Integration Manual

for MPC574XG WDG Driver

Document Number: IM35WDGASR4.2 Rev0002 RTM 1.0.0
Rev. 2.3.0

Contents

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

Chapter 6
Main API Requirements

Chapter 7
Memory Allocation

Chapter 8
Configuration parameters considerations

Chapter 9
Integration Steps

Chapter 10
External Assumptions for WDG driver

Chapter 1
Revision History

Table 1-1. Revision History

Revision	Date	Author	Description
1.0.0	19/08/2014	Livia Firan	Updated Manual for MPC574XG AUTOSAR 4.0 Beta 0.9.0 Release
2.0.0	24/04/2015	Bach Nguyen	Updated Manual for MPC574XG AUTOSAR 4.0 RTM 1.0.0 Release
2.1.0	12/07/2015	Bach Nguyen	Updated Manual for MPC574XG AUTOSAR 4.0 RTM 1.0.1 Release
2.2.0	12/08/2016	Tuyen Nguyen Duy	Updated Manual for MPC574XG AUTOSAR 4.0 RTM 1.0.2 Release
2.3.0	17/02/2017	Tuyen Nguyen Duy	Updated Manual for MPC574XG AUTOSAR 4.2 RTM 1.0.0 Release

Chapter 2
Introduction

This integration manual describes the integration requirements for Wdg Driver for MPC574XG microcontrollers.

2.1 Supported Derivatives

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

Table 2-1. MPC574XG Derivatives

NXP Semiconductor	MPC5748G_LQFP176, MPC5748G_MAPBGA256, MPC5748G_MAPBGA324, MPC5747G_LQFP176, MPC5747G_MAPBGA256, MPC5747G_MAPBGA324, MPC5746G_LQFP176, MPC5746G_MAPBGA256, MPC5746G_MAPBGA324, MPC5748C_LQFP176, MPC5748C_MAPBGA256, MPC5748C_MAPBGA324, MPC5747C_LQFP176, MPC5747C_MAPBGA256, MPC5747C_MAPBGA324, MPC5746C_LQFP176, MPC5746C_MAPBGA256, MPC5746C_MAPBGA324, MPC5746C_MAPBGA100, MPC5745C_LQFP176, MPC5745C_MAPBGA256, MPC5745C_MAPBGA100, MPC5744C_LQFP176, MPC5744C_MAPBGA256, MPC5744C_MAPBGA100, MPC5746B_LQFP176, MPC5746B_MAPBGA256, MPC5746B_MAPBGA100, MPC5744B_LQFP176, MPC5744B_MAPBGA256,

Table 2-1. MPC574XG Derivatives

	MPC5744B_MAPBGA100, MPC5745B_LQFP176, MPC5745B_MAPBGA256, MPC5745B_MAPBGA100

All of the above microcontroller devices are collectively named as MPC574XG .

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
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	AUTOSAR 4.2 Rev0002Wdg Driver Software Specification Document.	R4.2 Rev 2
2	MPC5748G Reference Manual	Rev. 5, 12/2016
3	MPC5746C Reference Manual	Rev. 4, 12/2016
4	MPC5748G_1N81M_Rev.2 (official document) (1N81M)	Jun-16
5	MPC5748G_1N81M_0N78S_Comparison_Summary_v2_0 (internal document) (1N81M, 0N78S)	31.10.2016
6	MPC5746C_1N06M_Rev.4 (official document) (1N06M)	Jul-16
7	MPC5746C_cut1.1_cut2.0_cut2.1_comparison_v0 (internal document) (1N06M, 0N84S, 1N84S)	14-Sep-16
8	C3M_cut2.1_new_errata_20170113 (internal document) (1N84S)	13-Jan-17

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 SemiconductorMPC574XG . It also explains the EB Tresos Studio plugin setup procedure.

