Atmel AVR4903: ASF - USB Device HID Mouse Application

Atmel Microcontrollers

Application Note

Features

- **USB 2.0 compliance**
  - **Chapter 9 compliance**
  - **HID compliance**
  - **Low-speed (1.5Mb/s) and full-speed (12Mb/s) data rates**
- **Standard USB HID mouse implementation**
  - **XY move**
  - **Left, right, middle buttons**
  - **Wheel scrolling**
- **Remote wake up support**
- **USB bus powered support**
- **Real time (OS compliance, interrupt driven)**
- **Supports 8-bit and 32-bit Atmel® AVR® MCUs**

1 Introduction

The aim of this document is to provide an easy way to integrate a USB mouse device application on a new or existing project.

Figure 1-1. USB device HID mouse application.

Rev. 8409A-AVR-08/11

2 Abbreviations

ASF:	AVR Software Framework
CD:	Composite device: a USB device with more than one interface
FS:	USB full speed
HID:	Human interface device
HS:	USB high speed
LS:	USB low speed
UDC:	USB device controller
UDD:	USB device descriptor
UDI:	USB device interface
USB:	Universal serial bus
SOF:	Start of frame
ZLP:	Zero length packet

3 Overview

This document includes four sections for all types of requirements when building a USB device HID mouse application:

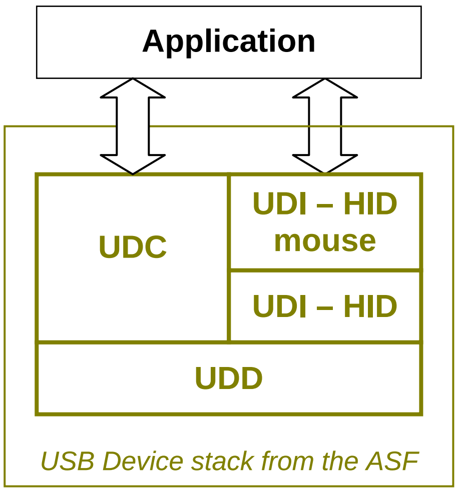
- **Quick start**
  Describes how to start a ready to use HID mouse device example
- **Example description**
  Describes a HID mouse device example
- **Building a USB device mouse**
  Describes how to add a USB device mouse interface in a project
- **Mouse in a USB composite device**
  Describes how to integrate a mouse interface in a composite device project

For all these sections, it is recommended to know the main module organization of a HID mouse application:

- User application
- USB device interface HID mouse (UDI-HID mouse)
- USB device interface HID (UDI-HID)
- USB device controller (UDC)
- USB device driver (UDD)

For more advanced information concerning the USB stack implementation, please refer to the application note, "AVR4900: ASF - USB Device Stack."

Figure 3-1. USB device HID mouse architecture.



NOTE The USB device stack is found in the ASF in the common/services/usb directory.

4 Quick start

USB device mouse examples are available in Atmel® AVR Studio® 5.

AVR Studio 5 allows the creation of a *New Example Project*. In the examples list, select a *USB device HID mouse example* corresponding to the Atmel board used.

A *USB device HID mouse example* is available for all AVR parts with USB hardware interface.

The project does not require any modification and only needs to be compiled, loaded, and run.

The USB device HID mouse example may run with or without external power (only USB cable connection), depending on the hardware used.

4.1 User interface

The user interface depends on switches and LEDs available on the hardware.

With two LEDs available:

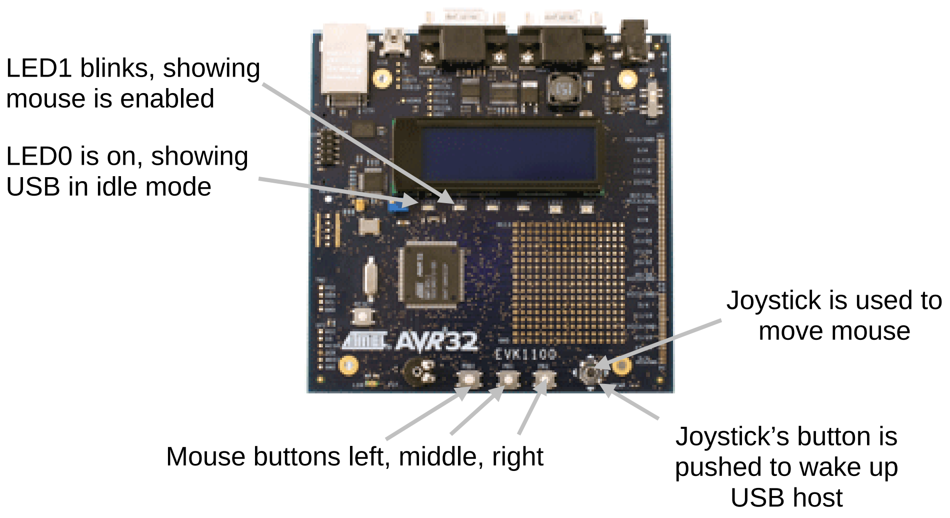
- LED0 is on when the USB device is in idle mode and is off in suspend mode
- LED1 blinks showing that the mouse interface is enabled by the USB host

