MCAL User Manual for GPT

32-bit TriCore™ AURIX™ TC3xx microcontroller

About this document

Scope and purpose

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

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

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

Intended audience

This document is intended for anyone using the Gpt module of the TC3xx MCAL software.

Document conventions

Table 1 Conventions

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

Reference documents

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

- AURIX™ TC3xx User Manual General
- Specification of GPT Driver, AUTOSAR_SWS_GPT_Driver, AUTOSAR Release 4.2.2
- Specification of GPT Driver, AUTOSAR_SWS_GPT_Driver, AUTOSAR Release 4.4.0

Table of contents

1 Gpt driver

1.1 User information

1.1.1 Description

The GPT driver is responsible for providing APIs for standard timer functions specified by AUTOSAR. The underlying timer engines are GTM timer channel [TOM/ATOM slice] and GPT12 timers that are available in AURIX2G controllers. The user can configure the driver with multiple channels and the following operations can be performed on each channel:

- Start / Stop the timer
- Enable/Disable wakeup functionality
- Enable / Disable notification
- One Shot or Continuous mode of operation
- GPT Predef Timer feature implemented using TOM slices.

Apart from the configurable parameters specified by AUTOSAR, the GPT driver also provides additional parameters to configure the GTM timer slice and GPT12 timers.

1.1.2 Hardware-software mapping

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

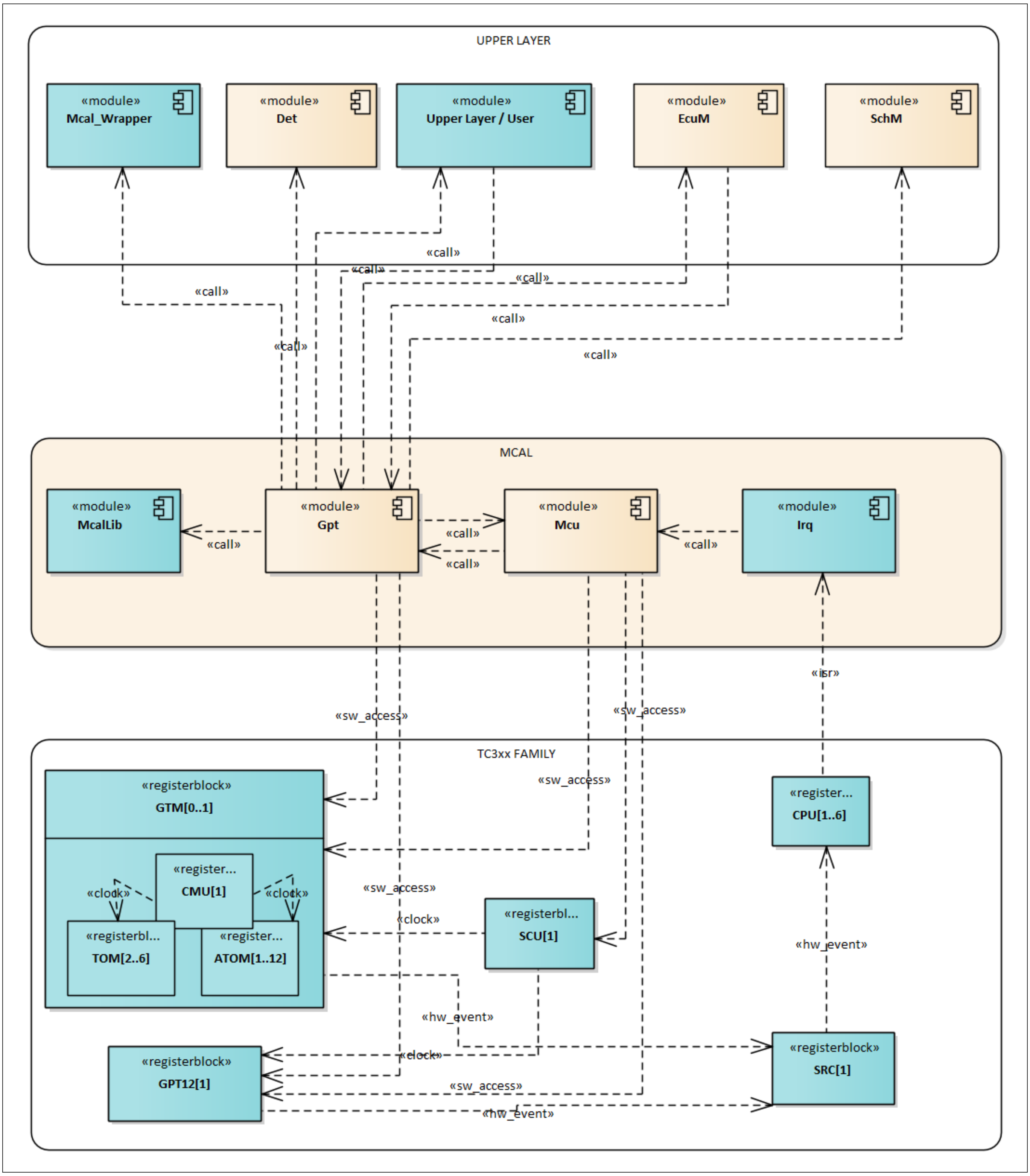

Figure 1 Mapping of hardware-software interfaces

1.1.2.1 GPT12: primary hardware peripheral

Hardware functional features

The GPT driver uses the GPT12 IP for realizing the continuous timer mode and one-shot timer mode.

Implementation of a continuous mode logical channel using GPT1 block needs two timers, core timer T3 and one of the auxillary timer(T2 or T4). Implementation of a continuous mode logical channel using GPT2 block needs only T6(uses CAPREL register as reload register).

Implementation of one-shot mode logical channel needs only one timer from either of GPT1 or GPT2 block.

Predef timer is not supported by GPT12 IP.

The key hardware functional feature of GPT12 IP used by the GPT driver is:

- Timer mode
- Reload functionality

The unsupported features of the GPT12 IP are:

- Gated Timer Mode
- Counter mode
- Incremental Interface Mode
- Capture functionality

Users of the hardware

The GPT12 IP is shared between the GPT and the ICU drivers. The MCU driver provides an API to initialize the GPT12 control registers. The GPT driver uses this API to initialize the GPT12 control registers. Updates to channel-specific SFRs are performed by the GPT driver. Since the GPT12 timer/s used by GPT driver are exclusively reserved for the GPT channels, access to the channel-specific SFRs from other drivers or user software is not allowed.

Hardware diagnostic features

Not applicable.

Hardware events

The GPT driver uses the following hardware event from the GPT12 IP:

- Timer counter underflow event to provide notification when target time is elapsed.

1.1.2.2 GTM: primary hardware peripheral

Hardware functional features

The GPT driver uses the GTM IP for realizing the continuous timer mode, one-shot timer mode and predef timers.

Continuous and one-shot mode logical channels need only one timer channel from TOM/ATOM slice. Predef timers use only the TOM slice of the GTM IP because TOM is 16-bit timer and derivation of 16-, 24- and 32-bits are easier compared to ATOM which is 24 bit. If the fGTM is configured such that, the required frequency for the particular timer(1MHz for 1us predef timer and 10KHz for 100us predef timer) can be directly derived , then the 16-bit predef timer needs only one TOM channel(x) and the 24- or 32- bits predef timers need two TOM channels(x and x+1) respectively. The TRIGOUT feature of the hardware where a previous TOM channel triggers the next TOM channel on a compare match event, is used in case of 24-/32-bits timer. If the fGTM is configured such that, the required frequency cannot be directly derived, then an extra channel(x-1, apart from the normal number of channels needed) is used as a prescalar to derive the required tick frequency for the predef timer. Here also the TRIGOUT feature of the hardware is used to achieve the 1us/100us tick frequency.

For example, if both 24-/32-bit 1us and 100us predef timers are configured and the fGTM is configured such that the required frequency for the particular timer cannot be directly derived, then a total of six TOM channels are required.

The key hardware functional features of GTM-TOM/ATOM IP used by the GPT driver are:

- Continuous counting up mode
- One-shot counting up mode
- ATOM signal output mode PWM (SOMP)

The unsupported features of the GTM-TOM/ATOM IP are:

- Duty cycle, period and clock frequency update mechanisms
- Continuous counting up-down mode
- One-shot counting up-down mode
- Pulse count modulation mode
- Trigger generation
- ATOM signal output mode immediate (SOMI)
- ATOM signal output mode compare (SOMC)
- ATOM signal output mode serial (SOMS)
- ATOM signal output mode buffered compare (SOMB)

Users of the hardware

The GTM-TOM/ATOM IP is shared by multiple drivers such as GPT, OCU, PWM, WDG and ADC. The MCU driver provides APIs to program the GTM SFRs. The GPT driver uses these APIs to write the GTM SFRs. Additionally, updates to channel-specific SFRs are performed by the GPT driver. Since these channels are exclusively reserved for the GPT driver, access to the channel-specific SFRs from other drivers or user software is not allowed.

Hardware diagnostic features

Not applicable.

Hardware events

The GPT driver uses the following hardware event from the GTM-TOM/ATOM IP:

- Period match interrupt to provide notification when target time is elapsed.

1.1.2.3 SCU: dependent hardware peripheral

Hardware functional features

The GPT driver depends upon the SCU IP for the clock. The driver requires the fGTM and fSPB clock signals for functioning.

Users of the hardware

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

Hardware diagnostic features

The SMU alarms configured for the SCU IP are not monitored by the GPT driver.

