

## Objective

This example demonstrates the implementation of the Bluetooth Low Energy (BLE) HID over GATT Profile where the device operates as a HID Mouse.

---

## Overview

The design demonstrates the core functionality of the BLE Component configured as a HID Device (GATT Server). It simulates the mouse movement and button click in Boot and Protocol modes. Also, the design demonstrates how to handle a suspend event from the central device and enter Low-Power mode when suspended.

The project supports up to four connections.

## Requirements

**Tool:** [PSoC Creator™ 4.2](#) or later

**Programming Language:** C (Arm® GCC 5.4-2016-q2-update or later)

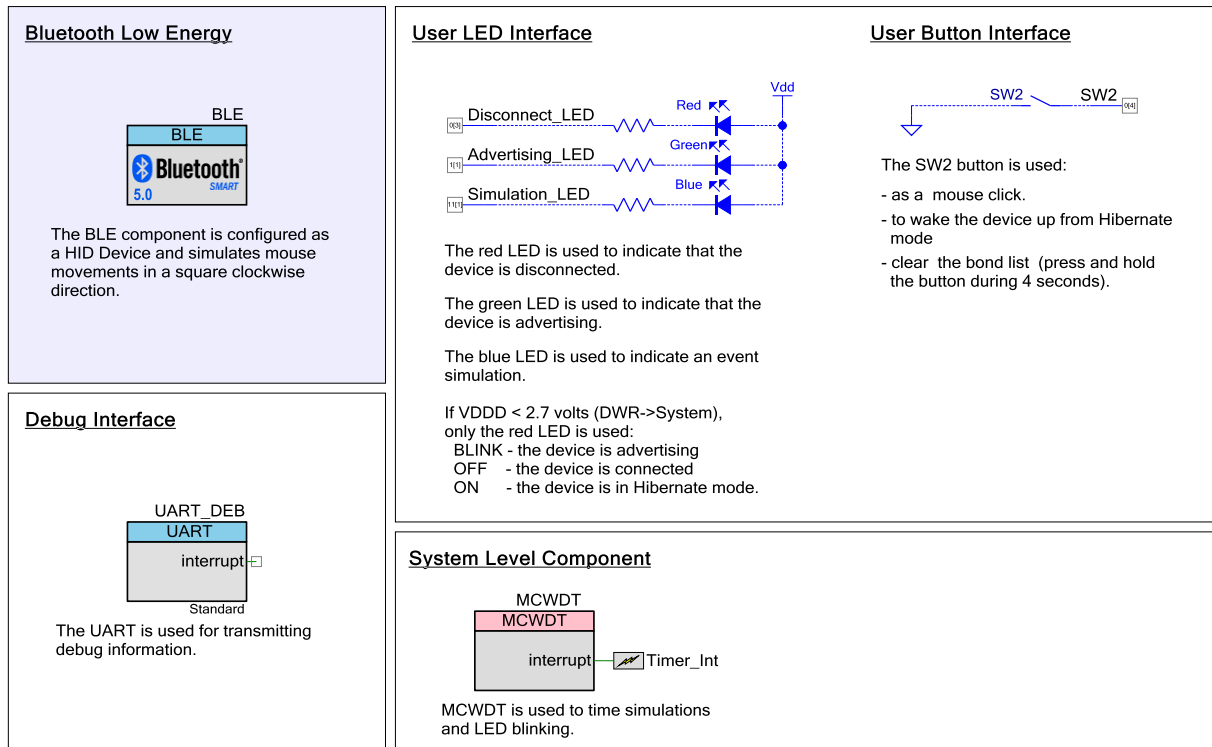
**Associated Parts:** All [PSoC® 6 MCU with BLE Connectivity \(PSoC 6 BLE\)](#) parts

**Related Hardware:** [CY8CKIT-062 PSoC 6 BLE Pioneer Kit](#)

## Design

Figure 1 shows the top design schematic.

Figure 1. BLE HID Mouse Code-Example Schematic



The BLE Component implements a HID over the GATT Profile in the HID Device role (GATT Server).

After a start, the device performs the BLE Component initialization. The four callback functions are required in this project for the BLE operation:

- AppCallBack() is required to receive generic events from the BLE Stack.
- HidsCallBack(), BasCallBack(), and ScpsCallBack() are required to receive events from the services.

The CY\_BLE\_GAPP\_StartAdvertisement() function is called after the CY\_BLE\_EVT\_STACK\_ON event to start advertising with the packet as shown in Figure 7. On an advertisement timeout, the system enters Hibernate mode. Press the mechanical button **SW2** on the PSoC 6 BLE Pioneer Kit to wake up the system and start advertising. The BLE subsystem and CPU enter Low-Power Deep Sleep mode between the connection and advertising intervals. The BLE subsystem automatically wakes up to maintain connection and advertising data transfer.

The green LED blinks to indicate that the device is advertising. The red LED is turned ON after disconnection to indicate that no client is connected to the device. When a client is connected successfully, the red and green LEDs are turned OFF. When a client enables notifications, blinking blue LED indicates a simulated data transfer from the HID device to the host.

