Renesas Synergy™ Platform

GPT HAL Module Guide

R11AN0091EU0101
Rev.1.01
Aug 10, 2017

Introduction

This module guide will enable you to effectively use a module in your own design. Upon completion of this guide, you will be able to add this module to your own design, configure it correctly for the target application and write code, using the included application project code as a reference and efficient starting point. References to more detailed API descriptions and suggestions of other application projects that illustrate more advanced uses of the module are available in the Renesas Synergy Knowledge Base (as described in the References section at the end of this document) and should be valuable resources for creating more complex designs.

The General PWM Timer (GPT) HAL module is a high-level API for timer applications and is implemented on r_gpt. The GPT HAL module uses the GPT peripheral on the Synergy MCU. A user-defined callback can be created to respond to a timer event.

Contents

1. GPT HAL Module Features

The GPT HAL module configures a timer to a user-specified period. When the period elapses, any of the following events can occur:

- CPU interrupt that calls a user-callback function, if provided
- Toggle a port pin
- Data transfer using DMAC/DTC if configured with Transfer Interface
- Starting of another peripheral if configured with events and peripheral definitions

General PWM Timer (GPT)

- Multiple Channels
  - S7G2: 32-bit x 14 channels
  - S3A7: 32-bit x 10 channels
  - S124: 32-bit x 1 channel and 16-bit x 6 channels
- PCLKD as core clock
- Two I/O pins per channel
- Up/down-counting saw/triangle waves
- Two output compare and input capture registers
- Can be configured as a response to 8 Event Link Controllers (ELCs), and as an event in ELC

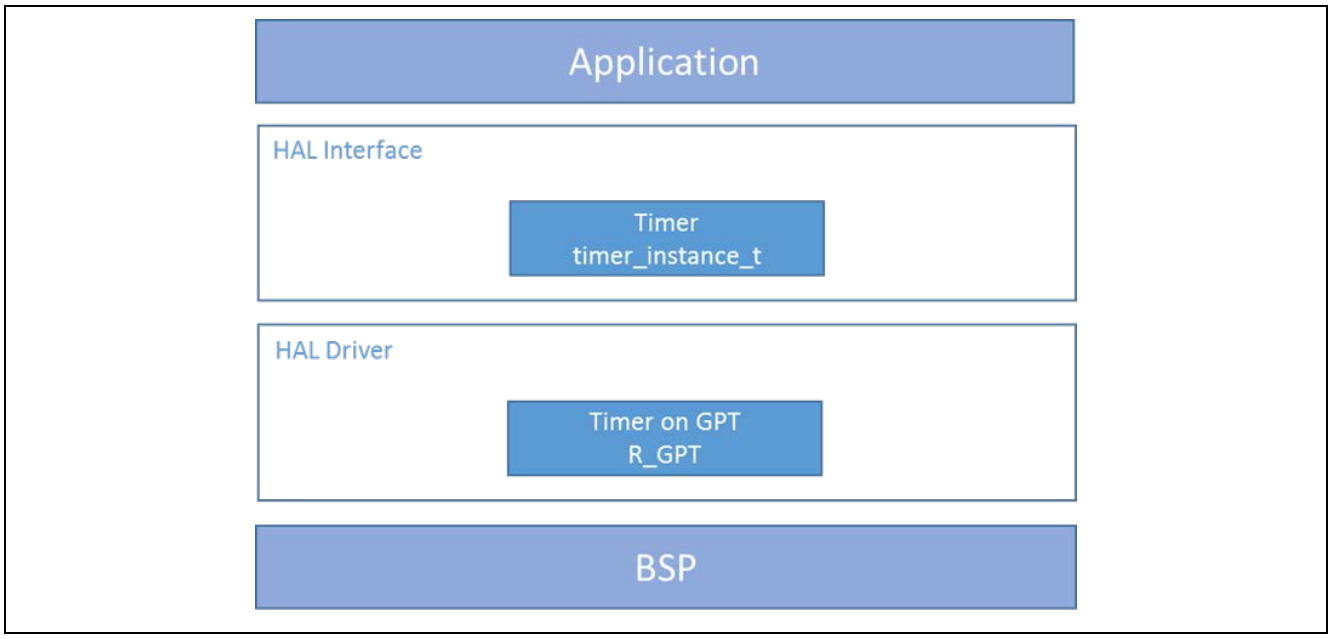

Figure 1 GPT HAL Module Block Diagram

2. GPT HAL Module APIs Overview

The GPT HAL module defines APIs for opening, closing, starting and stopping timers. A complete list of the available APIs, an example API call and a short description of each can be found in the following table. A table of status return values follows the API summary table.

Table 1 GPT HAL Module API Summary

Function Name	Example API Call and Description
.open	`g_timer0.p_api->open(g_timer0.p_ctrl, g_timer0.p_cfg)` Initial configuration.
.stop	`g_timer0.p_api->stop(g_timer0.p_ctrl)` Stop the counter.
.start	`g_timer0.p_api->start(g_timer0.p_ctrl)` Start the counter.
.reset	`g_timer0.p_api->reset(g_timer0.p_ctrl)` Reset the counter to the initial value.
.counterGet	`g_timer0.p_api->counterGet(g_timer0.p_ctrl, &value)` Get current counter value and store it in provided pointer value.
.periodSet	`g_timer0.p_api->periodSet(g_timer0.p_ctrl, period, unit)` Set the time until the timer expires.
.dutyCycleSet	`g_timer0.p_api->dutyCycleSet(g_timer0.p_ctrl, period, unit, pin)` Sets the time until the duty cycle expires.
.infoGet	`g_timer0.p_api->infoGet(g_timer0.p_ctrl, &info)` Get the time until the timer expires in clock counts and store it in provided pointer info.
.close	`g_timer0.p_api->close(g_timer0.p_ctrl)` Allows driver to be reconfigured and may reduce power consumption.
.versionGet	`g_timer0.p_api->versionGet(&version)` Get version and store it in provided pointer version.