Hardware events

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

1.1.3 File structure

1.1.3.1 C file structure

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

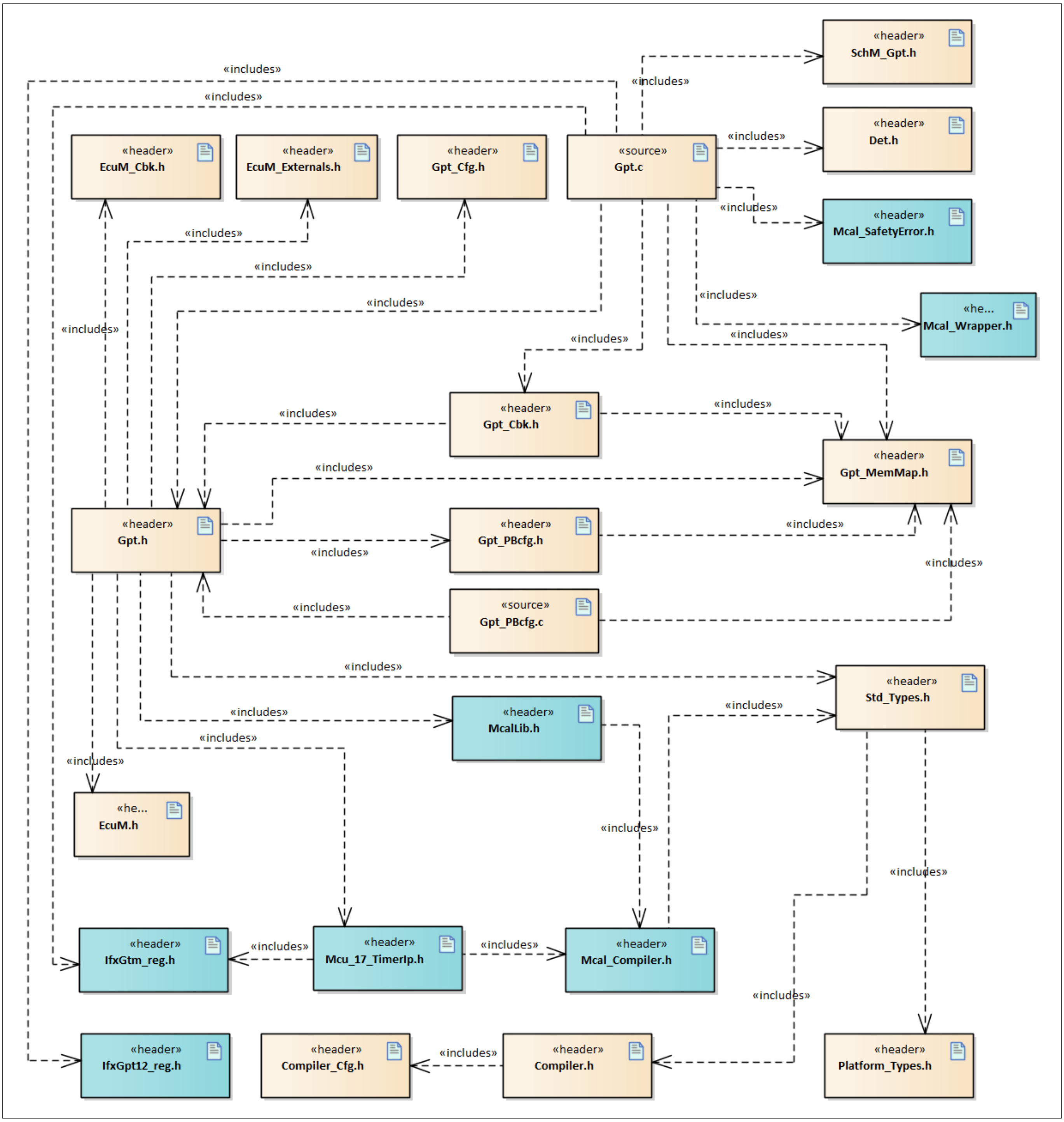

Figure 2 Gpt_C_File_Structure-1.png

Table 2 C file structure

File name	Description
`Compiler.h`	Provides abstraction from compiler-specific keywords
`Compiler_Cfg.h`	Configuration header file for compiler abstraction
`Det.h`	Provides the exported interfaces of Development Error Tracer
`EcuM.h`	Header file exporting the declarations of the EcuM
`EcuM_Cbk.h`	Header file containing declarations of the EcuM callbacks. *Note: This file is available only for AUTOSAR version 4.2.2*
`EcuM_Externals.h`	Header file containing declarations of the EcuM callbacks. *Note: This file is available only for AUTOSAR version 4.4.0*
`Gpt.c`	C file providing implementation of APIs
`Gpt.h`	Header file providing prototypes of APIs and data types
`Gpt_Cbk.h`	Header file providing prototypes of callback APIs
`Gpt_Cfg.h`	Generated header file containing definitions for all Pre-compile time configuration parameters defined as pre-processor directive (#define) for the Gpt driver.
`Gpt_MemMap.h`	File containing the memory section definitions used by the GPT driver.
`Gpt_PBcfg.c`	Generated header file containing configuration data of the user
`Gpt_PBcfg.h`	File (Generated) containing declaration of the post-build configuration data structures
`IfxGpt12_reg.h`	SFR header file for GPT12
`IfxGtm_reg.h`	SFR header file for GTM
`McalLib.h`	Static header file defining prototypes of data structure and APIs exported by the MCALLIB.
`Mcal_Compiler.h`	Header file providing abstraction for TriCore™-intrinsic instruction.
`Mcal_SafetyError.h`	Header file containing the prototype of the API for reporting safety-related errors
`Mcal_Wrapper.h`	Provides the exported interfaces for Production Error and Runtime Development Errors. Implemented by default to include functions of Dem.h and Det.h files. This file can be modified by the user but function prototype is not user modifiable.
`Mcu_17_TimerIp.h`	Header file defining prototypes of data structures and APIs of Timer IPs (GTM, CCU6 and GPT12), containing functions such as initialization, enable, interrupt handlers and other services and is included by Mcu_17_TimerIp.c source file
`Platform_Types.h`	Platform-specific type declaration file as defined by AUTOSAR
`SchM_Gpt.h`	File containing the critical sections declarations
`Std_Types.h`	Standard type declaration file as defined by AUTOSAR. It is independent of compiler or platform.

1.1.3.2 Code generator plugin files

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

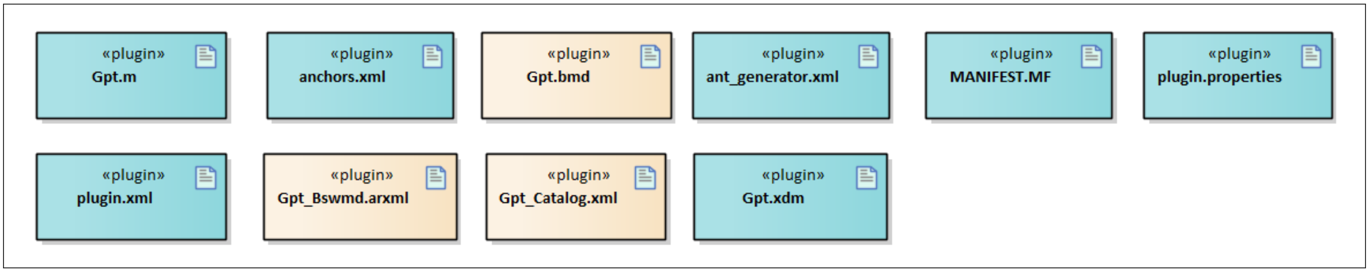

Figure 3 **Gpt_Code_Generator_Plugin_Files-1.png**

Table 3 **Code generator plugin files**

File name	Description
`Gpt.bmd`	Code template macro file for the GPT driver
`Gpt.m`	Code template macro file for the GPT driver
`Gpt.xdm`	Tresos format XML data model schema file
`Gpt_Bswmd.arxml`	AUTOSAR format module description file
`Gpt_Catalog.xml`	AUTOSAR format catalog file
`MANIFEST.MF`	Tresos plugin support file containing the metadata for the GPT driver
`anchors.xml`	AUTOSAR format module description file
`ant_generator.xml`	Tresos support file to generate and rename multiple post-build configuration when using variation point
`plugin.properties`	Tresos plugin support file for the GPT driver
`plugin.xml`	Tresos plugin support file for the GPT driver.

1.1.4 Integration hints

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

1.1.4.1 Intergration with AUTOSAR stack

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

- **EcuM**

  The ECU Manager module is a part of the AUTOSAR stack that manages common aspects of ECU. Specifically, in the context of MCAL, EcuM is used for initialization and de-initialization of the software drivers. The EcuM is also responsible for the wakeup operation of configured wakeup capable channels when the driver is in sleep mode therefore the user must configure the wake-up information in EcuM configuration, which will be assigned to every wake-up capable GPT channel. The EcuM module provided in the MCAL package is a stub code and needs to be replaced with a complete EcuM module during the integration phase.

- **Memory mapping**

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

  The `Gpt_MemMap.h` file is provided in the MCAL package as a stub code. The integrator must place appropriate compiler pragmas within the memory-section macros. The pragmas ensure that the elements are re-located to the correct memory region. A sample implementation listing the memory-section macros is shown as follows.

```
/* Sample implementation of Gpt_MemMap.h */
/* Core specific data [ DSRAM[X] ] */
#if defined GPT_START_SEC_VAR_CLEARED_ASIL_B_CORE[X]_32
 /* User pragmas to allocate to DSRAM[X] */
 #undef GPT_START_SEC_VAR_CLEARED_ASIL_B_CORE[X]_32
 #undef MEMMAP_ERROR
#elif defined GPT_STOP_SEC_VAR_CLEARED_ASIL_B_CORE[X]_32
 /* User pragmas to end to allocation to DSRAM[X] */
 #undef GPT_STOP_SEC_VAR_CLEARED_ASIL_B_CORE[X]_32
 #undef MEMMAP_ERROR

/* Global data [NON-CACHED LMU] */
#elif defined GPT_START_SEC_VAR_INIT_ASIL_B_GLOBAL_32
 /* User pragmas to allocate to Non-cached LMU */
 #undef GPT_START_SEC_VAR_INIT_ASIL_B_GLOBAL_32
 #undef MEMMAP_ERROR
#elif defined GPT_STOP_SEC_VAR_INIT_ASIL_B_GLOBAL_32
 /* User pragmas to end allocation to Non-cached LMU */
 #undef GPT_STOP_SEC_VAR_INIT_ASIL_B_GLOBAL_32
 #undef MEMMAP_ERROR

/* Core specific constant data [ PFLASH[X] ] */
#elif defined GPT_START_SEC_CONST_ASIL_B_CORE[X]_8
 /* User pragmas to allocate to PFLASH[X] */
 #undef GPT_START_SEC_CONST_ASIL_B_CORE[X]_8
 #undef MEMMAP_ERROR
#elif defined GPT_STOP_SEC_CONST_ASIL_B_CORE0_8
 /* User pragmas to end allocation to PFLASH[X] */
 #undef GPT_STOP_SEC_CONST_ASIL_B_CORE[X]_8
 #undef MEMMAP_ERROR

/* Global configuration data [ PFLASH[X] ] */
#elif defined GPT_START_SEC_CONFIG_DATA_ASIL_B_GLOBAL_UNSPECIFIED
 /* User pragmas to allocate to PFLASH[X] */
 #undef GPT_START_SEC_CONFIG_DATA_ASIL_B_GLOBAL_UNSPECIFIED
 #undef MEMMAP_ERROR
#elif defined GPT_STOP_SEC_CONFIG_DATA_ASIL_B_GLOBAL_UNSPECIFIED
 /* User pragmas to end allocation to PFLASH[X] */
 #undef GPT_STOP_SEC_CONFIG_DATA_ASIL_B_GLOBAL_UNSPECIFIED
 #undef MEMMAP_ERROR

/* Code [ PFLASH[X] ] */
#elif defined GPT_START_SEC_CODE_ASIL_B_GLOBAL
 /* User pragmas to allocate to PFLASH[X] */
 #undef GPT_START_SEC_CODE_ASIL_B_GLOBAL
 #undef MEMMAP_ERROR
#elif defined GPT_STOP_SEC_CODE_ASIL_B_GLOBAL
 /* User pragmas to end allocation to PFLASH[X] */
 #undef GPT_STOP_SEC_CODE_ASIL_B_GLOBAL
 #undef MEMMAP_ERROR
#endif
```

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

- **DET**

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

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

- **Mcal_Wrapper**

  This Driver performs reporting of the Production and Runtime errors. The Handling of the reported errors shall be done by the user. The `Mcal_Wrapper_Det_ReportRuntimeError()` API, `Mcal_Wrapper_Dem_SetEventStatus()` API and `Mcal_Wrapper_Dem_ReportErrorStatus()` API are provided in the `Mcal_Wrapper.c` and `Mcal_Wrapper.h` files as a stub code, and can be updated by the integrator to handle the reported errors. The files `Mcal_Wrapper.c` and `Mcal_Wrapper.h` are user modifiable but the function prototype is not user modifiable and by default the Mcal Wrapper function shall call AUTOSAR DEM and DET Modules.

  The user of the GPT driver shall process all the Runtime errors reported to the `Mcal_Wrapper` module. The interface used for reporting Runtime error is `Mcal_Wrapper_Det_ReportRuntimeError()` API. Production error is not applicable for GPT driver. The `Mcal_Wrapper.c` and `Mcal_Wrapper.h` files are provided in the MCAL package as a stub code and can be replaced with a user specific Runtime error handling module/s during the integration phase.

- **SchM**

  The SchM module is a part of the RTE that manages the Basic Software Scheduler. The GPT driver uses the exclusive areas defined in the `SchM_Gpt.h` file to protect the SFRs and variables from concurrent accesses from different threads. SchM is used for predef timers only. The SchMs identified for the GPT driver are:

- Get100UsPredefTimerValue
- Get1UsPredefTimerValue
- GtmStartTimer
- Gpt12StartTimer

  The files `SchM_Gpt.h` and `SchM_Gpt.c` are provided in the MCAL package as an example code and needs to updated by the integrator. The user must implement the SchM functions defined by the GPT driver as **suspend / resume** of interrupts for the CPU on which the API is invoked. A sample implementation of the SchM functions is shown as follows:

```
/**** Sample implementation of SchM_Adc.c ****/
#include "Os.h"

void SchM_Enter_Gpt_Get1UsPredefTimerValue(void)
{
 /* Start of Critical Section */
 SuspendAllInterrupts();/* Suspend CPU core interrupt */
}

void SchM_Exit_Gpt_Get1UsPredefTimerValue(void)
{
 /* End of Critical Section */
 ResumeAllInterrupts();/* Resume CPU core interrupt */
}

void SchM_Enter_Gpt_Get100UsPredefTimerValue(void)
{
 /* Start of Critical Section */
 SuspendAllInterrupts();/* Suspend CPU core interrupt */
}

void SchM_Exit_Gpt_Get100UsPredefTimerValue(void)
{
 /* End of Critical Section */
 ResumeAllInterrupts();/* Resume CPU core interrupt */
}

void SchM_Enter_Gpt_GtmStartTimer(void)
{
 /* Start of Critical Section */
 SuspendAllInterrupts();/* Suspend CPU core interrupt */
}

void SchM_Exit_Gpt_GtmStartTimer(void)
{
 /* End of Critical Section */
 ResumeAllInterrupts();/* Resume CPU core interrupt */
}

void SchM_Enter_Gpt_Gpt12StartTimer(void)
{
 /* Start of Critical Section */
 SuspendAllInterrupts();/* Suspend CPU core interrupt */
}

void SchM_Exit_Gpt_Gpt12StartTimer(void)
{
 /* End of Critical Section */
 ResumeAllInterrupts();/* Resume CPU core interrupt */
}
```

- **Safety error**

  The GPT driver will report all the detected safety errors through the API `Mcal_ReportSafetyError()`.

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

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

- **Notifications and callbacks**

  The GPT driver itself does not implement any notifications. However, GPT driver invokes the notification function configured for a GPT channel on timer count match. These notification functions can be configured by the user in Tresos for each GPT Channel(capable of issuing notifications) separately.

  GPT does not provide any call-backs.