Additionally, this project implements the Battery Service. By default, the battery level is simulated and changed from 2 to 20 percent.

## Design Considerations

### Using UART for Debugging

Download and install a serial port communication program. Freeware such as Bray's Terminal, and PuTTY are available on the web.

1. Connect the PC and kit with a USB cable.
2. Open the device manager program in your PC, find a COM port that the kit is connected to, and note the port number.
3. Open the serial port communication program and select the previously noted COM port.
4. Configure the Baud rate, Parity, Stop bits, and Flow control information in the PuTTY configuration window. The default settings: Baud rate – 115200, Parity – None, Stop bits – 1, Flow control – XON/XOFF. These settings must match the configuration of the PSoC Creator UART component in the project.
5. Start communicating with the device as explained in the [Operation](#) section.

The UART debugging can be disabled by setting the `DEBUG_UART_ENABLED` to `DISABLED` in the `common.h` file.

### LED Behavior for $V_{DD}$ Voltage < 2.7 V

If the  $V_{DD}$  voltage is set to less than 2.7 V in the DWR settings of the **System** tab, only the red LED is used. The red LED blinks to indicate that the device is advertising. The red LED is OFF when the device is connected to a peer device. When the device is in Hibernate mode, the red LED stays ON.

### Switching the CPU Cores Usage

This section describes how to switch between different CPU cores usage (Single core and Dual core) in the BLE Peripheral Driver Library (PDL) examples.

The BLE Component has the CPU Core parameter that defines the cores usage. It can take the following values:

- **Single core (Complete Component on CM0+)** – only CM0+ core will be used.
- **Single core (Complete Component on CM4)** – only CM4 core will be used.
- **Dual core (Controller on CM0+, Host and Profiles on CM4)** – both cores will be used: CM0+ for the Controller and CM4 for the Host and Profiles.

The BLE examples' structure allows easy switching between different CPU cores options.

Important to remember:

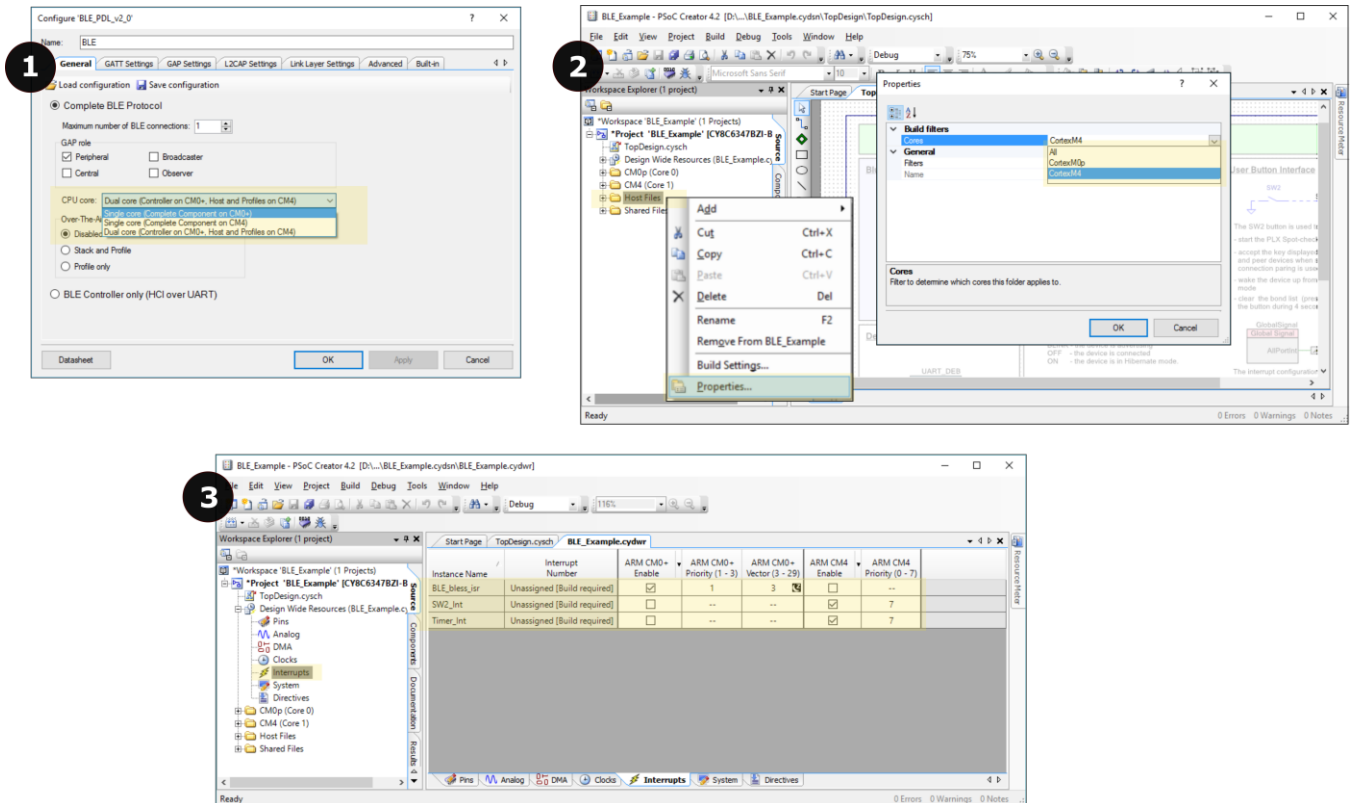
- All application host-files must be run on the host core.
- The BLE Subsystem (BLESS) interrupt must be assigned to the core where the controller runs.
- All additional interrupts (SW2, MCWDT, etc.) used in the example must be assigned to the host core.

