

# MCAL User Manual for Wdg\_17\_Scu

## 32-bit TriCore™ AURIX™ TC3xx microcontroller

### About this document

#### Scope and purpose

This User Manual is intended to enable users to integrate the Microcontroller Abstraction Layer (MCAL) software for the TriCore™ AURIX™ family of 32-bit microcontrollers.

This document describes responsibilities of integrator in-charge of integrating MCAL software with the basic software (BSW) stack. This document also provides detailed information on safety, configuration and functions along with examples of usage of significant features.

*Note:* Detailed information about package installation, safety and other generic information that are common across all modules are provided in MCAL User Manual General.

#### Intended audience

This document is intended for anyone using the Wdg\_17\_Scu module of the TC3xx MCAL software.

#### Document conventions

**Table 1** Conventions

Convention	Explanation
<b>Bold</b>	Emphasizes heading levels, column headings, table and figure captions, screen names, windows, dialog boxes, menus, sub-menus
<i>Italics</i>	Denotes variable(s) and reference(s)
<code>Courier</code>	Denotes APIs, functions, interrupt handlers, events, data types, error handlers, file/folder names, directories, command line inputs, code snippets
<code>New</code>	
<b>&gt;</b>	Indicates that a cascading sub-menu opens when you select a menu item
[cover parentID=<alpha numeric value>]	Used for traceability completeness. Reader should ignore these.

#### Reference documents

This User Manual should be read in conjunction with the following documents:

- AURIX™ TC3xx MCAL User Manual General
- Specification of WDG Driver, AUTOSAR\_SWS\_WDG\_Driver, AUTOSAR Release 4.2.2
- Specification of WDG Driver, AUTOSAR\_SWS\_WDG\_Driver, AUTOSAR Release 4.4.0

---

**Table of contents**
**Table of contents**

	<b>About this document</b> .....	1
	<b>Table of contents</b> .....	2
<b>1</b>	<b>Wdg driver</b> .....	5
1.1	User information .....	5
1.1.1	Description .....	5
1.1.2	Hardware-software mapping .....	5
1.1.2.1	GTM: dependent hardware peripheral .....	6
1.1.2.2	STM: dependent hardware peripheral .....	7
1.1.2.3	SMU: dependent hardware peripheral .....	7
1.1.2.4	SCU: primary hardware peripheral .....	8
1.1.3	File structure .....	9
1.1.3.1	C file structure .....	9
1.1.3.2	Code generator plugin files .....	10
1.1.4	Integration hints .....	11
1.1.4.1	Integration with AUTOSAR stack .....	11
1.1.4.2	Multicore and Resource Manager .....	16
1.1.4.3	MCU support .....	17
1.1.4.4	Port support .....	19
1.1.4.5	DMA support .....	19
1.1.4.6	Interrupt connections .....	19
1.1.5	Key architectural considerations .....	22
1.1.5.1	WDG driver states .....	22
1.1.5.2	WDG driver critical section .....	22
1.1.5.3	WDG driver timer configuration and services .....	22
1.1.5.4	WDG driver timer dependency .....	22
1.1.5.5	WDG driver multicore error reporting .....	22
1.1.5.6	WDG driver password access .....	22
1.1.5.7	WDG driver trigger .....	23
1.1.5.8	WDG driver code generator dependency .....	23
1.1.5.9	WDG driver variation point .....	23
1.1.5.10	STM timer plausibility check .....	23
1.2	Assumptions of Use (AoU) .....	24
1.3	Reference information .....	25
1.3.1	Configuration interfaces .....	25
1.3.1.1	Container: GtmTimerConfiguration .....	26
1.3.1.2	Container: WdgSettingsConfig .....	26
1.3.1.3	Container: CommonPublishedInformation .....	26
1.3.1.3.1	ArMajorVersion .....	26
1.3.1.3.2	ArMinorVersion .....	26

---

**Table of contents**

1.3.1.3.3	ArPatchVersion .....	27
1.3.1.3.4	ModuleId .....	27
1.3.1.3.5	Release .....	28
1.3.1.3.6	SwMajorVersion .....	28
1.3.1.3.7	SwMinorVersion .....	29
1.3.1.3.8	SwPatchVersion .....	29
1.3.1.3.9	VendorApiInfix .....	30
1.3.1.3.10	VendorId .....	30
1.3.1.4	Container: GtmTimerOutputModuleConfiguration .....	31
1.3.1.4.1	GtmTimerClockSelect .....	31
1.3.1.4.2	GtmTimerUsed .....	32
1.3.1.5	Container: Wdg .....	33
1.3.1.5.1	IMPLEMENTATION_CONFIG_VARIANT .....	33
1.3.1.6	Container: WdgDemEventParameterRefs .....	33
1.3.1.6.1	WDG_E_DISABLE_REJECTED .....	34
1.3.1.6.2	WDG_E_MODE_FAILED .....	34
1.3.1.7	Container: WdgExternalConfiguration .....	35
1.3.1.8	Container: WdgGeneral .....	35
1.3.1.8.1	WdgDevErrorDetect .....	35
1.3.1.8.2	WdgDisableAllowed .....	36
1.3.1.8.3	WdgEcucPartitionRef .....	36
1.3.1.8.4	WdgIndex .....	37
1.3.1.8.5	WdgInitApiMode .....	37
1.3.1.8.6	WdgInitCheckApi .....	38
1.3.1.8.7	WdgInitialTimeout .....	39
1.3.1.8.8	WdgMaxTimeout .....	39
1.3.1.8.9	WdgRunArea .....	40
1.3.1.8.10	WdgRuntimeApiMode .....	40
1.3.1.8.11	WdgSafetyEnable .....	41
1.3.1.8.12	WdgTriggerLocation .....	42
1.3.1.8.13	WdgTriggerTimerSelection .....	42
1.3.1.8.14	WdgVersionInfoApi .....	43
1.3.1.9	Container: WdgPublishedInformation .....	43
1.3.1.9.1	WdgTriggerMode .....	44
1.3.1.10	Container: WdgSettingsConfig .....	44
1.3.1.10.1	WdgCPUDisableAllowed .....	44
1.3.1.10.2	WdgCPUInitialPassowrd .....	45
1.3.1.10.3	WdgCPUInitialTimeout .....	45
1.3.1.10.4	WdgCPUMaxTimeout .....	46
1.3.1.10.5	WdgCoreId .....	47
1.3.1.10.6	WdgDefaultMode .....	47
1.3.1.10.7	WdgSystemClockRef .....	48

---

**Table of contents**

1.3.1.11	Container: WdgSettingsFast .....	49
1.3.1.11.1	WdgFastModeTimeoutValue .....	49
1.3.1.12	Container: WdgSettingsOff .....	49
1.3.1.13	Container: WdgSettingsSlow .....	50
1.3.1.13.1	WdgSlowModeTimeoutValue .....	50
1.3.1.14	Container: WdgTriggerTimerSetting .....	50
1.3.1.14.1	WdgFastRefreshTime .....	51
1.3.1.14.2	WdgSlowRefreshTime .....	51
1.3.2	Functions - Type definitions .....	52
1.3.2.1	Wdg_17_Scu_ConfigType .....	52
1.3.3	Functions - APIs .....	52
1.3.3.1	Wdg_17_Scu_Init .....	53
1.3.3.2	Wdg_17_Scu_InitCheck .....	54
1.3.3.3	Wdg_17_Scu_SetMode .....	55
1.3.3.4	Wdg_17_Scu_SetTriggerCondition .....	56
1.3.3.5	Wdg_17_Scu_GetVersionInfo .....	57
1.3.4	Notifications and Callbacks .....	58
1.3.4.1	Wdg_17_Scu_Isr .....	58
1.3.5	Scheduled functions .....	59
1.3.6	Interrupt service routines .....	59
1.3.7	Callout .....	59
1.3.8	Errors Handling .....	59
1.3.9	Deviations and limitations .....	61
1.3.9.1	Deviations .....	61
1.3.9.1.1	Software specification deviations .....	61
1.3.9.1.2	AMDC Violations .....	62
1.3.9.1.3	VSMD Violations .....	62
1.3.9.2	Limitations .....	65
	<b>Revision history</b> .....	66
	<b>Disclaimer</b> .....	67

---

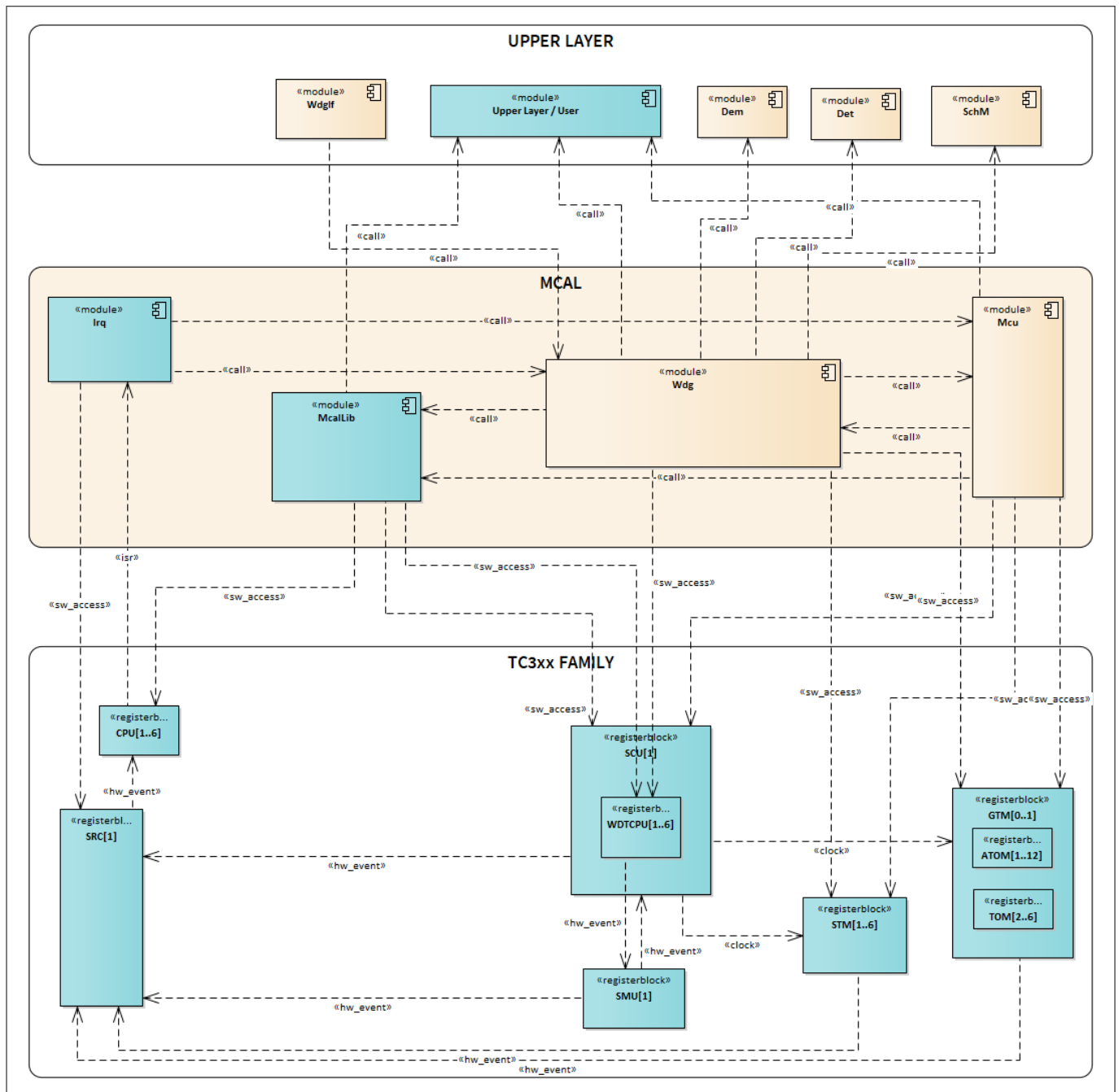
**1 Wdg driver****1 Wdg driver****1.1 User information****1.1.1 Description**

In AUTOSAR, the watchdog stack implements three monitoring mechanisms: Alive supervision (for supervision of timing of periodic software), Deadline supervision (for aperiodic software), Logical supervision (for supervision of the correctness of the execution sequence). The Watchdog(WDG) driver implements watchdog SWS as specified by AUTOSAR, which provides services for initialization, changing the operation mode and setting the trigger condition (timeout), for use by the user of watchdog hardware (WDT). The users of the WDG driver are: Mode management modules (for initialization), WdgM through WdgIf for setting trigger condition and driver operation Mode setting. The WdgM implements the above mentioned supervision mechanisms. AUTOSAR extends triggering concept to support windowed watchdog mode. This calls for use of hardware timers, to self-trigger WDT until the user-set trigger condition is met. For this purpose, the WDG driver provides options to select one of the two timers, either STM or GTM timers. Since code base would be common for all CPUs, to avoid presence of unused code, the timer selection is made static (precompile) for all WDTs (the driver uses either GTM or STM for all WDTs).

**1.1.2 Hardware-software mapping**

This section describes the system view of the WDG driver and peripherals administered by it.

### 1 Wdg driver



**Figure 1** Mapping of hardware-software interfaces

#### 1.1.2.1 GTM: dependent hardware peripheral

##### Hardware functional features

The WDG driver depends on the GTM for compare match event, which is required for servicing the WDT. Each WDT is mapped to a unique TOM/ATOM channel.

The GTM TOM/ATOM functional block feature configured/accessed by the WDG driver is:

ATOM/TOM compare match interrupt events feature is used to achieve WDG functionality. Rest of the GTM features are not used.

##### Users of the hardware

---

## 1 Wdg driver

The TOM/ATOM channels are used by the WDG, PWM, GPT and ADC drivers. To avoid conflicts for common GTM registers, the access requests to the GTM channels is performed via the MCU driver. For GTM channel specific registers, WDG exclusively own it and use to restart the GTM counters.

### Hardware diagnostic features

Not applicable

### Hardware events

The WDG driver uses the following hardware event from the GTM:

- Compare match event is used from the GTM to service the WDT.

### 1.1.2.2 STM: dependent hardware peripheral

#### Hardware functional features

The WDG driver depends on the STM for the compare match event which is required for servicing the watchdog timer.

The STMxTIM0 (x = Core Id) register is used by watchdog timer for calculating mode-specific refresh time to service the watchdog timer. Each watchdog timer is mapped to a unique STM timer compare register.

#### Users of the hardware

The McalLib, MCU and WDG drivers are users of the STM. To avoid conflicts, the access to the STM registers takes place through APIs provided by the MCU driver.

#### Hardware diagnostic features

Not applicable.

#### Hardware events

The WDG driver uses the following hardware events from the STM for compare match:

- STMIR0 is used when CMP0 is configured for the watchdog
- STMIR1 is used when CMP1 is configured for the watchdog

### 1.1.2.3 SMU: dependent hardware peripheral

#### Hardware functional features

The following features of SMU can be used

- An overflow of the WDT can trigger an alarm request to the SMU
- SMU can be configured to reset the system in case of the WDT overflow

#### Users of the hardware

The SMU configuration is handled only by the SMU driver.

#### Hardware diagnostic features

The SMU alarms are not monitored by the WDG driver.

#### Hardware events

Not applicable.

---

## 1 Wdg driver

### 1.1.2.4 SCU: primary hardware peripheral

#### Hardware functional features

The TC3xx contains the following WDTs:

- One safety WDT
- One WDT per CPU

The WDG driver uses the SCU IP for the clock, ENDINIT and reset functionalities. The driver requires the fSPB clock signal for functioning.

The key SCU features used by the WDG driver are:

- The WDG driver uses the WDTs for servicing at defined intervals as per application requirements.
- 16-bit WDT counter is used to implement watchdog functionality (WDTCPuSR).
- 16-bit user-definable reload value is used for normal WDT operation (WDTCPuCON0). The reload value for normal mode shall be computed based on trigger condition at run time (A fixed reload value provided in the hardware for timeout mode shall only be used for updating CPU critical registers).
- Supports disabling of WDT(WDTCPuCON1).
- Selectable input frequency is fSPB/64, fSPB/256 or fSPB/16384 (WDTCPuCON1). Frequency fSPB/64 is not used (fast mode frequency is fSPB/256 and slow mode frequency is fSPB/16384).

The unsupported features of the WDT are:

- Safety WDT (Support is provided for CPU WDT only).
- Incorporation of the corresponding ENDINIT bit and monitoring of its modifications. The WDG ENDINIT feature will only be used for CPU critical registers and will not be supported by WDG driver.
- Automatic password sequencing mechanism is not supported. A fixed reload password will be used.
- The UR bit in WDTCPu0CON1 register will not be configurable through WDG driver. Updating this bit depends on the state of the SMU which is not monitored by the WDG driver.
- Time Check Password feature is not supported.

#### Users of the hardware

The SCU IP supplies clock for all the peripherals and the MCU driver is responsible for configuring the clock tree. To avoid conflicts due to simultaneous writes, updates to all the ENDINIT protected registers are performed using the MCALLIB APIs.

The WDG and MCALLIB drivers exclusively utilize the WDT.

The MCALLIB driver uses the WDT to implement the ENDINIT protection feature in the timeout mode.

To avoid conflicts, the WDG and MCALLIB drivers use the critical section with the same exclusive area name while accessing the shared resources. User shall ensure that core-specific interrupts are disabled in the critical sections since CON0 register is updated inside these critical sections by both the drivers. User shall ensure that the software on each core accesses only the core-specific WDT registers.

