Renesas Synergy™ Platform

Capacitive Touch Button Framework 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 Capacitive Touch Button Framework Module Reference Information section at the end of this document). It should be valuable resources for creating more complex designs.

The Capacitive Touch Button Framework module is a high-level ThreadX aware API for Capacitive Touch Button applications and is implemented on `sf_touch_ctsu` using the ThreadX RTOS. The Capacitive Touch Button Framework module uses the CTSU (Capacitive Touch Sensor Unit) peripheral on a Synergy MCU. A user-defined callback can be created to respond to each button in the order in which they are present.

Contents

1. Capacitive Touch Button Framework Module Features

The Capacitive Touch Button Framework module is used to interpret the CTSU data for all the buttons that are present in the system. Some of the key features are:

- Works in conjunction with the Capacitive Touch Workbench for the Renesas Synergy™ (CTW for Synergy) tool which generates configuration data.
- Provides a callback function to events.
  - Performs debouncing
  - Supports multiple types of events including Press, Release, and LongTouch
  - Performs the callback for each button in the order in which they are present in the button configuration table.

The Capacitive Touch Button Framework requires the Capacitive Touch framework. In most cases, all the needed configuration information is automatically added to the modules. The ISDE configuration options are described further in this document.

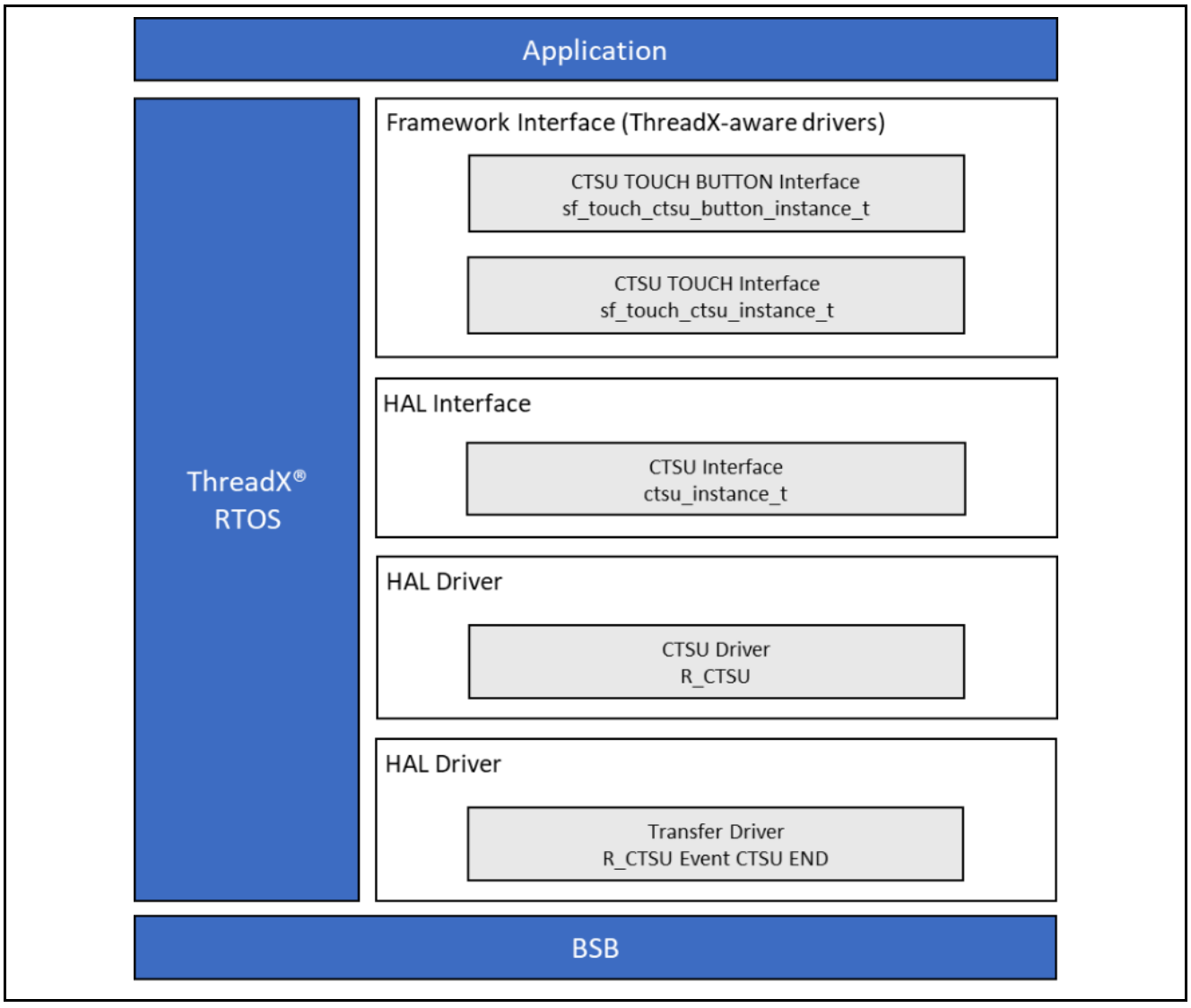

Figure 1. Capacitive Touch Button Framework Module Block Diagram

2. Capacitive Touch Button Framework Module APIs Overview

The Capacitive Touch Button defines APIs for open, enable, disable, and close. 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 return status values follows the API summary table.

Table 1. Capacitive Touch Button Framework Module API Summary

Function Name	Example API Call and Definition
.open	`g_sf_touch_button0.p_api->open( g_sf_touch_button0.p_ctrl, g_sf_touch_button0.p_cfg);` Initializes the CTSU Button Framework. Configures the lower level hardware and register callback functions for all the buttons.
.enable	`g_sf_touch_button0.p_api->enable( g_sf_touch_button0.p_ctrl, button_id);` Enables callback notification for a configured button.
.disable	`g_sf_touch_button0.p_api->disable( g_sf_touch_button0.p_ctrl, button_id);` Disables callback notification for a configured button.
.close	`g_sf_touch_button0.p_api->open( g_sf_touch_button0.p_ctrl);` Closes the CTSU Button Framework. Closes the button framework and lower layers if no other modules are using it.
.versionGet	`g_sf_touch_button0.p_api->versionGet(&p_version);` Gets version and stores it in the provided pointer p_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 successful.
SSP_ERR_ASSERTION	Assertion error.
SSP_ERR_IN_USE	The framework has already been initialized by this particular widget.
SSP_ERR_NOT_OPEN	Device not open.
SSP_ERR_INTERNAL	Internal error.

