Renesas Synergy™ Platform

I2C SCI HAL Module Guide

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 an 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 I²C SCI HAL Module Reference Information section), and should be valuable resources for creating more complex designs.

The I²C SCI HAL module is a high-level API for I²C Master applications with the module implemented on `r_sci_i2c`. The I²C SCI HAL module uses the SCI peripheral on the Synergy MCU device. Callbacks are provided for transmit complete and receive complete.

Contents

1. I²C SCI HAL Module Features

The I²C SCI HAL module provides support for the following features:

- Support for I²C SCI operations
- Supports the following operations with a slave I²C SCI device
  - Read
  - Write
  - Reset
- Callback support
  - Transfer aborted
  - Transmit complete (number of bytes transmitted provided)
  - Receive complete (number of bytes received provided)

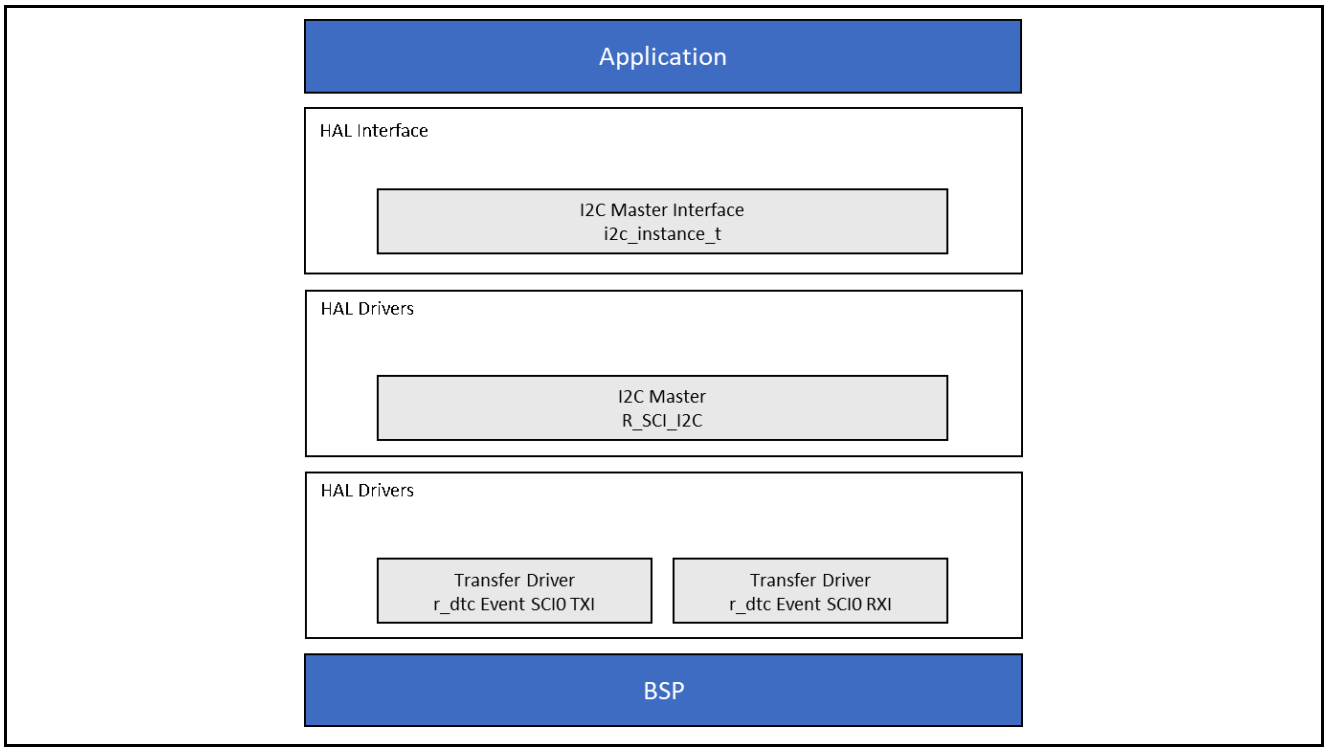

Figure 1. I²C SCI HAL Module Block Diagram

2. I²C SCI HAL Module APIs Overview

The I²C SCI HAL module defines APIs for reading and writing using a master I²C device. 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. I²C SCI HAL Module API Summary

Function Name	Example API Call and Description
.open	`g_i2c.p_api->open(g_i2c.p_ctrl, g_i2c.p_cfg);` Open the instance and initialize the hardware.
.close	`g_i2c.p_api->close(g_i2c.p_ctrl);` Closes the driver and releases the I2C device.
.read	`g_i2c.p_api->read(g_i2c.p_ctrl, &destination, bytes, restart);` Performs a read operation on an I2C device.
.write	`g_i2c.p_api->write(g_i2c.p_ctrl, &destination, bytes, restart);` Performs a write operation on an I2C device.
.reset	`g_i2c.p_api->reset(g_i2c.p_ctrl);` Reset the peripheral.
.versionGet	`g_i2c.p_api->versionGet(&version);` Retrieve the API version with the version pointer.

