Integration Manual

for S32K14X WDG Driver

Document Number: IM2WDGASR4.3 Rev0001R1.0.1
Rev. 1.0

Contents

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

Chapter 1
Revision History

Table 1-1. Revision History

Revision	Date	Author	Description
1.0	21/06/2019	NXP MCAL Team	Updated version for ASR 4.3.1S32K14XR1.0.1

Chapter 2
Introduction

This integration manual describes the integration requirements for Wdg Driver for S32K14X microcontrollers.

2.1 Supported Derivatives

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

Table 2-1. S32K14X Derivatives

NXP Semiconductors	s32k148_lqfp144, s32k148_lqfp176, s32k148_mapbga100, s32k146_lqfp144, s32k146_lqfp100, s32k146_lqfp64, s32k146_mapbga100, s32k144_lqfp100, s32k144_lqfp64, s32k144_mapbga100, s32k142_lqfp100, s32k142_lqfp64, s32k118_lqfp48, s32k118_lqfp64, s32k142_lqfp48, s32k144_lqfp48, s32k148_lqfp100

All of the above microcontroller devices are collectively named as S32K14X .

2.2 Overview

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

AUTOSAR

- paves the way for innovative electronic systems that further improve performance, safety and environmental friendliness.

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

2.3 About this Manual

This Technical Reference employs the following typographical conventions:

Boldface type: Bold is used for important terms, notes and warnings.

Italic font: Italic typeface is used for code snippets in the text. Note that C language modifiers such "const" or "volatile" are sometimes omitted to improve readability of the presented code.

Notes and warnings are shown as below:

Note

This is a note.

2.4 Acronyms and Definitions

Table 2-2. Acronyms and Definitions

Abbreviation and Definitions	Description
BSW	Basic Software
DEM	Diagnostic Event Manager
DET	Development Error Tracer
ECU	Electronic Control Unit
WDG	Watchdog
MCU	MicroController Unit
MCL	MicroController Library
GPT	General Purpose Timers
ISR	interrupt Service Routine
OS	Operating System
RAM	Random Access Memory
ROM	Read-only Memory

Table continues on the next page...

Table 2-2. Acronyms and Definitions (continued)

Abbreviation and Definitions	Description
GUI	Graphical User Interface
EcuM	ECU state Manager
API	Application Programming Interface
PB Variant	Post Build Variant
PC Variant	Pre Compile Variant

2.5 Reference List

Table 2-3. Reference List

#	Title	Version
1	Specification of Wdg Driver	AUTOSAR Release 4.3.1
2	S32K14X Reference Manual	Reference Manual, Rev. 9, 9/2018
3	S32K142 Mask Set Errata for Mask 0N33V (0N33V)	30/11/2017
4	S32K144 Mask Set Errata for Mask 0N57U (0N57U)	30/11/2017
5	S32K146 Mask Set Errata for Mask 0N73V (0N73V)	30/11/2017
6	S32K148 Mask Set Errata for Mask 0N20V (0N20V)	25/10/2018
7	S32K118 Mask Set Errata for Mask 0N97V (0N97V)	07/01/2019

Chapter 3
Building the Driver

This section describes the source files and various compilers, linker options used for building the Autosar Wdg driver for NXP SemiconductorsS32K14X . It also explains the EB Tresos Studio plugin setup procedure.

3.1 Build Options

The Wdg driver files are compiled using

- Green Hills Multi 7.1.4 / Compiler 2017.1.4
- (Linaro GCC 6.3-2017.06~dev) 6.3.1 20170509 (Wed Jan 24 16:21:45 CST 2018 build.sh rev=g27a1317 s=L631 Earmv7 -V release_g27a1317_build_Fed_Earmv7)
- IAR: V8.11.2

The compiler, linker flags used for building the driver are explained below:

Note

The TS_T40D2M10I1R0 plugin name is composed as follow:

TS_T = Target_Id

D = Derivative_Id

M = SW_Version_Major

I = SW_Version_Minor

R = Revision

(i.e. Target_Id = 40 identifies CORTEXM architecture and Derivative_Id = 2 identifies the S32K14X)