Note: For details on operation and definitions for the function data structures, typedefs, defines, API data, API structures and function variables, review the *SSP User's Manual* **API References** for the associated module.

Table 2 Status Return Values

Name	Description
SSP_SUCCESS	Operation is successful.
SSP_ERR_ASSERTION	Parameter is NULL or configuration setting is not allowed.
SSP_ERR_IN_USE	The channel specified has already been opened.
SSP_ERROR_NOT_OPEN	The channel is not open.
SSP_ERR_INVALID_ARGUMENT	Invalid argument provided.

Note: Lower-level drivers may return common error codes. Refer to the *SSP User's Manual* **API References** for the associated module for a definition of all relevant status return values.

3. GPT HAL Module Operational Overview

The GPT HAL module configures a timer to a user-specified period. When the period elapses, the CPU can be interrupted, a port pin can be toggled, a transfer of data using the DMAC or DTC can be initiated, or another peripheral can be triggered to begin operation.

The following figure shows a flowchart for toggling a port pin or generating a CPU interrupt after a specified period:

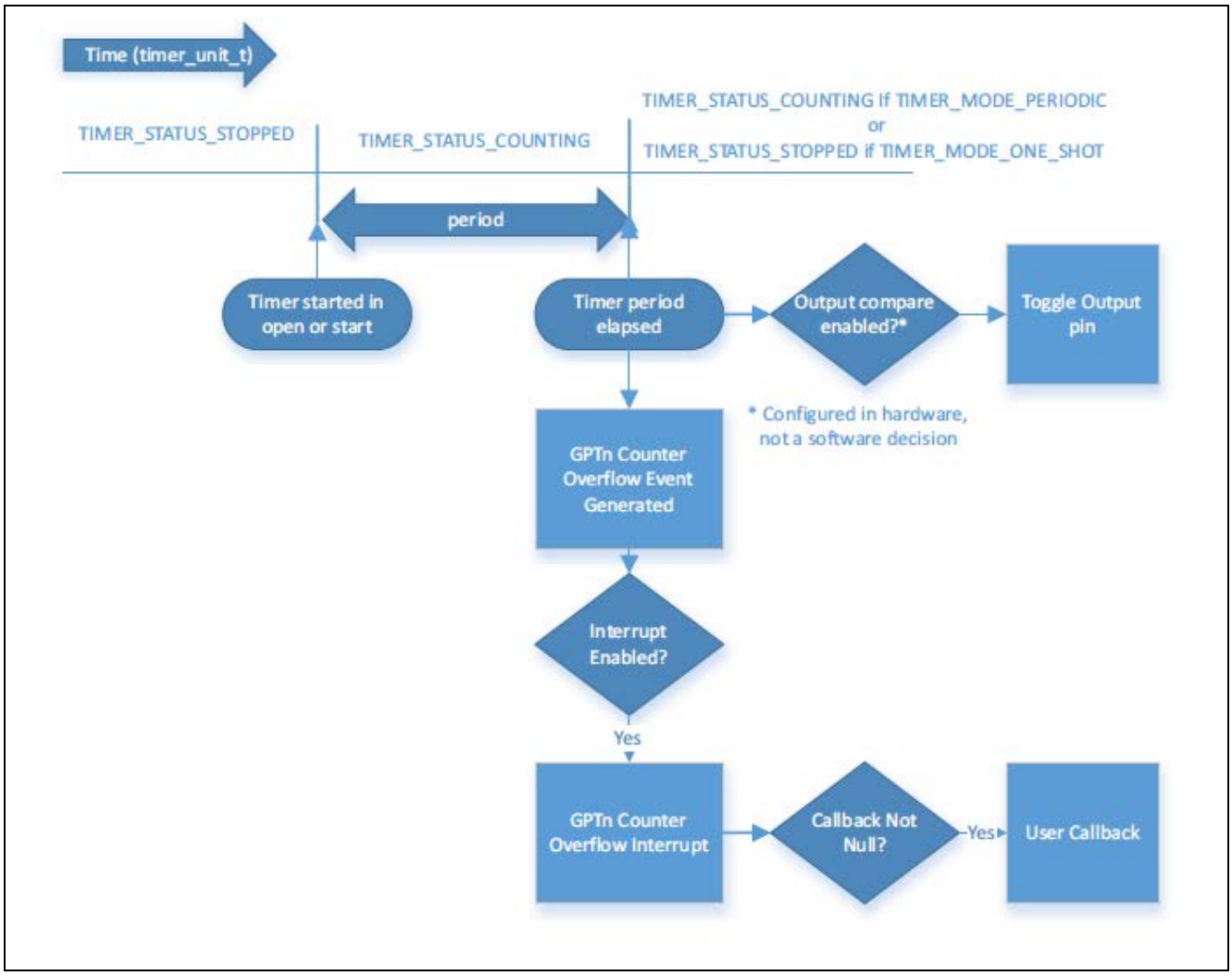

Figure 2 GPT Timer-Periodic or One-Shot Mode

Two different timer modules, the GPT and the AGT, are supported in the SSP. The following sections provide information on both modules so that the developer can compare and contrast the capabilities of each module for a particular application. For additional information on the AGT, refer to the AGT User's Guide.

The GPT module is recommended for most generic timer applications, but either module can be used for a basic timer functionality. The following use cases describe why one timer module would be preferred over the other.