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	API Call Successful
SSP_ERR_IN_USE	Attempted to open an already open device instance
SSP_ERR_ABORTED	Device was closed while a transfer was in progress
SSP_ERR_INVALID_RATE	The requested rate cannot be set
SSP_ERR_ASSERTION	The parameter `p_ctrl` is NULL
SSP_ERR_NOT_OPEN	Device was not even opened
SSP_ERR_IRQ_BSP_DISABLED	Event information could not be found

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. I²C SCI HAL Module Operational Overview

The I²C SCI HAL module supports transactions with an I²C slave device. Callbacks are provided to interrupt the CPU when a transmission has been completed or aborted or a receive has been completed. The I²C SCI HAL module invokes the callback with the argument `i2c_callback_args_t`, indicating the number of received or transmitted bytes in the buffer, pointer to user provided context and the event `i2c_event_t`.

3.1 I²C SCI HAL Module Important Operational Notes and Limitations

3.1.1 I²C SCI HAL Module Operational Notes

Interrupts

- The I²C interrupts (SCI Error (EEI), Receive Buffer Full (RXI), Transmit Buffer Empty (TXI), and Transmit End (TEI)) for the selected channel must be enabled in the board support package (BSP), without consideration for whether the user wants to use callbacks.
- Setting the interrupts to different priority levels could result in improper operation.

IIC Rate Calculation

- The I²C SCI HAL module calculates the internal baud-rate setting based on the configured transfer rate and passes this to open. The closest possible baud rate that can be achieved (less than or equal to the requested rate) at the current PCLKB settings is calculated and used.
- If a valid clock rate could not be calculated, an error is returned.

Triggering DMAC/DTC with the IIC

- DTC transfer support is added by default in the configurator, and this can be removed for CPU transfer cases. The DTC is configured in the module. No user configuration is required for this.
- DMA transfer is not supported.

Triggering ELC Events with the IIC

- The I²C SCI HAL module can trigger the start of other peripherals. See the *ELC User Guide* for further information.

Multiple Devices on the Bus

- The `slaveAddressSet()` API can be used to switch the slave devices without reconfiguring (no need to close and open) the bus in an application for using multiple slave devices on the same bus.
- Control instance and bus configuration remains the same, but the slave address and addressing changes.
- Other ways of communication with different slave devices can be used by closing the first device and opening a new device. This is recommended if slaves are having different configurations.
- Applications using multiple devices connected on the same channel need to define the following macro in the pre-processor settings of your project (or the project may not build correctly):
  `SSP_SUPPRESS_ISR_<device_name>`
  Where <device_name> is the name of the additional device connected to the same channel.

3.1.2 I²C SCI HAL Module Limitations

The I²C SCI HAL module in IRQ mode may not work with certain slave devices. You need to enable DTC transfer mode to work with such devices.

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

4. Including the I²C SCI HAL Module in an Application

This section describes how to include the I²C SCI HAL module in an application using the SSP configurator.

Note: It is assumed that you are familiar with creating a project, adding threads, adding a stack to a thread and configuring a block within the stack. If you are unfamiliar with any of these steps, refer to the first few chapters of the SSP User's Manual to learn how to manage these important steps in creating SSP-based applications.

To add the I²C SCI Driver to an application, simply add it to a thread using the stacks selection sequence in the following table. (The default name for the I²C SCI HAL Driver is `g_i2c0`. This name can be changed in the associated Properties window.)

Table 3. I²C SCI HAL Module Selection Sequence

Resource	ISDE Tab	Stacks Selection Sequence
g_i2c0 I²C Master Driver on r_sci_i2c	Threads	New Stack> Driver> Communications> I2C Master Driver on r_sci_i2c

When the I²C SCI HAL Module on `r_sci_i2c` is added to the thread stack as shown in the following figure, the configurator automatically adds any needed lower-level drivers. Modules with a Gray band are individual modules that stand alone.

