# Integration Manual

for S32K3 WDG Driver

Document Number: IM34WDGASRR21-11 Rev0000R3.0.0 Rev. 1.0

# Chapter 1

# Revision History

| Revision | Date | Author | Description |
|---|---|---|---|
| 1.0 | 31.03.2023 | NXP RTD Team | S32K3 Real-Time Drivers AUTOSAR 4.4 & R21-11 Version 3.0.0 |

# Chapter 2

# Introduction



This integration manual describes the integration requirements for the *Wdg* Driver for *S32K3XX* microcontrollers.

## 2.1 Supported Derivatives

The software described in this document is intended to be used with the following microcontroller devices of NXP Semiconductors:

- s32k310_mqfp100
- s32k310_lqfp48
- s32k311_mqfp100 / MWCT2015S_mqfp100
- s32k311_lqfp48
- s32k312_mqfp100 / MWCT2016S_mqfp100
- s32k312_mqfp172 / MWCT2016S_mqfp172
- s32k314_mqfp172
- s32k314_mapbga257
- s32k322_mqfp100 / MWCT2D16S_mqfp100
- s32k322_mqfp172 / MWCT2D16S_mqfp172
- s32k324_mqfp172 / MWCT2D17S_mqfp172
- s32k324_mapbga257

- s32k341_mqfp100
- s32k341_mqfp172
- s32k342_mqfp100
- s32k342_mqfp172
- s32k344_mqfp172
- s32k344_mapbga257
- s32k394_mapbga289
- s32k396_mapbga289
- s32k358_mqfp172
- s32k358_mapbga289
- s32k328_mqfp172
- s32k328_mapbga289
- s32k338_mqfp172
- s32k338_mapbga289
- s32k348_mqfp172
- s32k348_mapbga289
- s32m274_lqfp64
- s32m276_lqfp64

All of the above microcontroller devices are collectively named as S32K3.

Note: MWCT part numbers contain NXP confidential IP for Qi Wireless Power.

## 2.2 Overview

**AUTOSAR (AUTomotive Open System ARchitecture)** is an industry partnership working to establish standards for software interfaces and software modules for automobile electronic control systems.

AUTOSAR:

- paves the way for innovative electronic systems that further improve performance, safety and environmental friendliness.
- is a strong global partnership that creates one common standard: "Cooperate on standards, compete on implementation".
- is a key enabling technology to manage the growing electrics/electronics complexity. It aims to be prepared for the upcoming technologies and to improve cost-efficiency without making any compromise with respect to quality.
- facilitates the exchange and update of software and hardware over the service life of the vehicle.

## 2.3 About This Manual

This Technical Reference employs the following typographical conventions:

- **Boldface** style: Used for important terms, notes and warnings.
- *Italic* style: Used for code snippets in the text. Note that C language modifiers such "const" or "volatile" are sometimes omitted to improve readability of the presented code.

Notes and warnings are shown as below:

Note

This is a note.

Warning

This is a warning

## 2.4 Acronyms and Definitions

| Term | Definition |
|---|---|
| API | Application Programming Interface |
| ASM | Assembler |
| BSMI | Basic Software Make file Interface |
| C/CPP | C and C++ Source Code |
| DEM | Diagnostic Event Manager |
| DET | Development Error Tracer |
| DMA | Direct Memory Access |
| ECU | Electronic Control Unit |
| LSB | Least Signifigant Bit |
| MCU | Micro Controller Unit |
| MIDE | Multi Integrated Development Environment |
| MSB | Most Significant Bit |
| N/A | Not Applicable |
| RAM | Random Access Memory |
| SIU | Systems Integration Unit |
| SWS | Software Specification |
| VLE | Variable Length Encoding |
| XML | Extensible Markup Language |
| SWT | Software Watchdog Timer |

## 2.5 Reference List

| # | Title | Version |
|---|---|---|
| 1 | Specification of Wdg Driver | AUTOSAR Release R21-11 |
| 2 | S32K3xx Reference Manual | Rev.6, Draft B, 01/2023 |
| 3 | S32K39 and S32K37 Reference Manual | Rev. 2 Draft A, 11/2022 |
| 4 | S32M27x Reference Manual | Rev.2, Draft A, — 02/2023 |
| 5 | S32K3xx Data Sheet | Rev. 6, 11/2022 |
| 6 | S32K396 Data Sheet | Rev. 1.1 — 08/2022 |
| 7 | S32M27x Data Sheet | Rev. 2 RC — 12/2022 |
| 8 | S32K358_0P14E Mask Set Errata | Rev. 28, 9/2022 |
| 9 | S32K396_0P40E Mask Set Errata | Rev. DEC2022, 12/2022 |
| 10 | S32K311_0P98C Mask Set Errata | Rev. 6/March/2023, 3/2023 |
| 11 | S32K312: Mask Set Errata for Mask 0P09C | Rev. 25/April/2022 |
| 12 | S32K342: Mask Set Errata for Mask 0P97C | Rev. 10, 11/2022 |
| 13 | S32K3x4: Mask Set Errata for Mask 0P55A/1P55A | Rev. 14/Oct/2022 |

# Chapter 3

# Building the driver

- Build Options
- Files required for compilation
- Setting up the plugins

This section describes the source files and various compilers, linker options used for building the driver.

It also explains the EB Tresos Studio plugin setup procedure.

## 3.1 Build Options

- GCC Compiler/Assembler/Linker Options
- DIAB Compiler/Assembler/Linker Options
- GHS Compiler/Assembler/Linker Options
- IAR Compiler/Assembler/Linker Options

The RTD driver files are compiled using:

- NXP GCC 10.2.0 20200723 (Build 1728 Revision g5963bc8)
- Wind River Diab Compiler 7.0.4
- Compiler Versions: Green Hills Multi 7.1.6d / Compiler 2021.1.4
- Compiler Versions: IAR ANSI C/C++ Compiler V8.50.10 (safety version)

The compiler, assembler, and linker flags used for building the driver are explained below.

The TS_T40D34M30I0R0 part of the plugin name is composed as follows:

- T = Target_Id (e.g. T40 identifies Cortex-M architecture)
- D = Derivative_Id (e.g. D34 identifies S32K3 platform)
- M = SW_Version_Major and SW_Version_Minor
- I = SW_Version_Patch
- R = Reserved

### 3.1.1 GCC Compiler/Assembler/Linker Options

#### 3.1.1.1 GCC Compiler Options

| Compiler Option | Description |
|---|---|
| -mcpu=cortex-m7 | Targeted ARM processor for which GCC should tune the performance of the code |
| -mthumb | Generates code that executes in Thumb state |
| -mlittle-endian | Generate code for a processor running in little-endian mode |
| -mfpu=fpv5-sp-d16 | Specifies the floating-point hardware available on the target |
| -mfloat-abi=hard | Specifies the floating-point ABI to use. "hard" allows generation of floating-point instructions and uses FPU-specific calling conventions |
| -std=c99 | Specifies the ISO C99 base standard |
| -Os | Optimize for size. Enables all -O2 optimizations except those that often increase code size |
| -ggdb3 | Produce debugging information for use by GDB using the most expressive format available, including GDB extensions if at all possible. Level 3 includes extra information, such as all the macro definitions present in the program |
| -Wall | Enables all the warnings about constructions that some users consider questionable, and that are easy to avoid (or modify to prevent the warning), even in conjunction with macros |
| -Wextra | This enables some extra warning flags that are not enabled by -Wall |
| -pedantic | Issue all the warnings demanded by strict ISO C. Reject all programs that use forbidden extensions. Follows the version of the ISO C standard specified by the aforementioend -std option |
| -Wstrict-prototypes | Warn if a function is declared or defined without specifying the argument types |
| -Wundef | Warn if an undefined identifier is evaluated in an #if directive. Such identifiers are replaced with zero |
| -Wunused | Warn whenever a function, variable, label, value, macro is unused |
| -Werror=implicit-function-declaration | Make the specified warning into an error. This option throws an error when a function is used before being declared |
| -Wsign-compare | Warn when a comparison between signed and unsigned values could produce an incorrect result when the signed value is converted to unsigned. |
| -Wdouble-promotion | Give a warning when a value of type float is implicitly promoted to double |
| -fno-short-enums | Specifies that the size of an enumeration type is at least 32 bits regardless of the size of the enumerator values. |
| -funsigned-char | Let the type char be unsigned by default, when the declaration does not use either signed or unsigned |
| -funsigned-bitfields | Let a bit-field be unsigned by default, when the declaration does not use either signed or unsigned |

| Compiler Option | Description |
|---|---|
| -fno-common | Makes the compiler place uninitialized global variables in the BSS section of the object file. This inhibits the merging of tentative definitions by the linker so you get a multiple-definition error if the same variable is accidentally defined in more than one compilation unit |
| -fstack-usage | This option is only used to build test for generation Ram/↩ Stack size report. Makes the compiler output stack usage information for the program, on a per-function basis |
| -fdump-ipa-all | This option is only used to build test for generation Ram/↩ Stack size report. Enables all inter-procedural analysis dumps |
| -c | Stop after assembly and produce an object file for each source file |
| -DS32K3XX | Predefine S32K3XX as a macro, with definition 1 |
| -D $ (DERIVATIVE) | Predefine S32K3's derivative as a macro, with definition 1. For example: Predefine for S32K344 will be -DS32K344. |
| -DGCC | Predefine GCC as a macro, with definition 1 |
| -DUSE_SW_VECTOR_MODE | Predefine USE_SW_VECTOR_MODE as a macro, with definition 1. By default, the drivers are compiled to handle interrupts in Software Vector Mode |
| -DD_CACHE_ENABLE | Predefine D_CACHE_ENABLE as a macro, with definition 1. Enables data cache initalization in source file system.↩ c under the Platform driver |
| -DI_CACHE_ENABLE | Predefine I_CACHE_ENABLE as a macro, with definition 1. Enables instruction cache initalization in source file system.c under the Platform driver |
| -DENABLE_FPU | Predefine ENABLE_FPU as a macro, with definition 1. Enables FPU initalization in source file system.c under the Platform driver |
| -DMCAL_ENABLE_USER_MODE_SUPPORT | Predefine MCAL_ENABLE_USER_MODE_SUPPORT as a macro, with definition 1. Allows drivers to be configured in user mode. |
| –sysroot= | Specifies the path to the sysroot, for Cortex-M7 it is /arm-none-eabi/newlib |
| -specs=nano.specs | Use Newlib nano specs |
| -specs=nosys.specs | Do not use printf/scanf |