Selecting the GPT Timer Module

The GPT module uses a high-resolution 32-bit counter that can only be clocked by PCLKD. There are more GPT channels than AGT channels on Synergy devices, so using GPT is less likely to cause a resource conflict.

Selecting the AGT Timer Module

The AGT module uses a 16-bit counter that can be clocked by PCLKB, LOCO, or Fsub. If clocked by LOCO or Fsub, the AGT interrupt can be used to wake the MCU from sleep modes. There are two channels, and channel 1 can be clocked by channel 0 underflow, effectively creating a 32-bit cascaded timer.

3.1 GPT HAL Module Important Operational Notes and Limitations

3.1.1 GPT HAL Module Operational Notes

The maximum time period depends on the timer type and the input clock frequency.

- On a GPT with 32-bit resolution with PCLKD running at 120 MHz, the maximum period is approximately 36650 seconds, which is just over 10 hours (AGT Count Clock is PCLKD/1024).
- On a GPT with 16-bit resolution with PCLKD running at 32 MHz, the maximum period is approximately 2.09 seconds (AGT Count Clock is PCLKD/1024).
- On an AGT with 16-bit resolution with PCLKB running at 60 MHz, the maximum period is approximately 8.7 ms (AGT Count Clock is PCLKB/8).
- On an AGT with 16-bit resolution with Fsub running at 32 kHz, the maximum period is approximately 2.0 seconds (AGT Count Clock is AGTSCLK/128).

The AGT counter underflow interrupt for the selected channel used must be enabled in the board support package (BSP) in the following situations:

1. To get a software interrupt when the timer period has elapsed.
2. To use one-shot mode

When the AGTn AGTI interrupt is enabled in the BSP, the corresponding ISR is defined in the timer driver. The ISR calls a user-callback function if one was registered in open.

Note: Interrupts may be skipped when used with the DTC peripheral with IRQ set to TRANSFER_IRQ_END.

GPT Output Timer Signal

If the timer output is configured (GTIOCA/B Output Enabled set to True), the output pin will start at the GTIOCA/B Stop Level and toggle every time the period elapses, beginning with the first time the period elapses after the timer is started.

In one-shot mode, the output is also configured to toggle when the timer starts counting. This generates a pulse - the timer toggles from the stop level when counting begins and toggles back to the stop level when counting ends.

Timer Period Calculation

The timer period is defined as the time until the timer expires. When output compare is used, the output pin will toggle once per period, so the traditional period (from rising edge to rising edge) is twice the period specified in the software.

Runtime period calculation based on the current clock settings is available from `open` and `periodSet`.

If the specified timer period is different than the raw counts, the period is calculated using the current timer clock frequency (see Configuring the GPT Clocks or Configuring the AGT Clocks). The current timer clock frequency is determined using s`ystemClockFreqGet`. This frequency will be used in the appropriate formula from the following table as clk_freq_hz.

Table 3 Timer period calculation

Timer Units	Formula
TIMER_UNIT_PERIOD_NSEC	Counts = (period * clk_freq_hz) / 1000000000
TIMER_UNIT_PERIOD_USEC	Counts = (period * clk_freq_hz) / 1000000
TIMER_UNIT_PERIOD_MSEC	Counts = (period * clk_freq_hz) / 1000
TIMER_UNIT_PERIOD_SEC	Counts = (period * clk_freq_hz)
TIMER_UNIT_FREQUENCY_HZ	Counts = (clk_freq_hz) / period
TIMER_UNIT_FREQUENCY_KHZ	Counts = (clk_freq_hz) / 1000 * period

If the requested period is larger than the counter size (32-bit or 16-bit), the driver selects the smallest divisor that allows the result to fit in the counter size. If the counter value is larger than the counter size with the largest divisor (1024), an error code (SSP_ERR_INVALID_ARGUMENT) is returned.

Triggering DMAC/DTC with GPT

To trigger a transfer of data using the DMAC or DTC peripheral when the timer period elapses, configure the DMAC/DTC transfer with activation_source set to ELC_EVENT_GPTn_COUNTER_OVERFLOW (where n is the GPT channel number). For details, see DMAC or DTC module guides.

Note: If you use the timer in a one-shot mode with the DTC, the entire transfer completes before the interrupt stops the timer if the IRQ is set to TRANSFER_IRQ_END. To generate only one transfer after the timer period elapses, set IRQ to TRANSFER_IRQ_EACH, or use the DMAC for the transfer.

Triggering ELC Events with GPT

The GPT timer can trigger the start of other peripherals. The ELC guide provides a list of all available peripherals.

Triggering DMAC/DTC with AGT

To trigger a transfer of data using the DMAC or DTC peripheral when the timer period elapses, configure the DMAC/DTC transfer with activation_source set to ELC_EVENT_AGTn_AGTI (where n is the AGT channel number). See Transfer Interface for further information.

Note: If you use the timer in one-shot mode with the DTC, the entire transfer completes before the interrupt stops the timer if IRQ is set to TRANSFER_IRQ_END. To generate only one transfer after the timer period elapses, set IRQ to TRANSFER_IRQ_EACH or use the DMAC for the transfer.

Triggering ELC Events with AGT

The AGT timer can trigger the start of other peripherals. The ELC guide provides a list of all available peripherals listed in elc_peripheral_t. See events and peripheral definitions for further information.

Cascaded AGT Timers creating a 32-bit timer

AGT Channel 1 can be clocked by the AGT Channel 0 underflow, creating a cascaded 32-bit timer.

