Integration Manual

for MPC574XG GPT Driver

Document Number: IM35GPTASR4.2 Rev0002 R1.0.0
Rev. 5.0.0

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 GPT driver

Chapter 1
Revision History

Table 1-1. Revision History

Revision	Date	Author	Description
1.0.0	22/08/2014	Son Nguyen	Integration Manual for MPC574XG - 0.9.0 Release
2.0.0	24/04/2015	Phap Nguyen	Integration Manual for MPC574XG - 1.0.0 Release
3.0.0	10/07/2015	Phap Nguyen	Integration Manual for CALYPSO - 1.0.1 Release
4.0.0	12/08/2016	Nghia LE	Integration Manual for CALYPSO - 1.0.2 Release
5.0.0	17/02/2017	Nghia Tran Thinh	Integration Manual for CALYPSO AUTOSAR4.2.2 - 1.0.0 Release

Chapter 2
Introduction

This integration manual describes the integration requirements for GPT 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 Acronyms and Definitions

Table 2-2. Acronyms and Definitions

Term	Definition
API	Application Programming Interface
AUTOSAR	Automotive Open System Architecture
ASM	Assembler
BSMI	Basic Software Make file Interface
C/CPP	C and C++ Source Code
DET	Diagnostic Event Traceability
EMIOS	Configurable Enhanced Modular IO Subsystem
GPT	General Purpose Timer
N/A	Not Applicable
MCU	Micro Controller Unit
PIT	Periodic Interrupt Timer
RTI	Real Time Interrupt
RTC	Real Time Clock
STM	System Timer Module
VLE	Variable Length Encoding

2.3 Reference List

Table 2-3. Reference List

#	Title	Version
1	AUTOSAR 4.2 Rev0002GPT Driver Software Specification Document.	V3.2.0
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

Table continues on the next page...

Table 2-3. Reference List (continued)

#	Title	Version
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 GPT driver for NXP SemiconductorMPC574XG . It also explains the EB Tresos Studio plugin setup procedure.

3.1 Build Options

The GPT 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 GHS Compiler/Linker/Assembler Options

Table 3-1. 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 3-2. 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 3-3. 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

3.1.2 DIAB Compiler/Linker/Assembler Options

Table 3-4. 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.

Table continues on the next page...

Table 3-4. Compiler Options (continued)

Option	Description
-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.
-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 3-5. 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

Table continues on the next page...

Table 3-5. Assembler Options (continued)

Option	Description
-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 3-6. 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

3.2 Files required for Compilation

This section describes the include files required to compile, assemble (if assembler code) and link the Autosar GPT driver for NXP Semiconductor's MPC574XG microcontroller.

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.

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.

Plugin Files:

com.freescale.tools.tresos.xpath.jar - This plugin file is necessary to compile the GPT module. It will be installed in the EB Tresos plugins folder.

GPT Files:

..\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_eMios_Irq.c

..\Gpt_TS_T2D35M10I0R0\src\Gpt_Pit.c

..\Gpt_TS_T2D35M10I0R0\src\Gpt_Stm.c

..\Gpt_TS_T2D35M10I0R0\src\Gpt_Rtc.c

..\Gpt_TS_T2D35M10I0R0\include\eMios_Common_Types.h

..\Gpt_TS_T2D35M10I0R0\include\Gpt.h

..\Gpt_TS_T2D35M10I0R0\include\Gpt_Irq.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_Irq.h

..\Gpt_TS_T2D35M10I0R0\include\Gpt_eMios.h

..\Gpt_TS_T2D35M10I0R0\include\Gpt_eMios_Types.h

..\Gpt_TS_T2D35M10I0R0\include\Gpt_eMios_Irq.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_TS_T2D35M10I0R0\include\Gpt_Rtc.h

..\Gpt_TS_T2D35M10I0R0\include\Gpt_Rtc_Types.h

..\Gpt_TS_T2D35M10I0R0\include\Reg_eSys_Rtc.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

Other include files:

Files from Base folder:

..\Base_TS_T2D35M10I0R0\include\MemMap.h

..\Base_ TS_T2D35M10I0R0\include\Platform_Types.h

..\Base_TS_T2D35M10I0R0\include\Std_Types.h

..\Base_TS_T2D35M10I0R0\include\Soc_Ips.h

..\Base_TS_T2D35M10I0R0\include\Reg_eSys.h

..\Base_TS_T2D35M10I0R0\include\SilRegMacros.h

..\Base_TS_T2D35M10I0R0\include\Compiler.h

..\Base_ TS_T2D35M10I0R0\include\Mcal.h

Files from Det folder:

..\Det_TS_T2D35M10I0R0\include\Det.h

Files from Mcl folder:

..\Mcl_TS_T2D35M10I0R0\src\CDD_Mcl.c

..\Mcl_TS_T2D35M10I0R0\src\eMios_Common.c

..\Mcl_TS_T2D35M10I0R0\src\Mcl_Dma.c

..\Mcl_TS_T2D35M10I0R0\src\Mcl_Dma_Irq.c

..\Mcl_TS_T2D35M10I0R0\src\Mcl_DmaMux.c

..\Mcl_TS_T2D35M10I0R0\src\Mcl_IPW.c

..\Mcl_TS_T2D35M10I0R0\include\CDD_Mcl.h

..\Mcl_TS_T2D35M10I0R0\include\eMios_Common.h

..\Mcl_TS_T2D35M10I0R0\include\eMios_Common_Types.h

..\Mcl_TS_T2D35M10I0R0\include\Mcl.h

..\Mcl_TS_T2D35M10I0R0\include\Mcl_Dma.h

..\Mcl_TS_T2D35M10I0R0\include\Mcl_Dma_Types.h

..\Mcl_TS_T2D35M10I0R0\include\Mcl_DmaMux_Types.h

..\Mcl_TS_T2D35M10I0R0\include\Mcl_EnvCfg.h

..\Mcl_TS_T2D35M10I0R0\include\Mcl_IPW.h

..\Mcl_TS_T2D35M10I0R0\include\Mcl_IPW_Notif.h

..\Mcl_TS_T2D35M10I0R0\include\Mcl_IPW_Types.h

..\Mcl_TS_T2D35M10I0R0\include\Mcl_Notif.h

..\Mcl_TS_T2D35M10I0R0\include\Mcl_Types.h

..\Mcl_TS_T2D35M10I0R0\include\Reg_eSys_Dma.h

..\Mcl_TS_T2D35M10I0R0\include\Reg_eSys_DmaMux.h

..\Mcl_TS_T2D35M10I0R0\include\Reg_eSys_eMios.h

Files from EcuM folder:

..\EcuM_TS_T2D35M10I0R0\include\EcuM.h

..\EcuM_TS_T2D35M10I0R0\include\EcuM_Cbk.h

3.3 Setting up the Plug-ins

All the Autosar MCAL drivers for MPC574XG were designed to be configured using Tresos® Studio configuration and code generation tool from EB tresos Studio 21.0.0 b160607-0933.

Location of various files inside the plugin folder is explained below.

- VSMD (Vendor Specific Module Definition) file in EB tresos Studio XDM format:
  - ..\GPT_TS_T2D35M10I0R0\config\Gpt.xdm
  - ..\Mcu_TS_T2D35M10I0R0\config\Mcu.xdm
  - ..\EcuC_TS_T2D35M10I0R0\config\EcuC.xdm

    - ..\EcuM_TS_T2D35M10I0R0\config\EcuM.xdm
    - ..\Resource_TS_T2D35M10I0R0\config\Resource.xdm

- VSMD (Vendor Specific Module Definition) file(s) in AUTOSAR compliant EPD format:
    - ..\GPT_TS_T2D35M10I0R0\autosar\Gpt_<subderivative_name>.epd
    - ..\Mcu_TS_T2D35M10I0R0\autosar\Mcu.epd
    - ..\Resource_TS_T2D35M10I0R0\autosar\Resource_<subderivative_name>.epd

- Code Generation Templates for Pre-Compile time configuration parameters:
    - ..\GPT_TS_T2D35M10I0R0\output\src\Gpt_PBCfg.c
    - ..\GPT_TS_T2D35M10I0R0\output\include\Gpt_Cfg.h
    - ..\Mcu_TS_T2D35M10I0R0\output\src\Mcu_PBCfg.c
    - ..\Mcu_TS_T2D35M10I0R0\output\include\Mcu_Cfg.h
    - ..\EcuM_TS_T2D35M10I0R0\output\src\EcuM_Cfg.h

