Renesas Synergy™ Platform

Capacitive Touch Slider 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 References section at the end of this document) and should be valuable resources for creating more complex designs.

This module is a high-level ThreadX®-aware API for Capacitive Touch slider and wheel applications and is implemented on `sf_touch_ctsu_slider`. The module uses the Capacitive Touch Sensing Unit (CTSU) on the Synergy MCU. The framework is designed to be used with the configuration data generated by the Capacitive Touch Workbench for Renesas Synergy™ (CTW for Synergy) tool. In the tool, the sliders, wheels, and channels used are configured. A user-defined callback can be set up to process data available from the hardware after a finger scan occurs.

Contents

1. Capacitive Touch Slider Framework Module Features

The Capacitive Touch Slider Framework module is used to interpret data from the Capacitive Touch Sensing Unit (CTSU) for all the slider configurations initialized by the system. Key features available include:

- Support for slider and wheel
- Support for multiple instances of sliders and wheels
- Callbacks are used to simplify touch processing
  - Use configuration data generated from CTW
  - When a state changes a callback is generated
  - Callbacks are associated with each slider and include the event and position
  - Callbacks are called in the order they appear in the configuration table
- Supports multi-touch detection (can be optionally disabled at build-time)

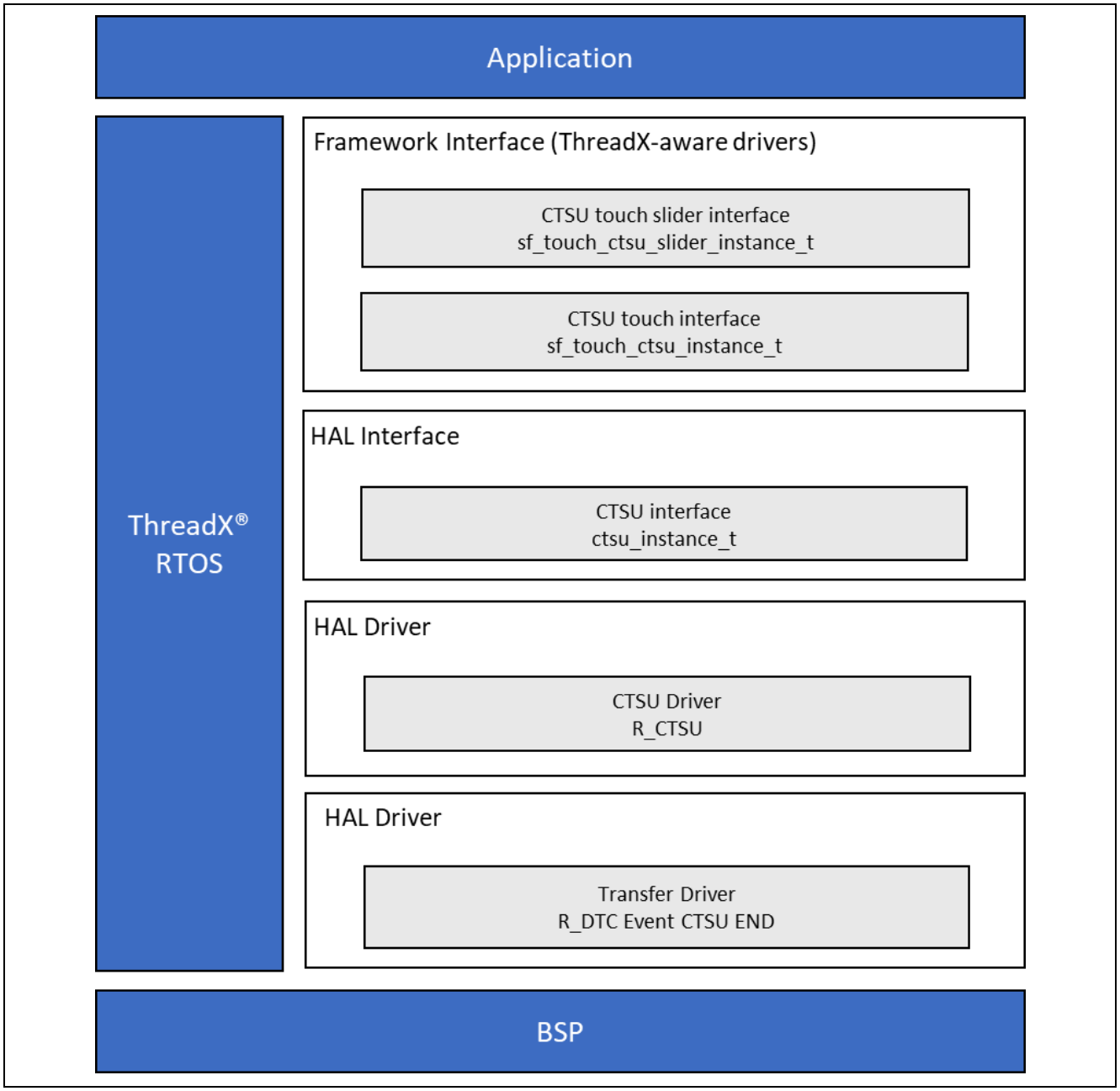

Figure 1. Capacitive Touch Slider Framework Module Block Diagram

2. Capacitive Touch Slider Framework Module APIs Overview

The Capacitive Touch Slider Framework defines APIs for open, close, enable and disable. A complete list of the available APIs, an example API call and a brief 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 Slider Framework Module API Summary