#### 3.1.1.2 GCC Assembler Options

| Assembler Option | Description |
|---|---|
| -Xassembler-with-cpp | Specifies the language for the following input files (rather than letting the compiler choose a default based on the file name suffix) |
| -mcpu=cortexm7 | Targeted ARM processor for which GCC should tune the performance of the code |
| -mfpu=fpv5-sp-d16 | Specifies the floating-point hardware available on the target |
| -mfloat-abi=hard | Specifies the floating-point ABI to use. "hard" allows generation of floating-point instructions and uses FPU-specific calling conventions |
| -mthumb | Generates code that executes in Thumb state |
| -c | Stop after assembly and produce an object file for each source file |

#### 3.1.1.3 GCC Linker Options

| Linker Option | Description |
|---|---|
| -Wl,-Map,filename | Produces a map file |
| -T linkerfile | Use linkerfile as the linker script. This script replaces the default linker script (rather than adding to it) |
| –entry=Reset_Handler | Specifies that the program entry point is Reset_Handler |
| -nostartfiles | Do not use the standard system startup files when linking |
| -mcpu=cortexm7 | Targeted ARM processor for which GCC should tune the performance of the code |
| -mthumb | Generates code that executes in Thumb state |
| -mfpu=fpv5-sp-d16 | Specifies the floating-point hardware available on the target |
| -mfloat-abi=hard | Specifies the floating-point ABI to use. "hard" allows generation of floating-point instructions and uses FPU-specific calling conventions |
| -mlittle-endian | Generate code for a processor running in little-endian mode |
| -ggdb3 | Produce debugging information for use by GDB using the most expressive format available, including GDB extensions if at all possible. Level 3 includes extra information, such as all the macro definitions present in the program |
| -lc | Link with the C library |
| -lm | Link with the Math library |
| -lgcc | Link with the GCC library |
| -specs=nano.specs | Use Newlib nano specs |
| -specs=nosys.specs | Do not use printf/scanf |

### 3.1.2 DIAB Compiler/Assembler/Linker Options

#### 3.1.2.1 DIAB Compiler Options

| Compiler Option | Description |
|---|---|
| -tARMCORTEXM7MG:simple | Selects target processor (hardware single-precision, software double-precision floating-point) |
| -mthumb | Selects generating code that executes in Thumb state |
| -std=c99 | Follows the C99 standard for C |
| -Oz | Like -O2 with further optimizations to reduce code size |
| -g | Generates DWARF 4.0 debug information |
| -fstandalone-debug | Emits full debug info for all types used by the program |
| -Wstrict-prototypes | Warn if a function is declared or defined without specifying the argument types |
| -Wsign-compare | Produce warnings when comparing signed type with unsigned type |
| -Wdouble-promotion | Give a warning when a value of type float is implicitly promoted to double |
| -Wunknown-pragmas | Issues a warning for unknown pragmas |
| -Wundef | Warns if an undefined identifier is evaluated in an #if directive. Such identifiers are replaced with zero |

| Compiler Option | Description |
|---|---|
| -Wextra | Enables some extra warning flags that are not enabled by '-Wall' |
| -Wall | Enables all of the most useful warnings (for historical reasons this option does not literally enable all warnings) |
| -pedantic | Emits a warning whenever the standard specified by the -std option requires a diagnostic |
| -Werror=implicit-function-declaration | Generates an error whenever a function is used before being declared |
| -fno-common | Compile common globals like normal definitions |
| -fno-signed-char | Char is unsigned |
| -fno-trigraphs | Do not process trigraph sequences |
| -V | Displays the current version number of the tool suite |
| -c | Stop after assembly and produce an object file for each source file |
| -DS32K3XX | Predefine S32K3XX as a macro, with definition 1 |
| -D $ (DERIVATIVE) | Predefine S32K3's derivative as a macro, with definition 1 |
| -DDIAB | Predefine DIAB as a macro, with definition 1 |
| -DUSE_SW_VECTOR_MODE | Predefine USE_SW_VECTOR_MODE as a macro, with definition 1. By default, the drivers are compiled to handle interrupts in Software Vector Mode |
| -DD_CACHE_ENABLE | Predefine D_CACHE_ENABLE as a macro, with definition 1. Enables data cache initalization in source file system.↩ c under the Platform driver |
| -DI_CACHE_ENABLE | Predefine I_CACHE_ENABLE as a macro, with definition 1. Enables instruction cache initalization in source file system.c under the Platform driver |
| -DENABLE_FPU | Predefine ENABLE_FPU as a macro, with definition 1. Enables FPU initalization in source file system.c under the Platform driver |
| -DMCAL_ENABLE_USER_MODE_SUPPORT | Predefine MCAL_ENABLE_USER_MODE_SUPPORT as a macro, with definition 1. Allows drivers to be configured in user mode |

#### 3.1.2.2 DIAB Assembler Options

| Assembler Option | Description |
|---|---|
| -mthumb | Selects generating code that executes in Thumb state |
| -Xpreprocess-assembly | Invokes C preprocessor on assembly files before running the assembler |
| -Xassembly-listing | Produces an .lst assembly listing file |
| -c | Stop after assembly and produce an object file for each source file |
| -tARMCORTEXM7MG:simple | Selects target processor (hardware single-precision, software double-precision floating-point) |

#### 3.1.2.3 DIAB Linker Options

| Linker Option | Description |
|---|---|
| -e Reset_Handler | Make the symbol Reset_Handler be treated as a root symbol and the start label of the application |
| linker_script_file.dld | Use linker_script_file.dld as the linker script. This script replaces the default linker script (rather than adding to it) |
| -m30 | m2 + m4 + m8 + m16 |
| -Xstack-usage | Gathers and display stack usage at link time |
| -Xpreprocess-lecl | Perform pre-processing on linker scripts |
| -Llibrary_path | Points to the libraries location for ARMV7EMMG to be used for linking |
| -lc | Links with the standard C library |
| -lm | Links with the math library |
| -tARMCORTEXM7MG:simple | Selects target processor (hardware single-precision, software double-precision floating-point) |

### 3.1.3 GHS Compiler/Assembler/Linker Options

#### 3.1.3.1 GHS Compiler Options

| Compiler Option | Description |
|---|---|
| -cpu=cortexm7 | Selects target processor: Arm Cortex M7 |
| -thumb | Selects generating code that executes in Thumb state |
| -fpu=vfpv5_d16 | Specifies hardware floating-point using the v5 version of the VFP instruction set, with 16 double-precision floating-point registers |
| -fsingle | Use hardware single-precision, software double-precision FP instructions |
| -C99 | Use (strict ISO) C99 standard (without extensions) |
| –ghstd=last | Use the most recent version of Green Hills Standard mode (which enables warnings and errors that enforce a stricter coding standard than regular C and C++) |
| -Osize | Optimize for size |
| –gnu_asm | Enables GNU extended asm syntax support |
| -dual_debug | Generate DWARF 2.0 debug information |
| -G | Generate debug information |
| -keeptempfiles | Prevents the deletion of temporary files after they are used. If an assembly language file is created by the compiler, this option will place it in the current directory instead of the temporary directory |
| -Wimplicit-int | Produce warnings if functions are assumed to return int |
| -Wshadow | Produce warnings if variables are shadowed |
| -Wtrigraphs | Produce warnings if trigraphs are detected |
| -Wundef | Produce a warning if undefined identifiers are used in #if preprocessor statements |
| –unsigned_chars | Let the type char be unsigned, like unsigned char |
| –unsigned_fields | Bitfelds declared with an integer type are unsigned |

| Compiler Option | Description |
|---|---|
| –no_commons | Allocates uninitialized global variables to a section and initializes them to zero at program startup |
| –no_exceptions | Disables C++ support for exception handling |
| –no_slash_comment | C++ style // comments are not accepted andgenerate errors |
| –prototype_errors | Controls the treatment of functions referenced or called when no prototype has been provided |
| –incorrect_pragma_warnings | Controls the treatment of valid #pragma directives that use the wrong syntax |
| -c | Stop after assembly and produce an object file for each source file |
| -DS32K3XX | Predefine S32K3XX as a macro, with definition 1 |
| -D $ (DERIVATIVE) | Predefine S32K3's derivative as a macro, with definition 1. For example: Predefine for S32K344 will be -DS32K344. |
| -DGHS | Predefine GHS as a macro, with definition 1 |
| -DUSE_SW_VECTOR_MODE | Predefine USE_SW_VECTOR_MODE as a macro, with definition 1. By default, the drivers are compiled to handle interrupts in Software Vector Mode |
| -DD_CACHE_ENABLE | Predefine D_CACHE_ENABLE as a macro, with definition 1. Enables data cache initalization in source file system.↩ c under the Platform driver |
| -DI_CACHE_ENABLE | Predefine I_CACHE_ENABLE as a macro, with definition 1. Enables instruction cache initalization in source file system.c under the Platform driver |
| -DENABLE_FPU | Predefine ENABLE_FPU as a macro, with definition 1. Enables FPU initalization in source file system.c under the Platform driver |
| -DMCAL_ENABLE_USER_MODE_SUPPORT | Predefine MCAL_ENABLE_USER_MODE_SUPPORT as a macro, with definition 1. Allows drivers to be configured in user mode |