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. Capacitive Touch Button Framework Module Operational Overview

The Capacitive Touch Button Framework is used to interpret the CTSU data for all the buttons that are present in the system. It also initializes the CTSU framework layer, which in turn initializes the hardware layer. The Capacitive Touch Button Framework registers a callback with the CTSU Framework layer, which is called each time processed data is available. The Capacitive Touch Button Framework then uses this processed data (binary) to perform debouncing and to determine which of the configured events (Press, Release, LongTouch, and so forth) are valid for each button.

The Framework calls the callback for each button in the order in which they are present in the button configuration table. Unless the user stops the scan process, the scan continues to be triggered by the timer and data is written into the user buffer, which is treated by the Framework as a circular buffer. The name and length of the buffer are specified through the ISDE configurator.

3.1 Capacitive Touch Button Framework Module Operational Notes

This framework is designed to be used in conjunction with the configuration data generated by the Workbench6 CTW for Synergy tool. Refer to the *Workbench6 User's Manual* on the Renesas Synergy website for details on how to use the tool to generate configuration data.

3.2 Capacitive Touch Button Framework Module Limitations

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

4. Including the Capacitive Touch Button Framework Module in an Application

This section describes how to include the Capacitive Touch Button Framework 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 tasks, 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 Capacitive Touch Button Framework to an application, simply add it to a thread using the stack's selection sequence given in the following table. (The default name for the Capacitive Touch Button Framework is sf_touch_ctsu_button. This name can be changed in the associated Properties window.)

Table 3 Capacitive Touch Button Framework Module Selection Sequence

Resource	ISDE Tab	Stacks Selection Sequence
g_sf_touch_button0 Cap Touch Button Framework on sf_touch_ctsu_button	Threads	New Stack > Framework > Input > Cap Touch Button Framework on sf_touch_ctsu_button

When the Capacitive Touch Button Framework on sf_touch_ctsu_button is added to the thread stack as shown in the following figure, the configurator automatically adds any needed lower-level modules. Any lower-level modules that need additional configuration information have text highlighted in Red. Modules with a Gray band are individual modules that stand alone. Modules with a Blue band are shared or common and need only be added once, and can be used by multiple stacks. Modules with a Pink band can require the selection of lower-level modules; these are either optional or recommended (this is indicated in the block with the inclusion of this text.) If the addition of lower-level modules is required, the module description includes Add in the text. Clicking on any Pink banded modules brings up the New icon and displays possible choices.

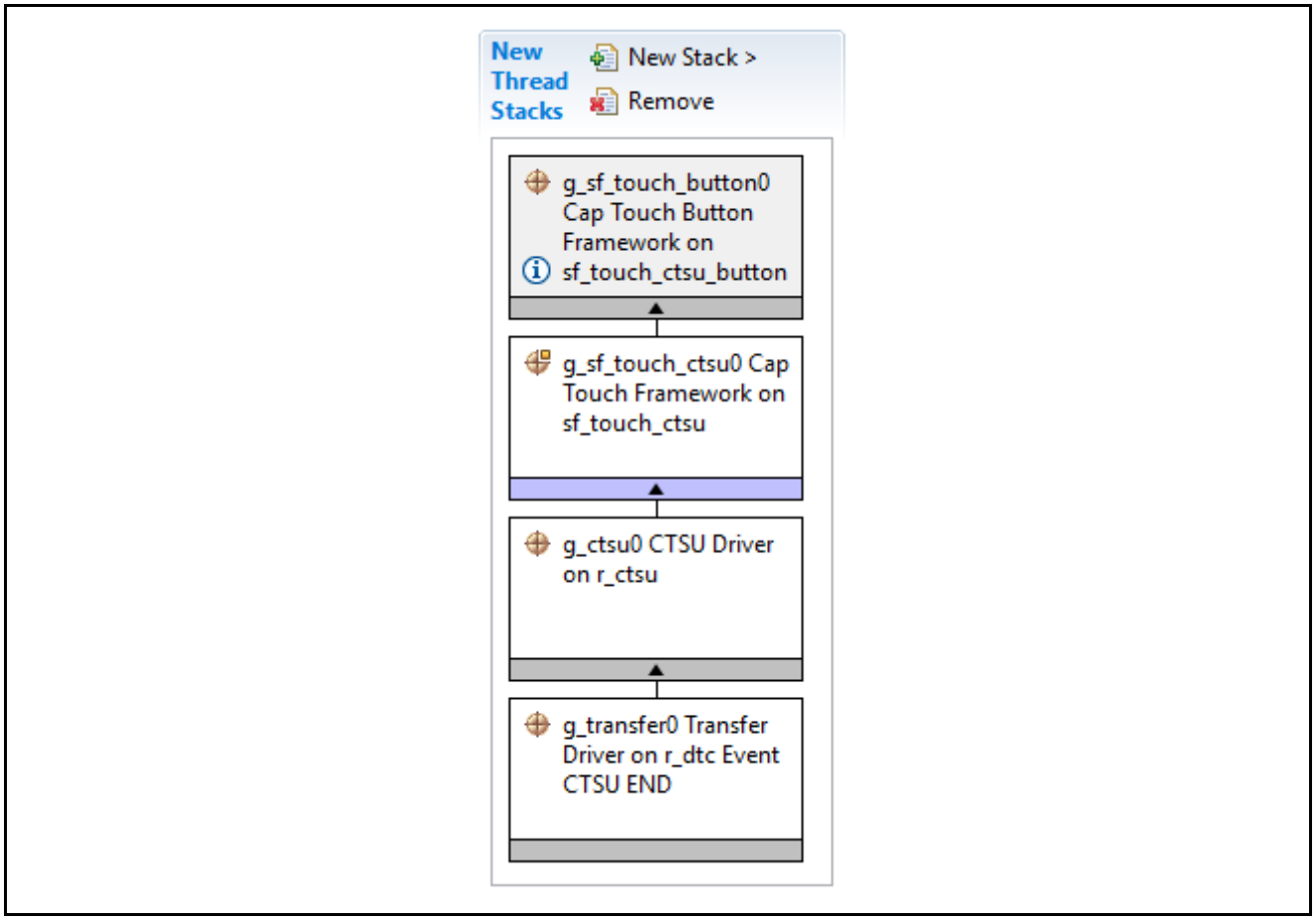

Figure 2. Capacitive Touch Button Framework Module Stack

5. Configuring the Capacitive Touch Button Framework Module

The Capacitive Touch Button Framework module 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 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 include the validity of the setting based on the targeted MCU (CM4 or CM0+). This level of detail is not in the following configuration properties table 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 while 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 Capacitive Touch Button Framework Module on sf_touch_ctsu_button

