USB OHCI Host Controller User's Guide

Version 1.20

For use with USBH OHCI Host Controller versions 2.13 and above

Date: 27-Mar-2015 16:59

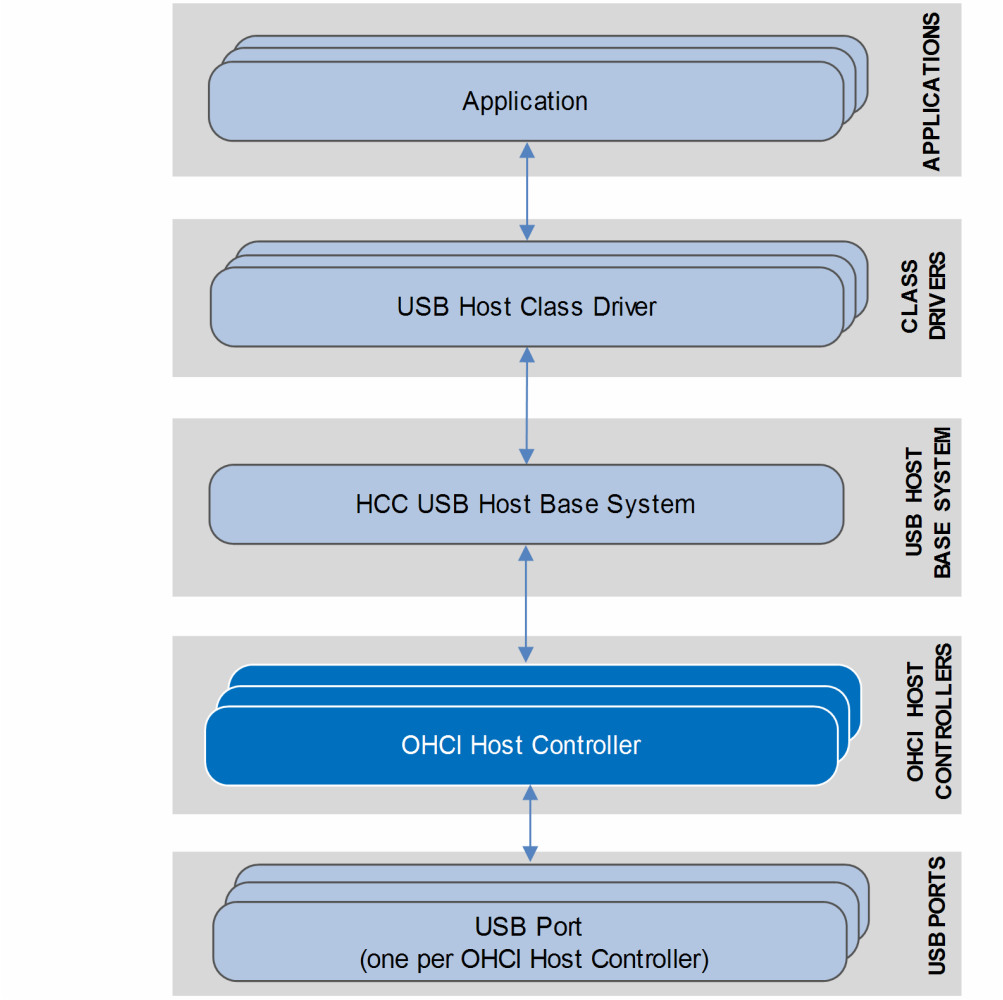
Table of Contents

1 System Overview

1.1 Introduction

This guide is for those who want to configure and use HCC's Open Host Controller Interface (OHCI) module with HCC's USB host stack. The OHCI module provides a high speed USB 1.1 host controller which provides full speed 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.

The OHCI Host Controller package provides a host controller for a USB stack, as 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 HCC USB Host stack and all its class drivers.
- It can be used with any OHCI-compliant USB host controller.
- It supports multiple simultaneous OHCI 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_ohci	The USB OHCI 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 OHCI Host Controller User's Guide

This is this document.

1.4 Change History

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

Version	Changes
2.13	Updated to work with USB host base major version 3.
2.12	Changed PSP header file name.
2.11	Fixed faulty behavior when removing device address zero.

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_ohci.h** is the only file that should be included by an application using this module. It declares the usbh_ohci_hc() function.

2.2 Configuration File

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

2.3 Source Code

These are the source code files. **These files should only be modified by HCC.**

File	Description
src/usb-host/usb-driver/ohci/ohci.c	Source file for OHCI code.
src/usb-host/usb-driver/ohci/ohci.h	Header file for OHCI public functions.
src/usb-host/usb-driver/ohci/ohci_hc.c	Source file for OHCI HC descriptor.
src/usb-host/usb-driver/ohci/ohci_hc.h	OHCI-specific header file.
src/usb-host/usb-driver/ohci/ohci_hub.c	Source file for OHCI hub.
src/usb-host/usb-driver/ohci/ohci_hub.h	Header file for OHCI hub public functions.
src/usb-host/usb-driver/ohci/ohci_reg.h	OHCI register file.

