specific watchdog device or platform.

WDGIF013: The data type for the watchdog device index shall be uint8. The lowest value to be used for this device index shall be 0. The allowed range of indices thus shall be 0 .. `WDGIF_NUMBER_OF_DEVICES-1`.

8.2 Type definitions

8.2.1 WdgIf_StatusType

WdgIf_StatusType

Name:	`WdgIf_StatusType`	
Type:	`Enumeration`	
Range:	`WDGIF_UNINIT`	The watchdog driver is not initialized or not usable.
	`WDGIF_IDLE`	The watchdog driver is currently idle, i.e. it is not being switched between modes or triggered.
	`WDGIF_BUSY`	The watchdog driver is currently being switched between modes or triggered.
Description:	Status type of the WdgIf module	

WDGIF015: This status shall be used internally by the underlying watchdog driver(s) if they are configured for development mode.

WDGIF014: This (WDGIF_UNINIT) shall be the default value after reset. This status shall have the value 0.

8.2.2 WdgIf_ModeType

Wdgif_ModeType

Name:	WdgIf_ModeType	
Type:	Enumeration	
Range:	WDGIF_OFF_MODE	In this mode, the watchdog driver is disabled (switched off).
	WDGIF_SLOW_MODE	In this mode, the watchdog driver is set up for a long timeout period (slow triggering).
	WDGIF_FAST_MODE	In this mode, the watchdog driver is set up for a short timeout period (fast triggering).
Description:	Mode type of the WdgIf module	

WDGIF016: These values shall be passed as parameters to the watchdog drivers mode switching function (Wdg_SetMode). The hardware specific settings behind these modes shall be given in the watchdog drivers configuration set.

8.3 Function definitions

WDGIF017: The API specified in this chapter shall be mapped to the API of the underlying drivers. For functional behavior refer to the specification of the watchdog driver

WDGIF018: The parameter `DeviceIndex` shall be used for selection of watchdog drivers. If only one watchdog driver is configured, the parameter `DeviceIndex` shall be ignored.

WDGIF019: If only one watchdog driver is configured, the Watchdog Driver Interface shall be implemented as a set of macros mapping the Watchdog Driver Interface API to the watchdog driver API.

Example:

```
#define WdgIf_SetMode(DeviceIndex, WdgMode) \
      Wdg_SetMode(WdgMode)
```

WDGIF020: If more than one watchdog driver is configured, the Watchdog Driver Interface shall use efficient mechanisms to map the API calls to the appropriate watchdog driver. One solution is to use tables of pointers to functions where the parameter `DeviceIndex` is used as array index.

Example:

```
#define WdgIf_SetMode(DeviceIndex, WdgMode) \
  SetModeFctPtr[DeviceIndex](WdgMode)
```

Note: The service IDs are related to the service IDs of the watchdog driver specification (see [5]). For that reason, they may not start with 0.

8.3.1 WdgIf_SetMode

Wdgif_SetMode

WDGIF042:

Service name:	WdgIf_SetMode	

Syntax:	`Std_ReturnType WdgIf_SetMode(`
	`    uint8 DeviceIndex,`
	`    WdgIf_ModeType WdgMode`
	`)`	

Service ID[hex]:	0x01	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	DeviceIndex	--
	WdgMode	--
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	--
Description:	map the service WdgIF_SetMode to the service Wdg_SetMode of the corresponding Watchdog Driver	

WDGIF043: Mapped to service: Wdg_SetMode.

8.3.2 WdgIf_Trigger

Wdgif_Trigger

WDGIF044:

Service name:	WdgIf_Trigger	

Syntax:	`void WdgIf_Trigger(`
	`    uint8 DeviceIndex`
	`)`	

Service ID[hex]:	0x02	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	DeviceIndex	--
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	map the service WdgIF_Trigger to the service Wdg_Trigger of the corresponding Watchdog Driver	

WDGIF045: Mapped to service: Wdg_Trigger.

8.3.3 WdgIf_GetVersionInfo

WDGIF046:

Service name:	WdgIf_GetVersionInfo	

Syntax:	`void WdgIf_GetVersionInfo(`
	`    Std_VersionInfoType* VersionInfoPtr`
	`)`	

Service ID[hex]:	0x03	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (inout):	None	
Parameters (out):	VersionInfoPtr	Pointer to where to store the version information of this module.
Return value:	None	
Description:	Returns the version information.	

WDGIF035: The WdgIf_GetVersionInfo service returns the version information of this module. The version information includes:

- Module Id
- Vendor Id
- Vendor specific version numbers (BSW00407).