3.1.1 GHS Compiler/Linker/Assembler Options

Table 3-1. Compiler Options

Option	Description
-cpu=cortexm4	Selects target processor: Arm Cortex M4
-cpu=cortexm0plus	Selects target processor: Arm Cortex M0+
-ansi	Specifies ANSI C with extensions. This mode extends the ANSI X3.159-1989 standard with certain useful and compatible constructs.
-Osize	Optimize for size.
-dual_debug	Enables the generation of DWARF, COFF, or BSD debugging information in the object file
-G	Generates source level debugging information and allows procedure call from debugger's command line.
--no_exceptions	Disables support for exception handling
-Wundef	Generates warnings for undefined symbols in preprocessor expressions
-Wimplicit-int	Issues a warning if the return type of a function is not declared before it is called
-Wshadow	Issues a warning if the declaration of a local variable shadows the declaration of a variable of the same name declared at the global scope, or at an outer scope
-Wtrigraphs	Issues a warning for any use of trigraphs
-Wall	Enables all the warnings about constructions that some users consider questionable, and that are easy to avoid even in conjunction with macros.
--prototype_errors	Generates errors when functions referenced or called have no prototype
--incorrect_pragma_warnings	Valid #pragma directives with wrong syntax are treated as warnings
-noslashcomment	C++ like comments will generate a compilation error
-preprocess_assembly_files	Preprocesses assembly files
-nostartfile	Do not use Start files
--short_enum	Store enumerations in the smallest possible type
-c	Produces an object file (called input-file.o) for each source file.
--no_commons	Allocates uninitialized global variables to a section and initializes them to zero at program startup.
-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. Produces an object file (called input-file.o) for each source file.
-list	Creates a listing by using the name of the object file with the .lst extension. Assembler option
-DAUTOSAR_OS_NOT_USED	-D defines a preprocessor symbol and optionally can set it to a value. AUTOSAR_OS_NOT_USED: By default in the package, the drivers are compiled to be used without Autosar OS. If the drivers are used with Autosar OS, the compiler option '-DAUTOSAR_OS_NOT_USED' must be removed from project options
-DDISABLE_MCAL_INTERMODULE_ASR_CHECK	-D defines a preprocessor symbol to disable the inter-module version check for AR_RELEASE versions. DISABLE_MCAL_INTERMODULE_ASR_CHECK: By default in the package, drivers are compiled to perform the inter-module version check as per Autosar BSW004. When the inter-module version check needs to be disabled then the DISABLE_MCAL_INTERMODULE_ASR_CHECK global define must be added to the list of compiler options.
-DGHS	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the GHS preprocessor symbol.

Table 3-2. Assembler Options

Option	Description
-cpu=cortexm4	Selects target processor: Arm Cortex M4
-cpu=cortexm0plus	Selects target processor: Arm Cortex M0+
-c	Produces an object file (called input-file.o) for each source file.
-preprocess_assembly_files	Preprocesses assembly files
-asm=list	Creates a listing by using the name of the object file with the .lst extension. Assembler option

Table 3-3. Linker Options

Option	Description
-Mn	Map file numeric ordering
-delete	Removal from the executable of functions that are unused and unreferenced
-v	Display removed unused functions
-ignore_debug_references	Ignores relocations from DWARF debug sections when using -delete.
-map	Creates a detailed map file
-keepmap	Keep the map file in the event of a link error
-lstartup	Link libstartup library -Run-time environment startup routines
-lsys	Link libsys library -Run-time environment system routines
-larch	Link libarch library -Target-specific run-time support. Any file produced by the Green Hills Compiler may depend on symbols in this library.
-lansi	Link libansi library -the standard C library
-L(/lib/thumb2)	Link thumb2 library
-lutf8_s32	Include utf8_s32.a to use the Wide Character Functions

3.1.2 IAR Compiler/Linker/Assembler Options

Table 3-4. Compiler Options

Option	Description
--cpu=Cortex-M4	Selects target processor: Arm Cortex M4
--cpu=Cortex-M0+	Selects target processor: Arm Cortex M0+
--cpu_mode=thumb	Selects generating code that executes in Thumb state.
--endian=little	Specifies the endianess of core: little endian.
-Ohz	Sets the optimization level to High, favoring size.
-c	Produces an object file (called input-file.o) for each source file.
--no_clustering	Disables static clustering optimizations.
--no_mem_idioms	Makes the compiler to not optimize code sequences that clear, set, or copy a memory region.
--no_explicit_zero_opt	Places the zero initialized variables in data section instead of bss.
--debug	Makes the compiler include information in the object modules.

Table continues on the next page...

Table 3-4. Compiler Options (continued)

Option	Description
--diag_suppress=Pa050	Suppresses diagnostic messages (warnings) about non-standard line endings.
-DAUTOSAR_OS_NOT_USED	-D defines a preprocessor symbol and optionally can set it to a value. AUTOSAR_OS_NOT_USED: By default in the package, the drivers are compiled to be used without Autosar OS. If the drivers are used with Autosar OS, the compiler option '-DAUTOSAR_OS_NOT_USED' must be removed from project options
-DIAR	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the IAR preprocessor symbol.
--require_prototypes	Forces the compiler to verify that all functions have proper prototypes.
--no_wrap_diagnostics	Disables line wrapping of diagnostic messages issued by compiler.
--no_system_include	Disables the automatic search for system include files.
-e	Enables language extensions. This option is needed by FLS driver which uses _packed structures.

Table 3-5. Assembler Options

Option	Description
--cpu=Cortex-M4	Selects target processor: Arm Cortex M4
--cpu=Cortex-M0+	Selects target processor: Arm Cortex M0+
--cpu_mode=thumb	Selects generating code that executes in Thumb state.
-g	Use this option to disable the automatic search for system include files.

Table 3-6. Linker Options

Option	Description
--cpu=Cortex-M4	Selects target processor: Arm Cortex M4
--cpu=Cortex-M0+	Selects target processor: Arm Cortex M0+
--map filename	Produces a map file.
--no_library_search	Disables automatic runtime library search.
--entry _start	Treats the symbol _start as a root symbol and as the start of the application.
--enable_stack_usage	Enables stack usage analysis.
--skip_dynamic_initialization	Suppress dynamic initialization during system startup.
--no_wrap_diagnostics	Disables line wrapping of diagnostic messages issued by linker.
--config	Specifies the configuration file to be used by the linker.