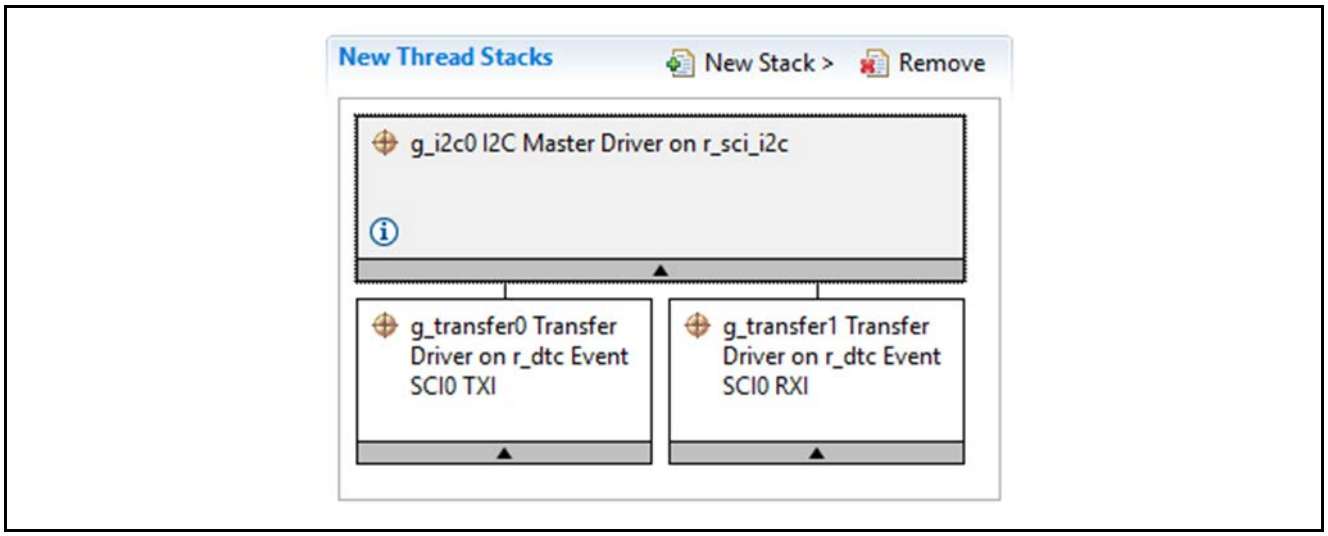

Figure 2. I²C SCI HAL Module Stack

5. Configuring the I²C SCI HAL Module

You must configure the I²C SCI 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. Only those properties that can be changed without causing conflicts are available for modification. Other properties are 'locked' and 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 table later in this document 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 includes an indication as to the validity of the setting based on the MCU targeted (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 and create the module and explore the property settings in parallel with looking over the following configuration table settings. This step helps to orient you and can be a useful hands-on approach to learning the ins and outs of developing with SSP.

Table 4. Configuration Settings for the I²C SCI HAL Module on r_sci_i2c

ISDE Property	Value	Description
Parameter Checking	Enabled, Disabled, BSP (Default: BSP)	Selects if code for parameter checking is to be included in the build
Name	Default: g_i2c0	Module name
Channel	0	Channel Number
Rate	Standard, Fast mode (Default: Standard)	Rate selection
Slave Address	Default: 0x00	Slave address
Address Mode	7-bit, 10-bit (Default: 7-bit)	Address mode
SDA Output Delay (nanoseconds)	300	SDA output delay value
Bit Rate Modulation Enable	Enabled, Disable (Default: Enable)	Bit rate modulation setting
Callback	Default: NULL	User defined callback
Receive Interrupt Priority	Priority 0 (highest)-15 (lowest), Disabled (Default: Priority 2)	Receive Interrupt priority
Transmit Interrupt Priority	Priority 0 (highest)-15 (lowest), Disabled (Default: Priority 2)	Transmit Interrupt priority
Transmit End Interrupt Priority	Priority 0 (highest)-15 (lowest), Disabled (Default: Priority 2)	Transmit end interrupt priority

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

In some cases, settings other than the defaults for stack modules can be desirable. For example, it might be useful to select different slave addresses or address modes. The configurable properties for the lower-level stack modules are given in the following sections for completeness and as a reference.

Note: Most property settings for modules are fairly intuitive and usually can be determined by inspection of the associated Properties window from the SSP configurator.