#### 3.1.3.2 GHS Assembler Options

| Assembler Option | Description |
|---|---|
| -cpu=cortexm7 | Selects target processor: Arm Cortex M7 |
| -fpu=vfpv5_d16 | Specifies hardware floating-point using the v5 version of the VFP instruction set, with 16 double-precision floating-point registers |
| -fsingle | Use hardware single-precision, software double-precision FP instructions |
| -preprocess_assembly_files | Controls whether assembly files with standard extensions such as .s and .asm are preprocessed |
| -list | Creates a listing by using the name and directory of the object file with the .lst extension |
| -c | Stop after assembly and produce an object file for each source file |

#### 3.1.3.3 GHS Linker Options

| Linker Option | Description |
|---|---|
| -e Reset_Handler | Make the symbol Reset_Handler be treated as a root symbol and the start label of the application |
| -T linker_script_file.ld | Use linker_script_file.ld as the linker script. This script replaces the default linker script (rather than adding to it) |
| -map | Produce a map file |
| -keepmap | Controls the retention of the map file in the event of a link error |
| -Mn | Generates a listing of symbols sorted alphabetically/numerically by address |
| -delete | Instructs the linker to remove functions that are not referenced in the final executable. The linker iterates to find functions that do not have relocations pointing to them and eliminates them |
| -ignore_debug_references | Ignores relocations from DWARF debug sections when using -delete. DWARF debug information will contain references to deleted functions that may break some third-party debuggers |
| -Llibrary_path | Points to library_path (the libraries location) for thumb2 to be used for linking |
| -larch | Link architecture specific library |
| -lstartup | Link run-time environment startup routines. The source code for themodules in this library is provided in the src/libstartup directory |
| -lind_sd | Link language-independent library, containing support routines for features such as software floating point, run-time error checking, C99 complex numbers, and some general purpose routines of the ANSI C library |
| -v | Prints verbose information about the activities of the linker, including the libraries it searches to resolve undefined symbols |
| -keep=C40_Ip_AccessCode | Avoid linker remove function C40_Ip_AccessCode from Fls module because it is not referenced explicitly |
| -nostartfiles | Controls the start files to be linked into the executable |

### 3.1.4 IAR Compiler/Assembler/Linker Options

#### 3.1.4.1 IAR Compiler Options

| Compiler Option | Description |
|---|---|
| –cpu Cortex-M7 | Targeted ARM processor for which IAR should tune the performance of the code |
| –cpu_mode thumb | Generates code that executes in Thumb state |
| –endian little | Generate code for a processor running in little-endian mode |
| –fpu VFPv5-SP | Use this option to generate code that performs floating-point operations using a Floating Point Unit (FPU). Single-precision variant. |
| -e | Enables all IAR C language extensions |
| -Ohz | Optimize for size. the compiler will emit AEABI attributes indicating the requested optimization goal. This information can be used by the linker to select smaller or faster variants of DLIB library functions |
| –debug | Makes the compiler include debugging information in the object modules. Including debug information will make the object files larger |

| Compiler Option | Description |
|---|---|
| –no_clustering | Disables static clustering optimizations. Static and global variables defined within the same module will not be arranged so that variables that are accessed in the same function are close to each other |
| –no_mem_idioms | Makes the compiler not optimize certain memory access patterns |
| –do_explicit_zero_opt_in_named_sections | Disable the exception for variables in user-named sections, and thus treat explicit initializations to zero as zero initializations, not copy initializations |
| –require_prototypes | Force the compiler to verify that all functions have proper prototypes. Generates an error otherwise |
| –no_wrap_diagnostics | Does not wrap long lines in diagnostic messages |
| –diag_suppress Pa050 | Suppresses diagnostic message Pa050 |
| -DS32K3XX | Predefine S32K3XX as a macro, with definition 1 |
| -D $ (DERIVATIVE) | Predefine S32K3's derivative as a macro, with definition 1. For example: Predefine for S32K344 will be -DS32K344. |
| -DIAR | Predefine IAR as a macro, with definition 1 |
| -DUSE_SW_VECTOR_MODE | Predefine USE_SW_VECTOR_MODE as a macro, with definition 1. By default, the drivers are compiled to handle interrupts in Software Vector Mode. |
| -DD_CACHE_ENABLE | Predefine D_CACHE_ENABLE as a macro, with definition 1. Enables data cache initalization in source file system.↩ c under the Platform driver |
| -DI_CACHE_ENABLE | Predefine I_CACHE_ENABLE as a macro, with definition 1. Enables instruction cache initalization in source file system.c under the Platform driver |
| -DENABLE_FPU | Predefine ENABLE_FPU as a macro, with definition 1. Enables FPU initalization in source file system.c under the Platform driver |
| -DMCAL_ENABLE_USER_MODE_SUPPORT | Predefine MCAL_ENABLE_USER_MODE_SUPPORT as a macro, with definition 1. Allows drivers to be configured in user mode. |

#### 3.1.4.2 IAR Assembler Options

| Assembler Option | Description |
|---|---|
| –cpu Cortex-M7 | Targeted ARM processor for which IAR should generate the instruction set |
| –fpu VFPv5-SP | Use this option to generate code that performs floating-point operations using a Floating Point Unit (FPU). Single-precision variant. |
| –cpu_mode thumb | Selects the thumb mode for the assembler directive CODE |
| -g | Disables the automatic search for system include files |
| -r | Generates debug information |

#### 3.1.4.3 IAR Linker Options

| Linker Option | Description |
|---|---|
| –map filename | Produces a map file |
| –config linkerfile | Use linkerfile as the linker script. This script replaces the default linker script (rather than adding to it) |
| –cpu=Cortex-M7 | Selects the ARM processor variant to link the application for |
| –fpu VFPv5-SP | Use this option to generate code that performs floating-point operations using a Floating Point Unit (FPU). Single-precision variant. |
| –entry _start | Treats _start as a root symbol and start label |
| –enable_stack_usage | Enables stack usage analysis. If a linker map file is produced, a stack usage chapter is included in the map file |
| –skip_dynamic_initialization | Dynamic initialization (typically initialization of C++ objects with static storage duration) will not be performed automatically during application startup |
| –no_wrap_diagnostics | Does not wrap long lines in diagnostic messages |

## 3.2 Files required for compilation

This section describes the include files required to compile, assemble (if assembler code) and link the *Wdg* driver for *S32K3XX* microcontrollers. To avoid integration of incompatible files, all the include files from other modules shall have the same AR_MAJOR_VERSION and AR_MINOR_VERSION, i.e. only files with the same AUTOSAR major and minor versions can be compiled.

Wdg Files
- ..\Wdg_TS_T40D34M30I0R0\include\Wdg_43_Instance0.h
- ..\Wdg_TS_T40D34M30I0R0\include\Wdg_43_Instance1.h
- ..\Wdg_TS_T40D34M30I0R0\include\Wdg_43_Instance2.h
- ..\Wdg_TS_T40D34M30I0R0\include\Wdg_43_Instance3.h
- ..\Wdg_TS_T40D34M30I0R0\include\Wdg_43_Instance4.h
- ..\Wdg_TS_T40D34M30I0R0\src\Wdg_43_Instance0.c
- ..\Wdg_TS_T40D34M30I0R0\src\Wdg_43_Instance1.c
- ..\Wdg_TS_T40D34M30I0R0\src\Wdg_43_Instance2.c
- ..\Wdg_TS_T40D34M30I0R0\src\Wdg_43_Instance3.c
- ..\Wdg_TS_T40D34M30I0R0\src\Wdg_43_Instance4.c
- ..\Wdg_TS_T40D34M30I0R0\include\Wdg_Channel.h
- ..\Wdg_TS_T40D34M30I0R0\src\Wdg_Channel.c
- ..\Wdg_TS_T40D34M30I0R0\include\Wdg_ChannelTypes.h
- ..\Wdg_TS_T40D34M30I0R0\include\Wdg_Ipw.h
- ..\Wdg_TS_T40D34M30I0R0\src\Wdg_Ipw.c
- ..\Wdg_TS_T40D34M30I0R0\include\Wdg_Ipw_Types.h
- ..\Wdg_TS_T40D34M30I0R0\include\Swt_Ip.h
- ..\Wdg_TS_T40D34M30I0R0\include\Swt_Ip_Irq.h
- ..\Wdg_TS_T40D34M30I0R0\src\Swt_Ip.c
- ..\Wdg_TS_T40D34M30I0R0\include\Swt_Ip_Types.h
- ..\Wdg_TS_T40D34M30I0R0\src\Swt_Ip_Irq.c
- ..\Wdg_TS_T40D34M30I0R0\include\Wdg_EnvCfg.h
- ..\Wdg_TS_T40D34M30I0R0\include\Swt_Ip_FeatureDefines.h

