Overview

This Host HID example supports the mouse device and the keyboard device.
The application prints the mouse operation when the mouse device is attached. The application prints the pressed keyboard key when the keyboard is attached. Because this is a simple demo, special function keys and long press function are not supported.
The application supports three types of devices:

- The single mouse device or the single keyboard device.
- The composite device that contains one mouse and one keyboard.
- HUBs connected with one mouse device and one keyboard device.

System Requirement

Hardware requirements

- Mini/micro USB cable
- USB A to micro AB cable
- Hardware (Tower module/base board, and so on) for a specific device
- Personal Computer(PC)

Software requirements

- The project path is:
  <MCUXpresso_SDK_Install>/boards/<board>/usb_examples/usb_host_hid_mouse_keyboard/<rtos>/<toolchain>.

  Note

  The <rtos> is Bare Metal or FreeRTOS OS.

Getting Started

Hardware Settings

- The Jumper settings:

  J22 1-2, J11 5-6, J24 1-2 for micro USB connector. 1-2, J17 1-2, J24 2-3, and remove J11 5-6 for using TWR-SER mini USB connector.

  Note

  Set the hardware jumpers (Tower system/base module) to default settings.
  Host hid example doesn't support HID report descriptor analysis, this example assume that the device data are sent by specific order.
  For more detail, please refer to the code. For the device list we tested,
  please refer to chapter Peripheral devices tested with the USB Host stack in "SDK Release Notes xxxx(board name)".

Prepare the example

1. Download the program to the target board.
2. Power off the target board and power on again.
3. Connect devices to the board.

Note

For detailed instructions, see the appropriate board User's Guide.

Run the example

1. Connect the board UART to the PC and open the COM port in a terminal tool.
2. Plug in the HUB, mouse or keyboard device to the board. The attached information prints out in the terminal.
3. If one mouse is plugged in, the mouse operation information prints in the terminal when moving the mouse.

   The application prints the mouse operation information in one line. Each line contains the following sequential string: "Left Click", "Middle Click", "Right Click", "Right"/"Left" movement, "UP"/"Down" movement and "Wheel Down"/"Wheel Up" movement. White space replaces the above string if the mouse doesn't have the corresponding operation.

   For example when the mouse moves right and up,

   ```
   "                                   Right UP              "
   ```

   prints in the terminal.
4. If one keyboard is plugged in, the keyboard pressed key information prints in the terminal when operating the keyboard.

   for example: when F key is pressed, the

   ```
   'F'
   ```

   prints in the terminal.

The following picture is an example for attaching a HUB, a mouse, and a keyboard.

Figure 1: Attach one hub, one mouse and one keyboard

Note

The Host keyboard doesn't support the long pressed key. To implement the long press function, follow these steps:

1. Implement one timer and the interval is 10 ms, for example PIT.
2. Define one 6-byte array to count time.
3. Update the time array:
   (a) When the timer times out: For every key in the lastPressData, add the time. For example, if the second key is valid in the lastPressData, the second value of the time array adds 1.
   (b) When the lastPressData is updated: Update the time array to map the time with the lastPressData. For example, the second value of the time array is the time for the seconds of lastPressData.
4. When one time is greater than 5, the key is long pressed. Print the key and reset the time value for the next long press.