- Code Generation Templates for Post-Build time configuration parameters:
    - ..\GPT_TS_T2D35M10I0R0\output\src\Gpt_PBCfg.c
    - ..\GPT_TS_T2D35M10I0R0\output\include\Gpt_Cfg.h
    - ..\Mcu_TS_T2D35M10I0R0\output\src\Mcu_PBCfg.c
    - ..\Mcu_TS_T2D35M10I0R0\output\include\Mcu_Cfg.h
    - ..\EcuM_TS_T2D35M10I0R0\output\src\EcuM_Cfg.h

Steps to generate configurations:

1.Copy the module folder (Base_TS_T2D35M10I0R0), (Gpt_TS_T2D35M10I0R0), (EcuM_TS_T2D35M10I0R0), (Mcu_TS_T2D35M10I0R0), (Mcl_TS_T2D35M10I0R0), (Resource_TS_T2D35M10I0R0), (Dem_TS_T2D35M10I0R0), (Det_TS_T2D35M10I0R0), (Rte_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 Tresos GUI to modify configuration parameters values.

4.Generate the Pre-Compile and Post-Build files.

Chapter 4
Function calls to module

4.1 Function Calls during Start-up

GPT shall be initialized during STARTUP1 phase of EcuM initialization. The API to be called for this is Gpt_Init() MCU module shall be initialized before GPT is initialized.

4.2 Function Calls during Shutdown

If GptWakeupFunctionalityApi and GptWakeupSourceRef are enabled, Gpt_SetMode(GPT_MODE_SLEEP) API shall be called during GO SLEEP phase of EcuM to configure the hardware for Sleep mode.

4.3 Function Calls during Wake-up

For the platforms where the GPT driver controls wakeup hw sources, if the GptWakeupFunctionalityApi and GptWakeupSourceRef are enabled, the driver shall report the wakeup event to EcuM through EcuM_SetWakeupEvent(Source) upon the hw source event.

Chapter 5
Module requirements

5.1 Peripheral Hardware Requirements

Driver implements 126 channels on 4 MPC574XG peripherals.

- 96 channels are implemented on the Enhanced Modular IO Subsystem (eMIOS) modules, 32 channels are implemented on each module and 3 modules for the platform.

- 12 channels are implemented on System Timer Module (STM) modules, 4 channels are implemented on each module and 3 modules for the platform.

- 17 channels are implemented on Interrupt Timer (PIT) modules, 16 PIT channels on PIT_0 module and a RTI channel on PIT_RTI module.

- 1 channels are implemented on 1 Real Time Clock (RTC) modules

Refer Table GPT Hardware Channel availability for MPC574XG family in User Manual

5.2 ISR to configure within OS – dependencies

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

Table 5-1. GPT ISR's

ISR Name	Hardware interrupt vector
For EMIOS0 Timers	
ISR(EMIOS_0_CH_0_CH_1_ISR)	706
ISR(EMIOS_0_CH_2_CH_3_ISR)	707
ISR(EMIOS_0_CH_4_CH_5_ISR)	708
ISR(EMIOS_0_CH_6_CH_7_ISR)	709
ISR(EMIOS_0_CH_8_CH_9_ISR)	710

Table continues on the next page...

Table 5-1. GPT ISR's (continued)