3.1.2 GPT HAL Module Limitations

For GPT Power Down, the GPT module does not set the Module Stop bit (MSTP) for GPT in the `close`() API. This is intentional because the GPT module stop bits control the power to multiple GPT channels, and the GPT module cannot know if other GPT modules are used in the application. Use the following procedure to set the module stop bit for GPT to reduce power consumption when no GPT channels are in use:

1. In the hardware manual for your MCU, go to the Low Power Modes chapter and look in the Register Descriptions chapter for Module Stop Control Register D.
2. Look at MSTPD5 and MSTPD6. Identify which of these bits contains the channel used by your application.
3. Add the LPM driver to your project on the **Threads** tab of the Synergy Configuration tool:
   New> Driver > Power > Low Power Modes Driver on `r_lpm`.
4. In the application code, confirm that no other GPT channels in the MSTP bit identified in step 2 are currently used by the application. If no channels are being used, use the LPM API to power down the set of GPT channels:
   A. To power down GPT channels controlled by MSTPD5 with an LPM module named `g_lpm0 call`:
   `g_lpm0.p_api`->moduleStop(LPM_MSTP_GPT_CH7_0); // Ignore the channel numbers in the enum value – this is for MSTPD5
   B. To power down GPT channels controlled by MSTPD6, call:
   `g_lpm0.p_api`->moduleStop(LPM_MSTP_GPT_CH13_8); // Ignore the channel numbers in the enum value – this is for MSTPD6

Refer to the most recent SSP Release Notes for any additional operational limitations for this module.

4. Including the GPT HAL Module in an Application

This section describes how to include the GPT HAL module in an application using the SSP configurator.

Note: This section assumes that you are familiar with creating a project, adding threads, adding a stack to a thread and configuring a block within the stack.

To add the GPT Timer driver to an application, simply add it to a thread using the stacks selection sequence given in the following table. (The default name for the GPT Timer driver is `r_gpt`. This name can be changed in the associated Properties window.)

Table 4 GPT Selection Sequence

Resource	ISDE Tab	Stacks Selection Sequence
g_timer0 Timer Driver on r_gpt	Threads -> HAL/Common	New Stack> Driver> Timers> Timer Driver on r_gpt

When the GPT HAL module is added to the thread stack as shown in the following figure, the configurator automatically adds any needed lower level modules. Modules with a Gray band are individual modules that stand alone.

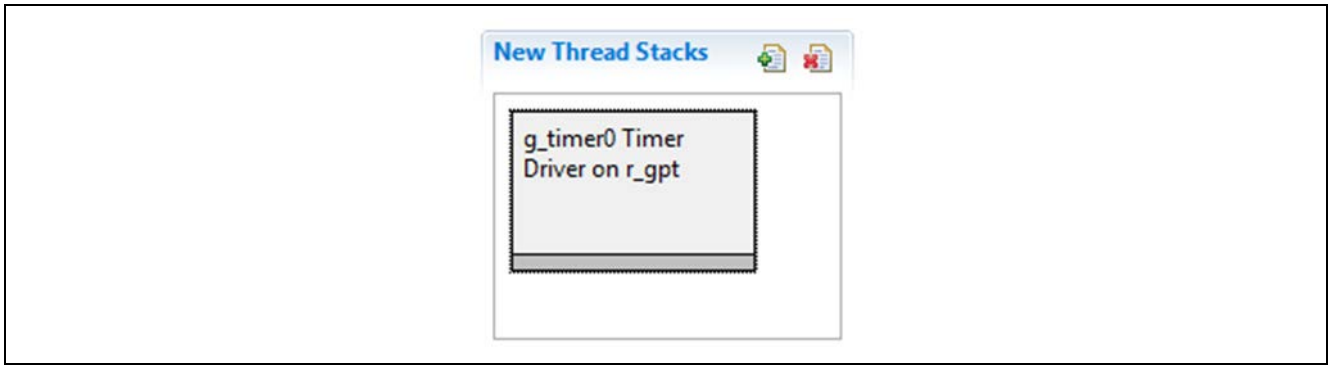

Figure 3 GPT HAL Module Stack

5. Configuring the GPT HAL Module

The user configures the GPT HAL module for the desired operation. The SSP configuration window automatically identifies (by highlighting the block in red) any required configuration selections, such as interrupts or operating modes, which must be configured for lower-level modules for successful operation. Furthermore, only those properties that can be changed without causing conflicts are available for modification. Other properties are 'locked' and are not available for changes, and are identified with a lock icon for the 'locked' property in the Properties window in the ISDE. This approach simplifies the configuration process and makes it much less error-prone than previous 'manual' approaches to configuration. The available configuration settings and defaults for all the user-accessible properties are given in the Properties tab within the SSP configurator and are shown in the following tables for easy reference.

One of the properties most often identified as requiring a change is the interrupt priority; this configuration setting is available within the Properties window of the associated module. Simply select the indicated module and then view the Properties window; the interrupt settings are often toward the bottom of the properties list, so scroll down until they become available. Also note that the interrupt priorities listed in the Properties window in the ISDE indicates the validity of the setting based on the targeted MCU (CM4 or CM0+). This level of detail is not included in the following configuration properties tables, but is easily visible with the ISDE when configuring interrupt-priority levels.

Note: You may want to open your ISDE, create the module, and explore the property settings in parallel with looking over the following configuration table settings. These property settings help to orient you and can be a useful 'hands-on' approach to learning the ins and outs of developing with SSP.