3.1.3 GCC Compiler/Linker/Assembler Options

Table 3-7. Compiler Options

Option	Description
-c	Produces an object file (called input-file.o) for each source file.
-Os	Use optimization for size.
-ggdb3	Produce debugging information for use by GDB. Level 3 includes extra information, such as all the macro definitions present in the program.
-mcpu=cortex-m4	Selects target processor: Arm Cortex M4
-mcpu=cortex-m0plus	Selects target processor: Arm Cortex M0+
-mthumb	Selects generating code that executes in Thumb state.
-ansi	Specifies ANSI C with extensions.
-mlittle-endian	Generate code for a processor running in little-endian mode.
-fomit-frame-pointer	Removes the frame pointer for all functions, which might make debugging harder.
-msoft-float	Use software floating-point instructions.
-fno-common	Specifies that the compiler should place uninitialized global variables in the data section of the object file, rather than generating them as common blocks.
-Wall	Enables all the warnings about constructions that some users consider questionable, and that are easy to avoid even in conjunction with macros.
-Wextra	Enables some extra warning flags that are not enabled by '-Wall'.
-Wstrict-prototypes	Warn if a function is declared or defined without specifying the argument types.
-Wno-sign-compare	Do not warn when a comparison between signed and unsigned values could produce an incorrect result when the signed value is converted to unsigned.
-fstack-usage	Geneates an extra file that specifies the maximum amount of stack used, on a per-function basis.
-fdump-ipa-all	Enables all inter-procedural analysis dumps.
-Werror=implicit-function-declaration	Generates an error when the prototype of the function is not defined..
-DAUTOSAR_OS_NOT_USED	-D defines a preprocessor symbol and optionally can set it to a value. AUTOSAR_OS_NOT_USED: By default in the package, the drivers are compiled to be used without Autosar OS. If the drivers are used with Autosar OS, the compiler option '-DAUTOSAR_OS_NOT_USED' must be removed from project options
-DGCC	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the GCC preprocessor symbol.
-std=c99	C programming language standard version c99

Table 3-8. Assembler Options

Option	Description
-mcpu=cortex-m4	Selects target processor: Arm Cortex M4
-mcpu=cortex-m0plus	Selects target processor: Arm Cortex M0+
-c	Produces an object file (called input-file.o) for each source file.
-mthumb	This option specifies that the assembler should start assembling Thumb instructions.
-x assembler-with-cpp	Indicates that the assembly code contains C directives and the C preprocessor must be run.

Table 3-9. Linker Options

Option	Description
-Map=filename	Print a link map to the file mapfile.
-T scriptfile	Use scriptfile as the linker script. This script replaces ld's default linker script(rather than adding to it), so commandfile must specify everything necessary to describe the output file.
--disable-newlib-supplied-syscalls -specs=nosys.specs	These options support for using newlib on core M0+
-u _printf_float -u _scanf_float	These options support generating profile report.
-nostartfiles	Do not use the standard system startup files when linking
-e _start	Specify that the program entry point is _start
-static	The --static flag tells the linker to link a static, not a dynamically linked
-lc	The -lc flag tells the linker to link this binary against the C library, which is newlib in our case.
-lnosys	The -lnosys flag tells the linker to link this binary against the "nosys" library
$(TOOLCHAIN_DIR)/arm-none-eabi/newlib/lib/thumb/v6-m $(TOOLCHAIN_DIR)/lib/gcc/arm-none-eabi/6.3.1/thumb/v6-m	Library for core M0+, added with -L and -B option
$(TOOLCHAIN_DIR)/arm-none-eabi/newlib/lib/thumb $(TOOLCHAIN_DIR)/arm-none-eabi/newlib/lib)	Library for core M4, added with -L and -B option

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 S32K14X 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_T40D2M10I1R0\include\Wdg.h
- ..\Wdg_TS_T40D2M10I1R0\include\Wdg_Channel.h
- ..\Wdg_TS_T40D2M10I1R0\include\Wdg_Wdog_Types.h
- ..\Wdg_TS_T40D2M10I1R0\include\Wdg_Wdog.h
- ..\Wdg_TS_T40D2M10I1R0\include\Wdg_Irq.h
- ..\Wdg_TS_T40D2M10I1R0\include\Wdg_IPW_Types.h
- ..\Wdg_TS_T40D2M10I1R0\include\Wdg_IPW.h
- ..\Wdg_TS_T40D2M10I1R0\include\Wdg_EnvCfg.h
- ..\Wdg_TS_T40D2M10I1R0\include\Reg_eSys_Wdog_defines.h

- ..\Wdg_TS_T40D2M10I1R0\include\Reg_eSys_Wdog.h
- ..\Wdg_TS_T40D2M10I1R0\src\Wdg.c
- ..\Wdg_TS_T40D2M10I1R0\src\Wdg_Channel.c
- ..\Wdg_TS_T40D2M10I1R0\src\Wdg_Wdog.c
- ..\Wdg_TS_T40D2M10I1R0\src\Wdg_Wdog_Isr.c

Wdg Generated Files

- Wdg_Cfg.c (For PC Variant) - For driver compilation, this file should be generated by the user using a configuration tool
- Wdg_PBcfg.c (For PB Variant) - For driver compilation, this file should be generated by the user using a configuration tool
- Wdg_Lcfg.c (For LT Variant) - For driver compilation, this file should be generated by the user using a configuration tool
- Wdg_Cfg.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_T40D2M10I1R0\include\Compiler.h
- ..\Base_TS_T40D2M10I1R0\include\Compiler_Cfg.h
- ..\Base_TS_T40D2M10I1R0\include\ComStack_Types.h
- ..\Base_TS_T40D2M10I1R0\include\Wdg_MemMap.h
- ..\Base_TS_T40D2M10I1R0\include\Mcal.h
- ..\Base_TS_T40D2M10I1R0\include\Platform_Types.h
- ..\Base_TS_T40D2M10I1R0\include\Std_Types.h
- ..\Base_TS_T40D2M10I1R0\include\Reg_eSys.h
- ..\Base_TS_T40D2M10I1R0\include\Soc_Ips.h
- ..\Base_TS_T40D2M10I1R0\include\SilRegMacros.h