3.1 Build Options

The Wdg driver files are compiled using

- Windriver DIAB DIAB_5_9_6_2
- Green Hills Multi 7.1.4 / Compiler 2015.1.6

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

Note

The TS_T2D35M10I0R0 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 = 2 identifies PA architecture and Derivative_Id = 35 identifies the MPC574XG)

3.1.1 DIAB Compiler/Linker/Assembler Options

Table 3-1. Compiler Options

Option	Description
-tPPCE200Z4204N3VEN:simple	Sets target processor to PPCE200Z4204N3VEN, generates ELF using EABI conventions, No floating point support (minimizes the required runtime), selects simple environment settings for Startup Module and Libraries
-tPPCE200Z210N3VEN:simple	Sets target processor to PPCE200Z210N3VEN, generates ELF using EABI conventions, No floating point support (minimizes the required runtime), selects simple environment settings for Startup Module and Libraries
-Xdialect-ansi	Follow the ANSI C standard with some additions
-XO	Enables extra optimizations to produce highly optimized code
-g3	Generate symbolic debugger information and do all optimizations.
-Xsize-opt	Optimize for size rather than speed when there is a choice
-Xsmall-data=0	Set Size Limit for 'small data' Variables to zero.
-Xsmall-const=0	Set Size Limit for “small const” Variables to zero.
-Xaddr-sconst=0x11	Specify addressing for constant static and global variables with size less than or equal to -Xsmall-const to far-absolute.
-Xaddr-sdata=0x11	Specify addressing for non-constant static and global variables with size less than or equal to -Xsmall-data in size to far-absolute.
-Xno-common	Disable use of the 'COMMON' feature so that the compiler or assembler will allocate each uninitialized public variable in the .bss section for the module defining it, and the linker will require exactly one definition of each public variable
-Xnested-interrupts	Allow nested interrupts
-Xdebug-dwarf2	Generate symbolic debug information in dwarf2 format
-Xdebug-local-all	Force generation of type information for all local variables
-Xdebug-local-cie	Create common information entry per module
-Xdebug-struct-all	Force generation of type information for all typedefs, struct, union and class types
-Xforce-declarations	Generates warnings if a function is used without a previous declaration
-ee1481	Generate an error when the function was used before it has been declared
-Xmacro-undefined-warn	Generates a warning when an undefined macro name occurs in a #if preprocessor directive
-Xlink-time-lint	Enable the checking of object and function declarations across compilation units, as well as the consistency of compiler options used to compile source files
-W:as:,-l	Pass the option '-l' (lower case letter L) to the assembler to get an assembler listing file
-Wa,-Xisa-vle	Instruct the assembler to expect and assemble VLE (Variable Length Encoding) instructions rather than BookE instructions.
-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
-DUSE_SW_VECTOR_MODE	-D defines a preprocessor symbol and optionally can set it to a value. USE_SW_VECTOR_MODE: By default in the package, drivers are compiled to be used with interrupt controller configured to be in hardware vector mode. In case of AUTOSAR_OS_NOT_USED, the compiler option "-DUSE_SW_VECTOR_MODE" must be added to the list of compiler options to be used with interrupt controller configured to be in software vector mode.

Table continues on the next page...

Table 3-1. Compiler Options (continued)

Option	Description
-DDIAB	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the DIAB preprocessor symbol.
-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.
-c	Stop after assembly, produce object file.

Table 3-2. Assembler Options

Option	Description
-tPPCE200Z4204N3VEN:simple	Sets target processor to PPCE200Z4204N3VEN, generates ELF using EABI conventions, No floating point support (minimizes the required runtime), selects simple environment settings for Startup Module and Libraries
-tPPCE200Z210N3VEN:simple	Sets target processor to PPCE200Z210N3VEN, generates ELF using EABI conventions, No floating point support (minimizes the required runtime), selects simple environment settings for Startup Module and Libraries
-g	Dump the symbols in the global symbol table in each archive file.
-Xisa-vle	Expect and assemble VLE (Variable Length Encoding) instructions rather than Book E instructions. The default code section is named .text_vle instead of .text, and the default code section fill "character" is set to 0x44444444 instead of 0. The .text_vle code section will have ELF section header flags marking it as VLE code, not Book E code.
-Xasm-debug-on	Generate debug line and file information
-Xdebug-dwarf2	Generate symbolic debug information in dwarf2 format
-Xsemi-is-newline	Treat the semicolon (;) as a statement separator instead of a comment character.

