USB Device Low Level Driver for MUSB CPPI User Guide

Version 1.00 BETA

For use with USBD Low Level Driver for MUSB CPPI versions 1.11 and above

Exported on 02/11/2019

Table of Contents

1 System Overview

This chapter contains the fundamental information for this module.

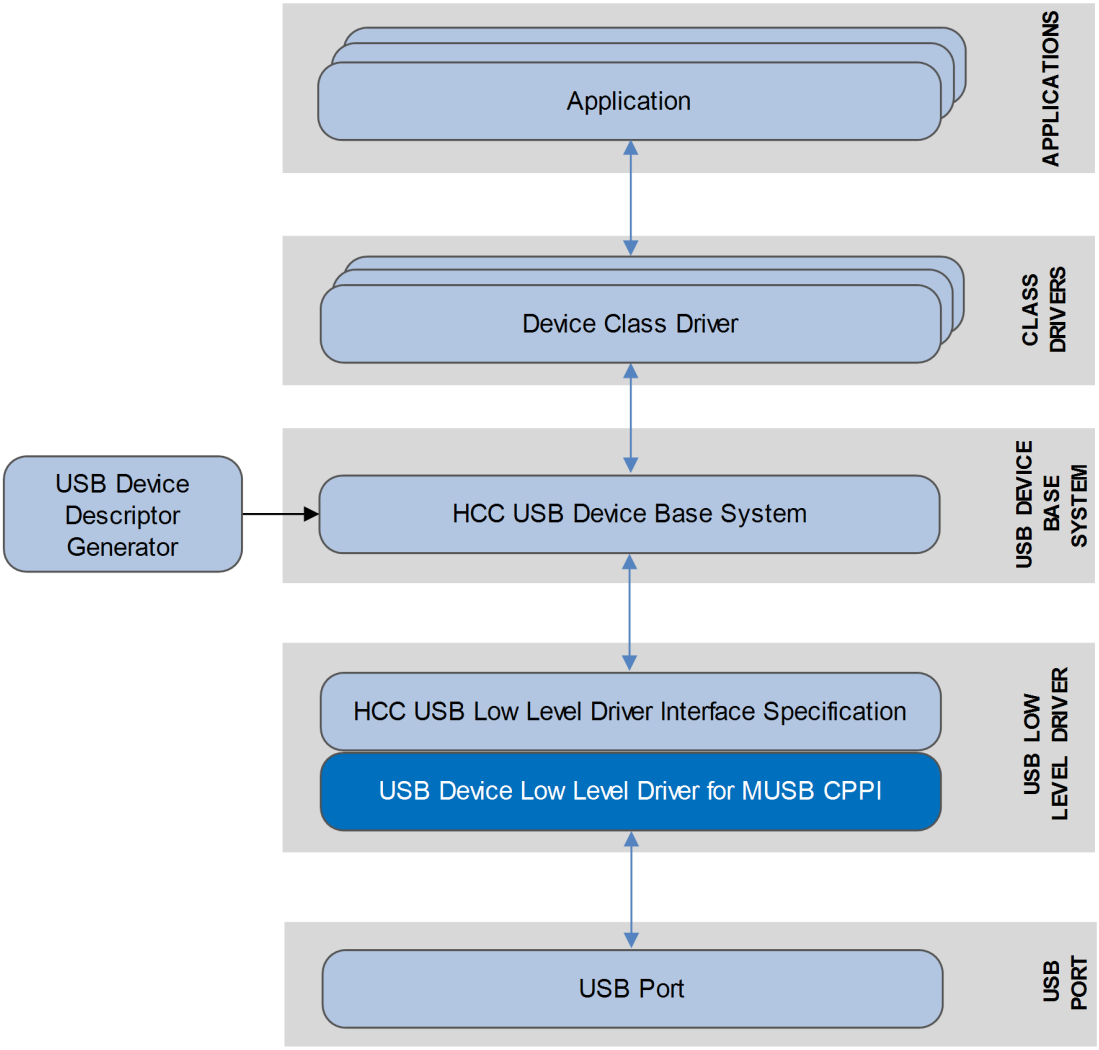
The component sections are as follows:

- Introduction – describes the main elements of the module.
- Feature Check – summarizes the main features of the module as bullet points.
- Packages and Documents – the *Packages* section lists the packages that you need in order to use this module. The *Documents* section lists the relevant user guides.
- Change History – lists the earlier versions of this manual, giving the software version that each manual describes.