- **Operating system(OS)**

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

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

1.1.4.2 Multicore and resource manager

The GPT driver supports execution of its APIs simultaneously from all CPU cores. The user should allocate resources of TOM/ATOM to CPU cores at pre-compile time using the Resource manager module. The following are the key points to be considered with respect to multicore in the driver:

- GPT channels can be allocated to CPU cores at pre-compile time.
- Predefined timers are available only after master core initialization.
- GPT channel configured as Predef timer can be allocated to any core but will be controlled (started/stopped) only by the logical master core. All slave core can access(/read) the Predef timer count.
- It must be ensured that GPT Channel passed as parameter while invoking an API belong to the same core on which the API is invoked.
- DETs will be raised in case APIs are invoked with mismatch of core and GPT Channel.
- GPT channels using GTM-TOM[i]_CH[X] and TOM[i]_CH[x+1] must be allocated to the same core as these two channels share the same interrupt line.
- GPT channels using GTM-ATOM[i]_CH[X] and ATOM[i]_CH[x+1] must be allocated to the same core as these two channels share the same interrupt line.
- Interrupts raised by individual TOM/ATOM/GPT12 channels must be serviced by the CPU core to which the TOM/ATOM/GPT12 channels have been allocated to.
- Locating constants, variables and configuration data to correct memory space should be done by the user. Memory sections are marked GLOBAL (common to all cores) and CORE[x](specific to a CPU core). The following should be considered by the user to ensure better performance of the driver:

  Code section:

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

  Data section:

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

Configuration data and constants:

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

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

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

1.1.4.3 MCU support

The GPT driver is dependent on the MCU driver for clock configuration and timer IP-related services. The initialization of GPT driver must be started only after completing the MCU initialization. The following must be considered while configuring the MCU driver in the EBtresos:

- GPT driver uses the services of the MCU to configure the GTM-related trigger events during runtime. The GTM-TOM/GTM-ATOM channels used by GPT driver must be reserved in the MCU configuration for exclusive use by the GPT driver.

1.1.4.4 Port support

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

1.1.4.5 DMA support

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

1.1.4.6 Interrupt connections

If wakeups/notifications are required, user should enable interrupts in the interrupt configuration register.

The interrupt configuration registers of different hardware used by GPT channels are given below:

Table 4 SRC registers

Hardware used	SRC register
GTM-TOM	SRC_GTMTOMwx (w= TOM module, x= 2n and 2n+1 TOM channel)
GTM-ATOM	SRC_GTMATOMwx (w= ATOM module, x= 2n and 2n+1 ATOM channel)
GPT12	SRC_GPT12_GPT120_Tx (x=2 to 6)

All the ISR to GTM-TOM, GTM-ATOM and GPT12 must be routed to the `Mcu_17_Gtm_TomChannelIsr`, `Mcu_17_Gtm_AtomChannelIsr` and `Mcu_17_Gpt12_ChannelIsr` APIs respectively which in turn invokes the `Gpt_Isr` API.

The user must ensure that the MCU interrupt handler is invoked from the ISR for which the sample code is shown as follows:

```
ISR(GTMTOM0SR0_ISR)
{
 /* Enable Global Interrupts */
 ENABLE();

 /* Parameter is TOM Module 0 and Channel 0 */
 Mcu_17_Gtm_TomChannelIsr(0, 0);
}
```

```
ISR(GTMATOM0SR0_ISR)
{
 /* Enable Global Interrupts */
 ENABLE();

 /* Parameter is ATOM Module 0 and Channel 0 */
 Mcu_17_Gtm_AtomChannelIsr(0, 0);
}
```

```
ISR(GPT12_T3_ISR)
{
 /* Enable Global Interrupts */
 ENABLE();

 /* Parameter is GPT12 timer1 */
 Mcu_17_Gpt12_ChannelIsr(1);
}
```

1.1.4.7 Example usage

Pre-condition

For any functionality of GPT, MCU should be initialized, all the clock configurations should be completed and PLL clock should be distributed to all the peripherals.

Example initialization sequence is shown as follows.

```
/*Initialize Mcu */
Mcu_Init(&Mcu_Config);

/* Initialize the PLLs and other MCU specific clock options */
(void)Mcu_InitClock( 0 );

while(Mcu_GetPllStatus() != MCU_PLL_LOCKED)
{
};

/* Select the PLL clock as source for MCU clock tree distribution */
(void)Mcu_DistributePllClock();
```

Initialization

User must include Gpt_PBcfg.h to access the GPT configuration structure needed for initialization.

```
/* Include Gpt.h to access interfaces of GPT driver */
#include "Gpt.h"

/* Module Initialization */
void Gpt_Sample_Init(void)
{
 /* Initialize Gpt*/
 Gpt_Init(&Gpt_Config);
}
```

Start and stop timer

The user must call `Gpt_StartTimer` API with the logical channel ID and the target time for any normal (non-predefined timer) channel. The timer can be stopped by invoking the `Gpt_StopTimer` API.

```
/* GPT module Initialization is necessary to start timer */
Gpt_Init(&Gpt_Config);

/* Start the timer */
Gpt_StartTimer(<LogicalChannelSymbolicName>, <targetCount>);

/* Count the number of ticks currently */
Gpt_ValueType timeElapsed = Gpt_GetTimeElapsed(<LogicalChannelSymbolicName>);

/* Count the number of ticks remaining to reach the target count */
Gpt_ValueType timeRemaining = Gpt_GetTimeRemaining(<LogicalChannelSymbolicName>);

/* Stop the timer */
Gpt_StopTimer(<LogicalChannelSymbolicName>);
```

General API

`Gpt_EnableNotification` and `Gpt_DisableNotification` should be invoked after calling `Gpt_Init` and only on a channel that has a valid notification function.

`Gpt_EnableWakeup` and `Gpt_DisableWakeup` should be invoked after the call to `Gpt_Init` and only on a wakeup capable channel.

`Gpt_SetMode` should be invoked after the call to `Gpt_Init` to change the state of GPT driver to SLEEP or NORMAL.

`Gpt_DeInit`, should be invoked after the call to `Gpt_Init` to reset the initialization state of the Gpt driver. After the call to `Gpt_DeInit`, `Gpt_Init` should be invoked again to start any functionality of the GPT driver.

```
/* GPT module Initialization is necessary to start timer */
Gpt_Init(&Gpt_Config);

/* Enable wakeup for a wakeup capable channel */
Gpt_EnableWakeup(<WakeupCapableLogicalChannelSymbolicName>);

/* Change mode to sleep */
Gpt_SetMode(GPT_MODE_SLEEP);

/* provide signal on the wakeup capable channel */
/* EcuMCheckWakeup will be invoked from ISR to indicate a wakeup signal */

/* Change mode to normal */
Gpt_SetMode(GPT_MODE_NORMAL);

/* Disable wakeup for a wakeup capable channel */
Gpt_DisableWakeup(<WakeupCapableLogicalChannelSymbolicName>>);

/* Change mode to sleep */
Gpt_SetMode(GPT_MODE_SLEEP);

/* provide signal on the wakeup capable channel */
/* EcuMCheckWakeup will not be invoked */
```

1.1.5 Key architectural considerations

1.1.5.1 Hardware dependency

The hardware provides GTM (TOM and ATOM slice) and GPT12 timers, which are used as the underlying hardware for GPT. As all TOM, ATOM and GPT12 channels are independent, each channel can be mapped to one of the cores. GTM (TOM and ATOM Slice) is not available for the TC35xx devices.

1.1.5.2 User mode support

The Gpt module supports User-1 and Supervisor modes. Additional configuration parameters are not provided to switch the modes.[cover parentID GPT={1A65EADD-AFD0-4845-B2D2-8257E086DD67}]

1.2 Assumptions of Use (AoU)

The AoU for the GPT driver are as follows:

- **Configuration check**

User shall ensure that the generated configuration is correct against the GUI configurations provided by the user.

[cover parentID GPT={6223236A-4BFD-40eb-812A-2AE1B0A896DD}]

- **Disabling the interrupts in critical section**

User shall protect the read sequence of Predef timer with resolution more than 16 bits by disabling the interrupts in the critical section Get1UsPredefTimerValue and Get100UsPredefTimerValue.

[cover parentID GPT={F38180FC-D9DF-4e32-9DD1-37B3464914AB}]

- **Execution sequence for initialization check**

User shall invoke the Gpt_InitCheck API only after invoking the Gpt_Init API, and before invoking any runtime API other than the Gpt_GetVersionInfo API.

[cover parentID GPT={B75D5D36-FD97-4130-9A97-7A58DADDAB11}]

- **Freedom from interference for MCAL data**

User shall provide protection for the MCAL memory and SFRs from the QM software.

[cover parentID GPT={53C19FC7-7847-481a-B828-3CC536391721}]

- **Handling of underflow situation for one-shot mode with GPT12**

The Gpt_GetTimeElapsed and Gpt_GetTimeRemaining APIs for channels using GPT12 auxillary timers should not be invoked from interrupts, which have higher priority than the particular GPT12 channel interrupt.

[cover parentID GPT={DEE11E9A-DEE3-4e23-93E5-8591E2F7235D}]

1.3 Reference information

1.3.1 Configuration interfaces

Supported configuration variant: Post-Build

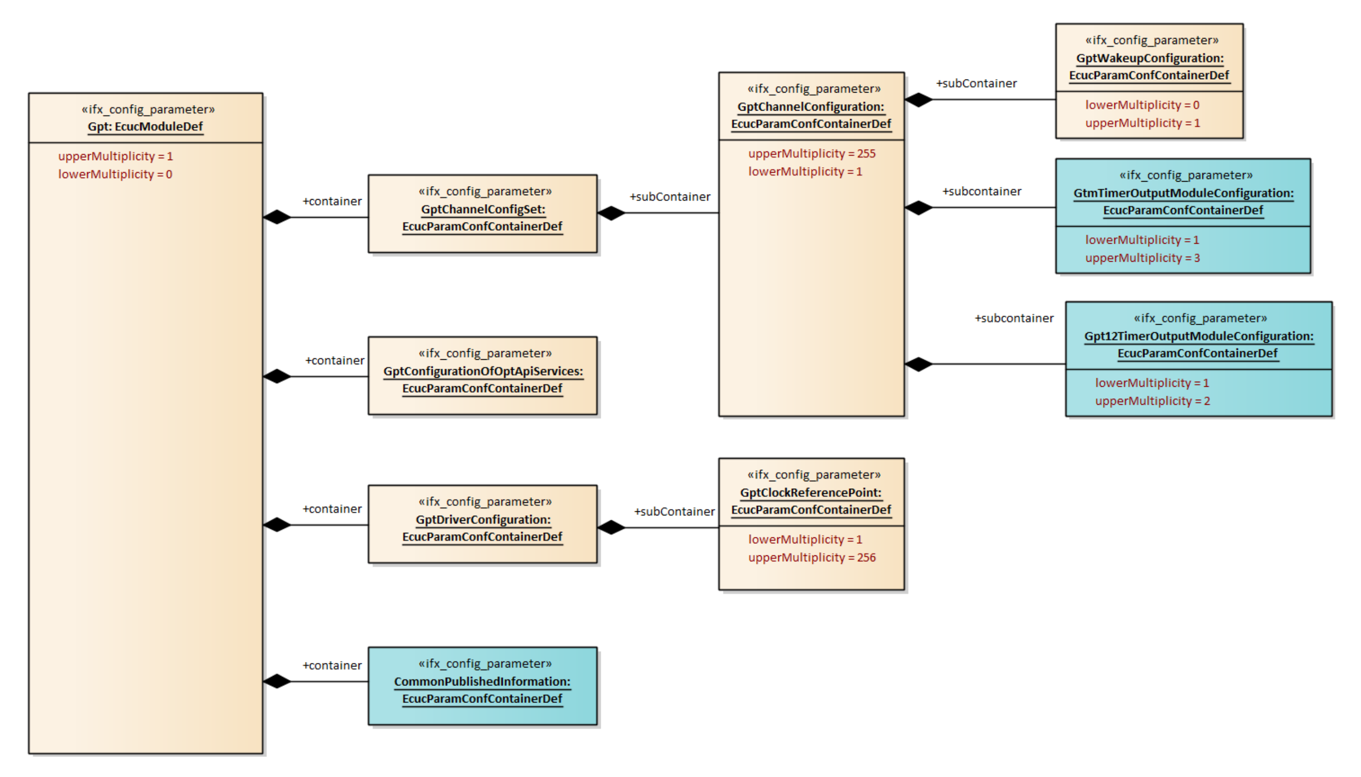

Figure 4 Container hierarchy along with their configuration parameters

1.3.1.1 Container: CommonPublishedInformation

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.1.1 ArMajorVersion

Table 5 Specification for ArMajorVersion

Name	`ArMajorVersion`		
Description	Major version number of AUTOSAR specification on which the appropriate implementation is based on.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	4		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-

(table continues...)