Parameter	Value	Description
Parameter Checking	BSP, Enabled, Disabled (Default: BSP)	Controls whether to include code for API parameter checking
Number of Buttons	1	Number of buttons configured in CTW for Synergy
Short hold debounce multiplier	1	Specify the multiplier for a button short-hold event
Long hold debounce multiplier	5	Specify the multiplier for a button long-hold event
Stuck in debounce multiplier	10	Multiplier used in debounce function
Multi touch enable	Enabled, Disabled (Default: Disabled)	Enables or disables multi-touch detection
Enable stuck at condition detection	Enabled, Disabled (Default: Disabled)	Specify the multiplier for a button stuck-at event
Name	g_sf_touch_button0	Module name
Button Configuration Structure Name	touch_buttons	Name of button configuration structure generated by CTW for Synergy
Callback	g_button_framework_user_callback	Name of callback function created by user application.

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

Settings other than the defaults for lower-level modules can be desirable in some cases. The configurable properties for the lower-level stack modules are given in the following sections for completeness and as a reference.

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

5.1 Configuration Settings for the Capacitive Touch Button Framework Low Level Modules

Typically, only a small number of settings must be modified from the default for lower-level modules as indicated using the red text in the thread stack block. Notice that some of the configuration properties must be set to a certain value for proper framework operation and are locked to prevent user modification. The following tables identify all the settings within the properties section for the module.

Table 5. Configuration Settings for the Capacitive Touch Framework on sf_touch_ctsu

ISDE Property	Value	Description
Parameter Checking	BSP, Enabled, Disabled (Default: BSP)	Enables or disables the parameter error checking.
Name	g_sf_touch_ctsu0	Module name.
Thread Priority	3	User determined based on priority of other threads in system. Recommend high priority.
Update Hz	50	Rate at which the CTSU hardware scans should occur. The maximum scan rate should be less than the tick rate set for ThreadX. The total time to complete a hardware scan is determined by the number of channels used. Each channel takes approximately 600 µs to complete a scan. So, if 10 channels are used for 10 buttons in self-capacitance mode, the total hardware scan time is ~6 ms or 166 Hz. Similarly, if 7 channels are used for 5 buttons in mutual-capacitance mode (5x2 layout), the total hardware scan time is ~4.2 ms or 250 Hz. The software processing time is additional and varies depending on the core clock and the software filter depth used by the CTSU. Thus, the maximum scan rate should be lesser than the time required to scan and process all the channels used in the configuration and lesser than the RTOS tick rate.
Callback	NULL	Name of user callback that is called when each scan is complete and also when new processed data is available. Can be NULL if not used.

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

Table 6. Configuration Settings for the CTSU HAL Module on r_ctsu

ISDE Property	Value	Description
Parameter Checking Enable	BSP, Enabled, Disabled (Default: BSP)	Include parameter checking code
Offset Adjustment	Enabled, Disabled (Default: Enabled)	Module Name
Drift Compensation	Enabled, Disabled (Default: Enabled)	Configures the start mode as register start or auto-start
Drift Compensation Method (valid only if Drift Compensation is enabled above)	Alternate Method 1	Controls whether WDT is started during initialization
Steady state drift compensation rate, drift compensation is applied per n scans	500	WDT timeout period
Startup drift compensation rate (Should be less than the steady state drift compensation)	5	WDT clock divider

ISDE Property	Value	Description
Channel release compensation rate (Should be less than the steady state drift compensation rate)	500	Permitted refresh period start position.
Default filter depth (used in sensor count filter provided by driver)	1	Permitted refresh period end position.
Runtime rate of tuning of sensor values (if drift compensation is used this value is overridden to be twice the rate of the steady state drift compensation)	800	Select whether WDT should reset the MCU or generate an NMI.
Perform auto-tune and drift compensation only when all channels are untouched	True, False (Default: True)	Select whether the WDT should stop counting in low power modes.
Max. active channels	1	Callback. A user callback function can be registered in open. If this callback function is provided, it is called from the interrupt service routine (ISR) each time the IRQn triggers. **Important:** Since the callback is called from an ISR, care should be taken not to use blocking calls or lengthy processing. Spending excessive time in an ISR can affect the responsiveness of the system.
Name	g_ctsu0	Module name.
CTSU configuration used	g_ctsu_config	Name of the configuration structure generated by CTW for Synergy.
Callback	NULL	The user callback function that is invoked each time the scan is complete as well as when newly processed data is available. Can be set to NULL if not used.
Data Processing Option	Default Processing (recommended), No Processing (tuning only) (Default: Default Processing)	Can be used to specify if scans should start after the data from the previous one is completed, if auto-calibration should be run between scans, and so forth. Use the Default Processing unless you are tuning.
Write Interrupt Priority	Priority 0 (highest), Priority 1:2, Priority 3 (CM4: valid, CM0+: lowest- not valid if using ThreadX), Priority 4:14 (CM4: valid, CM0+: invalid), Priority 15 (CM4 lowest - not valid if using ThreadX, CM0+: invalid)	Write interrupt priority selection.

ISDE Property	Value	Description
	(Default: Disabled)	
Read Interrupt Priority	Priority 0 (highest), Priority 1:2, Priority 3 (CM4: valid, CM0+: lowest- not valid if using ThreadX), Priority 4:14 (CM4: valid, CM0+: invalid), Priority 15 (CM4 lowest - not valid if using ThreadX, CM0+: invalid) (Default: Disabled)	Read interrupt priority selection.
End Interrupt Priority	Priority 0 (highest), Priority 1:2, Priority 3 (CM4: valid, CM0+: lowest- not valid if using ThreadX), Priority 4:14 (CM4: valid, CM0+: invalid), Priority 15 (CM4 lowest - not valid if using ThreadX, CM0+: invalid) (Default: Disabled)	End interrupt priority selection.