Table 5 Configuration Settings for the GPT HAL Module on r_gpt

ISDE Property	Value	Description
Parameter Checking	BSP, Enabled, Disabled (Default: BSP)	Parameter selection.
Name	g_timer0	Module name.
Channel	0	Physical hardware channel. 0-13 for S7G2, 0-9 for S3A7, 0-6 for S124
Mode	Periodic, One Shot, PWM (Default: Periodic)	Mode selection. Note: One Shot functionality is not available in the GPT hardware, so it is implemented in software by stopping the timer in the ISR called when the period expires. For this reason, ISR's must be enabled for one-shot mode even if the callback is unused.
Period Value	10	Period value selection. See the Timer Period Calculation section earlier in this document.
Period Unit	Raw Counts, Nanoseconds, Microseconds, Milliseconds, Seconds, Hertz, Kilohertz (Default:	Period unit selection. See the Timer Period Calculation section earlier in this document.

ISDE Property	Value	Description
	Milliseconds)	
Duty Cycle Value	50	Duty cycle value selection.
Duty Cycle Unit	Unit Raw Counts, Unit Percent, Unit Percent x 1000 (Default: Unit Raw Counts)	Duty cycle unit selection.
Auto Start	True, FALSE (Default: True)	Auto start selection. Set to true to start the timer after configuring or false to leave the timer stopped until start is called.
GTIOCA Output Enabled	True, False (Default: False)	GTIOCA output enabled selection. Set to true to output the timer signal on a port pin configured for GPT. Set to false for no output of the timer signal.
GTIOCA Stop Level	Pin Level Low, Pin Level High, Pin Level Retained (Default: Pin Level Low)	Controls output pin level when the timer is stopped.
GTIOCB Output Enabled	True, False (Default: False)	GTIOCB output enabled selection.
GTIOCB Stop Level	Pin Level Low, Pin Level High, Pin Level Retained (Default: Pin Level Low)	GTIOCB stop level selection.
Callback	NULL	Callback selection. A user callback function can be registered in open. If this callback function is provided, it will be called from the interrupt service routine (ISR) each time the timer period elapses.
Interrupt Priority	Priority 0(highest)-15(lowest), Disabled (Default: Disabled)	Interrupt priority selection.

Note: The example values and defaults are for a project using the Synergy S7G2 MCU. Other MCUs may have different default values and available configuration settings.

In some cases, settings other than the defaults for a module can be desirable. For example, it might be useful to select a different clock source than the default. The configurable properties for the lower-level stack modules are provided in the following sections for completeness and as a reference.

5.1 GPT HAL Module Clock Configuration

The GPT timer is clocked based on the PCLKD frequency. You can set the PCLKD frequency using the clock configurator in the ISDE Configuring Clocks tab or the CGC Interface at run-time.

5.2 GPT HAL Module Pin Configuration

The GPT peripheral module uses pins on the MCU to communicate to external devices. I/O pins must be selected and configured as required by the external device. The first table lists the method used to select pins within the SSP configuration window and the following table lists an example selection for the associated pins.

Note: The operation mode selection determines what peripheral signals are available and what MCU pins are required.

Table 6: Pin Selection Sequence for GPT HAL Driver

Resource	ISDE Tab	Pin selection Sequence
GPT	Pins	Select Peripherals > Timer: GPT>GPT0

Note: The selection sequence assumes GPT0 is the desired hardware target for the driver.

Table 7: Pin Configuration Settings for GPT HAL Driver

Property	Value	Description
Pin Group Selection	Mixed, _A Only, _B Only (Default: Mixed)	Select pin group mapping.
Operation Mode	Disabled, GTIOCA or GTIOCB, GTIOCA and GTIOCB (Default: Disabled)	Select timer operation mode.
GTIOCA:	None, P300, P512 (Default: P512)	GTIOCA Pin.
GTIOCB:	None, P108, P511 (Default: P511)	GTIOCB Pin.

Note: The example values are for a project using the Synergy S7G2 MCU and the SK-S7G2 Kit. Other Synergy MCUs and other Synergy Kits may have different available pin configuration settings.

6. Using the GPT Module in an Application

The typical steps in using the GPT HAL module in an application are:

1. Initialize the GPT HAL module using the `open` API.
2. Start the GPT HAL module by calling the `start` API.
3. Respond to the timer callback as needed (user code).

Note: The GPT period and duty cycle parameters can be reconfigured based on the application needs.

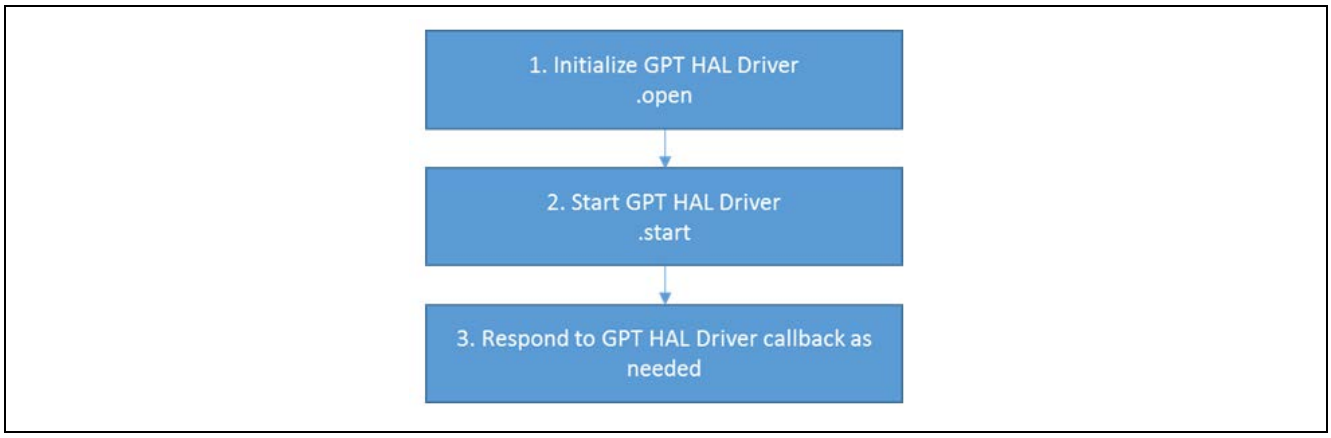

Figure 4 Typical GPT HAL Module Application Flow Chart

7. The GPT HAL Module Application Project

The application project associated with this module guide demonstrates the aforementioned steps in a full design. You may want to import and open the application project within the ISDE and view the configuration settings for the GPT HAL module. You can also read over the code (in <GPT_HAL_MG.c>) which is used to illustrate the GPT HAL module APIs in a complete design.