Table 5. Transfer Driver on r_dtc Event SCI0 TXI

ISDE Property	Value	Description
Parameter Checking	BSP, Enabled, Disabled (Default: BSP)	Selects if code for parameter checking is to be included in the build
Software Start	Enabled, Disabled (Default: Disabled)	Software start selection
Linker section to keep DTC vector table	`.ssp_dtc_vector_table`	Linker section selection
Name	g_transfer0	Module name
Mode	Normal	Mode selection
Transfer Size	1 Byte	Transfer size selection
Destination Address Mode	Fixed	Destination address mode selection
Source Address Mode	Incremented	Source address mode selection
Repeat Area (Unused in Normal Mode	Source	Repeat area selection
Interrupt Frequency	After all transfers have completed	Interrupt frequency selection
Destination Pointer	NULL	Destination pointer selection
Source Pointer	NULL	Source pointer selection
Number of Transfers	0	Number of transfers selection
Number of Blocks (Valid only in Block Mode)	0	Number of blocks selection
Activation Source (Must enable IRQ)	Event SCI0 TXI	Activation source selection
Auto Enable	FALSE	Auto enable selection
Callback (Only valid with Software start)	NULL	Callback selection
ELC Software Event Interrupt Priority	Priority 0(highest)-15(lowest), Disabled (Default: Disabled)	Interrupt priority for ELC Event

Note: The example values and defaults are for a project using the Synergy S7 MCU Group. Other MCUs may have different default values and available configuration settings.

Table 6. Transfer Driver on r_dtc Event SCI0 RXI

ISDE Property	Value	Description
Parameter Checking	BSP, Enabled, Disabled (Default: BSP)	Selects if code for parameter checking is to be included in the build
Software Start	Enabled, Disabled (Default: Disabled)	Software start selection
Linker section to keep DTC vector table	`.ssp_dtc_vector_table`	Linker section selection
Name	`g_transfer1`	Module name
Mode	Normal	Mode selection
Transfer Size	1 Byte	Transfer size selection
Destination Address Mode	Incremented	Destination address mode selection
Source Address Mode	Fixed	Source address mode selection
Repeat Area (Unused in Normal Mode	Destination	Repeat area selection
Interrupt Frequency	After all transfers have completed	Interrupt frequency selection
Destination Pointer	NULL	Destination pointer selection
Source Pointer	NULL	Source pointer selection
Number of Transfers	0	Number of transfers selection
Number of Blocks (Valid only in Block Mode)	0	Number of blocks selection
Activation Source (Must enable IRQ)	Event SCI0 RXI	Activation source selection
Auto Enable	FALSE	Auto enable selection
Callback (Only valid with Software start)	NULL	Callback selection
ELC Software Event Interrupt Priority	Priority 0 (highest) -15 (lowest), Disabled (Default: Disabled)	Interrupt priority for ELC Event

Note: The example values and defaults are for a project using the Synergy S7 MCU Group. Other MCUs may have different default values and available configuration settings.

5.1 I²C SCI HAL Module Clock Configuration

The SCI peripheral module uses PCLKB as its clock source. The actual I²C transfer rate is calculated and set internally by the driver (depending on the selected transfer rate). If the PCLKB is configured in such a manner that the selected internal rate cannot be achieved, an error is returned when initializing the driver.

5.2 I²C SCI HAL Module Pin Configuration

The SCI 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 following table illustrates the method for selecting the pins within the SSP configuration window and the subsequent table illustrates an example selection for the pins.

Note: For some peripherals, the operation mode selection determines what peripheral signals are available and what MCU pins are required for the I²C SCI HAL Module.

Table 7. Pin Selection Sequence for the I²C SCI HAL Module

Resource	ISDE Tab	Pin selection Sequence
SCI	Pins	Select Peripherals > Connectivity: SCI > SCI0

Note: The selection sequence assumes that SCI0 is the desired hardware target for the driver.

Table 8. Pin Configuration Settings for the I²C SCI HAL Module

Property	Value	Description
Pin Group Selection	_A only, _B only, Mixed (Default: _A only)	Pin group selection
Operation Mode	Enabled, Disabled (Default: Disabled)	Enable or disable peripheral module
SDA	None, P401, P407 (Default: None)	SDA Pin
SCL	None, P400, P204 (Default: None)	SCL Pin

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

6. Using the I²C SCI HAL Module in an Application

The typical steps in using the I²C SCI HAL module in an application are:

1. Initialize and open the I²C SCI HAL module using the `open` API.
2. Transfer data to the slave using the `write` API.
3. Receive data from the slave using the `read` API.
4. Operate on the received data as needed by the application.
5. Reset the instance with the `reset` API (if needed).
6. Perform transactions with slave device (if needed).
7. Close the channel using the `close` API.