#### Hardware diagnostic features

Following are the hardware error status flags monitored in the WDG driver:

Overflow error detection.

Access error detection.

The SMU alarms configured for the WDG are not monitored by the WDG driver.



**1 Wdg driver**

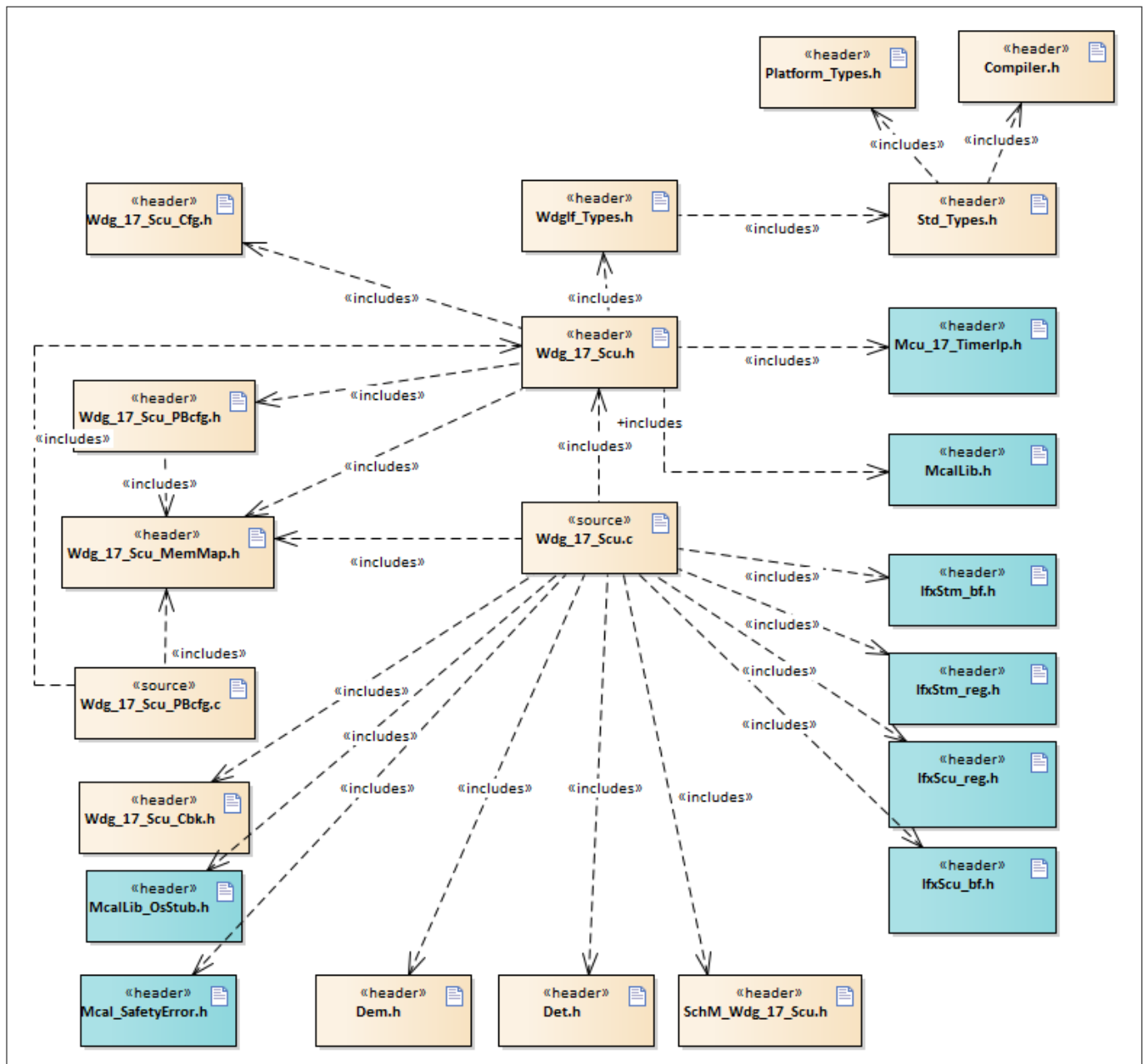
## Hardware events

Hardware events from the SCU are not used by the WDG driver.

### 1.1.3 File structure

### 1.1.3.1 C file structure

This section provides details of the C files of the WDG driver.



**Figure 2** **Wdg\_C\_File\_Structure-1.png**

## 1 Wdg driver

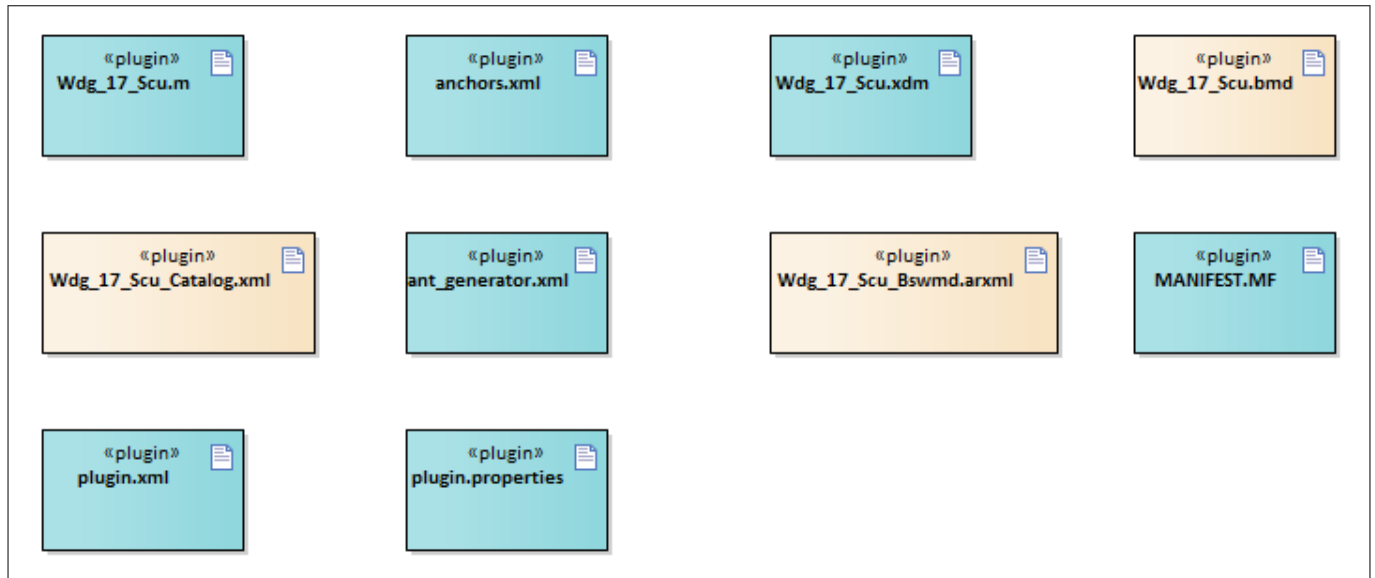
**Table 2 C file structure**

File name	Description
Compiler.h	Provides abstraction from compiler-specific keywords
Dem.h	Provides the exported interfaces of Diagnostic Event Manager
Det.h	Provides the exported interfaces of Development Error Tracer
IfxScu_bf.h	SFR header file for SCU
IfxScu_reg.h	SFR header file for SCU
IfxStm_bf.h	SFR header file for STM
IfxStm_reg.h	SFR header file for STM
McalLib.h	Static header file defining prototypes of data structure and APIs exported by the MCALLIB.
McalLib_OsStub.h	McalLib_OsStub.h provides macros to support user mode of Tricore. This shall be included by other drivers to call OS APIs.
Mcal_SafetyError.h	Header file containing the prototype of the API for reporting safety-related errors
Mcu_17_TimerIp.h	Header file defining prototypes of data structures and APIs of Timer IPs (GTM, CCU6 and GPT12), containing functions such as initialization, enable, interrupt handlers and other services and is included by Mcu_17_TimerIp.c source file
Platform_Types.h	Platform-specific type declaration file as defined by AUTOSAR
SchM_Wdg_17_Scu.h	Export header for SchM functions of WDG driver.
Std_Types.h	Standard type declaration file as defined by AUTOSAR. It is independent of compiler or platform.
WdgIf_Types.h	Provides type definitions of the Watchdog Interface
Wdg_17_Scu.c	Source file providing implementation of APIs.
Wdg_17_Scu.h	Header file providing prototypes of APIs, data types and interrupt handlers.
Wdg_17_Scu_Cbk.h	Header file contains call back function of the WDG driver.
Wdg_17_Scu_Cfg.h	Generated header file containing macros.
Wdg_17_Scu_MemMap.h	File (Static) containing the memory section definitions used by the WDG driver.
Wdg_17_Scu_PBcfg.c	Generated source file containing configuration data of the user.
Wdg_17_Scu_PBcfg.h	Generated header file containing configuration data of the user.

### 1.1.3.2 Code generator plugin files

This section provides details of the code generator plugin files of the WDG driver.

## 1 Wdg driver



**Figure 3** Wdg\_Code\_Generator\_Plugin\_Files-1.png

**Table 3** Code generator plugin files

File name	Description
MANIFEST.MF	Tresos plugin support file containing the metadata for the WDG driver.
Wdg_17_Scu.bmd	AUTOSAR format XML data model schema file (for each device).
Wdg_17_Scu.m	Code template macro file for WDG driver.
Wdg_17_Scu.xdm	Tresos format XML data model schema file.
Wdg_17_Scu_Bswmd.arxml	AUTOSAR format module description file.
Wdg_17_Scu_Catalog.xml	AUTOSAR format catalog file.
anchors.xml	Tresos anchors support file for the WDG driver.
ant_generator.xml	Tresos support file to generate and rename multiple post-build configuration when using variation point.
plugin.properties	Tresos plugin support file for the WDG driver.
plugin.xml	Tresos plugin support file for the WDG driver.

### 1.1.4 Integration hints

This section lists the key points that an integrator or user of the WDG driver must consider.

#### 1.1.4.1 Integration with AUTOSAR stack

This section lists the modules, which are not part of the MCAL, but are required to integrate the WDG driver.

- EcuM**

The ECU Manager module is a part of the AUTOSAR stack that manages common aspects of the ECU. Specifically, in the context of MCAL, EcuM is used for initialization and de-initialization of the software

---

## 1 Wdg driver

drivers. The EcuM module provided in the MCAL package is a stub code and needs to be replaced with a complete EcuM module during the integration phase.

- **Memory mapping**

Memory mapping is a concept from AUTOSAR that allows relocation of text, variables, constants and configuration data to user-specific memory regions. To achieve this, all the relocatable elements of the driver are encapsulated in different memory-section macros. These macros are defined in the `Wdg_17_Scu_MemMap.h` file.

The `Wdg_17_Scu_MemMap.h` file is provided in the MCAL package as a stub code. The integrator must place appropriate compiler pragmas within the memory-section macros. The pragmas ensure that the elements

## 1 Wdg driver

are re-located to the correct memory region. A sample implementation listing the memory-section macros is shown as follows:

```
#define MEMMAP_ERROR
/*To be used for all global or static variables.*/
/*Variable to be cleared at startup or reset is placed here - .bss*/
#if defined WDG_17_SCU_START_SEC_VAR_CLEARED_ASIL_B_GLOBAL_32
/*User pragmas here*/
#undef WDG_17_SCU_START_SEC_VAR_CLEARED_ASIL_B_GLOBAL_32
#undef MEMMAP_ERROR

#elif defined WDG_17_SCU_STOP_SEC_VAR_CLEARED_ASIL_B_GLOBAL_32
/*User pragmas here*/
#undef WDG_17_SCU_STOP_SEC_VAR_CLEARED_ASIL_B_GLOBAL_32
#undef MEMMAP_ERROR

/*Constants with attributes that show that they reside in one segment for
module
configuration.*/
#elif defined WDG_17_SCU_START_SEC_CONFIG_DATA_ASIL_B_CORE0_UNSPECIFIED
/*User pragmas here*/
#undef WDG_17_SCU_START_SEC_CONFIG_DATA_ASIL_B_CORE0_UNSPECIFIED
#undef MEMMAP_ERROR

#elif defined WDG_17_SCU_STOP_SEC_CONFIG_DATA_ASIL_B_CORE0_UNSPECIFIED
/*User pragmas here*/
#undef WDG_17_SCU_STOP_SEC_CONFIG_DATA_ASIL_B_CORE0_UNSPECIFIED
#undef MEMMAP_ERROR

/* Code Sections */
#elif defined WDG_17_SCU_START_SEC_CODE_ASIL_B_GLOBAL
/*User pragmas here*/
#undef WDG_17_SCU_START_SEC_CODE_ASIL_B_GLOBAL
#undef MEMMAP_ERROR

#elif defined WDG_17_SCU_STOP_SEC_CODE_ASIL_B_GLOBAL
/*User pragmas here*/
#undef WDG_17_SCU_STOP_SEC_CODE_ASIL_B_GLOBAL
#undef MEMMAP_ERROR

#endif
#if defined MEMMAP_ERROR
#error "Wdg_17_Scu_MemMap.h, wrong pragma command"
#endif
```

### • DET

The DET module is a part of the AUTOSAR stack that handles all the development and runtime errors reported by the BSW modules. The WDG driver reports all the development errors to the DET module through the `Det_ReportError()` API. The user of the WDG driver must process all the errors reported to the DET module through the `Det_ReportError()` API.

---

## 1 Wdg driver

The `Det.h` and `Det.c` files are provided in the MCAL package as a stub code and needs to be replaced with a complete DET module during the integration phase.

- **DEM**

The DEM module is a part of the AUTOSAR stack that handles all the production errors reported by the BSW modules.

The WDG driver reports all the production errors through the interfaces provided by the DEM module. The user of the WDG driver shall process all the production errors (fail/pass) reported to the DEM module. The interface used for reporting in AUTOSAR version 4.2.2 is `Dem_ReportErrorStatus()` and for AUTOSAR version 4.4.0 is `Dem_SetEventStatus()`. The `Dem.h` and `Dem.c` files are provided in the MCAL package as a stub code and needs to be replaced with a complete DEM module during the integration phase.

- **SchM**

The SchM module is a part of the RTE that manages the BSW Scheduler. The WDG driver uses the exclusive areas defined in `SchM_Wdg_17_Scu.h` file to protect the SFRs and variables from concurrent accesses from different threads. The SchMs identified for the WDG driver are:

- TimerHandling
- ChangeMode
- CpuEndInit

The `SchM_Wdg_17_Scu.h` and `SchM_Wdg_17_Scu.c` files are provided in the MCAL package as an example code and needs to be updated by the integrator. The user must implement the SchM functions

## 1 Wdg driver

defined by the WDG driver as **suspend / resume** of interrupts for the CPU on which the API is invoked. A sample implementation of the SchM functions is shown as follows:

```
/*Sample implementation of SchM_Wdg_17_Scu.c*/

#include "IFX_Os.h"
#include "SchM_Wdg_17_Scu.h"

void SchM_Enter_Wdg_17_Scu_TimerHandling(void)
{
    SuspendAllInterrupts(); /* Suspend CPU core interrupt */
}

void SchM_Exit_Wdg_17_Scu_TimerHandling(void)
{
    ResumeAllInterrupts(); /* Resume CPU core interrupt */
}

void SchM_Enter_Wdg_17_Scu_ChangeMode(void)
{
    SuspendAllInterrupts(); /* Suspend CPU core interrupt */
}

void SchM_Exit_Wdg_17_Scu_ChangeMode(void)
{
    ResumeAllInterrupts(); /* Resume CPU core interrupt */
}

void SchM_Enter_Wdg_17_Scu_CpuEndInit(void)
{
    SuspendAllInterrupts(); /* Suspend CPU core interrupt */
}

void SchM_Exit_Wdg_17_Scu_CpuEndInit(void)
{
    ResumeAllInterrupts(); /* Resume CPU core interrupt */
}
```

- **Safety error**

The WDG driver will report all the detected safety errors through the `Mcal_ReportSafetyError()` API. The driver performs only detection and reporting of the safety errors. The handling of the reported errors shall be done by the user. The `Mcal_ReportSafetyError()` is provided in the `Mcal_SafetyError.c` and `Mcal_SafetyError.h` files as a stub code, and must be updated by the integrator to handle the reported errors.

*Note: All DET errors are also reported as safety errors (error code used is same as DET).*

- **Notifications and call backs**

The WDG driver does not expect any call backs from application but it needs a call back from MCU driver for ISR handling.

- **Operating system (OS)**

---

## 1 Wdg driver

The OS or the application must ensure correct type of service and interrupt priority is configured in the SR register. Enabling and disabling of interrupts must also be managed by the OS or the application.

The OS files provided by the MCAL package are only an example code and must be updated by the integrator with the actual OS files for the desired function.

### 1.1.4.2 Multicore and Resource Manager

#### Multicore concept for WDG driver

Every CPU has one dedicated WDT. The global status variable is used to maintain the core-specific reload value, status (IDLE, BUSY, UNINIT), timeout counter value and mode (FAST, SLOW, OFF) of the WDT. Initialization is core specific (CPUx can update/modify registers and variables for WDTx only. x = 0,1,2,3,4,5). At least one CPU WDT must be configured if the WDG driver is used in an application. The key points to be considered with respect to multicore in the driver are as follows:

- Locating the constants, variables and configuration data to correct memory space should be carried out by the user. Memory sections are marked GLOBAL (common to all cores) and CORE[x] (specific to a CPU core). The following should be considered by the user to ensure better performance of the driver:

#### Code section:

The executable code of the WDG driver is placed under single MemMap section. It can be relocated to any PFlash region.