The following actions can be done using switches or sensors:

- move pointer on X and Y axes
- move scroll wheel
- click left, right, middle buttons
- remote wake up

The user interface description (specific to the board) is defined at the end of the *ui.c* source file. This file is available within the project folder.

Figure 4-1. Example with EVK1100 board.


LED1 blinks, showing mouse is enabled
LED0 is on, showing USB in idle mode
Joystick is used to move mouse
Mouse buttons left, middle, right
Joystick's button is pushed to wake up USB host

5 Example description

The ASF provides a USB device mouse example for various Atmel AVR products.

All these examples share common files.

File descriptions and execution behavior are described below in this section.

5.1 Example content

Table 5-1 introduces a summary of the main files included in the *USB device HID mouse example*. These files are associated with the modules described in Figure 3-1. **USB device HID mouse architecture.**

Table 5-1. USB Device HID mouse example files.

Modules	Files	ASF paths	Description
Application	main.c ui.c conf_usb.h	Examples folder	Main loop Set up hardware switches to mouse operations USB device configuration
UDI HID mouse	udi_hid_mouse.c/h	common/services/usb/class/hid/device/mouse/	Standard HID mouse class implementation
	udi_hid_mouse_desc.cudi_hid_mouse_conf.h	common/services/usb/class/hid/device/mouse/	USB descriptors for a USB device with HID mouse interface (not applicable for USB composite device)
UDI HID	udi_hid.c/h	common/services/usb/class/hid/device/	Common HID class implementation
	usb_protocol_hid.h	common/services/usb/class/hid/	HID protocol constants
UDC	udc.c/h udc_desc.h udi.h udd.h	common/services/usb/udc/	USB device core
	usb_protocol.h usb_atmel.h	common/services/usb/	USB protocol constants
UDD	usbb_device.c/h usbc_device.c/h usb_device.c/h	avr32/drivers/usbb/ avr32/drivers/usbc/ xmega/drivers/usb/	USB drivers

