Renesas Synergy™ Platform

ADC 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 ADC HAL module is an API for analog-to-digital conversions and is implemented on r_adc. It supports both the ADC12 and ADC14 peripherals available on the Synergy microcontroller hardware; a user-defined callback can be created to process the data each time a new sample is complete.

Contents

1. ADC HAL Module Features 2
2. ADC HAL Module APIs Overview 2
3. ADC HAL Module Operational Overview 3
3.1 ADC HAL Module Important Operational Notes and Limitations 4
3.1.1 ADC HAL Module Operational Notes 4
3.1.2 ADC HAL Module Limitations 5
4. Including the ADC HAL Module in an Application 5
5. Configuring the ADC HAL Module 6
5.1 ADC HAL Module Clock Configuration 9
5.2 ADC HAL Module Pin Configuration 9
6. Using the ADC HAL Module in an Application 9
7. The ADC HAL Module Application Project 10
8. Customizing the ADC HAL Module for a Target Application 12
9. Running the ADC HAL Module Application Project 12
10. ADC HAL Module Conclusion 13
11. ADC HAL Module Next Steps 13
12. ADC HAL Module Reference Information 13

1. ADC HAL Module Features

- 14-Bit A/D Converter (S3A7, S124) or 12-Bit A/D Converter (S7G2)
- Multiple Operation Modes
  - Single Scan
  - Group Scan
  - Continuous Scan
- Multiple Channels
  - 13 channels (unit 0) or 12 channels (unit 1) for S7G2
  - 18 channels for S124
  - 28 channels for S3A7

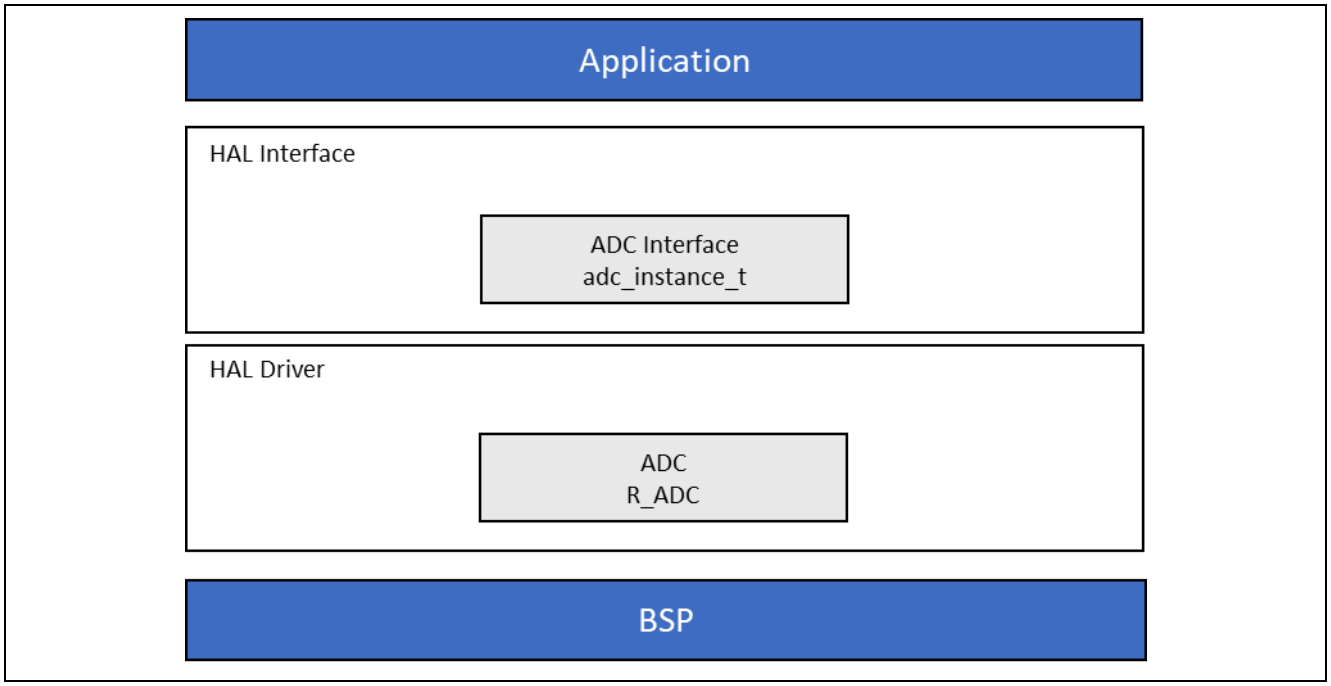

Figure 1. ADC HAL Module Block Diagram

2. ADC HAL Module APIs Overview

The ADC HAL module defines APIs to open, configure scans, start scans, stop scans, read the conversion results the ADC scans, and close the ADC unit. 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. ADC HAL Module API Summary