Table 5 (continued) Specification for ArMajorVersion

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

1.3.1.1.2 ArMinorVersion

Table 6 Specification for ArMinorVersion

Name	`ArMinorVersion`		
Description	Minor version number of AUTOSAR specification on which the appropriate implementation is based on.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	As per Autosar version		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.3 ArPatchVersion

Table 7 Specification for ArPatchVersion

Name	`ArPatchVersion`		
Description	Patch level version number of AUTOSAR specification on which the appropriate implementation is based on.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	As per Autosar version		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-

(table continues...)

Table 7 (continued) Specification for ArPatchVersion

Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.4 ModuleId

Table 8 Specification for ModuleId

Name	`ModuleId`		
Description	Module ID of this module from Module List		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 65535		
Default value	100		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.5 Release

Table 9 Specification for Release

Name	`Release`		
Description	Aurix derivative used for the implementation *Note: Default value will be selected based on the target device to ensure the relevance.*		
Multiplicity	1..1	**Type**	EcucStringParamDef
Range	String		
Default value	_TRICORE_TC3xx (xx will be replaced as per target device. Ex: For TC399 xx will be 99)		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.6 SwMajorVersion

Table 10 Specification for SwMajorVersion

Name	`SwMajorVersion`		
Description	Major version number of the vendor specific implementation of the module. The numbering is vendor specific.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	As per driver		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.7 SwMinorVersion

Table 11 Specification for SwMinorVersion

Name	`SwMinorVersion`		
Description	Minor version number of AUTOSAR specification on which the appropriate implementation is based on.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	As per driver		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.8 SwPatchVersion

Table 12 Specification for SwPatchVersion

Name	`SwPatchVersion`

(table continues...)

Table 12 (continued) Specification for SwPatchVersion

Description	Patch level version number of the vendor specific implementation of the module. The numbering is vendor specific.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	As per driver		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.9 VendorId

Table 13 Specification for VendorId

Name	`VendorId`		
Description	This parameter provides the Vendor Id		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 65535		
Default value	17		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.2 Container: Gpt12TimerOutputModuleConfiguration

This container contains the configuration elements for configuring GPT channel of GPT12 timer hardware and the input clock divider selection for the respective timer. For One-shot mode, the multiplicity is 1..1. For Continuous mode the multiplicity is 1..2

Post-Build Variant Multiplicity: FALSE

Multiplicity Configuration Class: Pre-Compile

1.3.1.2.1 Gpt12ChannelClockDivider

Table 14 Specification for Gpt12ChannelClockDivider

Name	`Gpt12ChannelClockDivider`		
Description	This parameter decides the clock divider value for GPT12 timer. The value of the parameter cannot be changed across variants since the dependent parameter, Gpt12TimerUsed refers to pre-compile parameter. Default value is zero because the default value of the corresponding register bits are zero.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 7		
Default value	0		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.2.2 Gpt12TimerUsed

Table 15 Specification for Gpt12TimerUsed

Name	`Gpt12TimerUsed`		
Description	This parameter is defined by IFX. It is the reference to GPT12 timer used by GPT Driver. *Note: Referred timer channel in MCU should have timer usage as GPT_TIMER_USED_BY_GPT_DRIVER.* The value of the parameter cannot be changed across variants since the referred value McuGpt12ModuleAllocationConf is a pre-compile parameter. Since the name of the dependent container is user configurable, the default value is kept as NULL		
Multiplicity	1..1	**Type**	EcucReferenceDef
Range	Reference to Node:		
Default value	NULL		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.3 Container: GtmTimerOutputModuleConfiguration

This container contains the configuration elements for configuring TOM/ATOM channel of GTM timer hardware and the input clock selection for the respective timer. For Continuous mode and One-shot mode, the multiplicity is 1..1. For Predef timers, the multiplicity is 2..3. This container is not configurable for devices with no GTM.

Post-Build Variant Multiplicity: FALSE

Multiplicity Configuration Class: Pre-Compile

1.3.1.3.1 GtmTimerClockSelect

Table 16 Specification for GtmTimerClockSelect

Name	`GtmTimerClockSelect`		
Description	This parameter decides the clock source for TOM/ATOM timer. Default value: GTM_FIXED_CLOCK_0 selected because TOM channels will run with fixed clock. The value of the parameter cannot be changed across variants since the dependent parameter, GtmTimerUsed refers to pre-compile parameter. This parameter is not configurable for devices with no GTM IP.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	GTM_CONFIGURABLE_CLOCK_X: Configurable Clock X is selected for ATOM module, where X = 0 to 7 GTM_FIXED_CLOCK_0: Fixed Clock 0 is selected for TOM module. GTM_FIXED_CLOCK_1: Fixed Clock 1 is selected for TOM module. GTM_FIXED_CLOCK_2: Fixed Clock 2 is selected for TOM module. GTM_FIXED_CLOCK_3: Fixed Clock 3 is selected for TOM module. GTM_FIXED_CLOCK_4: Fixed Clock 4 is selected for TOM module.		
Default value	GTM_FIXED_CLOCK_0		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	GtmTimerUsed, GptTimerChannelUsage		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.3.2 GtmTimerUsed

Table 17 Specification for GtmTimerUsed

Name	`GtmTimerUsed`

(table continues...)

Table 17 (continued) Specification for GtmTimerUsed

Description	This parameter is defined by IFX. It is the reference to GTM timer channel (TOM/ATOM) used by GPT Driver. *Note: Referred timer channel in MCU should have TomChannelUsage/ AtomChannelUsage as GTM_TOM_CHANNEL_USED_BY_GPT / GTM_ATOM_CHANNEL_USED_BY_GPT.* Since the name of the dependent container is user configurable, the default value is kept as NULL. The value of the parameter cannot be changed across variants since the referred parameters, McuGtmTomChannelAllocationConf and McuGtmAtomChannelAllocationConf are pre-compile parameters. This parameter is not configurable for devices with no GTM IP.		
Multiplicity	1..1	**Type**	EcucReferenceDef
Range	Reference to Node: McuGtmAtomChannelAllocationConf, McuGtmTomChannelAllocationConf		
Default value	NULL		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4 Container: Gpt

Configuration of the GPT (General Purpose Timer) driver

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.5 Container: GptChannelConfigSet

This container is the base of a Configuration Set which contains the configured GPT channels. This way, different configuration sets can be defined for the post-build process.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.6 Container: GptChannelConfiguration

Configuration of an individual GPT channel

Post-Build Variant Multiplicity: FALSE

Multiplicity Configuration Class: Pre-Compile

1.3.1.6.1 GptAssignedHwUnit

Table 18 Specification for GptAssignedHwUnit

Name	`GptAssignedHwUnit`		
Description	Specifies the hardware used for the GPT channel(GPT12 or GTM). The value of the parameter cannot be changed across variants since the dependent parameters Gpt12TimerUsed and GtmTimerUsed refer to pre-compile parameters. By default this will contain GPT12 since it is present across all TC3xx controller variants.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	GPT12: The GPT channel will be realized using GPT12 hardware. GTM: The GPT channel will be realized using GTM(ATOM/TOM) hardware.		
Default value	GPT12		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.6.2 GptChannelClkSrcRef

Table 19 Specification for GptChannelClkSrcRef

Name	`GptChannelClkSrcRef`		
Description	Reference to the GptClockReferencePoint from which the clock for the channel is derived.		
Multiplicity	1..1	**Type**	EcucReferenceDef
Range	Reference to Node: GptClockReferencePoint		
Default value	NA		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.6.3 GptChannelEcucPartitionRef

Table 20 Specification for GptChannelEcucPartitionRef

Name	`GptChannelEcucPartitionRef`		
Description	Maps an Gpt channel to zero or multiple ECUC partitions to limit the access to this channel. The ECUC partitions referenced are a subset of the ECUC partitions where the Gpt driver is mapped to. *Note: Parameter support is added only for AUTOSAR schema compliance, this parameter is not used in code generation logic, hence this parameter is made editable false.*		
Multiplicity	0..*	**Type**	EcucReferenceDef
Range	Reference to Node: EcucPartition		
Default value	NULL		
Post-build variant value	TRUE	**Post-build variant multiplicity**	TRUE
Value configuration class	Pre-Compile	**Multiplicity configuration class**	Pre-Compile
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar version 4.4.0.		

1.3.1.6.4 GptChannelId

Table 21 Specification for GptChannelId

Name	`GptChannelId`		
Description	Channel Id of the GPT channel. This value will be assigned to the symbolic name derived from the GptChannelConfiguration container short name. The value should be unique and continuous starting from 0. *Note: Default value is set to minimum value.*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - (Total number of channels - 1)		
Default value	0		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.6.5 GptChannelMode

Table 22 Specification for GptChannelMode

Name	`GptChannelMode`		
Description	Specifies the behavior of the timer channel after the target time is reached. By default this will refer to continues mode to ensure that the timer is running all the times.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	GPT_CH_MODE_CONTINUOUS: After reaching the target time, the timer continues running with the value "zero" again. GPT_CH_MODE_ONESHOT: After reaching the target time, the timer stops automatically (timer expired).		
Default value	GPT_CH_MODE_CONTINUOUS		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	GptTimerChannelUsage		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.6.6 GptChannelTickFrequency

Table 23 Specification for GptChannelTickFrequency

Name	`GptChannelTickFrequency`		
Description	Specifies the tick frequency of the timer channel in Hz. This is not used for implementation and disabled for the configuration GUI. The GPT channel tick value and tick frequency are handled internally with the internal pre-scalars and dividers coming from CMU. Therefore these configurations should be done at the MCU module and they will not be available at GPT channel configurations.		
Multiplicity	1..1	**Type**	EcucFloatParamDef
Range	0 - 65535		
Default value	1		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL

(table continues...)

Table 23 (continued) Specification for GptChannelTickFrequency

Dependency	-
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.1.6.7 GptChannelTickValueMax

Table 24 Specification for GptChannelTickValueMax

Name	`GptChannelTickValueMax`		
Description	Maximum value in ticks, the timer channel is able to count. With the next tick, the timer rolls over to zero. This is not used for implementation and disabled for the configuration GUI. The GPT channel tick value and tick frequency are handled internally with the internal pre-scalars and dividers coming from CMU. Therefore these configurations should be done at the MCU module and they will not be available at GPT channel configurations.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 18446744073709551615		
Default value	65535		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.6.8 GptEnableWakeup

Table 25 Specification for GptEnableWakeup

Name	`GptEnableWakeup`		
Description	Enables wakeup capability of MCU for a channel. *Note: Not applicable if the channel is configured for Predef Timer.* The optional features are disabled by default to minimize the executable code size.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-

(table continues...)

Table 25 (continued) Specification for GptEnableWakeup

Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	GptPredefTimer1usEnablingGrade, GptPredefTimer100us32bitEnable, GptReportWakeupSource		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.6.9 GptNotification

Table 26 Specification for GptNotification

Name	`GptNotification`		
Description	The GptNotification is used by the GPT driver to invoke the user-defined function for notification purposes(for non-wakeup notification). The parameter can be a name or the address(numeric value) of the notification function. *Note: The GPT driver does not validate the configured function name or address for correctness and the responsibility falls on the user.* *Note: Notification is not applicable if the channel is configured for Predef Timer.*		
Multiplicity	0..1	**Type**	EcucFunctionNameDef
Range	String		
Default value	IoHwAb_GptNotification[#channel]		
Post-build variant value	TRUE	**Post-build variant multiplicity**	TRUE
Value configuration class	Post-Build	**Multiplicity configuration class**	Post-Build
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	GptTimerChannelUsage, GptEnableDisableNotificationApi, GptPredefTimer1usEnablingGrade, GptPredefTimer100us32bitEnable		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.6.10 GptTimerChannelUsage

Table 27 Specification for GptTimerChannelUsage

Name	`GptTimerChannelUsage`

(table continues...)

Table 27 **(continued) Specification for GptTimerChannelUsage**

Description	Specifies the usage of the timer channel as a predefined timer or normal GPT timer. By default it will be normal channel because a normal channel can be used for all possible configurations (free running/ OSM etc). For devices where GTM IP is not available, only GPT_TIMER_CHANNEL_NORMAL option is available and the Predef timer support is disabled.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	GPT_PREDEF_TIMERCH_100US_32BIT: Channel is selected as Gpt Predef Timer with tick duration 100 us and range 32bit. GPT_PREDEF_TIMERCH_1US_16BIT: Channel is selected as Gpt Predef Timer with tick duration 1 us and range 16 bit. GPT_PREDEF_TIMERCH_1US_16_24BIT: Channel is selected as Gpt Predef Timer with tick duration 1 us and range 24bit. GPT_PREDEF_TIMERCH_1US_16_24_32BIT: Channel selected as Gpt Predef Timer with tick duration 1 us and range 32bit. GPT_TIMER_CHANNEL_NORMAL: Channel is selected as normal Gpt timer		
Default value	GPT_TIMER_CHANNEL_NORMAL		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.7 Container: GptClockReferencePoint

This container contains a parameter, which represents a reference to a container of the type McuClockReferencePoint (defined in module MCU). A container is needed to support multiple clock references (hardware dependent). Since the name of the dependent container is user configurable, the default value is kept as NULL

Post-Build Variant Multiplicity: FALSE

Multiplicity Configuration Class: Post-Build

1.3.1.7.1 GptClockReference

Table 28 **Specification for GptClockReference**

Name	`GptClockReference`
Description	Reference to a container of the type McuClockReferencePoint, to select an input clock. This parameter is kept non-editable since the MCU clock reference is provided by another parameter. This parameter is kept to maintain compatibility with Autosar.

