

## Objective

This example project demonstrates the Find Me Profile application workflow.

## Overview

This example project demonstrates the Find Me Profile operation of the BLE Component. The Find Me Target uses the Find Me Profile with one instance of the Immediate Alert Service to display the alerts if the Client has configured the device for alerting. The Find Me Target operates with other devices that implement the Find Me Locator Profile. The device switches to Deep Sleep mode between BLE connection intervals.

## Requirements

**Tool:** PSoC® Creator™ 4.2

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

**Associated Parts:** All PSoC 6 MCU with BLE Connectivity parts

**Related Hardware:** CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit

## Hardware Setup

This example uses the kit's default configuration. Refer to the [kit guide](#) to ensure the kit is configured correctly.

1. Connect the BLE Pioneer Kit to the computer's USB port.
2. Connect the BLE Dongle to one of the USB ports on the computer.

## LED Behavior

If the  $V_{DD}$  voltage is set to less than 2.7 V in the DWR settings **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 a device is connected to a peer device. When the device is in Hibernate mode, the red LED stays ON.

LED behavior for  $V_{DD} > 2.7$  volts is described in the [Operation](#) section.

## Software Setup

### BLE Host Emulation Tool

This example requires the CySmart application. Download and install either the [CySmart Host Emulation Tool](#) PC application or the CySmart app for [iOS](#) or [Android](#). You can test behavior with any of the two options, but the CySmart app is simpler. Scan one of the following QR codes from your mobile phone to download the CySmart app.

iOS



Android



## Terminal Tool

This example uses a terminal window. You must have terminal software, such as Tera Term or PuTTY.

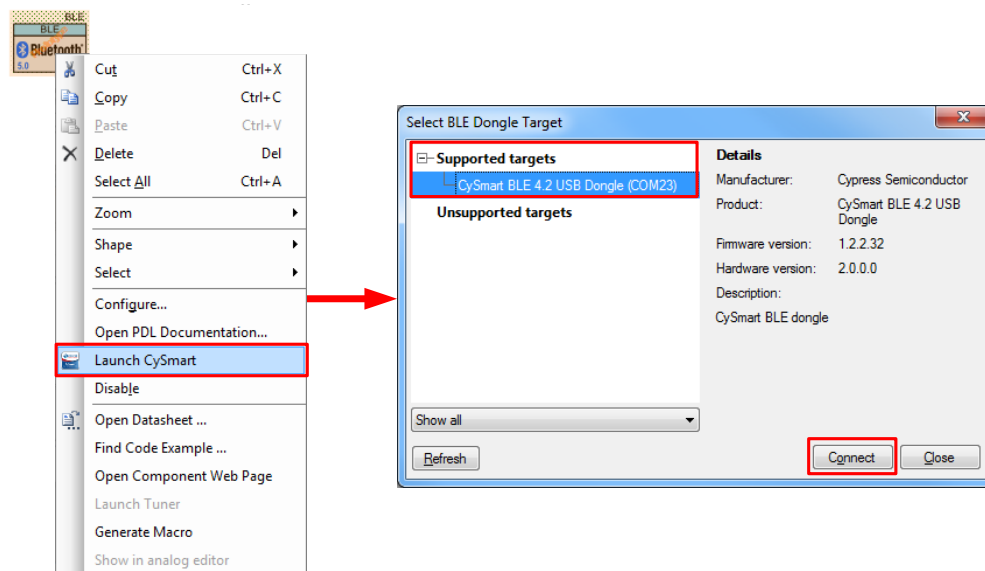
## Operation

You can connect to the Find Me Target device with a BLE 4.0 or BLE 4.1-compatible device configured in the GAP Central role and capable of discovering the Immediate Alert service and Alert Level characteristic. To connect to the Find Me Target device, send a connection request to the device when the device is advertising. The green LED is turned ON while the device is advertising. If the Client is connected to the Find Me Target, the Alert Level Characteristic can be written to the trigger alerts on a remote device. If the Alert Level is set to CY\_BLE\_MILD\_ALERT, the blue LED starts blinking. If the Alert Level is set to CY\_BLE\_HIGH\_ALERT, the blue LED is turned ON. To clear the alerts, send a request from the Client to set the Alert Level Characteristic to CY\_BLE\_NO\_ALERT. The alerts are also cleared when the connection with the Client is canceled or lost.

## Operation Steps

1. Plug the CY8CKIT-062-BLE kit board into your computer's USB port.
2. Build the project and program it into the PSoC 6 MCU device. Choose **Debug > Program**. For more information on device programming, see PSoC Creator Help. Flash for both CPUs is programmed in a single program operation.
3. Observe the green LED blinks while the device is advertising, and the output in the terminal window.
4. Do the following to test example, using the CySmart Host Emulation Tool application as Find Me Target Client:
  - a. Connect the BLE Dongle to your Windows PC. Wait for the driver installation to complete, if necessary.
  - b. Launch the CySmart Host Emulation Tool by right-clicking on the BLE Component and selecting **Launch CySmart**. Alternatively, you can launch the tool by navigating to **Start > Programs > Cypress** and clicking on **CySmart**.
  - c. CySmart automatically detects the BLE dongle connected to the PC. Click **Refresh** if the BLE dongle does not appear in the **Select BLE Dongle Target** pop-up window. Click **Connect**, as shown in Figure 1.