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

Table 7. Configuration Settings for the DTC HAL Module on r_dtc EVENT CTSU END

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 to keep DTC vector table
Name	g_transfer0	Module name
Mode	Block	Mode selection
Transfer Size	2 Bytes	Transfer size selection
Destination Address Mode	Incremented	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	1	Number of transfers selection
Number of Blocks (Valid only in Block Mode)	1	Number of blocks selection
Activation Source (Must enable IRQ)	Event CTSU END	Activation source selection
Auto Enable	FALSE	Auto enable selection
Callback (Only valid with Software start)	NULL	Callback selection

ISDE Property	Value	Description
ELC Software Event Interrupt Priority	Priority 0 (highest), Priority 1:2, Priority 3 (CM4: valid, CM0+: lowest- not valid if using ThreadX), Priority 4:14 (CM4: valid, CM0+: invalid), Priority 15 (CM4 lowest - not valid if using ThreadX, CM0+: invalid) (Default: Disabled)	ELC software event interrupt priority selection

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

5.2 Capacitive Touch Button Framework Module Clock Configuration

The CTSU Button Framework uses the same configuration as the CTSU HAL driver: Set the CTSU clock speed to match the clock speed selected in the capacitive touch tuning tool Workbench6. The clock speed of the tuning project cannot go higher than 24 MHz (PCLKB). The clock settings for the PCLKB in the application can go higher than 24 MHz.

5.3 Capacitive Touch Button Framework Module Pin Configuration

The CTSU framework uses the same configuration as the CTSU HAL driver. Set the pins as touch sensor pins TSxx and enable the TSCAP function 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 CTSU pins.

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

Table 8. Pin Selection Sequence for Capacitive Touch Sensing Unit (CTSU)

Resource	ISDE Tab	Pin selection Sequence
CTSU	Pins	Select **Peripherals > Input: CTSU > CTSU0**

Table 9. Pin Configuration Settings for CTSU

Pin Configuration Property	Value	Description
Operation Mode	Disabled, Enabled (Default: Enabled)	Select enabled as the operation mode for CTSU
TS00 to TS11	None, Pn, Pm (Default: TS00: P204, TS01: P206, P207, None, P408, P409, TS06: None, None, None, None, TS10: P414, TS11: P415)	TS pins
TSCAP	None, Pn, Pm (Default: 205)	TSCAP pin

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

6. Using the Capacitive Touch Button Framework Module in an Application

Before using the Capacitive Touch Button framework, add the framework to your project in ISDE. Then make sure that the configuration data from CTW for Synergy is added to the project. Update the properties of the framework including the number of buttons, the name of the configuration structure generated by CTW for Synergy and the callback to be used. Once this is accomplished, the typical steps in using the Capacitive Touch Button Framework module in an application are:

1. Initialize the Capacitive Touch Button Framework module using the `sf_touch_ctsu_button_api_t::open` API
2. Detect and Handle Touch Button events with the Callback Function.
3. Disable the Callback for a Button using the `sf_touch_ctsu_button_api_t::disable` API (optional).
4. Enable the Callback for a Button using the `sf_touch_ctsu_button_api_t::enable` API (optional).
5. Close the Capacitive Touch Button framework module using the `sf_touch_ctsu_button_api_t::close` API.

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

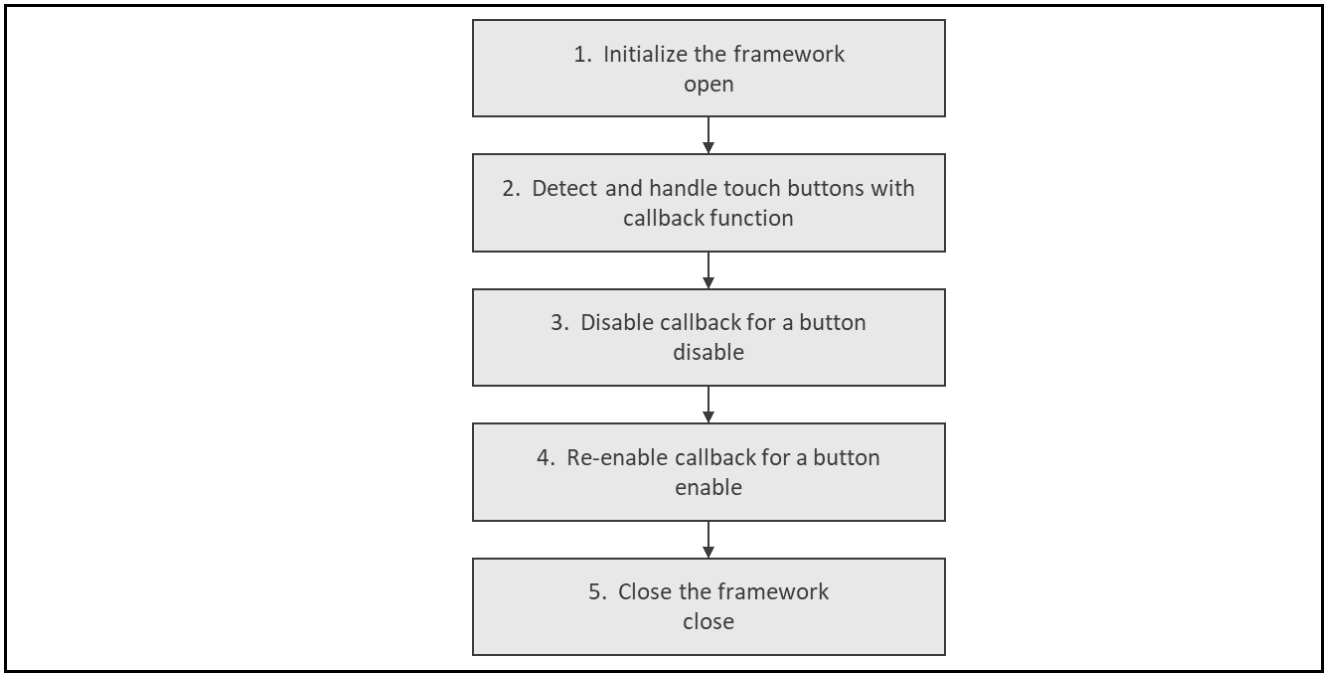

Figure 3. Typical Capacitive Touch Button Framework Module Application

7. Capacitive Touch Button Framework Module Application Project

The application project associated with this module guide demonstrates the aforementioned steps in a full design. The project can be found using the link provided in the Capacitive Touch Button Framework Module Reference Information section at the end of this document. You may want to import and open the application project within the ISDE and view the configuration settings for the Capacitive Touch Button Framework module. You can also read over the code (in `cap_touch_led_blinking.c`) used to illustrate the Capacitive Touch Button Framework APIs in a complete design.

The application project demonstrates the typical use of the Capacitive Touch Button Framework module APIs. The application project main thread entry initializes the structure containing information about onboard LEDs. Two Capacitive Touch buttons are used as the “next” and “previous” buttons and select which diode blinks. The following table identifies the target versions for the associated software and hardware used by the application project.