Function Name	Example API Call and Description
.open	`g_adc.p_api->open(g_adc.p_ctrl, g_adc.p_cfg);` Initialize ADC Unit; apply power, set the operational mode, trigger sources, interrupt priority, and configurations common to all channels and sensors.
.scanCfg	`g_adc.p_api->scanCfg(g_adc.p_ctrl, g_adc.p_channel_cfg);` Configure the scan including the channels, groups and scan triggers to be used for the unit that was initialized in the open call.
.scanStart	`g_adc.p_api->scanStart(g_adc.p_ctrl);` Start the scan (in case of a software trigger), or enable the hardware trigger.
.scanStop	`g_adc.p_api->scanStop(g_adc.p_ctrl);` Stop the ADC scan (in case of a software trigger), or disable the hardware trigger.
.scanStatusGet	`g_adc.p_api->scanStatusGet(g_adc.p_ctrl);` Check scan status.

Function Name	Example API Call and Description
.read	`g_adc.p_api->read(g_adc.p_ctrl, ADC_REG_CHANNEL_13, &adc_data);` Read ADC conversion result(s).
.sampleStateCountSet	`g_adc.p_api->sampleStateCountSet(g_adc.p_ctrl,&adc_sample);` Set the sample state count for the specified channel.
.close	`g_adc.p_api->close(g_adc.p_ctrl);` Close the specified ADC unit by ending any scan in progress, disabling interrupts, and removing power to the specified A/D unit.
.infoGet	`g_adc.p_api->infoGet(g_adc.p_ctrl, &adc_info);` Return the ADC data register address of the first (lowest number) channel and the total number of bytes to be read for the DTC/DMAC to read the conversion results of all configured channels.
.versionGet	`g_adc.p_api->versionGet(&version);` Retrieve the API version with the version pointer.

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

Table 2. Status Return Values

Name	Description
SSP_SUCCESS	API Call Successful
SSP_ERR_INVALID_ARGUMENT	Parameter has invalid value.
SSP_ERR_NOT_OPEN	Unit is not open
SSP_ERR_ASSERTION	The parameter p_ctrl or p_sample is NULL.
SSP_ERR_IN_USE	Peripheral is still running in another mode; perform R_ADC_Close first.
SSP_ERR_INVALID_POINTER	The parameter p_data is NULL.

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. ADC HAL Module Operational Overview

The ADC HAL module controls the ADC units on a Synergy microcontroller, as configured by the user. It directly controls the ADC hardware without using any RTOS elements and provides convenient APIs to simplify development. The module supports three operation modes: single-scan, continuous-scan, and group-scan modes.

- Single-scan mode sequentially converts the analog inputs of selected channels in the ascending order of the channel number once per trigger.
- Continuous-scan mode sequentially converts the analog inputs of selected channels continuously in the ascending order of the channel numbers; a single trigger is required to start off the scan.
- Group-scan mode allows the user to add channels to one of two groups (A and B) and converts the analog inputs of the selected channels for each group in the ascending order of the channel number, when the specified start trigger for that group is received.

In group-mode, operation channels can be assigned to one of two groups: group-A or group-B. A trigger is assigned for each group to start the scan; in group mode, only hardware triggers can be used, whereas in normal mode, software triggers or an external trigger can be used. With the priority configuration parameter, you can specify:

- Whether a trigger for one group can interrupt an ongoing scan for the other group.
- Whether an interrupted scan resumes, restarts, or simply aborts the current scan and waits for the next trigger.

Interrupts and Callbacks

When a scan is complete and a callback is provided by the user (with interrupts enabled), the ADC HAL module invokes the callback (p_callback) with the argument adc_callback_args_t, indicating the unit and the event adc_cb_event_t. If interrupts are not enabled, the API supports checking the scan status to poll if the scan is complete (`scanStatusGet`) and provides a function to read the converted ADC result.

The ADC HAL module supports two interrupts:

- The normal/group-A interrupt, which fires when a scan is completed in single-scan mode or when a group-A scan is completed in group mode.
- The group-B interrupt, which fires when a group-B scan is completed in group mode.