(table continues...)

Table 28 (continued) Specification for GptClockReference

Multiplicity	1..1	**Type**	EcucReferenceDef
Range	Reference to Node: McuClockReferencePoint		
Default value	NA		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.7.2 GptClockReferenceConfig

Table 29 Specification for GptClockReferenceConfig

Name	`GptClockReferenceConfig`		
Description	Reference to a container of the type McuClockReferencePointConfig, to select an input clock. The configuration editor for the GPT driver can support the integrator by only allowing a selection of those clock reference points that can be connected physically to the GPT hardware peripheral. The desired frequency (desired by the GPT driver) has to be the same as the selected and provided frequency of the MCU configuration. This has to be checked automatically. Since the name of the dependent container is user configurable, the default value is kept as NULL		
Multiplicity	1..1	**Type**	EcucReferenceDef
Range	Reference to Node: McuClockReferencePointConfig		
Default value	NA		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.8 Container: GptConfigurationOfOptApiServices

This container contains all configuration switches for configuring optional API services of the GPT driver.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.8.1 GptDeinitApi

Table 30 Specification for GptDeinitApi

Name	`GptDeinitApi`		
Description	Adds/removes the service Gpt_DeInit() from the code. *Note: The optional APIs are disabled by default to minimize the executable code size.*		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.8.2 GptEnableDisableNotificationApi

Table 31 Specification for GptEnableDisableNotificationApi

Name	`GptEnableDisableNotificationApi`		
Description	Adds/removes the services Gpt_EnableNotification() and Gpt_DisableNotification from the code. *Note: The optional APIs are disabled by default to minimize the executable code size.*		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL

(table continues...)

Table 31 (continued) Specification for GptEnableDisableNotificationApi

Dependency	-
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.1.8.3 GptInitCheckApi

Table 32 Specification for GptInitCheckApi

Name	`GptInitCheckApi`		
Description	Adds/removes the service Gpt_InitCheck() from the code. *Note: The detection of safety related errors is enable by default to ensure that safety issues are addressed during the product lifecycle.*		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.8.4 GptTimeElapsedApi

Table 33 Specification for GptTimeElapsedApi

Name	`GptTimeElapsedApi`		
Description	Adds/removes the service Gpt_GetTimeElapsed() from the code. *Note: The optional APIs are disabled by default to minimize the executable code size.*		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-

(table continues...)

Table 33 (continued) Specification for GptTimeElapsedApi

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

1.3.1.8.5 GptTimeRemainingApi

Table 34 Specification for GptTimeRemainingApi

Name	`GptTimeRemainingApi`		
Description	Adds/removes the service Gpt_GetTimeRemaining() from the code. *Note: The optional APIs are disabled by default to minimize the executable code size.*		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.8.6 GptVersionInfoApi

Table 35 Specification for GptVersionInfoApi

Name	`GptVersionInfoApi`		
Description	Adds/removes the service Gpt_GetVersionInfo() from the code. *Note: The optional APIs are disabled by default to minimize the executable code size.*		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		

(table continues...)

Table 35 (continued) Specification for GptVersionInfoApi

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

1.3.1.8.7 GptWakeupFunctionalityApi

Table 36 Specification for GptWakeupFunctionalityApi

Name	`GptWakeupFunctionalityApi`		
Description	Adds/removes the services Gpt_SetMode(), Gpt_EnableWakeup(), Gpt_DisableWakeup() and Gpt_CheckWakeup() from the code. *Note: The optional APIs are disabled by default to minimize the executable code size.*		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.9 Container: GptDriverConfiguration

This container contains the module-wide configuration (parameters) of the GPT driver

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.9.1 GptDevErrorDetect

Table 37 Specification for GptDevErrorDetect

Name	`GptDevErrorDetect`

(table continues...)

Table 37 (continued) Specification for GptDevErrorDetect

Description	Switches the Default Error Tracer (Det) detection and notification ON or OFF.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.9.2 GptEcucPartitionRef

Table 38 Specification for GptEcucPartitionRef

Name	`GptEcucPartitionRef`		
Description	Maps the GPT driver to zero or multiple ECUC partitions to make the driver API available in the according partition. *Note: Parameter support is added only for AUTOSAR schema compliance, this parameter is not used in code generation logic, hence this parameter is made editable false.*		
Multiplicity	0..*	**Type**	EcucReferenceDef
Range	Reference to Node:		
Default value	NULL		
Post-build variant value	TRUE	**Post-build variant multiplicity**	TRUE
Value configuration class	Pre-Compile	**Multiplicity configuration class**	Pre-Compile
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar version 4.4.0.		

1.3.1.9.3 GptKernelEcucPartitionRef

Table 39 Specification for GptKernelEcucPartitionRef

Name	`GptKernelEcucPartitionRef`		
Description	Maps the GPT kernel to zero or one ECUC partitions to assign the driver kernel to a certain core. The ECUC partition referenced is a subset of the ECUC partitions where the GPT driver is mapped to. *Note: Parameter support is added only for AUTOSAR schema compliance, this parameter is not used in code generation logic, hence this parameter is made editable false.*		
Multiplicity	0..1	**Type**	EcucReferenceDef
Range	Reference to Node:		
Default value	NULL		
Post-build variant value	TRUE	**Post-build variant multiplicity**	TRUE
Value configuration class	Pre-Compile	**Multiplicity configuration class**	Pre-Compile
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar version 4.4.0.		

1.3.1.9.4 GptMultiCoreErrorDetect

Table 40 Specification for GptMultiCoreErrorDetect

Name	`GptMultiCoreErrorDetect`		
Description	This parameter enables or disables the Multi core related default error tracer (Det) detection and reporting. It is applicable only when DETs are enabled. GptMultiCoreErrorDetect shall be set to FALSE if CPU selected is single core.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	GptDevErrorDetect		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.9.5 GptPredefTimer100us32bitEnable

Table 41 Specification for GptPredefTimer100us32bitEnable

Name	`GptPredefTimer100us32bitEnable`		
Description	Enables/disables the GPT Predef Timer 100us 32bit. The optional APIs are disabled by default to minimize the executable code size. This parameter is not configurable for devices with no GTM IP.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	ECU
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.9.6 GptPredefTimer1usEnablingGrade

Table 42 Specification for GptPredefTimer1usEnablingGrade

Name	`GptPredefTimer1usEnablingGrade`		
Description	Specifies the grade of enabling the GPT Predef Timers with 1 us tick duration. The optional APIs are disabled by default to minimize the executable code size. This parameter is not configurable for devices with no GTM IP.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	GPT_PREDEF_TIMER_1US_16BIT_ENABLED: 16bit timer enabled GPT_PREDEF_TIMER_1US_16_24BIT_ENABLED: 16 and 24bit timers enabled GPT_PREDEF_TIMER_1US_16_24_32BIT_ENABLED: 16, 24 and 32bit timers enabled GPT_PREDEF_TIMER_1US_DISABLED: Specifies that GPT Predef Timers with 1 us tick duration is disabled.		
Default value	GPT_PREDEF_TIMER_1US_DISABLED		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-

(table continues...)

Table 42 (continued) Specification for GptPredefTimer1usEnablingGrade

Origin	AUTOSAR_ECUC	**Scope**	ECU
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.9.7 GptReadAcrossCores

Table 43 Specification for GptReadAcrossCores

Name	`GptReadAcrossCores`		
Description	This parameter enables/disables reading the channel from the cores apart from the core for which the channel has been allocated. This applies to the APIs Gpt_GetTimeElapsed and Gpt_GetTimeRemaining only. If GptReadfromAllCores is TRUE, it allows the GPT normal channels to be read from all cores. If GptReadfromAllCores is FALSE, it allows the GPT normal channels to be read only from the core for which the GPT channels are configured.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.9.8 GptReportWakeupSource

Table 44 Specification for GptReportWakeupSource

Name	`GptReportWakeupSource`		
Description	Enables/Disables wakeup source reporting. The optional features are disabled by default to minimize the executable code size.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		

(table continues...)

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

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

1.3.1.9.9 GptRunTimeErrorDetect

Table 45 **Specification for GptRunTimeErrorDetect**

Name	`GptRunTimeErrorDetect`		
Description	Enables or disables the Runtime Error detection and reporting. If safety is enabled(by setting the parameter GptSafetyEnable to true), GptRunTimeErrorDetect shall be set to true. true: enabled (ON). false: disabled (OFF).		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	GptSafetyEnable		
Autosar Version	Applicable for Autosar version 4.4.0.		

1.3.1.9.10 GptSafetyEnable

Table 46 **Specification for GptSafetyEnable**

Name	`GptSafetyEnable`

(table continues...)

Table 46 **(continued) Specification for GptSafetyEnable**

Description	Switch to enable/disable the safety check and reporting. *Note: The detection of safety related errors are enabled by default to ensure the safety issues are addressed during the product life cycle.*		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	TRUE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.10 Container: GptWakeupConfiguration

Function pointer to callback function (for wakeup notification).

Post-Build Variant Multiplicity: FALSE

Multiplicity Configuration Class: Post-Build

1.3.1.10.1 GptWakeupSourceRef

Table 47 **Specification for GptWakeupSourceRef**

Name	`GptWakeupSourceRef`		
Description	In case the wakeup-capability is true this value is transmitted to the ECU State Manager. Since the name of the dependent container is user configurable, the default value is kept as NULL		
Multiplicity	1..1	**Type**	EcucSymbolicNameReferenceDef
Range	Reference to Node: EcuMWakeupSource		
Default value	NA		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL

(table continues...)

Table 47 (continued) Specification for GptWakeupSourceRef

Dependency	-
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.2 Functions - Type definitions

This section list all the data types of the GPT driver.

1.3.2.1 Gpt_ChannelType

Table 48 Specification for Gpt_ChannelType

Syntax	`Gpt_ChannelType`	
Type	uint8	
File	`Gpt.h`	
Range	0 - 196	The range is based on the number of TOM channels, ATOM channels and GPT12 timer channels for the device variant. The maximum number of channels will vary depending on the device variant. 197 is for the superset device variant.
Description	Numeric ID of a GPT channel. *Note: Maximum number of channels depends on the device.*	
Source	AUTOSAR	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.2 Gpt_ModeType

Table 49 Specification for Gpt_ModeType

Syntax	`Gpt_ModeType`	
Type	Enumeration	
File	`Gpt.h`	
Range	1 - GPT_MODE_NORMAL	Normal operation mode of the GPT
	0 - GPT_MODE_SLEEP	Operation for reduced power operation mode. In sleep mode only wakeup capable channels are available.
Description	None	
Source	AUTOSAR	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.3 Gpt_NotificationPtrType

Table 50 Specification for Gpt_NotificationPtrType

Syntax	`Gpt_NotificationPtrType`
Type	Pointer to a function of type void Function_Name (void)
File	`Gpt.h`
Description	Timer Channel notification function pointer type.
Source	IFX
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.2.4 Gpt_PredefTimerType

Table 51 Specification for Gpt_PredefTimerType

Syntax	`Gpt_PredefTimerType`	
Type	Enumeration	
File	`Gpt.h`	
Range	0 - GPT_PREDEF_TIMER_1US_16BIT	GPT Predef Timer with tick duration 1µs and range 16bit
	1 - GPT_PREDEF_TIMER_1US_24BIT	GPT Predef Timer with tick duration 1µs and range 24bit
	2 - GPT_PREDEF_TIMER_1US_32BIT	GPT Predef Timer with tick duration 1µs and range 32bit
	3 - GPT_PREDEF_TIMER_100US_32BIT	GPT Predef Timer with tick duration 100µs and range 32bit
Description	Type for GPT Predef Timers	
Source	AUTOSAR	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.5 Gpt_ConfigType

Table 52 Specification for Gpt_ConfigType

Syntax	`Gpt_ConfigType`	
Type	Structure	
File	`Gpt.h`	
Range	-[]	The elements of the data structure are specific to the micro-controller.
Description	This is the type of the data structure including the configuration set required for initializing the GPT timer unit.	
Source	AUTOSAR	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.6 Gpt_ValueType

Table 53 Specification for Gpt_ValueType

Syntax	`Gpt_ValueType`	
Type	uint32	
File	`Gpt.h`	
Range	0 - 0xFFFFFF	Maximum supported timer value/setting periodic timer values (in number of ticks).
Description	Type for reading and setting the timer values (in number of ticks).	
Source	AUTOSAR	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3 Functions - APIs

This section lists all the APIs of the GPT driver.