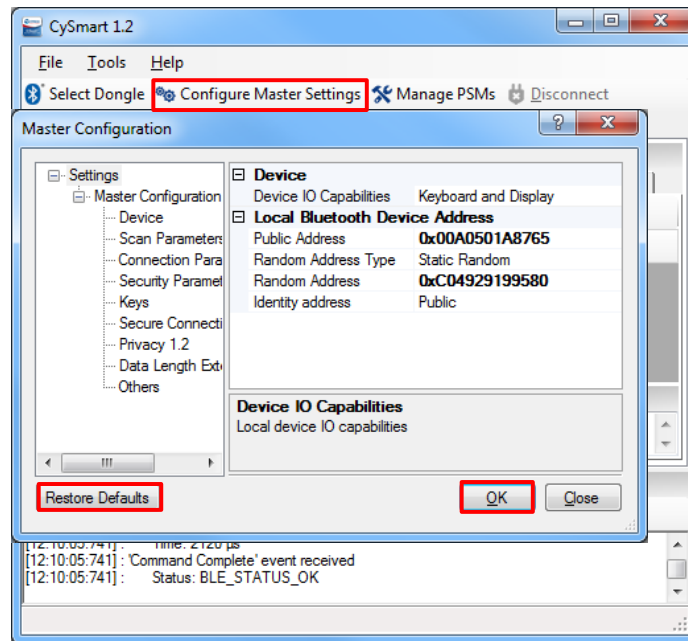
Figure 1. CySmart BLE Dongle Selection



**Note:** If the dongle firmware is outdated, you will be alerted with an appropriate message. You must upgrade the firmware before you can complete this step. Follow the instructions in the window to update the dongle firmware.

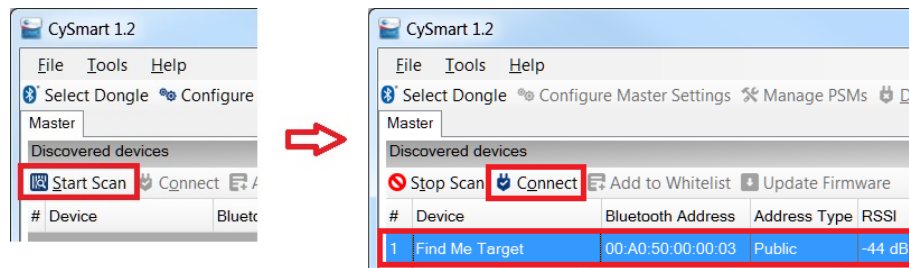
- d. Select **Configure Master Settings** and then click **Restore Defaults**, as Figure 2 shows, and then click **OK**.

Figure 2. CySmart Master Settings Configuration



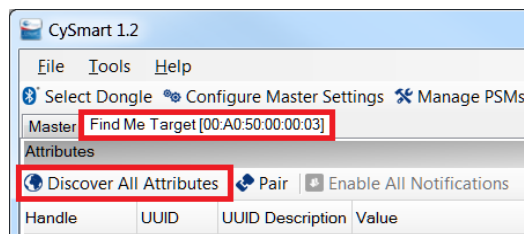
- e. Press the reset switch on the Pioneer Kit to start BLE advertisement if no device is connected or device is in Hibernate mode (red LED is on). Otherwise, skip this step.
- f. On the CySmart Host Emulation Tool, click **Start Scan**. Your device name (configured as **Find Me Target**) should appear in the Discovered devices list, as Figure 3 shows. Select the device and click **Connect** to establish a BLE connection between the CySmart Host Emulation Tool and your device.

Figure 3. CySmart Device Discovery and Connection



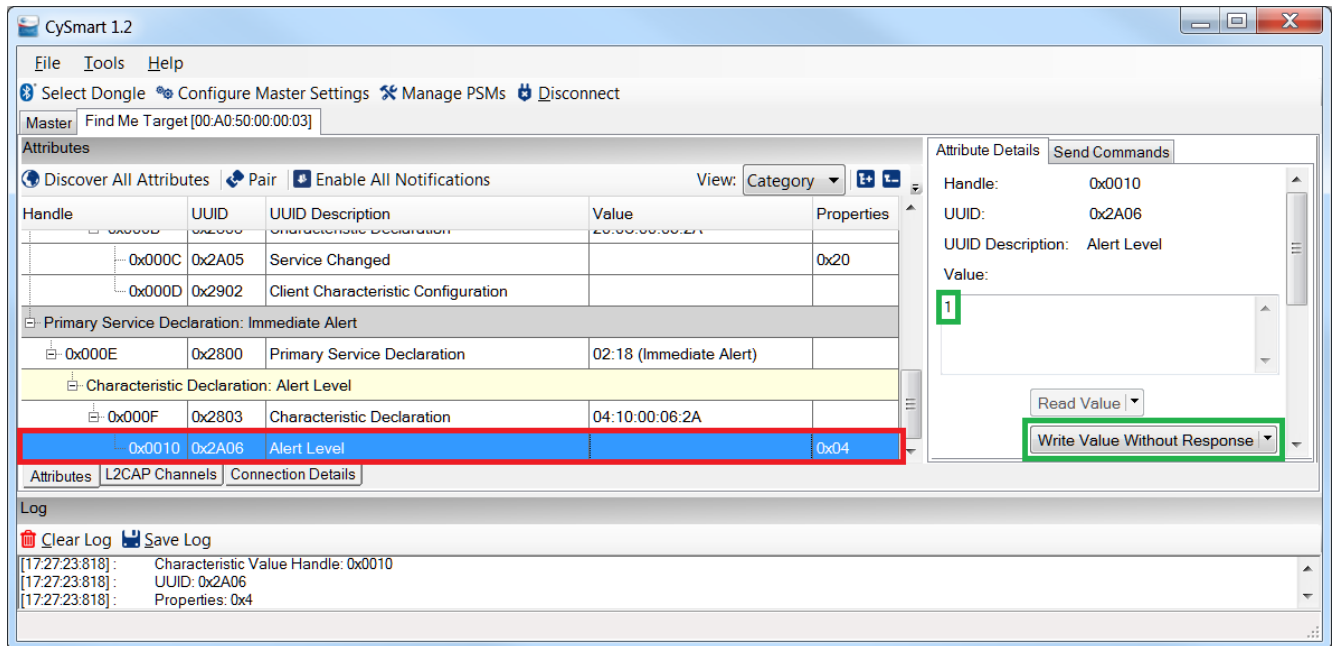
- g. Once connected, switch to the **Find-Me Target** device tab and **Discover all Attributes** on your design from the CySmart Host Emulation Tool, as shown in Figure 4.