Function Name	Example API Call and Definition
.open	`g_sf_touch_slider0.p_api->open( g_sf_touch_slider0.p_ctrl, g_sf_touch_slider0.p_cfg);` Initialize the Touch CTSU Button Framework, configure the lower level hardware and registers callback functions for all the buttons.
.enable	`g_sf_touch_slider0.p_api->enable( g_sf_touch_slider0.p_ctrl, slider_id);` Enable callback notification for a configured button.
.disable	`g_sf_touch_slider0.p_api->disable( g_sf_touch_slider0.p_ctrl, slider_id);` Disable callback notification for a configured button.
.close	`g_sf_touch_slider0.p_api->close( g_sf_touch_slider0.p_ctrl);` Close the Touch CTSU Button Framework, close the button framework and lower layers if no other modules are using it.
.versionGet	`g_sf_touch_slider0.p_api->versionGet( &p_version);` Get version and store it in 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	Framework already in use.
SSP_ERR_NOT_OPEN	Device not open.
SSP_ERR_INTERNAL	Internal error.
SSP_ERR_UNSUPPORTED	Device unsupported.
SSP_ERR_INVALID_ARGUMENT	Invalid argument.

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 Slider Framework Module Operational Overview

The Capacitive Touch Slider Framework module is used to interpret the CTSU data for all the slider configurations initialized by the system. The module also initializes the CTSU Framework layer. The Capacitive Touch Slider Framework registers a callback with the CTSU Framework layer, which is called each time processed data is available. The Slider Framework then uses this data (raw values) to determine if a touch or release occurred and, if so, where it occurred. If there is a state change, the framework calls the callback for each slider in the order in which they are present in the slider configuration table, along with the event and the position. The slider framework executes the callback at the update rate (sf_touch_ctsu configuration update_hz) between touch and release events. The application code should use these callbacks to track the position along the slider. The slider framework executes the callback at the update rate sf_touch_ctsu_cfg_t::update_hz) between the touch and release events. This feature can be optionally disabled at build-time.

3.1 Capacitive Touch Slider Framework Module Important Operational Notes and Limitations

3.1.1 Capacitive Touch Slider Framework Module Operational Notes

The framework is designed to be used with the configuration data generated by the CTW for Synergy tool. For details on how to use the tool to generate configuration data, see the Capacitive Touch application notes on the Renesas Synergy website.

3.1.2 Capacitive Touch Slider Framework Module Limitations

The framework only supports capacitive touch sliders/wheels that are laid out for self-capacitance mode of operation. There are no other known limitations for using this module. See the latest SSP release notes for additional limitations.

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

This section describes how to include the Capacitive Touch Slider Framework 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 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 Slider Framework to an application, simply add it to a thread using the stacks selection sequence given in the following table. (The default name for the Capacitive Touch Slider Framework is `sf_touch_ctsu_slider`. This name can be changed in the associated **Properties** window.)

Table 3. Capacitive Touch Slider Framework Module Selection Sequence

Resource	ISDE Tab	Stacks Selection Sequence
g_sf_touch_slider0 Cap Touch Slider/Wheel Framework on sf_touch_ctsu_slider	Threads	New Stack> Framework> Input> Cap Touch Slider Framework on sf_touch_ctsu_slider

When the Capacitive Touch Slider Framework on `sf_touch_ctsu_slider` 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 box 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.