ISR Name	Hardware interrupt vector
ISR(EMIOS_0_CH_16_CH_17_ISR)	714
ISR(EMIOS_0_CH_22_CH_23_ISR)	717
ISR(EMIOS_0_CH_24_CH_25_ISR)	718
For EMIOS1 Timers	
ISR(EMIOS_1_CH_0_CH_1_ISR)	722
ISR(EMIOS_1_CH_8_CH_9_ISR)	726
ISR(EMIOS_1_CH_16_CH_17_ISR)	730
ISR(EMIOS_1_CH_22_23_ISR)	733
ISR(EMIOS_1_CH_24_CH_25_ISR)	734
For PIT Timers	
ISR(Gpt_PIT_0_TIMER_0_ISR)	226
ISR(Gpt_PIT_0_TIMER_1_ISR)	227
ISR(Gpt_PIT_0_TIMER_2_ISR)	228
ISR(Gpt_PIT_0_TIMER_3_ISR)	229
ISR(Gpt_PIT_0_TIMER_4_ISR)	230
ISR(Gpt_PIT_0_TIMER_5_ISR)	231
ISR(Gpt_PIT_0_TIMER_6_ISR)	232
ISR(Gpt_PIT_0_TIMER_7_ISR)	233
ISR(Gpt_PIT_0_TIMER_8_ISR)	234
ISR(Gpt_PIT_0_TIMER_9_ISR)	235
ISR(Gpt_PIT_0_TIMER_10_ISR)	236
ISR(Gpt_PIT_0_TIMER_11_ISR)	237
ISR(Gpt_PIT_0_TIMER_12_ISR)	238
ISR(Gpt_PIT_0_TIMER_13_ISR)	239
ISR(Gpt_PIT_0_TIMER_14_ISR)	240
ISR(Gpt_PIT_0_TIMER_15_ISR)	241
ISR(Gpt_PIT_0_TIMER_PITRTI_ISR)	242
For STM0 Timers	
ISR(Gpt_STM_0_Ch_0_ISR)	36
ISR(Gpt_STM_0_Ch_1_ISR)	37
ISR(Gpt_STM_0_Ch_2_ISR)	38
ISR(Gpt_STM_0_Ch_3_ISR)	39
For STM1 Timers	
ISR(Gpt_STM_1_Ch_0_ISR)	40
ISR(Gpt_STM_1_Ch_1_ISR)	41
ISR(Gpt_STM_1_Ch_2_ISR)	42
ISR(Gpt_STM_1_Ch_3_ISR)	43
For STM2 Timers	
ISR(Gpt_STM_2_Ch_0_ISR)	44

Table continues on the next page...

Table 5-1. GPT ISR's (continued)

ISR Name	Hardware interrupt vector
ISR(Gpt_STM_2_Ch_1_ISR)	45
ISR(Gpt_STM_2_Ch_2_ISR)	46
ISR(Gpt_STM_2_Ch_3_ISR)	47
For RTC	
ISR(Gpt_RTC_0_Ch_0_ISR)	225

5.3 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:

1. OS is not used - AUTOSAR_OS_NOT_USED is defined:
   - 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.
   - 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.
2. NXP SemiconductorOS 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.
3. 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.

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

5.4 Other AUTOSAR modules - Dependencies

5.4.1 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 ‘GptDevErrorDetect’ configuration parameter.

5.4.2 Diagnostic Event Manager:

This module is necessary for enabling reporting of production relevant error status. This is not used with current GPT implementation as the production relevant error codes are not present.

5.4.3 ECU Manager:

This module is used for processing the Wakeup notifications of GPT. Whenever the module is in ‘Sleep’ mode and a wakeup event occurs on a wakeup capable channel, it is reported to EcuM by calling EcuM_SetWakeupEvent () API through the EcuM_CheckWakeup()API. This is configurable using the ‘GptReportWakeupSource’ configuration parameter.

5.4.4 MCL module:

This module is used to obtain the common interrupts sources.

5.4.5 MCU module:

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.

5.4.6 Configuration dependency to other module:

Dependency between GPT, ICU and PWM.

For generating configuration files of GPT, EcuM also is required as GPT refers to EcuM parameter. EcuM need to be configure first before generating configuration files of GPT.

Hence template files for EcuM are provided at

..\EcuM_TS_T2D35M10I0R0\autosar\EcuM.epd (Module Parameter Definition File – AUTOSAR Format)

..\EcuM_TS_T2D35M10I0R0\config\EcuM.xdm (Module Parameter Definition File – Tresos Format)

5.5 Exclusive areas to be defined in BSW scheduler

GPT_EXCLUSIVE_AREA_00 Used in function Gpt_Stm_ProcessCommonInterrupt to protect the updates to:

- STM_CIR_ADDR32

GPT_EXCLUSIVE_AREA_01 Used in function Gpt_Stm_StartTimer to protect the updates to:

- STM_CCR_ADDR32[]