#### Data section:

The RAM variable memory sections marked as specific to a core should be relocated to the DSPR/DLMU of the same core. The sections marked as global should be relocated to the non-cached LMU region.

#### Configuration data and constants:

The configuration data sections marked as specific to a core should be relocated to the PFlash of the same core. The sections marked as global should be relocated to the PFlash of the master core.

**Note:** Relocating code, data or constants to a distant memory region would impact execution timings.

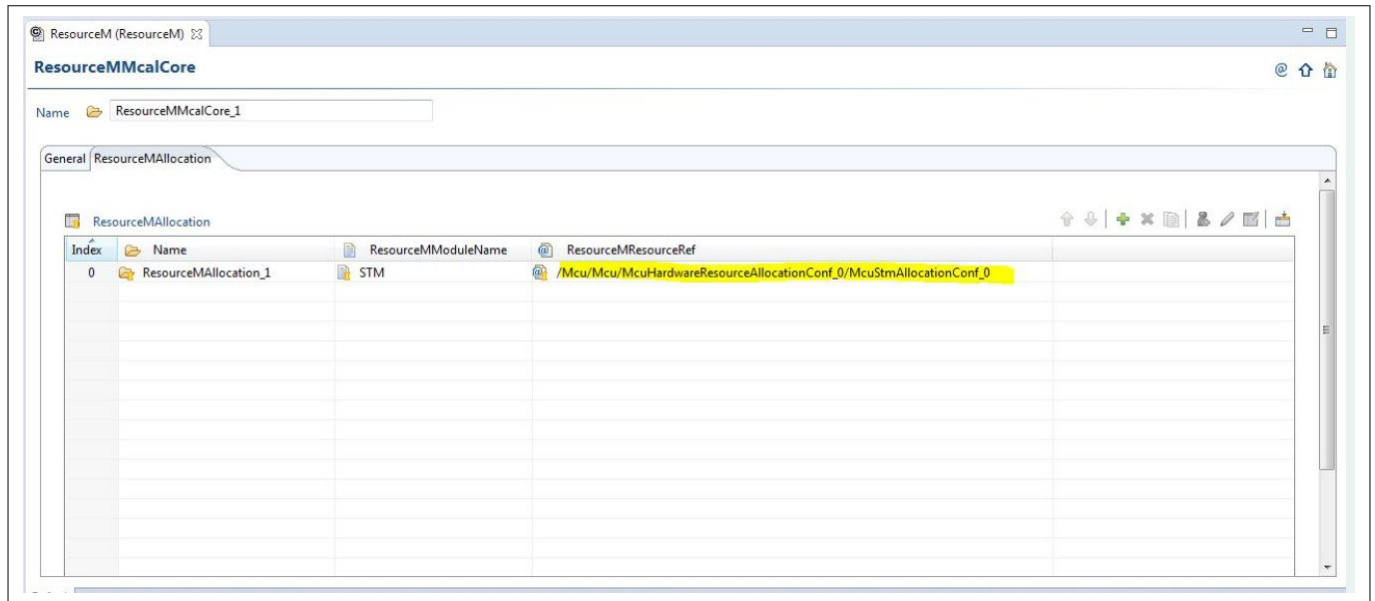
**Note:** If the driver operates from a single (master) core, all the sections may be relocated to the PFlash/DSPR/DLMU of the same CPU core.

#### Resource manager for WDG

The user should allocate resources of WDG to CPU cores at pre-compile time using the Resource manager module. WDG driver depends on resource manager if STM timer is configured for its timing needs. An STM timer should be allocated to the same core in the resource manager for which a CPU WDG is configured.



### 1 Wdg driver

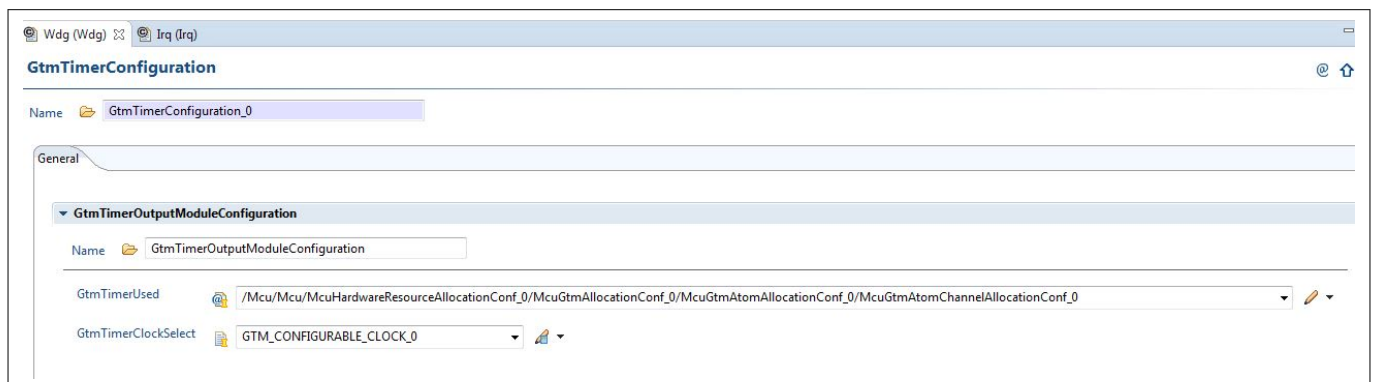


**Figure 4** STM timer allocation in Resource Manager

#### 1.1.4.3 MCU support

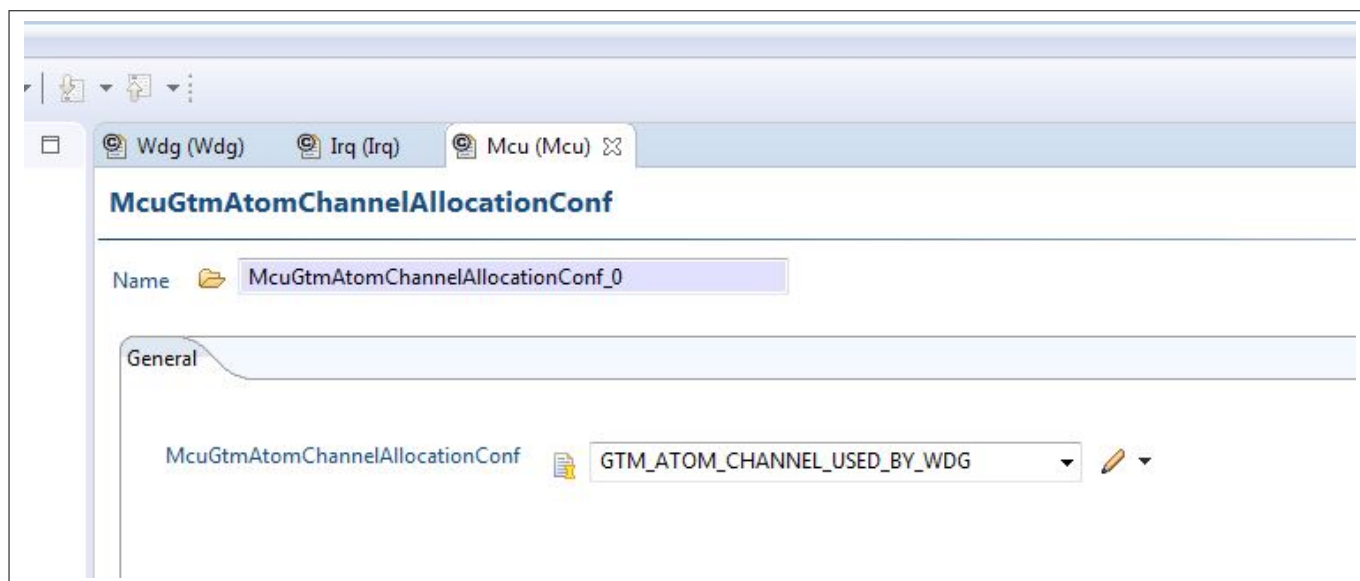
##### Timer support

The WDG driver is dependent on the MCU driver for clock configuration and GTM and STM timer services. The initialization of the WDG driver must be started only after completion of the MCU initialization.

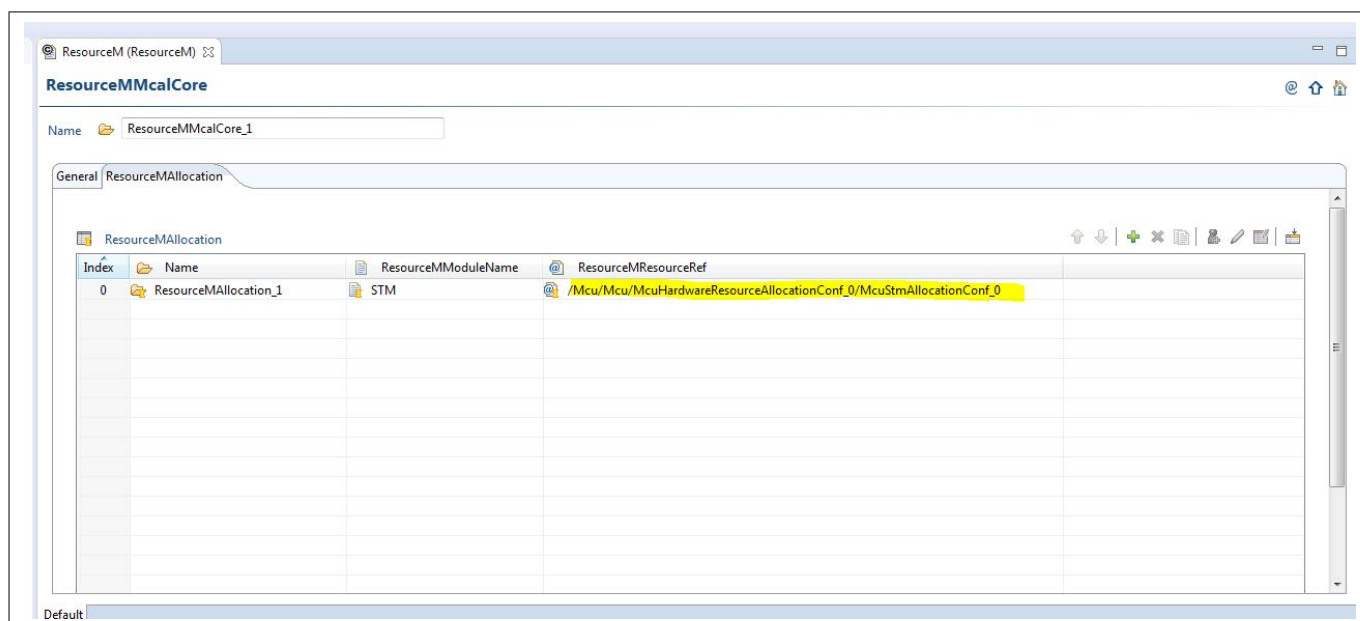


**Figure 5** WDG configuration with a reference to GTM TOM/ATOM channel in MCU

## 1 Wdg driver

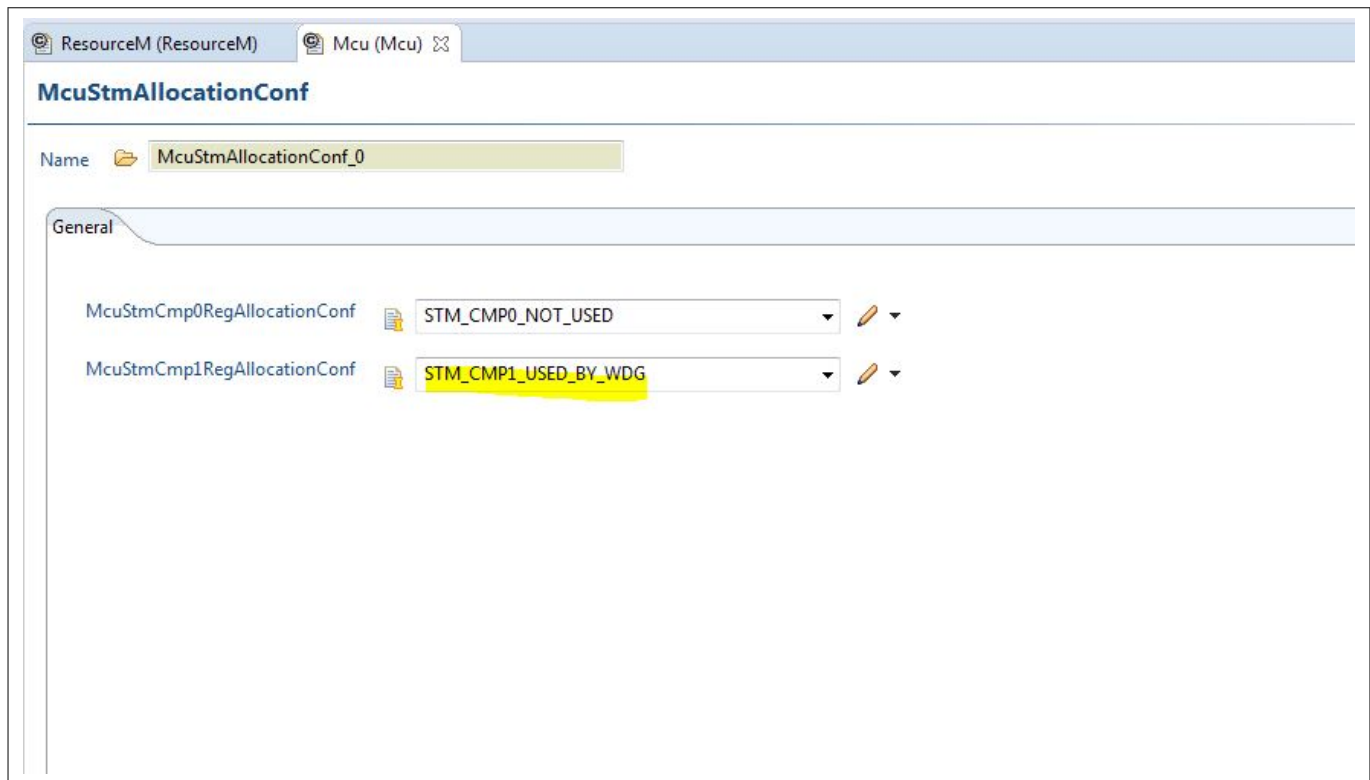


**Figure 6** MCU configuration for GTM TOM/ATOM channel



**Figure 7** STM timer reference in Resource Manager to MCU

## 1 Wdg driver



**Figure 8** STM Compare Register allocation for WDG in MCU

### 1.1.4.4 Port support

The WDG driver does not use any services provided by the PORT driver.

### 1.1.4.5 DMA support

The WDG driver does not use any services provided by the DMA driver.

### 1.1.4.6 Interrupt connections

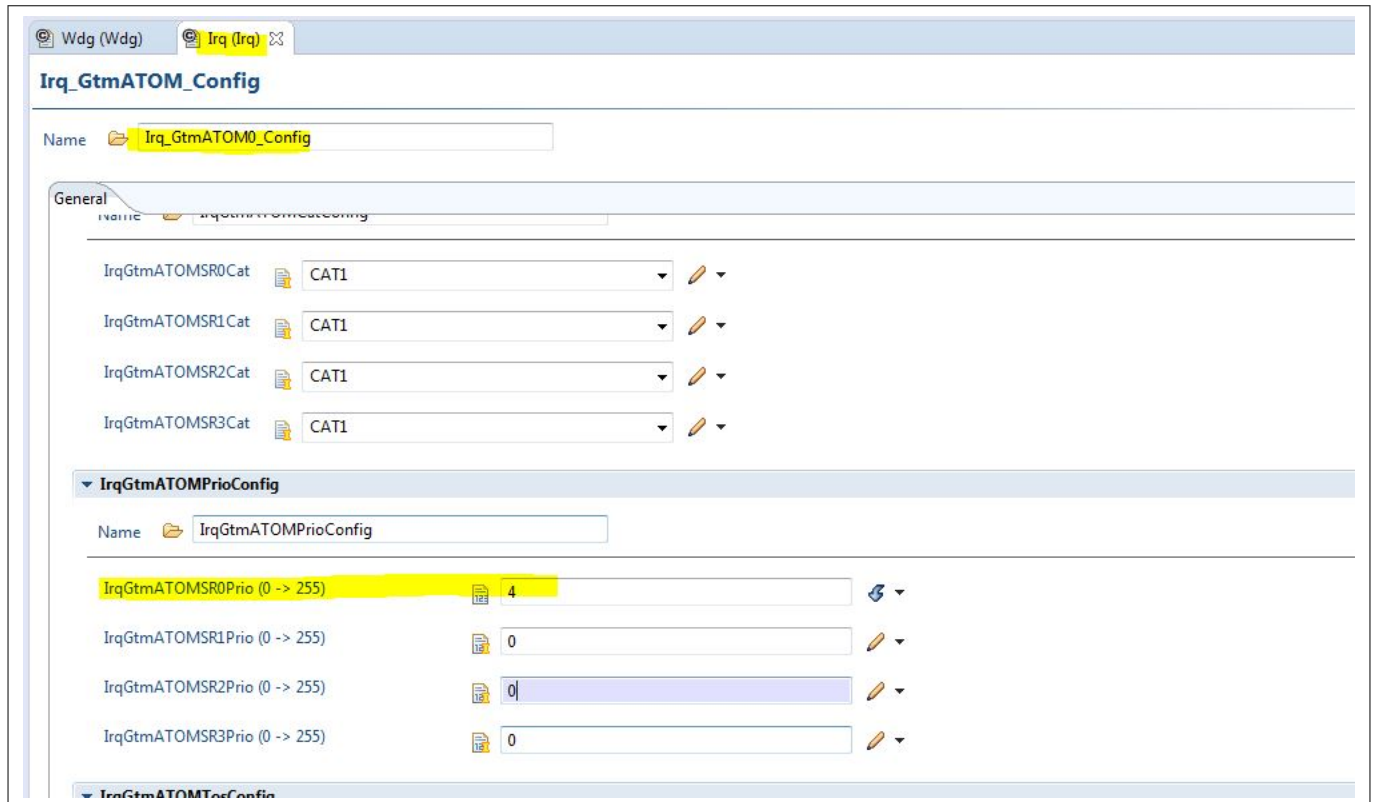
The interrupt connections of the Wdg driver are described in this section.