Do the following to switch the CPU cores usage:

1. In the BLE Component Customizer **General** tab, select appropriate CPU core option.
2. Change the core properties to CortexM4 or CortexC0p for the project folder Host Files based on the CPU core option selected in step 1. It should be:
  - For **Single core (Complete Component on CM0+)** option: CM0+
  - For **Single core (Complete Component on CM4)** option: CM4
  - For **Dual core (Controller on CM0+, Host and Profiles on CM4)** option: CM4

3. Assign the BLE\_bless\_isr and other peripheral (button – SW2, timer(s) etc.) interrupts to appropriate core in DWR-> interrupts tab:
  - For **Single core (Complete Component on CM0+)** option: BLE\_bless\_isr and peripheral interrupts on **CM0+**
  - For **Single core (Complete Component on CM4)** option: BLE\_bless\_isr and peripheral interrupts on **CM4**
  - For **Dual core (Controller on CM0+, Host and Profiles on CM4)** option: BLE\_bless\_isr interrupt on **CM0+**, other peripheral interrupts on **CM4**

Figure 2. Steps for Switching the CPU Cores Usage



## Hardware Setup

The code example was created for the [CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit](#).

[Table 1](#) lists the pin assignment and connections required on the development board for the supported kits.

Table 1. Pin Assignment

Pin Name	Development Kit	Comment
	CY8CKIT-062	
\UART_DEB:rx\	P5[0]	
\UART_DEB:tx\	P5[1]	
\UART_DEB:rts\	P5[2]	
\UART_DEB:cts\	P5[3]	
Advertising_LED	P1[1]	The green color of the RGB LED.
Disconnect_LED	P0[3]	The red color of the RGB LED.
Simulation_LED	P11[1]	The blue color of the RGB LED.
SW2	P0[4]	

## Components

[Table 2](#) lists the PSoC Creator Components used in this example and the hardware resources used by each of the components.

Table 2. PSoC Creator Components List

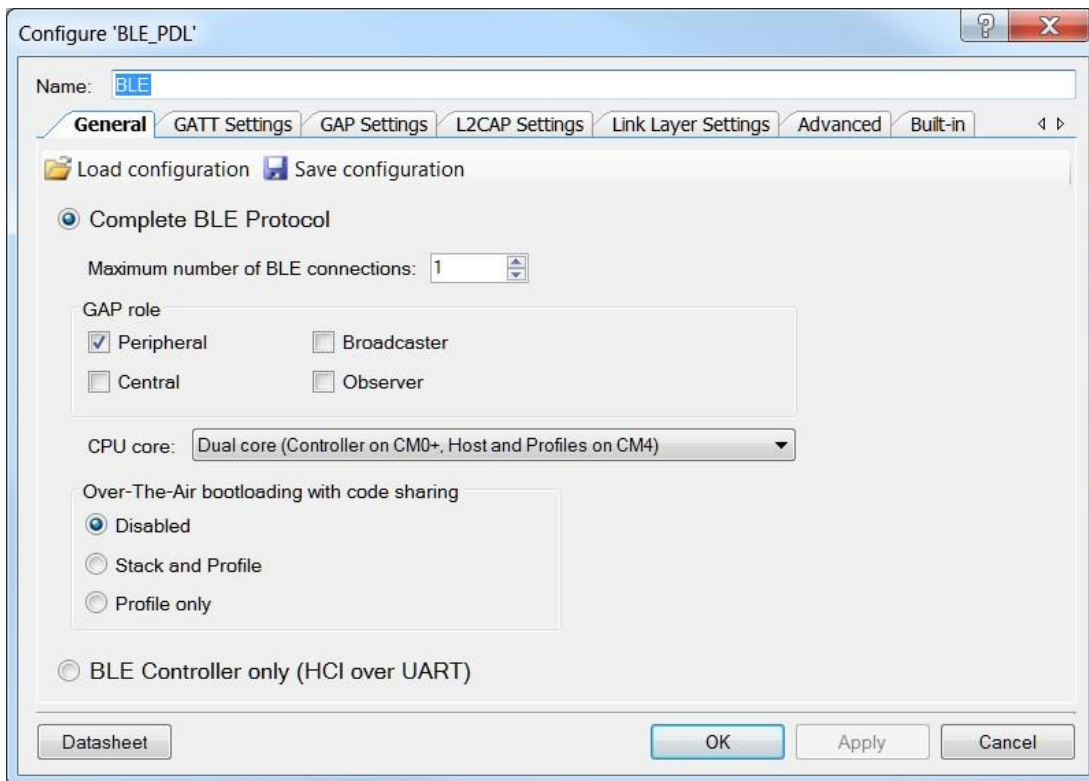
Component	Hardware Resources
BLE	1 BLE, 1 Interrupt
UART_DEB	1 SCB
SW2	1 pin
Wakeup_Interrupt	1 interrupt
Disconnect_LED, Advertising_LED, Simulation_LED	3 pins

