USB Renesas Host Controller User's Guide

Version 1.10

For use with USBH Renesas Host Controller versions 2.14 and above

Date: 30-Mar-2015 11:30

Table of Contents

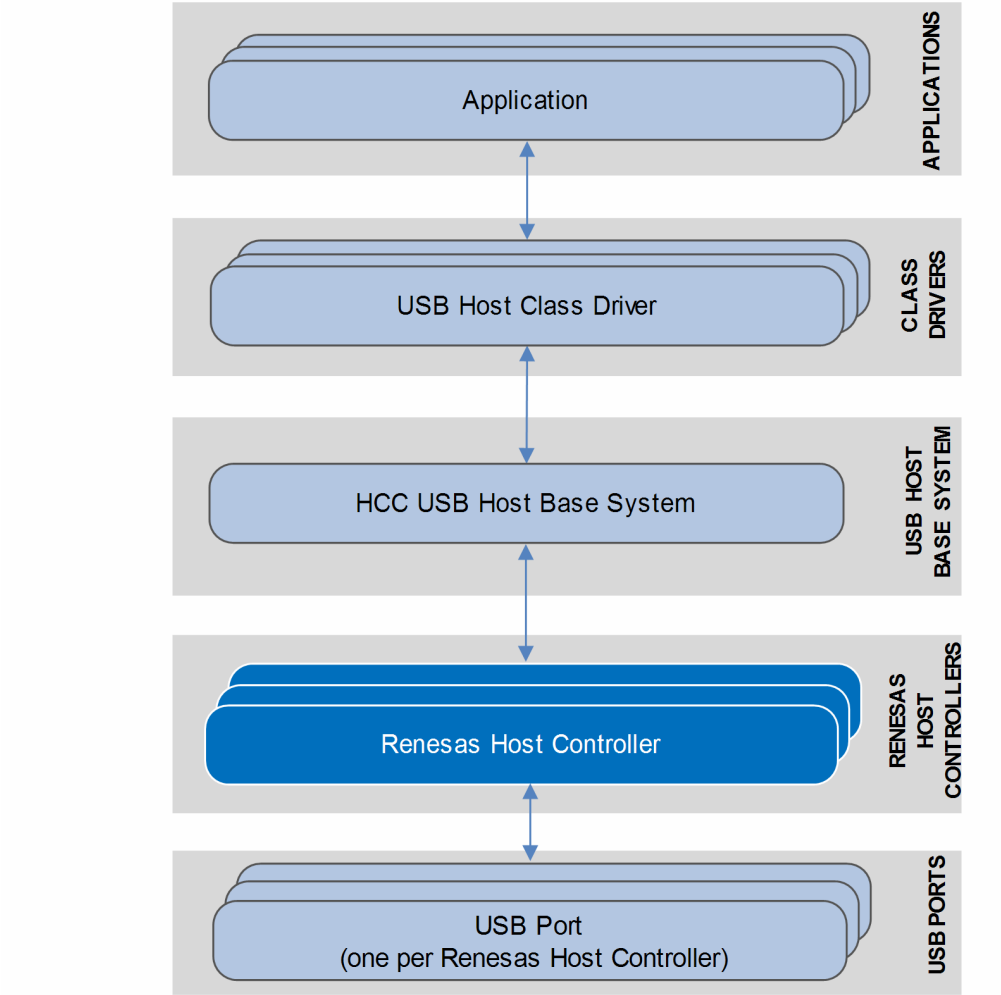
1 System Overview

1.1 Introduction

This guide is for those who want to implement HCC Embedded's Renesas USB Host Controller with the HCC USB host stack.

The Renesas module provides a high speed USB 2.0 host controller which provides both full and low speed USB functions. The controller can handle all USB transfer types and, in conjunction with the USB host stack, can be used with any USB class driver. Renesas MCUs supported include those in the SuperH SH-2A and SH7260 series.

The position of the host controller within the USB stack is shown below:



1.2 Feature Check

The main features of the host controller are the following:

- It conforms to the HCC Advanced Embedded Framework.
- It can be used with or without an RTOS.
- It is integrated with the HCC USB Host stack and all its class drivers.
- It supports multiple simultaneous Renesas controllers, each with multiple devices attached.
- It supports all USB transfer types: Control, Bulk, Interrupt and Isochronous.

1.3 Packages and Documents

Packages

The table below lists the packages that you need in order to use this module:

Package	Description
hcc_base_doc	This contains the two guides that will help you get started.
usbh_base	The USB host base package. This is the framework used by USB class drivers to communicate over USB using a specific USB host controller package.
usbh_drv_renesas	The USB Renesas host controller package described by this document.