#### GTM timer

If GTM timer is used, the priority number for the interrupt must be configured in the IRQ module based on the TOM or ATOM channel configured.

For example, if ATOM 0 and Channel 0 is used for WDT, the interrupts have to be configured as shown in the following image:

## 1 Wdg driver



**Figure 9** IRQ module configured for interrupt from ATOM 0 Channel 0 with priority number 4

Invoking the interrupt handlers provided by the driver must be done by the user. A sample invocation is shown as follows:

```
/* include MCU timer header file */
#include "Mcu_17_TimerIp.h"
/*****SRC_ GTMTOM0SR0*****/
ISR(GTMTOM0SR0_ISR)
{
    /* Enable Global Interrupts */
    ENABLE();
    /* Parameter is Channel Number */
    Mcu_17_Gtm_TomChannelIsr(0, 0);
}

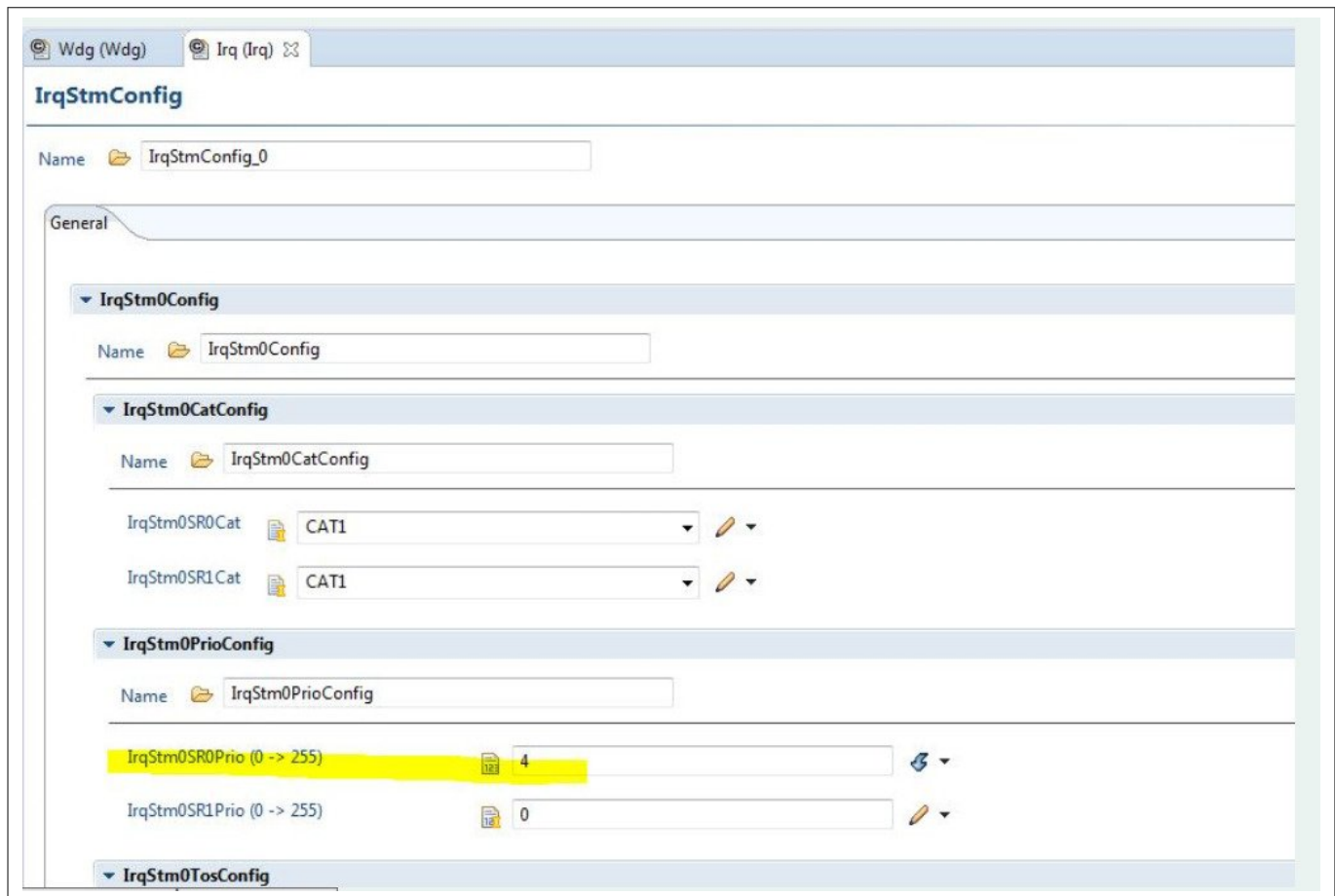
/*****SRC_ GTMATOM0SR0*****/
ISR(GTMATOM0SR0_ISR)
{
    /* Enable Global Interrupts */
    ENABLE();
    /* Parameter is Channel Number */
    Mcu_17_Gtm_AtomChannelIsr(0, 0);
}
```

### STM timer

If the STM timer is used, the priority number for the interrupt must be configured in the IRQ driver.

For example, if STM used for WDT, the interrupts have to be configured as shown in the following image:

## 1 Wdg driver



**Figure 10** IRQ module configured for interrupt from to STM0\_CMP0 with priority number 4

Invoking the interrupt handlers provided by the driver must be done by the user. A sample invocation is shown as follows:

```
/* include MCU timer header file */
#include "Mcu_17_TimerIp.h"
/*****SRC_ STM0SR0*****/
ISR(STM0SR0_ISR)
{
    /* Enable Global Interrupts */
    ENABLE();
    /* Call Interrupt function */
    Mcu_17_Stm_CompareMatchIsr(0U, 0U);
}
/*****SRC_ STM1SR0*****/
ISR(STM1SR0_ISR)
{
    /* Enable Global Interrupts */
    ENABLE();
    /* Call Interrupt function */
    Mcu_17_Stm_CompareMatchIsr(1U, 0U);
}
```

---

**1 Wdg driver****1.1.5 Key architectural considerations****1.1.5.1 WDG driver states**

The state of the WDG driver will be updated to WDG\_UNINIT or WDG\_BUSY or WDG\_IDLE irrespective of whether the DET or safety error check is enabled or not. The driver state check in the APIs will prevent any unintended SFR or global update.

**1.1.5.2 WDG driver critical section**

The SCU\_WDTCPUxCON0 (x is core Id) register is accessed by both MCALLIB and WDG (MCALLIB driver ensures that the REL bits of this register is not hampered and it writes back with the correct values once it exits the timeout mode). MCALLIB uses this register in the hardware timeout mode and WDG uses it in the hardware normal mode. Thus MCALLIB and WDG use the critical section with the same exclusive area (CpuEndinit).

**1.1.5.3 WDG driver timer configuration and services**

Every WDT configured for a core must be accompanied with an STM timer/GTM timer resource. All STM and GTM related registers will be configured by invoking MCU driver APIs.

If STM timer is used, the timer is allocated to the core using the Resource manager and the compare register is allocated using the MCU driver. The STM compare register used for WDT needs must be configured in the MCU driver. STM compare match interrupt from CMP0 will be routed through IR0 and CMP1 will be routed through IR1.

If GTM timer is used, the TOM/ATOM channel is allocated in the MCU driver and the respective clock is configured in the WDG driver. All the STM/GTM related registers required by the WDG driver are updated using services provided by the MCU driver. The source of the interrupt will be CMP0 (compare match interrupt) and not CMP1 (period match interrupt). The WDG driver mode specific reload values (SLOW/FAST) provided in the configuration is the timeout in seconds used by the WDG driver and the mode specific refresh (SLOW/FAST) values are the GTM/STM timer callback periods.

**1.1.5.4 WDG driver timer dependency**

Among the available timers, the GTM or STM timer is chosen to trigger the WDT. The GTM and STM have sufficient number of timers to support WDT on multiple cores. The GPT12 and CC6 timers are not considered due to the unavailability of the required number of timers and due to specific use cases of these timers. In an application, it could be possible that one STM timer is allocated per CPU. The same timer can be used by CPU WDT and is also accessed by the OS. Since the STM timer hardware provides two timer interrupts (CMP0, CMP1), one interrupt could be used by OS and the other by WDT.

**1.1.5.5 WDG driver multicore error reporting**

When the WDG initialization is done for a wrong core, the WDG\_E\_PARAM\_CONFIG development or safety error is reported.

**1.1.5.6 WDG driver password access**

The internal WDG hardware shall be accessed in the hardware normal mode by the WDG driver with a static password, which is configured by the user. The entire password access sequence is handled by the WDG driver. The WDG hardware supports both static password and automatic password sequence mechanism. However, according to the AUTOSAR 4.2.2, the program flow monitoring shall be fulfilled by the upper layer (WDG manager) where automatic password sequencing mechanism could be used. Even though the password is configured by the user at runtime, the password to unlock the watchdog timer may be changed by the user

---

## 1 Wdg driver

or by another driver. Hence, the password is always recalculated before SCU\_WDTCPUxCON0 (x = core Id) is accessed.

### 1.1.5.7 WDG driver trigger

The timeout passed to the `Wdg_SetTriggerCondition` API must be greater than the mode (SLOW/FAST) specific refresh time (configured by the user), to ensure that the WDT is serviced until the trigger counter value becomes zero. (If the mode is OFF, the WDT will not be serviced). If the timeout is less than the refresh time then the trigger counter will be zero and WDT will not be serviced and this will lead to WDG alarm when the timer overflows.

### 1.1.5.8 WDG driver code generator dependency

The WDG driver is dependent on MCU, MCALLIB and Resource Manager for successful code generation. In case these modules are not added to the configuration project, an error is raised.

### 1.1.5.9 WDG driver variation point

The Post-Build-Variant value of the parameters: `WdgInitialTimeout`, `WdgCPUInitialTimeout`, `WdgMaxTimeout` and `WdgCPUMaxTimeout` is set as TRUE since the values may vary across different variants configured across multiple cores.

### 1.1.5.10 STM timer plausibility check

If the WDG driver uses the STM timer for time-based triggers, the STM timer will generate interrupts to service the WDT as per the configured mode-specific refresh time (`WdgFastRefreshTime` and `WdgSlowRefreshTime`). The user shall check the plausibility of the STM timer and only if the plausibility check is successful, the `Wdg_17_Scu_Isr` API shall be invoked.

## 1 Wdg driver

### 1.2 Assumptions of Use (AoU)

The AoU for the WDG driver are as follows.

The safety measures which need to be fulfilled by the integrator will be provided as Assumptions of Use (AoU) in order to enable the usage of these drivers in safety relevant applications up to ASIL B and to facilitate flexibility in customer's safety concepts.

- **Critical section used while accessing CON0 register**

User shall ensure that the core-specific interrupts are disabled in the following critical sections:

-SchM\_Enter\_Wdg\_17\_Scu\_CpuEndInit and -SchM\_Enter\_McalLib\_CpuEndInit.

[cover parentID WDG={51D8C43C-2314-4b4d-BF4C-263237F16EAB}]

- **GTM channel allocation to WDG timer**

The user shall ensure the following while allocating GTM channels to a particular core-specific WDG timer.

The GTM channels sharing the same interrupt node shall be allocated to the same core in the IRQ driver. For example, if GTM channel x is allocated to the WDG driver and channel x+1 is allocated to the PWM driver, both the modules shall be in the same core since x and x+1 share interrupt node.

[cover parentID WDG={B578A86A-6687-4294-B6A7-A0B824FCE690}]

- **GTM channels sharing same interrupt nodes**

User shall ensure that timer channels of the GTM, which share the same interrupt nodes, shall not be configured for different CPU WDG timers.

[cover parentID WDG={49FDBDF3-10F6-4ba2-8F8E-06D92DC37EA4}]

- **WDG driver initialization with the correct configuration pointer**

The user shall ensure that the WDG initialization is done with the correct configuration pointer. The Wdg\_17\_Scu\_InitCheck() API shall be called after the WDG driver initialization and before starting any WDG driver functionality.

[cover parentID WDG={46AC4880-0030-4f96-85AA-465741024800}]

- **WDG overflow**

The user shall ensure that the watchdog overflow shall trigger an SMU alarm and enter into a safe state.

However, the overflow error flag will be checked before servicing the WDT and a safety error will be reported to the upper layer.

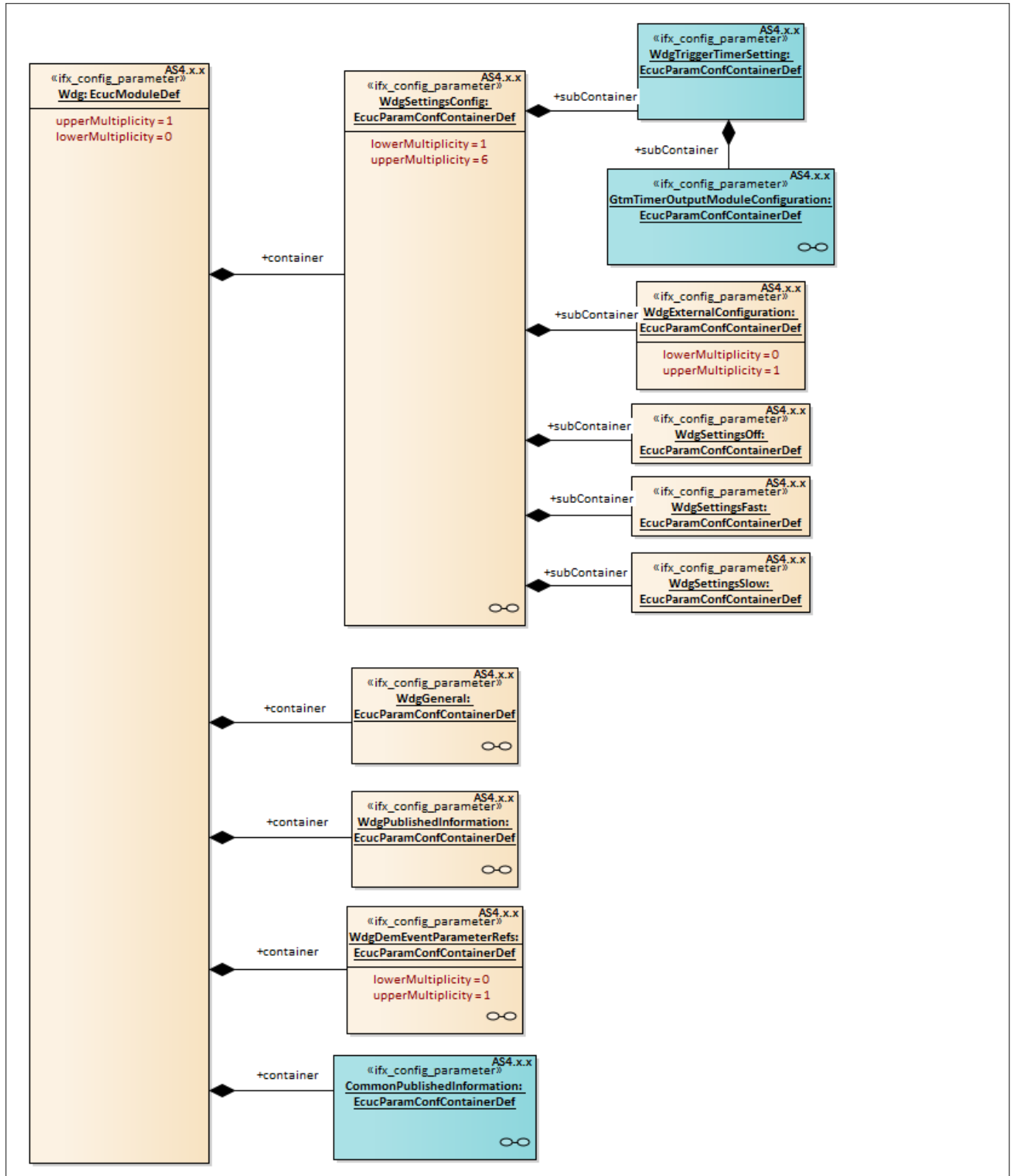
[cover parentID WDG={CBD32342-5082-4356-9803-61385D57337E}]



## 1 Wdg driver

### 1.3 Reference information

#### 1.3.1 Configuration interfaces



**Figure 11** Container hierarchy along with their configuration parameters

## 1 Wdg driver

### 1.3.1.1 Container: GtmTimerConfiguration

This container contains the configuration elements for configuring GTM timer hardware.