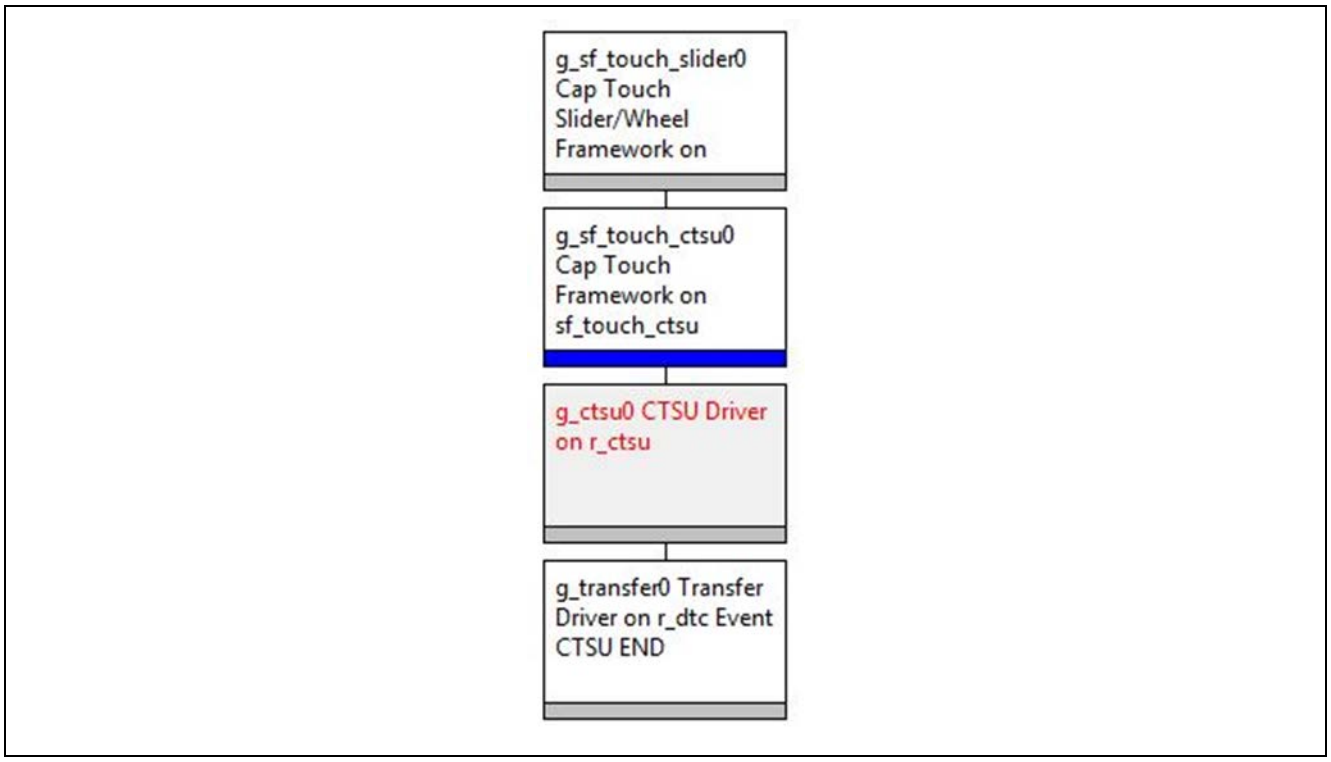

Figure 2. Capacitive Touch Slider Framework Module Stack

5. Configuring the Capacitive Touch Slider Framework Module

The Capacitive Touch Slider 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 includes 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 Slider Framework Module on sf_touch_ctsu_slider

Parameter	Value	Description
Parameter Checking	BSP, Enabled, Disabled (Default: BSP)	Controls whether to include code for API parameter checking
Number of Sliders/Wheels	1	Number of sliders configured in CTW for Synergy
Multi touch enable	Enabled, Disabled Default: Enabled	Enables or disables multi-touch detection
Name	g_sf_touch_slider0	Framework name
Slider/Wheel Configuration Structure Name (generated by CTW for Synergy)	all_sliders	Name of slider configuration structure generated by CTW for Synergy
Callback	g_slider_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. For example, it might be useful to set the maximum number of monitored threads to a desired value. 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 Slider Framework Low Level Modules

Typically, only a small number of settings must be modified from the default for lower-level modules as indicated via 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 determines 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. Thus, if 10 channels are used for 10 buttons in self-capacitance mode, the total hardware scan time is ~6 milliseconds 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 milliseconds 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 less than the time required to scan and process all the channels used in the configuration and lesser than the RTOS tick rate.

ISDE Property	Value	Description
Callback	NULL	Name of user callback that will be 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 will be 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
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. ATTENTION: Since the callback is called from an

ISDE Property	Value	Description
		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_ctsu0_config	Name of the configuration structure generated by CTW.
Callback	NULL	The user callback function that will be 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 etc. 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) (Default: Disabled)	Write interrupt priority selection.
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: Enabled	Software start selection
Linker section to keep DTC vector table	.ssp_dtc_vector_table	Linker section to keep DTC vector table

ISDE Property	Value	Description
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
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 Slider Framework Module Clock Configuration

The CTSU Slider 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 Workbench 6.

5.3 Capacitive Touch Slider 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 first table below illustrates the method for selecting the pins within the SSP configuration window and the following table illustrates an example selection for the CTSU pins.

Note: The operation mode selected determines the peripheral signals available and the required MCU pins.

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

Resource	ISDE Tab	Pin selection Sequence
CTSU	Pins	Select **Peripherals > CTSU> Cap_Touch_Sensing_Unit**

Table 9. Pin Configuration Settings for CTSU

Property	Value	Description
Operation Mode	Disabled, Enabled (Default: Disabled)	Select Enabled as the Operation Mode for CTSU
TS00 to TS11	None, Pn, Pm (Default: None)	TS Pins

Property	Value	Description
TSCAP	None, Pn, Pm (Default: None)	TSCAP Pin

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

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

Before using the Capacitive Touch Slider framework, add the framework to your project in ISDE. Ensure the configuration data from CTW for Synergy is added to the project and be sure the update to the framework properties includes the number of sliders, the name of the configuration structure generated by CTW for Synergy, and the callback to be used. Once this is done, the typical steps to using the Capacitive Touch Slider Framework module in an application involve:

1. Initializing the Capacitive Touch Slider Framework module using the sf_touch_ctsu_slider_api_t::`open` API
2. Detecting and handling Touch Slider events through the Callback Function.
3. Disabling Callback for the Slider with the sf_touch_ctsu_slider_api_t::`disable` API (Optional).
4. Enabling Callback for the Slider with the sf_touch_ctsu_slider_api_t::`enable` API (Optional).
5. Closing Capacitive Touch Button Framework module using the sf_touch_ctsu_slider_api_t::`close` API.