Files from WdgIf folder:

- ..\WdgIf_TS_T40D2M10I1R0\include\WdgIf_Cfg.h
- ..\WdgIf_TS_T40D2M10I1R0\include\WdgIf.h
- ..\WdgIf_TS_T40D2M10I1R0\include\WdgIf_Types.h
- ..\WdgIf_TS_T40D2M10I1R0\src\WdgIf.c

Files from Dem folder:

- ..\Dem_TS_T40D2M10I1R0\include\Dem.h
- ..\Dem_TS_T40D2M10I1R0\include\Dem_IntErrId.h
- ..\Dem_TS_T40D2M10I1R0\include\Dem_Types.h
- ..\Dem_TS_T40D2M10I1R0\src\Dem.c

Files from Gpt folder:

- ..\Gpt_TS_T40D2M10I1R0\src\Gpt.c
- ..\Gpt_TS_T40D2M10I1R0\src\Gpt_IPW.c
- ..\Gpt_TS_T40D2M10I1R0\src\Gpt_Ftm.c
- ..\Gpt_TS_T40D2M10I1R0\src\Gpt_LPit.c
- ..\Gpt_TS_T40D2M10I1R0\src\Gpt_Lptmr.c
- ..\Gpt_TS_T40D2M10I1R0\include\Gpt.h
- ..\Gpt_TS_T40D2M10I1R0\include\Gpt_EnvCfg.h
- ..\Gpt_TS_T40D2M10I1R0\include\Gpt_Ftm.h
- ..\Gpt_TS_T40D2M10I1R0\include\Gpt_Ftm_Types.h
- ..\Gpt_TS_T40D2M10I1R0\include\Gpt_IPW.h
- ..\Gpt_TS_T40D2M10I1R0\include\Gpt_IPW_Types.h
- ..\Gpt_TS_T40D2M10I1R0\include\Gpt_LPit.h
- ..\Gpt_TS_T40D2M10I1R0\include\Gpt_LPit_Types.h
- ..\Gpt_TS_T40D2M10I1R0\include\Gpt_Lptmr.h
- ..\Gpt_TS_T40D2M10I1R0\include\Gpt_Lptmr_Types.h
- ..\Gpt_TS_T40D2M10I1R0\include\Reg_eSys_LPit.h
- ..\Gpt_TS_T40D2M10I1R0\include\Reg_eSys_Lptmr.h
- Gpt_Cfg.c (For PC Variant) - This file should be generated by the user using a configuration tool for compilation
- Gpt_PBCfg.c (For PB Variant) - This file should be generated by the user using a configuration tool for compilation
- Gpt_Cfg.h - This file should be generated by the user using a configuration tool for compilation

Files from Mcl folder:

- ..\Mcl_TS_T40D2M10I1R0\src\CDD_Mcl.c
- ..\Mcl_TS_T40D2M10I1R0\src\Ftm_Common.c
- ..\Mcl_TS_T40D2M10I1R0\src\LPit_Common.c
- ..\Mcl_TS_T40D2M10I1R0\src\Lptmr_Common.c
- ..\Mcl_TS_T40D2M10I1R0\src\Mcl_Dma.c
- ..\Mcl_TS_T40D2M10I1R0\src\Mcl_Dma_Irq.c
- ..\Mcl_TS_T40D2M10I1R0\src\Mcl_DmaMux.c
- ..\Mcl_TS_T40D2M10I1R0\src\Mcl_IPW.c
- ..\Mcl_TS_T40D2M10I1R0\src\Mcl_TrgMux.c
- ..\Mcl_TS_T40D2M10I1R0\include\CDD_Mcl.h
- ..\Mcl_TS_T40D2M10I1R0\include\Ftm_Common_Types.h
- ..\Mcl_TS_T40D2M10I1R0\include\Mcl.h
- ..\Mcl_TS_T40D2M10I1R0\include\Mcl_Dma.h
- ..\Mcl_TS_T40D2M10I1R0\include\Mcl_Dma_Types.h
- ..\Mcl_TS_T40D2M10I1R0\include\Mcl_DmaMux_Types.h
- ..\Mcl_TS_T40D2M10I1R0\include\Mcl_EnvCfg.h
- ..\Mcl_TS_T40D2M10I1R0\include\Mcl_IPW.h

- ..\Mcl_TS_T40D2M10I1R0\include\Mcl_IPW_Notif.h
- ..\Mcl_TS_T40D2M10I1R0\include\Mcl_IPW_Types.h
- ..\Mcl_TS_T40D2M10I1R0\include\Mcl_Notif.h
- ..\Mcl_TS_T40D2M10I1R0\include\Mcl_TrgMux.h
- ..\Mcl_TS_T40D2M10I1R0\include\Mcl_TrgMux_Types.h
- ..\Mcl_TS_T40D2M10I1R0\include\Mcl_Types.h
- ..\Mcl_TS_T40D2M10I1R0\include\Reg_eSys_Dma.h
- ..\Mcl_TS_T40D2M10I1R0\include\Reg_eSys_DmaMux.h
- ..\Mcl_TS_T40D2M10I1R0\include\Reg_eSys_Ftm.h
- ..\Mcl_TS_T40D2M10I1R0\include\Reg_eSys_LPit.h
- ..\Mcl_TS_T40D2M10I1R0\include\Reg_eSys_Lptmr.h
- ..\Mcl_TS_T40D2M10I1R0\include\Reg_eSys_TrgMux.h