Post-Build Variant Multiplicity: TRUE

Multiplicity Configuration Class: Pre-Compile

### 1.3.1.2 Container: WdgSettingsConfig

This is a List of containers of Configuration items for the different watchdog settings.

*Note: All postbuild parameters are handled via this container.*

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

### 1.3.1.3 Container: CommonPublishedInformation

Container for common published information. (Based on BSW General)

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

#### 1.3.1.3.1 ArMajorVersion

**Table 4** Specification for ArMajorVersion

<b>Name</b>	ArMajorVersion		
<b>Description</b>	This parameter provides the major version of the AUTOSAR Specification.		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucIntegerParamDef
<b>Range</b>	0 - 255		
<b>Default value</b>	4		
<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Published-Information	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	IFX	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

#### 1.3.1.3.2 ArMinorVersion

**Table 5** Specification for ArMinorVersion

<b>Name</b>	ArMinorVersion		
<b>Description</b>	This parameter provides the minor version of the AUTOSAR Specification.		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucIntegerParamDef

## 1 Wdg driver

**Table 5 Specification for ArMinorVersion (continued)**

<b>Range</b>	0 - 255		
<b>Default value</b>	As per the selected Autosar version		
<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Published-Information	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	IFX	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.3.3 ArPatchVersion

**Table 6 Specification for ArPatchVersion**

<b>Name</b>	ArPatchVersion		
<b>Description</b>	This parameter provides the patch version of the AUTOSAR Specification.		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucIntegerParamDef
<b>Range</b>	0 - 255		
<b>Default value</b>	As per the selected Autosar version		
<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Published-Information	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	IFX	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.3.4 ModuleId

**Table 7 Specification for ModuleId**

<b>Name</b>	ModuleId		
<b>Description</b>	This parameter provides the Module Id.		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucIntegerParamDef
<b>Range</b>	0 - 65535		
<b>Default value</b>	102		
<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	-

## 1 Wdg driver

**Table 7 Specification for ModuleId (continued)**

<b>Value configuration class</b>	Published-Information	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	IFX	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.3.5 Release

**Table 8 Specification for Release**

<b>Name</b>	Release		
<b>Description</b>	Aurix2G derivative used for the implementation Default value will depend on the derivative chosen and will be read from the derivative .properties file.		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucStringParamDef
<b>Range</b>	String		
<b>Default value</b>	Depends on the Derivative		
<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Published-Information	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	IFX	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.3.6 SwMajorVersion

**Table 9 Specification for SwMajorVersion**

<b>Name</b>	SwMajorVersion		
<b>Description</b>	This parameter provides the major version of the Software.		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucIntegerParamDef
<b>Range</b>	0 - 255		
<b>Default value</b>	As per the driver version		
<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	-

## 1 Wdg driver

**Table 9 Specification for SwMajorVersion (continued)**

<b>Value configuration class</b>	Published-Information	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	IFX	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.3.7 SwMinorVersion

**Table 10 Specification for SwMinorVersion**

<b>Name</b>	SwMinorVersion		
<b>Description</b>	This parameter provides the minor version of the Software.		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucIntegerParamDef
<b>Range</b>	0 - 255		
<b>Default value</b>	As per the driver version		
<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Published-Information	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	IFX	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.3.8 SwPatchVersion

**Table 11 Specification for SwPatchVersion**

<b>Name</b>	SwPatchVersion		
<b>Description</b>	This parameter provides the patch version of the Software.		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucIntegerParamDef
<b>Range</b>	0 - 255		
<b>Default value</b>	As per the driver version		
<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Published-Information	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	IFX	<b>Scope</b>	LOCAL

## 1 Wdg driver

**Table 11 Specification for SwPatchVersion (continued)**

<b>Dependency</b>	-
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.

### 1.3.1.3.9 VendorApiInfix

**Table 12 Specification for VendorApiInfix**

<b>Name</b>	VendorApiInfix		
<b>Description</b>	<p>In driver modules which can be instantiated several times on a single ECU, BSW00347 requires that the name of APIs is extended by the VendorId and a vendor specific name. This parameter is used to specify the vendor specific name. For example, assuming that the VendorId of the implementer is 17 and the implementer chooses VendorApiInfix as Scu then the API name Wdg_SetMode defined in the SWS will translate to Wdg_17_Scu_SetMode.</p> <p>This parameter is mandatory for all modules with upper multiplicity greater than 1. It will not be used for modules with upper multiplicity equal to 1</p>		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucStringParamDef
<b>Range</b>	String		
<b>Default value</b>	Scu		
<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Published-Information	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	IFX	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.3.10 VendorId

**Table 13 Specification for VendorId**

<b>Name</b>	VendorId		
<b>Description</b>	This parameter provides the Vendor Id		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucIntegerParamDef
<b>Range</b>	0 - 65535		
<b>Default value</b>	17		
<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Published-Information	<b>Multiplicity configuration class</b>	-

## 1 Wdg driver

**Table 13 Specification for VendorId (continued)**

<b>Origin</b>	IFX	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.4 Container: GtmTimerOutputModuleConfiguration

This container contains the elements for configuring GTM timer hardware (TOM/ATOM). The settings here are used to configure the timing needs for GTM timer. Availability of this container depends on the underlying derivative.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

#### 1.3.1.4.1 GtmTimerClockSelect

**Table 14 Specification for GtmTimerClockSelect**

<b>Name</b>	GtmTimerClockSelect		
<b>Description</b>	<p>This parameter decides the Clock Source for TOM/ATOM timer.</p> <p>Values:</p> <p>If GtmTimerUsed is TOM module.</p> <p>GTM_FIXED_CLOCK_0: Fixed Clock 0 is selected.</p> <p>GTM_FIXED_CLOCK_1: Fixed Clock 1 is selected.</p> <p>GTM_FIXED_CLOCK_2: Fixed Clock 2 is selected.</p> <p>GTM_FIXED_CLOCK_3: Fixed Clock 3 is selected.</p> <p>GTM_FIXED_CLOCK_4: Fixed Clock 4 is selected.</p> <p>If GtmTimerUsed is ATOM module.</p> <p>GTM_CONFIGURABLE_CLOCK_0: Configurable Clock0 is selected.</p> <p>GTM_CONFIGURABLE_CLOCK_1: Configurable Clock1 is selected.</p> <p>GTM_CONFIGURABLE_CLOCK_2: Configurable Clock2 is selected.</p> <p>GTM_CONFIGURABLE_CLOCK_3: Configurable Clock3 is selected.</p> <p>GTM_CONFIGURABLE_CLOCK_4: Configurable Clock4 is selected.</p> <p>GTM_CONFIGURABLE_CLOCK_5: Configurable Clock5 is selected.</p> <p>GTM_CONFIGURABLE_CLOCK_6: Configurable Clock6 is selected.</p> <p>GTM_CONFIGURABLE_CLOCK_7: Configurable Clock7 is selected.</p> <p>Minimum CLOCK ID is selected as the default value.</p>		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucEnumerationParamDef
<b>Range</b>	<p>GTM_CONFIGURABLE_CLOCK_0: Configurable Clock 0 will be supplied to the Channel</p> <p>GTM_CONFIGURABLE_CLOCK_1: Configurable Clock 1 will be supplied to the Channel</p> <p>GTM_CONFIGURABLE_CLOCK_2: Configurable Clock 2 will be supplied to the Channel</p> <p>GTM_CONFIGURABLE_CLOCK_3: Configurable Clock 3 will be supplied to the Channel</p>		

## 1 Wdg driver

**Table 14 Specification for GtmTimerClockSelect (continued)**

	GTM_CONFIGURABLE_CLOCK_4: Configurable Clock 4 will be supplied to the Channel GTM_CONFIGURABLE_CLOCK_5: Configurable Clock 5 will be supplied to the Channel GTM_CONFIGURABLE_CLOCK_6: Configurable Clock 6 will be supplied to the Channel GTM_CONFIGURABLE_CLOCK_7: Configurable Clock 7 will be supplied to the Channel GTM_FIXED_CLOCK_0: Fixed Clock 0 will be supplied to the Channel GTM_FIXED_CLOCK_1: Fixed Clock 1 will be supplied to the Channel GTM_FIXED_CLOCK_2: Fixed Clock 2 will be supplied to the Channel GTM_FIXED_CLOCK_3: Fixed Clock 3 will be supplied to the Channel GTM_FIXED_CLOCK_4: Fixed Clock 4 will be supplied to the Channel		
<b>Default value</b>	GTM_FIXED_CLOCK_0		
<b>Post-build variant value</b>	TRUE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Post-Build	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	IFX	<b>Scope</b>	LOCAL
<b>Dependency</b>	GtmTimerUsed		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.4.2 GtmTimerUsed

**Table 15 Specification for GtmTimerUsed**

<b>Name</b>	GtmTimerUsed		
<b>Description</b>	<p>The TOM/ATOM Channel resource assigned to the watchdog timer.</p> <p>This parameter is list of all the GTM timer channels (TOM/ATOM) used by WDG Driver. Referred timer channel in MCU should have TomChannelUsage/ AtomChannelUsage as USED_BY_WDG_DRIVER.</p> <p><i>Note: If referred timer channel is not marked as "GTM_TOM_CHANNEL_USED_BY_WDG/ GTM_ATOM_CHANNEL_USED_BY_WDG" via the parameters McuGtmTomChannelAllocationConf / McuGtmAtomChannelAllocationConf in MCU module an error message will be raised.</i></p> <p><i>The TOM/ATOM channel should be unreserved in MCU, if WdgTriggerTimerSelection is changed to STM_TIMER or in MCU module an error message will be raised.</i></p> <p><i>Since the name of the dependent container is user configurable, the default value is kept as NULL.</i></p>		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucReferenceDef
<b>Range</b>	Reference to Node: McuGtmTomChannelAllocationConf, McuGtmAtomChannelAllocationConf		
<b>Default value</b>	NULL		



## 1 Wdg driver

**Table 15 Specification for GtmTimerUsed (continued)**

<b>Post-build variant value</b>	TRUE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Post-Build	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	IFX	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.5 Container: Wdg

Configuration of the Wdg module.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

#### 1.3.1.5.1 IMPLEMENTATION\_CONFIG\_VARIANT

**Table 16 Specification for IMPLEMENTATION\_CONFIG\_VARIANT**

<b>Name</b>	IMPLEMENTATION_CONFIG_VARIANT		
<b>Description</b>	WDG driver will support the configuration variant VARIANT-POST-BUILD hence this is the default value. <i>Note : Alias name for this parameter is Config Variant.</i>		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucEnumerationParamDef
<b>Range</b>	VariantPostBuild:		
<b>Default value</b>	VariantPostBuild		
<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Pre-Compile	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	IFX	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.6 Container: WdgDemEventParameterRefs

This container lists down all the Production error event configuration parameters for WDG driver.

If this container does not exist in the configuration, then DEM handling is not performed.

Post-Build Variant Multiplicity: FALSE

Multiplicity Configuration Class: Pre-Compile

## 1 Wdg driver

### 1.3.1.6.1 WDG\_E\_DISABLE\_REJECTED

**Table 17 Specification for WDG\_E\_DISABLE\_REJECTED**

<b>Name</b>	WDG_E_DISABLE_REJECTED		
<b>Description</b>	<p>The existence of this parameter decides if the WDG module would raise this particular dem.</p> <p>The user has to point to the right Dem event parameter for this fault. The short name of the pointed Dem parameter will be used as the symbol that will be passed as the first parameter to Dem_ReportEventStatus() function.</p> <p>The extended production error WDG_17_SCU_E_DISABLE_REJECTED will be reported with FAILED when disabling of the watchdog mode failed and will be reported with PASSED when disabling of the watchdog mode not failed.</p> <p>The minimum value of multiplicity depends on the WdgSafetyEnable. (if WdgSafetyEnable is enabled then multiplicity is 1..1 and if disabled then multiplicity is 0..1)</p>		
<b>Multiplicity</b>	0..1	<b>Type</b>	EcucSymbolicNameReferenceDef
<b>Range</b>	Reference to Node: DemEventParameter		
<b>Default value</b>	NULL		
<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	FALSE
<b>Value configuration class</b>	Pre-Compile	<b>Multiplicity configuration class</b>	Pre-Compile
<b>Origin</b>	AUTOSAR_ECUC	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.6.2 WDG\_E\_MODE\_FAILED

**Table 18 Specification for WDG\_E\_MODE\_FAILED**

<b>Name</b>	WDG_E_MODE_FAILED		
<b>Description</b>	<p>The existence of this parameter decides if the WDG module would raise this particular dem.</p> <p>The user has to point to the right Dem event parameter for this fault. The short name of the pointed Dem parameter will be used as the symbol that will be passed as the first parameter to Dem_ReportEventStatus() function.</p> <p>Reference to DemEventParameter.</p> <p>The extended production error WDG_17_SCU_E_MODE_FAILED will be reported with FAILED when setting of the watchdog mode failed and will be reported with PASSED when setting of the watchdog mode not failed.</p> <p>The minimum value of multiplicity depends on the WdgSafetyEnable.</p>		

## 1 Wdg driver

**Table 18 Specification for WDG\_E\_MODE\_FAILED (continued)**

	(if WdgSafetyEnable is enabled then multiplicity is 1..1 and if disabled then multiplicity is 0..1)		
<b>Multiplicity</b>	0..1	<b>Type</b>	EcucSymbolicNameReferenceDef
<b>Range</b>	Reference to Node: DemEventParameter		
<b>Default value</b>	NULL		
<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	FALSE
<b>Value configuration class</b>	Pre-Compile	<b>Multiplicity configuration class</b>	Pre-Compile
<b>Origin</b>	AUTOSAR_ECUC	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.7 Container: WdgExternalConfiguration

This container contains the configuration items for external watchdog hardware.

This container and its parameters are not used. The default container and parameter will be retained from AUTOSAR and disabled.

(This configuration is not provided as the external watchdog is not supported by Infineon.)

Post-Build Variant Multiplicity: TRUE

Multiplicity Configuration Class: Pre-Compile

### 1.3.1.8 Container: WdgGeneral

All general parameters of the watchdog driver are collected here.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

#### 1.3.1.8.1 WdgDevErrorDetect

**Table 19 Specification for WdgDevErrorDetect**

<b>Name</b>	WdgDevErrorDetect		
<b>Description</b>	Switches the default error detection and notification ON or OFF. True : enabled (ON). False: disabled (OFF).		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucBooleanParamDef
<b>Range</b>	TRUE FALSE		

## 1 Wdg driver

**Table 19 Specification for WdgDevErrorDetect (continued)**

<b>Default value</b>	FALSE		
<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Pre-Compile	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	AUTOSAR_ECUC	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.8.2 WdgDisableAllowed

**Table 20 Specification for WdgDisableAllowed**

<b>Name</b>	WdgDisableAllowed		
<b>Description</b>	<p>This parameter is disabled.</p> <p>Instead, WdgCPUDisableAllowed is provided to enable/disable CPU specific WDG for every CPU WDG configured.</p> <p>Compile switch to allow / forbid disabling the watchdog driver during runtime.</p> <p>True: Disabling the WDG driver at runtime is allowed.</p> <p>False: Disabling the WDG driver at runtime is not allowed.</p>		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucBooleanParamDef
<b>Range</b>	TRUE FALSE		
<b>Default value</b>	FALSE		
<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Pre-Compile	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	AUTOSAR_ECUC	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.8.3 WdgEcucPartitionRef

**Table 21 Specification for WdgEcucPartitionRef**

<b>Name</b>	WdgEcucPartitionRef
-------------	---------------------

## 1 Wdg driver

**Table 21 Specification for WdgEcucPartitionRef (continued)**

<b>Description</b>	Parameter support is added only for AUTOSAR schema compliance, this parameter is not used in code generation logic, hence this parameter is made editable false.		
<b>Multiplicity</b>	0..1	<b>Type</b>	EcucSymbolicNameReferenceDef
<b>Range</b>	Reference to Node: EcucPartition		
<b>Default value</b>	NULL		
<b>Post-build variant value</b>	TRUE	<b>Post-build variant multiplicity</b>	TRUE
<b>Value configuration class</b>	Pre-Compile	<b>Multiplicity configuration class</b>	Pre-Compile
<b>Origin</b>	AUTOSAR_ECUC	<b>Scope</b>	ECU
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar version 4.4.0.		

### 1.3.1.8.4 WdgIndex

**Table 22 Specification for WdgIndex**

<b>Name</b>	WdgIndex		
<b>Description</b>	Wdg driver ID which can be used by Wdg Interface as a reference. The default value can be modified by the user as per the needs of the application.		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucIntegerParamDef
<b>Range</b>	0 - 255		
<b>Default value</b>	0		
<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Pre-Compile	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	AUTOSAR_ECUC	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.8.5 WdgInitApiMode

**Table 23 Specification for WdgInitApiMode**

<b>Name</b>	WdgInitApiMode		
<b>Description</b>	MCAL_SUPERVISOR: Wdg Init API will run in Supervisor mode.		