Wdg Generated Files
- Wdg_43_Instance0_[VariantName]_PBcfg.c (For Instance 0) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective

variant.
• Wdg_43_Instance1_[VariantName]_PBcfg.c (For Instance 1) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Wdg_43_Instance2_[VariantName]_PBcfg.c (For Instance 2) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Wdg_43_Instance3_[VariantName]_PBcfg.c (For Instance 3) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Wdg_43_Instance4_[VariantName]_PBcfg.c (For Instance 4) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Wdg_43_Instance0_[VariantName]_PBcfg.h (For Instance 0) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Wdg_43_Instance1_[VariantName]_PBcfg.h (For Instance 1) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Wdg_43_Instance2_[VariantName]_PBcfg.h (For Instance 2) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Wdg_43_Instance3_[VariantName]_PBcfg.h (For Instance 3) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Wdg_43_Instance4_[VariantName]_PBcfg.h (For Instance 4) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Wdg_43_Instance0_Ipw_[VariantName]_PBcfg.c (For Instance 0) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Wdg_43_Instance1_Ipw_[VariantName]_PBcfg.c (For Instance 1) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Wdg_43_Instance2_Ipw_[VariantName]_PBcfg.c (For Instance 2) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Wdg_43_Instance3_Ipw_[VariantName]_PBcfg.c (For Instance 3) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Wdg_43_Instance4_Ipw_[VariantName]_PBcfg.c (For Instance 4) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Wdg_43_Instance0_Ipw_[VariantName]_PBcfg.h (For Instance 0) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Wdg_43_Instance1_Ipw_[VariantName]_PBcfg.h (For Instance 1) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Wdg_43_Instance2_Ipw_[VariantName]_PBcfg.h (For Instance 2) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Wdg_43_Instance3_Ipw_[VariantName]_PBcfg.h (For Instance 3) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective

variant.
• Wdg_43_Instance4_Ipw_[VariantName]_PBcfg.h (For Instance 4) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Swt_43_Instance0_Ip_[VariantName]_PBcfg.c (For Instance 0) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Swt_43_Instance1_Ip_[VariantName]_PBcfg.c (For Instance 1) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Swt_43_Instance2_Ip_[VariantName]_PBcfg.c (For Instance 2) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Swt_43_Instance3_Ip_[VariantName]_PBcfg.c (For Instance 3) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Swt_43_Instance4_Ip_[VariantName]_PBcfg.c (For Instance 4) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Swt_43_Instance0_Ip_[VariantName]_PBcfg.h (For Instance 0) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Swt_43_Instance1_Ip_[VariantName]_PBcfg.h (For Instance 1) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Swt_43_Instance2_Ip_[VariantName]_PBcfg.h (For Instance 2) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Swt_43_Instance3_Ip_[VariantName]_PBcfg.h (For Instance 3) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Swt_43_Instance4_Ip_[VariantName]_PBcfg.h (For Instance 4) - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Swt_Ip_Cfg_Defines.h - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Swt_Ip_DeviceRegisters.h For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.

• Wdg_Ipw_Cfg_Defines.h - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Wdg_Cfg_Defines.h - For driver compilation, this file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
• Wdg_Cfg.h - For driver compilation, this file should be generated by the user using a configuration tool
• Wdg_CfgExt.h - For driver compilation, this file should be generated by the user using a configuration tool
• Wdg_CfgExt.c - For driver compilation, this file should be generated by the user using a configuration tool

Files from Base common folder
• ..\Base_TS_T40D34M30I0R0\include\Compiler.h
• ..\Base_TS_T40D34M30I0R0\include\Compiler_Cfg.h
• ..\Base_TS_T40D34M30I0R0\include\ComStack_Types.h
• ..\Base_TS_T40D34M30I0R0\include\Wdg_MemMap.h
• ..\Base_TS_T40D34M30I0R0\include\Mcal.h

- ..\Base_TS_T40D34M30I0R0\include\Platform_Types.h
- ..\Base_TS_T40D34M30I0R0\include\Std_Types.h
- ..\Base_TS_T40D34M30I0R0\include\Reg_eSys.h
- ..\Base_TS_T40D34M30I0R0\include\Soc_Ips.h
- ..\Base_TS_T40D34M30I0R0\include\SilRegMacros.h

Files from WdgIf folder:
- ..\WdgIf_TS_T40D34M30I0R0\include\WdgIf_Types.h

Files from Dem folder:
- ..\Dem_TS_T40D34M30I0R0\include\Dem.h
- ..\Dem_TS_T40D34M30I0R0\include\Dem_IntErrId.h
- ..\Dem_TS_T40D34M30I0R0\include\Dem_Types.h

Files from Gpt folder:
- ..\Gpt_TS_T40D34M30I0R0\src\Gpt.c
- ..\Gpt_TS_T40D34M30I0R0\src\Rtc_Ip.c
- ..\Gpt_TS_T40D34M30I0R0\src\Gpt_Ipw.c
- ..\Gpt_TS_T40D34M30I0R0\src\Pit_Ip.c
- ..\Gpt_TS_T40D34M30I0R0\src\Stm_Ip.c
- ..\Gpt_TS_T40D34M30I0R0\src\Emios_Gpt_Ip.c
- ..\Gpt_TS_T40D34M30I0R0\include\Gpt.h
- ..\Gpt_TS_T40D34M30I0R0\include\Gpt_EnvCfg.h
- ..\Gpt_TS_T40D34M30I0R0\include\Rtc_Ip.h
- ..\Gpt_TS_T40D34M30I0R0\include\Rtc_Ip_Types.h
- ..\Gpt_TS_T40D34M30I0R0\include\Gpt_Ipw.h
- ..\Gpt_TS_T40D34M30I0R0\include\Gpt_Ipw_Irq.h
- ..\Gpt_TS_T40D34M30I0R0\include\Gpt_Ipw_Types.h
- ..\Gpt_TS_T40D34M30I0R0\include\Gpt_Irq.h
- ..\Gpt_TS_T40D34M30I0R0\include\Pit_Ip.h
- ..\Gpt_TS_T40D34M30I0R0\include\Pit_Ip_Types.h
- ..\Gpt_TS_T40D34M30I0R0\include\Emios_Gpt_Ip.h
- ..\Gpt_TS_T40D34M30I0R0\include\Emios_Gpt_Ip_Types.h
- ..\Gpt_TS_T40D34M30I0R0\include\Stm_Ip.h
- ..\Gpt_TS_T40D34M30I0R0\include\Stm_Ip_Types.h
- Gpt_Cfg.c (For PC Variant) - This file should be generated by the user using a configuration tool for compilation
- Gpt_PBCfg_VS_[index].c (For PB Variant, [index] is Variant Indexer) - This file should be generated by the user using a configuration tool for compilation

Files from Det folder:
- ..\Det_TS_T40D34M30I0R0\src\Det.c
- ..\Det_TS_T40D34M30I0R0\include\Det.h

Files from Os folder:
- ..\Os_TS_T40D34M30I0R0\src\Os_counter_api.c
- ..\Os_TS_T40D34M30I0R0\src\Os_multicore.c
- ..\Os_TS_T40D34M30I0R0\include\Os.h
- ..\Os_TS_T40D34M30I0R0\include\Os_multicore.h
- ..\Os_TS_T40D34M30I0R0\include\Os_counter_api.h

- ..\Os_TS_T40D34M30I0R0\include\Os_counter_types.h
- ..\Os_TS_T40D34M30I0R0\include\Os_types_basic.h
- ..\Os_TS_T40D34M30I0R0\include\Os_types_common_public.h
- ..\Os_TS_T40D34M30I0R0\include\Os_types_public.h
- ..\Os_TS_T40D34M30I0R0\include\Os_version.h
- ..\Os_TS_T40D34M30I0R0\generate_PC\include\Os_cfg.h

Files from Platform folder:
- ..\Platform_TS_T40D34M30I0R0\src\Platform.c
- ..\Platform_TS_T40D34M30I0R0\src\IntCtrl_Ip.c
- ..\Platform_TS_T40D34M30I0R0\include\Platform.h
- ..\Platform_TS_T40D34M30I0R0\include\IntCtrl_Ip.h

## 3.3 Setting up the plugins

The Wdg driver was designed to be configured by using the EB Tresos Studio (version 29.0.0 or later.)

Location of various files inside the plugin folder is explained below.
- VSMD (Vendor Specific Module Definition) file in EB tresos Studio XDM format:
- ..\Wdg_TS_T40D34M30I0R0\config\Wdg.xdm
- ..\Gpt_TS_T40D34M30I0R0\config\Gpt.xdm
- ..\Mcu_TS_T40D34M30I0R0\config\Mcu.xdm
- ..\EcuM_TS_T40D34M30I0R0\config\EcuM.xdm
- ..\EcuC_TS_T40D34M30I0R0\config\EcuC.xdm
- ..\Dem_TS_T40D34M30I0R0\config\Dem.xdm
- ..\Os_TS_T40D34M30I0R0\config\Os.xdm
- ..\Resource_TS_T40D34M30I0R0\config\Resource.xdm
- ..\Platform_TS_T40D34M30I0R0\config\Platform.xdm

- VSMD (Vendor Specific Module Definition) file(s) in AUTOSAR compliant EPD format:
- ..\Wdg_TS_T40D34M30I0R0\autosar\Wdg_<subderivative_name>.epd
- ..\Gpt_TS_T40D34M30I0R0\autosar\Gpt_<subderivative_name>.epd
- ..\Mcu_TS_T40D34M30I0R0\autosar\Mcu_<subderivative_name>.epd
- ..\EcuM_TS_T40D34M30I0R0\autosar\EcuM.epd
- ..\EcuC_TS_T40D34M30I0R0\autosar\EcuC.epd
- ..\Dem_TS_T40D34M30I0R0\autosar\Dem.epd
- ..\Os_TS_T40D34M30I0R0\autosar\Os.epd
- ..\Resource_TS_T40D34M30I0R0\autosar\Resource_<subderivative_name>.epd
- ..\Platform_TS_T40D34M30I0R0\autosar\Platform_<subderivative_name>.epd
- Code Generation Templates for Pre-Compile time configuration parameters:
- ..\Wdg_TS_T40D34M30I0R0\generate\src\Wdg_CfgExt.c
- ..\Wdg_TS_T40D34M30I0R0\generate\include\Wdg_CfgExt.h
- ..\Wdg_TS_T40D34M30I0R0\generate\include\Wdg_Cfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PC\src\Wdg_43_Instance0_Cfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PC\src\Wdg_43_Instance1_Cfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PC\src\Wdg_43_Instance2_Cfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PC\src\Wdg_43_Instance3_Cfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PC\src\Wdg_43_Instance4_Cfg.c
- ..\EcuM_TS_T40D34M30I0R0\generate_PC\include\EcuM_Cfg.h
- ..\Dem_TS_T40D34M30I0R0\generate_PC\include\Dem_intErrId.h