Files from Det folder:

- ..\Det_TS_T40D2M10I1R0\include\Det.h
- ..\Det_TS_T40D2M10I1R0\src\Det.c

3.3 Setting up the Plug-ins

All the Autosar MCAL drivers for S32K14X were designed to be configured using Tresos Studio (version EB tresos Studio 24.0.1 b180321-0610 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_T40D2M10I1R0\config\Wdg.xdm
  - ..\Gpt_TS_T40D2M10I1R0\config\Gpt.xdm
  - ..\Mcu_TS_T40D2M10I1R0\config\Mcu.xdm
  - ..\EcuM_TS_T40D2M10I1R0\config\EcuM.xdm
  - ..\EcuC_TS_T40D2M10I1R0\config\EcuC.xdm
  - ..\Dem_TS_T40D2M10I1R0\config\Dem.xdm
  - ..\Resource_TS_T40D2M10I1R0\config\Resource.xdm
- VSMD (Vendor Specific Module Definition) file(s) in AUTOSAR compliant EPD format:
  - ..\Wdg_TS_T40D2M10I1R0\autosar\Wdg_<subderivative_name>.epd
  - ..\Gpt_TS_T40D2M10I1R0\autosar\Gpt_<subderivative_name>.epd
  - ..\Mcu_TS_T40D2M10I1R0\autosar\Mcu_<subderivative_name>.epd
  - ..\EcuM_TS_T40D2M10I1R0\autosar\EcuM.epd
  - ..\EcuC_TS_T40D2M10I1R0\autosar\EcuC.epd
  - ..\Dem_TS_T40D2M10I1R0\autosar\Dem.epd
  - ..\Resource_TS_T40D2M10I1R0\autosar\Resource_<subderivative_name>.epd

- Code Generation Templates for Pre-Compile time configuration parameters:
  - ..\Wdg_TS_T40D2M10I1R0\generate\include\Wdg_Cfg.h
  - ..\Wdg_TS_T40D2M10I1R0\generate\src\Wdg_CfgExt.c
  - ..\Wdg_TS_T40D2M10I1R0\generate_PC\src\Wdg_Cfg.c
  - ..\Wdg_TS_T40D2M10I1R0\generate_PB\src\Wdg_PBcfg.c
  - ..\EcuM_TS_T40D2M10I1R0\generate_PC\include\EcuM_Cfg.h
  - ..\Dem_TS_T40D2M10I1R0\generate_PC\include\Dem_intErrId.h
  - ..\Gpt_TS_T40D2M10I1R0\generate_PC\include\Gpt_Cfg.h
  - ..\Gpt_TS_T40D2M10I1R0\generate_PC\src\Gpt_Cfg.c

- Code Generation Templates for Post-Build time configuration parameters:
  - ..\Wdg_TS_T40D2M10I1R0\generate\src\Wdg_CfgExt.c
  - ..\Wdg_TS_T40D2M10I1R0\generate\include\Wdg_Cfg.h
  - ..\Wdg_TS_T40D2M10I1R0\generate_PB\src\Wdg_PBcfg.c
  - ..\EcuM_TS_T40D2M10I1R0\generate_PC\include\EcuM_Cfg.h
  - ..\Dem_TS_T40D2M10I1R0\generate_PC\include\Dem_intErrId.h
  - ..\Gpt_TS_T40D2M10I1R0\generate_PC\include\Gpt_Cfg.h
  - ..\Gpt_TS_T40D2M10I1R0\generate_PB\src\Gpt_PBCfg.c

- Code Generation Templates for Link time configuration parameters:
  - ..\Wdg_TS_T40D2M10I1R0\generate\src\Wdg_CfgExt.c
  - ..\Wdg_TS_T40D2M10I1R0\generate\include\Wdg_Cfg.h
  - ..\Wdg_TS_T40D2M10I1R0\generate_LT\src\Wdg_Lcfg.c
  - ..\EcuM_TS_T40D2M10I1R0\generate_PC\include\EcuM_Cfg.h
  - ..\Dem_TS_T40D2M10I1R0\generate_PC\include\Dem_intErrId.h
  - ..\Gpt_TS_T40D2M10I1R0\generate_PC\include\Gpt_Cfg.h
  - ..\Gpt_TS_T40D2M10I1R0\generate_PB\src\Gpt_PBCfg.c

- Code Generation Templates for parameters without variation points:
  - ..\Wdg_TS_T40D2M10I1R0\generate\src\Wdg_CfgExt.c
  - ..\Wdg_TS_T40D2M10I1R0\generate\include\Wdg_Cfg.h
  - ..\Wdg_TS_T40D2M10I1R0\generate_PB\src\Wdg_PBcfg.c
  - ..\EcuM_TS_T40D2M10I1R0\generate_PC\include\EcuM_Cfg.h
  - ..\Dem_TS_T40D2M10I1R0\generate_PC\include\Dem_intErrId.h
  - ..\Gpt_TS_T40D2M10I1R0\generate_PC\include\Gpt_Cfg.h