The application project demonstrates the typical use of the GPT HAL APIs using three GPT modules. The application project main thread entry initializes the timers, starts and reconfigures the period and duty cycle parameters. When GPT 0 and GPT 2 expire, their respective callbacks toggle an LED. GPT 1 is configured to be a PWM timer. Information about the timers will be displayed on the Debug Console using the common semi-hosting function. The following table identifies the target versions for the associated software and hardware used by the application project:

Table 8 Software and Hardware Resources Used by the Application Project

Resource	Revision	Description
e^2 studio	5.3.1 or later	Integrated Solution Development Environment
SSP	1.2.0 or later	Synergy Software Platform
IAR EW for Synergy	7.71.2 or later	IAR Embedded Workbench® for Renesas Synergy™
SSC	5.3.1 or later	Synergy Standalone Configurator
SK-S7G2	v3.0 to v3.1	Starter Kit

A simple flow diagram of the application project is given in the following figure:

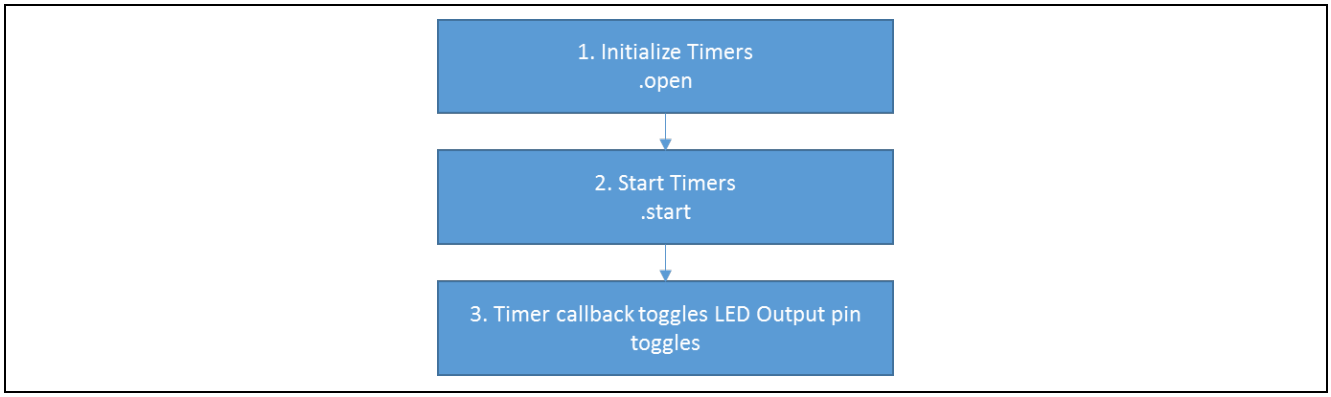

Figure 5 GPT HAL Module Application Project Flow Diagram

Three timers will be used in this project:

1. `g_timer_gpt_0` is set up as a periodic GPT.
2. `g_timer_gpt_1` is set up as a PWM GPT.
3. `g_timer_gpt _2` is set up as a one-shot GPT.

The following tables list the configuration properties for each of the timers used. Other settings keep their default values.

Table 9 Configuration Properties for g_timer_gpt_0

ISDE Property	Value Set
Name	g_timer_gpt_0
Channel	0
Mode	Periodic
Period Value	1
Period Unit	Seconds
Auto Start	False
Callback	timer0_callback
Interrupt Priority	Priority 2

Table 10 Configuration Properties for g_timer_gpt_1

ISDE Property	Value Set
Name	g_timer_gpt_1
Channel	1
Mode	PWM
Period Value	1
Period Unit	Seconds
Auto Start	False
GTIOCB Output Enabled	True
Interrupt Priority	Priority 5

Table 11 Configuration Properties for g_timer_gpt_2

ISDE Property	Value Set
Name	g_timer_gpt_2
Channel	2
Mode	One Shot
Period Value	10
Period Unit	Seconds
Auto Start	False
Callback	timer_2_callback
Interrupt Priority	Priority 5

In the **Pin** tab window, configure the following pins. The pin to be configured can be scoped to view the timer PWM output for `g_timer_gpt_1`:

Peripherals > Timer:GPT > GPT1.

- Pin Group Selection > Mixed
- Operation Mode: GTIOCA or GTIOCB.
- GTIOCB > P406. Expanding on P406, which will then navigate the configurator to the P406 configuration page, you should see that GPT1_GTIOCB has been set as the Chip input/output.

In a custom usage of the GPT module, P101 of pin group A can be used instead for the output of the GPT. GTIOCA can also be selected instead of GTIOCB. The user can select from options for pin group and operation mode according to the pins desired for the output.

`GPT_HAL_MG.c` file is located in the project once it has been imported into the ISDE. You can open this file within the ISDE and follow along with the description provided to help identify key uses of the APIs.