The following figure illustrates these common steps within a typical operational flow diagram.

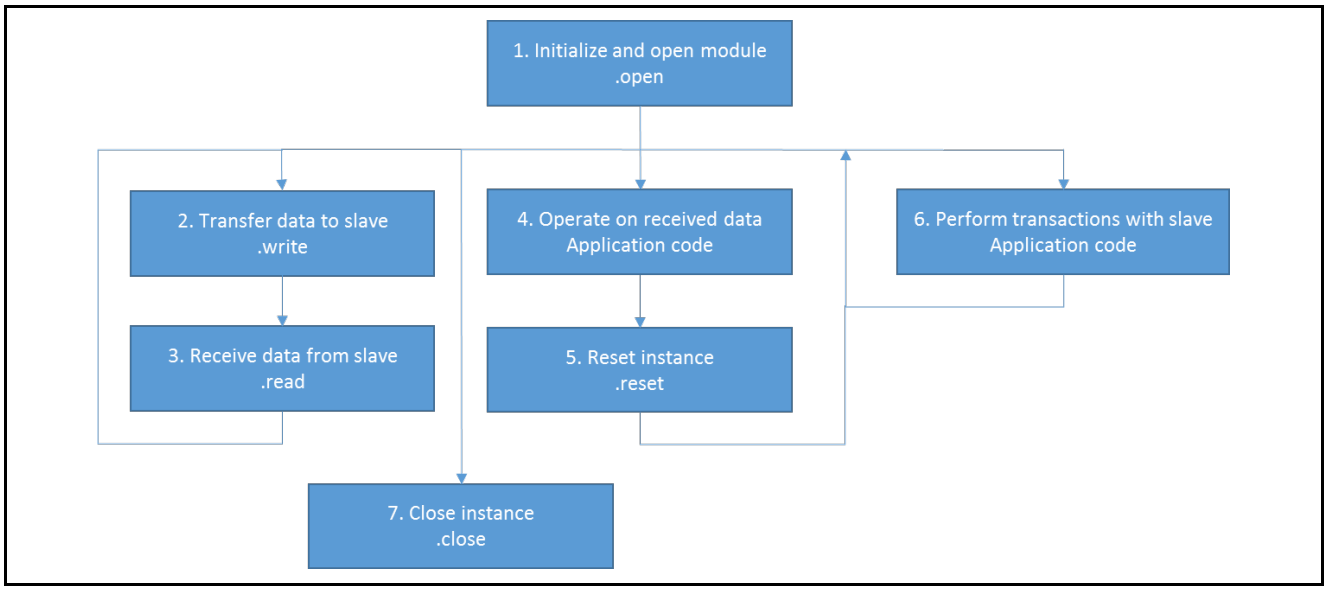

Figure 3. Flow Diagram of a Typical I²C SCI HAL Module Application

7. The I²C SCI HAL Module Application Project

The application project associated with this module guide demonstrates steps to perform in an example application. You may want to import and open the application project within the ISDE and view the configuration settings for the I²C SCI HAL module. You can also read over the code (in `sci_i2c_hal_slave_functions_mg.c`) which is used to illustrate the I²C SCI HAL module APIs in a complete design.

Table 9. I²C SCI HAL Module Configuration Settings for the Application Project

Resource	ISDE Property	Property / Configuration Setting
g_i2c I²C Master Driver on r_sci_i2c	Name	g_i2c
	Slave Address	0x48
	Channel	0
	Receive Interrupt Priority	Priority 3
	Transmit Interrupt Priority	Priority 3
	Transmit End Interrupt Priority	Priority 3
Pins tab > Pin Selection > Peripherals > Connectivity: SCI > SCI0	Pin Group Selection	Mixed
	Operation Mode	Simple I2C
	SDA	P411
	SCL	P410
Pins tab > Pin Selection > Ports > P4 > P411 and Pins tab > Pin Selection > Ports > P4 > P410	Mode	Peripheral Mode
	Pull up	None
	IRQ	None
	Drive Capacity	Low
	Output Type	n-ch open drain
Pins tab > Pin Selection > Ports > P6 > P609	Mode	Output Mode (Initial High)
	Pull up	None
	Drive Capacity	Low
	Output Type	CMOS
Connect these pins using a jumper wire on SK-S7G2 breakout board	SCL: P410	SCL: P512
	SDA: P411	SDA: P511

The application project demonstrates the typical use for the I²C SCI HAL module APIs. The configuration settings in the application project need to be customized for the specifics of the target kit and MCU device. The application project uses the r_sci_i2c module and channel 0 for I²C communication. The output pins for I²C communication are selected to conform to the signal connections from the touch controller. (P410 for SCL and P411 for SDA.) It can be helpful to open the application project in the ISDE and locate these settings in the Pin Configuration tab. These signals can also be located on the schematic for the SK-S7G2 target board as a check on the validity of the selected pins for the I²C signals. Finally, the external slave reset signal is connected to the GPIO pin P609 and must be enabled and configured for proper operation. All these application project-specific settings are in the prior configuration settings table.