Interrupts function differently in each mode:

- In single-scan mode, the normal interrupt is triggered when the scan is completed.
- In continuous-scan mode, the hardware will constantly scan the selected channels. In this mode, the driver will return an error if interrupts are enabled (and therefore must be disabled).
- In group mode, the ADC unit supports two interrupts: the normal interrupt (also called the group-A interrupt) and the group-B interrupt. The group-A interrupt is triggered when a group-A scan is completed; the group-B interrupt is triggered when a group-B scan is completed. In group mode, the normal interrupt is referred to as the group-A interrupt, even though it is the same vector.

In the following situations, you must enable the ADC scan-complete interrupt for the selected unit in the BSP in the following situations:

To get an interrupt when a normal scan is completed.
To get an interrupt when a group scan is completed.

3.1 ADC HAL Module Important Operational Notes and Limitations

3.1.1 ADC HAL Module Operational Notes

Sample-State Count Setting

The user can modify the setting of the sample-state count by calling the `sampleStateCountSet` API. You only need to modify the sample-state count settings from their default values if you want to increase the sampling time, either because the impedance of the input signal is too high to secure sufficient sampling time, or the ADCLK is too slow. To modify the sample-state count for a given channel, set the channel number `adc_sample_state_reg_t` and the number of states `num_states`. Valid sample state counts are 5-255. If you want to change the sample state count for multiple channels, call the `sampleStateCountSet` API repeatedly with modified arguments for each channel.

Triggering a Data Transfer with the ADC

To trigger a transfer of data when the ADC scan completes, configure the data transfer with `activation_source` set to `ELC_EVENT_ADCn_SCAN_END` or `ELC_EVENT_ADCn_SCAN_END_B` (where n is the ADC channel number). The ELC events are listed under `elc_event_t`. To retrieve the ADC unit-specific information to use with the `transfer` API, use the `infoGet` function call in the `transfer` API.

Triggering ELC Events with the ADC

The ADC unit can trigger the start of other peripherals listed in `elc_peripheral_t`. Refer to the “ELC Interface” in the *SSP User’s Manual* for more information.

Using the Temperature Sensor with the ADC

The ADC HAL module supports reading the data from the on-chip temperature sensor. The value returned from the sensor can be converted into degrees Celsius or Fahrenheit by the user application using the following formula, T = (Vs – V1)/slope + T1, where:

- T: Measured temperature (°C)
- Vs: Voltage output by the temperature sensor at the time of temperature measurement (Volts)
- T1: Temperature experimentally measured at one point (°C)
- V1: Voltage output by the temperature sensor at the time of measurement of T1 (Volts)
- T2: Temperature at the experimental measurement of another point (°C)
- V2: Voltage output by the temperature sensor at the time of measurement of T2 (Volts)
- Slope: Temperature gradient of the temperature sensor (V/°C); slope = (V2 – V1)/ (T2 – T1)

The slope value can be obtained from the hardware manual for each device under Electrical Characteristics::TSN Characteristics. The slope is positive for the S7/S5 devices and negative for the S3/S1 devices.

3.1.2 ADC HAL Module Limitations

When configuring the ADC channels to be used with the ADC HAL module, the temperature and voltage sensors must not be selected if any of the other available channels are also selected. It is possible to only use the temperature sensor, voltage sensor, or any number of the regular ADC channels.

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

4. Including the ADC HAL Module in an Application

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

Note: This section assumes 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 ADC HAL module to an application, simply add it to a HAL/Common thread using the stacks selection sequence given in the following table. (The default name for the ADC HAL module is g_adc0. This name can be changed in the associated Properties window.)

Table 3. ADC HAL Module Selection Sequence

Resource	ISDE Tab	Stacks Selection Sequence
g_adc0 ADC Driver on r_adc	Threads	New Stack> Driver> Analog> ADC Driver on r_adc

When the ADC HAL module on r_adc is added to the thread stack, as shown in the following figure, the configurator automatically adds any lower-level drivers that are required. Any drivers that need additional configuration information will be box-text highlighted in Red. Modules with a Gray band are individual modules that stand alone.