## Parameter Settings

### Bluetooth Low Energy (BLE)

The BLE Component is configured as a HID over a GATT Profile in the HID Device role (GATT Server). The HID device has one instance of the HID Service, Battery Service, Device Information Service, and Scan Parameters Service.

Figure 3. General Settings



The screenshot shows the 'Configure BLE\_PDL' dialog box with the 'General' tab selected. The 'Name' field is set to 'BLE'. The 'General' tab contains the following settings:

- Load configuration** (icon) **Save configuration** (icon)
- ☒ **Complete BLE Protocol**
  - Maximum number of BLE connections: 1
  - GAP role**
    - ☒ Peripheral ☐ Broadcaster
    - ☐ Central ☐ Observer
  - CPU core: Dual core (Controller on CM0+, Host and Profiles on CM4)
  - Over-The-Air bootloading with code sharing**
    - ☒ Disabled
    - ☐ Stack and Profile
    - ☐ Profile only
- ☐ **BLE Controller only (HCI over UART)**

At the bottom of the dialog are buttons for **Datasheet**, **OK**, **Apply**, and **Cancel**.

Figure 4. GATT Settings

Configure 'BLE\_PDL'

Name: BLE

General **GATT Settings** GAP Settings L2CAP Settings Link Layer Settings Advanced Built-in

Server instances: 1

+ Add Descriptor

GATT
 

- Server
  - Generic Access
  - Generic Attribute
  - Human Interface Device
    - Report In
      - Report Map**
      - HID Information
      - HID Control Point
      - Protocol Mode
    - Boot Mouse Input Report
  - Device Information
  - Battery
  - Scan Parameters

Characteristic: **Report Map**

The Report Map characteristic is used to define formatting information for Input Report, Output Report, and Feature Report data transferred between a HID Device and HID Host, information on how this data can be used, and other information regarding physical aspects of the device.