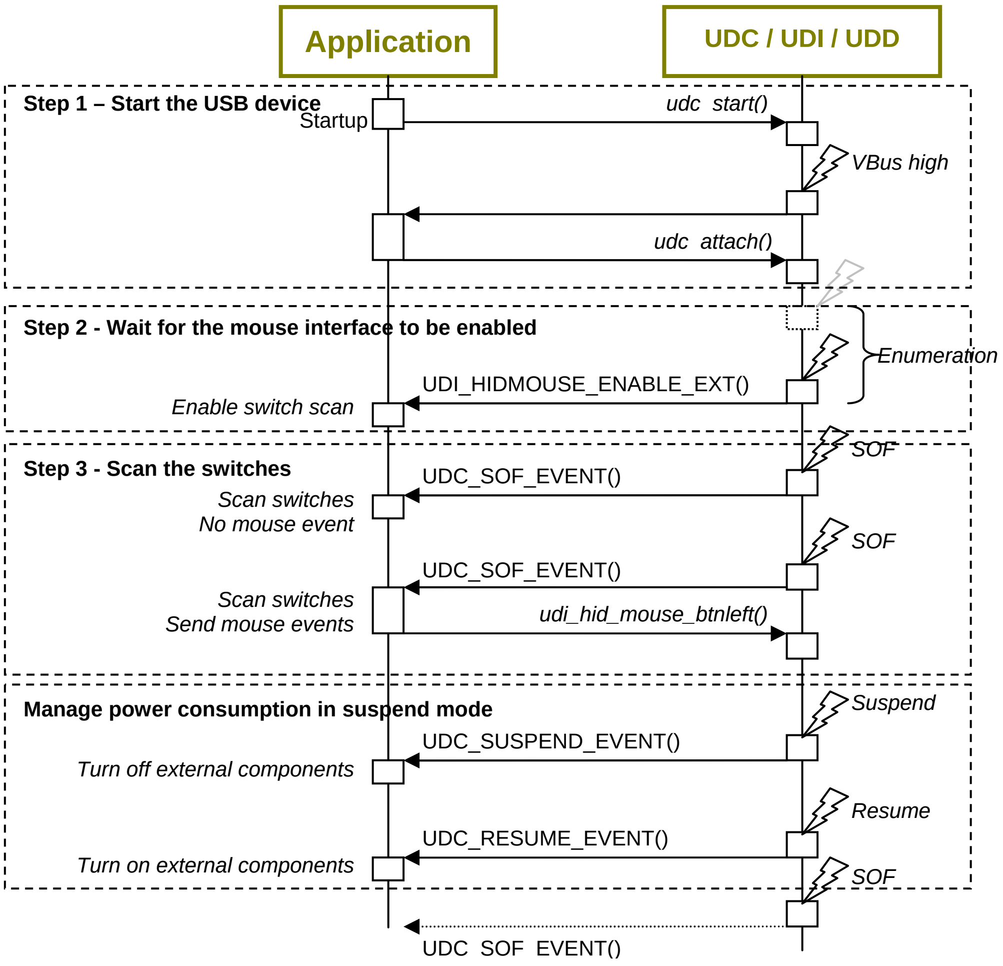
5.2 Example behavior

The *main.c* and *ui.c* files implement the user interface of the HID mouse application.

The implementation is comprised of three steps:

1. Start the USB Device
```
udc_start();
udc_attach(); // Must be called when the USB cable is plugged
              // Cable plugged is detected via VBus events
```
2. Wait for the HID mouse interface to be enabled via a callback
```
UDI_HIDMOUSE_ENABLE_EXT() // Authorize the HID mouse events
```
3. Scan the switches and send mouse events, if any
```
If(switch_left_down) udi_hid_mouse_btnleft(HID_MOUSE_BTN_DOWN);
```
Tip: To simplify the implementation, the SOF event is used to scan switches each 1ms.

Figure 5-1. Example behavior sequence.



6 Building a USB device mouse

The USB device mouse modules are available in Atmel AVR Studio 5, and can be imported into an AVR Studio 5 project.

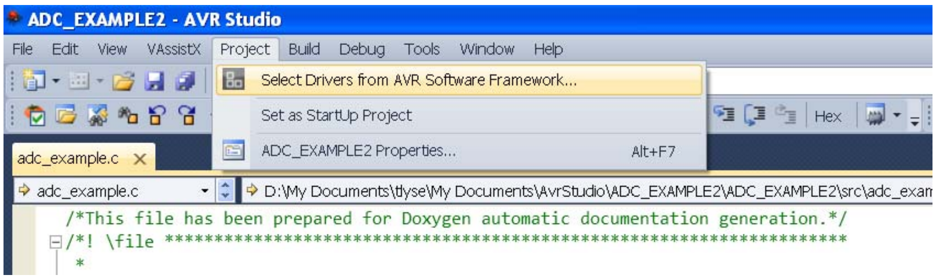
This section describes how to add a USB device mouse in a project:

1. Import the USB mouse module.
2. Configure personal USB parameters.
3. Call USB routines to run the USB device.

6.1 Import USB module

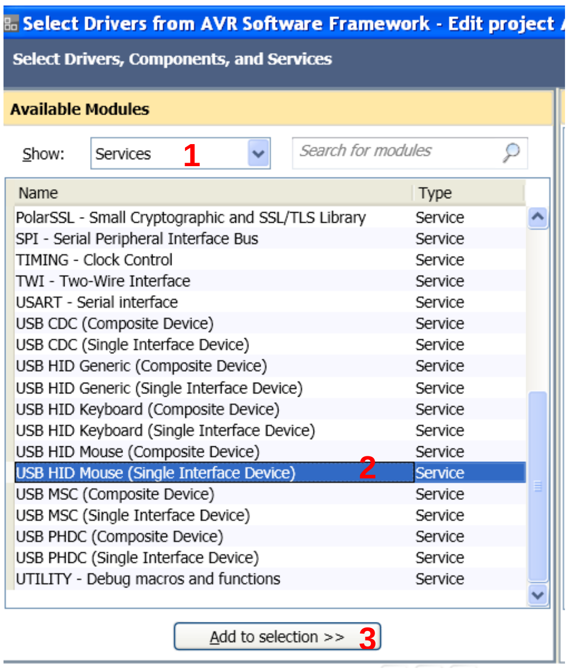
To import the USB HID mouse module, follow the instructions below:

1. Open or create your project:
2. From the Project menu, choose "Select Drivers from AVR Software Framework."


ADC_EXAMPLE2 - AVR Studio
File
Edit
View
VAssistX
Project
Build
Debug
Tools
Window
Help
Select Drivers from AVR Software Framework...
Set as StartUp Project
ADC_EXAMPLE2 Properties...
Alt+F7
Hex
adc_example.c
adc_example.c
D:\My Documents\tlyse\My Documents\AvrStudio\ADC_EXAMPLE2\ADC_EXAMPLE2\src\adc_exam

```
/*This file has been prepared for Doxygen automatic documentation generation.*/
/*! \file ************************************************************************
 *
```

3. Select Services, choose USB HID Mouse (Single Interface Device), and click on the "Add to selection" button.


Select Drivers from AVR Software Framework - Edit project
Select Drivers, Components, and Services
Available Modules
Show:
Services
1
Search for modules
2
Add to selection >>
3

Name	Type
PolarSSL - Small Cryptographic and SSL/TLS Library	Service
SPI - Serial Peripheral Interface Bus	Service
TIMING - Clock Control	Service
TWI - Two-Wire Interface	Service
USART - Serial interface	Service
USB CDC (Composite Device)	Service
USB CDC (Single Interface Device)	Service
USB HID Generic (Composite Device)	Service
USB HID Generic (Single Interface Device)	Service
USB HID Keyboard (Composite Device)	Service
USB HID Keyboard (Single Interface Device)	Service
USB HID Mouse (Composite Device)	Service
USB HID Mouse (Single Interface Device)	Service
USB MSC (Composite Device)	Service
USB MSC (Single Interface Device)	Service
USB PHDC (Composite Device)	Service
USB PHDC (Single Interface Device)	Service
UTILITY - Debug macros and functions	Service

6.2 USB configuration

All USB stack configurations are stored in the ***conf_usb.h*** file in the application module. These configurations are simple and do not require any specific USB knowledge.

There is one configuration section for each USB module: UDC, UDI, and UDD.

The UDC configuration possibilities are described in the application note, "AVR4900: ASF – USB Device Stack," Section 7.1.1 - USB device configuration.

The UDD configuration possibilities are described in the application note, "AVR4900: ASF – USB Device Stack," Section 7.1.3 - USB driver configuration.

The UDI, which is the mouse interface, does not require any configuration.

6.3 USB implementation

This section describes the source code to add and to run a USB device mouse application.

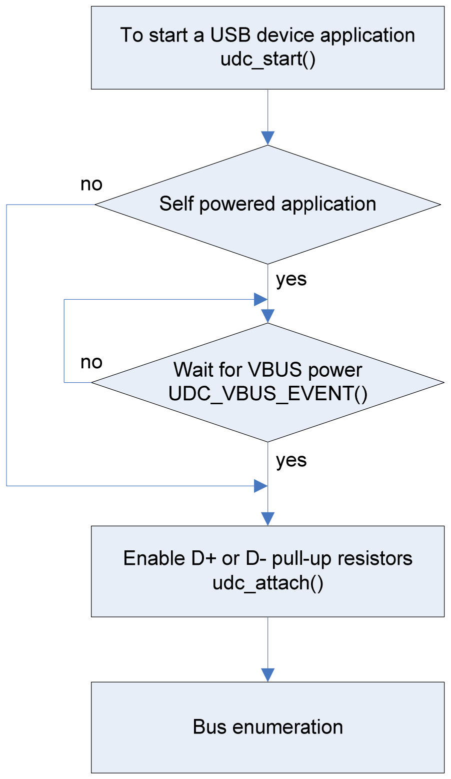
The implementation comprises three steps:

1. Start the USB device.
2. Wait for the HID mouse interface to be enabled by the host.
3. Scan the sensors and send USB mouse events, if any.

6.3.1 USB device control

Only two function calls are needed to start a USB device application (see Figure 6-1).

Figure 6-1. USB device application sequence.



NOTE In the case of a new project, the USB stack requires interrupts to be enabled and the clock and sleepmgr services to be initialized.

Example

```
<conf_usb.h>
#define UDC_VBUS_EVENT(b_vbus_high) \
       vbus_event(b_vbus_high)

<main C file>:
main() {
   // Authorize interrupts
   irq_initialize_vectors();
   cpu_irq_enable();
   // Initialize the sleep manager service
   sleepmgr_init();
   // Initialize the clock service
   sysclk_init();
   // Enable USB stack device
   udc_start();
}

vbus_event(b_vbus_high) {
   if (b_vbus_high) {
       // Connect USB device
       udc_attach();
   }else{
       // Disconnect USB device
       udc_detach();
   }
}
```

6.3.2 USB interface control

After the device enumeration (detecting and identifying USB devices), the USB host starts the device configuration. When the USB mouse interface from the device is accepted by the host, the USB host enables this interface and the *UDI_HIDMOUSE_ENABLE_EXT()* callback function is called.

