# USB Device Low Level Driver for LPC User Guide

Version 1.10

For use with USBD Low Level Driver for LPC versions 2.05 and above

**Date:** 16-Jun-2017 16:44

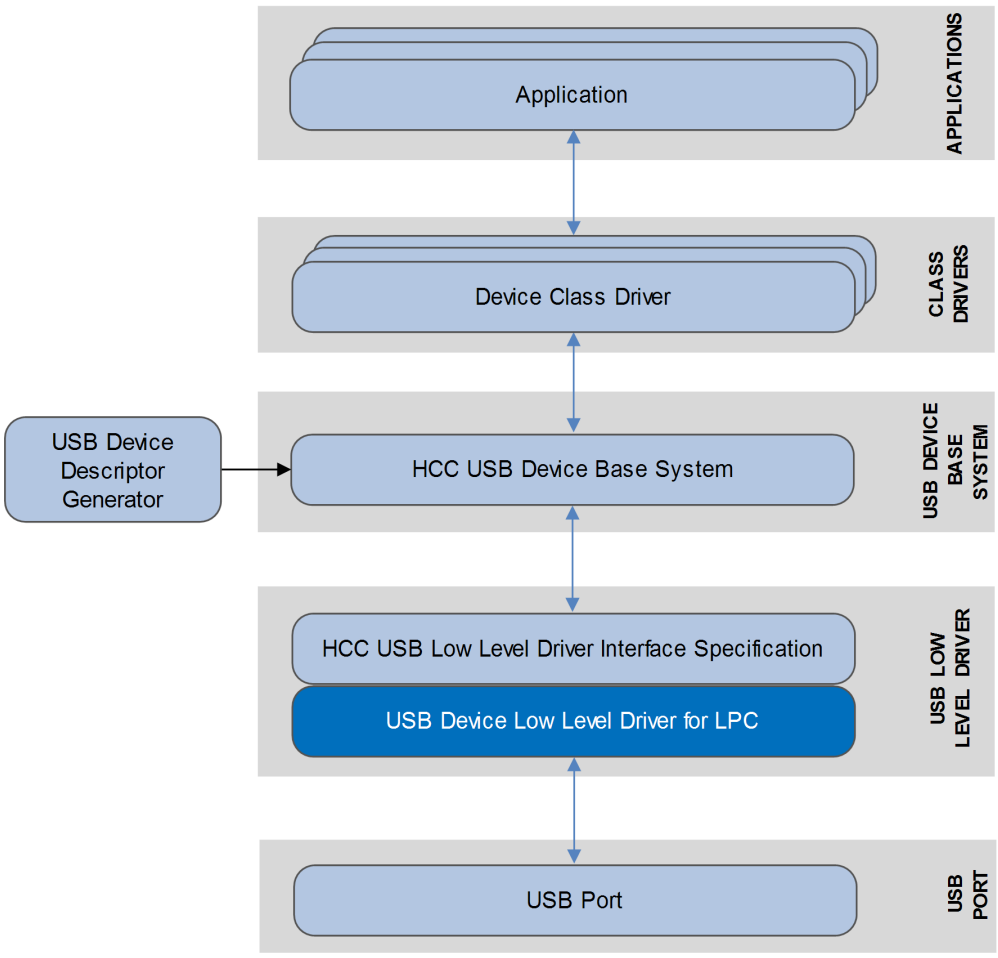
# Table of Contents

# 1 System Overview

## 1.1 Introduction

This guide is for those who want to configure and use the HCC Embedded Low Level Driver for LPC module with HCC's USB device stack. This module provides a USB device driver for LPC microcontrollers from NXP Semiconductors. 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 LPC controllers.
- Supports all USB transfer types: control, bulk, interrupt, and isochronous.

## 1.3 Packages and Documents

### Packages

This table 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_lpc** | The LPC 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 LPC User Guide**

This is this document.

## 1.4 Change History

This section describes past changes to this manual.

- To view or download earlier manuals, see Archive: USB Device Low Level Driver for LPC User Guide .
- For the history of changes made to the package code itself,see History: usbd_drv_lpc.

