USB Device Low Level Driver for Atmel SAM User Guide

Version 1.20

For use with USBD Low Level Driver for Atmel® SAM versions 2.03 and above

Date: 06-Apr-2018 11:52

Table of Contents

1 System Overview

This chapter contains the fundamental information for this module.

The component sections are as follows:

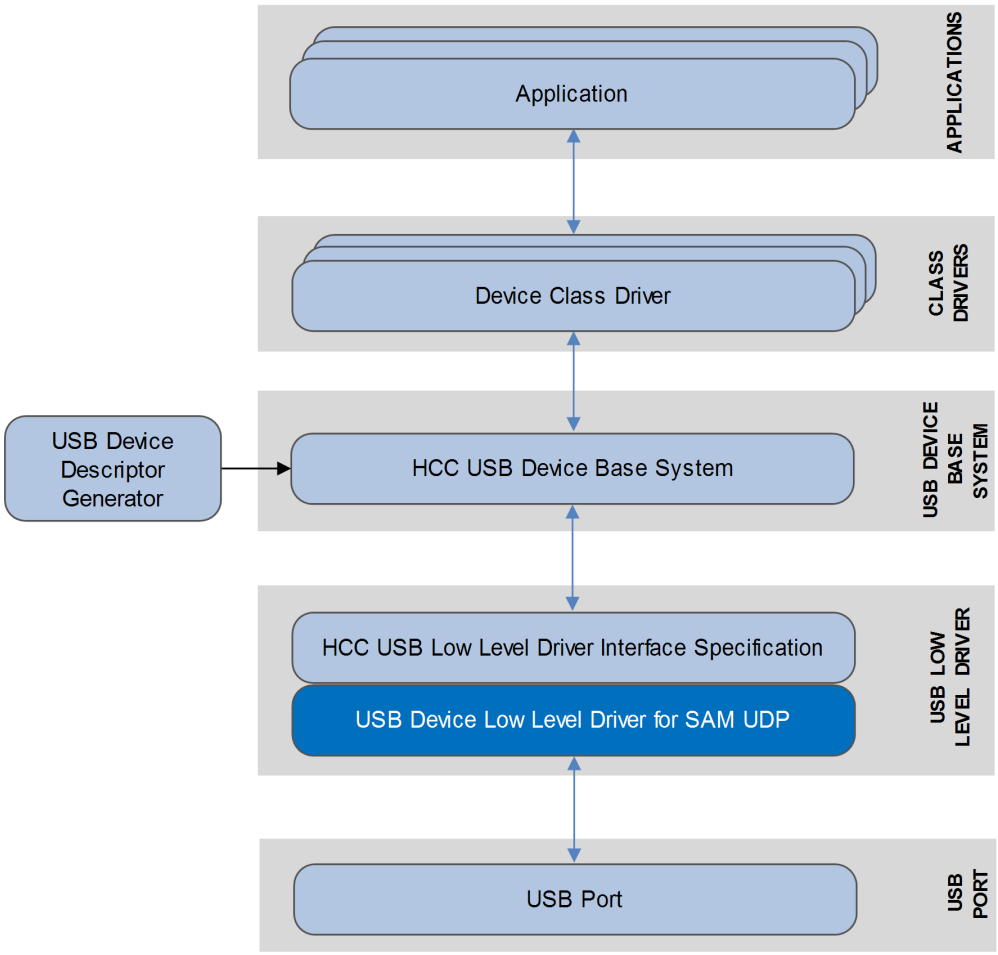
- Introduction – describes the main elements of the module.
- Feature Check – summarizes the main features of the module as bullet points.
- Packages and Documents – the *Packages* section lists the packages that you need in order to use this module. The *Documents* section lists the relevant user guides.
- 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 Atmel® SAM module with HCC's USB device stack. This module provides a USB device driver for Atmel® SAM controllers that have a USB 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 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_sam_udp	The Atmel® SAM 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.

USB Device Low Level Driver for Atmel® SAM User Guide

This is this document.

1.4 Change History

This section describes past changes to this manual.

- To view or download manuals, see USB Device PDFs.
- For the history of changes made to the package code itself, see History: usbd_drv_sam_udp.

The current version of this manual is 1.20. The full list of versions is as follows:

Manual version	Date	Software version	Reason for change
1.20	2018-04-06	2.03	Corrected *OS Abstraction Layer* table: removed event.
1.10	2015/11/12	2.03	Changed *PSP Configuration* and *PSP Porting* sections.
1.00	2015/04/22	2.03	First online 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 PSP files.

2.1 Source Code Files

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

File	Description
usbd_dev.h	USB driver-specific header file.
usbd_sam.c	Source code.
usbd_sam_regs.h	Register definitions.

2.2 Platform Support Package (PSP) Files

These files are in the directory **src/psp/target/usbd_sam**. 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_usbd_sam.c	Functions source code.
psp_usbd_sam.h	Functions header file. This has the PSP configuration options.

2.3 Version File

The file **src/version/ver_usbd_sam.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	0
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.

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 in the file **src/psp/target/usbd_sam/psp_usbd_sam.c**:

Function	Description
psp_usbd_hw_init()	Initializes the device.
psp_usbd_hw_start()	Starts the device.
psp_usbd_hw_stop()	Stops the device.
psp_usbd_hw_delete()	Deletes the device, releasing the associated resources.

These are described in the following sections.

PSP Configuration

Set the PSP configuration options in the file **src/psp/target/usbd_sam/psp_usbd_sam.h**. This section lists the available configuration options and their default values.

ON_CHIP_PULL_UP

Keep the default of 1 if the MCU has on-chip pull-up resistors. Otherwise, set this to 0.

> **Note:** If on-chip pull-up is not available you must provide a function to control the pull-up resistor on the USB interface. This code must be included in the **usbd_pup_on_off()** function documented in the *HCC USB Device Low Level Driver Interface Specification*.

HCC_UDP_BASE

The base address of the UDP module. The default is AT91C_BASE_UDP.

NO_OF_HW_EP

The number of hardware endpoints on the device, including EP0. The default is 6.

USBD_SAM_ISR_ID

The ISR ID. The default is AT91C_ID_UDP.

USBD_SAM_IT_PRIO

The ISR priority. The default is 2.

psp_usbd_hw_init

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

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

Format

```
int psp_usbd_hw_init ( void )
```

Arguments

None.

Return Values

Return value	Description
USBD_SUCCESS	Successful execution.
USBD_ERROR	Operation failed.

psp_usbd_hw_start

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

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

Format

```
int psp_usbd_hw_start ( void )
```

Arguments

None.

Return Values

Return value	Description
USBD_SUCCESS	Successful execution.
USBD_ERROR	Operation failed.

psp_usbd_hw_stop

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

Format

```
int psp_usbd_hw_stop ( void )
```

Arguments

None.

Return Values

Return value	Description
USBD_SUCCESS	Successful execution.
USBD_ERROR	Operation failed.

psp_usbd_hw_delete

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

Format

```
int psp_usbd_hw_delete( void )
```

Arguments

None.

Return Values

Return value	Description
USBD_SUCCESS	Successful execution.
USBD_ERROR	Operation failed.