1.3.3.1 Gpt_GetVersionInfo

Table 54 Specification for `Gpt_GetVersionInfo` API

Syntax	`void Gpt_GetVersionInfo`
	`(`
	`    Std_VersionInfoType * const VersionInfoPtr`
	`)`	

Service ID	0x00	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant	
Parameters (in)	-	-
Parameters (out)	VersionInfoPtr	Pointer where the version information of this driver is stored.
Parameters (in - out)	-	-
Return	void	-
Description	The function provides the version information of the GPT driver.	
Source	AUTOSAR	
Error handling	GPT_E_PARAM_POINTER	
Configuration dependencies	GptVersionInfoApi	
User hints	None	

(table continues...)

Table 54 (continued) Specification for `Gpt_GetVersionInfo` API

SFR accessed	-
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.2 Gpt_Init

Table 55 Specification for `Gpt_Init` API

Syntax	`void Gpt_Init`
	`(`
	`const Gpt_ConfigType * const ConfigPtr`
	`)`	

Service ID	0x01	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant	
Parameters (in)	ConfigPtr	Pointer to a selected configuration structure
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The function initializes the GPT driver in the context of the core from where the API is invoked. *Note: Initialization should be performed in the following sequence:* *1. Initialize Gpt driver from the master core.* *2. Initialize Gpt driver from all applicable slave cores.*	
Source	AUTOSAR	
Error handling	GPT_E_ALREADY_INITIALIZED, GPT_E_INIT_FAILED, GPT_E_MASTER_UNINIT, GPT_E_NOT_CONFIGURED	
Configuration dependencies	-	
User hints	None	

(table continues...)

Table 55 (continued) Specification for `Gpt_Init` API

SFR accessed	CPU_CORE_ID(r), GPT12_PISEL(rw), GPT12_T2(w), GPT12_T2CON(w), GPT12_T3(w), GPT12_T3CON(w), GPT12_T4(w), GPT12_T4CON(w), GPT12_T5(w), GPT12_T5CON(w), GPT12_T6(w), GPT12_T6CON(w), GTM_ATOM_AGC_ENDIS_CTRL(rw), GTM_ATOM_AGC_FUPD_CTRL(rw), GTM_ATOM_AGC_GLB_CTRL(w), GTM_ATOM_AGC_OUTEN_CTRL(rw), GTM_ATOM_CH_CM0(w), GTM_ATOM_CH_CM1(w), GTM_ATOM_CH_CN0(w), GTM_ATOM_CH_CTRL(w), GTM_ATOM_CH_IRQ_EN(w), GTM_ATOM_CH_IRQ_MODE(w), GTM_ATOM_CH_IRQ_NOTIFY(w), GTM_ATOM_CH_SR0(w), GTM_ATOM_CH_SR1(w), GTM_TOM_CH_CM0(w), GTM_TOM_CH_CM1(w), GTM_TOM_CH_CN0(w), GTM_TOM_CH_CTRL(w), GTM_TOM_CH_IRQ_EN(w), GTM_TOM_CH_IRQ_MODE(w), GTM_TOM_CH_IRQ_NOTIFY(w), GTM_TOM_CH_SR0(w), GTM_TOM_CH_SR1(w), GTM_TOM_TGC0_ENDIS_CTRL(rw), GTM_TOM_TGC0_ENDIS_STAT(w), GTM_TOM_TGC0_FUPD_CTRL(rw), GTM_TOM_TGC0_GLB_CTRL(w), GTM_TOM_TGC0_OUTEN_CTRL(rw), GTM_TOM_TGC0_OUTEN_STAT(w), GTM_TOM_TGC1_ENDIS_CTRL(rw), GTM_TOM_TGC1_ENDIS_STAT(w), GTM_TOM_TGC1_FUPD_CTRL(rw), GTM_TOM_TGC1_GLB_CTRL(w), GTM_TOM_TGC1_OUTEN_CTRL(rw), GTM_TOM_TGC1_OUTEN_STAT(w), SCU_CCUCON0(r), SCU_OSCCON(r), SCU_SYSPLLCON0(r), SCU_SYSPLLCON1(r), STM_TIM0(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.3 Gpt_DeInit

Table 56 Specification for `Gpt_DeInit` API

Syntax	`void Gpt_DeInit`
	`(`
	`  void`
	`)`	

Service ID	0x02	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant	
Parameters (in)	-	-
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The function de-initializes the GPT driver in the context of core from where this API is invoked. *Note: All slave cores must be de-initialized before the master core.*	

(table continues...)

Table 56 (continued) Specification for `Gpt_DeInit` API

Source	AUTOSAR
Error handling	GPT_E_UNINIT, GPT_E_BUSY, GPT_E_SLAVE_INIT
Configuration dependencies	GptDeinitApi
User hints	None
SFR accessed	CPU_CORE_ID(r), GPT12_PISEL(rw), GPT12_T2(w), GPT12_T2CON(w), GPT12_T3(w), GPT12_T3CON(w), GPT12_T4(w), GPT12_T4CON(w), GPT12_T5(w), GPT12_T5CON(w), GPT12_T6(w), GPT12_T6CON(w), GTM_ATOM_AGC_ENDIS_CTRL(w), GTM_ATOM_AGC_ENDIS_STAT(w), GTM_ATOM_AGC_GLB_CTRL(w), GTM_ATOM_AGC_OUTEN_CTRL(w), GTM_ATOM_AGC_OUTEN_STAT(w), GTM_ATOM_CH_IRQ_NOTIFY(w), GTM_TOM_CH_IRQ_NOTIFY(w), GTM_TOM_TGC0_ENDIS_CTRL(w), GTM_TOM_TGC0_ENDIS_STAT(w), GTM_TOM_TGC0_GLB_CTRL(w), GTM_TOM_TGC0_OUTEN_CTRL(w), GTM_TOM_TGC0_OUTEN_STAT(w), GTM_TOM_TGC1_ENDIS_CTRL(w), GTM_TOM_TGC1_ENDIS_STAT(w), GTM_TOM_TGC1_GLB_CTRL(w), GTM_TOM_TGC1_OUTEN_CTRL(w), GTM_TOM_TGC1_OUTEN_STAT(w), SCU_CCUCON0(r), SCU_OSCCON(r), SCU_SYSPLLCON0(r), SCU_SYSPLLCON1(r), STM_TIM0(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.4 Gpt_StartTimer

Table 57 Specification for `Gpt_StartTimer` API

Syntax	`void  Gpt_StartTimer`
	`(`
	`    const Gpt_ChannelType Channel,`
	`    const Gpt_ValueType Value`
	`)`	

Service ID	0x05	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channels	
Parameters (in)	Channel Value	Numeric identifier of the GPT channel. Target time in number of ticks.
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-

(table continues...)

Table 57 (continued) Specification for `Gpt_StartTimer` API

Description	The function starts the selected timer 'Channel' with a defined target time, 'Value', only if the channel is available in the core from where this API is invoked. *Note: The maximum range for the target time, 'Value', depends on the type of the timer channel. If the 'Channel' is:* *1. TOM channel, the MaxValue is 0xFFFF (16 Bit).* *2. ATOM channel, the MaxValue is 0xFFFFFF (24 Bit).*
Source	AUTOSAR
Error handling	GPT_E_UNINIT, GPT_E_PARAM_VALUE, GPT_E_PARAM_CHANNEL, GPT_E_BUSY, GPT_E_CORE_CHANNEL_MISMATCH, GPT_E_INVALID_START
Configuration dependencies	-
User hints	None
SFR accessed	CPU_CORE_ID(r), GPT12_CAPREL(w), GPT12_T2(w), GPT12_T2CON(rw), GPT12_T3(w), GPT12_T3CON(rw), GPT12_T4(w), GPT12_T4CON(rw), GPT12_T5(w), GPT12_T5CON(rw), GPT12_T6(w), GPT12_T6CON(rw), GTM_ATOM_CH_CM0(w), GTM_ATOM_CH_CN0(w), GTM_ATOM_CH_CTRL(r), GTM_ATOM_CH_IRQ_NOTIFY(w), GTM_ATOM_CH_SR0(w), GTM_TOM_CH_CM0(w), GTM_TOM_CH_CN0(w), GTM_TOM_CH_CTRL(r), GTM_TOM_CH_IRQ_NOTIFY(w), GTM_TOM_CH_SR0(w), GTM_TOM_TGC0_ENDIS_CTRL(w), GTM_TOM_TGC0_ENDIS_STAT(w), GTM_TOM_TGC0_OUTEN_CTRL(w), GTM_TOM_TGC0_OUTEN_STAT(w), GTM_TOM_TGC1_ENDIS_CTRL(w), GTM_TOM_TGC1_ENDIS_STAT(w), GTM_TOM_TGC1_OUTEN_CTRL(w), GTM_TOM_TGC1_OUTEN_STAT(w), SCU_CCUCON0(r), SCU_OSCCON(r), SCU_SYSPLLCON0(r), SCU_SYSPLLCON1(r), STM_TIM0(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.5 Gpt_StopTimer

Table 58 Specification for `Gpt_StopTimer` API

Syntax	`void Gpt_StopTimer`
	`(`
	`    const Gpt_ChannelType Channel`
	`)`	

Service ID	0x06	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channels	
Parameters (in)	Channel	Numeric identifier of the GPT channel.

(table continues...)

Table 58 (continued) Specification for `Gpt_StopTimer` API

Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The function stops the selected timer 'Channel', only if the channel is available in the core from where this API is invoked.	
Source	AUTOSAR	
Error handling	GPT_E_UNINIT, GPT_E_PARAM_CHANNEL, GPT_E_CORE_CHANNEL_MISMATCH	
Configuration dependencies	-	
User hints	None	
SFR accessed	CPU_CORE_ID(r), GPT12_T2CON(rw), GPT12_T3CON(rw), GPT12_T4CON(rw), GPT12_T5CON(rw), GPT12_T6CON(rw), GTM_ATOM_AGC_ENDIS_CTRL(w), GTM_ATOM_AGC_ENDIS_STAT(w), GTM_ATOM_AGC_OUTEN_CTRL(w), GTM_ATOM_AGC_OUTEN_STAT(w), GTM_TOM_TGC0_ENDIS_CTRL(w), GTM_TOM_TGC0_ENDIS_STAT(w), GTM_TOM_TGC0_OUTEN_CTRL(w), GTM_TOM_TGC0_OUTEN_STAT(w), GTM_TOM_TGC1_ENDIS_CTRL(w), GTM_TOM_TGC1_ENDIS_STAT(w), GTM_TOM_TGC1_OUTEN_CTRL(w), GTM_TOM_TGC1_OUTEN_STAT(w), SCU_CCUCON0(r), SCU_OSCCON(r), SCU_SYSPLLCON0(r), SCU_SYSPLLCON1(r), STM_TIM0(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.6 Gpt_GetTimeElapsed

Table 59 Specification for `Gpt_GetTimeElapsed` API

Syntax	`Gpt_ValueType  Gpt_GetTimeElapsed`
	`(`
	`    const Gpt_ChannelType Channel`
	`)`	

Service ID	0x03	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant	
Parameters (in)	Channel	Numeric identifier of the GPT channel.

(table continues...)

Table 59 (continued) Specification for `Gpt_GetTimeElapsed` API

Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Gpt_ValueType	Remaining timer value (in number of ticks).
Description	The function provides the time elapsed in ticks since the previous timer overflow/start, only if the channel is available in the core from where this API is invoked.	
Source	AUTOSAR	
Error handling	GPT_E_UNINIT, GPT_E_PARAM_CHANNEL, GPT_E_CORE_CHANNEL_MISMATCH	
Configuration dependencies	GptTimeElapsedApi	
User hints	None	
SFR accessed	CPU_CORE_ID(r), GPT12_T2(r), GPT12_T3(r), GPT12_T3CON(r), GPT12_T4(r), GPT12_T5(r), GPT12_T6(r), GPT12_T6CON(r), GTM_ATOM_CH_CM0(r), GTM_ATOM_CH_CN0(r), GTM_ATOM_CH_CTRL(r), GTM_ATOM_CH_IRQ_NOTIFY(r), GTM_TOM_CH_CM0(r), GTM_TOM_CH_CN0(r), GTM_TOM_CH_CTRL(r), GTM_TOM_CH_IRQ_NOTIFY(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.7 Gpt_GetTimeRemaining

Table 60 Specification for `Gpt_GetTimeRemaining` API

Syntax	`Gpt_ValueType  Gpt_GetTimeRemaining`
	`(`
	`    const Gpt_ChannelType Channel`
	`)`	

Service ID	0x04	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant	
Parameters (in)	Channel	Numeric identifier of the GPT channel.
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Gpt_ValueType	Remaining timer value (in number of ticks).

(table continues...)