Figure 4. CySmart Attribute Discovery



- h. select the **Alert Level** characteristic value and write '1' value in it (mild alert) and observe the blue LED is blinking.

Figure 5. CySmart Write Alert Level characteristic

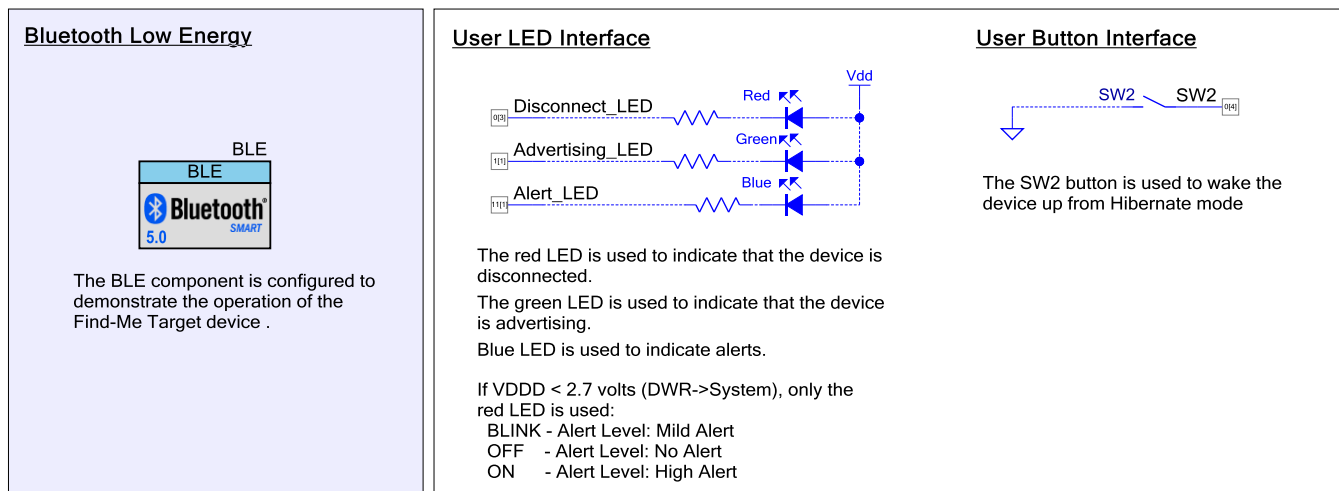


5. Do the following to test example, using the CySmart mobile app as **Find-Me Target Client**:
  - a. Launch CySmart mobile app and swipe down the screen to refresh the list of BLE devices available nearby.
  - b. Make sure that the development kit is advertising (green LED is blinking): you may need to press the **SW1** button in order to wake up the device from Hibernate mode.
  - c. Once the Find-Me Target device appears on the BLE devices list, connect to it and choose **Find Me** in the service selector.
  - d. Press **Select > Mild Alert** and observe the blue LED is blinking.

## Design and Implementation

Figure 6 shows the top design schematic.

Figure 6. BLE Find Me Profile Code Example Schematic



The project demonstrates the core functionality of the BLE Component configured as a Find Me Target. After a startup, the device performs initialization of the BLE Component. In this project, two callback functions are used for the BLE operation. One callback function (StackEventHandler()) is required for receiving generic events from the BLE stack, and the other (IasEventHandler()) is required for receiving events from the Immediate Alert Service. The CY\_BLE\_EVT\_STACK\_ON event indicates the successful initialization of the BLE stack. After this event is received, the Component starts advertising with the packet structure as configured in the BLE Component Customizer. The BLE Component stops advertising once the 180-second advertising period expires. On an advertisement event timeout, the device goes to a low-power mode (Stop mode) and waits for a device reset event to wake up the device again.

While connected to a Client and between the connection intervals, the device is put into Deep Sleep mode.

## Pin Assignments

Pin assignments and connections required on the development board for supported kits are in [Table 1](#).

Table 1. Pin Assignment

Pin Name	Development Kit	Comment
	CY8CKIT-062	
Advertising_LED	P1[1]	The green color of the RGB LED
Disconnect_LED	P0[3]	The red color of the RGB LED
Alert_LED	P11[1]	The blue color of the RGB LED
SW2	P0[4]	

## Components and Settings

[Table 2](#) lists the PSoC Creator Components used in this example, how they are used in the design, and the non-default settings required so they function as intended.