- Code Generation Templates for variant aware parameters:
  - ..\Wdg_TS_T40D2M10I1R0\generate\src\Wdg_CfgExt.c
  - ..\Wdg_TS_T40D2M10I1R0\generate\include\Wdg_Cfg.h
  - ..\Wdg_TS_T40D2M10I1R0\generate_PB\src\Wdg_PBcfg.c
  - ..\Gpt_TS_T40D2M10I1R0\generate_PC\include\Gpt_Cfg.h
  - ..\Gpt_TS_T40D2M10I1R0\generate_PB\src\Gpt_PBCfg.c

Steps to generate the configuration:

1. Copy the module folders Wdg_TS_T40D2M10I1R0 , Resource_TS_T40D2M10I1R0 , Base_TS_T40D2M10I1R0 , Det_TS_T40D2M10I1R0 , Dem_TS_T40D2M10I1R0, WdgIf_TS_T40D2M10I1R0, Gpt_TS_T40D2M10I1R0, Mcu_TS_T40D2M10I1R0, Mcl_TS_T40D2M10I1R0, Rte_TS_T40D2M10I1R0, EcuM_TS_T40D2M10I1R0, EcuC_TS_T40D2M10I1R0 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: s32k148_lqfp144, s32k148_lqfp176, s32k148_mapbga100, s32k146_lqfp144, s32k146_lqfp100, s32k146_lqfp64, s32k146_mapbga100, s32k144_lqfp100, s32k144_lqfp64, s32k144_mapbga100, s32k142_lqfp100, s32k142_lqfp64, s32k118_lqfp48, s32k118_lqfp64, s32k142_lqfp48, s32k144_lqfp48, s32k148_lqfp100 .
- **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.

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 0,1 and 2 available on the hardware,then the name of the init functions will be Wdg_43_Instance0_Init, Wdg_43_Instance1_Init and Wdg_43_Instance2_Init instead of Wdg_Init().

4.2 Function Calls during Shutdown

None.

4.3 Function Calls during Wake-up

None.

Chapter 5
Module requirements

5.1 Exclusive areas to be defined in BSW scheduler

WDG_EXCLUSIVE_AREA_00 Used by instance 0-2 of the WDG module to protect variable Wdg_au16Timeout in functions Wdg_ChannelTrigger.

WDG_EXCLUSIVE_AREA_01 Used by instance 0-2 of the WDG module to protect variable Wdg_au16Timeout in functions Wdg_ChannelInit.

WDG_EXCLUSIVE_AREA_02 Used by instance 0-2 of the WDG module to protect variable Wdg_au16Timeout in functions Wdg_ChannelSetMode.

WDG_EXCLUSIVE_AREA_03 Used by instance 0-2 of the WDG module to protect variable Wdg_au16Timeout in functions Wdg_ChannelSetTriggerCondition.

WDG_EXCLUSIVE_AREA_04 Used by instance 0-2 of the WDG module to protect variable Wdg_aeStatus in functions Wdg_ChannelValidateGlobalCall.

WDG_EXCLUSIVE_AREA_05 Used by instance 0-2 of the WDG module to protect variable Wdg_aeStatus in functions Wdg_ChannelEndValidateGlobalCall.

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 a table, it means that those 2 critical regions cannot interrupt each other.

The critical regions from interrupts are grouped in “Interrupt Service Routines Critical Regions (composed diagram)”. If an exclusive area is “exclusive” with the composed “Interrupt Service Routines Critical Regions (composed diagram)” group, it means that it is exclusive with each one of the ISR critical regions.

Table 5-1. Exclusive Areas

	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_00		X	X	X		
WDG_EXCLUSIVE_AREA_01	X		X	X		
WDG_EXCLUSIVE_AREA_02	X	X		X		
WDG_EXCLUSIVE_AREA_03	X	X	X			
WDG_EXCLUSIVE_AREA_04					X	X
WDG_EXCLUSIVE_AREA_05					X	X

5.2 Peripheral Hardware Requirements

- When interrupts are enabled (CS[INT] = 1):After a reset-triggering event (like a counter timeout or invalid refresh attempt), the watchdog first generates an interrupt request. Next, the watchdog delays 128 bus clocks (from the interrupt vector fetch, not the reset-triggering event) before forcing a reset, to allow the interrupt service routine (ISR) to perform tasks (like analyzing the stack to debug code).
- When interrupts are disabled (CS[INT] = 0): the watchdog does not delay the forcing a reset.

Software finishing its main control loop faster than expected could be an indication of a problem. Depending on the requirements of the application, the WDOG can be programmed to force a reset when refresh attempts are early. When Window mode is enabled, the watchdog must be refreshed after the counter has reached a minimum expected time value. Otherwise, the watchdog resets the MCU. The minimum expected time value is specified in the WIN register. Setting CS[WIN] enables Window mode.

5.3 ISR to configure within OS – dependencies

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

Table 5-2. Wdg ISR

ISR Name	Hardware interrupt vector
Wdg_Wdog0_Isr	22

5.4 ISR Macro

MCAL 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:

a. OS is not used - AUTOSAR_OS_NOT_USED is defined:

i. If USE_SW_VECTOR_MODE is defined:

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

In this case, drivers' interrupt handlers are normal C functions and the prolog/epilog handle the context save and restore.

ii. If USE_SW_VECTOR_MODE is not defined:

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

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

Custom OS is used - AUTOSAR_OS_NOT_USED is not defined

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

In this case, OS is handling the execution context when an interrupt occurs. Drivers' interrupt handlers are normal C functions.