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

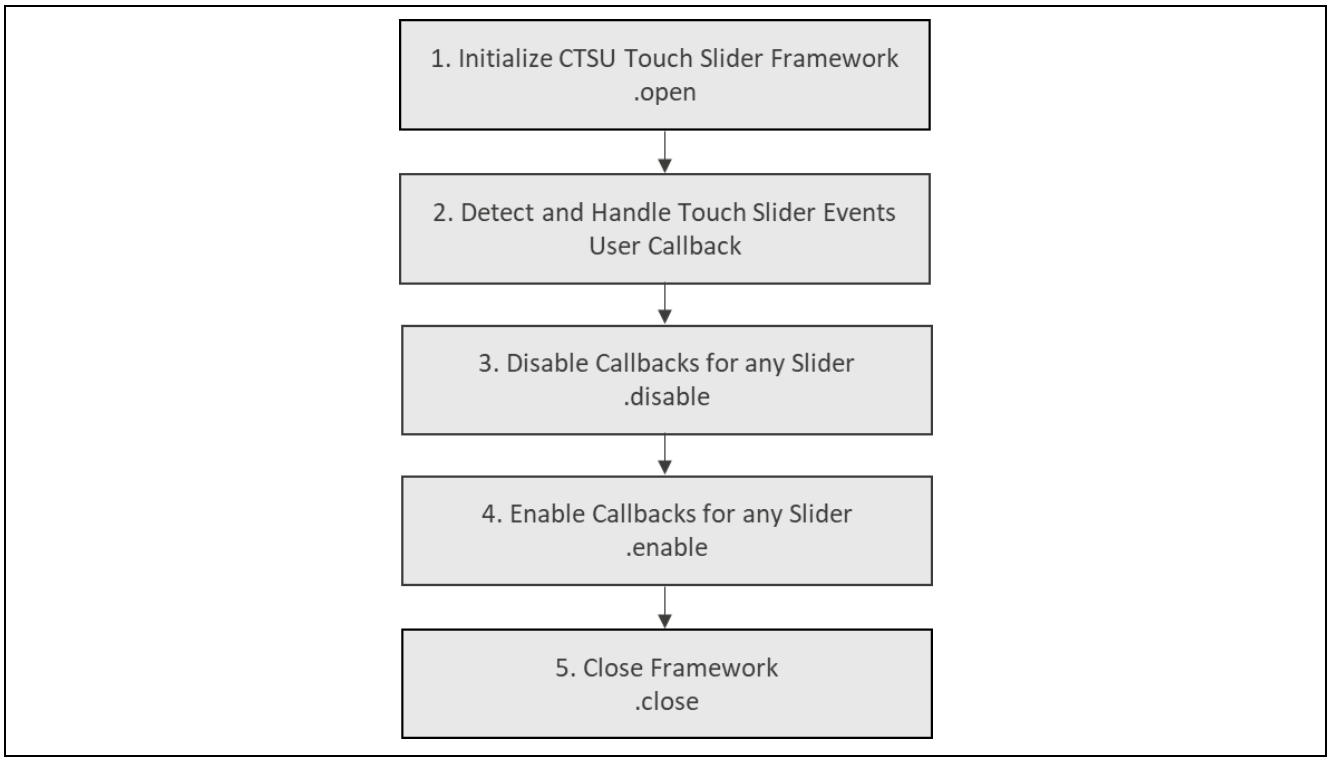

Figure 3. Typical Capacitive Touch Slider Framework Module Application

7. The Capacitive Touch Slider Framework Module Application Project

The application project associated with this module guide demonstrates the steps typical in a full design. The project can be found using the link provided in the References 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 Slider Framework module. You can also read over the code (in `capacitive_touch_led_blinking.c`, `capacitive_touch_led_blinking.h`, and `capacitive_touch_thread_entry.c`) used to show the Capacitive Touch Slider Framework APIs in a complete design.

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	1.5.0 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:

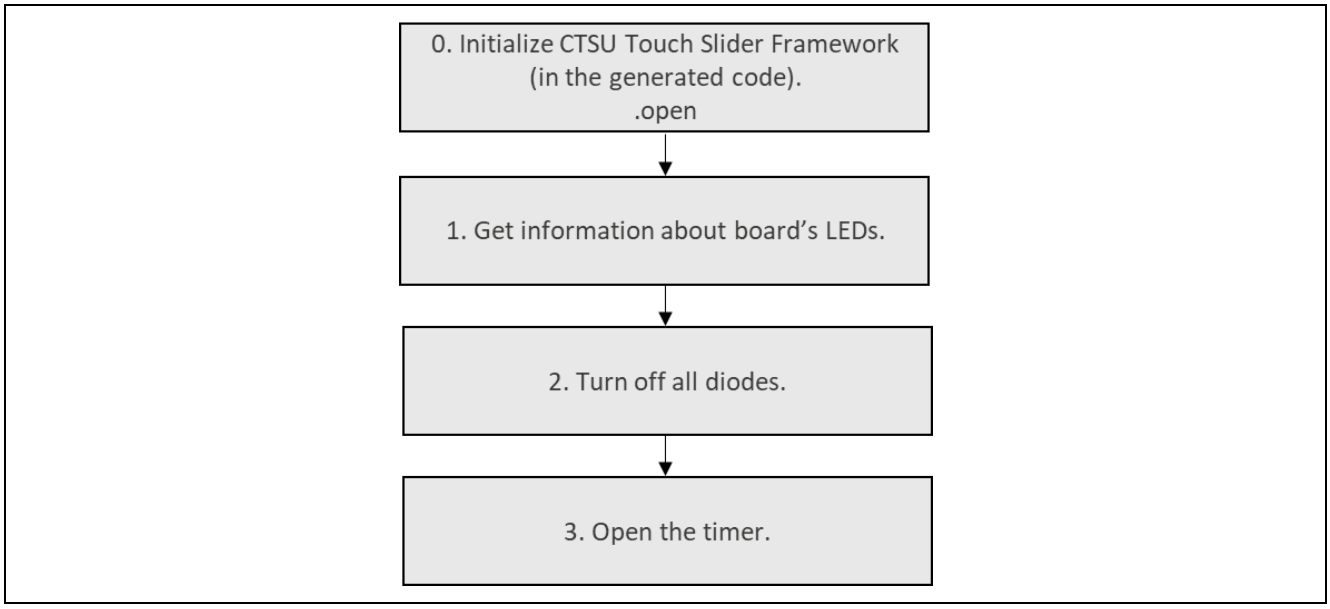

Figure 4. Capacitive Touch Slider Framework Module Application Project Flow Diagram – Thread Entry