1.1 Introduction

This guide is for those who want to configure and use the HCC Embedded Low Level Driver for MUSB module with HCC's USB device stack. This module provides a USB device driver for Texas Instruments Incorporated AM1808/AM1810 and AM335x microcontrollers that have the Mentor Graphics® MUSB device core. The driver can handle all USB transfer types and, in conjunction with the USB device stack, can be used with any USB device class driver.

This package provides a low level driver for a USB stack, as shown below.



The low level driver is always started automatically by the USB device stack. The driver is linked to the stack at compile time because each low level driver uses the same function names. This also means that only one driver can run in a system.

1.2 Feature Check

The main features of the low level driver are the following:

- Conforms to the HCC Advanced Embedded Framework.
- Designed for integration with both RTOS and non-RTOS based systems.
- Conforms to HCC's USB Device Low Level Driver Specification.
- Integrated with the HCC USB device stack and all its class drivers.
- Supports Texas Instruments Incorporated AM1808/AM1810 and AM335x microcontrollers that have the Mentor Graphics® MUSB device core.
- Compatible with other MCUs that use the Mentor Graphics® MUSB device controller.
- 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.
usbd_base	The USB device base package. Its source code includes the USB Driver device core.
usbd_drv_musb_cppi	The MUSB CPPI low level driver package described by this document.
util_hcc_mem	The HCC memory management utility.

Documents

For an overview of HCC's embedded USB stacks, see Product Information on the main HCC website.

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 Embedded USB Device Base System User Guide

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

HCC USB Device Low Level Driver for MUSB CPPI User Guide

This is this document.

1.4 Change History

To download this manual as a PDF, see USB Device PDFs.

For the history of changes made to the package code itself, see History: usbd_drv_musb_cppi.

The current version of this manual is 1.00.

Manual version	Date	Software version	Reason for change
1.00	2019-02-12	1.11	First release.

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 Configuration File

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

2.2 Source Code

These files in the directory **src/usb-device/usb-drivers** are the source code files. **These files should only be modified by HCC**.

File	Description
usbd_dev.h	USB driver-specific header file.
usbd_musb_cppi.c	Source code.

2.3 Version File

The file **src/version/ver_usbd_musb_cppi.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.

2.4 Platform Support Package (PSP) Files

These files are in the directory **src/psp/target**. These provide functions and elements the core code may need to use, depending on the hardware. Currently there is just one set, in the directory **psp_template_AM335x**.

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.
- In the package these files are offset to avoid overwriting an existing implementation. Copy them to the root **hcc** directory for use.

The files are as follows:

File	Description
include/hcc_am33x_reg.h	Register definitions.
usbd_musb_hw/usbd_musb_hw.c	Function source code.
usbd_musb_hw/usbd_musb_hw.h	Function header file.

The PSP also has the following version files in **src/version**:

File	Description
ver_psp_proc_reg.h	Version of register definitions.
ver_psp_usbd_musb_hw.h	PSP version.

3 Configuration Options

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

USBD_MUSB_CPPI_MODE

The type of MUSB CPPI used. Select one of the following:

- 0 for DM84x.
- 1 (the default) for AM18x, OMAPL13x.

USBD_USE_USB

The USB core to use, 0 or 1.

> **Note:** The following three options only apply if USBD_MUSB_CPPI_MODE is 0.