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

Resource	Revision	Description
e^2 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	v3.2 or later	Starter Kit

The following figure illustrates a simple flow diagram of the application project.

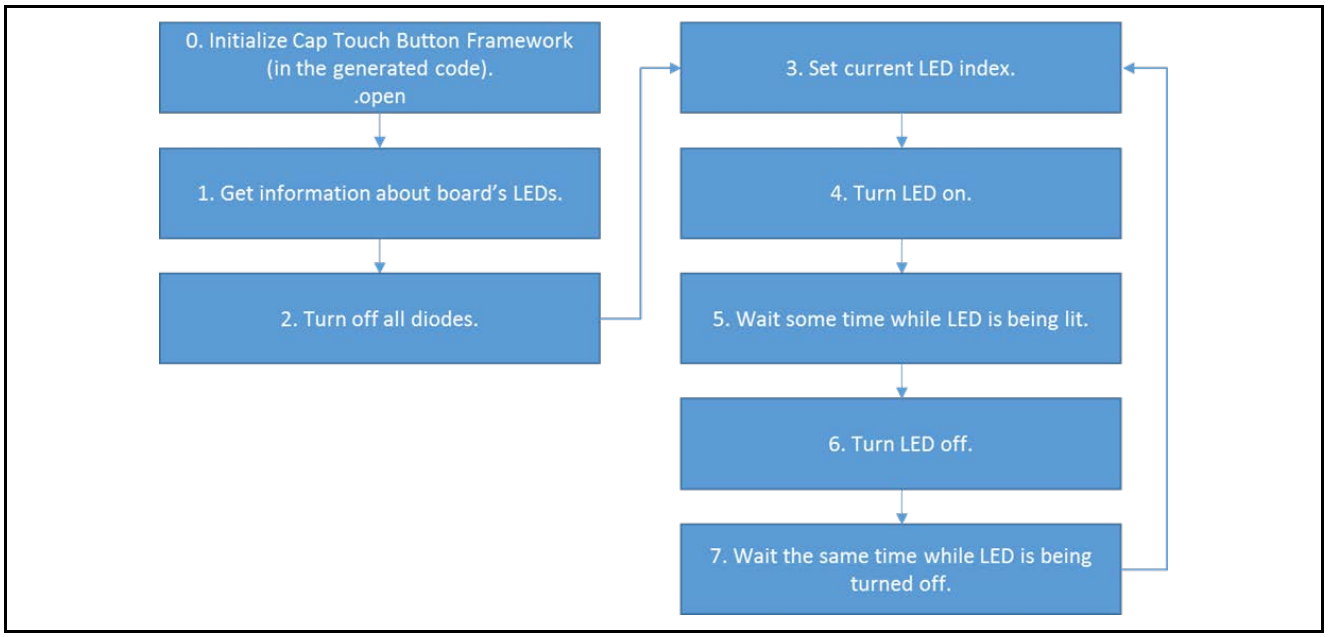

Figure 4. Capacitive Touch Button Framework Module Application Project Flow– Thread Entry

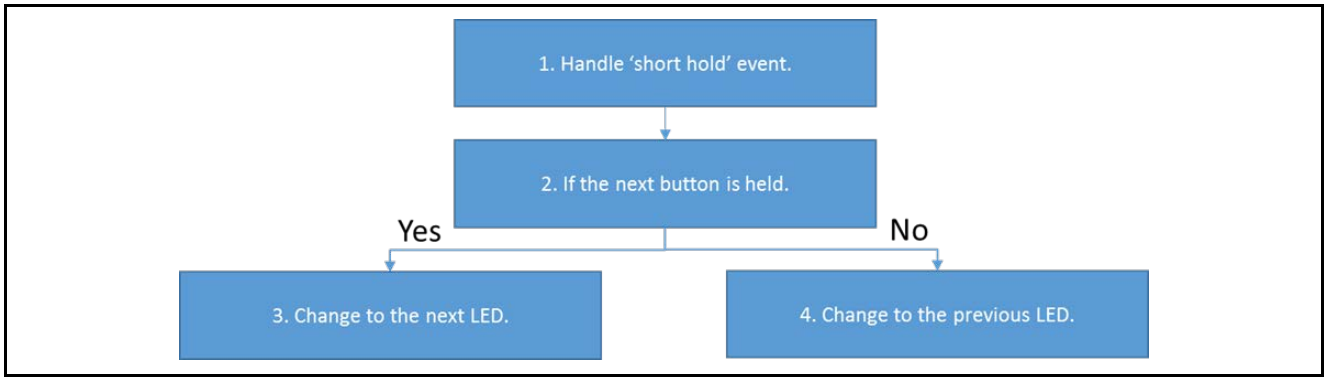

Figure 5. Capacitive Touch Button Framework Module Application Project Flow– Button Framework Callback

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

The first section of `capacitive_touch_led_blinking.c` has the header files which reference the Capacitive Touch Button Framework instance structure and a code section that contains macro and global variable definitions. The next section is the entry function for the main program control section. It acquires information about the board's LEDs and turns them off. The timer driver is opened and is used for counting down the time after which the LED state is toggled. Then, the thread sleep function suspends execution for an infinite time since all other operations are handled in the callback.

The next section is the Capacitive Touch Button Framework callback. This function handles "short hold" events only. It checks which button is held. If the "next" button (S1) is being pressed, then the index of the LED that is next driven is incremented. Otherwise it is decremented.

Note: The following section describes the method to generate Capacitive Touch button configurations in short description. However, for detailed instructions on how to generate Capacitive Touch Button configurations using Capacitive Touch Workbench, please refer the Tuning App project from the following link:

https://www.renesas.com/us/en/software/D6002773.html

However, the Application project bundled along with this App note already contains the Capacitive Touch Button configurations. So you can directly generate and build the project by skipping section 9.

To generate configuration files used by the CTSU HAL driver and the Capacitive Touch Button Framework, the Capacitive Touch Workbench for Synergy should be used. It runs several tests to adjust the parameters of the Capacitive Touch buttons used by the software.

1. It is important that the correct clock settings are used in the tuning project: PCLKB: 24 MHz.
2. First, import the tuning project for the SK-S7G2 board. This can be found in the CTW installation directory.
3. Open CTW and click the large blue **First Step Guide Start** button.
4. Carefully read the information and click **Next** button.
5. Select a path to the `Capacitive_Touch_FW_MG_AP` project and the previously imported tuning project. Click the **Next** button.