Table 60 **(continued) Specification for `Gpt_GetTimeRemaining` API**

Description	The function returns the time remaining until the target time is reached, only if the channel is available in the core from where this API is invoked..
Source	AUTOSAR
Error handling	GPT_E_UNINIT, GPT_E_PARAM_CHANNEL, GPT_E_CORE_CHANNEL_MISMATCH
Configuration dependencies	GptTimeRemainingApi
User hints	None
SFR accessed	CPU_CORE_ID(r), GPT12_T2(r), GPT12_T3(r), GPT12_T3CON(r), GPT12_T4(r), GPT12_T5(r), GPT12_T6(r), GPT12_T6CON(r), GTM_ATOM_CH_CM0(r), GTM_ATOM_CH_CN0(r), GTM_ATOM_CH_CTRL(r), GTM_ATOM_CH_IRQ_NOTIFY(r), GTM_TOM_CH_CM0(r), GTM_TOM_CH_CN0(r), GTM_TOM_CH_CTRL(r), GTM_TOM_CH_IRQ_NOTIFY(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.8 Gpt_GetPredefTimerValue

Table 61 **Specification for `Gpt_GetPredefTimerValue` API**

Syntax	`Std_ReturnType  Gpt_GetPredefTimerValue`
	`(`
	`    const Gpt_PredefTimerType PredefTimer,`
	`    uint32 * const TimeValuePtr`
	`)`	

Service ID	0x0D	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant	
Parameters (in)	PredefTimer	GPT Predef Timer
Parameters (out)	TimeValuePtr	Pointer to time value destination data in RAM
Parameters (in - out)	-	-
Return	Std_ReturnType	E_OK: No error has been detected E_NOT_OK: Aborted due to errors
Description	The function delivers the current value of the desired GPT Predef Timer. This function can be invoked from any core.	
Source	AUTOSAR	

(table continues...)

Table 61 (continued) Specification for `Gpt_GetPredefTimerValue` API

Error handling	GPT_E_MODE, GPT_E_UNINIT, GPT_E_PARAM_PREDEF_TIMER, GPT_E_PARAM_POINTER
Configuration dependencies	-
User hints	None
SFR accessed	CPU_CORE_ID(r), GTM_TOM_CH_CN0(r), GTM_TOM_CH_IRQ_NOTIFY(rw) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.9 Gpt_EnableNotification

Table 62 Specification for `Gpt_EnableNotification` API

Syntax	`void  Gpt_EnableNotification`
	`(`
	`    const Gpt_ChannelType Channel`
	`)`	

Service ID	0x07	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channels	
Parameters (in)	Channel	Numeric identifier of the GPT channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The function enables reporting of Notification for the GPT channel.	
Source	AUTOSAR	
Error handling	GPT_E_UNINIT, GPT_E_PARAM_CHANNEL, GPT_E_CORE_CHANNEL_MISMATCH	
Configuration dependencies	GptEnableDisableNotificationApi	
User hints	None	
SFR accessed	CPU_CORE_ID(r), GTM_ATOM_CH_IRQ_EN(w), GTM_ATOM_CH_IRQ_NOTIFY(w), GTM_TOM_CH_IRQ_EN(w), GTM_TOM_CH_IRQ_NOTIFY(w) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	

(table continues...)

Table 62 (continued) Specification for `Gpt_EnableNotification` API

Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.10 Gpt_DisableNotification

Table 63 Specification for `Gpt_DisableNotification` API

Syntax	`void  Gpt_DisableNotification`
	`(`
	`    const Gpt_ChannelType Channel`
	`)`	

Service ID	0x08	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channels	
Parameters (in)	Channel	Numeric identifier of the GPT channel.
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The function disables the interrupt notification for a channel (relevant in normal mode).	
Source	AUTOSAR	
Error handling	GPT_E_UNINIT, GPT_E_PARAM_CHANNEL, GPT_E_CORE_CHANNEL_MISMATCH	
Configuration dependencies	GptEnableDisableNotificationApi	
User hints	None	
SFR accessed	CPU_CORE_ID(r), GTM_ATOM_CH_IRQ_EN(w), GTM_ATOM_CH_IRQ_NOTIFY(w), GTM_TOM_CH_IRQ_EN(w), GTM_TOM_CH_IRQ_NOTIFY(w) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.11 Gpt_EnableWakeup

Table 64 Specification for `Gpt_EnableWakeup` API

Syntax	`void Gpt_EnableWakeup`
	`(`
	`    const Gpt_ChannelType Channel`
	`)`	

Service ID	0x0B	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channels	
Parameters (in)	Channel	Numeric identifier of the GPT channel.
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The function enables wakeup capability for the GPT Channel.	
Source	AUTOSAR	
Error handling	GPT_E_UNINIT, GPT_E_PARAM_CHANNEL, GPT_E_CORE_CHANNEL_MISMATCH	
Configuration dependencies	GptWakeupFunctionalityApi,GptReportWakeupSource	
User hints	None	
SFR accessed	CPU_CORE_ID(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.12 Gpt_DisableWakeup

Table 65 Specification for `Gpt_DisableWakeup` API

Syntax	`void Gpt_DisableWakeup`
	`(`
	`    const Gpt_ChannelType Channel`
	`)`

Service ID	0x0A
Sync/Async	Synchronous
Safety Level	Refer to the release notes for the safety related info

(table continues...)

Table 65 (continued) Specification for `Gpt_DisableWakeup` API

Re-entrancy	Reentrant for different channels	
Parameters (in)	Channel	Numeric identifier of the GPT channel.
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The function disables the wakeup interrupt of the channel (relevant in sleep mode only).	
Source	AUTOSAR	
Error handling	GPT_E_UNINIT, GPT_E_PARAM_CHANNEL, GPT_E_CORE_CHANNEL_MISMATCH	
Configuration dependencies	GptReportWakeupSource	
User hints	This API has no impact if invoked in Normal Mode.	
SFR accessed	CPU_CORE_ID(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.13 Gpt_CheckWakeup

Table 66 Specification for `Gpt_CheckWakeup` API

Syntax	`void Gpt_CheckWakeup`
	`(`
	`    const EcuM_WakeupSourceType WakeupSource`
	`)`	

Service ID	0x0C	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant	
Parameters (in)	WakeupSource	Information on wakeup source to be checked. The associated GPT channel can be determined from configuration data.
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-

(table continues...)

Table 66 (continued) Specification for `Gpt_CheckWakeup` API

Description	The function checks if a wakeup capable GPT channel is the source for a wakeup event and calls the ECU state manager service, EcuM_SetWakeupEvent, in case of a valid GPT channel wakeup event.
Source	AUTOSAR
Error handling	GPT_E_UNINIT
Configuration dependencies	GptWakeupFunctionalityApi,GptReportWakeupSource
User hints	None
SFR accessed	CPU_CORE_ID(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.14 Gpt_SetMode

Table 67 Specification for `Gpt_SetMode` API

Syntax	`void  Gpt_SetMode`
	`(`
	`    const Gpt_ModeType Mode`
	`)`	

Service ID	0x09	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant	
Parameters (in)	Mode	GPT_MODE_NORMAL: Normal operation mode of the GPT driver GPT_MODE_SLEEP: Sleep mode of the GPT driver (wakeup capable).
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The function sets the mode of the GPT Driver to 'SLEEP' mode or 'NORMAL' mode.	
Source	AUTOSAR	
Error handling	GPT_E_PARAM_MODE, GPT_E_UNINIT	
Configuration dependencies	GptWakeupFunctionalityApi,GptReportWakeupSource	

(table continues...)

Table 67 (continued) Specification for `Gpt_SetMode` API

User hints	For Predef timers only the master can start or stop the timers.
SFR accessed	CPU_CORE_ID(r), GPT12_T2CON(rw), GPT12_T3CON(rw), GPT12_T4CON(rw), GPT12_T5CON(rw), GPT12_T6CON(rw), GTM_ATOM_AGC_ENDIS_CTRL(w), GTM_ATOM_AGC_ENDIS_STAT(w), GTM_ATOM_AGC_OUTEN_CTRL(w), GTM_ATOM_AGC_OUTEN_STAT(w), GTM_ATOM_CH_IRQ_EN(w), GTM_ATOM_CH_IRQ_NOTIFY(w), GTM_TOM_CH_CM0(w), GTM_TOM_CH_CM1(w), GTM_TOM_CH_CN0(w), GTM_TOM_CH_CTRL(w), GTM_TOM_CH_IRQ_EN(w), GTM_TOM_CH_IRQ_MODE(w), GTM_TOM_CH_IRQ_NOTIFY(w), GTM_TOM_CH_SR0(w), GTM_TOM_CH_SR1(w), GTM_TOM_TGC0_ENDIS_CTRL(rw), GTM_TOM_TGC0_ENDIS_STAT(w), GTM_TOM_TGC0_FUPD_CTRL(rw), GTM_TOM_TGC0_GLB_CTRL(w), GTM_TOM_TGC0_OUTEN_CTRL(rw), GTM_TOM_TGC0_OUTEN_STAT(w), GTM_TOM_TGC1_ENDIS_CTRL(rw), GTM_TOM_TGC1_ENDIS_STAT(w), GTM_TOM_TGC1_FUPD_CTRL(rw), GTM_TOM_TGC1_GLB_CTRL(w), GTM_TOM_TGC1_OUTEN_CTRL(rw), GTM_TOM_TGC1_OUTEN_STAT(w), SCU_CCUCON0(r), SCU_OSCCON(r), SCU_SYSPLLCON0(r), SCU_SYSPLLCON1(r), STM_TIM0(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.15 Gpt_InitCheck

Table 68 Specification for `Gpt_InitCheck` API

Syntax	`Std_ReturnType  Gpt_InitCheck`
	`(`
	`    const Gpt_ConfigType ConfigPtr`
	`)`	

Service ID	0x0E	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant	
Parameters (in)	ConfigPtr	Pointer to a selected configuration structure
Parameters (out)	-	-
Parameters (in - out)	-	-

(table continues...)

Table 68 (continued) Specification for `Gpt_InitCheck` API

Return	Std_ReturnType	E_OK - Initialization comparison is success E_NOT_OK: In case of - Driver is not initialized for current core - Input config Pointer is Null - Input config pointer is other than the one used for Init - Gpt Driver is in Wrong mode in current core - Any of the channels allocated to the current core is not initialized - If predefined timer is not initialized when current core is master core
Description	The function verifies the initialization of the GPT driver in context to the core from where the API is invoked. *Note: Init check should be performed in the following sequence:* *1. Call Gpt_Init from a core (master/salve core).* *2. Call Gpt_InitCheck from the same core.* *Note: The function can run concurrently across cores.*	
Source	IFX	
Error handling	-	
Configuration dependencies	GptInitCheckApi	
User hints	None	
SFR accessed	GPT12_CLC(r), GPT12_PISEL(r), GPT12_T2CON(r), GPT12_T3CON(r), GPT12_T4CON(r), GPT12_T5CON(r), GPT12_T6CON(r), GTM_ATOM_AGC_ENDIS_STAT(r), GTM_ATOM_CH_CM0(r), GTM_ATOM_CH_CM1(r), GTM_ATOM_CH_CN0(r), GTM_ATOM_CH_CTRL(r), GTM_ATOM_CH_IRQ_EN(r), GTM_ATOM_CH_IRQ_MODE(r), GTM_ATOM_CH_SR0(r), GTM_ATOM_CH_SR1(r), GTM_TOM_CH_CM0(r), GTM_TOM_CH_CM1(r), GTM_TOM_CH_CN0(r), GTM_TOM_CH_CTRL(r), GTM_TOM_CH_IRQ_EN(r), GTM_TOM_CH_IRQ_MODE(r), GTM_TOM_CH_SR0(r), GTM_TOM_CH_SR1(r), GTM_TOM_TGC0_ENDIS_STAT(r), GTM_TOM_TGC1_ENDIS_STAT(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.4 Notifications and Callbacks

The GPT driver does not provide any notifications and callbacks.

1.3.5 Scheduled functions

The GPT driver does not provide any scheduled functions.

1.3.6 Interrupt service routines

This section lists all the interrupt handlers of the GPT driver.

1.3.6.1 Gpt_Isr

Table 69 Specification for `Gpt_Isr` API

Syntax	`void  Gpt_Isr`
	`(`
	`    const uint32 LogicalChId,`
	`    const uint32 StatusFlags`
	`)`	

Service ID	0x0F	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channels	
Parameters (in)	LogicalChId StatusFlags	GPT channel number This flag is not used.
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The ISR function is used to call the user-defined notification function as well as the Ecum_CheckWakeup function for wakeup notification.	
Source	IFX	
Error handling	GPT_E_NOT_CONFIGURED	
Configuration dependencies	GptEnableDisableNotificationApi,GptReportWakeupSource,GptWakeupFunctionalityApi	
User hints	None	
SFR accessed	CPU_CORE_ID(r), GPT12_T2CON(rw), GPT12_T3CON(rw), GPT12_T4CON(rw), GPT12_T5CON(rw), GPT12_T6CON(rw), GTM_ATOM_CH_CM0(r), GTM_ATOM_CH_CN0(w), GTM_TOM_CH_CM0(r), GTM_TOM_CH_CN0(w), GTM_TOM_TGC0_ENDIS_CTRL(w), GTM_TOM_TGC0_ENDIS_STAT(w), GTM_TOM_TGC0_OUTEN_CTRL(w), GTM_TOM_TGC0_OUTEN_STAT(w), GTM_TOM_TGC1_ENDIS_CTRL(w), GTM_TOM_TGC1_ENDIS_STAT(w), GTM_TOM_TGC1_OUTEN_CTRL(w), GTM_TOM_TGC1_OUTEN_STAT(w), SCU_CCUCON0(r), SCU_OSCCON(r), SCU_SYSPLLCON0(r), SCU_SYSPLLCON1(r), STM_TIM0(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.7 Callout

The GPT driver does not provide any callout.

