Renesas Synergy™ Platform

Key Matrix 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 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 Key Matrix HAL module is a high-level API for Key Matrix HAL applications and is implemented on r_kint. The Key Matrix HAL module uses the key-interrupt function peripheral on the Synergy MCU. A user-defined callback can be created to inform the CPU of a key press event.

Contents

1. Key Matrix HAL Module Features

This Key Matrix HAL module configures and controls the Key Interrupt (KINT) peripheral. It implements the following key functions:

- Supports both rising and falling edges on KINT channels
- Supports interrupt-based event notification
- Supports a bit-masking function to capture multiple events efficiently
- Supports a matrix keypad with edges on any two channels

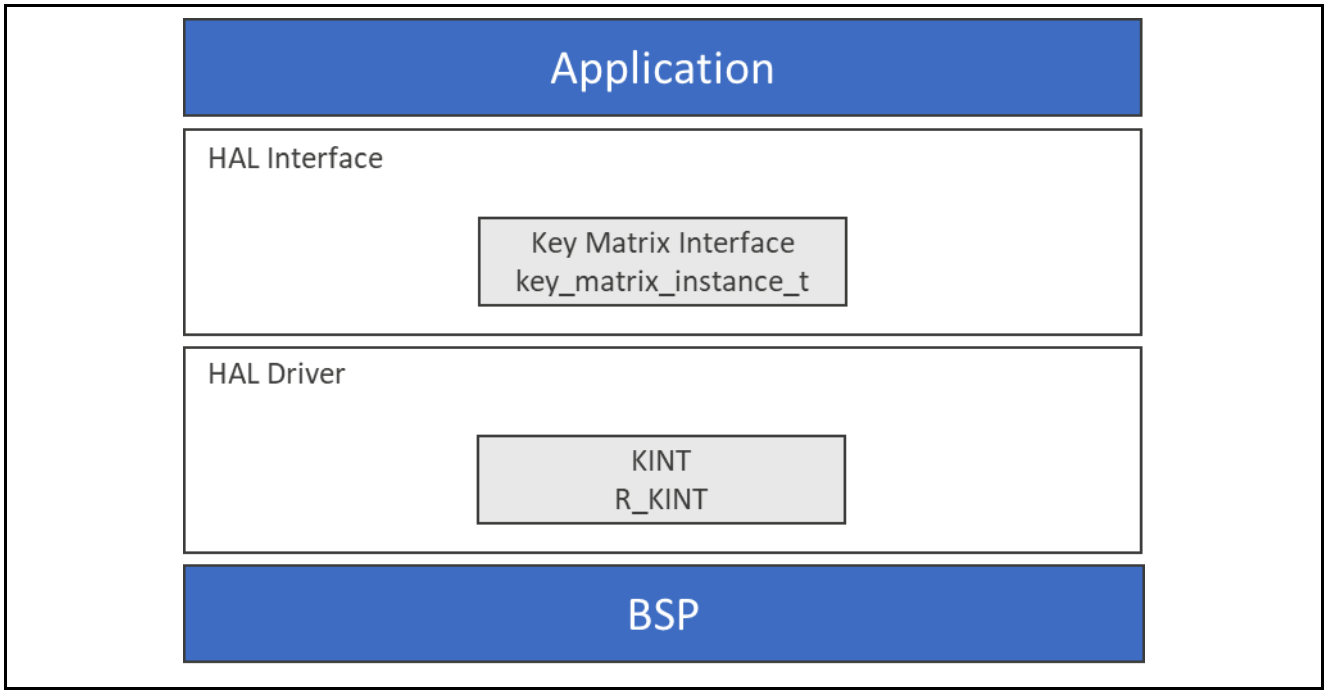

Figure 1. Key Matrix HAL Module Block Diagram

2. Key Matrix HAL Module APIs Overview

The Key Matrix HAL module defines APIs for opening, closing, enabling, and disabling key-interrupt functions. 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. Key Matrix HAL Module API Summary