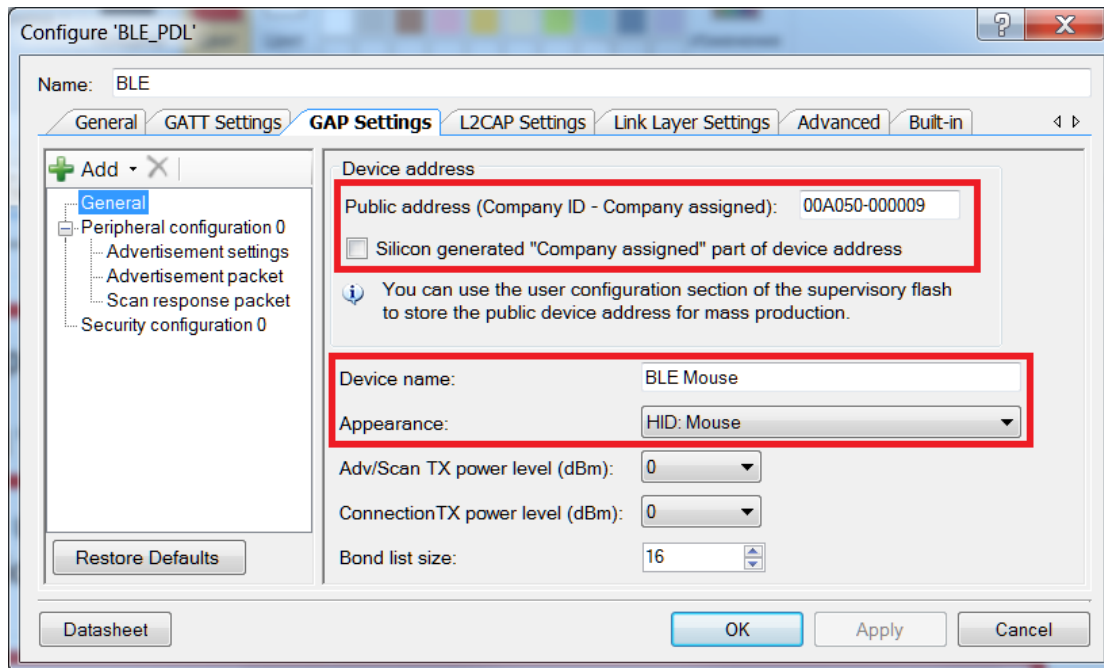
UUID: 2A4B

Name	Value	Bytes
<b>Fields</b>		
<b>Report Map Value</b>		
USAGE_PAGE	Generic Desktop Controls	05 01
USAGE	Mouse	09 02
COLLECTION	Application	A1 01
USAGE	Pointer	09 01
COLLECTION	Physical	A1 00
REPORT_ID	1	85 01
USAGE_PAGE	Button	05 09
USAGE_MINIMUM	1	19 01
USAGE_MAXIMUM	3	29 03
LOGICAL_MINIMUM	0	15 00
LOGICAL_MAXIMUM	1	25 01
REPORT_COUNT	3	95 03
REPORT_SIZE	1	75 01
INPUT	0x02 (Variable)	81 02
REPORT_COUNT	1	95 01
REPORT_SIZE	5	75 05
INPUT	0x03 (Constant, Variable)	81 03
USAGE_PAGE	Generic Desktop Controls	05 01
USAGE	X	09 30
USAGE	Y	09 31
LOGICAL_MINIMUM	-127	15 81
LOGICAL_MAXIMUM	127	25 7F
REPORT_SIZE	8	75 08
REPORT_COUNT	2	95 02
INPUT	0x06 (Variable, Relative)	81 06
END_COLLECTION		C0
END_COLLECTION		C0
<b>Properties</b>		
Read	Mandatory	
<b>Permissions</b>		

Attribute MTU size (bytes): 23

Datasheet OK Apply Cancel

Figure 5. GAP Settings



Configure 'BLE\_PDL'

Name: BLE

General | GATT Settings | **GAP Settings** | L2CAP Settings | Link Layer Settings | Advanced | Built-in

+ Add - X

General

- Peripheral configuration 0
  - Advertisement settings
  - Advertisement packet
  - Scan response packet
  - Security configuration 0

Restore Defaults

Datasheet

OK Apply Cancel

Device address

Public address (Company ID - Company assigned): 00A050-000009

☐ Silicon generated "Company assigned" part of device address

*You can use the user configuration section of the supervisory flash to store the public device address for mass production.*

Device name: BLE Mouse

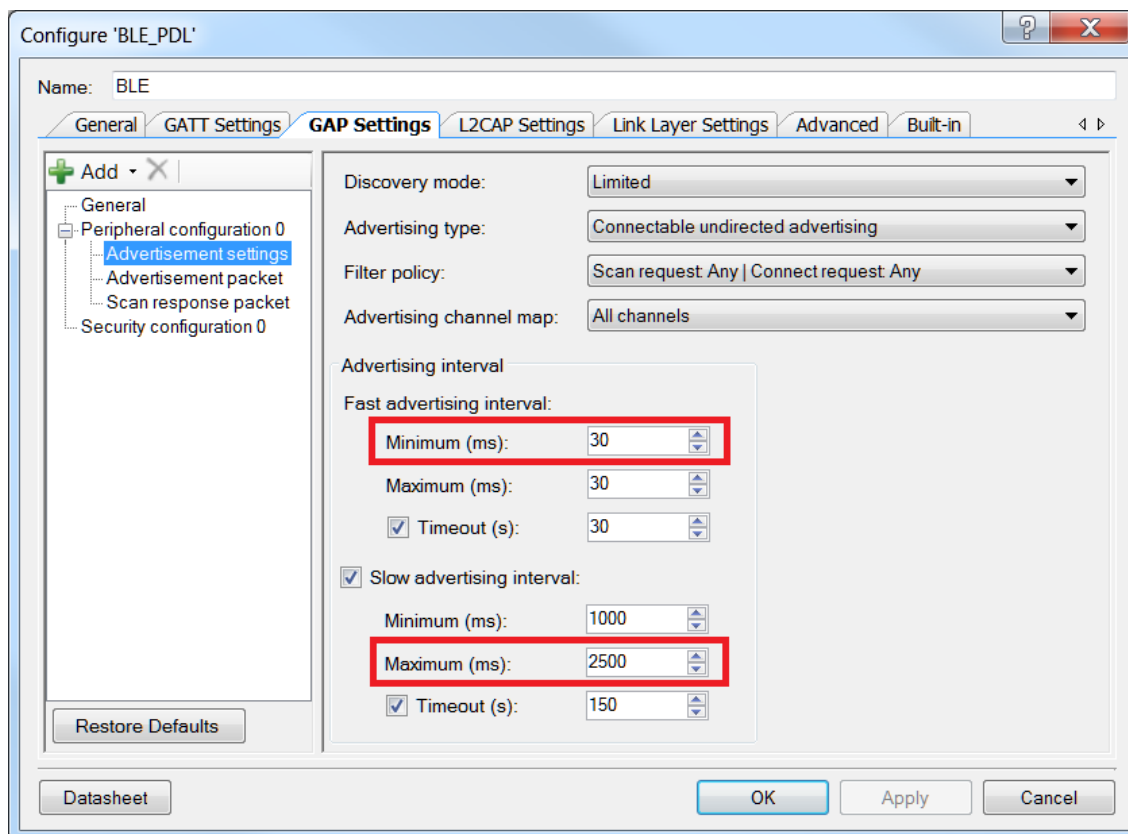
Appearance: HID: Mouse

Adv/Scan TX power level (dBm): 0

Connection TX power level (dBm): 0

Bond list size: 16

Figure 6. GAP Settings: Advertisement Settings



Configure 'BLE\_PDL'