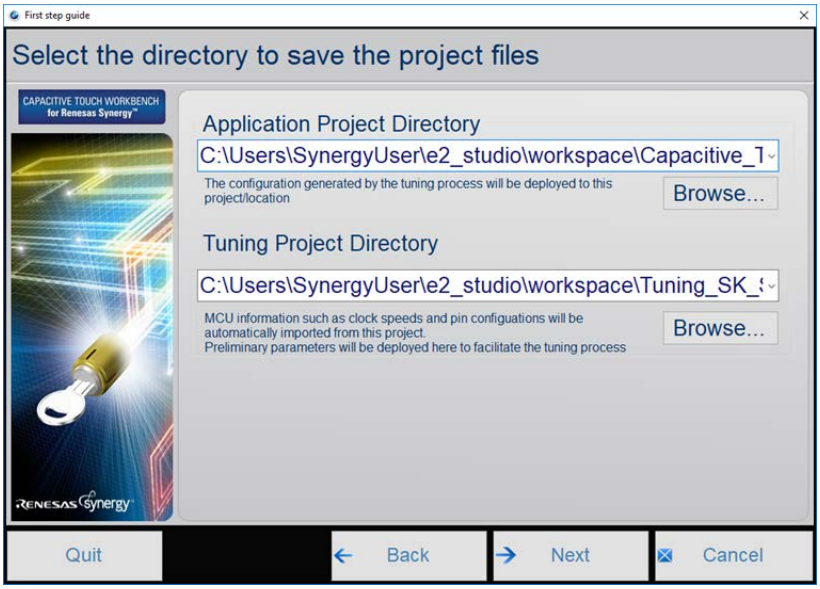

Figure 6. Selecting project directories in Capacitive Touch Workbench for Synergy

6. Choose the **Self capacitance method** and click the **Next** button.

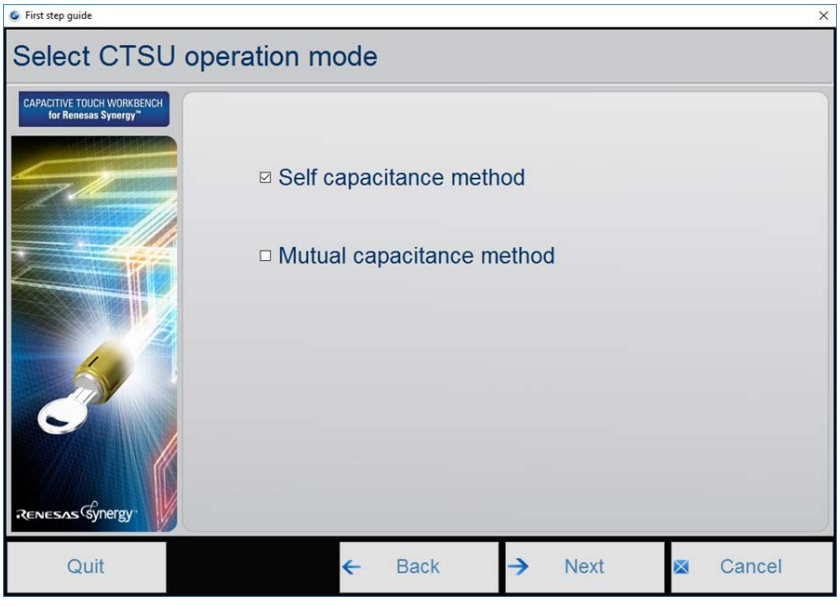

Figure 7. Selecting CTSU operation mode in Capacitive Touch Workbench for Synergy

7. Create a layout consisting of two buttons. Configure these widgets according to the following settings:
   - Button S1 – channel TS00.
   - Button S3 – channel TS01.

   Click the **Next** button.