Function Name	Example API Call and Description
.open	`g_keymatrix_on_kint.p_api->open(g_kint.p_ctrl, g_kint.p_cfg)` Initial configuration.
.enable	`g_keymatrix_on_kint.p_api->enable(g_kint.p_ctrl)` Enable Key interrupt.
.disable	`g_keymatrix_on_kint.p_api->disable(g_kint.p_ctrl)` Disable Key interrupt.
.triggerSet	`g_keymatrix_on_kint.p_api->triggerSet(g_kint.p_ctrl, trigger)` Set trigger for Key interrupt.
.close	`g_keymatrix_on_kint.p_api->close(g_kint.p_ctrl)` Allow driver to be reconfigured. May reduce power consumption.
.versionGet	`g_keymatrix_on_kint.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	Function successfully completed.
SSP_ERR_ASSERTION	Parameter has invalid value.
SSP_ERR_INVALID_ARGUMENT	Argument is invalid.
SSP_ERR_HW_LOCKED	The API has already been opened. It must be closed before it can be opened again.
SSP_ERR_NOT_OPEN	The peripheral is not opened.

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. HAL Module Operational Overview

The Key Matrix HAL module configures the Key Interrupt (KINT) peripheral to detect rising or falling edges on any of the KINT channels. When such an event is detected on any of the configured pins, the module generates an interrupt; the interrupt then calls the user callback (p_callback) with the callback argument keymatrix_callback_args_t that specifies the channel(s) on which the edge was detected using a bitmask.

Even though detection of an edge on any one channel generates the interrupt, the callback returns a bitmask keymatrix_channels_t of all the pins that were triggered at that time if any other pins also detected an edge. Thus, an interrupt is not necessarily generated for edge detection on each pin if an edge was also detected on another pin before the callback was called. If a new edge is detected after the callback was called, then the interrupt is triggered again, resulting in a new callback.

This module can be used to implement a matrix keypad with edges on any two channels indicating the actual key that was pressed; alternatively, the module can be used as a single input to detect an edge on an input pin.

3.1 Key Matrix HAL Module Important Operational Notes and Limitations

3.1.1 Key Matrix HAL Module Operational Notes

- To trigger a transfer of data using the DMAC or DTC peripheral when a trigger edge is detected, configure the DMAC/DTC transfer with activation_source set to ELC_EVENT_KEY_INT.
- The KINT module can trigger the start of other peripherals available to the ELC. For details, see the ELC section in the *SSP User's Manual*.
- You must enable the KINT (INTKR) interrupt in the BSP for this module to operate, regardless of whether a callback is used in the open call.

3.1.2 Key Matrix HAL Module Limitations

- This module does not support polling-mode operation.
- Refer to the latest SSP Release Notes for any additional operational limitations for this module.

4. Including the Key Matrix HAL Module in an Application

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

Note: It is assumed 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 items, refer to the first few chapters of the *SSP User's Manual* to learn how to manage each of these important steps in creating SSP-based applications.

To add the Key Matrix 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 Key Matrix Driver is `g_kint0`. This name can be changed in the associated Properties window.)

Table 3. Key Matrix HAL Module Selection Sequence

Resource	ISDE Tab	Stacks Selection Sequence
g_kint0 Key Matrix Driver on r_kint	Threads	New Stack> Driver> Input> Key Matrix Driver on r_kint

When the Key Matrix Driver on `r_kint` is added to the thread stack as shown in the following figure, the configurator automatically adds any needed lower-level modules. Any drivers that need additional configuration information is box text highlighted in Red. Modules with a Gray band are individual modules that stand alone.