WDGIF036: The WdgIf_GetVersionInfo function shall be pre compile time configurable `On/Off` by the configuration parameter: `WDGIF_VERSION_INFO_API`

Hint:
If source code for caller and callee of this function is available this function should be realized as a macro. The macro should be defined in the modules header file.

Configuration:
WDGIF040: The WdgIf_GetVersionInfo function is only available if the pre-processor switch `WDGIF_VERSION_INFO_API` is set.

8.4 Call-back notifications

This module does not provide any callback functions.

8.5 Scheduled functions

This module does not need any scheduled functions.

8.6 Expected Interfaces

In this chapter all interfaces required from other modules are listed.

8.6.1 Mandatory Interfaces

This chapter defines all interfaces which are required to fulfill the core functionality of the module.

WDGIF047:

API function	***Description***
Wdg_SetMode	Switches the watchdog into the mode Mode.
Wdg_Trigger	Triggers the watchdog hardware. It has to be called cyclically by some upper layer function (usually the watchdog manager) in order to prevent the watchdog hardware from expiring.

8.6.2 Optional Interfaces

This chapter defines all interfaces which are required to fulfill an optional functionality of the module.

WDGIF048:

API function	***Description***
Det_ReportError	Service to report development errors.

8.6.3 Configurable interfaces

There are no configurable interfaces for this module.

9 Sequence diagrams

Refer to specification of watchdog driver.

10 Configuration specification

In general, this chapter defines configuration parameters and their clustering into containers. In order to support the specification Chapter 10.1 describes fundamentals. It also specifies a template (table) you shall use for the parameter specification. We intend to leave Chapter 10.1 in the specification to guarantee comprehension.

Chapter 10.2 specifies the structure (containers) and the parameters of the module WdgIf.

Chapter 10.3 specifies published information of the module WdgIf.

10.1 How to read this chapter

In addition to this section, it is highly recommended to read the documents:

- AUTOSAR Layered Software Architecture
- AUTOSAR ECU Configuration Specification
  This document describes the AUTOSAR configuration methodology and the AUTOSAR configuration metamodel in detail.

The following is only a short survey of the topic and it will not replace the ECU Configuration Specification document.

10.1.1 Configuration and configuration parameters

Configuration parameters define the variability of the generic part(s) of an implementation of a module. This means that only generic or configurable module implementation can be adapted to the environment (software/hardware) in use during system and/or ECU configuration.

The configuration of parameters can be achieved at different times during the software process: before compile time, before link time or after build time. In the following, the term “configuration class” (of a parameter) shall be used in order to refer to a specific configuration point in time.

10.1.2 Containers

Containers structure the set of configuration parameters. This means:

- *all* configuration parameters are kept in containers.
- (sub-) containers can reference (sub-) containers. It is possible to assign a multiplicity to these references. The multiplicity then defines the possible number of instances of the contained parameters.

10.1.3 Specification template for configuration parameters

The following tables consist of three sections:

- the general section
- the configuration parameter section
- the section of included/referenced containers

Pre-compile time - specifies whether the configuration parameter shall be of configuration class *Pre-compile time* or not

Label	***Description***
x	The configuration parameter shall be of configuration class *Pre-compile time*.
--	The configuration parameter shall never be of configuration class *Pre-compile time*.

Link time - specifies whether the configuration parameter shall be of configuration class *Link time* or not

Label	***Description***
x	The configuration parameter shall be of configuration class *Link time*.
--	The configuration parameter shall never be of configuration class *Link time*.

Post Build - specifies whether the configuration parameter shall be of configuration class *Post Build* or not

Label	***Description***
x	The configuration parameter shall be of configuration class *Post Build* and no specific implementation is required.
L	*Loadable* - the configuration parameter shall be of configuration class *Post Build* and only one configuration parameter set resides in the ECU.
M	*Multiple* - the configuration parameter shall be of configuration class *Post Build* and is selected out of a set of multiple parameters by passing a dedicated pointer to the init function of the module.
--	The configuration parameter shall never be of configuration class *Post Build*.

10.2 Containers and configuration parameters

The following chapters summarize all configuration parameters. The detailed meanings of the parameters describe Chapters 7 and 8.

10.2.1 Variants

There are no variants specified for this module.

10.2.2 WdgIf

Module Name	**WdgIf**
Module Description	Configuration of the WdgIf (Watchdog Interface) module.

Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
WdgIfDevice	1..*	--
WdgIfGeneral	1	This container collects all generic watchdog interface parameters.