Table 3-3. Linker Options

Option	Description
-tPPCE200Z4204N3VEN:simple	Sets target processor to tPPCE200Z4204N3VEN, generates ELF using EABI conventions, No floating point support (minimizes the required runtime), selects simple environment settings for Startup Module and Libraries
-tPPCE200Z210N3VEN:simple	Sets target processor to tPPCE200Z210N3VEN, generates ELF using EABI conventions, No floating point support (minimizes the required runtime), selects simple environment settings for Startup Module and Libraries
-Xelf	Generates ELF object format for output file
-m6	Generates a detailed link map and cross reference table
-Xlink-time-lint	Enable the checking of object and function declarations across compilation units, as well as the consistency of compiler options used to compile source files

3.1.2 GHS Compiler/Linker/Assembler Options

Table 3-4. Compiler Options

Option	Description
-cpu=ppc5748gz4204	Selects target processor: ppc5748gz4204
-cpu=ppc5748gz210	Selects target processor: ppc5748gz210
-ansi	Specifies ANSI C with extensions. This mode extends the ANSI X3.159-1989 standard with certain useful and compatible constructs.
-noSPE	Disables the use of SPE and vector floating point instructions by the compiler.
-Ospace	Optimize for size.
-sda=0	Enables the Small Data Area optimization with a threshold of 0.
-vle	Enables VLE code generation
-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
--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
--diag_error 223	Sets the specified compiler diagnostic messages to the level of error
-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
-DUSE_SW_VECTOR_MODE	-D defines a preprocessor symbol and optionally can set it to a value. USE_SW_VECTOR_MODE: By default in the package, drivers are compiled to be used with interrupt controller configured to be in hardware vector mode. In case of AUTOSAR_OS_NOT_USED, the compiler option "-DUSE_SW_VECTOR_MODE" must be added to the list of compiler options to be used with interrupt controller configured to be in software vector mode.
-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.
-c	Produces an object file (called input-file.o) for each source file.

Table 3-5. Assembler Options

Option	Description
-cpu=ppc5748gz4204	Selects target processor: ppc5748gz4204
-cpu=ppc5748gz210	Selects target processor: ppc5748gz210
-G	Generates source level debugging information and allows procedure call from debugger's command line.
-list	Creates a listing by using the name of the object file with the .lst extension

Table 3-6. Linker Options

Option	Description
-cpu=ppc5748gz4204	Selects target processor: ppc5748gz4204
-cpu=ppc5748gz210	Selects target processor: ppc5748gz210
-nostartfiles	Do not use Start files.
-vle	Enables VLE code generation
--nocpp	Do not Generate Constructors/Destructors
-Mn	sort numerically the MAP file
-delete	The -delete option instructs the linker to remove functions that are not referenced in the final executable.
-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.
-keepmap	keeps the MAP file in case of link error

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 MPC574XG 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_T2D35M10I0R0 \include\Wdg_43_Instance0.h
- ..\ Wdg_TS_T2D35M10I0R0 \include\Wdg_43_Instance1.h
- ..\ Wdg_TS_T2D35M10I0R0 \include\Wdg_43_Instance2.h
- ..\ Wdg_TS_T2D35M10I0R0 \include\Wdg_Channel.h
- ..\ Wdg_TS_T2D35M10I0R0 \include\Wdg_SWT_Types.h
- ..\ Wdg_TS_T2D35M10I0R0 \include\Wdg_Swt.h
- ..\ Wdg_TS_T2D35M10I0R0 \include\Wdg_Irq.h