Name: BLE

General | GATT Settings | **GAP Settings** | L2CAP Settings | Link Layer Settings | Advanced | Built-in

+ Add - X

General

- Peripheral configuration 0
  - Advertisement settings
  - Advertisement packet
  - Scan response packet
  - Security configuration 0

Restore Defaults

Datasheet

OK Apply Cancel

Discovery mode: Limited

Advertising type: Connectable undirected advertising

Filter policy: Scan request: Any | Connect request: Any

Advertising channel map: All channels

Advertising interval

Fast advertising interval:

Minimum (ms): 30

Maximum (ms): 30

☒ Timeout (s): 30

☒ Slow advertising interval:

Minimum (ms): 1000

Maximum (ms): 2500

☒ Timeout (s): 150



Figure 7. GAP Settings &gt; Advertisement Packet

Configure 'BLE\_PDL'

Name: BLE

General | GATT Settings | **GAP Settings** | L2CAP Settings | Link Layer Settings | Advanced | Built-in

+ Add - X |  
 General  
 Peripheral configuration 0  
 Advertisement settings  
 Advertisement packet  
 Scan response packet  
 Security configuration 0

Advertisement data settings:

Name	Value
<input checked="" type="checkbox"/> Flags	
<input checked="" type="checkbox"/> Limited discoverable mode	
<input checked="" type="checkbox"/> BR/EDR not supported	
<input checked="" type="checkbox"/> Local Name	
Local name	Complete
<input type="checkbox"/> TX Power Level	
<input type="checkbox"/> Slave Connection Interval Range	
<input checked="" type="checkbox"/> Service UUID	
<input checked="" type="checkbox"/> Human Interface Device	
<input type="checkbox"/> Device Information	
<input checked="" type="checkbox"/> Battery	
<input type="checkbox"/> Scan Parameters	
<input type="checkbox"/> Service Solicitation	
<input type="checkbox"/> Service Data	
<input type="checkbox"/> Service Manager TK Value	
<input checked="" type="checkbox"/> Appearance	
Data	HID: Mouse
<input type="checkbox"/> Public Target Address	
<input type="checkbox"/> Random Target Address	
<input type="checkbox"/> Advertising Interval	
<input type="checkbox"/> LE Bluetooth Device Address	
<input type="checkbox"/> LE Role	
<input type="checkbox"/> URI	
<input type="checkbox"/> Manufacturer Specific Data	

Advertisement packet:

Description	Value	Index
AD Data 1: <<Flags>>		
Length	0x02	[0]
<<Flags>>	0x01	[1]
BR/EDR not supported   Limited discoverable mode	0x05	[2]
AD Data 2: <<Local Name>>		
Length	0x0A	[3]
<<Local Name>>	0x09	[4]
'B'	0x42	[5]
'L'	0x4C	[6]
'E'	0x45	[7]
'	0x20	[8]
'M'	0x4D	[9]
'o'	0x6F	[10]
'u'	0x75	[11]
's'	0x73	[12]
'e'	0x65	[13]
AD Data 3: <<More 16-bit UUIDs available>>		
Length	0x05	[14]
<<More 16-bit UUIDs available>>	0x02	[15]
Service: Human Interface Device		
[0]	0x12	[16]
[1]	0x18	[17]
Service: Battery		
[0]	0x0F	[18]
[1]	0x18	[19]
AD Data 4: <<Appearance>>		
Length	0x03	[20]
<<Appearance>>	0x19	[21]
Value: HID: Mouse		
[0]	0xC2	[22]
[1]	0x03	[23]

Restore Defaults

Datasheet

OK Apply Cancel

Figure 8. Security Settings

Configure 'BLE\_PDL'

Name: BLE

General | GATT Settings | **GAP Settings** | L2CAP Settings | Link Layer Settings | Advanced | Built-in

+ Add - X |  
 General  
 Peripheral configuration 0  
 Advertisement settings  
 Advertisement packet  
 Scan response packet  
 Security configuration 0

Security mode: Mode 1

Security level: Unauthenticated pairing with encryption

I/O capabilities: No Input No Output

Keypress notifications: No

Bonding requirement: Bonding

Encryption key size (bytes): 16

Restore Defaults

Datasheet

OK Apply Cancel

## Operation

You can connect an HID device to Windows 8. Windows 7 and older OS do not have HOGP drivers.