Table 2. PSoC Creator Components

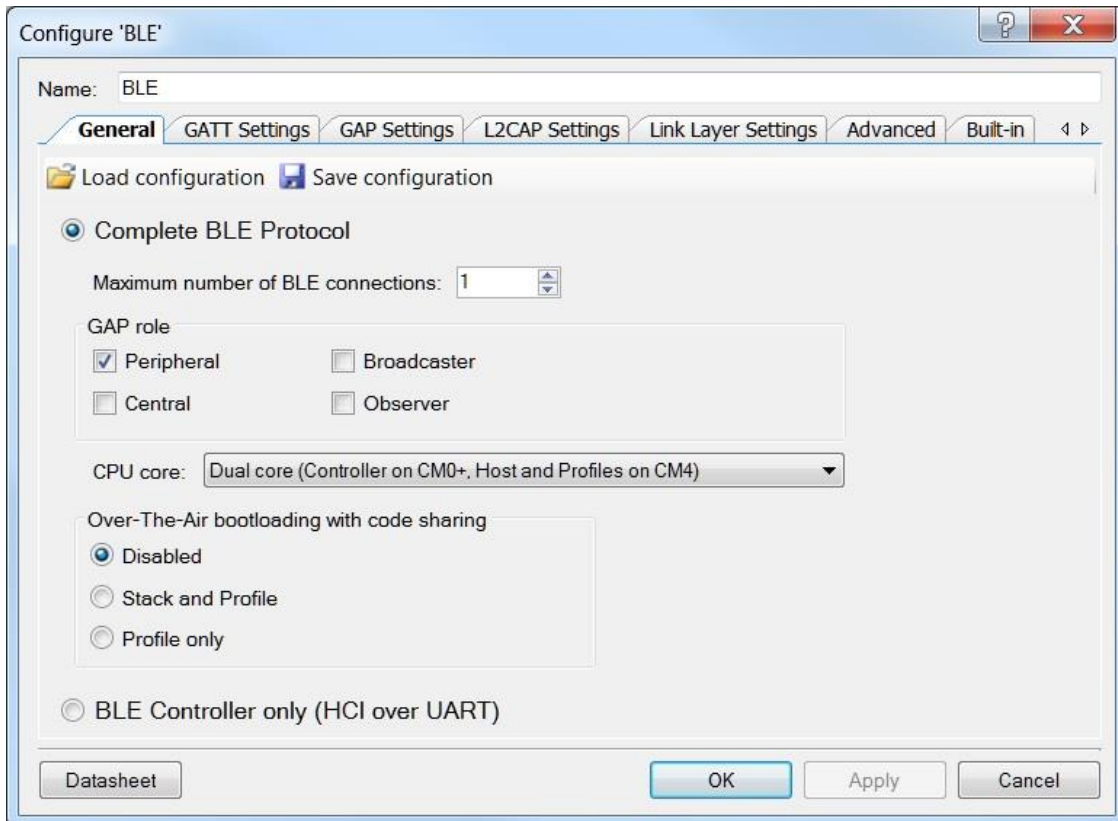
Component	Instance Name	Purpose	Non-default Settings
Bluetooth Low Energy (BLE)	BLE	The BLE component is configured to demonstrate the operation of the Find Me Target device.	See <a href="#">Parameter Settings</a>
Digital Input Pin	SW2	The <b>SW2</b> button is used to wake the device up from Hibernate mode.	<b>[General tab]</b> Uncheck HW connection Drive mode: Resistive Pull Up
Digital Output Pin	Disconnect_LED Advertising_LED Alert_LED	These GPIOs are configured as firmware-controlled digital output pins that control LEDs.	<b>[General tab]</b> Uncheck HW connection Drive mode: Strong Drive

For information on the hardware resources used by a Component, see the Component datasheet.

## Parameter Settings

The BLE Component is configured as the Find Me Target in the GAP Peripheral role with the settings shown in the figures below.

Figure 7. General Settings



The screenshot shows the 'Configure BLE' dialog box with the 'General' tab selected. The 'Name' field is set to 'BLE'. Below the tabs, there are 'Load configuration' and 'Save configuration' buttons. The 'Complete BLE Protocol' radio button is selected. The 'Maximum number of BLE connections' is set to 1. Under 'GAP role', 'Peripheral' is checked. The 'CPU core' dropdown is set to 'Dual core (Controller on CM0+, Host and Profiles on CM4)'. Under 'Over-The-Air bootloading with code sharing', 'Disabled' is selected. At the bottom, the 'BLE Controller only (HCI over UART)' radio button is unselected. The dialog has 'Datasheet', 'OK', 'Apply', and 'Cancel' buttons at the bottom.

Configure 'BLE'