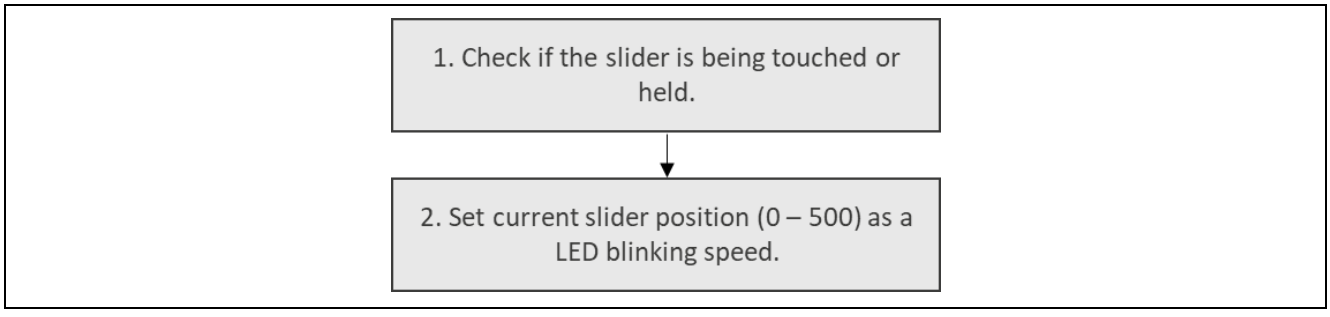

Figure 5. Capacitive Touch Slider Framework Module Application Project Flow Diagram – The Slider Framework Callback

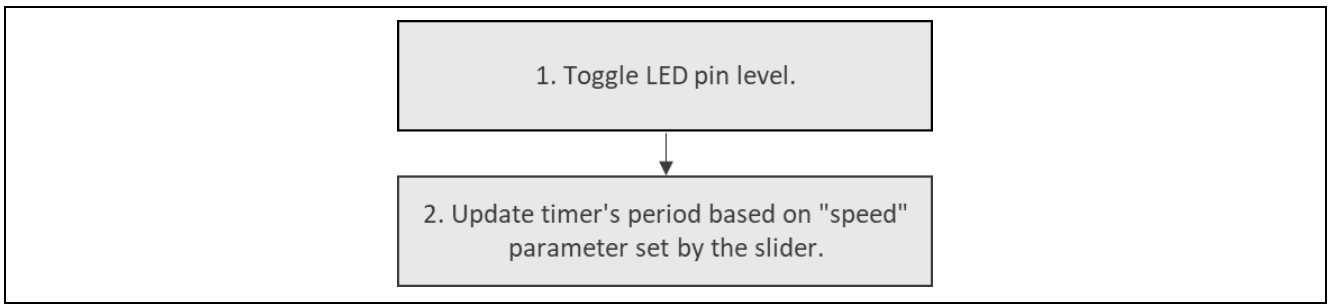

Figure 6. Capacitive Touch Slider Framework Module Application Project Flow Diagram - Timer Callback

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 header files referencing the Capacitive Touch Slider Framework instance structure, and a code section containing macro and global variable definitions. The next section has the entry function for the main program control section. It acquires information about the board's LEDs and turns the LEDs off. The timer driver is opened, it is used to count down the time, after which the LED state is toggled. Next, 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 Slider Framework callback. This function handles "slider touched" and "slider held" events. If one of these events occurs, the current slider position is stored as a LED blinking "speed" parameter.

The last section is the timer callback function. This function toggles the state of the LED and updates the timer period according to the "speed" parameter. This directly affects the LED blinking time.

To generate configuration files used by the CTSU HAL driver and the Capacitive Touch Slider Framework, CTW for Synergy should be used. The tool runs several tests to adjust the Capacitive Touch slider parameters used by the software.

1. Begin by importing the tuning project for SK-S7G2. This project can be found in CTW for Synergy installation directory.
2. Open CTW for Synergy and click large blue First Step Guide Start button.
3. Carefully read the information provided and click **Next**.
4. Select a path to `Capacitive_Touch_Slider_FW_MG_AP` project and the previously imported tuning project. Click **Next**.

