USB Device Low Level Driver for MUSB User Guide

Version 1.30

For use with USBD Low Level Driver for MUSB versions 1.06 and above

All rights reserved. This document and the associated software are the sole property of HCC Embedded. Reproduction or duplication by any means of any portion of this document without the prior written consent of HCC Embedded is expressly forbidden.

HCC Embedded reserves the right to make changes to this document and to the related software at any time and without notice. The information in this document has been carefully checked for its accuracy; however, HCC Embedded makes no warranty relating to the correctness of this document.

Table of Contents

1 System Overview

This chapter contains the fundamental information for this module.

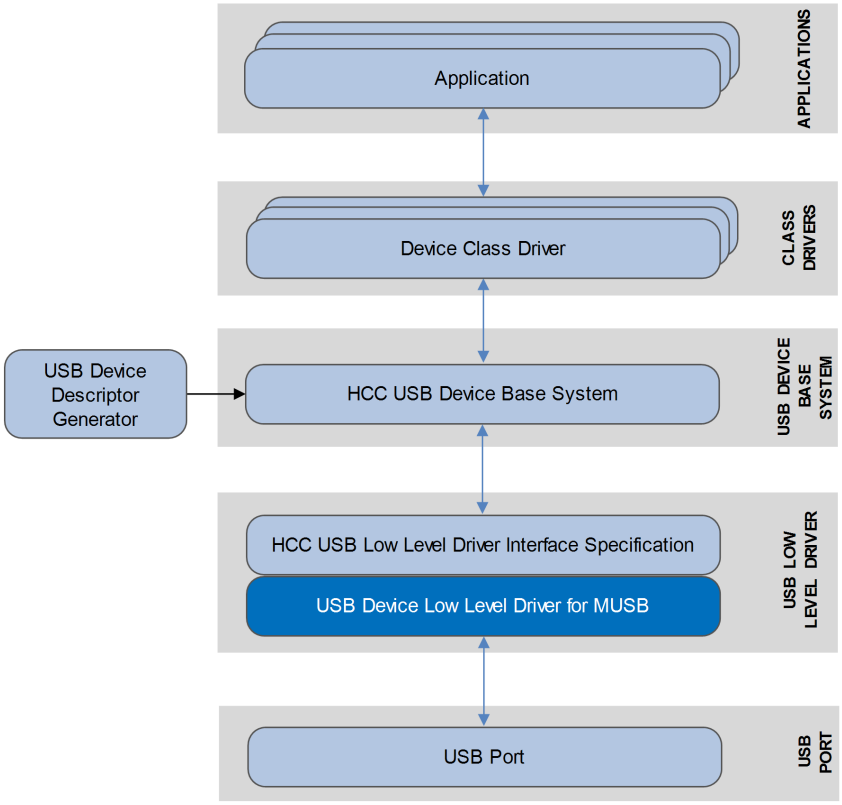
The component sections are as follows:

- 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 Analog Devices Blackfin® BF60x 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 all Analog Devices Blackfin® BF60x microcontrollers that have the Mentor Graphics® MUSB device core.
- Compatible with Blackfin® 5xx series controllers.
- 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	The MUSB low level driver package described by this document.

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 User Guide

This is this document.

1.4 Change History

This section describes past changes to this manual.

- To download this manual or a PDF describing an earlier software version, see USB Device PDFs.
- For the history of changes made to the package code itself, see History: usbd_drv_musb.

The current version of this manual is 1.30. The full list of versions is as follows:

Manual version	Date	Software version	Reason for change
1.30	2019-04-15	1.06	Added USBD_ADVANCE_TX_FROM_ISR configuration option.
1.20	2018-04-06	1.03	Corrected *OS Abstraction Layer* table: added ISR.
1.10	2017-06-16	1.02	New *Change History* format.
1.00	2015-11-16	1.01	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.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.c	Source code.

2.3 Version File

The file **src/version/ver_usbd_musb.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_bf60x/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 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/psp_bf60x_regs.h	Register definitions.
usbd_musb/psp_usbd_musb.c	Function source code.
usbd_musb/psp_usbd_musb.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.h	PSP version.

3 Configuration Options

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

USBD_ISR_ID

The HCC ID for USB0 used as a device. The default is PSP_ISR_USB_DEVICE_ID.

USBD_ISR_PRIORITY

The ISR priority. The default is 7.

USBD_USE_HIGH_SPEED

Keep this at the default of 1 if the USB controller is a high speed controller. Set it to 0 for full speed only.

USBD_ADVANCE_TX_FROM_ISR

This allows you to enable the driver to advance multi-packet Tx transfers from ISR rather than letting the user (class driver) advance the transfer by calling **usbd_transfer_status()**.

There are two options:

- 1 (the default) – advances multi-packet Tx transfers from the ISR. The advantage is that this is faster, but more time is spent on ISR execution. The user (class driver) is only notified when the multi-packet transfer is complete (that is, all packets have been transmitted over USB). This results in less overhead (context switches), therefore higher effective transfer speed. In return for that the program execution might spend significantly more time in ISR as a large amount of data will possibly be copied from within the ISR.
- 0 – the user (class driver) advances multi-packet Tx transfers. The advantage is that this minimizes ISR execution time, but the performance is slower.

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	0
Mutexes	1
Events	0
ISRs	1

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_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 BF6x family in the **psp_usbd_musb.c** file.

Function	Description
psp_usbd_musb_init()	Initializes the device.
psp_usbd_musb_start()	Starts the device.
psp_usbd_musb_stop()	Stops the device.
psp_usbd_musb_delete()	Deletes the device, releasing the associated resources.

These are described in the sections which follow.

psp_usbd_musb_init

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

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

Format

```
int psp_usbd_musb_init ( void )
```

Arguments

None.

Return Values

Return value	Description
USBD_SUCCESS	Successful execution.
USBD_ERROR	Operation failed.

psp_usbd_musb_start

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

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

Format

```
int psp_usbd_musb_start ( void )
```

Arguments

None.

Return Values

Return value	Description
USBD_SUCCESS	Successful execution.
USBD_ERROR	Operation failed.

psp_usbd_musb_stop

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

Format

```
int psp_usbd_musb_stop ( void )
```

Arguments

None.

Return Values

Return value	Description
USBD_SUCCESS	Successful execution.
USBD_ERROR	Operation failed.

psp_usbd_musb_delete

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

Format

```
int psp_usbd_musb_delete ( void )
```

Arguments

None.

Return Values

Return value	Description
USBD_SUCCESS	Successful execution.
USBD_ERROR	Operation failed.