2.4 Version File

The file **src/version/ver_usbh_ohci.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

These files are in the directory **src/psp/target/usb_host_ohci**. They 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 development board; you may need to modify these to work with a different microcontroller and/or board. See PSP Porting for details.

File	Description
psp_usbh_ohci.c	Functions source code.
psp_usbh_ohci.h	Functions header file.

3 Configuration Options

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

OHCI_TRANSFER_TASK_STACK_SIZE

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

OHCI_MAX_DEVICE

The maximum number of devices supported. The default is 4. This should correspond to the maximum number of physical devices that will be attached; each externally attached hub counts as a device.

OHCI_MAX_EP

The maximum number of Bulk, Isochronous, and Interrupt endpoints. The default is 8.

OHCI_MAX_TRANSFERS

The maximum number of simultaneous transfers. The default is 8.

OHCI_MAX_ISO_TRANSFERS

The maximum number of simultaneous isochronous transfers. The default is 32.

OHCI_ISO_LATEST_START

The latest bit in a frame at which an ISO transfer can be started. The default is 400.

OHCI_HC_COUNT

The number of OHCI controllers supported. (Some microcontrollers have more than one OHCI controller.) The maximum is 4 and the default is 1.

OHCI_RENDIAN

The reverse endianness between OHCI registers and the system. Set this to 1 if the OHCI controller is operating in reverse endianness relative to the processor. The default is zero.

OHCI_DBUFFER_ADDRESS

The dedicated buffer start address (zero = none). The default is zero.

OHCI_DBUFFER_SIZE

The dedicated buffer size. The default is zero.

OHCI_READ_REG_32, OHCI_WRITE_REG_32

These specify the read/write routines to use when accessing an OHCI register. Currently these are mapped to **psp_rreg32()** and **psp_wreg32()**.

These macros are defined in **psp/include/psp_reg.h** and the parameters are (base, offset) for read and (base, offset, value) for write.

> **Note:** For the following options, *n* is 0 to 3. Only set values for the host controllers which are used.

OHCI_BASE_n

The base address of the OHCI register space, required if processor-specific registers are available for additional settings. The default is zero.

OHCI_ISR_ID_n

The ISR identifier of the OHCI controller. The default is zero. This is always platform/RTOS-specific.

OHCI_ISR_PRIORITY_n

The ISR priority of the OHCI controller. The default is zero. This is always platform/RTOS-specific.

4 Starting the OHCI Controller

This section shows how to start the host controller and describes the task created.

4.1 usbh_ohci_hc

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

Format

```
extern void * const  usbh_ohci_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	*ohci_transfer_task_n* (n=0/1/2/3)
Priority	OAL_HIGHEST_PRIORITY (USBH_TRANSFER_TASK_PRIORITY)
Stack size	This depends on the RTOS. 1024 is the default.

4.3 Code Example

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

- There is only one external interface function, **usbh_ohci_hc()**. You call the **usbh_hc_init()** function with this function as a parameter to link this host controller to the system.
- The last parameter in the **usbh_hc_init()** call is the number of the host controller. In this example OHCI has two controller units so the first call uses 0 and the second call uses 1.

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

    if ( rc == 0 )
    {
        /* Attach second OHCI host controller */
        rc = usbh_hc_init( 0, usbh_ohci_hc, 1 );
    }
    if ( rc == 0 )
    {
        rc = usbh_start();   /* Start the USB host stack */
    }
    if ( rc == 0 )
    {
        rc = usbh_hc_start( 0 );   /* Start first OHCI Host controller */
    }
    if ( rc == 0 )
    {
        rc = usbh_hc_start( 1 );   /* Start second OHCI 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	One for each supported OHCI host controller (OHCI_HC_COUNT).

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 functions:

Function	Package	Element	Description
psp_memcpy()	psp_base	psp_string	Copies a block of memory. The result is a binary copy of the data.
psp_memset()	psp_base	psp_string	Sets the specified area of memory to the defined value.
psp_rreg32()	psp_base	psp_reg	Read routine used to access an OHCI register.
psp_wreg32()	psp_base	psp_reg	Write routine used to access an OHCI register.

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 in the **src/psp/target/usb-host-ohci/psp_usbh_ohci.c** file.

Function	Description
ohci_hw_init()	Initializes the device.
ohci_hw_start()	Starts the device.
ohci_hw_stop()	Stops the device.
ohci_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.

ohci_hw_init

This function must be provided by the PSP to initialize the device.

Format

```
int ohci_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.

ohci_hw_start

This function must be provided by the PSP to start the device.

Format

```
int ohci_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.

ohci_hw_stop

This function must be provided by the PSP to stop the device.

Format

```
int ohci_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.

ohci_hw_delete

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

Format

```
int ohci_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.