Documents

Readers should note the points in the HCC Documentation Guidelines on the HCC documentation website.

HCC Firmware Quick Start Guide

This document describes how to install packages provided by HCC in the target development environment. Also follow the *Quick Start Guide* when HCC provides package updates.

HCC Source Tree Guide

This document describes the HCC source tree. It gives an overview of the system to make clear the logic behind its organization.

HCC USB Host Base System User's Guide

This document defines the USB host base system upon which the complete USB stack is built.

HCC USB Renesas Host Controller User's Guide

This is this document.

1.4 Change History

This section includes recent changes to this product. For a list of all the changes, refer to the file **src/history/usb-host/usbh_hc_renesas.txt** in the distribution package.

Version	Changes
2.14	When running without an OS (oal_os_nos) the unplug/plug event was missed if the re-plug occurred between two **oal_task_poll()** calls.
2.13	Fixed the bus error that occurred in big-endian and OTG mode when switched between host and device mode. Eliminated compiler warnings.
2.12	Unaligned buffers can be transferred with usbh_renesas. Implemented 32-bit FIFO access for SH7269 devices.

2 Source File List

This section describes all the source code files included in the system. These files follow the HCC Embedded standard source tree system, described in the *HCC Source Tree Guide*. All references to file pathnames refer to locations within this standard source tree, not within the package you initially receive.

> **Note:** Do not modify any of these files except the configuration file and PSP files.

2.1 API Header File

The file **src/api/api_usbh_renesas.h** is the only file that should be included by an application using this module. It declares the usbh_renesas_hc() function. For details, see Starting the Host Controller.

2.2 Configuration File

The file **src/config/config_usbh_renesas.h** contains all the configurable parameters. Configure these as required. For details of these options, see Configuration Options.

2.3 Source Code

The source code files are in the directory **src/usb-host/usb-driver/renesas**. **These files should only be modified by HCC.**

File	Description
renesas.c	Source file for Renesas code.
renesas.h	Header file for Renesas public functions.
renesas_hc.c	Source file for the Renesas HC descriptor.
renesas_hc.h	HC descriptor header file.
renesas_hub.c	Source file for Renesas hub.
renesas_hub.h	Header file for Renesas hub public functions.
renesas_hw.h	Header file for Renesas hardware-specific functions.

2.4 Version File

The file **src/version/ver_usbh_renesas.h** contains the version number of this module. This version number is checked by all modules that use this module to ensure system consistency over upgrades.

2.5 PSP Files

There are two sets of files, in directories named **src/psp_rx63/target** and **src/psp_sh7269/target**. These provide functions and elements the core code may need to use, depending on the hardware.

> **Note:** These are PSP implementations for the specific microcontroller and board; you may need to modify these to work with a different microcontroller and/or development board. See PSP Porting for details.

The files are as follows:

File	Description
config/config_usbh_renesas.h	Configuration file.
target/include/hcc_rx63n_regs.h or **hcc_renesas_sh7269_regs.h**	Register definitions.
target/usb-host-renesas/psp_usbh_renesas.c	Functions source code.
target/usb-host-renesas/psp_usbh_renesas.h	Header file for functions.

3 Configuration Options

Set the system configuration options in the file **src/config/config_usbh_renesas.h**. This section lists the available configuration options and their default values.

USBH_PORT_USED

The port to use. The default is zero.

RENESAS_TRANSFER_TASK_STACK_SIZE

The stack size of the transfer task(s). The default is 1024.

RENESAS_ISR_ID

The ISR ID of the host controller. The default is HCC_VECT_USB0_USBI0.

RENESAS_INT_PRIO

The ISR priority of the host controller. The default is 1.

RENESAS_SUPPORT_HS

Set this to 1 to support high speed transfers (for high speed SH7264 devices). The default is zero.

RENESAS_BIGEND_FIFO_ACCESS

Keep this at the default of 1 if big-endian FIFO port control is needed (for high speed SH7264 devices).

RENESAS_32BIT_FIFO_ACCESS

Keep this at the default of 1 for high speed SH7264 devices. (SH7264 devices allow 32 bit access to FIFO.)

RENESAS_MAX_EP

The maximum number of bulk and interrupt endpoints, including EP0s. The default is 15.

RENESAS_ISOCHRONOUS_SUPPORT

Set this to 1 to support isochronous transfers. The default is zero.

RENESAS_NUM_BULK_PIPE_RX

The pipe to use for bulk RX transfers. The default is 2.

RENESAS_NUM_BULK_PIPE_TX

The pipe to use for bulk TX transfers. The default is 1.