Table 10.Software and Hardware Resources Used by the Application Project

Resource	Revision	Description
e² studio	6.2.1 or later	Integrated Solution Development Environment
SSP	v1.5.0 or later	Synergy Software Platform
IAR EW for Synergy	8.23.1 or later	IAR Embedded Workbench® for Renesas Synergy™
SSC	6.2.1 or later	Synergy Standalone Configurator
SK-S7G2	v3.0/ v3.1 or later	Starter Kit

Once the I²C SCI HAL module application project has been successfully added and configured, it can be used by the application program. The I²C application project implements steps similar to those in the previous figure; the key difference is that the read and write functions implement specific program functions to initialize, configure, and read data from the I²C slave device.

The following figure has a simple flow diagram showing the application project steps:

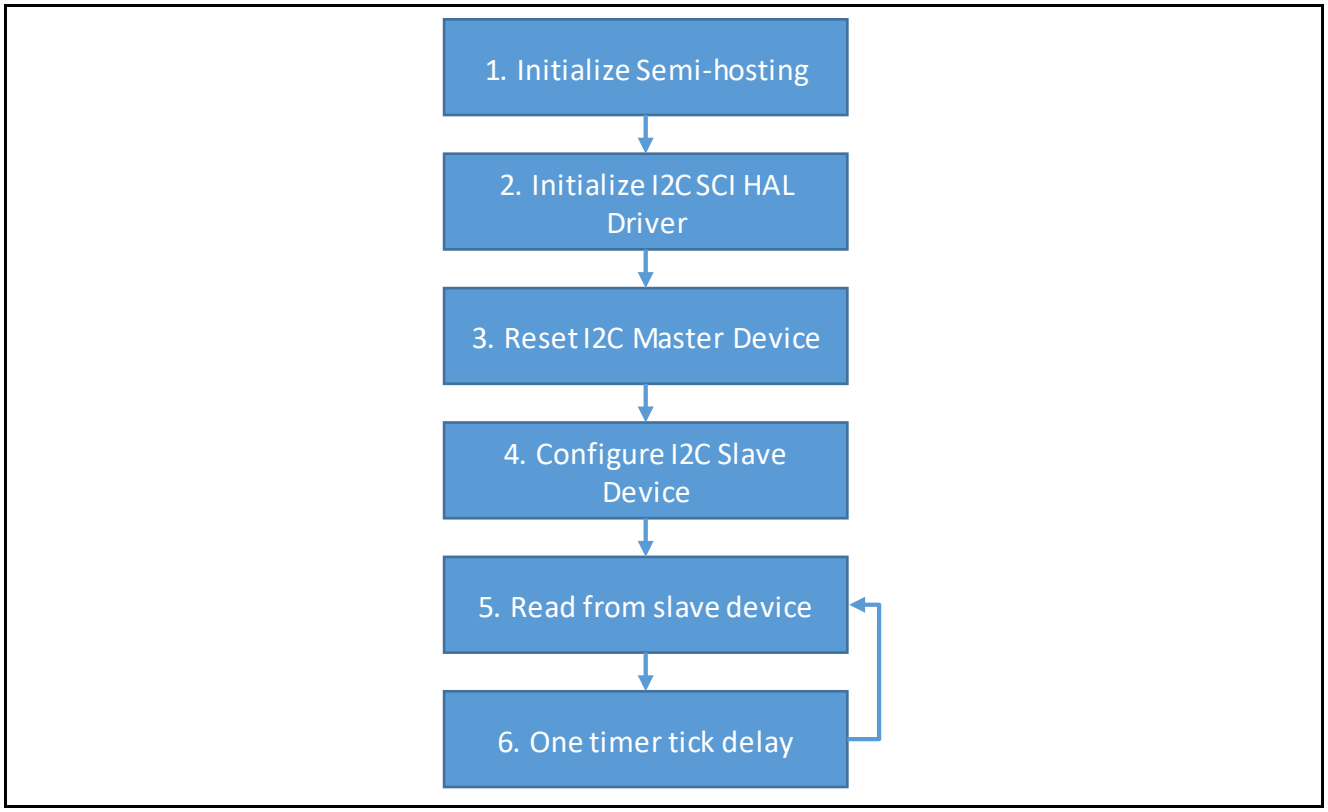

Figure 4. I²C SCI HAL Module Application Project Flow Diagram

After importing the application project into the ISDE, you can read through the code in `sci_i2c_hal_slave_functions_mg.c` along with the flow outlined in the previous figure. The first section of `sci_i2c_hal_slave_functions_mg.c` are the header files which reference the generated I²C instance structure. Additionally, the header files allow semi-hosting support to be implemented. Semi-hosting is a common technique used to display results using `printf()`.

The next section includes the function definitions for the I²C operations including initialization, reset, configuring the slave device, and reading from the slave device. The last section of the code, the `master_run()` function, is the HAL entry point. I²C functions declared in the previous sections are called in this function. Lastly, the read operation is called in the infinite loop to continuously read the I²C slave data.

Note: It is assumed that you are familiar with using `printf()` with the Debug Console in the Synergy Software Package. If you are unfamiliar with this function, refer to the *How do I Use Printf() with the Debug Console in the Synergy Software Package* Knowledge Base article, available in the References section at the end of this document. Alternatively, the user can see the results using the watch variables in the debug mode.

A few key properties are configured in this application project to support the required operations and the physical properties of the target board and MCU device. You can also open the application project and view these settings in the Properties window as a hands-on exercise.