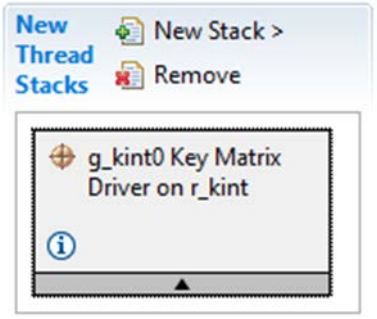

Figure 2. Key Matrix HAL Module Stack

5. Configuring the Key Matrix HAL Module

The Key Matrix HAL module on `r_kint` must be configured by the user 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 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 Integrated Solution Developer Environment (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 includes an indication as to 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 within 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. This 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 Key Matrix HAL Module on r_kint

ISDE Property	Value	Description
Parameter Checking	BSP, Enabled, Disabled (Default: BSP)	Enable or disable the parameter error checking.
Name	g_kint0	Module name
Keymatrix Channel Mask	Select Channels Below	This is a bit-mask with each bit specifying if that channel is to be enabled or not. Select the channels to be used.

ISDE Property	Value	Description
Channel 0:7	Unused, Used (Default: Unused)	Channel 0:7 selection.
Trigger Type	Rising Edge, Falling Edge (Default: Rising Edge)	Specify if the enabled channels detect a rising edge or a falling edge. NOTE: either all channels detecting a rising edge or all channels detecting a falling edge.
Interrupt enabled after initialization	True, False (Default: False)	Specify if the module interrupts must be enabled as part of the open call.
Callback	NULL	Callback selection.
Interrupt Priority	Priority 0 (highest), 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15 (lowest, not valid if using Thread X), Disabled (Default: Disabled)	Interrupt priority selection.

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

5.1 Key Matrix HAL Module Clock Configuration

The Key Matrix HAL module does not require a specific clock configuration.

5.2 Key Matrix HAL Module Pin Configuration

The KINT 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 KINT pins.

Table 5. Pin Selection for the Key Matrix HAL Module on r_kint

Resource	ISDE Tab	Pin selection Sequence
KINT	Pins	Select Peripherals > Input:KINT> KINT0

Note: The selection sequence assumes KINT0 is the desired hardware target for the driver.

Table 6. Pin Configuration Settings for the Key Matrix HAL Module on r_kint

Property	Value	Description
Operation Mode	Disabled, Custom (Default: Disabled)	Select Custom as the Operation Mode
KRM0:7	None, Pnn (Default: None)	Key Interrupt Pin selection

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

6. Using the Key Matrix HAL Module in an Application

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

1. Initialize the Key Matrix HAL module using the `open` API
2. If the `autostart` configuration setting is true, the module starts operation immediately
3. If the `autostart` is not set, use enable API to enable operation
4. Respond to key inputs
5. Disable operation using the `disable` API
6. To modify trigger edge after initialization, use the `triggerSet` API
7. Close the module by using the `close` API

The following figure illustrates these common steps are illustrated in a typical operational flow diagram:

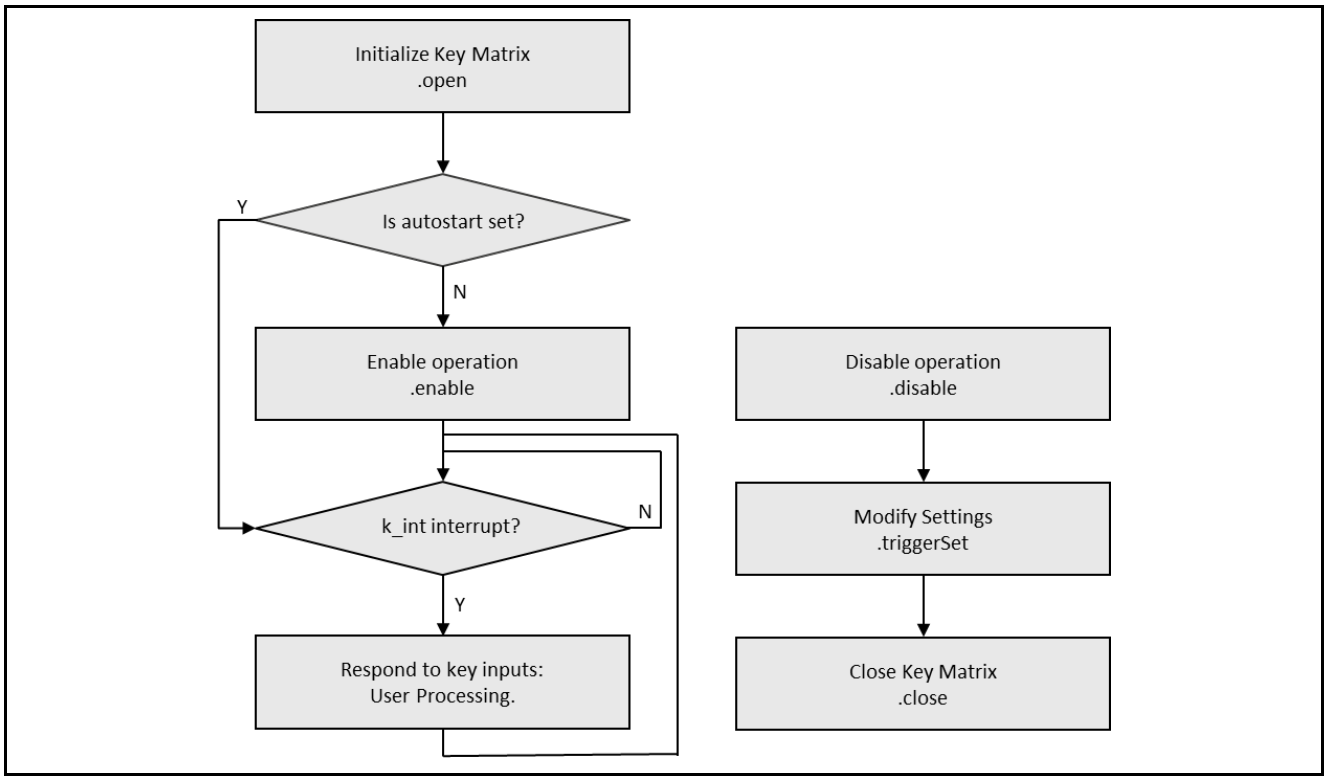

Figure 3. Flow Diagram of a Typical Key Matrix HAL Module Application

7. The Key Matrix HAL Module Application Project

The application project associated with this module guide demonstrates these 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 Key Matrix HAL module. You can also read over the code (in `kint_hal.c`) which is used to illustrate the Key Matrix HAL module APIs in a complete design.

The application project demonstrates the typical use of the Key Matrix HAL module APIs in a typical application, which is interfacing to a key matrix keypad. The key matrix keypad is a 3 x 4 configuration, where three column lines are actively controlled by the user application and the four row lines are interfaced to the KINT peripheral.

The application project initializes the Key Matrix HAL module and enables the key interrupt; additional modules are also initialized.

The control of the three column control lines is performed every 100 ms. One of the AGT (Asynchronous General Purpose Timer) timers is used for this purpose and is configured to generate a periodic interrupt every 100 ms. When a key press is detected, a KINT interrupt is generated and the KINT callback is called with the parameter identifying which row of the key matrix has been pressed. Subsequent control determines which column of the key matrix has been pressed, as done by the KINT-callback function.

As the application project is interrupt driven, once the Key Matrix HAL Module and AGT timer are active, the application enters an empty `while(1)` loop.

To run this application project, a 3x4 keypad is recommended. The following figure shows how the 3 x 4 keypad is connected to the SK-S7G2 board.