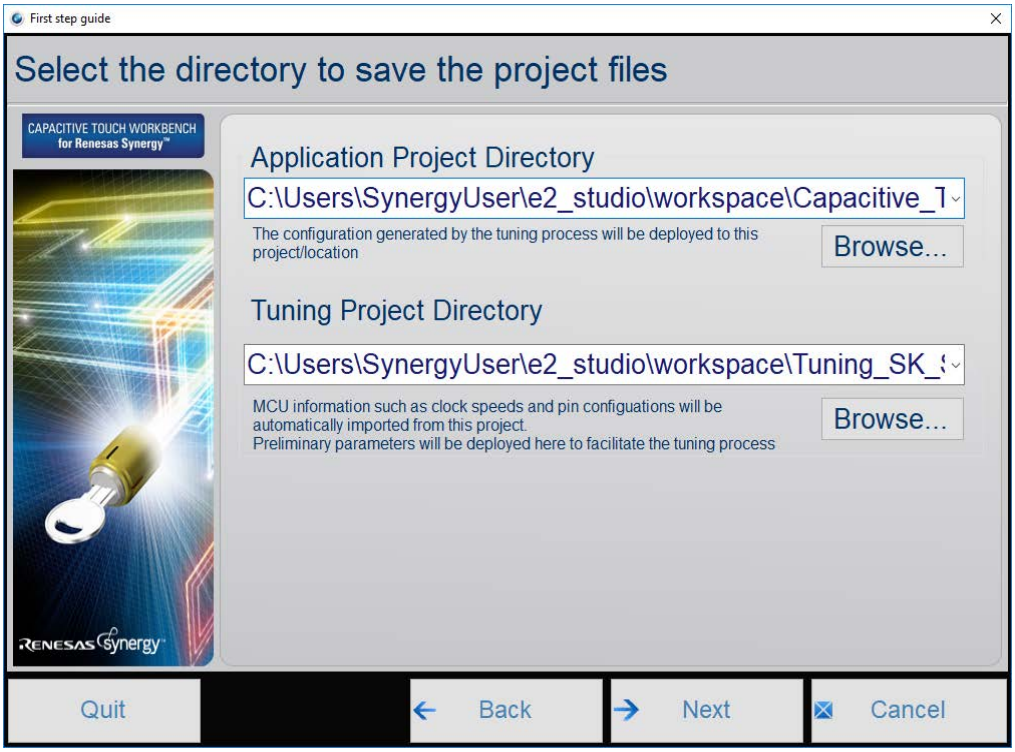

Figure 7. Selecting project directories in CTW for Synergy

5. Select box for the self-capacitance method, then click **Next**.

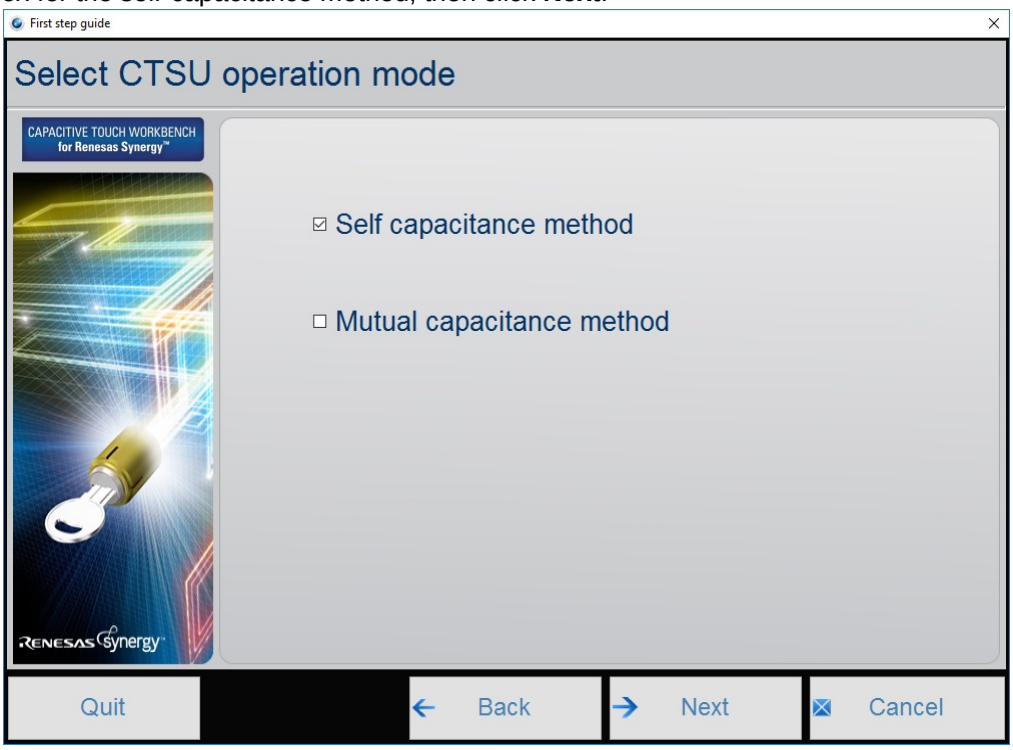

Figure 8. Selecting CTSU operation mode in Capacitive Touch Workbench for Synergy

6. Create layout consisting of one horizontal slider.
7. Configure this widget according to the following settings. Use Right click > **Setup**…

   Number of channels: 5
   Slider TS0: TS02
   Slider TS1: TS04
   Slider TS2: TS05
   Slider TS3: TS10
   Slider TS4: TS11