When the USB device is unplugged or is reset by the USB host, the USB interface is disabled and the *UDI_HIDMOUSE_DISABLE_EXT()* callback function is called.

Thus, it is recommended to enable/disable sensors used by the mouse in these functions.

Example

```
<conf_usb.h>
#define UDI_HIDMOUSE_ENABLE_EXT() \
       mouse_enable()
#define UDI_HIDMOUSE_DISABLE_EXT() \
       mouse_disable()

<main C file>:
mouse_enable() {
   // Enable sensor and start process to scan sensor
   …
   return true;
}
mouse_disable() {
   // Disable sensor and stop process to scan sensor
   …
}
```

6.3.3 USB mouse control

The USB HID mouse functions described in Table 6-1 allow the application to control the mouse.

Table 6-1. UDI HID mouse functions.

Declaration	Description
udi_hid_mouse_moveScroll(int8_t pos)	Sends a value at scroll wheel
udi_hid_mouse_moveY(int8_t pos_y)	Sends an Y axis value at mouse pointer
udi_hid_mouse_moveX(int8_t pos_x)	Sends an X axis value at mouse pointer
udi_hid_mouse_btnmiddle(bool b_state)	Sends a “middle click” event
udi_hid_mouse_btnright(bool b_state)	Sends a “right click” event
udi_hid_mouse_btnleft(bool b_state)	Sends a “left click” event

Example

```
<sensor C file>:
sensor_process() {
   if (is_sensor_move_up())
       udi_hid_mouse_moveY(sensor_get_move_value());
   if (is_button_left_press())
       udi_hid_mouse_btnleft(HID_MOUSE_BTN_DOWN);
   if (is_button_left_release())
       udi_hid_mouse_btnleft(HID_MOUSE_BTN_UP);
   ...
}
```

7 Mouse in a USB composite device

This section provides the HID mouse information to create a USB composite device together with the application note, “AVR4902 ASF - USB Composite Device.”

7.1 USB configuration

In addition to the USB configuration described in Section 6.2, the following values must be defined in the *conf_usb.h* file:

USB_DEVICE_EP_CTRL_SIZE: Endpoint control size. This must be:

- 8 for a low speed device
- 8, 16, 32, or 64 for a full speed device (8 is recommended to save RAM)
- 64 for a high speed device

UDI_HID_MOUSE_EP_IN*:* IN endpoint number used by the HID mouse interface.

UDI_HID_MOUSE_IFACE_NUMBER*:* Interface number of the HID mouse interface.

USB_DEVICE_MAX_EP*:* Total number of endpoints in the application. This must include one endpoint for the HID mouse interface.

7.2 USB descriptor

The USB device descriptor of the composite device, defined in the *conf_usb.h* file, must include the HID mouse interface descriptor:

```
//! Define structure of composite interfaces descriptor
#define    UDI_COMPOSITE_DESC_T                              \
    udi_hid_mouse_desc_t udi_hid_mouse;                      \
    ...

//! Fill composite interfaces descriptor for full speed
#define    UDI_COMPOSITE_DESC_FS                      \
    .udi_hid_mouse             = UDI_HID_MOUSE_DESC, \
    ...

//! Fill composite interfaces descriptor for high speed
#define    UDI_COMPOSITE_DESC_HS                      \
    .udi_hid_mouse             = UDI_HID_MOUSE_DESC, \
    ...

//! Fill interface APIs corresponding to interfaces descriptor
#define    UDI_COMPOSITE_API   \
    &udi_api_hid_mouse,       \
    …
```

8 Table of contents

Atmel Corporation
2325 Orchard Parkway
San Jose, CA 95131
USA
Tel: (+1)(408) 441-0311
Fax: (+1)(408) 487-2600
www.atmel.com

Atmel Asia Limited
Unit 01-5 & 16, 19F
BEA Tower, Milennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
HONG KONG
Tel: (+852) 2245-6100
Fax: (+852) 2722-1369

Atmel Munich GmbH
Business Campus
Parkring 4
D-85748 Garching b. Munich
GERMANY
Tel: (+49) 89-31970-0
Fax: (+49) 89-3194621

Atmel Japan
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chou-ku, Tokyo 104-0033
JAPAN
Tel: (+81) 3523-3551
Fax: (+81) 3523-7581