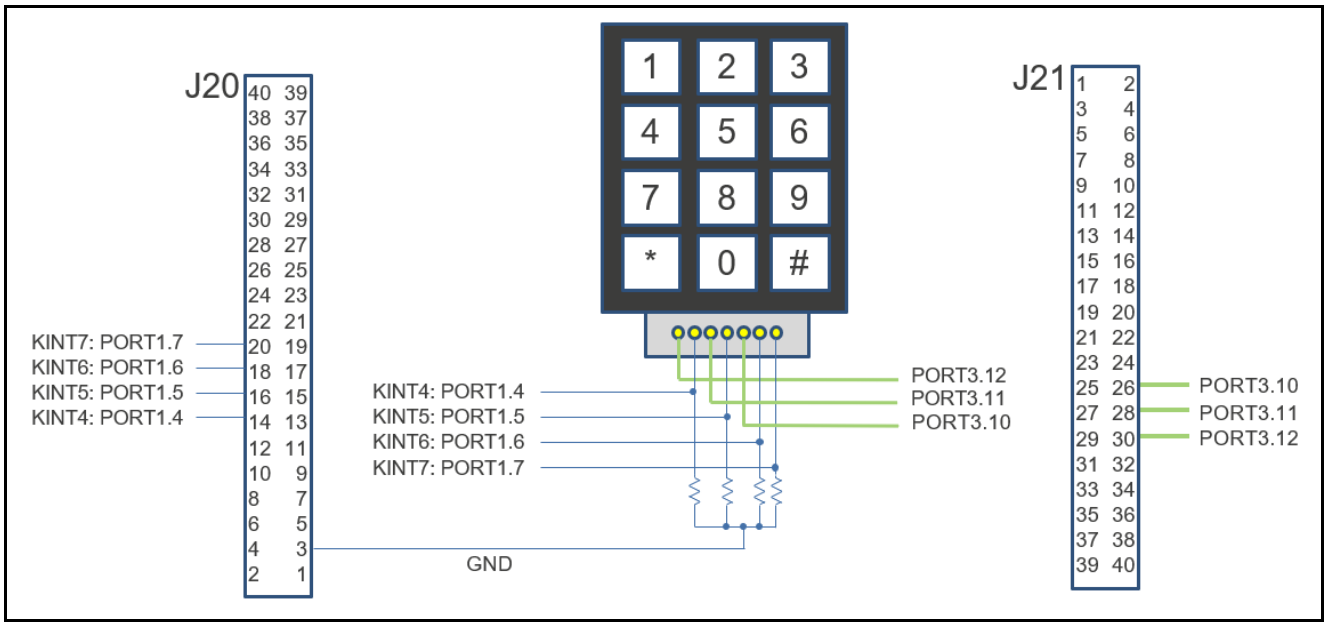

Figure 4. Connecting the Keypad to the SK-S7G2 board

In absence of a keypad, the user can verify their application by shorting the row and column lines using a wire. The following table shows how the keys are interfaced between rows and columns.

Table 7. Key Matrix

	Column 1 [P3:12]	Column 2[P3:11]	Column 3[P3:10]
Rows 1 P[1.7]	5	4	6
Rows 2 P[1.6]	8	7	9
Rows 3 P[1.5]	0	*	#
Rows 4 P[1.4]	2	1	3

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
Keypad	NA	Standard 3x4 Keypad

The following diagram shows a simple operational flow of the application project:

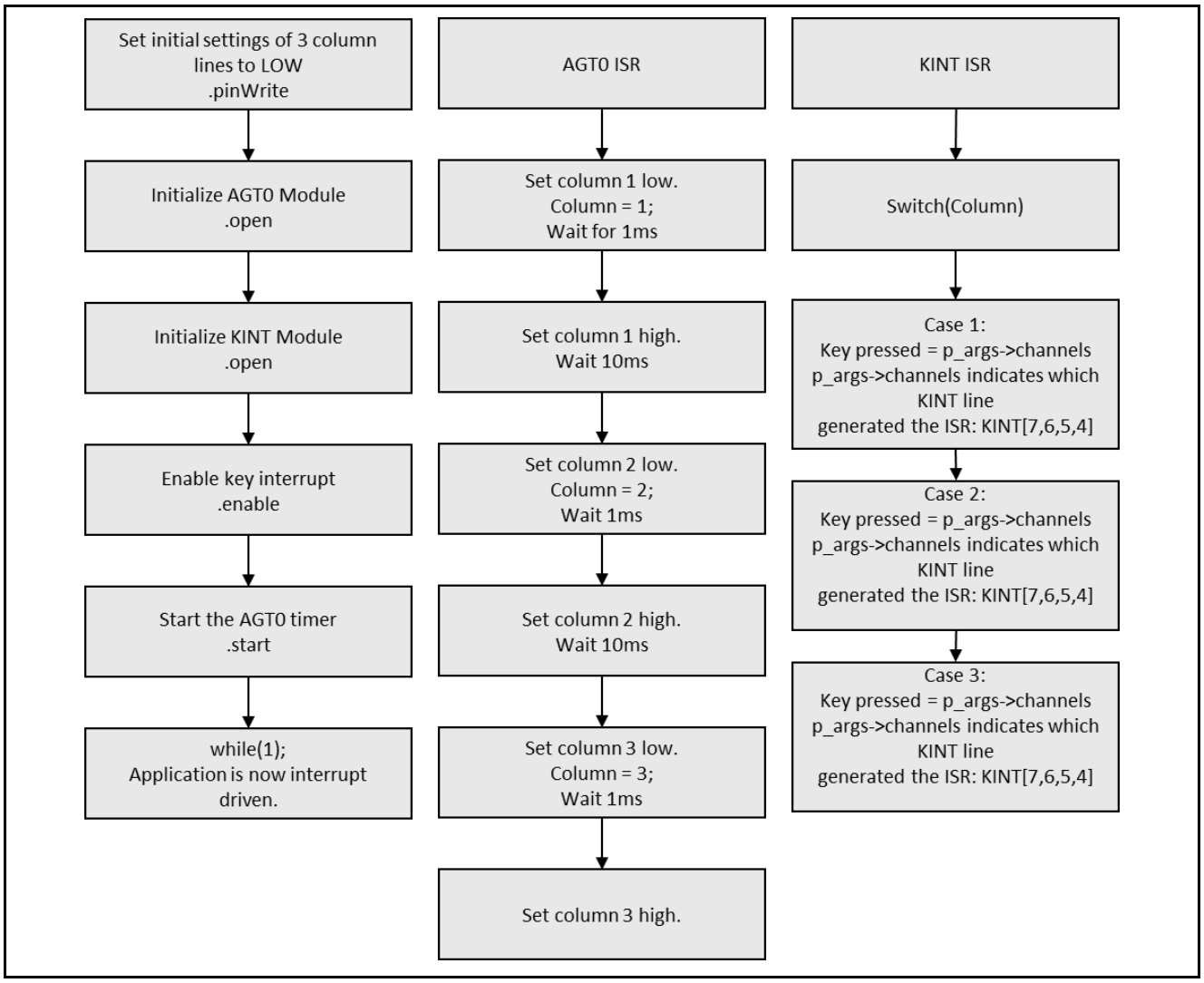

Figure 5. Key Matrix HAL Module Application Project Flow Diagram

The `kint_hal.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 APIs.

The first section of `kint_hal.c` defines keys on the key matrix, time delays in milliseconds, and output column number enumerate variables. This section also prototypes the functions of the file and defines an array which represents whether a specific key is pressed or not.

The column lines of the key matrix are controlled via three output pins. Even though the initial output state is defined in the Synergy Pin Configurator and set as part of board support package (BSP) initialization; the application sets these pins to a known state using the IOPORT `pinWrite` API.

In the next section, the AGT and KINT modules are opened. The AGT is used to generate a 100 ms interrupt, and the KINT is opened with its interrupt disabled. Once opened, the AGT is started and the KINT interrupt is enabled. It would be valid for the AGT open to start the timer automatically and for the KINT open to enable the interrupts; this method of discreet steps was chosen to demonstrate more API calls of the associated modules.

As the application project is interrupt driven, the application now executes an empty `while(1)` loop.

In the following section, the AGT interrupt callback function is called; this callback takes each of the column lines from low-to-high-to-low with a user-defined delay between the transitions by calling the user-defined function `change_pin()`.

The last section is the KINT interrupt callback function, which determines which key has been pressed. The callback function has a parameter identifying which row of the key matrix has been pressed. By knowing which column line was high at the time of the KINT ISR, and which KINT line is generating the interrupt, it is

a simple process to determine which key was pressed. The specific element of the key array variable is set to true, which represents the key pressed.