- ..\Gpt_TS_T40D34M30I0R0\generate_PC\include\Gpt_Cfg.h
- ..\Gpt_TS_T40D34M30I0R0\generate_PC\src\Gpt_Cfg.c
- ..\Os_TS_T40D34M30I0R0\generate_PC\include\Os_cfg.h
- ..\Platform_TS_T40D34M30I0R0\generate_PC\include\Platform_Cfg.h

- Code Generation Templates for Post-Build time configuration parameters:
- ..\Wdg_TS_T40D34M30I0R0\generate\src\Wdg_CfgExt.c
- ..\Wdg_TS_T40D34M30I0R0\generate\src\Wdg_CfgExt.h
- ..\Wdg_TS_T40D34M30I0R0\generate\include\Wdg_Cfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance0_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance1_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance2_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance3_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance4_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance0_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance1_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance2_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance3_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance4_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance0_Ipw_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance1_Ipw_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance2_Ipw_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance3_Ipw_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance4_Ipw_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance0_Ipw_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance1_Ipw_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance2_Ipw_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance3_Ipw_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance4_Ipw_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Swt_43_Instance0_Ip_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Swt_43_Instance1_Ip_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Swt_43_Instance2_Ip_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Swt_43_Instance3_Ip_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Swt_43_Instance4_Ip_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Swt_43_Instance0_Ip_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Swt_43_Instance1_Ip_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Swt_43_Instance2_Ip_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Swt_43_Instance3_Ip_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Swt_43_Instance4_Ip_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Swt_Ip_Cfg_Defines.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_Ipw_Cfg_Defines.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_Cfg_Defines.h
- ..\EcuM_TS_T40D34M30I0R0\generate_PC\include\EcuM_Cfg.h
- ..\Dem_TS_T40D34M30I0R0\generate_PC\include\Dem_intErrId.h
- ..\Gpt_TS_T40D34M30I0R0\generate_PC\include\Gpt_Cfg.h
- ..\Gpt_TS_T40D34M30I0R0\generate_PB\src\Gpt_PBCfg.c
- ..\Os_TS_T40D34M30I0R0\generate_PC\include\Os_cfg.h
- ..\Platform_TS_T40D34M30I0R0\generate\include\Platform_Cfg.h

- Code Generation Templates for parameters without variation points:
- ..\Wdg_TS_T40D34M30I0R0\generate\src\Wdg_CfgExt.c
- ..\Wdg_TS_T40D34M30I0R0\generate\include\Wdg_CfgExt.h

- ..\Wdg_TS_T40D34M30I0R0\generate\include\Wdg_Cfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance0_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance1_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance2_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance3_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance4_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance0_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance1_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance2_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance3_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance4_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance0_Ipw_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance1_Ipw_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance2_Ipw_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance3_Ipw_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance4_Ipw_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance0_Ipw_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance1_Ipw_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance2_Ipw_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance3_Ipw_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance4_Ipw_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Swt_43_Instance0_Ip_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Swt_43_Instance1_Ip_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Swt_43_Instance2_Ip_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Swt_43_Instance3_Ip_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Swt_43_Instance4_Ip_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Swt_43_Instance0_Ip_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Swt_43_Instance1_Ip_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Swt_43_Instance2_Ip_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Swt_43_Instance3_Ip_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Swt_43_Instance4_Ip_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Swt_Ip_Cfg_Defines.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_Ipw_Cfg_Defines.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_Cfg_Defines.h
- ..\EcuM_TS_T40D34M30I0R0\generate_PC\include\EcuM_Cfg.h
- ..\Dem_TS_T40D34M30I0R0\generate_PC\include\Dem_intErrId.h
- ..\Gpt_TS_T40D34M30I0R0\generate_PC\include\Gpt_Cfg.h
- ..\Os_TS_T40D34M30I0R0\generate_PC\include\Os_cfg.h
- ..\Platform_TS_T40D34M30I0R0\generate\include\Platform_Cfg.h
- Code Generation Templates for variant aware parameters:
- ..\Wdg_TS_T40D34M30I0R0\generate\src\Wdg_CfgExt.c
- ..\Wdg_TS_T40D34M30I0R0\generate\include\Wdg_CfgExt.h
- ..\Wdg_TS_T40D34M30I0R0\generate\include\Wdg_Cfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance0_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance1_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance2_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance3_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance4_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance0_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance1_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance2_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance3_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance4_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance0_Ipw_PBcfg.h

- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance1_Ipw_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance2_Ipw_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance3_Ipw_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_43_Instance4_Ipw_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance0_Ipw_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance1_Ipw_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance2_Ipw_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance3_Ipw_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Wdg_43_Instance4_Ipw_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Swt_43_Instance0_Ip_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Swt_43_Instance1_Ip_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Swt_43_Instance2_Ip_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Swt_43_Instance3_Ip_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Swt_43_Instance4_Ip_PBcfg.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Swt_43_Instance0_Ip_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Swt_43_Instance1_Ip_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Swt_43_Instance2_Ip_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Swt_43_Instance3_Ip_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\src\Swt_43_Instance4_Ip_PBcfg.c
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Swt_Ip_Cfg_Defines.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_Ipw_Cfg_Defines.h
- ..\Wdg_TS_T40D34M30I0R0\generate_PB\include\Wdg_Cfg_Defines.h
- ..\Gpt_TS_T40D34M30I0R0\generate_PC\include\Gpt_Cfg.h
- ..\Gpt_TS_T40D34M30I0R0\generate_PB\src\Gpt_PBCfg.c
- ..\Os_TS_T40D34M30I0R0\generate_PC\include\Os_cfg.h
- ..\Platform_TS_T40D34M30I0R0\generate\include\Platform_Cfg.h

Steps to generate the configuration:

1. Copy the module folders Wdg_TS_T40D34M30I0R0 , Dem_TS_T40D34M30I0R0 , Base_TS_T40D34↩ M30I0R0 , Resource_TS_T40D34M30I0R0 , EcuM_TS_T40D34M30I0R0, Gpt_TS_T40D34M30I0R0, Mcl_TS_T40D34M30I0R0, Mcu_TS_T40D34M30I0R0, Os_TS_T40D34M30I0R0, Platform_TS_T40↩ D34M30I0R0, Ae_TS_T40D34M30I0R0, Spi_TS_T40D34M30I0R0 into the Tresos plugins folder.

2. Set the desired Tresos Output location folder for the generated sources and header files.

3. Use the EB tresos Studio GUI to modify ECU configuration parameters values.

4. Generate the configuration files.
   Dependencies

   - RESOURCE is required to select processor derivative. Current driver has support for the following derivatives, each one having attached a Resource file: s32k344_mapbga257, s32k344_mqfp172, s32k358_↩ mapbga289, s32k358_mqfp172, s32k396_mapbga289, s32m276_lqfp64.
   - Base is required for platform specific files.
   - DET is required for signaling the development error detection (parameters out of range, null pointers, etc).
   - DEM is required for signaling the production error detection (hardware failure, etc).
   - WdgIf is required for retrieve the watchdog mode types

- GPT is required for handling the watchdog internal timer
- MCU is required for selecting the timebase for the watchdog internal timer, via Gpt
- MCL is required by Gpt to retrieve the timer specific files
- RTE is required for critical sections
- ECUM is required for selecting the reference to the wakeup source for every Gpt channel configured as a wakeup source.
- ECUC is required for selecting variant.
- OS is required for selecting core.
- Platform is required for selecting interrupt for GPT and WDG .
- Ae and Spi are required for configurating for AEWDOG .

# Chapter 4

# Function calls to module



## 4.1 Function Calls during Start-up

Wdg shall be initialized during STARTUP phase of EcuM initialization. The API to be called for this is Wdg_↩ Init(). The MCU and Gpt module should be initialized before the Wdg is initialized. Note : If there are multiple WDG hardware instances used on the platform, the API names will expand according to AUTOSAR requirement SRS_BSW_00347. For example, if there are instances 0 and 1 available on the hardware,then the name of the init functions will be Wdg_43_Instance0_Init and Wdg_43_Instance1_Init instead of Wdg_Init().

## 4.2 Function Calls during Shutdown

None.

## 4.3 Function Calls during Wake-up

None.