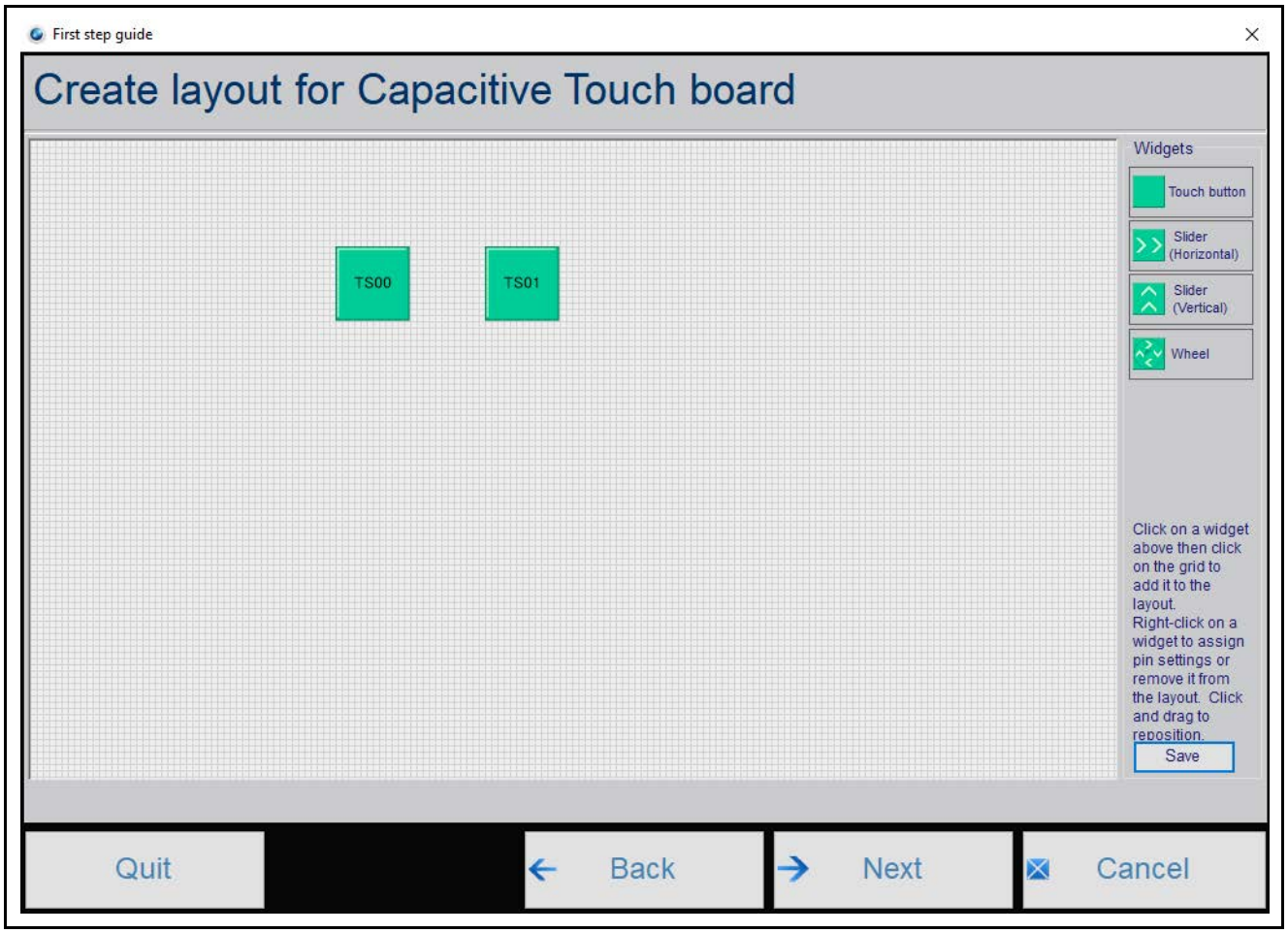

Figure 8. Creating layout in Capacitive Touch Workbench for Synergy

8. Set all TSn pins to 560 Ω and click the **Next** button.

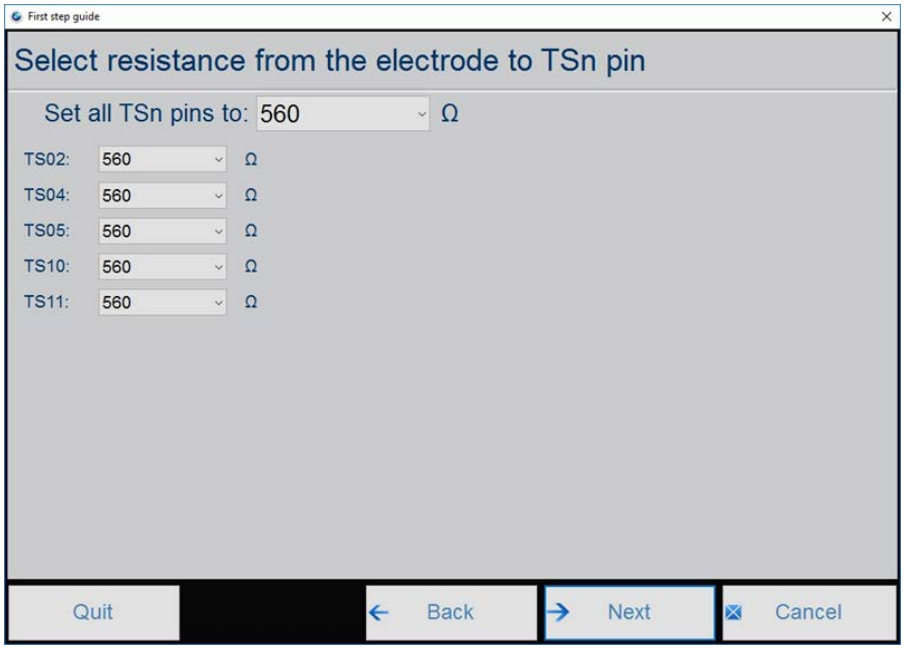

Figure 9. Selecting resistance to CTSU channels in Capacitive Touch Workbench for Synergy

9. Carefully read and follow instructions.

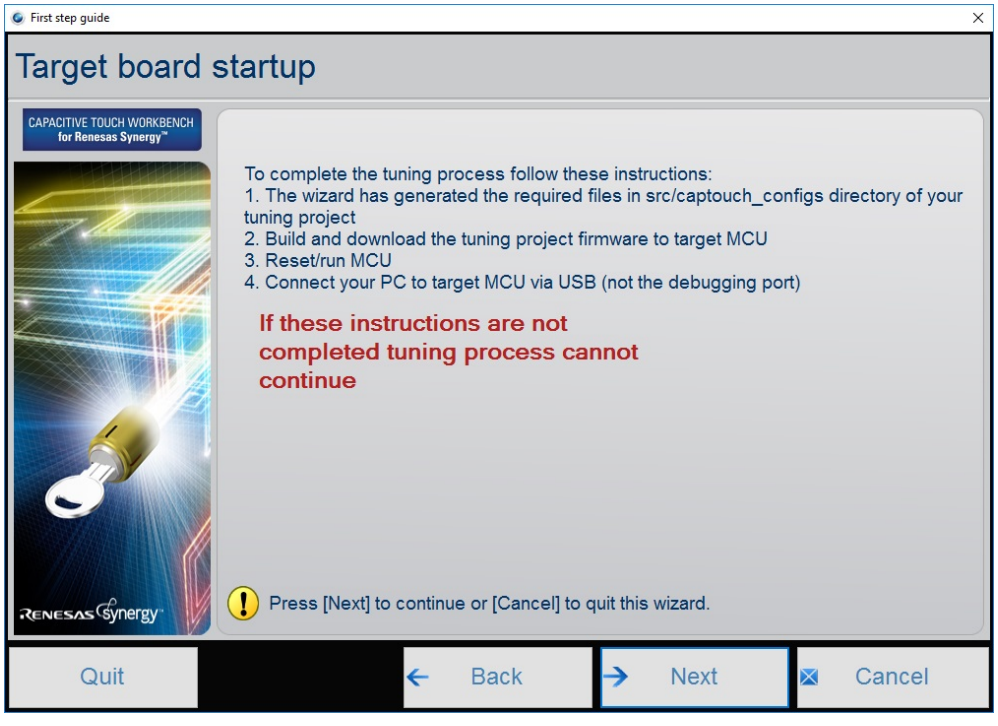

Figure 10. Preparing and running the tuning project in Capacitive Touch Workbench for Synergy

Note: Button configuration files generated by Capacitive Touch Workbench for Synergy have “short hold” and “long hold” events disabled by default. To enable the “short hold” event used by the application project, edit `src/captouch_configs/button_configs/self_button_config.c` and set **.event_enable.shorthold_enable** to true for **Self_Button_TS00** and **Self_Button_TS01** instances.

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. Below are the properties with the values set for this specific

project. You can also open the application project and view these settings in the Properties window as a hands-on exercise.