- ..\ Wdg_TS_T2D35M10I0R0 \include\Wdg_IPW_Types.h
- ..\ Wdg_TS_T2D35M10I0R0 \include\Wdg_IPW.h
- ..\ Wdg_TS_T2D35M10I0R0 \include\Wdg_EnvCfg.h
- ..\ Wdg_TS_T2D35M10I0R0 \include\Reg_eSys_Swt_defines.h
- ..\ Wdg_TS_T2D35M10I0R0 \include\Reg_eSys_Swt.h
- ..\ Wdg_TS_T2D35M10I0R0 \src\Wdg_Channel.c
- ..\ Wdg_TS_T2D35M10I0R0 \src\Wdg_Swt.c
- ..\ Wdg_TS_T2D35M10I0R0 \src\Wdg_Swt_Isr.c

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_Instance0_Cfg.c (For PC config - Instance 0) - For driver compilation, this file should be generated by the user using a configuration tool
- Wdg_43_Instance1_Cfg.c (For PC config - Instance 1) - For driver compilation, this file should be generated by the user using a configuration tool
- Wdg_43_Instance2_Cfg.c (For PC config - Instance 2) - For driver compilation, this file should be generated by the user using a configuration tool
- Wdg_43_Instance0_Lcfg.c (For LT Variant - Instance 0) - For driver compilation, this file should be generated by the user using a configuration tool
- Wdg_43_Instance1_Lcfg.c (For LT Variant - Instance 1) - For driver compilation, this file should be generated by the user using a configuration tool
- Wdg_43_Instance2_Lcfg.c (For LT Variant - Instance 2) - 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_T2D35M10I0R0 \include\Compiler.h
- ..\Base_TS_T2D35M10I0R0 \include\Compiler_Cfg.h
- ..\Base_TS_T2D35M10I0R0 \include\ComStack_Types.h
- ..\Base_TS_T2D35M10I0R0 \include\MemMap.h
- ..\Base_TS_T2D35M10I0R0 \include\Mcal.h

- ..\Base_TS_T2D35M10I0R0 \include\Platform_Types.h
- ..\Base_TS_T2D35M10I0R0 \include\Std_Types.h
- ..\Base_TS_T2D35M10I0R0 \include\Reg_eSys.h
- ..\Base_TS_T2D35M10I0R0 \include\Soc_Ips.h
- ..\Base_TS_T2D35M10I0R0 \include\SilRegMacros.h

Files from WdgIf folder:

- ..\WdgIf_TS_T2D35M10I0R0 \include\WdgIf_Types.h

Files from Dem folder:

- ..\Dem_TS_T2D35M10I0R0 \include\Dem.h
- ..\Dem_TS_T2D35M10I0R0 \include\Dem_IntErrId.h
- ..\Dem_TS_T2D35M10I0R0 \include\Dem_Types.h

Files from Gpt folder:

- ..\Gpt_TS_T2D35M10I0R0\src\Gpt.c
- ..\Gpt_TS_T2D35M10I0R0\src\Gpt_IPW.c
- ..\Gpt_TS_T2D35M10I0R0\src\Gpt_eMios.c
- ..\Gpt_TS_T2D35M10I0R0\src\Gpt_Pit.c
- ..\Gpt_TS_T2D35M10I0R0\src\Gpt_Stm.c
- ..\Gpt_TS_T2D35M10I0R0\include\Gpt.h
- ..\Gpt_TS_T2D35M10I0R0\include\Gpt_Local.h
- ..\Gpt_TS_T2D35M10I0R0\include\Gpt_EnvCfg.h
- ..\Gpt_TS_T2D35M10I0R0\include\Gpt_IPW.h
- ..\Gpt_TS_T2D35M10I0R0\include\Gpt_IPW_Types.h
- ..\Gpt_TS_T2D35M10I0R0\include\Gpt_IPW_Local.h
- ..\Gpt_TS_T2D35M10I0R0\include\Gpt_eMios.h
- ..\Gpt_TS_T2D35M10I0R0\include\Gpt_eMios_Types.h
- ..\Gpt_TS_T2D35M10I0R0\include\Reg_eSys_eMios.h
- ..\Gpt_TS_T2D35M10I0R0\include\Gpt_Pit.h
- ..\Gpt_TS_T2D35M10I0R0\include\Gpt_Pit_Types.h
- ..\Gpt_TS_T2D35M10I0R0\include\Reg_eSys_Pit.h
- ..\Gpt_TS_T2D35M10I0R0\include\Gpt_Stm.h
- ..\Gpt_TS_T2D35M10I0R0\include\Gpt_Stm_Types.h
- ..\Gpt_TS_T2D35M10I0R0\include\Reg_eSys_Stm.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_T2D35M10I0R0\src\eMios_Common.c
- ..\Mcl_TS_T2D35M10I0R0\include\eMios_Common.h
- ..\Mcl_TS_T2D35M10I0R0\include\eMios_Common_Types.h