## 1 Wdg driver

**Table 23 Specification for WdgInitApiMode (continued)**

	MCAL_USER1: Wdg Init API will run in USER1 mode. Since WDG driver accesses the SFRs, it is more efficient to operate the WDG driver in supervisor mode. Hence, the default mode of operation is supervisor.		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucEnumerationParamDef
<b>Range</b>	WDG_MCAL_SUPERVISOR: APIs runs in Supervisor mode. WDG_MCAL_USER1: APIs runs in USER1 mode.		
<b>Default value</b>	WDG_MCAL_SUPERVISOR		
<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Pre-Compile	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	IFX	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.8.6 WdgInitCheckApi

**Table 24 Specification for WdgInitCheckApi**

<b>Name</b>	WdgInitCheckApi		
<b>Description</b>	Switches the InitCheck API ON or OFF. True : enabled (ON). False: disabled (OFF). The detection of safety related errors is enabled by default to ensure that safety issues are addressed during the product lifecycle.		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucBooleanParamDef
<b>Range</b>	TRUE FALSE		
<b>Default value</b>	FALSE		
<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Pre-Compile	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	IFX	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		

## 1 Wdg driver

**Table 24 Specification for WdgInitCheckApi (continued)**

<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.
------------------------	--

### 1.3.1.8.7 WdgInitialTimeout

**Table 25 Specification for WdgInitialTimeout**

<b>Name</b>	WdgInitialTimeout		
<b>Description</b>	<p>This parameter is disabled.</p> <p>Instead, WdgCPUInitialTimeout is provided as initial timeout for every CPU WDG configured. This initial timeout (sec) for the trigger condition to be initialized during Init function. It will not be larger than WdgMaxTimeout.</p> <p>The default value is retained same as the WdgCPUInitialTimeout parameter. However, this parameter is disabled and not used by the WDG driver for any of its functionality.</p>		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucFloatParamDef
<b>Range</b>	0.001s - 65.535s		
<b>Default value</b>	5s		
<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Pre-Compile	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	AUTOSAR_ECUC	<b>Scope</b>	LOCAL
<b>Dependency</b>	WdgMaxTimeout		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.8.8 WdgMaxTimeout

**Table 26 Specification for WdgMaxTimeout**

<b>Name</b>	WdgMaxTimeout		
<b>Description</b>	<p>This parameter is disabled.</p> <p>Instead, WdgCPUMaxTimeout is provided as maximum CPU specific WDG timeout for every CPU WDG configured. This is the maximum window period of the Watchdog timer.</p> <p>The default value is retained same as the WdgCPUMaxTimeout parameter. However, this parameter is disabled and not used by the WDG driver for any of its functionality.</p>		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucFloatParamDef
<b>Range</b>	0.001s - 65.535s		
<b>Default value</b>	32s		
<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	-

## 1 Wdg driver

**Table 26 Specification for WdgMaxTimeout (continued)**

<b>Value configuration class</b>	Pre-Compile	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	AUTOSAR_ECUC	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.8.9 WdgRunArea

**Table 27 Specification for WdgRunArea**

<b>Name</b>	WdgRunArea		
<b>Description</b>	<p>This feature is not provided by Infineon and is disabled .</p> <p>The visibility is false for this parameter.</p> <p>Represents the WDG driver execution area is either from ROM(Flash) or RAM as required with the particular microcontroller.</p> <p>The default value is ROM. However, this parameter is disabled and not used by the WDG driver for any of its functionality.</p>		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucEnumerationParamDef
<b>Range</b>	<p>RAM: This feature is not provided by Infineon.</p> <p>This visibility is false. WDG driver to be executed out of RAM area</p> <p>ROM: This feature is not provided by Infineon.</p> <p>This parameter is disabled. WDG driver to be executed out of ROM area.</p>		
<b>Default value</b>	ROM		
<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Pre-Compile	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	AUTOSAR_ECUC	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.8.10 WdgRuntimeApiMode

**Table 28 Specification for WdgRuntimeApiMode**

<b>Name</b>	WdgRuntimeApiMode		
<b>Description</b>	<p>MCAL_SUPERVISOR: Wdg runtime APIs will run in Supervisor mode.</p> <p>MCAL_USER1: Wdg runtime APIs will run in USER1 mode.</p>		



## 1 Wdg driver

**Table 28 Specification for WdgRuntimeApiMode (continued)**

	Since WDG driver accesses the SFRs, it is more efficient to operate the WDG driver in supervisor mode. Hence, the default mode of operation is supervisor.		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucEnumerationParamDef
<b>Range</b>	WDG_MCAL_SUPERVISOR: APIs runs in Supervisor mode. WDG_MCAL_USER1: APIs runs in USER1 mode.		
<b>Default value</b>	WDG_MCAL_SUPERVISOR		
<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Pre-Compile	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	IFX	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.8.11 WdgSafetyEnable

**Table 29 Specification for WdgSafetyEnable**

<b>Name</b>	WdgSafetyEnable		
<b>Description</b>	Switch to enable reporting of safety DETs. True : enabled (ON). False: disabled (OFF). <i>Note: when this switch is enabled AUTOSAR DETs are enabled by default.</i> The detection of safety related errors is enabled by default to ensure that safety issues are addressed during the product lifecycle.		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucBooleanParamDef
<b>Range</b>	TRUE FALSE		
<b>Default value</b>	TRUE		
<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Pre-Compile	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	IFX	<b>Scope</b>	LOCAL
<b>Dependency</b>	WDG_E_MODE_FAILED, WDG_E_DISABLE_REJECTED, WdgDemEventParameterRefs		

## 1 Wdg driver

**Table 29 Specification for WdgSafetyEnable (continued)**

<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.
------------------------	--

### 1.3.1.8.12 WdgTriggerLocation

**Table 30 Specification for WdgTriggerLocation**

<b>Name</b>	WdgTriggerLocation		
<b>Description</b>	<p>This feature is not provided by Infineon and is disabled .</p> <p>The visibility is false for this parameter.</p> <p>Location (memory address) of the watchdog trigger routine.</p> <p>The default value is NULL_PTR. However, this parameter is disabled and not used by the WDG driver for any of its functionality.</p>		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucFunctionNameDef
<b>Range</b>	String		
<b>Default value</b>	NULL_PTR		
<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Pre-Compile	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	AUTOSAR_ECUC	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.8.13 WdgTriggerTimerSelection

**Table 31 Specification for WdgTriggerTimerSelection**

<b>Name</b>	WdgTriggerTimerSelection		
<b>Description</b>	<p>Hardware timer used to service WDG during window period. It is common for all WDTs.</p> <p>GTM_TIMER (Used to service WDTs.)</p> <p>STM_TIMER (Used to service WDTs,)</p> <p>The default value can be modified by the user as per the needs of the application.</p> <p><i>Note: GTM_TIMER will not be available for selection in devices without GTM.</i></p>		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucEnumerationParamDef
<b>Range</b>	<p>GTM_TIMER: GTM timer is used to service WDTs, in devices with GTM.</p> <p>STM_TIMER: STM timer is used to service WDTs.</p>		
<b>Default value</b>	STM_TIMER		

## 1 Wdg driver

**Table 31 Specification for WdgTriggerTimerSelection (continued)**

<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Pre-Compile	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	IFX	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.8.14 WdgVersionInfoApi

**Table 32 Specification for WdgVersionInfoApi**

<b>Name</b>	WdgVersionInfoApi		
<b>Description</b>	Compile switch to enable / disable the version information API. True : API enabled False: API disabled The optional APIs are disabled by default to minimize the executable code size.		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucBooleanParamDef
<b>Range</b>	TRUE FALSE		
<b>Default value</b>	FALSE		
<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Pre-Compile	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	AUTOSAR_ECUC	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.9 Container: WdgPublishedInformation

Container holding all WDG specific published information parameters

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

## 1 Wdg driver

### 1.3.1.9.1 WdgTriggerMode

**Table 33 Specification for WdgTriggerMode**

<b>Name</b>	WdgTriggerMode		
<b>Description</b>	Watchdog trigger mode (toggle/window/both) The default value is WDG_WINDOW since this the trigger mode supported in the WDG driver. This value is not editable.		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucEnumerationParamDef
<b>Range</b>	WDG_BOTH: Watchdog trigger mode is both Window and Toggle. WDG_TOGGLE: Watchdog trigger mode is Toggle. WDG_WINDOW: Watchdog trigger mode is Window.		
<b>Default value</b>	WDG_WINDOW		
<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Published-Information	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	AUTOSAR_ECUC	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.10 Container: WdgSettingsConfig

This is available under WDG main Container. This container configures exactly one watchdog timer per CPU.

Post-Build Variant Multiplicity: FALSE

Multiplicity Configuration Class: Pre-Compile

#### 1.3.1.10.1 WdgCPUDisableAllowed

**Table 34 Specification for WdgCPUDisableAllowed**

<b>Name</b>	WdgCPUDisableAllowed		
<b>Description</b>	Compile switch to allow/forbid the Watchdog Driver change mode to WDGIF_OFF_MODE at runtime for the CPU WDT selected by container. This parameter is considered only if the dependent parameter WdgCPUDisableAllowed is True.  This parameter is defined by Infineon.  The optional features are disabled by default to minimize the executable code size.		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucBooleanParamDef
<b>Range</b>	TRUE FALSE		

## 1 Wdg driver

**Table 34 Specification for WdgCPUDisableAllowed (continued)**

<b>Default value</b>	FALSE		
<b>Post-build variant value</b>	TRUE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Post-Build	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	IFX	<b>Scope</b>	LOCAL
<b>Dependency</b>	WdgDefaultMode		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.10.2 WdgCPUInitialPassowrd

**Table 35 Specification for WdgCPUInitialPassowrd**

<b>Name</b>	WdgCPUInitialPassowrd		
<b>Description</b>	<p>Initial password for the password access of the CPU WDG.</p> <p>The default password after Application Reset is 00000000111100B .</p> <p>The default value is same as the application reset value but can be modified by the user as per the needs of the application and it shall lie within the specified boundary value.</p>		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucIntegerParamDef
<b>Range</b>	0 - 16383		
<b>Default value</b>	60		
<b>Post-build variant value</b>	TRUE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Post-Build	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	IFX	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.10.3 WdgCPUInitialTimeout

**Table 36 Specification for WdgCPUInitialTimeout**

<b>Name</b>	WdgCPUInitialTimeout		
<b>Description</b>	<p>This is the initial window period that is active as soon Wdg_17_Scu_Init is called for the core. It is used to calculate the value of the trigger counter which is used to service the WDT just after initialization.</p> <p>This parameter is defined by Infineon.</p>		

## 1 Wdg driver

**Table 36 Specification for WdgCPUInitialTimeout (continued)**

	The default value can be modified by the user as per the needs of the application and it shall lie within the specified boundary value.		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucFloatParamDef
<b>Range</b>	0.001s - 65.535s		
<b>Default value</b>	5s		
<b>Post-build variant value</b>	TRUE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Post-Build	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	IFX	<b>Scope</b>	LOCAL
<b>Dependency</b>	WdgCPUMaxTimeout		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.10.4 WdgCPUMaxTimeout

**Table 37 Specification for WdgCPUMaxTimeout**

<b>Name</b>	WdgCPUMaxTimeout		
<b>Description</b>	<p>This is the maximum window period for the core specific watchdog timer. It is considered as the maximum window period for this particular core.</p> <p>This parameter is defined by Infineon.</p> <p>The default value can be modified by the user as per the needs of the application and it shall lie within the specified boundary value.</p>		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucFloatParamDef
<b>Range</b>	0.001s - 65.535s		
<b>Default value</b>	32s		
<b>Post-build variant value</b>	TRUE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Post-Build	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	IFX	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

## 1 Wdg driver

### 1.3.1.10.5 WdgCoreId

**Table 38 Specification for WdgCoreId**

<b>Name</b>	WdgCoreId		
<b>Description</b>	<p>This parameter refers to CORE ID to which the watchdog settings belong.</p> <p>This parameter is defined by Infineon.</p> <p>CPU Watchdog available in each device [Range]:</p> <ul style="list-style-type: none"> <li>- TC39x = 0-5,</li> <li>- TC38x = 0-3,</li> <li>- TC37x, TC35x = 0-2,</li> <li>- TC36x, TC33x(ED\ADAS) = 0-1,</li> <li>- TC33x = 0.</li> </ul> <p>The default value can be modified by the user as per the needs of the application and it shall lie within the specified boundary value.</p>		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucIntegerParamDef
<b>Range</b>	0 - depends on the device		
<b>Default value</b>	0		
<b>Post-build variant value</b>	FALSE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Pre-Compile	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	IFX	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.10.6 WdgDefaultMode

**Table 39 Specification for WdgDefaultMode**

<b>Name</b>	WdgDefaultMode		
<b>Description</b>	<p>Default mode of WDG Driver initialization for WDTx. The Description on each Wdg Mode is given below.</p> <ul style="list-style-type: none"> <li>- Off-Mode: The watchdog hardware is disabled / shut down.</li> <li>- Slow-Mode: Triggering the watchdog hardware can be done with a long timeout period. This mode for example can be used during system startup / initialization phase.</li> <li>- Fast-Mode: Triggering the watchdog hardware has to be done with a short timeout period. This mode for example can be used during normal operations of the ECU.</li> </ul> <p>ImplementationType: WdgIf_ModeType</p> <p>The default value can be modified by the user as per the needs of the application and it shall lie within the specified boundary value.</p>		

## 1 Wdg driver

**Table 39 Specification for WdgDefaultMode (continued)**

<b>Multiplicity</b>	1..1	<b>Type</b>	EcucEnumerationParamDef
<b>Range</b>	WDGIF_FAST_MODE: Default watchdog mode is "fast" WDGIF_OFF_MODE: Default watchdog mode is "Off" Dependency : This mode can be set if the WdgDisableAllowed is True and WdgCPUDisableAllowed is True WDGIF_SLOW_MODE: Default watchdog mode is "slow"		
<b>Default value</b>	WDGIF_SLOW_MODE		
<b>Post-build variant value</b>	TRUE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Post-Build	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	AUTOSAR_ECUC	<b>Scope</b>	LOCAL
<b>Dependency</b>	WdgDisableAllowed		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.10.7 WdgSystemClockRef

**Table 40 Specification for WdgSystemClockRef**

<b>Name</b>	WdgSystemClockRef		
<b>Description</b>	This parameter refers to the system clock configured, It will refer to a valid McuClockSetting Configuration.  <i>Note: WDG module will work correctly only for this system clock setting. No changes should be done to the system clock if WDG is to be kept functioning.</i>  Since the name of the dependent container is user configurable, the default value is kept as NULL.		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucReferenceDef
<b>Range</b>	Reference to Node: McuClockReferencePointConfig		
<b>Default value</b>	NULL		
<b>Post-build variant value</b>	TRUE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Post-Build	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	IFX	<b>Scope</b>	LOCAL
<b>Dependency</b>	McuClockReferencePoint		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		



## 1 Wdg driver

### 1.3.1.11 Container: WdgSettingsFast

Hardware dependent settings for the WDG driver fast mode.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

#### 1.3.1.11.1 WdgFastModeTimeoutValue

**Table 41** Specification for WdgFastModeTimeoutValue

Name	WdgFastModeTimeoutValue		
Description	<p>This parameter is the fast mode timeout in seconds for WDTx. It is used to calculate the reload value of the WDT in fast mode.</p> <p>For TC3xx, fast timeout maximum value can be calculated as follows:</p> <ul style="list-style-type: none"> <li>- fSPB = 100 MHz</li> <li>- Clock divider(fast) = 256</li> <li>- WDT frequency = <math>100\,000\,000/256 = 390625</math> Hz</li> <li>- 1 tick = <math>1/390625 = 0.00000256</math>s</li> </ul> <p>Minimum timeout = 0.00000256s Maximum timeout = <math>65536 * 0.00000256 = 0.16777216</math>s (approximately)</p> <p>Following condition should be satisfied :-</p> <ul style="list-style-type: none"> <li>-- WdgFastModeTimeoutValue less than WdgSlowModeTimeoutValue</li> <li>-- WdgFastModeTimeoutValue greater than WdgFastRefreshTime</li> </ul> <p>The default value can be modified by the user as per the needs of the application and it shall lie within the specified boundary value.</p>		
Multiplicity	1..1	Type	EcucFloatParamDef
Range	0.001s - 0.16777216s		
Default value	0.01s		
Post-build variant value	TRUE	Post-build variant multiplicity	-
Value configuration class	Post-Build	Multiplicity configuration class	-
Origin	IFX	Scope	LOCAL
Dependency	WdgFastRefreshTime, WdgSlowModeTimeoutValue		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.12 Container: WdgSettingsOff

Hardware dependent settings for the WDG driver off mode.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

## 1 Wdg driver

### 1.3.1.13 Container: WdgSettingsSlow

Hardware dependent settings for the WDG driver slow mode.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

#### 1.3.1.13.1 WdgSlowModeTimeoutValue

**Table 42 Specification for WdgSlowModeTimeoutValue**

<b>Name</b>	WdgSlowModeTimeoutValue		
<b>Description</b>	<p>Slow mode timeout in seconds for WDTx. It is used to calculate the reload value of the WDT in slow mode.</p> <p>For TC3xx, Slow timeout maximum value can be calculated as follows:</p> <ul style="list-style-type: none"> <li>- fSPB = 100 MHz</li> <li>- Clock divider(slow) = 16384</li> <li>- WDT frequency = <math>100\,000\,000 / 16384 = 6103.515625</math> Hz</li> <li>- 1 tick = <math>1 / 6103.515625 = 0.00016384</math>s</li> </ul> <p>Minimum timeout = 0.00016384s</p> <p>Maximum timeout = <math>65536 * 0.00016384 = 10.73741824</math>s (approximately)</p> <p>Following condition should be satisfied :-</p> <ul style="list-style-type: none"> <li>-- WdgSlowModeTimeoutValue greater than WdgSlowRefreshTime</li> </ul> <p>The default value can be modified by the user as per the needs of the application and it shall lie within the specified boundary value.</p>		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucFloatParamDef
<b>Range</b>	0.001s - 10.73741824s		
<b>Default value</b>	0.02s		
<b>Post-build variant value</b>	TRUE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Post-Build	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	IFX	<b>Scope</b>	LOCAL
<b>Dependency</b>	WdgSlowRefreshTime		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