RENESAS_NUM_INT_PIPE_RX

The pipe to use for interrupt RX transfers. The default is 3.

RENESAS_NUM_INT_PIPE_TX

The pipe to use for interrupt TX transfers. The default is 1.

4 Starting the Host Controller

This section shows how to start the host controller and describes the task created. It includes a code example.

4.1 usbh_renesas_hc

This external interface function provides the host controller descriptor required by the **usbh_hc_init()** function.

Format

```
extern void * const  usbh_renesas_hc
```

4.2 Host Controller Task

The host controller task handles all completed transfers. Callback requested for the transfer is executed from this task.

The task has the following attributes:

Attribute	Description
Entry point	*renesas_transfer_task*
Priority	USBH_TRANSFER_TASK_PRIORITY
Stack size	RENESAS_TRANSFER_TASK_STACK_SIZE. The default is 1024.

4.3 Code Example

This example shows how to initialize the host controller. Note the following:

- There is only one external interface function, **usbh_renesas_hc()**. To link this host controller to the system, you call the **usbh_hc_init()** function with this function as a parameter.
- The last parameter in the **usbh_hc_init()** call is the number of the host controller.

```
void start_usb_host_stack ( void )
{
int  rc;
rc = hcc_mem_init();
    if ( rc == 0 )
    {
        rc = usbh_init();    /* Initialize USB host stack */
    }
    if ( rc == 0 )
    {
        /* Attach Renesas host controller */
        rc = usbh_hc_init( 0, usbh_renesas_hc, 0 );
    }

    if ( rc == 0 )
    {
        rc = usbh_start();  /* Start USB host stack */
    }
    if ( rc == 0 )
    {
        rc = usbh_hc_start( 0 );  /* Start Renesas Host controller */
    }

.....
}
```

5 Integration

This section specifies the elements of this package that need porting, depending on the target environment.

5.1 OS Abstraction Layer (OAL)

All HCC modules use the OS Abstraction Layer (OAL) that allows the module to run seamlessly with a wide variety of RTOSes, or without an RTOS.

This module requires the following OAL elements:

OAL Resource	Number Required
Tasks	1
Mutexes	1
Events	1
ISRs	1

5.2 PSP Porting

The Platform Support Package (PSP) is designed to hold all platform-specific functionality, either because it relies on specific features of a target system, or because this provides the most efficient or flexible solution for the developer.

The module makes use of the following standard PSP macro:

Function	Package	Element	Description
PSP_RD_LE16	psp_base	psp_endianness	Reads a 16 bit value stored as little-endian from a memory location.

The host controller makes use of the following functions that must be provided by the PSP. These are designed for you to port them easily to work with your hardware solution. The package includes samples for the RX63 and SH7269 devices in the **psp_usbh_renesas.c** files.

Function	Description
renesas_hw_init()	Initializes the device.
renesas_hw_start()	Starts the device.
renesas_hw_stop()	Stops the device.
renesas_hw_delete()	Deletes the device, releasing the associated resources.

These functions are described in the following sections.

Note: HCC can provide samples for different configurations; contact support@hcc-embedded.com.

renesas_hw_init

This function is provided by the PSP to initialize the device.

Format

```
int renesas_hw_init ( t_usbh_unit_id unit )
```

Arguments

Argument	Description	Type
unit	The unit ID.	t_usbh_unit_id

Return Values

Return value	Description
USBH_SUCCESS	Successful execution.
USBH_ERROR	Operation failed.

renesas_hw_start

This function is provided by the PSP to start the device.

Format

```
int renesas_hw_start ( t_usbh_unit_id unit )
```

Arguments

Argument	Description	Type
unit	The unit ID.	t_usbh_unit_id

Return Values

Return value	Description
USBH_SUCCESS	Successful execution.
USBH_ERROR	Operation failed.

renesas_hw_stop

This function is provided by the PSP to stop the device.

Format

```
int renesas_hw_stop ( t_usbh_unit_id unit )
```

Arguments

Argument	Description	Type
unit	The unit ID.	t_usbh_unit_id

Return Values

Return value	Description
USBH_SUCCESS	Successful execution.
USBH_ERROR	Operation failed.

renesas_hw_delete

This function is provided by the PSP to delete the device, releasing associated resources.

Format

```
int renesas_hw_delete ( t_usbh_unit_id unit )
```

Arguments

Argument	Description	Type
unit	The unit ID.	t_usbh_unit_id

Return Values

Return value	Description
USBH_SUCCESS	Successful execution.
USBH_ERROR	Operation failed.