Other vendor's OS is used - AUTOSAR_OS_NOT_USED is not defined. Please refer to the OS documentation for description of the ISR macro.

5.5 Other AUTOSAR modules - dependencies

Watchdog Interface:

This module is necessary for importing the mode's type in which the watchdog can be set up. Configuration dependency to other module WDG Driver Integration Manual Module requirements None.

Dependencies

- **BASE:** The BASE module contains the common files/definitions needed by all MCAL modules.
- **DEM** The DEM module is used for enabling reporting of production relevant error status. The API function used is Dem_ReportErrorStatus().
- **DET:** The DET module is used for enabling Development error detection. The API function used is Det_ReportError(). The activation / deactivation of Development error detection is configurable using the 'WdgDevErrorDetect' configuration parameter.
- **ECUC:** The ECUC module is used for ECU configuration. MCAL 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 MCAL software modules. The clocks need to be initialized prior to using the Wdg driver.
- **GPT** This module is required to reset periodically the watchdog timeout counter.
- **MCL** Module is needed by Gpt functions
- **RESOURCE** Resource module is used to select microcontroller's derivatives. Current driver has support for the following derivatives, everyone having attached a Resource file: s32k148_lqfp144, s32k148_lqfp176, s32k148_mapbga100, s32k146_lqfp144, s32k146_lqfp100, s32k146_lqfp64, s32k146_mapbga100, s32k144_lqfp100, s32k144_lqfp64, s32k144_mapbga100, s32k142_lqfp100, s32k142_lqfp64, s32k118_lqfp48, s32k118_lqfp64, s32k142_lqfp48, s32k144_lqfp48, s32k148_lqfp100 .
- **RTE** The RTE module is needed for implementing data consistency of exclusive areas that are used by Wdg module.

Resource Parameters Configurations32k148_lqfp144, s32k148_lqfp176, s32k148_mapbga100, s32k146_lqfp144, s32k146_lqfp100, s32k146_lqfp64, s32k146_mapbga100, s32k144_lqfp100, s32k144_lqfp64, s32k144_mapbga100, s32k142_lqfp100, s32k142_lqfp64, s32k118_lqfp48, s32k118_lqfp64, s32k142_lqfp48, s32k144_lqfp48, s32k148_lqfp100

5.6 Data cache restriction

None

5.7 User Mode Support

Wdg driver can run in user mode.

Chapter 6
Main API Requirements

6.1 Main functions calls within BSW scheduler

None.

6.2 API Requirements

None

6.3 Calls to Notification Functions, Callbacks, Callouts

Call-back Notifications:

There are no call-back notifications from the WDG driver.

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

7.1 Sections to be defined in Wdg_MemMap.h

Tables descibe Sections to be defined in Wdg_MemMap.h:

Table 7-1. Sectionto be define

<Section name>	Tyep 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_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

Table continues on the next page...

Table 7-1. Sectionto be define (continued)

		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
WDG_STOP_SEC_VAR_INIT_16	Variables	End of above section.
WDG_START_SEC_VAR_NO_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 never cleared and never initialized by start-up code (BBS).
WDG_STOP_SEC_VAR_NO_INIT_UNSPECIFIED	Variables	End of above section.

7.2 Linker command file

Memory shall be allocated for every section defined in Wdg_MemMap.h

Chapter 8
Configuration parameters considerations

Configuration parameter class for Autosar Wdg driver fall into the following variants as defined below:

8.1 Configuration Parameters

Specifies whether the configuration parameter shall be of configuration class Post Build.

Table 8-1. Configuration Parameters

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
Wdg	IMPLEMENTATION_CONFIG_VARIANT	Pre Compile parameter for all Variants of Configuration	PreCompile or LinkTime or PostBuild
WdgDemEventParameterRefs	WDG_E_DISABLE_REJECTED	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WDG_E_MODE_FAILED	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WDG_E_CORRUPT_CONFIG	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WDG_E_UNLOCKED	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WDG_E_INVALID_PARAMETER	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WDG_E_FORBIDDEN_INVOCATION	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WDG_E_INVALID_CALL	Pre Compile parameter for all Variants of Configuration	Pre Compile
WdgGeneral	WdgDisableDemReportErrorStatus	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WdgDevErrorDetect	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WdgDisableAllowed	Pre Compile parameter for all Variants of Configuration	Pre Compile