1. Make sure that a PC with Windows 8 has Bluetooth 4.0 installed.
2. To connect to a HID device, click **Add a device** in the **Devices and Printers** window of the Control Panel.
3. Select the **BLE Mouse** device and click **Next**.

Figure 9. Windows 8 PC Connection to BLE Mouse

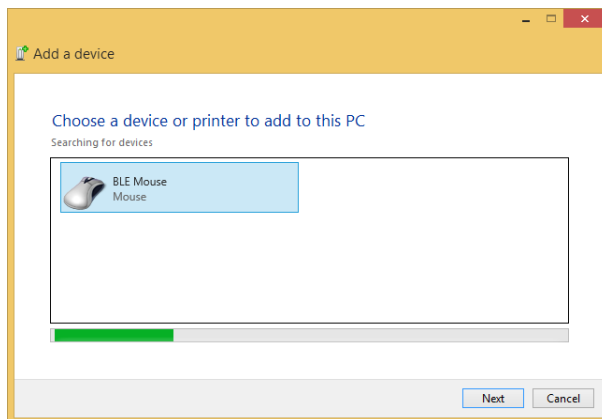
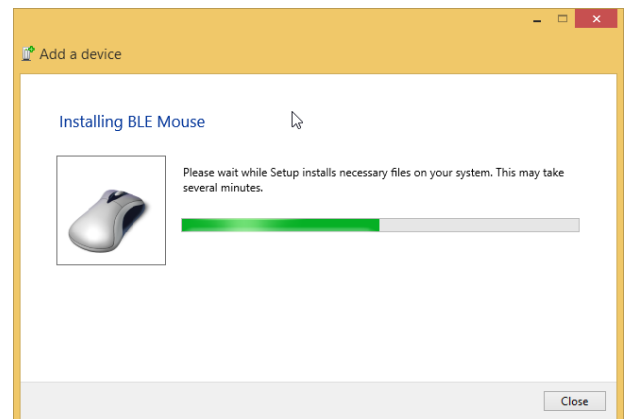
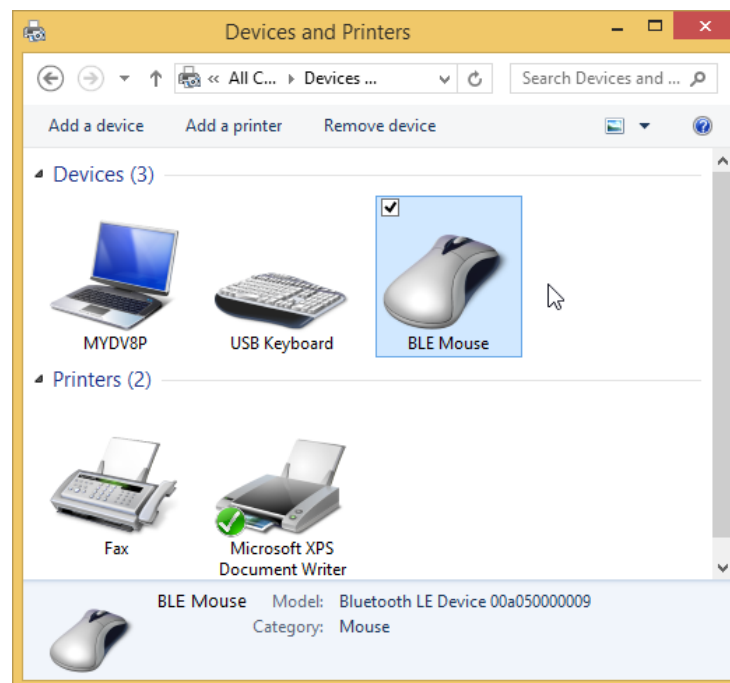


Figure 10. Device Drivers Are Installed, Mouse Cursor Starts Moving



4. The setup will automatically install the necessary files in the system.

Figure 11. Windows 8 PC Recognizes BLE Mouse as Input Device



After pairing, the mouse pointer moves in a square clockwise direction.

Also, you can connect an HID Device to an Android device with the Bluetooth 4.0 support: go to the phone's Bluetooth settings and pair it with your device (it should be recognized as BLE Mouse).