10.2.3 WdgIfGeneral

SWS Item	**--**
Container Name	WdgIfGeneral{WdgIf_ModuleConfiguration}
Description	This container collects all generic watchdog interface parameters.
Configuration Parameters	

SWS Item	**--**		
Name	WdgIfDevErrorDetect {WDGIF_DEV_ERROR_DETECT}		
Description	Pre-processor switch for enabling the development error detection and reporting. true: Development error detection enabled false: Development error detection disabled		
Multiplicity	1		
Type	BooleanParamDef		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: Module		

SWS Item	**--**	
Name	WdgIfNumberOfDevices {WDGIF_NUMBER_OF_DEVICES}	
Description	Constant specifying the number of controlled watchdog drivers. Minimum number of watchdog drivers shall be one, maximum number limited by type of device index parameter.	
Multiplicity	1	
Type	IntegerParamDef	
Range	1 .. 255	
Default value	--	

ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: Module		

SWS Item	**--**		
Name	WdgIfVersionInfoApi {WDGIF_VERSION_INFO_API}		
Description	Pre-processor switch to enable / disable the service returning the version information. true: Version information service enabled false: Version information service disabled		
Multiplicity	1		
Type	BooleanParamDef		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: Module		

No Included Containers

10.2.4 WdgIfDevice

SWS Item	**--**
Container Name	WdgIfDevice
Description	--
Configuration Parameters	

SWS Item	**--**		
Name	WdgIfDeviceIndex		
Description	Represents the watchdog interface ID so that it can be referenced by the watchdog manager.		
Multiplicity	1		
Type	IntegerParamDef (Symbolic Name generated for this parameter)		
Default value	--		
ConfigurationClass	***Pre-compile time***	--	
	Link time	--	
	Post-build time	--	
Scope / Dependency			

SWS Item	**--**		
Name	WdgRef		
Description	Reference to the watchdog drivers that are controlled by the watchdog interface.		
Multiplicity	1		
Type	Reference to WdgGeneral		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency			

No Included Containers

10.3 Published Information

Published information contains data defined by the implementer of the SW module that does not change when the module is adapted (i.e. configured) to the actual HW/SW environment. It thus contains version and manufacturer information.

The standard common published information like

vendorId (<Module>_VENDOR_ID),
moduleId (<Module>_MODULE_ID),
arMajorVersion (<Module>_AR_MAJOR_VERSION),
arMinorVersion (<Module>_ AR_MINOR_VERSION),
arPatchVersion (<Module>_ AR_PATCH_VERSION),
swMajorVersion (<Module>_SW_MAJOR_VERSION),
swMinorVersion (<Module>_ SW_MINOR_VERSION),
swPatchVersion (<Module>_ SW_PATCH_VERSION),
vendorApiInfix (<Module>_VENDOR_API_INFIX)

is provided in the BSW Module Description Template (see [7] Figure 4.1 and Figure 7.1).

Additional published parameters are listed below if applicable for this module.

11 Changes to Release 1

11.1 Deleted SWS Items

SWS Item	***Rationale***
WDGIF022	New SWS template (didn't fit with the new structure)
WDGIF023	New SWS template (didn't fit with the new structure)
WDGIF027	New SWS template (didn't fit with the new structure)

11.2 Replaced SWS Items

SWS Item of Release 1	***replaced by SWS Item***	***Rationale***
WDGIF021	WDGIF028	New SWS template (copy-paste did work on the paragraph but not on the SWS tag).

11.3 Changed SWS Items

SWS Item	***Rationale***
WDGIF013	Bugzilla entry #4639
WDGIF007	New SWS template

11.4 Added SWS Items

SWS Item	***Rationale***
WDGIF028	Replacement for WDGIF021 (copy-paste did work on the paragraph but not on the SWS tag)
WDGIF029	New SWS template
WDGIF030	New SWS template
WDGIF031	New SWS template
WDGIF032	New SWS template
WDGIF033	New SWS template
WDGIF034	New SWS template
WDGIF035	New SWS template
WDGIF036	New SWS template
WDGIF037	New SWS template

12 Changes during SWS Improvements by Technical Office

12.1 Deleted SWS Items

None

12.2 Replaced SWS Items

None

12.3 Changed SWS Items

None

12.4 Added SWS Items

SWS Item	*Rationale*
WDGIF041	UML model linking of the imported types
WDGIF042	UML model linking of the function WdgIf_SetMode
WDGIF043	Extracted during the UML model linking of the function WdgIf_SetMode
WDGIF044	UML model linking of the function WdgIf_Trigger
WDGIF045	Extracted during the UML model linking of the function WdgIf_Trigger
WDGIF046	UML model linking of the function WdgIf_GetVersionInfo
WDGIF047	UML model linking of the mandatory interfaces
WDGIF048	UML model linking of the optional interfaces