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
	Wdg Enable User Mode Support	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WdgIndex	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WdgInitialTimeout	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WdgMaxTimeout	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WdgRunArea	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WdgTriggerLocation	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WdgCallbackNotification0	Pre Compile parameter for all Variants of Configuration	Post Build
	WdgVersionInfoApi	Pre Compile parameter for all Variants of Configuration	Pre Compile
WdgSettingsConfig	WdgInstance	VariantPC or VariantPB	Pre Compile
	WdgDefaultMode	VariantPC or VariantPB	Post Build
	WdgExternalTriggerCounterRef	VariantPC or VariantPB	Post Build
	WdgInterruptContentEnable	Pre Compile parameter for all Variants of Configuration	Pre Compile
WdgSettingsConfig/ WdgExternalConfiguration	WdgExternalContainerRef	N/A	N/A
WdgSettingsConfig/ WdgSettingsFast	WdgClockValue	VariantPC or VariantPB	Post Build
	WdgClkSrcRef	VariantPC or VariantPB	Post Build
	WdgRunsInStopMode	VariantPC or VariantPB	Post Build
	WdgRunsInDebugMode	VariantPC or VariantPB	Post Build
	WdgRunsInWaitMode	VariantPC or VariantPB	Post Build
	WdgOperationMode	VariantPC or VariantPB	Post Build
	WdgClockSelection	VariantPC or VariantPB	Post Build
	WdgTimeoutPeriod	VariantPC or VariantPB	Post Build
	WdgWindowMode	VariantPC or VariantPB	Post Build
	WdgWindowPeriod	VariantPC or VariantPB	Post Build
	WdgPrescalerEnabled	VariantPC or VariantPB	Post Build
	WdgAllowUpdates	VariantPC or VariantPB	Post Build
	WdgTestMode	VariantPC or VariantPB	Post Build
WdgSettingsConfig/ WdgSettingsOff	WdgAllowUpdates	VariantPC or VariantPB	Post Build
WdgSettingsConfig/ WdgSettingsSlow	WdgClockValue	VariantPC or VariantPB	Post Build
	WdgClkSrcRef	VariantPC or VariantPB	Post Build

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
	WdgRunsInStopMode	VariantPC or VariantPB	Post Build
	WdgRunsInDebugMode	VariantPC or VariantPB	Post Build
	WdgRunsInWaitMode	VariantPC or VariantPB	Post Build
	WdgOperationMode	VariantPC or VariantPB	Post Build
	WdgClockSelection	VariantPC or VariantPB	Post Build
	WdgTimeoutPeriod	VariantPC or VariantPB	Post Build
	WdgWindowMode	VariantPC or VariantPB	Post Build
	WdgWindowPeriod	VariantPC or VariantPB	Post Build
	WdgPrescalerEnabled	VariantPC or VariantPB	Post Build
	WdgAllowUpdates	VariantPC or VariantPB	Post Build
	WdgTestMode	VariantPC or VariantPB	Post Build
WdgClockReferencePoint	WdgClockReference	VariantPC or VariantPB	Post Build
WdgPublishedInformation	WdgTriggerMode	Pre Compile parameter for all Variants of Configuration	Pre Compile
CommonPublishedInformation	ArReleaseMajorVersion	Pre Compile parameter for all Variants of Configuration	Pre Compile
	ArReleaseMinorVersion	Pre Compile parameter for all Variants of Configuration	Pre Compile
	ArReleaseRevisionVersion	Pre Compile parameter for all Variants of Configuration	Pre Compile
	ModuleId	Pre Compile parameter for all Variants of Configuration	Pre Compile
	SwMajorVersion	Pre Compile parameter for all Variants of Configuration	Pre Compile
	SwMinorVersion	Pre Compile parameter for all Variants of Configuration	Pre Compile
	SwPatchVersion	Pre Compile parameter for all Variants of Configuration	Pre Compile
	VendorApiInfix	Pre Compile parameter for all Variants of Configuration	Pre Compile
	VendorId	Pre Compile parameter for all Variants of Configuration	Pre Compile

Chapter 9
Integration Steps

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

- Generate the required Wdg configurations. For more details refer to section Files required for Compilation
- Allocate proper memory sections in Wdg_MemMap.h and linker command file. For more details refer to section Sections to be defined in Wdg_MemMap.h
- Compile & build the Wdg with all the dependent modules. For more details refer to section Building the Driver

Chapter 10
ISR Reference

ISR functions exported by the Wdg driver.

Chapter 11
External Assumptions for WDG driver

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

[SMCAL_CPR_EXT63]

<< The application shall ensure that Wdg_SetTriggerCondition() function is not preempting itself or any of the following WDG functions:

- Wdg_Init()

- Wdg_SetMode() >>

[SMCAL_CPR_EXT64]

<< The application must not concurrently call Wdg functions, with one exception: GetVersionInfo only can get interrupted or may interrupt. >>

[SMCAL_CPR_EXT66]

<< 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, Serr, 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.

Motivation 2:with GHS compiler and some optimization settings when Wdg runs from RAM and other modules which interact with WDG run from ROM there is a compiler error.

[SMCAL_CPR_EXT67]

<< 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, Serr, Gpt, SchM,application code will be executed from ROM). >>

[SMCAL_CPR_EXT163]

<< If interrupts are locked a centralized function pair to lock and unlock interrupts shall be used. >>

[SMCAL_CPR_EXT173]

<< It shall be the integrator responsibility to configure the additional harware resources (i.e. FCCU) to handle the signals from watchdog modules to generate either an interrupt or a reset. >>

[SMCAL_CPR_EXT174]

<< Due to the 128 bus clocks requirement for re-configuring the watchdog, some delays shall be inserted in the integration code before executing STOP or WAIT instructions after reconfiguring the watchdog.

>>

NOTE

Rationale: This ensures that the watchdog's new configuration takes effect before the MCU enters low power mode. Otherwise, the MCU may not be waken up from low power mode.

[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). >>

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer’s technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

While NXP has implemented advanced security features, all products may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on customer's applications and products, and NXP accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.

Document Number IM2WDGASR4.3 Rev0001R1.0.1
Revision 1.0