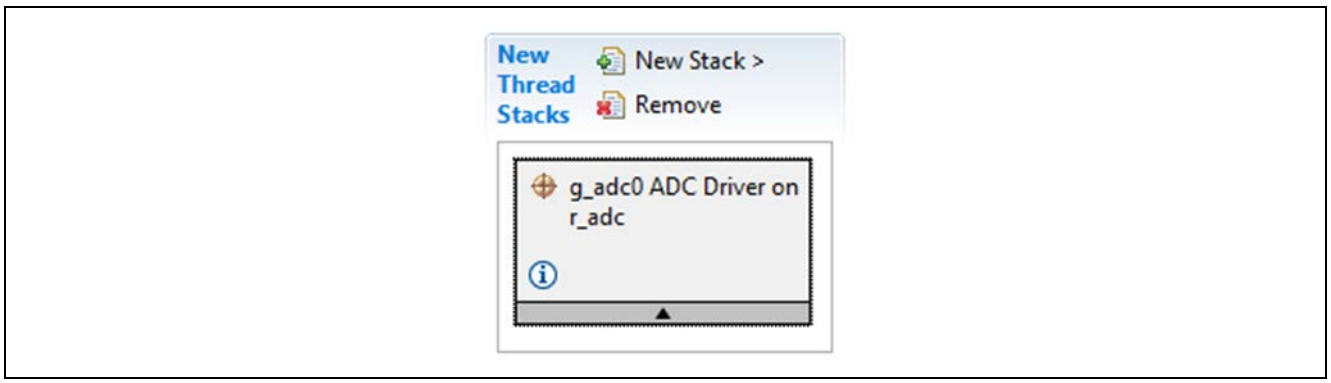

Figure 2. ADC HAL Module Stack

5. Configuring the ADC HAL Module

The ADC HAL 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 to ensure successful operation. Only properties that can be changed without causing conflicts are available for modification. Other properties are 'locked' with a lock icon in the Properties window in the ISDE and unavailable for changes. 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 shown in the following tables for easy reference and given in the Properties tab within the SSP configurator.

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 indicate 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 will help orient you and can be a useful 'hands-on' approach to learning the ins and outs of developing with the SSP.

Table 4. Configuration Settings for the ADC HAL Module on r_adc

ISDE Property	Value	Description
Parameter Checking	BSP, Enabled, Disabled Default: Enabled	If selected code for parameter checking is included in the build.
Name	g_adc0	Module name
Unit	0, 1 (S7G2 Only) Default: 0	Specify the ADC Unit to be used. The s7g2 has two units; 0 and 1.
Resolution	14-Bit (S3A7/S124 Only), 12-Bit, 10-Bit (S7G2) Default: 8-Bit (S7G2 Only)	Specify the conversion resolution for this unit.
Alignment	Right, Left Default: Right	Specify the conversion result alignment.
Clear after read	Off, On Default: On	Specify if the result register must be automatically cleared after the conversion result is read. Note: If this is enabled, then watching the result register using a debugger always results in a 0.
Mode	Single Scan, Continuous Scan, Group Scan Default: Single Scan	Specify the mode that this ADC unit is used in.
Channels 0-6	Unused, Use in Normal/Group A, Use in Group B Default: Unused	In Normal mode of operation, this field is used to specify the channels enabled in that ADC unit. In group mode, this field is used to specify which channels belong to group A or B.
Channels 7-10 (S3A7/S124 Only)	Unused, Use in Normal/Group A, Use in Group B Default: Unused	In Normal mode of operation, this field is used to specify the channels enabled in that ADC unit. In group mode, this field is used to specify which channels belong to group A or B.