#### 1.3.1.14 Container: WdgTriggerTimerSetting

This container contains the configuration elements for configuring GTM/STM hardware. The settings here are used to configure the timing needs for WDG module

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

## 1 Wdg driver

### 1.3.1.14.1 WdgFastRefreshTime

**Table 43 Specification for WdgFastRefreshTime**

<b>Name</b>	WdgFastRefreshTime		
<b>Description</b>	<p>Fast mode GTM/STM callback period in seconds for WDTx. It is used to calculate the value of the trigger counter used to service the WDT for fast mode.</p> <p>Following condition should be satisfied :-</p> <ul style="list-style-type: none"> <li>-- WdgFastRefreshTime less than WdgSlowRefreshTime.</li> <li>-- WdgFastRefreshTime less than WdgFastModeTimeoutValue</li> </ul> <p>The default value can be modified by the user as per the needs of the application and it shall lie within the specified boundary value.</p>		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucFloatParamDef
<b>Range</b>	0.001s - 0.16777216s		
<b>Default value</b>	0.007s		
<b>Post-build variant value</b>	TRUE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Post-Build	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	IFX	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.1.14.2 WdgSlowRefreshTime

**Table 44 Specification for WdgSlowRefreshTime**

<b>Name</b>	WdgSlowRefreshTime		
<b>Description</b>	<p>Slow mode GTM/STM callback period in seconds for WDTx.</p> <p>It is used to calculate the value of the trigger counter used to service the WDT for slow mode.</p> <p>Following condition should be satisfied :-</p> <ul style="list-style-type: none"> <li>-- WdgSlowRefreshTime greater than WdgFastRefreshTime</li> <li>-- WdgSlowRefreshTime less than WdgSlowModeTimeoutValue</li> <li>-- For GTM, WdgSlowRefreshTime less than 10.73741824s, to ensure no overflow during the Wdg timeout calculation.</li> <li>-- For STM, WdgSlowRefreshTime less than 4.294s, to ensure no overflow during the Wdg timeout calculation.</li> </ul> <p>The default value can be modified by the user as per the needs of the application and it shall lie within the specified boundary value.</p>		
<b>Multiplicity</b>	1..1	<b>Type</b>	EcucFloatParamDef
<b>Range</b>	0.001s - 10.73741824s		

## 1 Wdg driver

**Table 44**      **Specification for WdgSlowRefreshTime (continued)**

<b>Default value</b>	0.015s		
<b>Post-build variant value</b>	TRUE	<b>Post-build variant multiplicity</b>	-
<b>Value configuration class</b>	Post-Build	<b>Multiplicity configuration class</b>	-
<b>Origin</b>	IFX	<b>Scope</b>	LOCAL
<b>Dependency</b>	-		
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.		

### 1.3.2 Functions - Type definitions

This section describes all the type definitions that are used by APIs.

For other generic datatypes refer AUTOSAR SWS.

#### 1.3.2.1 Wdg\_17\_Scu\_ConfigType

**Table 45**      **Specification for Wdg\_17\_Scu\_ConfigType**

Syntax	Wdg_17_Scu_ConfigType	
Type	Structure	
File	Wdg_17_Scu.h	
Range	hardware dependent structure	The elements of the data structure are specific to the microcontroller.
Description	The elements of the data structure are specific to the microcontroller. This type definition will be detailed in Design based on data structure design. ( Used for pointers to structures holding configuration data provided to the WDG module initialization routine for configuration of the module and watchdog hardware. )	
Source	AUTOSAR	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

### 1.3.3 Functions - APIs

This section lists all the APIs for the WDG driver.

The names of the APIs are extended with VendorId and VendorApiInfix since the upper multiplicity of the WDG driver is greater than 1.

## 1 Wdg driver

### 1.3.3.1 Wdg\_17\_Scu\_Init

**Table 46 Specification for wdg\_17\_Scu\_Init API**

<b>Syntax</b>	<pre>void Wdg_17_Scu_Init (     const Wdg_17_Scu_ConfigType * const ConfigPtr )</pre>	
<b>Service ID</b>	0x00	
<b>Sync/Async</b>	Synchronous	
<b>ASIL Level</b>	B	
<b>Re-entrancy</b>	Non Reentrant	
<b>Parameters (in)</b>	ConfigPtr	Pointer to configuration set.
<b>Parameters (out)</b>	-	-
<b>Parameters (in - out)</b>	-	-
<b>Return</b>	void	-
<b>Description</b>	<p>This service is used to initialize each WDTx with configuration for the core from which this API invoked.</p> <p>To initialize WDTx, this API shall be called from CPUx (corresponding CPU core), with the pointer of corresponding core config data.</p>	
<b>Source</b>	AUTOSAR	
<b>Error handling</b>	WDG_17_SCU_E_DISABLE_REJECTED, WDG_17_SCU_E_MODE_FAILED, WDG_17_SCU_E_PARAM_CONFIG, WDG_17_SCU_E_INIT_FAILED, WDG_17_SCU_E_BUSY	
<b>Configuration dependencies</b>	-	
<b>User hints</b>	<p>- MCU should be in an initialized state before this function is called. Wdg_17_Scu_Init depends on MCU SPB clock.</p> <p>-The Wdg_17_Scu_Init API should be invoked by the user only from the initialization and should not be called anytime after the initialization procedure is completed.</p>	
<b>SFR accessed</b>	CPU_BIV(w), CPU_BTV(w), CPU_CORE_ID(r), CPU_DCON0(w), CPU_ISP(w), CPU_PCON0(w), CPU_PMA0(w), CPU_PMA1(w), CPU_SEGEN(w), GTM_ATOM_AGC_ENDIS_CTRL(w), GTM_ATOM_AGC_ENDIS_STAT(w), GTM_ATOM_AGC_OUTEN_CTRL(w), GTM_ATOM_AGC_OUTEN_STAT(w), GTM_ATOM_CH_CM0(w), GTM_ATOM_CH_CM1(w), GTM_ATOM_CH_CN0(w), GTM_ATOM_CH_CTRL(w), GTM_ATOM_CH_IRQ_EN(w), GTM_ATOM_CH_IRQ_MODE(w), GTM_ATOM_CH_IRQ_NOTIFY(w), GTM_ATOM_CH_SR0(w), GTM_ATOM_CH_SR1(w), GTM_ATOM_CH_CM0(w), GTM_ATOM_CH_CM1(w), GTM_ATOM_CH_CN0(w), GTM_ATOM_CH_CTRL(w), GTM_ATOM_CH_IRQ_EN(w), GTM_ATOM_CH_IRQ_MODE(w), GTM_ATOM_CH_IRQ_NOTIFY(w), GTM_ATOM_CH_SR0(w), GTM_ATOM_CH_SR1(w), GTM_ATOM_TGC0_ENDIS_CTRL(rw), GTM_ATOM_TGC0_ENDIS_STAT(w), GTM_ATOM_TGC0_FUPD_CTRL(rw), GTM_ATOM_TGC0_GLB_CTRL(w), GTM_ATOM_TGC0_OUTEN_CTRL(rw), GTM_ATOM_TGC0_OUTEN_STAT(w), GTM_ATOM_TGC1_ENDIS_CTRL(rw), GTM_ATOM_TGC1_ENDIS_STAT(w), GTM_ATOM_TGC1_FUPD_CTRL(rw), GTM_ATOM_TGC1_GLB_CTRL(w),	

## 1 Wdg driver

**Table 46 Specification for Wdg\_17\_Scu\_Init API (continued)**

	GTM_TOM_TGC1_OUTEN_CTRL(rw), GTM_TOM_TGC1_OUTEN_STAT(w), SCU_CCUCON0(r), SCU_OSCCON(r), SCU_SYSPLLCON0(r), SCU_SYSPLLCON1(r), SCU_WDTCPU_CON0(rw), SCU_WDTCPU_CON1(rw), SCU_WDTCPU_SR(r), STM_CMCON(rw), STM_CMP(w), STM_ICR(rw), STM_ISCR(rw), STM_TIM0(r)  <i>Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.</i>
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.

### 1.3.3.2 Wdg\_17\_Scu\_InitCheck

**Table 47 Specification for Wdg\_17\_Scu\_InitCheck API**

<b>Syntax</b>	<pre>Std_ReturnType Wdg_17_Scu_InitCheck (     const Wdg_17_Scu_ConfigType * const ConfigPtr )</pre>	
<b>Service ID</b>	0x06	
<b>Sync/Async</b>	Synchronous	
<b>ASIL Level</b>	B	
<b>Re-entrancy</b>	Non Reentrant	
<b>Parameters (in)</b>	ConfigPtr	Pointer to configuration set.
<b>Parameters (out)</b>	-	-
<b>Parameters (in - out)</b>	-	-
<b>Return</b>	Std_ReturnType	E_OK: if initialization was successful. E_NOT_OK: if initialization was unsuccessful.
<b>Description</b>	<p>This API returns the status of the module initialization in context to the core from where the API is invoked.</p> <p>The API is available when safety is enabled or initialization check is explicitly enabled.</p> <p>This API does not check any running counter values of GTM, STM and global variables.</p> <p><i>Note: Init check should be performed in the following sequence:</i></p> <ul style="list-style-type: none"> <li>- Call Wdg_17_Scu_Init from a core.</li> <li>- Call Wdg_17_Scu_InitCheck from the same core.</li> </ul>	
<b>Source</b>	IFX	
<b>Error handling</b>	-	
<b>Configuration dependencies</b>	WdgInitCheckApi	

## 1 Wdg driver

**Table 47**      **Specification for `Wdg_17_Scu_InitCheck` API (continued)**

<b>User hints</b>	None.
<b>SFR accessed</b>	CPU_CORE_ID(r), GTM_ATOM_AGC_ENDIS_STAT(r), GTM_ATOM_CH_CM0(r), GTM_ATOM_CH_CM1(r), GTM_ATOM_CH_CN0(r), GTM_ATOM_CH_CTRL(r), GTM_ATOM_CH_IRQ_EN(r), GTM_ATOM_CH_IRQ_MODE(r), GTM_ATOM_CH_SR0(r), GTM_ATOM_CH_SR1(r), GTM_TOM_CH_CM0(r), GTM_TOM_CH_CM1(r), GTM_TOM_CH_CN0(r), GTM_TOM_CH_CTRL(r), GTM_TOM_CH_IRQ_EN(r), GTM_TOM_CH_IRQ_MODE(r), GTM_TOM_CH_SR0(r), GTM_TOM_CH_SR1(r), GTM_TOM_TGC0_ENDIS_STAT(r), GTM_TOM_TGC1_ENDIS_STAT(r), SCU_WDTCPU_CON0(r), SCU_WDTCPU_CON1(r), SCU_WDTCPU_SR(r), STM_CMCON(r), STM_ICR(r)  <i>Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.</i>
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.

### 1.3.3.3 Wdg\_17\_Scu\_SetMode

**Table 48**      **Specification for `Wdg_17_Scu_SetMode` API**

<b>Syntax</b>	<pre>Std_ReturnType Wdg_17_Scu_SetMode (     const WdgIf_ModeType Mode )</pre>	
<b>Service ID</b>	0x01	
<b>Sync/Async</b>	Synchronous	
<b>ASIL Level</b>	B	
<b>Re-entrancy</b>	Non Reentrant	
<b>Parameters (in)</b>	Mode	One of the following statically configured modes: 1. WDGIF_OFF_MODE 2. WDGIF_SLOW_MODE 3. WDGIF_FAST_MODE
<b>Parameters (out)</b>	-	-
<b>Parameters (in - out)</b>	-	-
<b>Return</b>	Std_ReturnType	E_OK: if mode change is successfull E_NOT_OK: if mode change is unsucessfull
<b>Description</b>	Switches the watchdog into the Mode Requested. To change mode for WDTx, the API shall be called from CPUx (corresponding CPU core). It services the watchdog for safety to ensure window does not expire in corner cases.	
<b>Source</b>	AUTOSAR	

## 1 Wdg driver

**Table 48 Specification for Wdg\_17\_Scu\_SetMode API (continued)**

<b>Error handling</b>	WDG_17_SCU_E_MODE_FAILED, WDG_17_SCU_E_DISABLE_REJECTED, WDG_17_SCU_E_PARAM_MODE, WDG_17_SCU_E_DRIVER_STATE
<b>Configuration dependencies</b>	-
<b>User hints</b>	WDG should be in an initialized state before this function is called. This function shall be called before watchdog timeout timer elapses.
<b>SFR accessed</b>	CPU_BIV(w), CPU_BTV(w), CPU_CORE_ID(r), CPU_DCON0(w), CPU_ISP(w), CPU_PCON0(w), CPU_PMA0(w), CPU_PMA1(w), CPU_SEGEN(w), GTM_ATOM_AGC_ENDIS_CTRL(w), GTM_ATOM_AGC_ENDIS_STAT(w), GTM_ATOM_AGC_OUTEN_CTRL(w), GTM_ATOM_AGC_OUTEN_STAT(w), GTM_ATOM_CH_CM0(w), GTM_ATOM_CH_CM1(w), GTM_ATOM_CH_CN0(w), GTM_ATOM_CH_CTRL(w), GTM_ATOM_CH_IRQ_EN(w), GTM_ATOM_CH_IRQ_MODE(w), GTM_ATOM_CH_IRQ_NOTIFY(w), GTM_ATOM_CH_SR0(w), GTM_ATOM_CH_SR1(w), GTM_TOM_CH_CM0(w), GTM_TOM_CH_CM1(w), GTM_TOM_CH_CN0(w), GTM_TOM_CH_CTRL(w), GTM_TOM_CH_IRQ_EN(w), GTM_TOM_CH_IRQ_MODE(w), GTM_TOM_CH_IRQ_NOTIFY(w), GTM_TOM_CH_SR0(w), GTM_TOM_CH_SR1(w), GTM_TOM_TGC0_ENDIS_CTRL(rw), GTM_TOM_TGC0_ENDIS_STAT(w), GTM_TOM_TGC0_FUPD_CTRL(rw), GTM_TOM_TGC0_GLB_CTRL(w), GTM_TOM_TGC0_OUTEN_CTRL(rw), GTM_TOM_TGC0_OUTEN_STAT(w), GTM_TOM_TGC1_ENDIS_CTRL(rw), GTM_TOM_TGC1_ENDIS_STAT(w), GTM_TOM_TGC1_FUPD_CTRL(rw), GTM_TOM_TGC1_GLB_CTRL(w), GTM_TOM_TGC1_OUTEN_CTRL(rw), GTM_TOM_TGC1_OUTEN_STAT(w), SCU_CCUCON0(r), SCU_OSCCON(r), SCU_SYSPLLCON0(r), SCU_SYSPLLCON1(r), SCU_WDTCPU_CON0(rw), SCU_WDTCPU_CON1(rw), SCU_WDTCPU_SR(r), STM_CMCON(rw), STM_CMP(w), STM_ICR(rw), STM_ISCR(rw), STM_TIM0(r)  <i>Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.</i>
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.

### 1.3.3.4 Wdg\_17\_Scu\_SetTriggerCondition

**Table 49 Specification for Wdg\_17\_Scu\_SetTriggerCondition API**

<b>Syntax</b>	<pre>void Wdg_17_Scu_SetTriggerCondition (     const uint16 timeout )</pre>	
<b>Service ID</b>	0x03	
<b>Sync/Async</b>	Synchronous	
<b>ASIL Level</b>	B	
<b>Re-entrancy</b>	Non Reentrant	
<b>Parameters (in)</b>	timeout	Timeout value (milliseconds) for setting the trigger counter.



## 1 Wdg driver

**Table 49 Specification for Wdg\_17\_Scu\_SetTriggerCondition API (continued)**

<b>Parameters (out)</b>	-	-
<b>Parameters (in - out)</b>	-	-
<b>Return</b>	void	-
<b>Description</b>	Sets the timeout value for the trigger counter. It services the watchdog for safety to ensure window does not expire in corner cases.	
<b>Source</b>	AUTOSAR	
<b>Error handling</b>	WDG_17_SCU_E_PARAM_TIMEOUT, WDG_17_SCU_E_DRIVER_STATE	
<b>Configuration dependencies</b>	-	
<b>User hints</b>	WDG should be in an initialized state before this function is called. This function shall be called before watchdog timeout timer elapses. To set the timeout value for WDTx, the API must be called from CPUx (corresponding CPU core).	
<b>SFR accessed</b>	CPU_CORE_ID(r), GTM_ATOM_CH_CN0(w), GTM_TOM_CH_CN0(w), SCU_WDTCPU_CON0(rw), STM_CMCON(rw), STM_CMP(w), STM_ICR(rw), STM_ISCR(rw), STM_TIM0(r) <i>Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.</i>	
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.	