# Chapter 5

# Module requirements

- Exclusive areas to be defined in BSW scheduler
- Exclusive areas not available on this platform
- Peripheral Hardware Requirements
- ISR to configure within AutosarOS - dependencies
- ISR Macro
- Other AUTOSAR modules - dependencies
- Data Cache Restrictions
- User Mode support
- Multicore support

## 5.1 Exclusive areas to be defined in BSW scheduler

In the current implementation, WDG is using the services of Schedule Manager (SchM) for entering and exiting the exclusive areas. The following critical regions are used in the WDG driver:

**Exclusive Areas are used in High level driver layer (HLD)**

**WDG_EXCLUSIVE_AREA_00** is used in function Wdg_Cbk_GptNotification0 to protect the updates for the global array:

- Wdg_au32Timeout[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_00** is used in function Wdg_Cbk_GptNotification1 to protect the updates for the global array:

- Wdg_au32Timeout[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_00** is used in function Wdg_Cbk_GptNotification2 to protect the updates for the global array:

- Wdg_au32Timeout[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_00** is used in function Wdg_Cbk_GptNotification3 to protect the updates for the global array:

- Wdg_au32Timeout[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_00** is used in function Wdg_Cbk_GptNotification4 to protect the updates for the global array:

- Wdg_au32Timeout[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_01** is used in function Wdg_43_Instance0_Init to protect the updates for the global array:

- Wdg_au32Timeout[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_01** is used in function Wdg_43_Instance1_Init to protect the updates for the global array:

- Wdg_au32Timeout[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_01** is used in function Wdg_43_Instance2_Init to protect the updates for the global array:

- Wdg_au32Timeout[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_01** is used in function Wdg_43_Instance3_Init to protect the updates for the global array:

- Wdg_au32Timeout[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_01** is used in function Wdg_43_Instance4_Init to protect the updates for the global array:

- Wdg_au32Timeout[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_02** is used in function Wdg_43_Instance0_Init to protect the updates for the global array:

- Wdg_aePreviousMode[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_02** is used in function Wdg_43_Instance1_Init to protect the updates for the global array:

- Wdg_aePreviousMode[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_02** is used in function Wdg_43_Instance2_Init to protect the updates for the global array:

- Wdg_aePreviousMode[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_02** is used in function Wdg_43_Instance3_Init to protect the updates for the global array:

- Wdg_aePreviousMode[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_02** is used in function Wdg_43_Instance4_Init to protect the updates for the global array:

- Wdg_aePreviousMode[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_03** is used in function Wdg_43_Instance0_SetMode to protect the updates for the global array:

- Wdg_au32Timeout[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_03** is used in function Wdg_43_Instance1_SetMode to protect the updates for the global array:

- Wdg_au32Timeout[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_03** is used in function Wdg_43_Instance2_SetMode to protect the updates for the global array:

- Wdg_au32Timeout[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_03** is used in function Wdg_43_Instance3_SetMode to protect the updates for the global array:

- Wdg_au32Timeout[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_04** is used in function Wdg_43_Instance0_SetMode to protect the updates for the global array:

- Wdg_aePreviousMode[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_04** is used in function Wdg_43_Instance1_SetMode to protect the updates for the global array:

- Wdg_aePreviousMode[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_04** is used in function Wdg_43_Instance2_SetMode to protect the updates for the global array:

- Wdg_aePreviousMode[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_04** is used in function Wdg_43_Instance3_SetMode to protect the updates for the global array:

- Wdg_aePreviousMode[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_05** is used in function Wdg_43_Instance0_SetTriggerCondition to protect the updates for the global array:

- Wdg_au32Timeout[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_05** is used in function Wdg_43_Instance1_SetTriggerCondition to protect the updates for the global array:

- Wdg_au32Timeout[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_05** is used in function Wdg_43_Instance2_SetTriggerCondition to protect the updates for the global array:

- Wdg_au32Timeout[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_05** is used in function Wdg_43_Instance3_SetTriggerCondition to protect the updates for the global array:

- Wdg_au32Timeout[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_05** is used in function Wdg_43_Instance4_SetTriggerCondition to protect the updates for the global array:

- Wdg_au32Timeout[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_06** is used in function Wdg_43_Instance0_Init to protect the updates for the global array:

- Wdg_aeStatus[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_06** is used in function Wdg_43_Instance1_Init to protect the updates for the global array:

- Wdg_aeStatus[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_06** is used in function Wdg_43_Instance2_Init to protect the updates for the global array:

- Wdg_aeStatus[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_06** is used in function Wdg_43_Instance3_Init to protect the updates for the global array:

- Wdg_aeStatus[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_06** is used in function Wdg_43_Instance4_Init to protect the updates for the global array:

- Wdg_aeStatus[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_06** is used in function Wdg_43_Instance0_SetMode to protect the updates for the global array:

- Wdg_aeStatus[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_06** is used in function Wdg_43_Instance1_SetMode to protect the updates for the global array:

- Wdg_aeStatus[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_06** is used in function Wdg_43_Instance2_SetMode to protect the updates for the global array:

- Wdg_aeStatus[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_06** is used in function Wdg_43_Instance3_SetMode to protect the updates for the global array:

- Wdg_aeStatus[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_07** is used in function Wdg_43_Instance0_Init to protect the updates for the global array:

- Wdg_aeStatus[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_07** is used in function Wdg_43_Instance1_Init to protect the updates for the global array:

- Wdg_aeStatus[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_07** is used in function Wdg_43_Instance2_Init to protect the updates for the global array:

- Wdg_aeStatus[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_07** is used in function Wdg_43_Instance3_Init to protect the updates for the global array:

- Wdg_aeStatus[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_07** is used in function Wdg_43_Instance4_Init to protect the updates for the global array:

- Wdg_aeStatus[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_07** is used in function Wdg_43_Instance0_SetMode to protect the updates for the global array:

- Wdg_aeStatus[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_07** is used in function Wdg_43_Instance1_SetMode to protect the updates for the global array:

- Wdg_aeStatus[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_07** is used in function Wdg_43_Instance2_SetMode to protect the updates for the global array:

- Wdg_aeStatus[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_07** is used in function Wdg_43_Instance3_SetMode to protect the updates for the global array:

- Wdg_aeStatus[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_08** is used in function Wdg_43_Instance0_ClearResetRequest to protect the updates for the global array:

- Wdg_aePreviousMode[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_08** is used in function Wdg_43_Instance1_ClearResetRequest to protect the updates for the global array:

- Wdg_aePreviousMode[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_08** is used in function Wdg_43_Instance2_ClearResetRequest to protect the updates for the global array:

- Wdg_aePreviousMode[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_08** is used in function Wdg_43_Instance3_ClearResetRequest to protect the updates for the global array:

- Wdg_aePreviousMode[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_08** is used in function Wdg_43_Instance4_ClearResetRequest to protect the updates for the global array:

- Wdg_aePreviousMode[Wdg_Instance]

**WDG_EXCLUSIVE_AREA_09** is used in function Wdg_43_Instance0_Service to protect the SWT_SR register from read/modify/write operation in Swt_Ip_Service.

**WDG_EXCLUSIVE_AREA_09** is used in function Wdg_43_Instance1_Service to protect the SWT_SR register from read/modify/write operation in Swt_Ip_Service.

**WDG_EXCLUSIVE_AREA_09** is used in function Wdg_43_Instance2_Service to protect the SWT_SR register from read/modify/write operation in Swt_Ip_Service.

**WDG_EXCLUSIVE_AREA_09** is used in function Wdg_43_Instance3_Service to protect the SWT_SR register from read/modify/write operation in Swt_Ip_Service.

**WDG_EXCLUSIVE_AREA_09** is used in function Wdg_43_Instance4_Service to protect the SWT_SR register from read/modify/write operation in Swt_Ip_Service.

**WDG_EXCLUSIVE_AREA_10** is used in function Wdg_43_Instance0_SetMode to protect the SWT_SR register from read/modify/write operation in Swt_Ip_StartTimer.

**WDG_EXCLUSIVE_AREA_10** is used in function Wdg_43_Instance1_SetMode to protect the SWT_SR register from read/modify/write operation in Swt_Ip_StartTimer.

**WDG_EXCLUSIVE_AREA_10** is used in function Wdg_43_Instance2_SetMode to protect the SWT_SR register from read/modify/write operation in Swt_Ip_StartTimer.

**WDG_EXCLUSIVE_AREA_10** is used in function Wdg_43_Instance3_SetMode to protect the SWT_SR register from read/modify/write operation in Swt_Ip_StartTimer.

**WDG_EXCLUSIVE_AREA_11** is used in function Wdg_43_Instance0_SetMode to protect the SWT_SR register from read/modify/write operation in Swt_Ip_StopTimer.

**WDG_EXCLUSIVE_AREA_11** is used in function Wdg_43_Instance1_SetMode to protect the SWT_SR register from read/modify/write operation in Swt_Ip_StopTimer.

**WDG_EXCLUSIVE_AREA_11** is used in function Wdg_43_Instance2_SetMode to protect the SWT_SR register from read/modify/write operation in Swt_Ip_StopTimer.

**WDG_EXCLUSIVE_AREA_11** is used in function Wdg_43_Instance3_SetMode to protect the SWT_SR register from read/modify/write operation in Swt_Ip_StopTimer.

**Exclusive Areas implemented in Low level driver layer (IPL)**

**WDG_EXCLUSIVE_AREA_09** is used in function Swt_Ip_Service to to protect the updates for:

- SWT_SR register

**WDG_EXCLUSIVE_AREA_10** is used in function Swt_Ip_StartTimer to to protect the updates for:

- SWT_SR register

**WDG_EXCLUSIVE_AREA_11** is used in function Swt_Ip_StopTimer to protect the updates for the function:

- SWT_SR register

**Critical Region Exclusive Matrix**
Below is the table depicting the exclusivity between different critical region IDs from the WDG driver. If there is an X in the table, it means that those 2 critical regions cannot interrupt each other.

| Exclusive Area Matrix | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Exclusive Area ID | WDG_EXCLUSIVE_AREA_00 | WDG_EXCLUSIVE_AREA_01 | WDG_EXCLUSIVE_AREA_02 | WDG_EXCLUSIVE_AREA_03 | WDG_EXCLUSIVE_AREA_04 | WDG_EXCLUSIVE_AREA_05 | WDG_EXCLUSIVE_AREA_06 | WDG_EXCLUSIVE_AREA_07 | WDG_EXCLUSIVE_AREA_08 | WDG_EXCLUSIVE_AREA_09 | WDG_EXCLUSIVE_AREA_10 | WDG_EXCLUSIVE_AREA_11 | |
| WDG_EXCLUSIVE_AREA_00 | x | x | | x | | x | | | | | | | |
| WDG_EXCLUSIVE_AREA_01 | x | x | | x | | x | | | | | | | |
| WDG_EXCLUSIVE_AREA_02 | | | x | | x | | | | x | | | | |
| WDG_EXCLUSIVE_AREA_03 | x | x | | x | | x | | | | | | | |
| WDG_EXCLUSIVE_AREA_04 | | | x | | x | | | | | | | | |
| WDG_EXCLUSIVE_AREA_05 | x | x | | x | | x | | | | | | | |
| WDG_EXCLUSIVE_AREA_06 | | | | | | | x | x | | | | | |
| WDG_EXCLUSIVE_AREA_07 | | | | | | | x | x | | | | | |
| WDG_EXCLUSIVE_AREA_08 | | | x | | | | | | x | | | | |
| WDG_EXCLUSIVE_AREA_09 | | | | | | | | | | x | x | x | |
| WDG_EXCLUSIVE_AREA_10 | | | | | | | | | | x | x | x | |
| WDG_EXCLUSIVE_AREA_11 | | | | | | | | | | x | x | x | |

## 5.2 Exclusive areas not available on this platform

None.