The first section of `GPT_HAL_MG.c` contains the prototypes for the functions used to configure the timers. `Hal_entry.c` calls the `init_timers()` function that opens, starts and configures the timers. The timers are opened with the `open_timers()` function. This function calls the `open` API for each timer. The `start_timers()` function then calls the `start` API for each timer.

The next section of the code configures the periods and duty cycle of the periods. This section is not needed if you do not wish to change these parameters at run time. The period is changed using the `set_timer_period` function, which calls the `periodSet` API. The duty cycle is changed using the `set_timer_pwm`, which calls the `dutyCycleSet` API. The code snippet does the following:

1. Change timer period of `g_timer_gpt_0` to 2 seconds. The on-board LED1 toggle every 2 seconds.
2. Change duty cycle of `g_timer_gpt_1` to 30%. The PWM output can be viewed on a scope by probing P406.
3. Counter value of all the timers are obtained and printed on the debug console. `get_timer_value` function does this by calling the `counterGet` API.
4. Information of all the timers are obtained and printed on the debug console. `get_timer_info` function does by calling the `infoGet` API.

The last section is the user callback function. Timer0_callback is the callback function defined for when `g_timer_gpt_0` expires. It is called whenever `g_timer_gpt_0` expires, and toggles LED0. The timer2_callback is the callback function defined for when `g_timer_gpt_2` expires. It is called when `g_timer_gpt_2` expires, and toggles LED1.

Expected output of the application project

1. LED0 toggles every 2 seconds. This is based on the period of `g_timer_gpt_0`.
2. LED1 toggles once since it is set by the callback function for `g_timer_gpt_2`, which is a one-shot timer. You may start `g_timer_gpt_2` again to toggle LED1 one more time.
3. A 30% PWM output on P406.

Note: It is assumed that you are familiar using printf() with the Debug Console in the SSP. If you are unfamiliar with printf(), refer to “How do I Use Printf() with the Debug Console in the Synergy Software Package,” which is

available as described in the References section at the end of this document. Alternatively, you can see results via the watch variables in the debug mode.

8. Customizing the GPT HAL Module for a Target Application

Some configuration settings will normally be changed by the developer from those shown in the application project. For example, the user can easily change the configuration settings for the GPT clock by updating the PCLKD in the Clock tab. The user can also change the output port pins of the GPT. This can be done by using the Pins tab in the configurator. In the Pins tab, the user can select from options for the Peripheral:GPT pin group and operation mode according to the pins desired for the output.

The application project also provides functions to reset, stop and close the timer. These functions can be used as per the user's application.

9. Running the GPT HAL Module Application Project

To run the GPT HAL module application project and to see it executed on a target kit, you can simply import it into your ISDE, compile and run debug.

To implement the GPT HAL module application in a new project, use the following steps for defining, configuring, auto-generating files, adding code, compiling and debugging on the target kit. This hands-on approach can help make the development process with SSP more practical, where just reading over this guide will tend to be more theoretical.

Note: The following steps are described in sufficient detail for someone experienced with the basic flow through the Synergy development process. If these steps are not familiar, refer to the first few chapters of the *SSP User's Manual* available as described in the References section at the end of this document.

To create and run the GPT HAL module application project, simply follow these steps:

1. Create a new Renesas Synergy project for the <SK-S7G2> called <GPT_HAL_MG_AP>
2. Add to <HAL/Common> the Timer Drivers
3. Click on the "Generate Project Content" button.
4. Add the code from the supplied project file `<GPT_HAL_MG>.c`, `<GPT_HAL_MG>.h`, and `<hal_entry>.c` or copy over the files.
5. Connect to the host PC via a micro USB cable to J19 on SK-S7G2
6. Start to debug the application
7. The output can be viewed in the Renesas Debug Virtual Console.

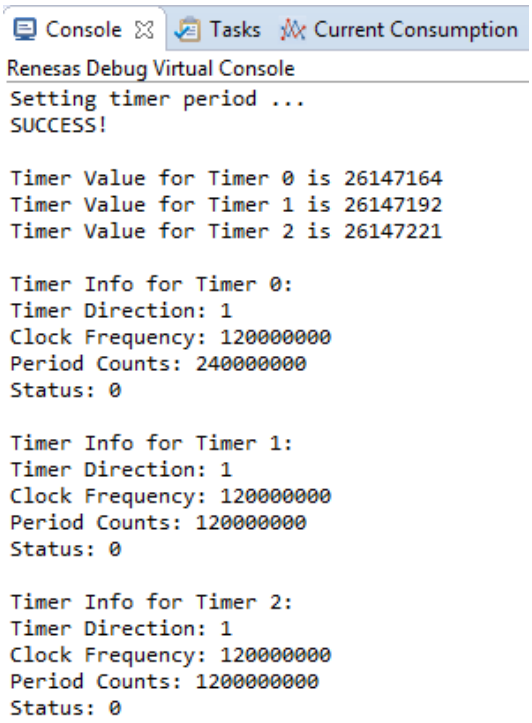

Figure 6 Example Output from Timer Drivers Application Project

In addition, the PWM GPT output can be viewed on an oscilloscope. The PWM GPT is output on P406.