### 1.3.3.5 Wdg\_17\_Scu\_GetVersionInfo

**Table 50 Specification for Wdg\_17\_Scu\_GetVersionInfo API**

<b>Syntax</b>	<pre>void Wdg_17_Scu_GetVersionInfo (     Std_VersionInfoType * const versioninfo )</pre>	
<b>Service ID</b>	0x04	
<b>Sync/Async</b>	Synchronous	
<b>ASIL Level</b>	B	
<b>Re-entrancy</b>	Reentrant	
<b>Parameters (in)</b>	-	-
<b>Parameters (out)</b>	versioninfo	Pointer to where to store the version information of this module.
<b>Parameters (in - out)</b>	-	-
<b>Return</b>	void	Pointer to where to store the version information of this module.

## 1 Wdg driver

**Table 50 Specification for Wdg\_17\_Scu\_GetVersionInfo API (continued)**

<b>Description</b>	Returns the version information of the module.
<b>Source</b>	AUTOSAR
<b>Error handling</b>	WDG_17_SCU_E_PARAM_POINTER
<b>Configuration dependencies</b>	WdgVersionInfoApi
<b>User hints</b>	None.
<b>SFR accessed</b>	-
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.

### 1.3.4 Notifications and Callbacks

This section lists all the notification and callbacks of the WDG driver.

#### 1.3.4.1 Wdg\_17\_Scu\_Isr

**Table 51 Specification for Wdg\_17\_Scu\_Isr API**

<b>Syntax</b>	<pre>void Wdg_17_Scu_Isr (     const uint32 LogicalChId,     const uint32 IsrStatus )</pre>	
<b>Service ID</b>	0x05	
<b>Sync/Async</b>	Synchronous	
<b>ASIL Level</b>	B	
<b>Re-entrancy</b>	Non Reentrant	
<b>Parameters (in)</b>	LogicalChId IsrStatus	channel ID This parameter gives information about which comparator caused the interrupt.
<b>Parameters (out)</b>	-	-
<b>Parameters (in - out)</b>	-	-
<b>Return</b>	void	-
<b>Description</b>	ISR for GTM/STM interrupts	
<b>Source</b>	IFX	
<b>Error handling</b>	WDG_17_SCU_E_DRIVER_STATE, WDG_17_SCU_E_MODE_DISABLED, WDG_17_SCU_E_INVALID_INTERRUPT_SOURCE, WDG_17_SCU_E_ACCESS, WDG_17_SCU_E_OVERFLOW	

## 1 Wdg driver

**Table 51 Specification for Wdg\_17\_Scu\_Isr API (continued)**

<b>Configuration dependencies</b>	-
<b>User hints</b>	<p>This is a callback function invoked from Mcu_17_TimerIp.c when GTM/STM interrupt has occurred to service the watchdog driver.</p> <p>Timing constraint: This callback function shall be called before the watchdog timer overflows and resets the system.</p>
<b>SFR accessed</b>	<p>CPU_CORE_ID(r), GTM_ATOM_AGC_ENDIS_CTRL(w), GTM_ATOM_AGC_ENDIS_STAT(w), GTM_ATOM_AGC_OUTEN_CTRL(w), GTM_ATOM_AGC_OUTEN_STAT(w), GTM_TOM_TGC0_ENDIS_CTRL(w), GTM_TOM_TGC0_ENDIS_STAT(w), GTM_TOM_TGC0_OUTEN_CTRL(w), GTM_TOM_TGC0_OUTEN_STAT(w), GTM_TOM_TGC1_ENDIS_CTRL(w), GTM_TOM_TGC1_ENDIS_STAT(w), GTM_TOM_TGC1_OUTEN_CTRL(w), GTM_TOM_TGC1_OUTEN_STAT(w), SCU_CCUCON0(r), SCU_OSCCON(r), SCU_SYSPLLCON0(r), SCU_SYSPLLCON1(r), SCU_WDTCPU_CON0(rw), SCU_WDTCPU_SR(r), STM_CMCON(rw), STM_CMP(w), STM_ICR(rw), STM_ISCR(rw), STM_TIM0(r)</p> <p><i>Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.</i></p>
<b>Autosar Version</b>	Applicable for Autosar versions 4.2.2 and 4.4.0.

### 1.3.5 Scheduled functions

The WDG driver does not provide any scheduled functions.

### 1.3.6 Interrupt service routines

The WDG driver does not provide any interrupt handlers.

### 1.3.7 Callout

The driver does not support any callout functions.

### 1.3.8 Errors Handling

This section describes the various errors reported by the WDG driver.

Error Name: Description	Source	Error ID (AS422)	Type (AS422)	Error ID (AS440)	Type (AS440)
<b>WDG_17_SCU_E_ACCESS:</b> If the Access Error Flag is Set while servicing an interrupt. (This bit is set when an illegal Password Access or Modify Access to register WDTxCON0 was attempted).	IFX	0xCB	SAFETY	0xCB	SAFETY

## 1 Wdg driver

Error Name: Description	Source	Error ID (AS422)	Type (AS422)	Error ID (AS440)	Type (AS440)
<b>WDG_17_SCU_E_BUSY:</b> If the state of the driver is BUSY when initialization is called asynchronously.	IFX	0xCC	SAFETY	0xCC	SAFETY
<b>WDG_17_SCU_E_DISABLE_REJECTED:</b> Initialization or watchdog mode switch failed because it would disable the watchdog though this is not allowed in this configuration . - FAILED: when disabling of the watchdog mode failed. - PASSED: when disabling of the watchdog mode not failed.	AUTOSAR	Assigned by DEM	DEM	Assigned by DEM	DEM
<b>WDG_17_SCU_E_INIT_FAILED:</b> Initialization called with NULL pointer.	AUTOSAR	0x150	DET_SAFETY	0x15	DET_SAFETY
<b>WDG_17_SCU_E_INVALID_INTERRUPT_SOURCE:</b> WDG ISR is triggered if interrupt source is not CMP0 in case of GTM.  <i>Note : This error check is not done in case of STM since it is expected that the interrupt when CMP0 is configured will always arrive at STMIR0 and CMP1 is configured it will always arrive at STMIR1.</i>	IFX	0xC9	SAFETY	0xC9	SAFETY
<b>WDG_17_SCU_E_MODE_DISABLED:</b> If ISR is triggered when WDG is disabled.	IFX	0xC8	SAFETY	0xC8	SAFETY
<b>WDG_17_SCU_E_MODE_FAILED:</b> Setting watchdog mode failed. - FAILED : Setting Watchdog mode failed. - PASSED : Setting Watchdog mode passed.	AUTOSAR	Assigned by DEM0	DEM	Assigned by DEM	DEM
<b>WDG_17_SCU_E_OVERFLOW:</b> If the Overflow Error Flag is Set while servicing an interrupt. (This bit is set when the WDT overflows from FFFFH to 0000H.).	IFX	0xCA	SAFETY	0xCA	SAFETY
<b>WDG_17_SCU_E_DRIVER_STATE:</b> API service used in wrong	AUTOSAR	0x10	DET_SAFETY	0x10	DET_SAFETY

## 1 Wdg driver

Error Name: Description	Source	Error ID (AS422)	Type (AS422)	Error ID (AS440)	Type (AS440)
context (for example, module not initialized).					
<b>WDG_17_SCU_E_PARAM_CONFIG</b> : API service called with wrong / inconsistent parameter(s) . This DET will be reported if an API is called from a wrong core and if mode change was unsuccessful during initialization.	AUTOSAR	0x12	DET_SAFETY	0x12	DET_SAFETY
<b>WDG_17_SCU_E_PARAM_MODE</b> : API service called with wrong / inconsistent parameter(s).	AUTOSAR	0x11	DET_SAFETY	0x11	DET_SAFETY
<b>WDG_17_SCU_E_PARAM_POINTER</b> : API is called with wrong pointer value (for example, NULL pointer) other than initialization	AUTOSAR	0x14	DET_SAFETY	0x14	DET_SAFETY
<b>WDG_17_SCU_E_PARAM_TIMEOUT</b> : The passed timeout value is higher than the maximum timeout value.	AUTOSAR	0x13	DET_SAFETY	0x13	DET_SAFETY

### 1.3.9 Deviations and limitations

This section describes the deviations and limitations of the WDG driver.

#### 1.3.9.1 Deviations

This section describes the deviations of the WDG driver.

##### 1.3.9.1.1 Software specification deviations

This section describes the deviations from software specification.

**Table 52 Known deviations**

SWS_Wdg_00105: Rte_Dem_Type.h	The datatypes related for DEM are availed via Dem.h instead of Rte_Dem_Type.h. <i>Note: Applicable for Autosar version 4.4.0 only.</i>
File structure	WdgIf.h is included instead of WdgIf_Types.h, which is deviation for Watchdog driver Header file structure requirement.

## 1 Wdg driver

**Table 52**      **Known deviations (continued)**

	WdgIf.h includes WdgIf_Types.h, hence no functional impact. <i>Note: Applicable for Autosar version 4.2.2 only.</i>
--	--

### 1.3.9.1.2      AMDC Violations

The WDG driver does not have any AMDC violations.

### 1.3.9.1.3      VSMD Violations

This section describes the violations reported by the EB VSMD checker tool with respect to AUTOSAR.

**Table 53**      **Violations reported by VSMD checker tool for EB03**

Rule ID:	EB03
VSMD Node(s):	/AURIX2G/EcucDefs/Wdg/ WdgDemEventParameterRefs /AURIX2G/EcucDefs/Wdg/ WdgDemEventParameterRefs/ WDG_E_DISABLE_REJECTED /AURIX2G/EcucDefs/Wdg/ WdgDemEventParameterRefs/WDG_E_MODE_FAILED /AURIX2G/EcucDefs/Wdg/WdgGeneral/ WdgEcucPartitionRef /AURIX2G/EcucDefs/Wdg/WdgSettingsConfig/ WdgExternalConfiguration /AURIX2G/EcucDefs/Wdg/WdgSettingsConfig/ WdgExternalConfiguration/WdgExternalContainerRef
Description:	The StMD node has LOWER-MULTIPLICITY=0 and UPPER-MULTIPLICITY=1. The VSMD-node shall get the OPTIONAL-attribute instead of creating a list!
Additional Information:	

**Table 54**      **Violations reported by VSMD checker tool for EB09**

Rule ID:	EB09
VSMD Node(s):	/AURIX2G/EcucDefs/Wdg
Description:	EB specific rule to check consistency of parameter postBuildVariantUsed.
Additional Information:	

**Table 55**      **Violations reported by VSMD checker tool for EcucSws\_1014**

Rule ID:	EcucSws_1014
VSMD Node(s):	/AURIX2G/EcucDefs/Wdg /AURIX2G/EcucDefs/Wdg/WdgGeneral

## 1 Wdg driver

**Table 55** *Violations reported by VSMD checker tool for EcucSws\_1014 (continued)*

	/AURIX2G/EcucDefs/Wdg/WdgSettingsConfig
Description:	Additional vendor specific parameter definitions (using ParameterTypes), container definitions and references shall be added to the VSMD according to the alphabetical order.
Additional Information:	

**Table 56** *Violations reported by VSMD checker tool for EcucSws\_1035*

Rule ID:	EcucSws_1035
VSMD Node(s):	/AURIX2G/EcucDefs/Wdg /AURIX2G/EcucDefs/Wdg/ WdgDemEventParameterRefs /AURIX2G/EcucDefs/Wdg/ WdgDemEventParameterRefs/ WDG_E_DISABLE_REJECTED /AURIX2G/EcucDefs/Wdg/ WdgDemEventParameterRefs/WDG_E_MODE_FAILED /AURIX2G/EcucDefs/Wdg/WdgGeneral /AURIX2G/EcucDefs/Wdg/WdgGeneral/ WdgDevErrorDetect /AURIX2G/EcucDefs/Wdg/WdgGeneral/ WdgDisableAllowed /AURIX2G/EcucDefs/Wdg/WdgGeneral/WdgIndex /AURIX2G/EcucDefs/Wdg/WdgGeneral/ WdgInitialTimeout /AURIX2G/EcucDefs/Wdg/WdgGeneral/ WdgMaxTimeout /AURIX2G/EcucDefs/Wdg/WdgGeneral/WdgRunArea /AURIX2G/EcucDefs/Wdg/WdgGeneral/ WdgTriggerLocation /AURIX2G/EcucDefs/Wdg/WdgGeneral/ WdgVersionInfoApi /AURIX2G/EcucDefs/Wdg/WdgPublishedInformation /AURIX2G/EcucDefs/Wdg/WdgPublishedInformation/ WdgTriggerMode /AURIX2G/EcucDefs/Wdg/WdgSettingsConfig /AURIX2G/EcucDefs/Wdg/WdgSettingsConfig/ WdgDefaultMode /AURIX2G/EcucDefs/Wdg/WdgSettingsConfig/ WdgExternalConfiguration /AURIX2G/EcucDefs/Wdg/WdgSettingsConfig/ WdgExternalConfiguration/WdgExternalContainerRef /AURIX2G/EcucDefs/Wdg/WdgSettingsConfig/ WdgSettingsFast

## 1 Wdg driver

**Table 56** *Violations reported by VSMD checker tool for EcucSws\_1035 (continued)*

	/AURIX2G/EcucDefs/Wdg/WdgSettingsConfig/ WdgSettingsOff /AURIX2G/EcucDefs/Wdg/WdgSettingsConfig/ WdgSettingsSlow
Description:	For Containers, Parameters and References elements UUID must be unique (also between StMD and VSMD).
Additional Information:	

**Table 57** *Violations reported by VSMD checker tool for EcucSws\_2101*

Rule ID:	EcucSws_2101
VSMD Node(s):	/AURIX2G/EcucDefs/Wdg/ POST_BUILD_VARIANT_USED
Description:	For each ConfigurationVariant supported by the ModuleDef, there must be one ImplementationConfigClass element. In VSMD, the ImplementationConfigClass is mandatory.
Additional Information:	

**Table 58** *Violations reported by VSMD checker tool for EcucSws\_6003*

Rule ID:	EcucSws_6003
VSMD Node(s):	/AURIX2G/EcucDefs/Wdg
Description:	The SHORT-NAME of the AR-PACKAGEs of StMD and VSMD must be different to ensure a unique SHORT- NAME-path.
Additional Information:	

**Table 59** *Violations reported by VSMD checker tool for EcucSws\_6008*

Rule ID:	EcucSws_6008
VSMD Node(s):	/AURIX2G/EcucDefs/Wdg/WdgSettingsConfig
Description:	The LOWER-MULTIPLICITY of an element in the VSMD must be bigger or equal and the UPPER-MULTIPLICITY must be equal or less than in the StMD
Additional Information:	Since the container multiplicity is dependent on the number of watchdog timers/number of cores for a device. The multiplicity will be from 0 to 5. (According to AUTOSAR it is 1)

**Table 60** *Violations reported by VSMD checker tool for TpsEcuc\_06051\_ASR41*

Rule ID:	TpsEcuc_06051_ASR41
VSMD Node(s):	/AURIX2G/EcucDefs/Wdg/ POST_BUILD_VARIANT_USED



## 1 Wdg driver

**Table 60**      *Violations reported by VSMD checker tool for TpsEcuc\_06051\_ASR41 (continued)*

Description:	The implementationConfigClass of an EcucParameterDef or EcucAbstractReferenceDef in VSMD shall be the same or higher (where PreCompile configuration class is considered to be the lowest and PostBuild the highest) as in StMD with respect to the selected subset defined by the actually implemented supportedConfigVariant.
Additional Information:	

### 1.3.9.2 Limitations

This section describes the limitations of the WDG driver.

**Table 61**      **Known limitations**

Reference	Limitation
WDT in suspend mode off	Wdg_17_Scu_InitCheck shall not be invoked if debugger is connected and suspend mode is off, since the reload value will be fixed with 0xFFFC (default timeout mode reload value) and will not match with the configured reload value.

---

**Revision history**

## Revision history

**Table 62**                      **Revision History**

Date	Version	Description
2021-02-18	3.0	Released
2021-02-05	2.1	Deviation added for file structure AUTOSAR version 4.2.2 WdgIf.h inclusion.
2020-11-14	2.0	Released
2020-11-14	1.1	Updated the limitation of suspend mode
2020-07-31	1.0	Released
2020-07-30	0.1	<ul style="list-style-type: none"> <li>- Initial Version</li> <li>- Wdg_17_Scu driver chapter moved from MC-ISAR_TC3xx_UM_Basic to this document</li> <li>- LogicalChId of Wdg_17_Scu_Isr made as uint32</li> <li>- System WDT replaced as Safety WDT in SCU primary hardware peripheral section</li> <li>- Updated the range, description and variant value of Configuration Parameters WdgFastModeTimeoutValue, WdgSlowModeTimeoutValue, WdgFastRefreshTime, WdgSlowRefreshTime, WdgInitialTimeout and WdgMaxTimeout</li> </ul>

## Trademarks

All referenced product or service names and trademarks are the property of their respective owners.

**Edition 2021-02-18**

**Published by**  
**Infineon Technologies AG**  
**81726 Munich, Germany**

**© 2021 Infineon Technologies AG**  
**All Rights Reserved.**

**Do you have a question about any**  
**aspect of this document?**  
**Email: [erratum@infineon.com](mailto:erratum@infineon.com)**

**Document reference**  
**IFX-ocr1484806431059**

## IMPORTANT NOTICE

The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics ("Beschaffungsgarantie").

With respect to any examples, hints or any typical values stated herein and/or any information regarding the application of the product, Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind, including without limitation warranties of non-infringement of intellectual property rights of any third party.

In addition, any information given in this document is subject to customer's compliance with its obligations stated in this document and any applicable legal requirements, norms and standards concerning customer's products and any use of the product of Infineon Technologies in customer's applications.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer's technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

## WARNINGS

Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies' products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.