Table 11. Capacitive Touch Button Framework Module Configuration Settings for the Application Project

ISDE Property	Value Set
Parameter Checking	Default (BSP)
Number of Buttons	2
Short hold debounce multiplier	5
Long hold debounce multiplier	10
Stuck in debounce multiplier	10
Multi touch enable	Disabled
Enable stuck at condition detection	Disabled
Name	g_sf_touch_button
Button Configuration Structure Name (generated by Workbench)	touch_buttons
Callback	g_button_framework_user_callback

Table 12. Capacitive Touch Framework Module Configuration Settings for the Application Project

ISDE Property	Value Set
Parameter Checking	Default (BSP)
Name	g_sf_touch_ctsu
Thread Priority	3
Update Hz	50
Callback	NULL

Table 13.CTSU Driver Module Configuration Settings for the Application Project

ISDE Property	Value Set
Parameter Checking Enable	Default (BSP)
Offset Adjustment	Enabled
Drift Compensation	Enabled
Drift Compensation Method (valid only if Drift Compensation is enabled above)	Alternate method 1
Steady state drift compensation rate, drift compensation will be applied per n scans	500
Startup drift compensation rate (should be less than the steady state drift compensation rate)	5
Channel release compensation rate (should be less than the steady state drift compensation rate)	500
Default filter depth (used in sensor count filter provided by driver)	1
Runtime rate of tuning of sensor values (if drift compensation is used this value is overridden to be twice the rate of the steady state drift compensation)	800
Perform auto-tune and drift compensation only when all channels are untouched	True
Maximum active channels	5
Name	g_ctsu
CTSU configuration used	g_ctsu_config_self
Callback	NULL
Data Processing Option	Default Processing (Recommended)
Write Interrupt Priority	Priority 4 (CM4: valid, CM0+: invalid)
Read Interrupt Priority	Priority 4 (CM4: valid, CM0+: invalid)
End Interrupt Priority	Priority 4 (CM4: valid, CM0+: invalid)

Table 14.Transfer Driver Module Configuration Settings for the Application Project

ISDE Property	Value Set
Parameter Checking	Default (BSP)
Software Start	Disabled
Linker section to keep DTC vector table	.ssp_dtc_vector_table
Name	g_transfer
Mode	Block
Transfer Size	2 Bytes
Destination Address Mode	Incremented
Source Address Mode	Incremented
Repeat Area (unused in Normal mode)	Source
Interrupt Frequency	After all transfers have completed
Destination Pointer	NULL
Source Pointer	NULL
Number of Transfers	1
Number of Blocks (valid only in Block Mode)	1
Activation Source (must enable IRQ)	Event CTSU END
Auto Enable	FALSE
Callback (only valid with Software start)	NULL
ELC Software Event Interrupt Priority	Disabled

8. Customizing the Capacitive Touch Button Framework Module for a Target Application

Some configuration settings are normally changed by the developer from those shown in the application project. For example, the user can easily change the callback to handle more events such as the “long hold” event. The user can also change the debounce multipliers. These multipliers directly affect the time period after which “short hold” and “long hold” events get propagated.

9. Running the Capacitive Touch Button Framework Module Application Project

To run the Capacitive Touch Button Framework 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 *Renesas Synergy™ Project Import Guide* (11an0023eu0121-synergy-ssp-import-guide.pdf, included in this package) for instructions on importing the project into e^2 studio or IAR embedded workbench and building/running the application.

To implement the Capacitive Touch Button Framework 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 on how to accomplish these steps.

To create and run the Capacitive Touch Button Framework application project, simply follow these steps:

1. Create a new Renesas Synergy project for the SK-S7G2 board called **Capacitive_Touch_FW_MG_AP**.
2. Select the **Threads** tab.
3. Add a new thread as follows:
   - Symbol capacitive_touch_thread
   - Name Capacitive Touch Thread
4. Add the Capacitive Touch Button Framework to this thread.
5. Configure the blocks according to the above tables.
6. Click on the **Generate Project Content** button.
7. Run the tuning process using Capacitive Touch Workbench for Synergy.
8. Edit the file generated by Capacitive Touch Workbench `src/captouch_configs/button_configs/self_button_config.c` and set **.event_enable.shorthold_enable** to true for **Self_Button_TS00** and **Self_Button_TS01** instances.
9. Add the code from supplied project files `capacitive_touch_thread_entry.c`, `capacitive_touch_led_blinking.h`, `capacitive_touch_led_blinking.c`, or copy over the generated files, `capacitive_touch_thread_entry.c`, `capacitive_touch_led_blinking.h`, and `capacitive_touch_led_blinking.c`.
10. Connect to the host PC through a micro USB cable to J19 on the SK-S7G2 board.
11. Start to debug the application.
12. The LED should start blinking. The user can select the next or the previous LED by pressing and holding **S1** or **S3** buttons.

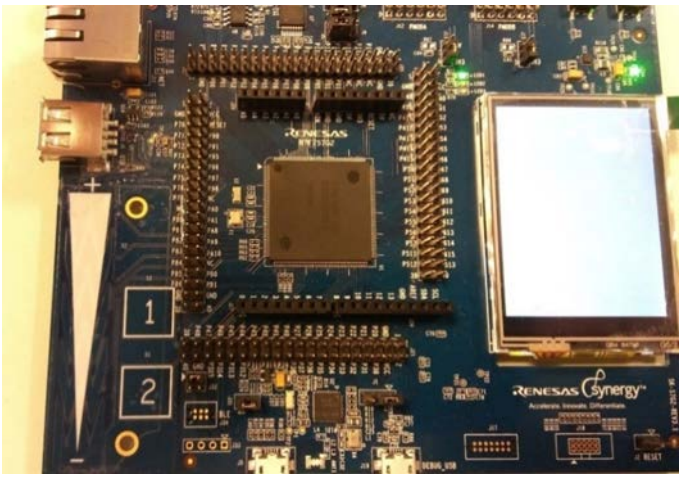

Figure 11. Example Output from the Capacitive Touch Button Framework Application Project

10. Capacitive Touch Button Framework 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 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. Capacitive Touch Button Framework Module Next Steps

After you have mastered a simple Capacitive Touch Button Framework module project you may want to review a more complex example. Other application projects and application notes that demonstrate Capacitive Touch Button Framework use can be found as described in the reference section at the end of this document.

To evaluate a more complex Capacitive Touch solution, see the AE-CAP1 Application Example for Capacitive Touch. More information can be found on the official Renesas website.

12. Capacitive Touch Button Framework 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 sf_touch_ctsu_button 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/SF_Touch_CTSU_Button_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	Sep.11.17		Initial version
1.01	Feb.04.19		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/.