ISDE Property	Value	Description
Channels 11-15 (S3A7 Only)	Unused, Use in Normal/Group A, Use in Group B Default: Unused	In Normal mode of operation, this field is used to specify the channels enabled in that ADC unit. In group mode, this field is used to specify which channels belong to group A or B.
Channels 16-20	Unused, Use in Normal/Group A, Use in Group B (Default: Unused)	In Normal mode of operation, this field is used to specify the channels enabled in that ADC unit. In group mode, this field is used to specify which channels belong to group A or B.
Channel 21 (Unit 0 Only)	Unused, Use in Normal/Group A, Use in Group B Default: Unused	In Normal mode of operation, this field is used to specify the channels enabled in that ADC unit. In group mode, this field is used to specify which channels belong to group A or B.
Channel 22 (S3A7/S124 Only)	Unused, Use in Normal/Group A, Use in Group B Default: Unused	In Normal mode of operation, this field is used to specify the channels enabled in that ADC unit. In group mode, this field is used to specify which channels belong to group A or B.
Channels 23-27 (S3A7 Only)	Unused, Use in Normal/Group A, Use in Group B Default: Unused	In Normal mode of operation, this field is used to specify the channels enabled in that ADC unit. In group mode, this field is used to specify which channels belong to group A or B.
Temperature Sensor	Unused, Use in Normal/Group A, Use in Group B Default: Unused	Temperature sensor use selection for Channel Scan Mask
Voltage Sensor	Unused, Use in Normal/Group A, Use in Group B Default: Unused	Voltage sensor use selection for Channel Scan Mask
Scan Mask Group B	Use #define ADC_MASK_xxx which are defined in r_adc.h. Use (ADC_MASK_xxx|ADC_MASK_xxx) for multiple channels.	This field is valid only in group mode. It is used to specify which channels belong to group B. Warning: Be sure that the same channels are not specified in group A and B.
Normal/Group A Trigger	None, Asynchronous External Trigger 0, ELC Event, Software Default: Software	Specify the trigger type to be used for this unit. If group mode is used adc_cfg_t::mode, then this field is used to set the Group A trigger. Note: The only valid option in group mode is the ELC trigger.
Group B Trigger (Valid Only in Group Scan Mode)	ELC Event (The only valid trigger for either group in Group Scan Mode)	Specify the group B trigger. This option is only valid if group mode is chosen in adc_cfg_t::mode.
Group Priority (Valid only in Group Scan Mode)	Group A cannot interrupt Group B, Group A can interrupt Group B; Group B scan restarts at next trigger, Group A can interrupt Group B; Group B scan restarts immediately, Group A can interrupt Group B; Group B scan restarts immediately and scans continuously (Default: Group A cannot interrupt Group B)	Determines whether an ongoing group B scan can be interrupted by a group A trigger, whether it should abort on a group A trigger, or if it should pause to allow group A scan and restart immediately after group A scan is complete. Note: This field is valid only in group mode.

ISDE Property	Value	Description
Add/Average Count	Disabled, Add two samples, Add three samples, Add four samples, Add sixteen samples, Average two samples, Average four samples Default: Disabled	Specify if addition or averaging needs to be done for any of the channels in this unit. The actual channels are specified by using a channel mask adc_channel_cfg_t::add_mask.
Channels 0-27	Disabled, Enabled Default: Disabled	This field is valid only if adc_cfg_t::add_average_count is enabled. This field determines what channels results are to be averaged or summed.
Temperature Sensor	Disabled, Enabled Default: Disabled	Temperature sensor use selection for Addition/Averaging Mask
Voltage Sensor	Disabled, Enabled Default: Disabled	Voltage sensor use selection for Addition/Averaging Mask
Channels 0-2	Disabled, Enabled Default: Disabled	Determines which of channels 0, 1 and 2 are using the updated sample-and-hold states value specified in adc_channel_cfg_t::sample_hold_states. This field must only be set if it is desired to modify the default sample and hold count value for channels 0, 1 and 2.
Sample Hold States (Applies only to the 3 channels selected above)	24	Specifies the updated sample-and-hold count for the channel dedicated sample-and-hold circuit. This field is valid only if adc_channel_cfg_t::sample_hold_mask is not 0. Only channels 0, 1 and 2 have dedicated sample and hold circuits. Note: Use this to modify the default number of states (24) for which the value is sampled. Each state is equal to 1/ADCLK time.
Callback	NULL	A user callback function can be registered in adc_api_t::open. If this callback function is provided, it is called from the interrupt service routine (ISR) each time the ADC scan completes. Warning: 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.
Scan 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	Scan End Interrupt Priority selection

ISDE Property	Value	Description
Scan End Group B 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	Scan End Group B Interrupt Priority selection

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

5.1 ADC HAL Module Clock Configuration

The ADC HAL module uses the PCLKC as its clock source (ADCLK.) The only restriction when configuring this clock is that it should be set to less than the max ADC clock; there is also a restriction on the ratio of the PCLKC and PCLKB clocks specified in the hardware manual.

The ADC-conversion time depends on the PCLKC setting.

To set the PCLKB and PCLKC frequencies, use the clock configurator in the ISDE.

To change the clock frequency at run-time, use the CGC Interface.

5.2 ADC HAL Module Pin Configuration

To use the ADC HAL module, the port pins for the channels receiving the analog input must be set as input pins in the pin configurator in the ISDE. The following table illustrates the method for selecting the pins within the ISDE configuration window:

Table 5. Pin Selection Sequence for the ADC HAL Module

Resource	ISDE Tab	Pin selection Sequence
ADC	Pins	Select Peripherals > Analog Pins>ADC0\1>AN_XX

6. Using the ADC HAL Module in an Application

Once the module has been configured and the files generated, the ADC is ready to be used in an application.