Figure 12. Android Settings for Paired Bluetooth Device

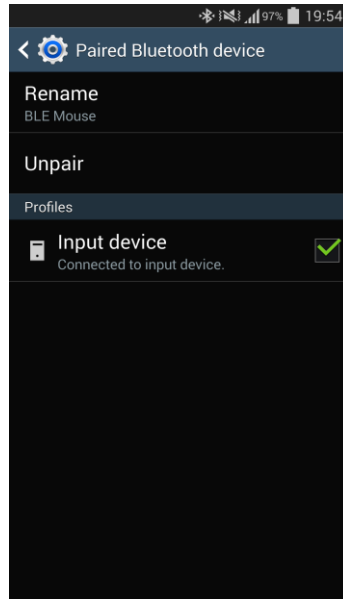


Figure 13. Mouse Cursor is Placed in Screen Center

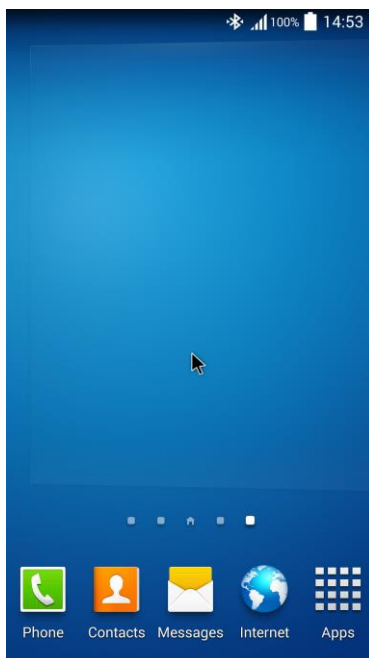
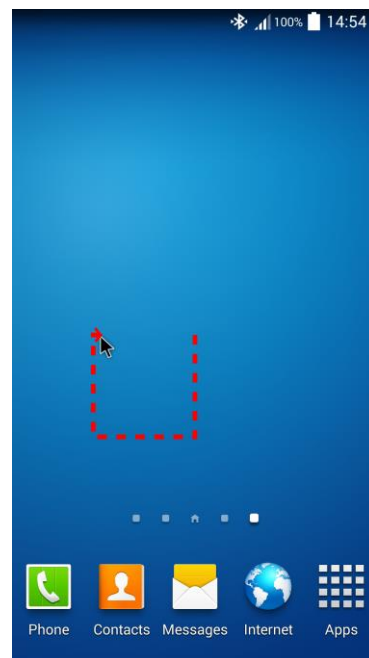


Figure 14. Mouse Cursor is Moving in Square Clockwise Direction



## Related Documents

Application Notes		
<a href="#">AN210781</a>	Getting Started with PSoC 6 MCU with Bluetooth Low Energy (BLE) Connectivity	Describes the PSoC 6 MCU with BLE Connectivity, and how to build a basic code example.
<a href="#">AN215656</a>	PSoC 6 MCU Dual-Core CPU System Design	Presents the theory and design considerations related to this code example.
Software and Drivers		
<a href="#">CySmart – BLE Test and Debug Tool</a>		CySmart is a BLE host emulation tool for Windows PCs. The tool provides an easy-to-use GUI to enable the user to test and debug their BLE Peripheral applications.
PSoC Creator Component Datasheets		
<a href="#">Bluetooth Low Energy (BLE_PDL) Component</a>		The Bluetooth Low Energy (BLE_PDL) Component provides a comprehensive GUI-based configuration window to facilitate designing applications requiring BLE connectivity.
Device Documentation		
<a href="#">PSoC 6 MCU: PSoC 63 with BLE Datasheet Programmable System-on-Chip</a>		<a href="#">PSoC 6 MCU: PSoC 63 with BLE Architecture Technical Reference Manual (TRM)</a>
Development Kit (DVK) Documentation		
<a href="#">CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit</a>		

■

## Document History

Document Title: CE215122 - BLE HID Mouse with PSoC 6 MCU with BLE Connectivity

Document Number: 002-15122

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	5968181	NPAL	11/21/2017	New Code Example.

## Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at [Cypress Locations](#).

## Products

ARM® Cortex® Microcontrollers	<a href="#">cypress.com/arm</a>
Automotive	<a href="#">cypress.com/automotive</a>
Clocks & Buffers	<a href="#">cypress.com/clocks</a>
Interface	<a href="#">cypress.com/interface</a>
Internet of Things	<a href="#">cypress.com/iot</a>
Memory	<a href="#">cypress.com/memory</a>
Microcontrollers	<a href="#">cypress.com/mcu</a>
PSoC	<a href="#">cypress.com/psoc</a>
Power Management ICs	<a href="#">cypress.com/pmic</a>
Touch Sensing	<a href="#">cypress.com/touch</a>
USB Controllers	<a href="#">cypress.com/usb</a>
Wireless Connectivity	<a href="#">cypress.com/wireless</a>

## PSoC® Solutions

[PSoC 1](#) | [PSoC 3](#) | [PSoC 4](#) | [PSoC 5LP](#) | [PSoC 6](#)

## Cypress Developer Community

[Forums](#) | [WICED IOT Forums](#) | [Projects](#) | [Videos](#) | [Blogs](#) | [Training](#) | [Components](#)

## Technical Support

[cypress.com/support](#)

All other trademarks or registered trademarks referenced herein are the property of their respective owners.



Cypress Semiconductor  
198 Champion Court  
San Jose, CA 95134-1709

© Cypress Semiconductor Corporation, 2017. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC ("Cypress"). This document, including any software or firmware included or referenced in this document ("Software"), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress's patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage ("Unintended Uses"). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, WICED, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit [cypress.com](#). Other names and brands may be claimed as property of their respective owners.