## 5.3 Peripheral Hardware Requirements

A Software watchdog timer (SWT) with programmable interrupt response is available in S32K3XX. There are three user-defined responses to a time-out:
- If a time-out occurs, the SWT generates an interrupt to the processor core.
- The first time-out generates an interrupt to the processor, and if not serviced, then a second time-out generates a system reset and sets the MC_RGM[FES] flag.
- If a time-out occurs, the SWT generates a system reset and sets the MC_RGM[FES] flag.

In addition to these three modes of operation, the watchdog timer also supports a windowed mode. In this mode, the service sequence must be performed in the last part of the time-out period defined by the window register. The window is open when the down counter is less than the value in the SWT_WN register. Outside of this window, service sequence writes are invalid accesses and generate a bus error or reset depending on the value of the SWT↩ _CR.RIA

## 5.4 ISR to configure within AutosarOS - dependencies

The following ISR's are used by the WDG driver:

| ISR Name | HW INT Vector | Observations |
|---|---|---|
| ISR(Swt_Ip_Swt0_Isr) | 42 | None |
| ISR(Swt_Ip_Swt1_Isr) | 43 | None |
| ISR(Swt_Ip_Swt2_Isr) | 44 | None |

## 5.5 ISR Macro

RTD drivers use the ISR macro to define the functions that will process hardware interrupts. Depending on whether the OS is used or not, this macro can have different definitions.

### 5.5.1 Without an Operating System

The macro `_USING_OS_AUTOSAROS_` must not be defined.

#### 5.5.1.1 Using Software Vector Mode

The macro `_USE_SW_VECTOR_MODE_` must be defined and the ISR macro is defined as:

```
#define ISR(IsrName) void IsrName(void)
```

In this case, the drivers' interrupt handlers are normal C functions and their prologue/epilogue will handle the context save and restore.

#### 5.5.1.2 Using Hardware Vector Mode

The macro _USE_SW_VECTOR_MODE_ must not defined and the ISR macro is defined as:

#define ISR(IsrName) INTERRUPT_FUNC void IsrName(void)

In this case, the drivers' interrupt handlers must also handle the context save and restore.

### 5.5.2 With an Operating System

Please refer to your OS documentation for description of the ISR macro.

## 5.6 Other AUTOSAR modules - dependencies

Dependencies:
• WDGIF: The Watchdog Interface module is necessary for importing the Mode type in which the watchdog can configured.
• BASE: The BASE module contains the common files/definitions needed by all RTD modules.
• DEM: The DEM module is used for enabling reporting of production relevant error status. The API function used is Dem_SetEventStatus().
• DET: The DET module is used for enabling Development error detection. The API function used is Det_Report↩ Error(). The activation/deactivation of Development Error Detection is configurable using the 'WdgDevErrorDetect' configuration parameter.
• ECUC: The ECUC module is used for ECU configuration. RTD modules need ECUC to retrieve the variant information.
• MCU: The MCU driver provides services for basic microcontroller initialization, power down functionality, reset and microcontroller specific functions required by other RTD software modules. The clocks need to be initialized prior to using the Wdg driver.
• GPT: The GPT driver is required to periodically reset the watchdog timeout counter.
• MCL: The MCL driver is a dependency for certain Gpt driver functions.
• RESOURCE: The RESOURCE module is used to select microcontroller's derivatives.
Current driver has support for the following derivatives, each having attached a Resource file: s32k344_mapbga257, s32k344_mqfp172, s32k358_mapbga289, s32k358_mqfp172, s32k396_mapbga289, s32m276_lqfp64.
• RTE: The RTE module is needed for implementing data consistency of exclusive areas that are used in the Wdg driver.

## 5.7 Data Cache Restrictions

None.

## 5.8 User Mode support

- User Mode configuration in the module
- User Mode configuration in AutosarOS

### 5.8.1 User Mode configuration in the module

Wdg driver can not run in user mode.

### 5.8.2 User Mode configuration in AutosarOS

When User mode is enabled, the driver may has the functions that need to be called as trusted functions in AutosarOS context. Those functions are already defined in driver and declared in the header <IpName>_Ip_TrustedFunctions.h. This header also included all headers files that contains all types definition used by parameters or return types of those functions. Refer the chapter User Mode configuration in the module for more detail about those functions and the name of header files they are declared inside. Those functions will be called indirectly with the naming convention below in order to AutosarOS can call them as trusted functions.

`Call_<Function_Name>_TRUSTED(parameter1,parameter2,...)`

That is the result of macro expansion `OsIf_Trusted_Call` in driver code:

`#define` OsIf_Trusted_Call[1-6params](name,param1,...,param6) Call_##name##_TRUSTED(param1,...,param6)

So, the following steps need to be done in AutosarOS:

- Ensure `MCAL_ENABLE_USER_MODE_SUPPORT` macro is defined in the build system or somewhere global.
- Define and declare all functions that need to call as trusted functions follow the naming convention above in Integration/User code. They need to visible in `Os.h` for the driver to call them. They will do the marshalling of the parameters and call `CallTrustedFunction()` in OS specific manner.
- `CallTrustedFunction()` will switch to privileged mode and call `TRUSTED_<Function_Name>()`.
- `TRUSTED_<Function_Name>()` function is also defined and declared in Integration/User code. It will unmarshalling of the parameters to call <Function_Name>() of driver. The <Function_Name>() functions are already defined in driver and declared in <IpName>_Ip_TrustedFunctions.h. This header should be included in OS for OS call and indexing these functions.

See the sequence chart below for an example calling `Linflexd_Uart_Ip_Init_Privileged()` as a trusted function.

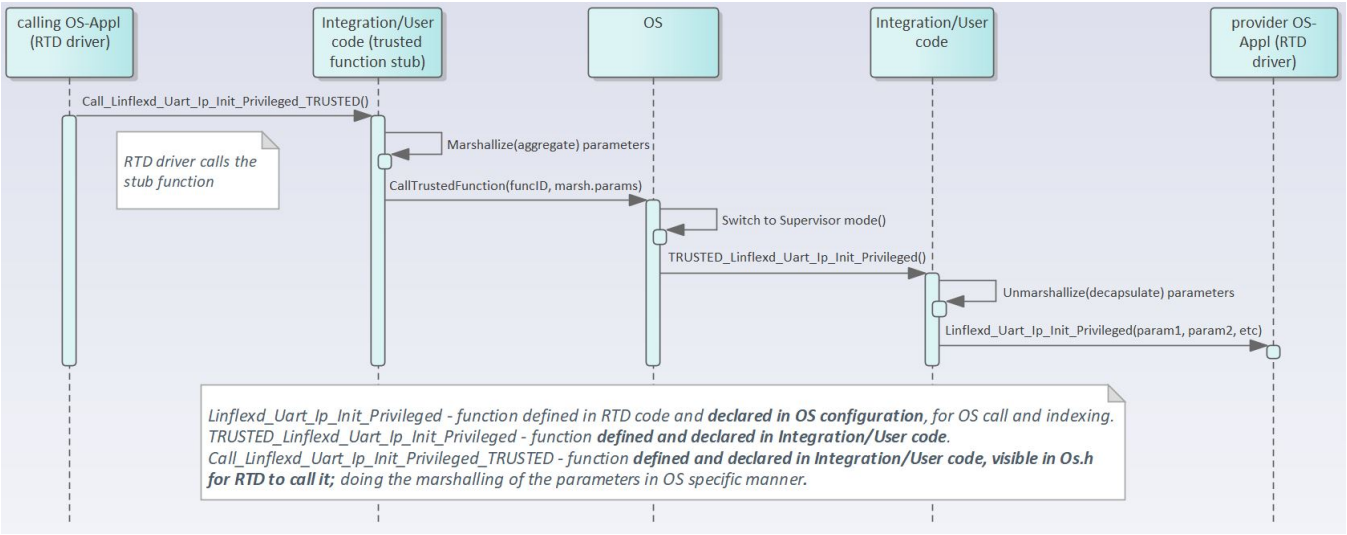


**Figure 5.1 Example sequence chart for calling `Linflexd_Uart_Ip_Init_Privileged` as trusted function**

## 5.9 Multicore support

The **Wdg** implements the "Autosar 4.4 RTD Multicore Distribution" according to type II, in which the mappable element is set to channel . For additional details, please refer to **AUTOSAR_EXP_BSWDistributionGuide**.