Files from Det folder:

- ..\Det_TS_T2D35M10I0R0 \include\Det.h

3.3 Setting up the Plug-ins

All the Autosar MCAL drivers for MPC574XG were designed to be configured using Tresos Studio (version EB tresos Studio 21.0.0 b160607-0933 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_T2D35M10I0R0 \config\Wdg.xdm
  - ..\ Gpt _ TS_T2D35M10I0R0 \config\Gpt.xdm
  - ..\ Mcu _ TS_T2D35M10I0R0 \config\Mcu.xdm
  - ..\ EcuM _ TS_T2D35M10I0R0 \config\EcuM.xdm
  - ..\ EcuC _ TS_T2D35M10I0R0 \config\EcuC.xdm
  - ..\ Dem _ TS_T2D35M10I0R0 \config\Dem.xdm
  - ..\ Resource _ TS_T2D35M10I0R0 \config\Resource.xdm
- VSMD (Vendor Specific Module Definition) file(s) in AUTOSAR compliant EPD format:
  - ..\ Wdg _ TS_T2D35M10I0R0 \autosar\Wdg_<subderivative_name>.epd
  - ..\ Gpt _ TS_T2D35M10I0R0 \autosar\Gpt_<subderivative_name>.epd
  - ..\ Mcu _ TS_T2D35M10I0R0 \autosar\Mcu_<subderivative_name>.epd
  - ..\ EcuM _ TS_T2D35M10I0R0 \autosar\EcuM.epd
  - ..\ EcuC _ TS_T2D35M10I0R0 \autosar\EcuC.epd
  - ..\ Dem _ TS_T2D35M10I0R0 \autosar\Dem.epd
  - ..\ Resource _ TS_T2D35M10I0R0 \autosar \Resource_<subderivative_name>.epd
- Code Generation Templates for Pre-Compile time configuration parameters:
  - ..\ Wdg _ TS_T2D35M10I0R0 \generate\src\Wdg_CfgExt.c
  - ..\ Wdg _ TS_T2D35M10I0R0 \generate\include\Wdg_Cfg.h
  - ..\ Wdg _ TS_T2D35M10I0R0 \generate_PC\src\Wdg_43_Instance0_Cfg.c
  - ..\ Wdg _ TS_T2D35M10I0R0 \generate_PC\src\Wdg_43_Instance1_Cfg.c
  - ..\ Wdg _ TS_T2D35M10I0R0 \generate_PC\src\Wdg_43_Instance2_Cfg.c
  - ..\ Wdg _ TS_T2D35M10I0R0 \generate_PB\src\Wdg_43_Instance0_PBcfg.c
  - ..\ Wdg _ TS_T2D35M10I0R0 \generate_PB\src\Wdg_43_Instance1_PBcfg.c

    - ..\ Wdg _ TS_T2D35M10I0R0 \generate_PB\src\Wdg_43_Instance2_PBcfg.c
    - ..\ EcuM _ TS_T2D35M10I0R0 \generate_PC\include\EcuM_Cfg.h
    - ..\ Dem _ TS_T2D35M10I0R0 \generate_PC\include\Dem_intErrId.h
    - ..\ Gpt _ TS_T2D35M10I0R0 \generate_PC\include\Gpt_Cfg.h
    - ..\ Gpt _ TS_T2D35M10I0R0 \generate_PC\src\Gpt_Cfg.c