Name: BLE

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

Load configuration | Save configuration

☒ 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

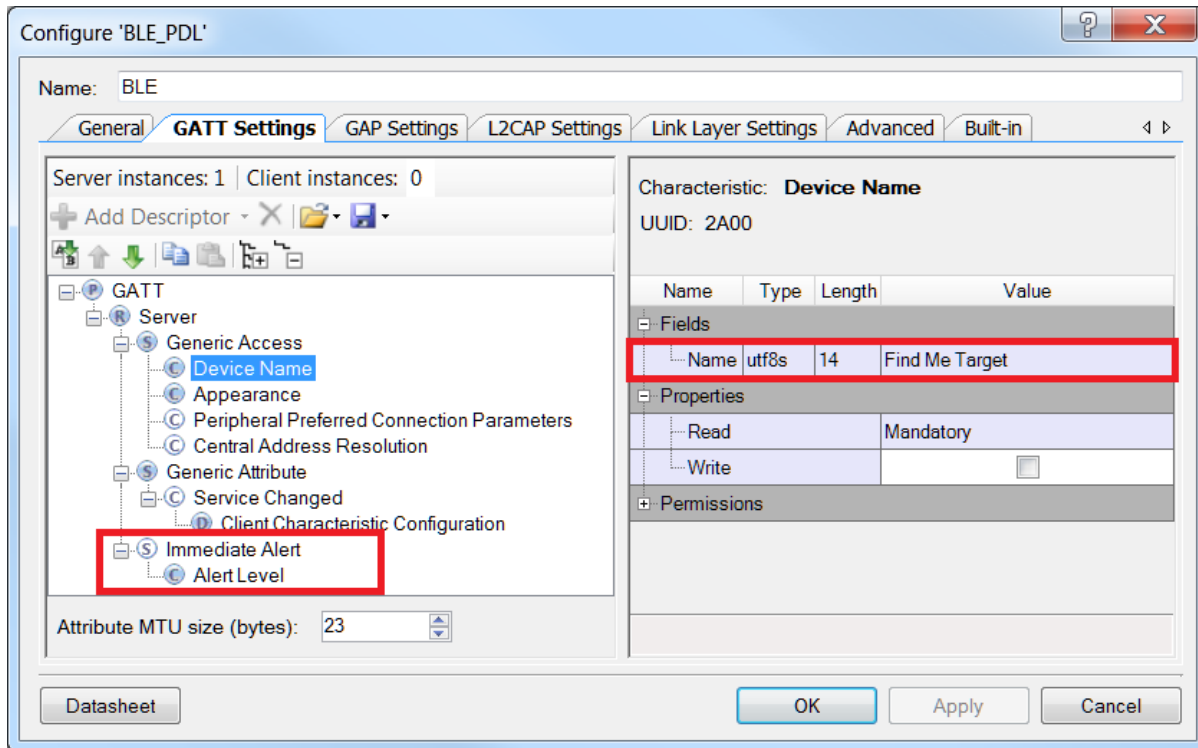
☐ Stack and Profile

☐ Profile only

☐ BLE Controller only (HCI over UART)

Datasheet OK Apply Cancel

Figure 8. GATT Settings



Configure 'BLE\_PDL'

Name: BLE

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

Server instances: 1 | Client instances: 0

+ Add Descriptor - X

Tree View:

- GATT
  - Server
    - Generic Access
    - Device Name**
    - Appearance
    - Peripheral Preferred Connection Parameters
    - Central Address Resolution
    - Generic Attribute
    - Service Changed
    - Client Characteristic Configuration
    - Immediate Alert**
    - Alert Level

Attribute MTU size (bytes): 23

Characteristic: **Device Name**  
UUID: 2A00

Name	Type	Length	Value
Find Me Target	utf8s	14	

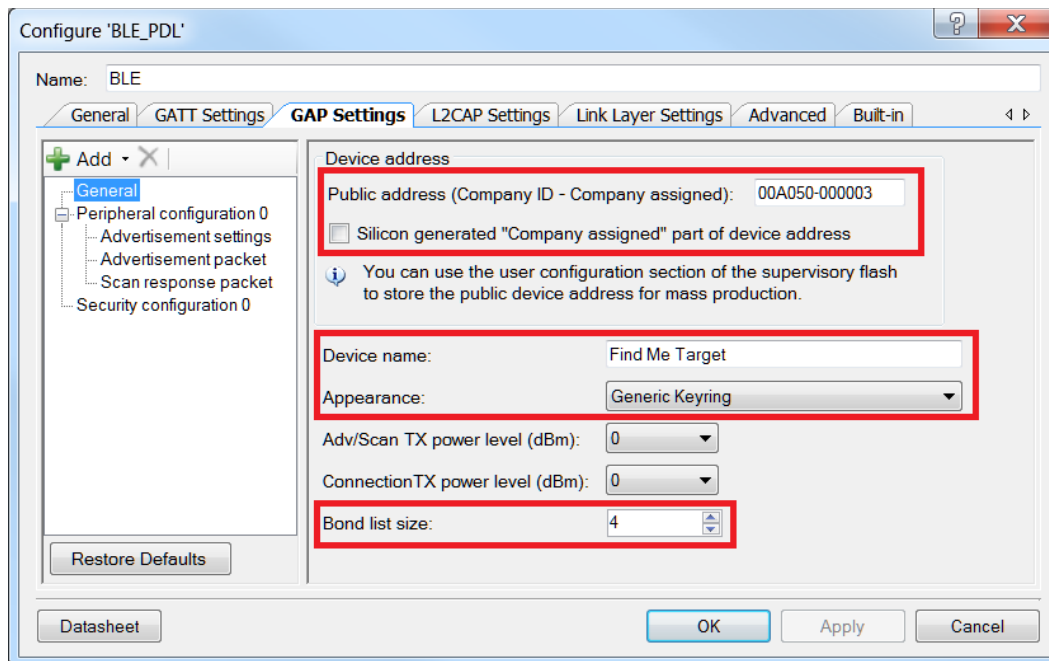
Properties:

Read	Write
Mandatory	<input type="checkbox"/>

Permissions:

Buttons: Datasheet, OK, Apply, Cancel

Figure 9. GAP Settings



Configure 'BLE\_PDL'