1.3.8 Errors Handling

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

Error Name: Description	Source	Error ID (AS422)	Type (AS422)	Error ID (AS440)	Type (AS440)
GPT_E_ALREADY_INITIALIZED: Error reported when the driver is already initialized for the current core.	AUTOSAR	0xD	DET	0xD	DET
GPT_E_BUSY: Error reported when timer channel is still busy (running).	AUTOSAR	0xB	DET	0xB	RUNTIME
GPT_E_CORE_CHANNEL_MISMATCH: Error reported when the parameter 'Channel' is not configured for the core from where the API is invoked. For the APIs Gpt_GetTimeElapsed and Gpt_GetTimeRemaining, this error is not applicable when the parameter GptReadAcrossCores is ON.	IFX	0x65	DET	0x65	DET
GPT_E_INIT_FAILED: Error reported when Gpt_Init is called with null pointer as argument.	AUTOSAR	0xE	DET	0xE	DET
GPT_E_INVALID_START: Error reported when Gpt_StartTimer is called when Gpt driver is in sleep mode.	IFX	0xC8	SAFETY	0xC8	SAFETY
GPT_E_MASTER_UNINIT: Error reported when slave core initializing is called without initializing Master core.	IFX	0x66	DET	0x66	DET
GPT_E_MODE: Error reported when the driver in sleep mode	AUTOSAR	0x0C	DET	0x0C	RUNTIME
GPT_E_NOT_CONFIGURED: Error reported when GPT driver is not configured for the core from where the API is invoked.	IFX	0x64	DET	0x64	DET
GPT_E_PARAM_CHANNEL: Error reported when the parameter 'Channel' is invalid.	AUTOSAR	0x14	DET	0x14	DET

Error Name: Description	Source	Error ID (AS422)	Type (AS422)	Error ID (AS440)	Type (AS440)
GPT_E_PARAM_MODE: Error reported when Gpt_SetMode is called with invalid parameter (Mode).	AUTOSAR	0x1F	DET	0x1F	DET
GPT_E_PARAM_POINTER: Error reported when the parameter passed is a null pointer.	AUTOSAR	0x16	DET	0x16	DET
GPT_E_PARAM_PREDEF_TIMER: Error reported when the parameter Predef Timer is invalid.	AUTOSAR	0x17	DET	0x17	DET
GPT_E_PARAM_VALUE: Error reported when the value is not within the allowed range.	AUTOSAR	0x15	DET	0x15	DET
GPT_E_SLAVE_INIT: Error reported when master de-Initialization is called without de-Initializing slave core.	IFX	0x67	DET	0x67	DET
GPT_E_UNINIT: Error reported when the driver is not initialized in context to the core from where the API is invoked.	AUTOSAR	0X0A	DET	0X0A	DET

1.3.9 Deviations and limitations

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

1.3.9.1 Deviations

This section describes the deviation of the GPT driver.

1.3.9.1.1 Software specification deviations

This section describes the deviations from software specification.

Table 70 Known deviations

Reference	Deviation
Predefined timer	The maximum value for 16-bit predefined timer must be 65535, butdue to hardware limitation the maximum count for 16-bit predefinedtimer is 65534.

(table continues...)

Table 70 (continued) Known deviations

Reference	Deviation
Safety error for unintended service request	Safety Error will be reported for spurious interrupts when interrupt occurs for adjacent TOM/ATOM channels which shares the same interrupts or consecutive interrupts for same TOM/ATOM channel occurs and the second hardware interrupt triggered in between the start of an ISR and clearing of the interrupt flag. Refer to 'Reporting of unintended service requests' in MCAL User Manual General.

1.3.9.1.2 AMDC Violations

The GPT driver does not have any AMDC violations.

1.3.9.1.3 VSMD Violations

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

Table 71 Violations reported by VSMD checker tool for EB03

Rule ID:	EB03
VSMD Node(s):	/AURIX2G/EcucDefs/Gpt/GptChannelConfigSet/GptChannelConfiguration/GptNotification /AURIX2G/EcucDefs/Gpt/GptChannelConfigSet/GptChannelConfiguration/GptWakeupConfiguration /AURIX2G/EcucDefs/Gpt/GptDriverConfiguration/GptKernelEcucPartitionRef
Description:	The StMD node has LOWER-MULTIPLICITY=0 and UPPER-MULTIPLICITY=1. The VSMD-node shall get the OPTIONAL-attribute instead of creating a list!
Additional Information:	-

Table 72 Violations reported by VSMD checker tool for EB09

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

Table 73 Violations reported by VSMD checker tool for EcucSws_1014

Rule ID:	EcucSws_1014

(table continues...)

Table 73 (continued) Violations reported by VSMD checker tool for EcucSws_1014

VSMD Node(s):	/AURIX2G/EcucDefs/Gpt/AURIX2G/EcucDefs/Gpt/GptChannelConfigSet/GptChannelConfiguration /AURIX2G/EcucDefs/Gpt/GptDriverConfiguration
Description:	Additional vendor specific parameter definitions (using ParameterTypes), container definitions and references shall be added to the VSMD according to the alphabetical order.
Additional Information:	-

Table 74 Violations reported by VSMD checker tool for EcucSws_2101

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

Table 75 Violations reported by VSMD checker tool for EcucSws_1035

Rule ID:	EcucSws_1035

(table continues...)

Table 75 (continued) Violations reported by VSMD checker tool for EcucSws_1035

VSMD Node(s):	/AURIX2G/EcucDefs/Gpt/AURIX2G/EcucDefs/Gpt/GptChannelConfigSet /AURIX2G/EcucDefs/Gpt/GptChannelConfigSet/GptChannelConfiguration /AURIX2G/EcucDefs/Gpt/GptChannelConfigSet/GptChannelConfiguration/GptChannelClkSrcRef /AURIX2G/EcucDefs/Gpt/GptChannelConfigSet/GptChannelConfiguration/GptChannelEcucPartitionRef /AURIX2G/EcucDefs/Gpt/GptChannelConfigSet/GptChannelConfiguration/GptChannelId /AURIX2G/EcucDefs/Gpt/GptChannelConfigSet/GptChannelConfiguration/GptChannelMode /AURIX2G/EcucDefs/Gpt/GptChannelConfigSet/GptChannelConfiguration/GptChannelTickFrequency /AURIX2G/EcucDefs/Gpt/GptChannelConfigSet/GptChannelConfiguration/GptChannelTickValueMax /AURIX2G/EcucDefs/Gpt/GptChannelConfigSet/GptChannelConfiguration/GptEnableWakeup /AURIX2G/EcucDefs/Gpt/GptChannelConfigSet/GptChannelConfiguration/GptNotification /AURIX2G/EcucDefs/Gpt/GptChannelConfigSet/GptChannelConfiguration/GptWakeupConfiguration /AURIX2G/EcucDefs/Gpt/GptChannelConfigSet/GptChannelConfiguration/GptWakeupConfiguration/GptWakeupSourceRef /AURIX2G/EcucDefs/Gpt/GptConfigurationOfOptApiServices /AURIX2G/EcucDefs/Gpt/GptConfigurationOfOptApiServices/GptDeinitApi/ AURIX2G/EcucDefs/Gpt/GptConfigurationOfOptApiServices/GptEnableDisableNotificationApi /AURIX2G/EcucDefs/Gpt/GptConfigurationOfOptApiServices/GptTimeElapsedApi /AURIX2G/EcucDefs/Gpt/GptConfigurationOfOptApiServices/GptTimeRemainingApi /AURIX2G/EcucDefs/Gpt/GptConfigurationOfOptApiServices/GptVersionInfoApi /AURIX2G/EcucDefs/Gpt/GptConfigurationOfOptApiServices/GptWakeupFunctionalityApi /AURIX2G/EcucDefs/Gpt/GptDriverConfiguration

(table continues...)

Table 75 (continued) Violations reported by VSMD checker tool for EcucSws_1035

	/AURIX2G/EcucDefs/Gpt/GptDriverConfiguration/GptClockReferencePoint /AURIX2G/EcucDefs/Gpt/GptDriverConfiguration/GptClockReferencePoint/GptClockReference /AURIX2G/EcucDefs/Gpt/GptDriverConfiguration/GptDevErrorDetect /AURIX2G/EcucDefs/Gpt/GptDriverConfiguration/GptEcucPartitionRef /AURIX2G/EcucDefs/Gpt/GptDriverConfiguration/GptKernelEcucPartitionRef /AURIX2G/EcucDefs/Gpt/GptDriverConfiguration/GptPredefTimer100us32bitEnable /AURIX2G/EcucDefs/Gpt/GptDriverConfiguration/GptPredefTimer1usEnablingGrade /AURIX2G/EcucDefs/Gpt/GptDriverConfiguration/GptReportWakeupSource
Description:	For Containers, Parameters and References elements UUID must be unique (also between StMD and VSMD).
Additional Information:	-

Table 76 Violations reported by VSMD checker tool for EcucSws_6003

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

Table 77 Violations reported by VSMD checker tool for TpsEcuc_06051_ASR41

Rule ID:	TpsEcuc_06051_ASR41
VSMD Node(s):	/AURIX2G/EcucDefs/Gpt/POST_BUILD_VARIANT_USED
Description:	The implementationConfigClass of an EcucParameterDef or EcucAbstractReferenceDef in VSMD shall be the same or higher (where PreCompile configuration class is considered to be the lowest and PostBuild the highest) as in StMD with respect to the selected subset defined by the actually implemented supportedConfigVariant.
Additional Information:	-

1.3.9.2 Limitations

This section describes the limitations of the GPT driver.

Table 78 Known limitations

Reference	Limitation
Gpt_SetMode API	The `Gpt_SetMode` API when called with **SLEEP** shall stop all non-wakeup capable timers. However, in multicore environment, the `Gpt_SetMode` API will stop the predefined timer, only when invoked from the logical master core.
Gpt_GetPredefTimerValue API	In a special case where there is no GPT channel allocated to a slave-core and the PREDEF timer must be accessed from this slave core, then at least one GPT timer channel shall be defined and allocated to the slave-core.

Revision history

Table 79 Revision history

Date	Version	Description
2023-12-07	6.0	Document is released
2023-12-01	5.1	- In section 1.1.4.6 Interrupt connections sample code, corrected the TOM/ATOM and GPT12 channel Ids corresponding to the interrupts. - In section 1.3.9.1, Added deviation for 'Safety error for unintended service requests'
2023-07-04	5.0	Document is released
2023-05-23	4.1	- Mcal_Wrapper.h is added in the "1.1.3.1 C file structure" section "Figure 2 Mcu_C_file_structure-1.png" and "Table 2 C file structure". - DEM module removed and Mcal_Wrapper module added in "1.1.4.1 Integration with AUTOSAR stack" section - ASIL level field changed to Safety level with value as "refer to release notes" for all APIs under "1.3.3 Functions - APIs" - In Section 1.3.3.13 Gpt_InitCheck() Api E_NOT_OK case description is updated to list all the conditions when E_NOT_OK is returned.
2022-08-02	4.0	Document is released
2022-07-29	3.1	Added info on accessing Predef timer from slave core in 'Limitations' section.
2021-11-16	3.0	Document is released
2021-11-16	2.1	Config variant attribute table information is removed and added this information in 'Configuration interfaces' section.
2020-11-17	2.0	Document is released
2020-11-13	1.1	Updated HSI of APIs in section 1.3.3
2020-08-13	1.0	Document is released
2020-08-03	0.1	- Initial Version - GPT driver chapter moved from MC-ISAR_TC3xx_UM_Basic to this document - Removed violation ECUC_Gpt_00330 : GptClockReference - Added VSMD violations - Added new configuration parameter GptReadAcrossCores - Added new configuration parameter GptRunTimeErrorDetect to handle Autosar version 4.4.0 changes. - Added dependency to files EcuM_Externals.h and EcuM.h in file structure to handle Autosar version 4.4.0 changes

Trademarks

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

Edition 2023-12-07

Published by

Infineon Technologies AG

81726 Munich, Germany

Do you have a question about any aspect of this document?

Email: erratum@infineon.com

Document reference
IFX-ocr1484806431059

Important notice

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

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

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

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

Warnings

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

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