8. Click **Next**.

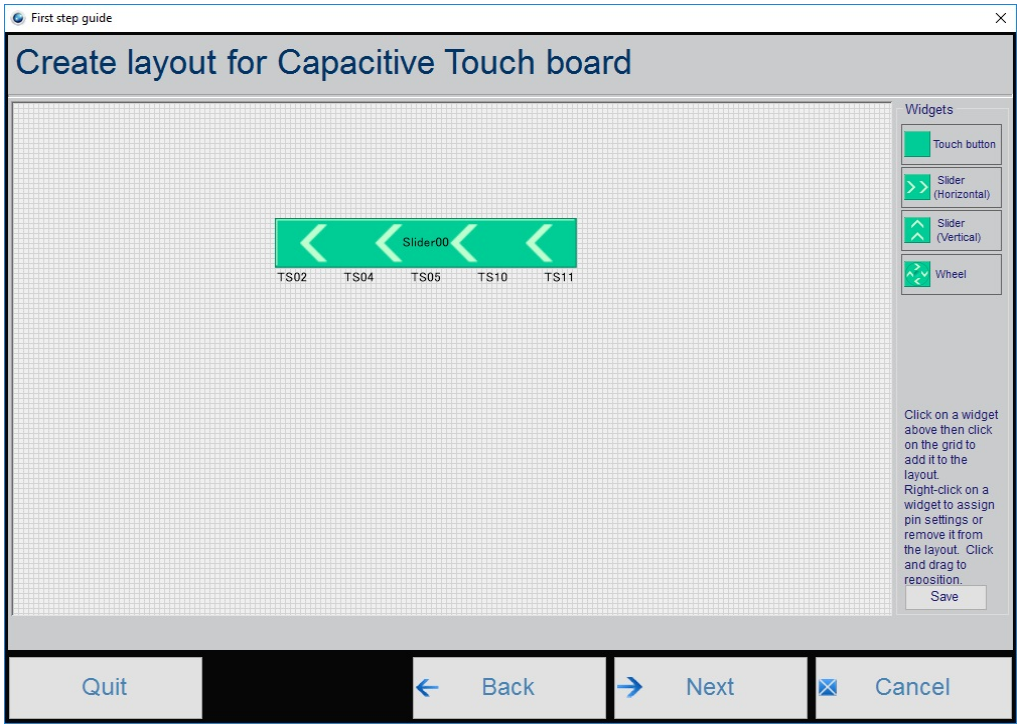

Figure 9. Creating layout in Capacitive Touch Workbench for Synergy

9. Set all TSn pins to 560 Ω and click **Next**.

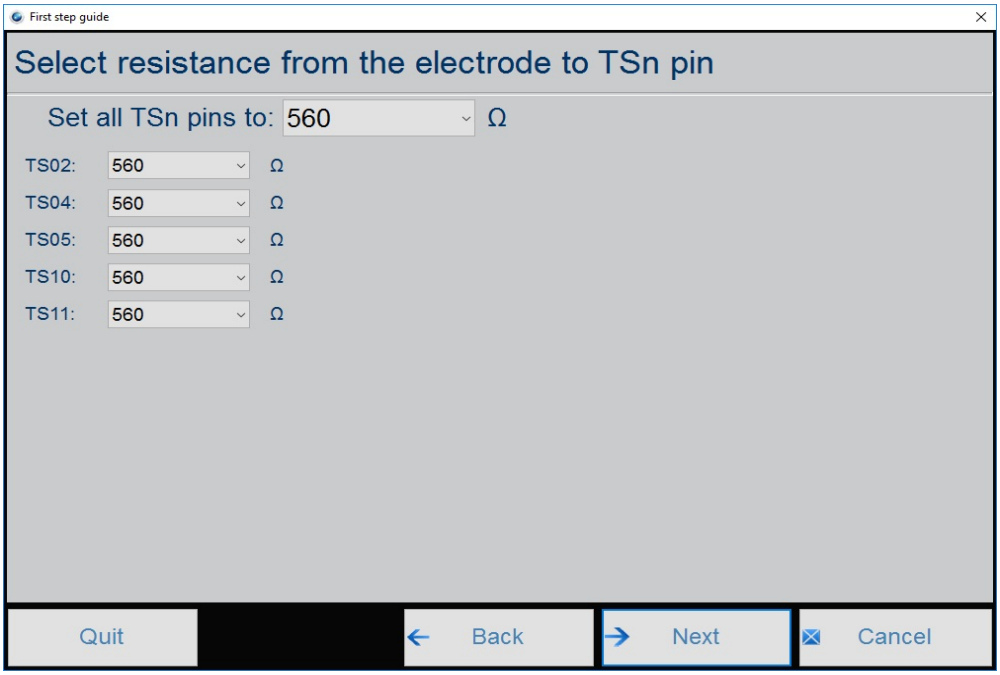

Figure 10. Selecting resistance to CTSU channels in Capacitive Touch Workbench for Synergy

10. Carefully read and follow further instructions.

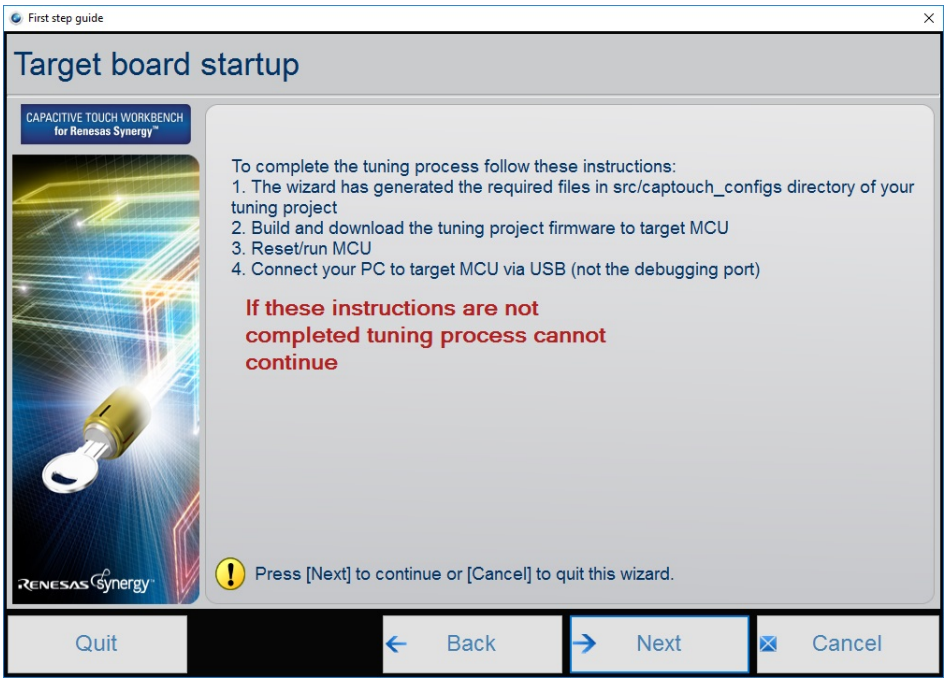

Figure 11. Preparing and running the tuning project in Capacitive Touch Workbench for Synergy

A few key properties are configured in this application project. These key properties support required operations and the physical properties of the target board and the MCU.