- Code Generation Templates for Post-Build time configuration parameters:
    - ..\ Wdg _ TS_T2D35M10I0R0 \generate\src\Wdg_CfgExt.c
    - ..\ Wdg _ TS_T2D35M10I0R0 \generate\include\Wdg_Cfg.h
    - ..\ Wdg _ TS_T2D35M10I0R0 \generate_PB\src\Wdg_43_Instance0_PBcfg.c
    - ..\ Wdg _ TS_T2D35M10I0R0 \generate_PB\src\Wdg_43_Instance1_PBcfg.c
    - ..\ Wdg _ TS_T2D35M10I0R0 \generate_PB\src\Wdg_43_Instance2_PBcfg.c
    - ..\ EcuM _ TS_T2D35M10I0R0 \generate_PC\include\EcuM_Cfg.h
    - ..\ Dem _ TS_T2D35M10I0R0 \generate_PC\include\Dem_intErrId.h
    - ..\ Gpt _ TS_T2D35M10I0R0 \generate_PC\include\Gpt_Cfg.h
    - ..\ Gpt _ TS_T2D35M10I0R0 \generate_PB\src\Gpt_PBCfg.c

- Code Generation Templates for parameters without variation points:
    - ..\ Wdg _ TS_T2D35M10I0R0 \generate\src\Wdg_CfgExt.c
    - ..\ Wdg _ TS_T2D35M10I0R0 \generate\include\Wdg_Cfg.h
    - ..\ Wdg _ TS_T2D35M10I0R0 \generate_PB\src\Wdg_43_Instance0_PBcfg.c
    - ..\ Wdg _ TS_T2D35M10I0R0 \generate_PB\src\Wdg_43_Instance1_PBcfg.c
    - ..\ Wdg _ TS_T2D35M10I0R0 \generate_PB\src\Wdg_43_Instance2_PBcfg.c
    - ..\ EcuM _ TS_T2D35M10I0R0 \generate_PC\include\EcuM_Cfg.h
    - ..\ Dem _ TS_T2D35M10I0R0 \generate_PC\include\Dem_intErrId.h
    - ..\ Gpt _ TS_T2D35M10I0R0 \generate_PC\include\Gpt_Cfg.h

- Code Generation Templates for variant aware parameters:
    - ..\ Wdg _ TS_T2D35M10I0R0 \generate\src\Wdg_CfgExt.c
    - ..\ Wdg _ TS_T2D35M10I0R0 \generate\include\Wdg_Cfg.h
    - ..\ Wdg _ TS_T2D35M10I0R0 \generate_PB\src\Wdg_43_Instance0_PBcfg.c
    - ..\ Wdg _ TS_T2D35M10I0R0 \generate_PB\src\Wdg_43_Instance1_PBcfg.c
    - ..\ Wdg _ TS_T2D35M10I0R0 \generate_PB\src\Wdg_43_Instance2_PBcfg.c
    - ..\ Gpt _ TS_T2D35M10I0R0 \generate_PC\include\Gpt_Cfg.h
    - ..\ Gpt _ TS_T2D35M10I0R0 \generate_PB\src\Gpt_PBCfg.c

Steps to generate the configuration:

1. Copy the module folders Wdg _ TS_T2D35M10I0R0 , Dem_ TS_T2D35M10I0R0 , Base_ TS_T2D35M10I0R0 , Resource_ TS_T2D35M10I0R0 , EcuM_ TS_T2D35M10I0R0, Gpt_ TS_T2D35M10I0R0, Mcl_ TS_T2D35M10I0R0, Mcu_ TS_T2D35M10I0R0 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: MPC5748G_LQFP176, MPC5748G_MAPBGA256, MPC5748G_MAPBGA324, MPC5747G_LQFP176, MPC5747G_MAPBGA256, MPC5747G_MAPBGA324, MPC5746G_LQFP176, MPC5746G_MAPBGA256, MPC5746G_MAPBGA324, MPC5748C_LQFP176, MPC5748C_MAPBGA256, MPC5748C_MAPBGA324, MPC5747C_LQFP176, MPC5747C_MAPBGA256, MPC5747C_MAPBGA324, MPC5746C_LQFP176, MPC5746C_MAPBGA256, MPC5746C_MAPBGA324, MPC5746C_MAPBGA100, MPC5745C_LQFP176, MPC5745C_MAPBGA256, MPC5745C_MAPBGA100, MPC5744C_LQFP176, MPC5744C_MAPBGA256, MPC5744C_MAPBGA100, MPC5746B_LQFP176, MPC5746B_MAPBGA256, MPC5746B_MAPBGA100, MPC5744B_LQFP176, MPC5744B_MAPBGA256, MPC5744B_MAPBGA100, MPC5745B_LQFP176, MPC5745B_MAPBGA256, MPC5745B_MAPBGA100 .
- **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 SWT instances used on the platform, the API names will expand according to AUTOSAR requirement BSW00347. 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

A Software watchdog timer (SWT) with programmable interrupt response is available in MPC574XG. 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 MRSR[SWTR] flag.
- If a time-out occurs, the SWT generates a system reset and sets the MRSR [SWTR] 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

5.3 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_Isr0	32
Wdg_Isr1	33
Wdg_Isr2	34

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

Development Error Tracer:

This module is necessary for enabling Development error detection. The API function used is Det_ReportError(). The activation / deactivation of Development error detection is configurable using 'WdgDevErrorDetect' configuration parameter.

Diagnostic Event Manager:

This module is necessary for enabling reporting of production relevant error status. The API function used is Dem_ReportErrorStatus ().

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

- **GPT** is required to reset periodically the watchdog timeout counter.
- **MCU** module shall be initialized before using GPT .This module is required for setting the global prescaler value and to set the system clock frequency.
- **MCL** module is needed by Gpt functions
- **RESOURCE** is required to select processor derivative. Current driver has support for the following derivatives, everyone having attached a Resource file: MPC5748G_LQFP176, MPC5748G_MAPBGA256, MPC5748G_MAPBGA324, MPC5747G_LQFP176, MPC5747G_MAPBGA256, MPC5747G_MAPBGA324, MPC5746G_LQFP176, MPC5746G_MAPBGA256, MPC5746G_MAPBGA324, MPC5748C_LQFP176, MPC5748C_MAPBGA256, MPC5748C_MAPBGA324, MPC5747C_LQFP176, MPC5747C_MAPBGA256, MPC5747C_MAPBGA324, MPC5746C_LQFP176, MPC5746C_MAPBGA256, MPC5746C_MAPBGA324, MPC5746C_MAPBGA100, MPC5745C_LQFP176, MPC5745C_MAPBGA256, MPC5745C_MAPBGA100, MPC5744C_LQFP176, MPC5744C_MAPBGA256, MPC5744C_MAPBGA100, MPC5746B_LQFP176, MPC5746B_MAPBGA256, MPC5746B_MAPBGA100, MPC5744B_LQFP176, MPC5744B_MAPBGA256, MPC5744B_MAPBGA100, MPC5745B_LQFP176, MPC5745B_MAPBGA256, MPC5745B_MAPBGA100 .
- **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).

Resource Parameters ConfigurationMPC5748G_LQFP176, MPC5748G_MAPBGA256, MPC5748G_MAPBGA324, MPC5747G_LQFP176, MPC5747G_MAPBGA256, MPC5747G_MAPBGA324, MPC5746G_LQFP176, MPC5746G_MAPBGA256, MPC5746G_MAPBGA324, MPC5748C_LQFP176, MPC5748C_MAPBGA256, MPC5748C_MAPBGA324, MPC5747C_LQFP176, MPC5747C_MAPBGA256, MPC5747C_MAPBGA324, MPC5746C_LQFP176, MPC5746C_MAPBGA256, MPC5746C_MAPBGA324, MPC5746C_MAPBGA100, MPC5745C_LQFP176, MPC5745C_MAPBGA256, MPC5745C_MAPBGA100, MPC5744C_LQFP176, MPC5744C_MAPBGA256, MPC5744C_MAPBGA100, MPC5746B_LQFP176, MPC5746B_MAPBGA256, MPC5746B_MAPBGA100, MPC5744B_LQFP176, MPC5744B_MAPBGA256, MPC5744B_MAPBGA100, MPC5745B_LQFP176, MPC5745B_MAPBGA256, MPC5745B_MAPBGA100