GPT_EXCLUSIVE_AREA_02 Used in function Gpt_Stm_StopTimer to protect the updates to:

- STM_CCR_ADDR32
- STM_CIR_ADDR32

GPT_EXCLUSIVE_AREA_03 Used in Gpt_Pit_ProcessCommonInterrupt function to protect the updates to:

- PIT_TFLG_ADDR32

GPT_EXCLUSIVE_AREA_04 Used in Gpt_Pit_StartTimer function to protect the updates to:

- PIT_TCTRL_LOCKABLE_ADDR32

GPT_EXCLUSIVE_AREA_05 Used in Gpt_Pit_StopTimer function to protect the updates to:

- PIT_TFLG_ADDR32
- PIT_TCTRL_LOCKABLE_ADDR32

GPT_EXCLUSIVE_AREA_06 Used in Gpt_Pit_EnableInterrupt function to protect the updates to:

- PIT_TFLG_ADDR32
- PIT_TCTRL_LOCKABLE_ADDR32

GPT_EXCLUSIVE_AREA_07 Used in Gpt_Pit_DisableInterrupt function to protect the updates to:
- PIT_TFLG_ADDR32
- PIT_TCTRL_LOCKABLE_ADDR32

GPT_EXCLUSIVE_AREA_08 Used in Gpt_Rtc_StartTimer function to protect the updates to:
- RTC_CTRL_ADDR32

GPT_EXCLUSIVE_AREA_09 Used in Gpt_Rtc_StopTimer function to protect the updates to:
- RTC_CTRL_ADDR32

GPT_EXCLUSIVE_AREA_10 Used in Gpt_Rtc_SetClockMode function to protect the updates to:
- RTC_CTRL_ADDR32

GPT_EXCLUSIVE_AREA_11 Used in Gpt_Rtc_EnableInterrupt function to protect the updates to:
- RTC_CTRL_ADDR32

GPT_EXCLUSIVE_AREA_12 Used in Gpt_Rtc_DisableInterrupt function to protect the updates to:
- RTC_CTRL_ADDR32

GPT_EXCLUSIVE_AREA_13 Used in Gpt_eMios_StartTimer function to protect the updates to:
- EMIOS_CCR_ADDR32

GPT_EXCLUSIVE_AREA_14 Used in Gpt_eMios_StopTimer function to protect the updates to:
- EMIOS_CCR_ADDR32

GPT_EXCLUSIVE_AREA_15 Used in Gpt_eMios_EnableInterrupt function to protect the updates to:
- EMIOS_CCR_ADDR32

GPT_EXCLUSIVE_AREA_16 Used in Gpt_eMios_DisableInterrupt function to protect the updates to:
- EMIOS_CCR_ADDR32

GPT_EXCLUSIVE_AREA_17 Used in Gpt_eMios_SetClockMode function to protect the updates to:
- EMIOS_CCR_ADDR32

Critical Region Exclusive Matrix

Please see more detail in AUTOSAR_MCAL_GPT_EXCLUSIVE_AREAS.xlsx file that was in design folder.

Chapter 6
Main API Requirements

6.1 Main functions calls within BSW scheduler

None

6.2 Calls to notification functions, callbacks, callouts

6.2.1 Call-back Notifications:

There are no call-back notifications defined inside the GPT driver.

6.2.2 User Notification:

The GPT Driver provides a notification per channel 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 Gpt_Notification_<channel>

(

void

)

An extern declaration of this function is available in Gpt_PBcfg.c. The function has to be implemented by the user.

Chapter 7
Memory Allocation

7.1 Sections to be defined in Gpt_MemMap.h

Tables descibe Sections to be defined in Gpt_MemMap.h:

Table 7-1. Sectionto be define

<Section name>	Tyep of section	Description
GPT_START_SEC_CONFIG_DATA_<ALIGNMENT>	Configuration Data	Start of Memory Section for Config Data.
GPT_STOP_SEC_CONFIG_DATA_<ALIGNMENT>	Configuration Data	End of Memory Section for Config Data.
GPT_START_SEC_CODE	Code	Start of memory Section for Code in Flash.
GPT_STOP_SEC_CODE	Code	Stop of memory Section for Code in Flash.
GPT_START_SEC_RAMCODE	Code	Start of memory Section for Code in Ram.
GPT_STOP_SEC_RAMCODE	Code	Stop of memory Section for Code in Ram.
GPT_START_SEC_VAR_< INIT_POLICY>_<ALIGNMENT>	Variables	Start of memory Section for Variables.
GPT_STOP_SEC_VAR_ <INIT_POLICY>_<ALIGNMENT>	Variables	Stop of memory Section for Variables.
GPT_START_SEC_CONST_<ALIGNMENT>	Constant data	Start of memory Section for Constant.
GPT_STOP_SEC_CONST_<ALIGNMENT>	Constant data	Stop of memory Section for Constant.