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

Table 11. Capacitive Touch Slider Framework Module Configuration Settings

ISDE Property	Value Set
Parameter Checking	Default (BSP)
Number of Sliders/Wheels	1
Multi touch enable	Disabled
Name	g_sf_touch_slider
Slider/Wheel Configuration Structure Name (generated by Workbench)	all_sliders
Callback	g_slider_framework_user_callback

Table 12. Capacitive Touch Framework Module Configuration Settings

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

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
Max. 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

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

Table 15. The Timer Driver Module Configuration Settings for the Application Project

ISDE Property	Value Set
Parameter Checking	Default (BSP)
Name	g_timer
Channel	0
Mode	Periodic
Period Value	5
Period Unit	Hertz
Duty Cycle Value	50
Duty Cycle Unit	Unit Raw Counts
Auto Start	True
GTIOCA Output Enabled	False
GTIOCA Stop Level	Pin Level Low
GTIOCB Output Enabled	False
GTIOCB Stop Level	Pin Level Low
Callback	g_timer_user_callback
Interrupt Priority	Priority 2

8. Customizing the Capacitive Touch Slider 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 configuration settings for the CTSU HAL Driver. The user can also change the CTSU port pins to select the desired channels. This can be done using the “Pins” tab in the configurator. The Capacitive Touch Slider Framework Module application project used channels 2, 4, 5, 10 and 11, which are connected to the onboard capacitive touch slider.

9. Running the Capacitive Touch Slider Framework Module Application Project

To run the Capacitive Touch Slider 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* (r11an0023eu0121-synergy-ssp-import-guide.pdf), included in this package, for instructions on importing the project into e^2 studio ISDE or the IAR EW for Synergy, and then building/running the application.

To implement the Capacitive Touch Slider 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 SSP more practical, while just reading over this guide tends to be more theoretical.

Note: These instructions are given in sufficient detail for someone experienced with the basic flow through the Synergy development process. If these steps are not familiar, read the first few chapters of the *SSP User’s Manual*.

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

1. Create a new Renesas Synergy project for the S7G2-SK called **Capacitive_Touch_Slider_FW_MG_AP**.
2. Select the **Threads** tab.
3. Add a new thread called:
   A. Symbol: **capacitive_touch_thread**
   B. Name: **Capacitive Touch Thread**
4. Add the **Capacitive Touch Slider Framework** to this thread.
5. Add the Timer Driver on **r_gpt** to this thread.
6. Configure the blocks according to the above tables.
7. Click **Generate Project Content**.
8. Run the tuning process using Capacitive Touch Workbench for Synergy.

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 generated `capacitive_touch_thread_entry.c`, `capacitive_touch_led_blinking.h`, `capacitive_touch_led_blinking.c` files.
10. Connect to the host PC via a micro USB cable to J19 on SK-S7G2.
11. Start to debug the application.
    As shown the green LED should start blinking. The user can adjust the period using the capacitive slider.

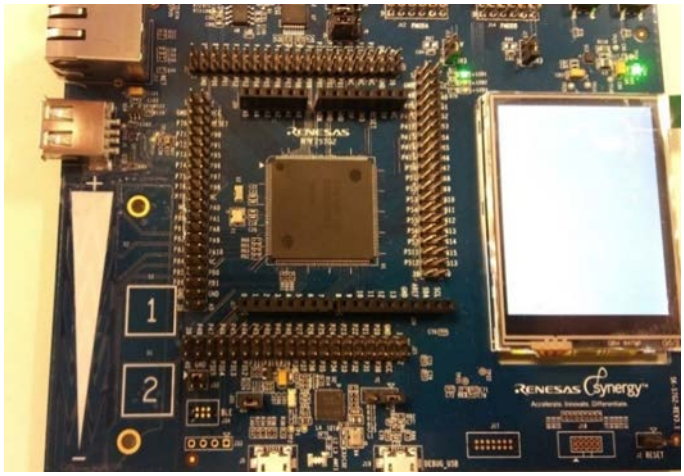

Figure 12. Example Output from the Capacitive Touch Slider Framework Application Project

10. Capacitive Touch Slider 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) provides additional development-time savings 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. Capacitive Touch Slider Framework Module Next Steps

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

If you wish to evaluate more complex Capacitive Touch solution, you might be interested in AE-CAP1 Application Example for Capacitive Touch. More information may be found at the Renesas website: www.renesas.com/us/en/products/synergy/hardware/kits/ae-cap1.html.

12. Capacitive Touch Slider Framework 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 sf_touch_ctsu_slider module reference materials and resources are available on the Synergy Knowledge Base: https://en-support.renesas.com/knowledgeBase/16977556.

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

		Description	
Rev.	**Date**	**Page**	**Summary**
1.00	Sep.13.17	—	Initial release
1.01	Feb.08.19	—	Updates to hardware and software configuration

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/.