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

1. Initialize the ADC using the open API
2. Configure the channels using the scanCfg API
3. Start a conversion using the desired trigger with the scanStart API
   If a hardware trigger is used, this call enables the ADC unit to be triggered by the hardware trigger. If a software trigger is used, then this call starts the ADC scan.
4. If interrupts are disabled, use the scanStatusGet call to determine if the scan is complete
5. If interrupts are enabled, the callback will be invoked when the scan is complete
6. Read the results of the conversion using the read API
7. Stop the ADC scan by calling the scanStop API
   This prevents the ADC from being triggered by an external trigger or a hardware trigger; it also forces a stop of a software-triggered scan if one is ongoing.
8. Operate on the received data as needed by the application.
9. Use the `close` call to power down the peripheral.

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

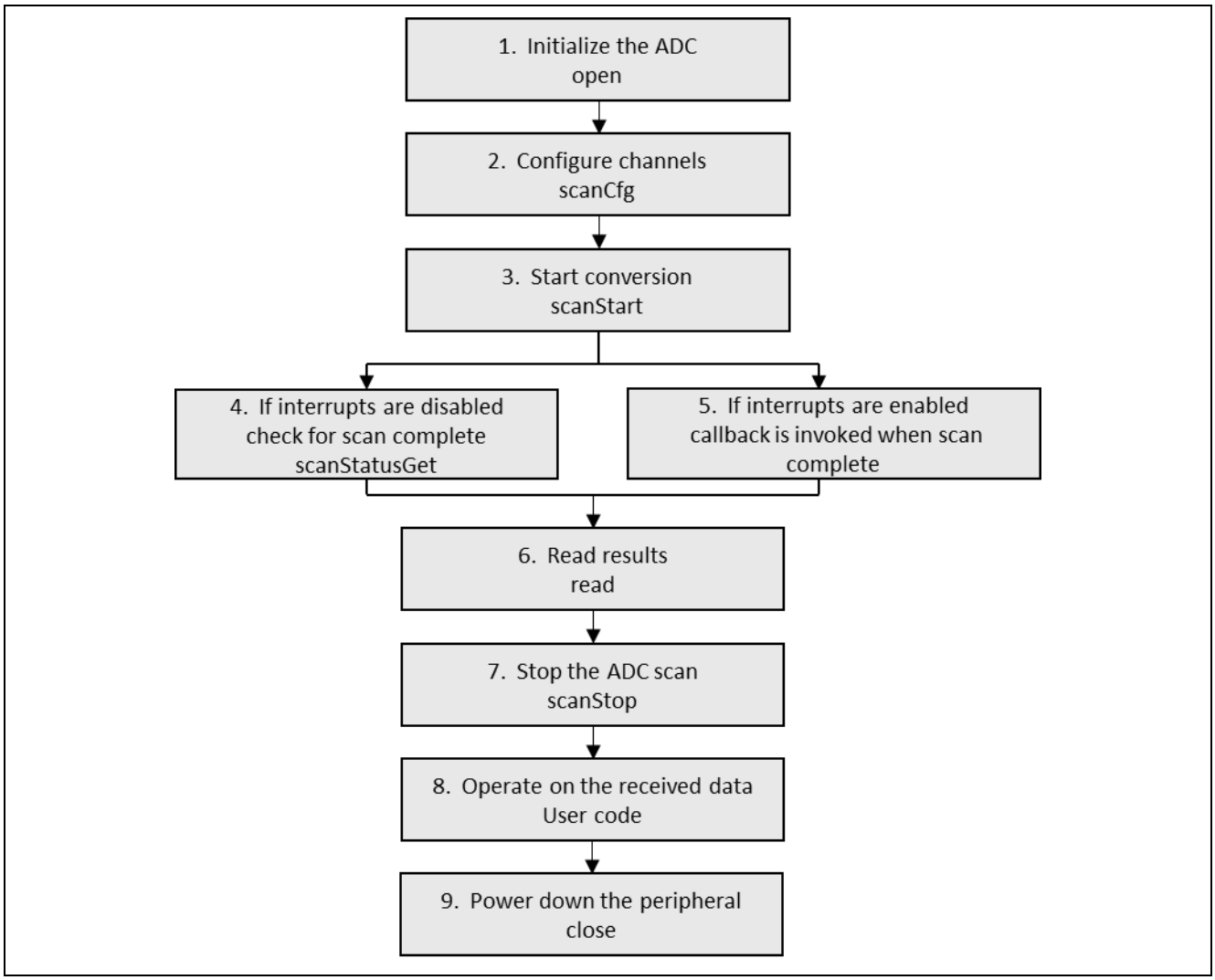

Figure 3. Flow Diagram of a Typical ADC HAL Module Application

7. The ADC HAL Module Application Project

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

The application project demonstrates the typical use of the ADC HAL module APIs. The application project initializes the ADC and periodically scans the temperature sensor. The scan result is placed in a user-specified buffer. A user-callback function is entered when the scan result is available. The user-specified callback function prints the result on the debug console using the common semi-hosting function. The following table identifies the target versions for the associated software and hardware used by the application project:

Table 6. Software and Hardware Resources Used by the Application Project

Resource	Revision	Description
e² studio	5.4.0 or later	Integrated Solution Development Environment
SSP	1.2.1 or later	Synergy Software Platform
IAR EW for Renesas Synergy	7.71.2 or later	IAR Embedded Workbench® for Renesas Synergy™
SSC	5.4.0 or later	Synergy Standalone Configurator
SK-S7G2	v3.0 to v3.1	Starter Kit

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

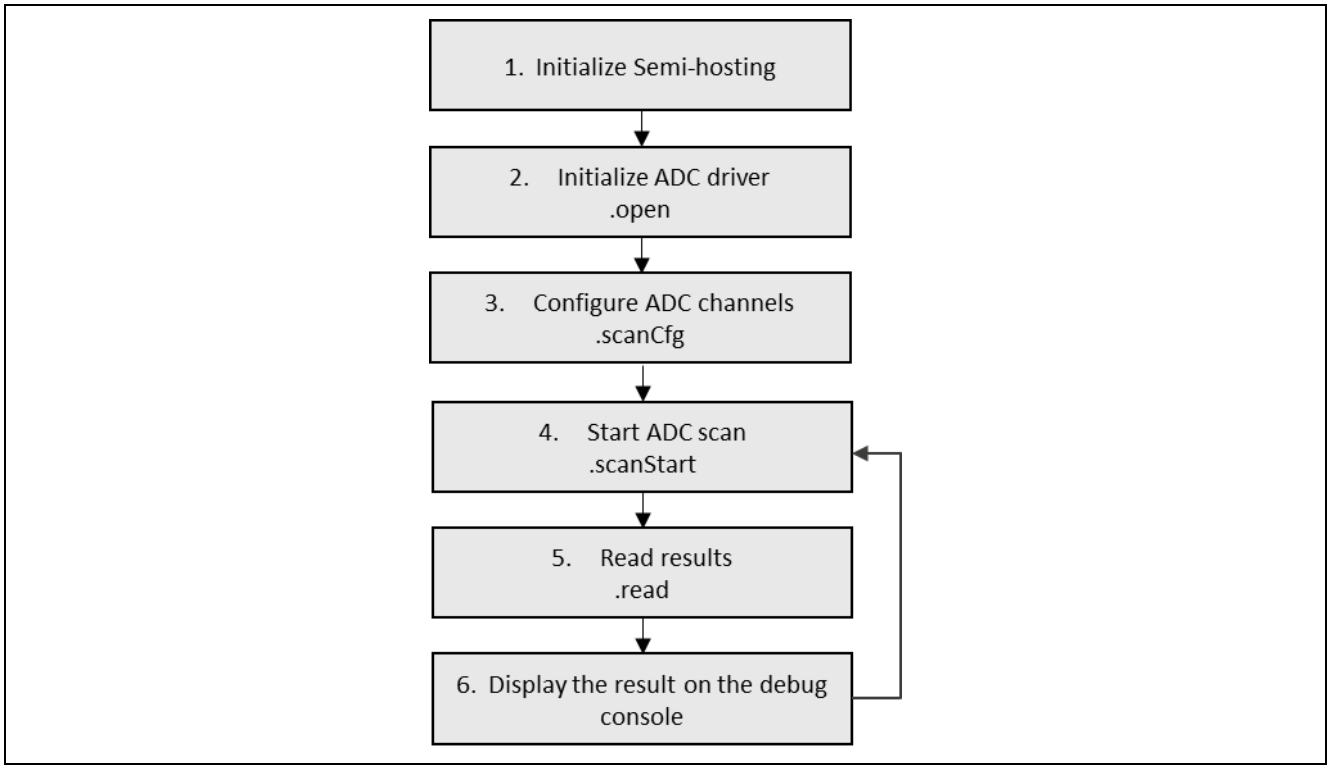

Figure 4. ADC HAL Module Application Project Flow Diagram

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

The first section of the `adc_hal.c` has the header files which reference the ADC instance structure and a code section which allows semi-hosting to display results using `printf()`. The following section is the entry function for the main program-control section. The ADC HAL module is initialized using the `open` API. Inside the 'forever' while loop, the ADC scan is started using the `start` API. If the status return code is non-zero, an error has occurred and the interior-while loop traps on the error. If the return status is zero, the scan has completed normally and the ADC is stopped (using the `stop` API.) A thread sleep function pauses execution for a single ThreadX-timer tick and then the while loop functions are repeated.