8. Customizing the I²C SCI HAL Module for a Target Application

Some configuration settings are normally changed by the developer from those shown in the application project. For example, you can easily change the configuration settings for the I²C rate. You can also add more slaves to the I²C bus and use different instances of the I²C HAL module to address that slave by changing the slave address and instance name. The API can be used to change the slave-address at the run time with the same bus configuration and control data structure. The I²C HAL configuration also provides flexibility to use 7-bit or 10-bit addressing modes and callback functions for user-defined interrupt handling.

9. Running the I²C SCI HAL Module Application Project

To run the I²C SCI 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. Refer to the *Importing a Renesas Synergy Project* (r11an0023eu0121-synergy-ssp-import-guide.pdf, included in this package) for instructions on importing the project into e² studio ISDE or IAR Embedded Workbench and building/running the application.

To implement the I²C SCI HAL Module application in a new project, follow the steps for defining, configuring, auto-generating files, adding code, compiling, and debugging on the target kit. Following these steps is a hands-on approach that can help make the development process with the SSP more practical, while just reading over this guide tends 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 for a description of how to accomplish these steps.

To create and run the I²C SCI application project, simply follow these steps:

1. Import the attached application project SCI_I2C_HAL_MG_AP to e2 studio ISDE or the IAR EW for Synergy.
2. Compile the application without errors or warnings.
3. Connect to the host PC through a micro USB cable to J19 on SK-S7G2 target kit.
4. Use jumper wires to connect P411 (SDA (data line) of SCI module) to P511 (SDA (data line of touch controller) and P410 (SCL (clock line) of SCI module) to P512 (SCL (clock line) of touch controller).
5. Start to debug the application.
6. LED1-3 blinks when communication is ongoing. Also, if semi-hosting is uncommented in `i2c_hal_slave_functions_mg.h`, the output can be viewed in Debug Console window (see figure). Touching the touch screen causes values of the received data in the console to change.