USBD_CPPI_ISR_ID

The ID of CPPI DMA interrupts. The default is 17.

USBD_CPPI_ISR_PRIORITY

The priority of CPPI DMA interrupts. The default is 0.

USBD_ISR_ID

The USB interrupt ID.

- If USBD_MUSB_CPPI_MODE is 0 and USBD_USE_USB is set, this is set to 19.
- If USBD_MUSB_CPPI_MODE is 0 and USBD_USE_USB is set, this is set to 18.
- If USBD_MUSB_CPPI_MODE is not 0, this is set to 58.

USBD_ISR_PRIORITY

The interrupt priority. The default is 0.

NO_OF_HW_EP

The number of hardware endpoints. The options are:

- 8 for AM18x (the default). This gives 4 TX and 4 RX endpoints.
- 30 for DM84x.

USBD_USE_DMA

This specifies whether to use DMA:

- 0: (the default) use MUSB core and **memcpy()** to FIFO.
- 1: use CPPI DMA.

USBD_USE_HL_PD_IT

The type of interrupt to use:

- 1 (the default): use Highlander or PD interrupts. "Highlander" interrupts are used for MODE = 0, "PD" interrupts are used in MODE = 1.
- 0: use Mentor interrupts.

USBD_ENABLE_VBUS_DETECTION

Enable or disable VBUS detection. The default is 1.

USBD_VBUS_DETECT_INTERVAL

The VBUS detection time. This is the time interval in ms for VBUS detection. The default is 100.

In the worst case VBUS valid and invalid (connect and disconnect) is detected a maximum of USBD_VBUS_DETECT_INTERVAL ms after it really happens.

USBD_VBUS_STACK_SIZE

The VBUS detection task stack size. The default is 1024.

4 Integration

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

4.1 OS Abstraction Layer

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	0
Events	0
ISRs	2

4.2 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. For full details of its functions, see the *HCC Base Platform Support Package User Guide*.

The module makes use of the following standard PSP function:

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.

The module makes use of the following PSP functions, provided by the PSP to perform particular tasks. Their design makes it easy for you to port them to work with your hardware solution. The package includes samples for the AM335x family in the **usbd_musb_hw.c** file.

Function	Description
usbd_musb_hw_init()	Initializes the device.
usbd_musb_hw_start()	Starts the device.
usbd_musb_hw_stop()	Stops the device.
usbd_musb_hw_delete()	Deletes the device, releasing the associated resources.
usbd_musb_pup_on_off()	Enables or disables USB pull-up.

These are described in the sections which follow.

usbd_musb_hw_init

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

> **Note:** Call this function first.

Format

```
int usbd_musb_hw_init ( void )
```

Arguments

None.

Return Values

Return value	Description
USBD_SUCCESS	Successful execution.
USBD_ERROR	Operation failed.

usbd_musb_hw_start

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

> **Note:** Call **usbd_musb_hw_init()** before this.

Format

```
int usbd_musb_hw_start ( void )
```

Arguments

None.

Return Values

Return value	Description
USBD_SUCCESS	Successful execution.
USBD_ERROR	Operation failed.

usbd_musb_hw_stop

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

Format

```
int usbd_musb_hw_stop ( void )
```

Arguments

None.

Return Values

Return value	Description
USBD_SUCCESS	Successful execution.
USBD_ERROR	Operation failed.

usbd_musb_hw_delete

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

Format

```
int usbd_musb_hw_delete ( void )
```

Arguments

None.

Return Values

Return value	Description
USBD_SUCCESS	Successful execution.
USBD_ERROR	Operation failed.

usbd_musb_pup_on_off

This function is provided by the PSP to enable or disable USB pull-up.

Format

```
void usbd_musb_pup_on_off ( uint8_t on )
```

Arguments

Parameter	Description	Type
on	The pull-up state flag. Set this to 1 to enable pull-up, 0 to disable it.	uint8_t

Return Values

None.