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. The properties with the values set for this specific project are listed in the following tables. You can also open the application project and view these settings in the Properties window as a hands-on exercise.

Table 9. Key Matrix HAL Module Configuration Settings for the Application Project

ISDE Property	Value Set
Name	g_kint
Keymatrix Channel Mask	Select Channels Below
Channel 0	Unused
Channel 1	Unused
Channel 2	Unused
Channel 3	Unused
Channel 4	Used
Channel 5	Used
Channel 6	Used
Channel 7	Used
Trigger type	Rising edge
Interrupt enabled after initialization	False
Callback	g_kint_callback
Interrupt Priority	Priority 4 (CM4: valid, CM0+: invalid)

Table 10. AGT HAL Module Configuration Settings for the Application Project

ISDE Property	Value Set
Name	g_agt0
Channel	0
Mode	Periodic
Period Value	100
Period Unit	Milliseconds
Auto Start	False
Count Source	LOCO
AGT0 Output Enable	False
AGTIO Output Enable	False
Output Inverted	False
Callback	g_agt0_callback
Interrupt Priority	Priority 8 (CM4: valid, CM0+: invalid)

In addition, the application project requires the pin configuration as presented in the following table:

Table 11. Pin Configuration

Pin Selection Sequence	Pin Configuration Property	Setting
Ports > P3 > P312	Mode	Mode: Output mode (Initial Low)
Ports > P3 > P311	Mode	Mode: Output mode (Initial Low)
Ports > P3 > P310	Mode	Mode: Output mode (Initial Low)
Peripherals > Input:KINT > KRM4	KRM4	P104
Peripherals > Input:KINT > KRM5	KRM5	P105
Peripherals > Input:KINT > KRM6	KRM6	P106
Peripherals > Input:KINT > KRM7	KRM7	P107

8. Customizing the Key Matrix 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 channels you want to use for input and the pins you want to use for output. You can also change the number of channels or outputs to meet the requirements of your own hardware.

9. Running the Key Matrix HAL Module Application Project

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

1. Create a new Renesas Synergy project for the SK-S7G2 called **KINT_HAL_MG_AP**.
2. Select the **Threads** tab.
3. Add the **Key Matrix HAL** module to HAL/Common and set its parameters.
4. Click on the **Generate Project Content** button.
5. Add the code from the supplied project files `kint_hal.c`.
6. Build the project.
7. Connect to the host PC with USB cable using DEBUG_USB (J19) socket.
8. Start to debug the application
9. If semi hosting is enabled, the output can be viewed on Renesas Debug Virtual Console as shown in Figure 6Figure 6

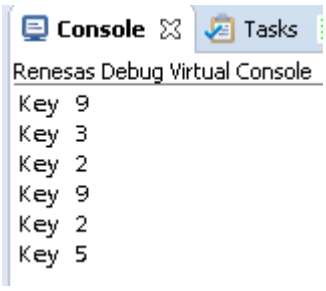

Figure 6. Example Output from Key Matrix HAL Module Application Project

10. Key Matrix 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 drivers. The use of high-level APIs (as demonstrated in the application project) illustrates the additional development time savings achieved by allowing work to begin at a high level; avoiding the time required in older development environments to use or, in some cases, create, lower-level drivers.

11. Key Matrix HAL Module Next Steps

After you have mastered a simple Key Matrix HAL module project, you may want to review a more complex example. You may find that the running a Key Matrix function in a separate thread is better for your application. You may then want to run the ThreadX® RTOS to create a multi-thread design.

12. Key Matrix HAL Module Reference Information

SSP User Manual: Available in HTML format at www.renesas.com/us/en/products/synergy/software/ssp.html as a SSP distribution package, and also as a pdf from the Synergy Gallery.

Links to all the most up-to-date r_kint module reference materials and resources are available on the Synergy Knowledge Base: https://en-support.renesas.com/knowledgeBase/16977493.

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	Jun.16.17	—	Initial Release
1.01	Aug.30.17	7	Update to Hardware and Software Resources Table
1.02	Feb.27.19	—	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/.