5.6 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 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	Constant Data	Start of Memory Section for Config Data.
WDG_STOP_SEC_CONST_8	Constant Data	End of Memory Section for Config Data.
WDG_START_SEC_CONST_UNSPECIFIED	Constant Data	Start of Memory Section for Config Data.
WDG_STOP_SEC_CONST_UNSPECIFIED	Constant 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	Pre Compile
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
	WdgIndex1	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WdgIndex2	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	VariantPC or VariantPB	Post Build
	WdgCallbackNotification1	VariantPC or VariantPB	Post Build
	WdgCallbackNotification2	VariantPC or VariantPB	Post Build
	WdgVersionInfoApi	Pre Compile parameter for all Variants of Configuration	Pre Compile
WdgClockReferencePoint	WdgClockReference	Pre Compile parameter for all Variants of Configuration	Pre Compile
WdgSettingsConfig/General	WdgInstance	Pre Compile parameter for all Variants of Configuration	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
	WdgMasterAccessProtectionforMaster0	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WdgMasterAccessProtectionforMaster1	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WdgMasterAccessProtectionforMaster2	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WdgMasterAccessProtectionforMaster3	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WdgMasterAccessProtectionforMaster4	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
	WdgMasterAccessProtectionforMaster5	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WdgMasterAccessProtectionforMaster6	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WdgMasterAccessProtectionforMaster7	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WdgKeyedService	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WdgServiceKeyValue	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WdgSoftLockConfiguration	VariantPC or VariantPB	Post Build
	WdgHardLockConfiguration	VariantPC or VariantPB	Post Build
	WdgRunsInStopMode	VariantPC or VariantPB	Post Build
	WdgRunsInDebugMode	VariantPC or VariantPB	Post Build
	WdgOperationMode	VariantPC or VariantPB	Post Build
	WdgResetOnInvalidAccess	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
WdgSettingsConfig/ WdgSettingsOff	WdgSoftLockConfiguration	VariantPC or VariantPB	Post Build
	WdgHardLockConfiguration	VariantPC or VariantPB	Post Build
WdgSettingsConfig/ WdgSettingsSlow	WdgClockValue	VariantPC or VariantPB	Post Build
	WdgClkSrcRef	VariantPC or VariantPB	Post Build
	WdgMasterAccessProtectionforMaster0	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WdgMasterAccessProtectionforMaster1	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WdgMasterAccessProtectionforMaster2	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WdgMasterAccessProtectionforMaster3	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WdgMasterAccessProtectionforMaster4	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WdgMasterAccessProtectionforMaster5	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WdgMasterAccessProtectionforMaster6	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WdgMasterAccessProtectionforMaster7	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WdgKeyedService	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
	WdgServiceKeyValue	Pre Compile parameter for all Variants of Configuration	Pre Compile
	WdgSoftLockConfiguration	VariantPC or VariantPB	Post Build
	WdgHardLockConfiguration	VariantPC or VariantPB	Post Build
	WdgRunsInStopMode	VariantPC or VariantPB	Post Build
	WdgRunsInDebugMode	VariantPC or VariantPB	Post Build
	WdgOperationMode	VariantPC or VariantPB	Post Build
	WdgResetOnInvalidAccess	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
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
- Compile & build the Wdg with all the dependent modules. For more details refer to section Building the Driver

Chapter 10
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. >>

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Document Number IM35WDGASR4.2 Rev0002 RTM
1.0.0
Revision 2.3.0