The last section is the user-callback function. This function initializes semi-hosting and then accesses the result buffer (`g_user_buffer`) using the `buffer_index` returned by the scan result with the `p_args` pointer. The `printf` function displays the short message and the value of the temperature sensor read by the ADC.

Note: This description assumes you are familiar with using `printf()` with the Debug Console in the Synergy Software Package. If you are unfamiliar with this, refer to the "How do I Use Printf() with the Debug Console in the Synergy Software Package" Knowledge Base article, available as described in the References section at the end of this document. Alternatively, the user can see results via the watch variables in the debug mode.

A few key properties are configured in this application project to support the required operations and the physical properties of the target board and MCU. The properties with the values set for this specific project are listed in the following table. You can also open the application project and view these settings in the Properties window as a hands-on exercise.

Table 7. ADC HAL Module Configuration Settings for the Application Project

ISDE Property	Value Set
ADC0_SCAN_END	Priority 3
Name	g_adc
Unit	1
Resolution	12-Bit
Mode	Single Scan
Temperature Sensor	Used in Normal/ Group A
Callback	adc_user_callback

8. Customizing the ADC HAL Module for a Target Application

Some configuration settings will normally be changed by the developer from those shown in the application project. For example, the user can easily change the configuration settings for the ADC clock by updating the PCLKC in the clock tab. The user can also change the ADC port pins to select the desired analog input; this can be done using the Pins tab in the configurator. The ADC application project uses channel 13 which is connected to the onboard potentiometer and the associated pin is configured as an analog input.

9. Running the ADC HAL Module Application Project

To run the ADC HAL module application project and to see it executed on a target kit, you can simply import it into your ISDE, compile, and run debug. Refer to the Synergy Project Import Guide (r11an0023eu0121-synergy-ssp-import-guide.pdf), included in this package, for instructions on importing the project into e^2 studio or IAR EW for Synergy, and building/running the application.

To implement the ADC HAL module application in a new project, use the following steps to define, configure, auto-generate files, add code, compile, and debug the project on the target kit. Following these steps is a hands-on approach that can help make the development process with SSP more practical.

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 unfamiliar, refer to the first few chapters of the *SSP User's Manual* for detail instructions.

To create and run the ADC application project simply follow these steps:

1. Create a new Renesas Synergy project for the S7G2-SK called CGC_HAL_MG_AP.
2. Select the **Threads** tab.
3. Add a thread called, **adc_thread**, then add the ADC Driver on r_adc module using the Driver > Analog selection path.
4. Click on the **Generate Project Content** button.
5. Add the code from the supplied project file "`adc_thread.c` " or copy over the generated "`adc_thread.c` " file.
6. Connect to the host PC using the USB cable (use J19 DEBUG_USB connector).
7. Start to debug the application.
   The output can be viewed in the Renesas Debug Console.

```
Internal Temperature Sensor value measured by ADC: 1568
Internal Temperature Sensor value measured by ADC: 1569
Internal Temperature Sensor value measured by ADC: 1570
Internal Temperature Sensor value measured by ADC: 1573
```

Figure 5. Example Output from ADC HAL Module Application Project

10. ADC 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 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. ADC HAL Module Next Steps

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

You may find that the ADC Periodic Framework is a better fit for your target application. The ADC Periodic Framework module guide illustrates the use of the ADC within a ThreadX-based implementation. This guide is available as described in the References section at the end of this document.

12. ADC 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_adc module reference materials and resources are available on the Synergy Knowledge Base: https://en-support.renesas.com/knowledgeBase/16977469.

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	May.15.17	—	Initial Release
1.01	Jun.16.17	11	Removed “thread” from description in Section 7 paragraph 2
1.02	Aug.23.17	—	Various edits for consistency, update to resource table
1.03	Feb.22.19	—	Updated Channel property description and Table 7

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/.