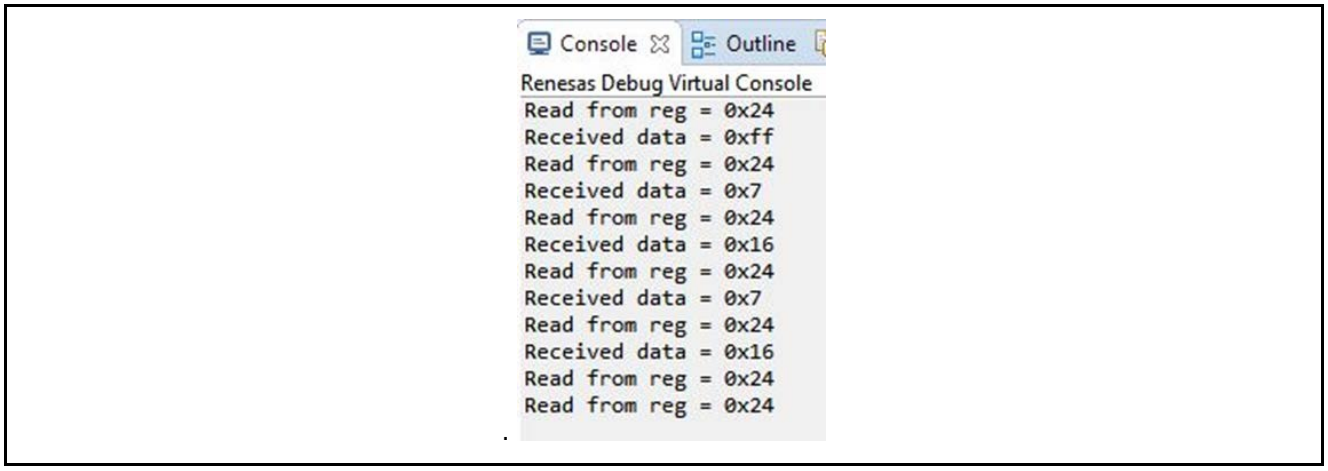

Figure 5. Example Output from I²C SCI HAL Module Application Project

10. I²C SCI HAL Module Conclusion

This module guide has provided all the background information needed to select, add, configure, and use the component 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 prevents common errors, like conflicting configuration settings or the incorrect selection of lower-level drivers. The use of high-level APIs (as demonstrated in the application project) illustrates the 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. I²C SCI HAL Module Next Steps

After you have mastered a simple I²C HAL module application project, you may want to review a more complex example. The I²C Framework is a set of ThreadX®-aware Framework APIs. The I²C Framework handles the integration and synchronization of multiple I²C peripherals on the I²C bus. With the I²C Framework, you can create one or more I²C buses and connect multiple I²C peripherals to each I²C bus. The I²C Framework uses a single interface to access both SCI I²C and RIIC drivers. You can learn more about

the I²C Framework by reading the associated module guides listed in the References section at the end of this document.

12. I²C SCI HAL Module Reference Information

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

To find the most up to date reference materials and their locations, visit the Synergy Knowledge Base and do a search for the module name and include **module guide references** in the search.

For example, if you are looking at the References for the `r_sci_i2c` module, visit https://en-us.knowledgebase.renesas.com/English_Content/Renesas_Synergy™_Platform/Renesas_Synergy_Knowledge_Base and enter `r_sci_i2c` **module guide references** in the search bar. The search will bring up a list of results, and the top one will be the References Page for that Module Guide. The following url will take you directly to the search results for this module example.

https://en-us.knowledgebase.renesas.com/English_Content/Renesas_Synergy%E2%84%A2_Platform/Renesas_Synergy_Knowledge_Base/r_sci_i2c_Module_Guide_Resources

Website and Support

Visit the following vanity URLs to learn about key elements of the Synergy Platform, download components and related documentation, and get support.

Synergy Software	www.renesas.com/synergy/software
Synergy Software Package	www.renesas.com/synergy/ssp
Software add-ons	www.renesas.com/synergy/addons
Software glossary	www.renesas.com/synergy/softwareglossary
Development tools	www.renesas.com/synergy/tools
Synergy Hardware	www.renesas.com/synergy/hardware
Microcontrollers	www.renesas.com/synergy/mcus
MCU glossary	www.renesas.com/synergy/mcuglossary
Parametric search	www.renesas.com/synergy/parametric
Kits	www.renesas.com/synergy/kits
Synergy Solutions Gallery	www.renesas.com/synergy/solutionsgallery
Partner projects	www.renesas.com/synergy/partnerprojects
Application projects	www.renesas.com/synergy/applicationprojects
Self-service support resources:	
Documentation	www.renesas.com/synergy/docs
Knowledgebase	www.renesas.com/synergy/knowledgebase
Forums	www.renesas.com/synergy/forum
Training	www.renesas.com/synergy/training
Videos	www.renesas.com/synergy/videos
Chat and web ticket	www.renesas.com/synergy/resourcelibrary

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Aug.2.19	—	First release document
1.01	Feb.06.19	—	Updated to SSP v1.5.0

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 claims 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, drawings, charts, programs, algorithms, and 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 reverse engineer 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, copying or reverse engineering.
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; 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.
   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or 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 damage (space system; 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 any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user's manual or other Renesas Electronics document.
6. When using 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 dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of 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. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a 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. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating 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. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using 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 comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
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.

(Note1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.
(Note2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

(Rev.4.0-1 November 2017)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/.