Which the shortcut '<ALIGNMENT >' means the variable alignment. In order to avoid memory gaps in the allocation variables are allocated according their size. Possible ALIGNMENT postfixes are described in the table at the end of this section.

The shortcut '<INIT_POLICY>' means the initialization policy of variables. Possible '<INIT_POLICY>' postfixes are described in the table at the end of this section.

Tables descibe value range of shortcut ALIGMENT, INIT_POLICY:

Table 7-2. Range of <ALIGNMENT>

<ALIGMENT>	Description
BOOLEAN	Used for variables and constants of size 1 bit
8	Used for variables and constants which have to be aligned to 8 bit. For instance used for variables of size 8 bit or used for composite data types: arrays, structs and unions containing elements of maximum 8 bits
16	Used for variables and constants 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 and unions containing elements of maximum 16 bits
32	Used for variables and constants which have to be aligned to 32 bit. For instance used for variables of size 32 bit or used for composite data types: arrays, structs and unions containing elements of maximum 32 bits
UNSPECIFIED	Used for variables, constants, structure, array and unions when SIZE (alignment) does not fit the criteria of 8,16 or 32 bit. For instance used for variables of unknown size

Table 7-3. Range of <INIT_POLICY>

<INIT_POLICY>	Description
NO-INIT	Used for variables that are never cleared and never initialized by start up code (BSS)
INIT	Used for variables that are initialized with values after every reset