The current version of this manual is 1.10. The full list of versions is as follows:

| Manual version | Date | Software version | Reason for change |
|---|---|---|---|
| 1.10 | 2017-06-16 | 2.05 | New *Change History* format. |
| 1.00 | 2015-04-24 | 2.05 | 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 files except the PSP files.

## 2.1 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_lpc.c** | Source file for LPC code. |
| **usbd_lpc_regs.h** | Header file for register functions. |

## 2.2 Platform Support Package (PSP) Files

These files are in the directory **src/psp/target/usbd_lpc**. 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 |
|---|---|
| **lpc_otg_i2c.c** | Functions source code. |
| **lpc_otg_i2c.h** | Functions header file. |
| **usbd_lpc_config.c** | Clock and register configuration. |
| **usbd_lpc_config.h** | Configuration header file. |

## 2.3 Version File

The file **src/version/ver_usbd_lpc.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.

# 3 Integration

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

## 3.1 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 | 0 |
| Events | 1 |
| ISRs | 1 |

## 3.2 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 and macros, see the *HCC Base Platform Support Package User Guide*.

The module makes use of the following standard PSP macros:

| Macro | Package | Element | Description |
|---|---|---|---|
| PSP_RD_LE16 | psp_base | psp_endianness | Reads a 16 bit value stored as big-endian from a memory location. |
| PSP_RD_LE32 | psp_base | psp_endianness | Reads a 32 bit value stored as little-endian from a memory location. |
| PSP_WR_LE32 | psp_base | psp_endianness | Reads a 32 bit value stored as little-endian from a memory location. |

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. These functions are provided by the PSP to perform various tasks. Their design makes it easy for you to port them to work with your hardware solution. The package includes samples in the PSP file **src/psp/target/usbd_lpc/lpc_otg_i2c.c**:

| Function | Description |
|---|---|
| **otg_i2c_init()** | Initializes the device. |
| **otg_i2c_start()** | Starts the device. |
| **otg_i2c_stop()** | Stops the device. |
| **otg_i2c_delete()** | Deletes the device, releasing the associated resources. |
| **otg_i2c_read()** | Reads from the device. |
| **otg_i2c_write()** | Writes to the device. |

These functions are described in the following sections.

## otg_i2c_init

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

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

### Format

```
int otg_i2c_init ( void )
```

### Arguments

None.

### Return Values

| Return value | Description |
|---|---|
| USBD_SUCCESS | Successful execution. |
| USBD_ERROR | Operation failed. |

## otg_i2c_start

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

> **Note:** Call **otg_i2c_init()** before this.

### Format

```
int otg_i2c_start ( void )
```

### Arguments

None.

### Return Values

| Return value | Description |
|---|---|
| USBD_SUCCESS | Successful execution. |
| USBD_ERROR | Operation failed. |

## otg_i2c_stop

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

### Format

```
int otg_i2c_stop( void )
```

### Arguments

None.

### Return Values

| Return value | Description |
|---|---|
| USBD_SUCCESS | Successful execution. |
| USBD_ERROR | Operation failed. |

## otg_i2c_delete

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

### Format

```
int otg_i2c_delete( void )
```

### Arguments

None.

### Return Values

| Return value | Description |
|---|---|
| USBD_SUCCESS | Successful execution. |
| USBD_ERROR | Operation failed. |

## otg_i2c_read

This function is provided by the PSP to read from the device.

### Format

```
uint8_t otg_i2c_read ( uint8_t addr )
```

### Arguments

| Parameter | Description | Type |
|---|---|---|
| addr | The address to read from. | uint8_t |

### Return Values

| Return value | Description |
|---|---|
| USBD_SUCCESS | Successful execution. |
| USBD_ERROR | Operation failed. |

## otg_i2c_write

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

### Format

```
void otg_i2c_write (
    uint8_t   addr,
    uint8_t   val )
```

### Arguments

| Parameter | Description | Type |
|---|---|---|
| addr | The address to write to. | uint8_t |
| val | The value to be written. | uint8_t |

### Return Values

None.