The **Wdg** and the mappable elements can only be allocated one ECUC partitions, by means of "WdgEcucPartition↩ Ref". If there is only one instance of Wdg is used, the **Wdg** behavior reverts to single-core implementation, similar to previous Autosar versions. If the number of **Wdg** is more than one, the **Wdg** enforces the following multicore assumptions:
**1.** The **Wdg** assumes there is a single EcucPartition allocated per core. Internally, the module will use the Core ID returned by GetCoreID API to reference the appropriate global data and configuration elements.
**2.** The **Wdg** assumes the EcucCoreIDs are defined in a compact/consecutive order, starting from zero. The rationale is that the number of EcucPartitions is used for dimensioning the Wdg internal variables and the EcucCoreIDs are used for indexing those variables.
**3.** The **Wdg** assumes that initialization is performed on each core, Wdg_43_InstanceX_Init() is called separately for each core, using a different configuration structure. ( Type II).
**4.** The **Wdg** initialization expects the upper layer will pass the correct initialization pointer, specific to the partition in which the driver is to be used. For example: EcucPartition_1 is assigned to CoreID 1; Wdg_43_Instance1_Init function will be called with configuration structure init, on Core 1.
**5.** The **Wdg** will check upon each API call if the requested resource is configured to be available on the current core, if DET error reporting is enabled.
**6.** The **Wdg** assumes that RTE module implements the EXCLUSIVE AREAS to be coreaware only. The rationale is that the module implementation ensures data integrity by separating the mappable elements for different cores already, thus implementing the EXCLUSIVE AREAS in a blocking manner (ex: spin-lock) on a multicore scope, might affect the performance of the drivers on the two cores, although they might access separate HW elements. For single-core scope, the EXCLUSIVE AREAS keep the same purpose as on previous AUTOSAR implementations. (to be updated per Wdg usecase, to be detailed/removed if some modules require such kind of functionality for critical features which cannot be atomically shared among cores).
**7.** The **Wdg** assumes that each interrupt is routed by the system only to the core on which is supposed to be serviced.

Note : X is the number of instance is used (Wdg_43_InstanceX_Init)

# Chapter 6

# Main API Requirements



## 6.1 Main function calls within BSW scheduler

None.

## 6.2 API Requirements

None.

## 6.3 Calls to Notification Functions, Callbacks, Callouts

Call-back Notifications: In Gpt Triggered Mode, a Gpt callback notification will be configured, which will periodically trigger the Wdg servicing routine. This notification will be named Wdg_Cbk_GptNotificationX, depending on the Wdg instance number "X" used.
User Notification:
The WDG Driver provides a notification that is called whenever the defined time period is over. The notifications can be configured as pointers to user defined functions. If notification is not desired, NULL_PTR shall be configured.
An example of the syntax of this function is as follows:
void Wdg_Notification (void)
The function has to be implemented by the user.

# Chapter 7

# Memory allocation

- Sections to be defined in Wdg_MemMap.h
- Linker command file

## 7.1 Sections to be defined in Wdg_MemMap.h

| Section name | Type of section | Description |
|---|---|---|
| WDG_START_SEC_CONST_8 | Configuration Data | Start of Memory Section for Config Data |
| WDG_STOP_SEC_CONST_8 | Configuration Data | End of Memory Section for Config Data. |
| WDG_START_SEC_CONST_32 | Configuration Data | Start of Memory Section for Config Data. |
| WDG_STOP_SEC_CONST_32 | Configuration Data | End of Memory Section for Config Data. |
| WDG_START_SEC_CONST_↩ UNSPECIFIED | Configuration Data | Start of Memory Section for Config Data. |
| WDG_STOP_SEC_CONST_↩ UNSPECIFIED | Configuration Data | End of Memory Section for Config Data. |
| WDG_START_SEC_CONFIG_DATA↩ _UNSPECIFIED | Configuration Data | Start of Memory Section for Config Data. |
| WDG_STOP_SEC_CONFIG_DATA_↩ UNSPECIFIED | Configuration Data | End of Memory Section for Config Data. |
| WDG_START_SEC_CODE | Code | Start of memory Section for Code in flash. |
| WDG_STOP_SEC_CODE | Code | Stop of memory Section for Code in flash. |
| WDG_START_SEC_RAMCODE | Code | Start of memory Section for Code in ram. |
| WDG_STOP_SEC_RAMCODE | Code | Stop of memory Section for Code in ram. |
| WDG_START_SEC_VAR_INIT_↩ UNSPECIFIED | Variables | Used for variables, structures, arrays, when the SIZE (alignment) does not fit the criteria of 8,16 or 32 bit. These variables are initialized with values after every reset |
| WDG_STOP_SEC_VAR_INIT_↩ UNSPECIFIED | Variables | End of above section. |
| WDG_START_SEC_VAR_INIT_16 | Variables | Used for variables which have to be aligned to 16 bit. For instance used for variables of size 16 bit or used for composite data types: arrays, structs containing elements of maximum 16 bits. These variables are initialized with values after every reset |

| Section name | Type of section | Description |
|---|---|---|
| WDG_STOP_SEC_VAR_INIT_16 | Variables | End of above section. |
| WDG_START_SEC_VAR_INIT_32 | Variables | Used for variables which have to be aligned to 32 bit. For instance used for variables of size 32 bit or used for composite data types: arrays, structs containing elements of maximum 32 bits. These variables are initialized with values after every reset |
| WDG_STOP_SEC_VAR_INIT_32 | Variables | End of above section. |
| WDG_START_SEC_VAR_CLEARED↩ _UNSPECIFIED | Variables | Used for variables, structures, arrays when the SIZE (alignment) does not fit the criteria of 8,16 or 32 bit. These variables are cleared to zero by start-up code (BBS). |
| WDG_STOP_SEC_VAR_CLEARED_↩ UNSPECIFIED | Variables | End of above section. |

## 7.2 Linker command file

Memory shall be allocated for every section defined in the driver's "<Module>"_MemMap.h.

# Chapter 8

# Integration Steps

This section gives a brief overview of the steps needed for integrating this module:

1. Generate the required module configuration(s). For more details refer to section Files Required for Compilation
2. Allocate the proper memory sections in the driver's memory map header file ("<Module>"_MemMap.h) and linker command file. For more details refer to section Sections to be defined in <Module>_MemMap.h
3. Compile & build the module with all the dependent modules. For more details refer to section Building the Driver

# Chapter 9

# External assumptions for driver

The section presents requirements that must be complied with when integrating the WDG driver into the application.

| External Assumption Req ID | External Assumption Text |
|---|---|
| SWS_Wdg_00144 | The Wdg Manager (or other entities) shall control the watchdog driver via a so called trigger condition: as long as the trigger condition is valid the Wdg Driver services the watchdog hardware, if the trigger condition becomes invalid the Wdg Driver stops triggering and the watchdog expires. The semantics of the trigger condition can be interpreted as a "permission to service the watchdog for the next n milliseconds". Within this time frame the trigger condition has to be updated by the controlling entity else the watchdog will expire. Handover of the watchdog control logic is simply done by shared usage of the trigger condition (e.g. during startup / shutdown). |
| EA_RTD_00053 | The application shall ensure that Wdg_SetTriggerCondition() function is not preempting itself or any of the following WDG functions: - Wdg_Init() - Wdg_SetMode() |
| EA_RTD_00055 | If WdgRunArea is set to RAM, then the application shall execute all the code which interacts with WDG from RAM (this means also at least DET, DEM, Gpt, SchM,application code will be executed from RAM). Note↩ : Motivation 1: Except the boot loader use case when entire software may run from RAM, there is no other obvious use case when WDG should run from RAM , especially a use case when WDG should run from RAM and other modules from ROM. |
| EA_RTD_00056 | If WdgRunArea is set to ROM, then the application shall execute all the code which interacts with WDG from ROM (this means also at least DET, DEM, Gpt, SchM,application code will be executed from ROM). |
| EA_RTD_00071 | If interrupts are locked, a centralized function pair to lock and unlock interrupts shall be used. |
| EA_RTD_00079 | It shall be the integrator responsibility to configure the additional hardware resources (i.e. FCCU) to handle the signals from watchdog modules to generate either an interrupt or a reset. |
| EA_RTD_00082 | When caches are enabled and data buffers are allocated in cacheable memory regions the buffers involved in DMA transfer shall be aligned with both start and end to cache line size. Note: **Rationale**: This ensures that no other buffers/variables compete for the same cache lines. |

| External Assumption Req ID | External Assumption Text |
|---|---|
| EA_RTD_00092 | The integrator shall allocate a single EcucPartition per core or the partition in which the Wdg is allocated shall be exclusively mapped to a core. Note: Internally, the Wdg will use the Core ID returned by GetCoreID API to reference the appropriate global data and configuration elements, that is why a core should reference only one configured partition. |
| EA_RTD_00093 | The application shall define EcucCoreIDs in a compact/consecutive order, starting from zero. |
| EA_RTD_00094 | When multicore support is enabled, the application shall call Wdg_Init() for each core, using the dedicated configuration pointer for that core. |
| EA_RTD_00096 | The application shall pass the correct initialization pointer, specific to the partition in which the driver is to be used. |
| EA_RTD_00097 | The application, after a successful call of Wdg_ClearResetRequest, shall set the watchdog into an active mode using Wdg_SetMode. |
| EA_RTD_00106 | Standalone IP configuration and HL configuration of the same driver shall be done in the same project |
| EA_RTD_00107 | The integrator shall use the IP interface only for hardware resources that were configured for standalone IP usage. Note:• The integrator shall not directly use the IP interface for hardware resources that were allocated to be used in HL context. |
| EA_RTD_00108 | The integrator shall use the IP interface to a build a CDD, therefore the BSWMD will not contain reference to the IP interface |
| EA_RTD_00113 | When RTD drivers are integrated with AutosarOS and User mode support is enabled, the integrator shall assure that the definition and declaration of all RTD functions needed to be called as trusted functions follow the naming convention Call<Function_Name>TRUSTED(parameter1,parameter2,...) in Integration/User code. They need to visible in Os.h for the driver to call them. They will call RTD <Function_Name>() as trusted functions in OS specific manner. |

***How to Reach Us:***

**Home Page:**
nxp.com

**Web Support:**
nxp.com/support