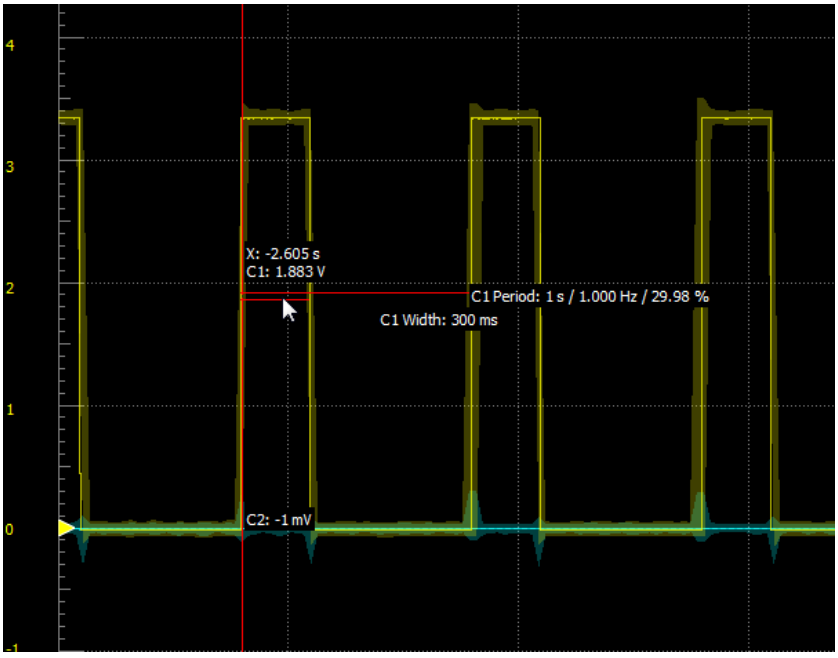

Figure 7 g_timer_gpt_1 set to PWM mode with a 1s period and 30% duty cycle

10. GPT HAL Module Conclusion

This module guide has provided all the background information needed to select, add, configure and use the module in an example project. Many of these steps were time consuming and error-prone activities in previous generations of embedded systems. The Renesas Synergy Platform makes these steps much less time consuming and removes the common errors like conflicting configuration settings or the incorrect selection of lower-level modules. The use of high-level APIs (as demonstrated in the application project) illustrates additional development-time savings by allowing work to begin at a high level and avoiding the time required in older development environments to use, or, in some cases, create, lower-level drivers.

11. GPT HAL Module Next Steps

After you have mastered a simple GPT HAL module project, you may want to review a more complex example. Perhaps you may consider integrating a GPT HAL module into other module guides made available by Renesas.

Based on your timing needs, you may find that changing the frequency of the clock source, PCLKD, may be necessary.

12. GPT HAL Module Reference Information

SSP User Manual: Available in html format in the SSP distribution package and as a pdf from the Synergy Gallery.

Links to all the most up-to-date r_gpt module reference materials and resources are available on the Synergy Knowledge Base: https://en-us.knowledgebase.renesas.com/English_Content/Renesas_Synergy%E2%84%A2_Platform/Renesas_Synergy_Knowledge_Base/r_gpt_Module_Guide_Resources.

Website and Support

Support: https://synergygallery.renesas.com/support

Technical Contact Details:

- America: https://renesas.zendesk.com/anonymous_requests/new
- Europe: https://www.renesas.com/en-eu/support/contact.html
- Japan: https://www.renesas.com/ja-jp/support/contact.html

All trademarks and registered trademarks are the property of their respective owners.

Revision History

		Description	
Rev.	**Date**	**Page**	**Summary**
1.00	May 22, 2017	—	Initial Release
1.01	Aug 10, 2017	10	Update to Hardware and Software Resources Table

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other disputes involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawing, chart, program, algorithm, application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics products.
5. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The intended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below.

   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots etc.

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Renesas Electronics products are neither intended nor authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems, surgical implantations etc.), or may cause serious property damages (space and undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for which the product is not intended by Renesas Electronics.
6. When using the Renesas Electronics products, refer to the latest product information (data sheets, user's manuals, application notes, "General Notes for Handling and Using Semiconductor Devices" in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat radiation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions or failure or accident arising out of the use of Renesas Electronics products beyond such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please ensure to implement safety measures to guard them against the possibility of bodily injury, injury or damage caused by fire, and social damage in the event of failure or malfunction of Renesas Electronics products, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures by your own responsibility as warranty for your products/system. Because the evaluation of microcomputer software alone is very difficult and not practical, please evaluate the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please investigate applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive carefully and sufficiently and use Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall not use Renesas Electronics products or technologies for (1) any purpose relating to the development, design, manufacture, use, stockpiling, etc., of weapons of mass destruction, such as nuclear weapons, chemical weapons, or biological weapons, or missiles (including unmanned aerial vehicles (UAVs)) for delivering such weapons, (2) any purpose relating to the development, design, manufacture, or use of conventional weapons, or (3) any other purpose of disturbing international peace and security, and you shall not sell, export, lease, transfer, or release Renesas Electronics products or technologies to any third party whether directly or indirectly with knowledge or reason to know that the third party or any other party will engage in the activities described above. When exporting, selling, transferring, etc., Renesas Electronics products or technologies, you shall comply with any applicable export control laws and regulations promulgated and administered by the governments of the countries asserting jurisdiction over the parties or transactions.
10. Please acknowledge and agree that you shall bear all the losses and damages which are incurred from the misuse or violation of the terms and conditions described in this document, including this notice, and hold Renesas Electronics harmless, if such misuse or violation results from your resale or making Renesas Electronics products available any third party.
11. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

(Rev.3.0-1 November 2016)

RENESAS

SALES OFFICES

Renesas Electronics Corporation

http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
2801 Scott Boulevard Santa Clara, CA 95050-2549, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
Room 1709, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100191, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, P. R. China 200333
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre, Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 1207, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics India Pvt. Ltd.
No.777C, 100 Feet Road, HAL II Stage, Indiranagar, Bangalore, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.
12F., 234 Teheran-ro, Gangnam-Gu, Seoul, 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

© 2017 Renesas Electronics Corporation. All rights reserved.

Colophon 6.0