7.2 Linker command file

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

Chapter 8
Configuration parameters considerations

Configuration parameter class for Autosar GPT 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
Gpt	IMPLEMENTATION_CONFIG_VARIANT	Pre Compile parameter for all Variants of Configuration	Pre Compile
GptChannelConfigSet/ GptChannelConfiguration	GptChannelId	VariantPC or VariantPB	VariantPB
	GptHwChannel	VariantPC or VariantPB	VariantPB
	GptChannelMode	VariantPC or VariantPB	VariantPB
	GptChannelTickFrequency	VariantPC or VariantPB	VariantPB
	GptChannelClkSrcRef	VariantPC or VariantPB	VariantPB
	GptRtcChannelClkSrc	VariantPC or VariantPB	VariantPB
	GptStmChannelClkSrc	VariantPC or VariantPB	VariantPB
	GptChannelPrescaler	VariantPC or VariantPB	VariantPB
	GptChannelPrescalerAlternate	VariantPC or VariantPB	VariantPB
	GptChannelTickValueMax	VariantPC or VariantPB	VariantPB
	GptFreezeEnable	VariantPC or VariantPB	VariantPB
	GptEnableWakeup	VariantPC or VariantPB	VariantPB
	GptNotification	VariantPC or VariantPB	VariantPB
GptNonAUTOSAR	GptEnableDualClockMode	VariantPC or VariantPB	VariantPB
	GPT Enable User Mode Support	VariantPC or VariantPB	VariantPB
GptConfigurationOfOptApiServices	GptDeinitApi	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
	GptEnableDisableNotificationApi	Pre Compile parameter for all Variants of Configuration	Pre Compile
	GptTimeElapsedApi	Pre Compile parameter for all Variants of Configuration	Pre Compile
	GptTimeRemainingApi	Pre Compile parameter for all Variants of Configuration	Pre Compile
	GptVersionInfoApi	Pre Compile parameter for all Variants of Configuration	Pre Compile
	GptWakeupFunctionalityApi	Pre Compile parameter for all Variants of Configuration	Pre Compile
	GptPredefTimerFunctionalityApi	Pre Compile parameter for all Variants of Configuration	Pre Compile
GptPredefTimerConfiguration/ GptPredefTimer_1us_16Bit	GptHwChannel	VariantPC or VariantPB	VariantPB
	GptChannelClkSrcRef	VariantPC or VariantPB	VariantPB
	GptChannelPrescaler	VariantPC or VariantPB	VariantPB
	GptStmChannelClkSrc	VariantPC or VariantPB	VariantPB
	GptFreezeEnable	VariantPC or VariantPB	VariantPB
GptPredefTimerConfiguration/ GptPredefTimer_1us_24Bit	GptHwChannel	VariantPC or VariantPB	VariantPB
	GptChannelClkSrcRef	VariantPC or VariantPB	VariantPB
	GptChannelPrescaler	VariantPC or VariantPB	VariantPB
	GptStmChannelClkSrc	VariantPC or VariantPB	VariantPB
	GptFreezeEnable	VariantPC or VariantPB	VariantPB
GptPredefTimerConfiguration/ GptPredefTimer_1us_32Bit	GptHwChannel	VariantPC or VariantPB	VariantPB
	GptChannelClkSrcRef	VariantPC or VariantPB	VariantPB
	GptChannelPrescaler	VariantPC or VariantPB	VariantPB
	GptStmChannelClkSrc	VariantPC or VariantPB	VariantPB
	GptFreezeEnable	VariantPC or VariantPB	VariantPB
GptPredefTimerConfiguration/ GptPredefTimer_100us_32Bit	GptHwChannel	VariantPC or VariantPB	VariantPB
	GptChannelClkSrcRef	VariantPC or VariantPB	VariantPB
	GptChannelPrescaler	VariantPC or VariantPB	VariantPB
	GptStmChannelClkSrc	VariantPC or VariantPB	VariantPB
	GptFreezeEnable	VariantPC or VariantPB	VariantPB
GptDriverConfiguration	GptDevErrorDetect	Pre Compile parameter for all Variants of Configuration	Pre Compile
	GptPredefTimer100us32bitEnable	Pre Compile parameter for all Variants of Configuration	Pre Compile
	GptPredefTimer1usEnablingGrade	Pre Compile parameter for all Variants of Configuration	Pre Compile
	GptReportWakeupSource	Pre Compile parameter for all Variants of Configuration	Pre Compile
	GptRegisterLocking	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
	Gpt Timeout	Pre Compile parameter for all Variants of Configuration	Pre Compile
	Gpt Disable Production Error Reporting	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 General Purpose Timer :

- Generate the required GPT configurations. For more details refer to section
- Allocate proper memory sections in GPT_MemMap.h and linker command file. For more details refer to section
- Compile & build the GPT with all the dependent modules. For more details refer to section Building the Driver

Chapter 10
External Assumptions for GPT driver

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

[SMCAL_CPR_EXT40]

<< The application shall not preempt a channel related function (like starting/stoping a timer) by calling Gpt_SetMode() or Gpt_DeInit(). >>

[SMCAL_CPR_EXT41]

<< The application shall not preempt a GPT function working on a GPT channel by calling another GPT function targeting the same channel. >>

[SMCAL_CPR_EXT42]

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

NOTE

A transversal GPT functions are those functions addressing the entire set of channels, like Gpt_Init(), Gpt_DeInit(), Gpt_SetMode(), Gpt_PeriodicCheck(), ...

[SMCAL_CPR_EXT43]

<< The application shall not call any function of the GPT module before having called Gpt_Init. >>

[SMCAL_CPR_EXT44]

<< Wakeup enabled timers shall be started or stopped only when GPT driver is in GPT_MODE_NORMAL mode.The external application shall invoke Gpt_EnableWakeup() and Gpt_DisableWakeup() only when GPT driver is in GPT_MODE_NORMAL mode. >>

NOTE

If Gpt_EnableWakeup(), Gpt_DisableWakeup(), Gpt_StartTimer() and Gpt_StopTimer() are called while GPT is already in SLEEP mode, the GPT driver behavior is not guaranteed. Therefore any wakeup channel configuration shall be done before entering in sleep mode.

[SMCAL_CPR_EXT163]

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

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Document Number IM35GPTASR4.2 Rev0002 R1.0.0
Revision 5.0.0