Name: BLE

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

Tree View:

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

Buttons: Restore Defaults, Datasheet, OK, Apply, Cancel

Device address

Public address (Company ID - Company assigned): 00A050-000003

☐ 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: Find Me Target

Appearance: Generic Keyring

Adv/Scan TX power level (dBm): 0

Connection TX power level (dBm): 0

Bond list size: 4

Figure 10. 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

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): 20

Maximum (ms): 30

☒ Timeout (s): 30

☒ Slow advertising interval:

Minimum (ms): 1000

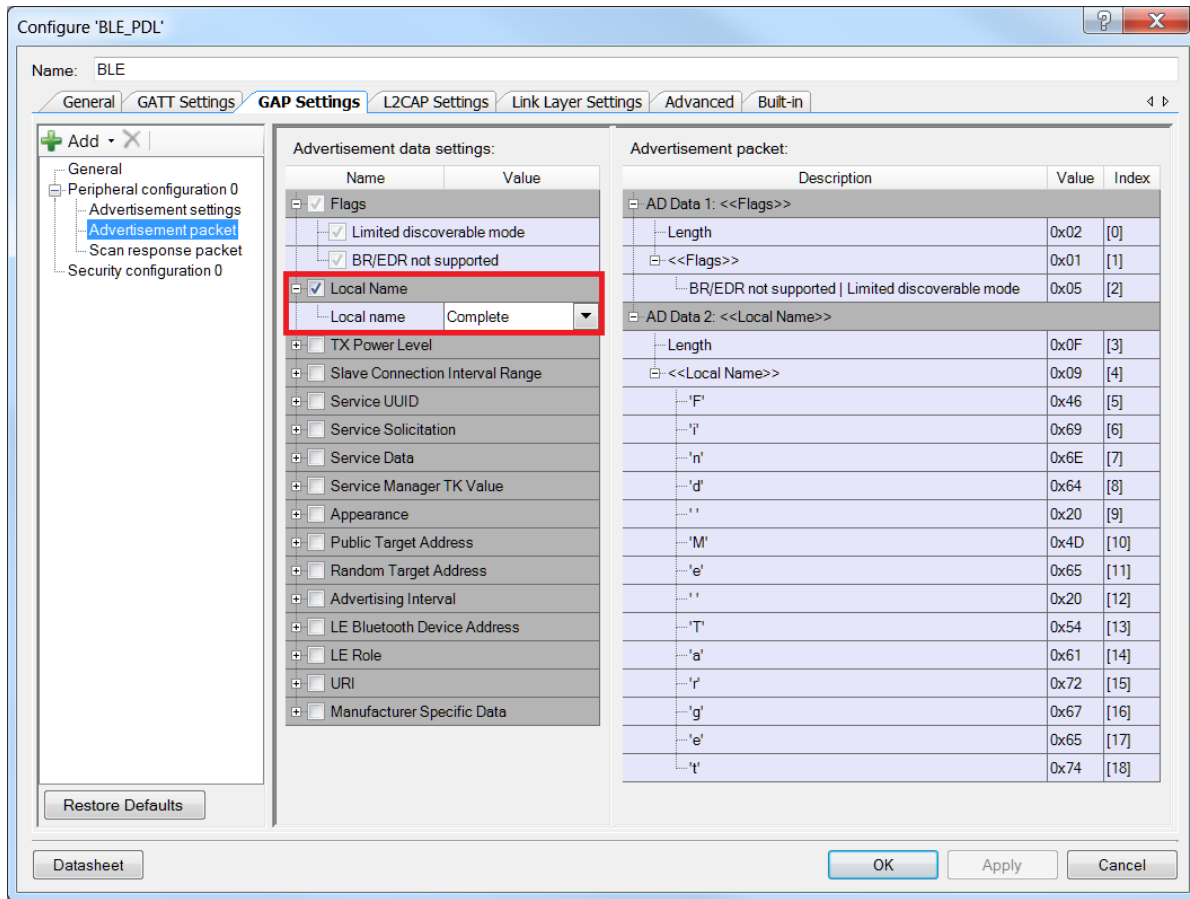
Maximum (ms): 1000

☒ Timeout (s): 150

Datasheet OK Apply Cancel



Figure 11. GAP Settings: 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

Restore Defaults

Datasheet

OK Apply Cancel

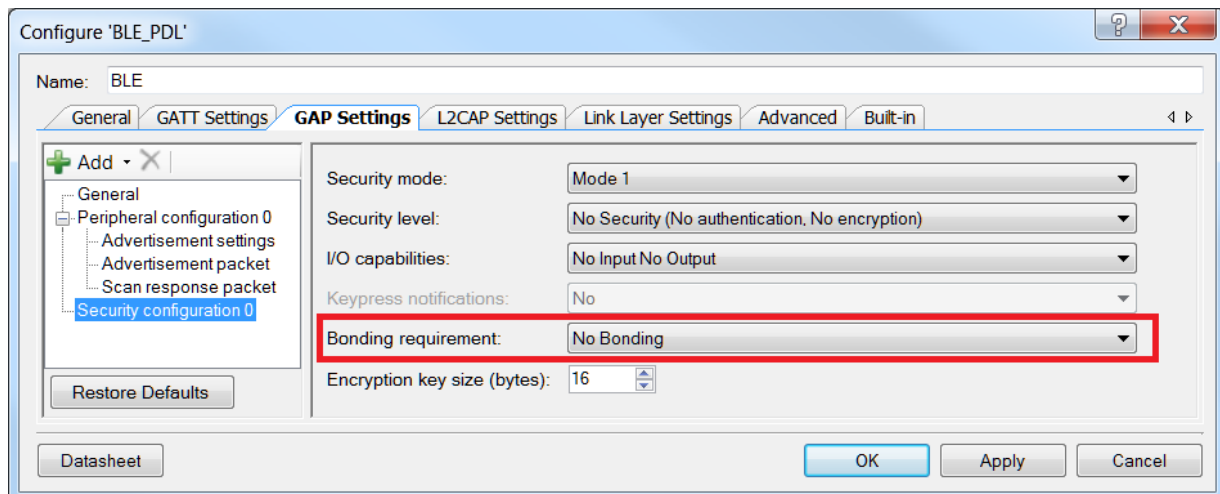
Advertisement data settings:

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 type="checkbox"/> Service UUID	
<input type="checkbox"/> Service Solicitation	
<input type="checkbox"/> Service Data	
<input type="checkbox"/> Service Manager TK Value	
<input type="checkbox"/> Appearance	
<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	0x0F	[3]
<<Local Name>>	0x09	[4]
'F'	0x46	[5]
'i'	0x69	[6]
'n'	0x6E	[7]
'd'	0x64	[8]
''	0x20	[9]
'M'	0x4D	[10]
'e'	0x65	[11]
''	0x20	[12]
'T'	0x54	[13]
'a'	0x61	[14]
'r'	0x72	[15]
'g'	0x67	[16]
'e'	0x65	[17]
't'	0x74	[18]

Figure 12. 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**

Restore Defaults

Datasheet

OK Apply Cancel

Security mode: Mode 1

Security level: No Security (No authentication, No encryption)

I/O capabilities: No Input No Output

Keypress notifications: No

**Bonding requirement: No Bonding**

Encryption key size (bytes): 16

## Switching the CPU Cores Usage

This section describes how to switch between different CPU cores usage (Single core/ Dual core) in the BLE 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+ will be used.
- Single core (Complete Component on CM4) – only CM4 will be used.
- Dual core (Controller on CM0+, Host and Profiles on CM4) – CM0+ and CM4 will be used: CM0+ for the Controller and CM4 for the Host and Profiles.

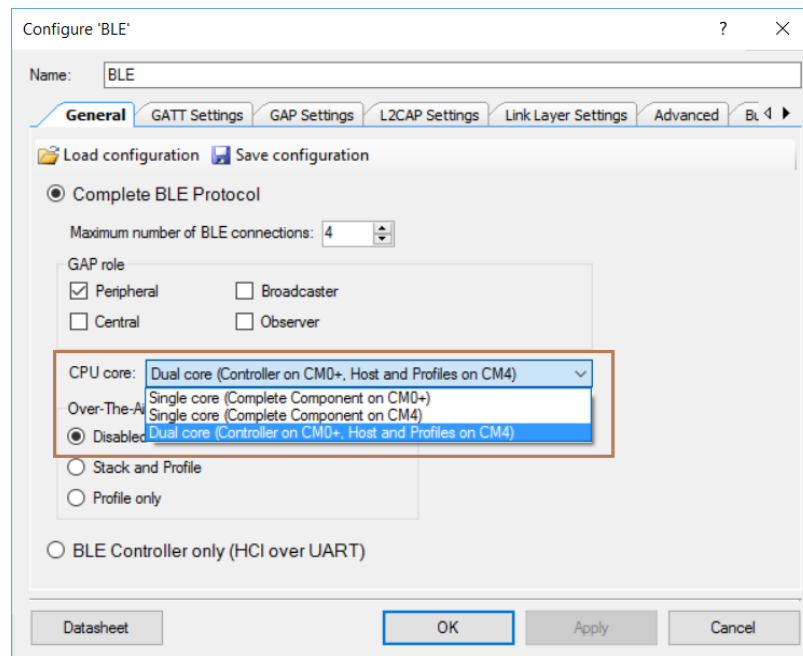
The BLE example structure allows easy switching between different CPU cores options. Important to remember:

- All application host-files must be run on the host core.
- The BLESS interrupt must be assigned to the core where the controller runs.

Steps for switching the CPU Cores usage:

1. In the BLE customizer **General** tab, select appropriate CPU core option.

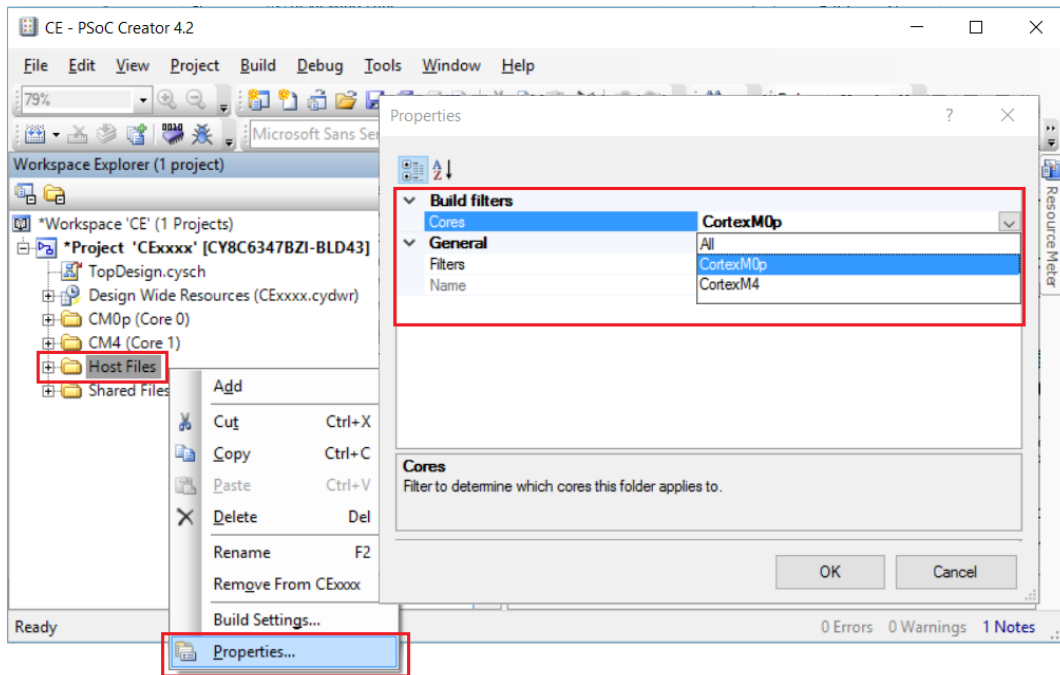
Figure 13. Select CPU Core



2. Identify the core on which host files will run. In the workspace explorer panel, right click **Host Files**, choose **Properties**. Set the **Cores** property corresponding to the CPU core chosen in step 1, as shown in [Figure 14](#).

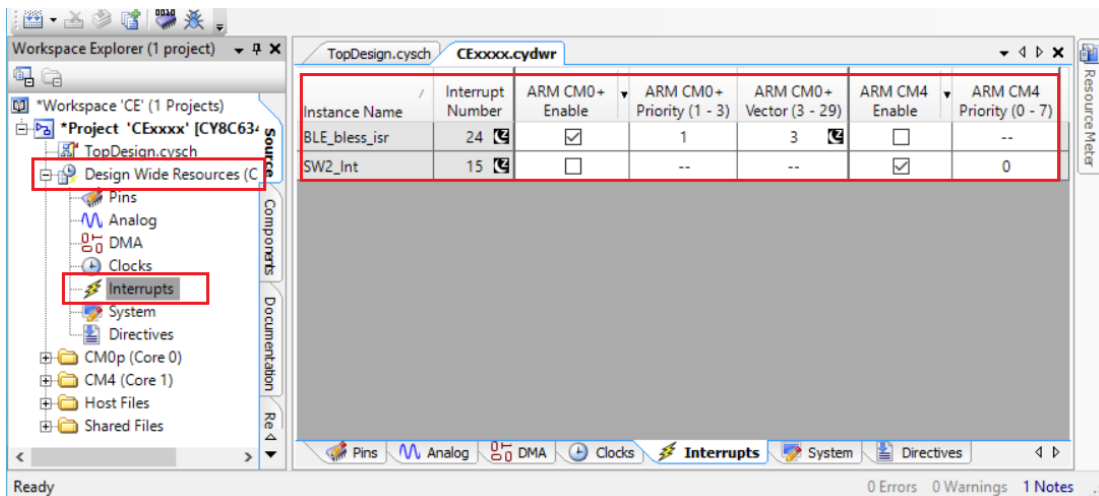
- for Single core (Complete Component on CM0+) option – CM0+
- for Single core (Complete Component on CM4) option – CM4
- for Dual core (Controller on CM0+, Host and Profiles on CM4) option – CM4

Figure 14. Change Core Properties



3. Assign the BLE\_bless\_isr and other peripheral (button – SW2, timer(s) etc.) interrupts to the 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 15. Assign Interrupts



## Reusing This Example

This example is designed for the CY8CKIT-062-BLE pioneer kit. To port the design to a different PSoC 6 MCU device and/or kit, change the target device using the Device Selector and update the pin assignments in the Design Wide Resources Pins settings as needed.

## Related Documents

Application Notes		
<a href="#">AN210781</a>	Getting Started with PSoC 6 MCU with Bluetooth Low Energy (BLE) Connectivity	Describes PSoC 6 BLE, and how to build a basic code example.
<a href="#">AN215656</a>	PSoC 6 MCU Dual-CPU System Design	Presents the theory and design considerations related to this code example.
Software and Drivers		
<a href="#">CySmart – Bluetooth® LE Test and Debug Tool</a>		CySmart is a Bluetooth® LE host emulation tool for Windows PCs. The tool provides an easy-to-use Graphical User Interface (GUI) to enable the user to test and debug their Bluetooth LE 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.</a>		<a href="#">PSoC® 6 MCU: PSoC 63 with BLE Architecture Technical Reference Manual</a>
Development Kit (DVK) Documentation		
<a href="#">CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit</a>		

## Document History

Document Title: CE217637 - BLE Find Me Profile with PSoC 6 MCU with BLE Connectivity

Document Number: 002-17637

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	6090384	NPAL	06/05/2018	New spec

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

### PSoC® Solutions

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

### Cypress Developer Community

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

### Technical Support

[cypress.com/support](http://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, 2018. 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. No computing device can be absolutely secure. Therefore, despite security measures implemented in Cypress hardware or software products, Cypress does not assume any liability arising out of any security breach, such as unauthorized access to or use of a Cypress product. In addition, the products described in these materials may contain design defects or errors known as errata which may cause the product to deviate from published specifications. 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](http://cypress.com). Other names and brands may be claimed as property of their respective owners.