

## Objective

This code example demonstrates connectivity between the PSoC® 6 MCU with Bluetooth Low Energy (BLE) and CySmart™ BLE host emulation tool or mobile device running the CySmart mobile application, to transfer CapSense® proximity sensing information.

## Overview

This code example demonstrates connectivity between the PSoC 6 MCU with BLE Connectivity (PSoC 6 MCU), which acts as a Peripheral and GATT Server device, and CySmart BLE host emulation PC tool or mobile device running the CySmart mobile application (acting as a Central and GATT Client). A custom BLE service is used for the proximity sensor.

In more detail:

- An “always-on” E-INK display that shows the instructions to use the code example. The E-INK display remains ON after a restart, while consuming no power for display retention.
- CapSense proximity sensor
- BLE connectivity
  - Advertisement and connection with any Central device
  - Custom BLE profile and service
  - Data transfer over BLE using notifications

This code example assumes that you are familiar with the PSoC 6 MCU and the PSoC Creator™ Integrated Design Environment (IDE). If you are new to PSoC 6 MCU, you can find introductions in the application note [AN210781 – Getting Started with PSoC 6 MCU with Bluetooth Low Energy \(BLE\) Connectivity](#).

## Requirements

**Tool:** [PSoC Creator 4.2](#)

**Programming Language:** C (Arm® GCC 5.4.1)

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

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

## Design

The E-INK display shows the instructions to use this code example at startup and is then turned OFF to save power. E-INK displays consume no power to retain the display. For more details on E-INK display, see the code example [CE218133 – PSoC 6 MCU E-INK Display with CapSense](#).

The BLE profile in this code example consists of a BLE custom service called CapSense Proximity. The CapSense Proximity service consists of a custom characteristic that is used to send data as notifications to the GATT Client device. The notification data consists of the proximity signal read by the CapSense Component from a proximity wire attached to header **J13** on the Pioneer Board. This characteristic supports notification, which allows the GATT Server to send data to the connected Client device whenever new data is available. The properties for the custom service/characteristics are configured in the BLE Component under the **GATT Settings** tab, as shown in [Figure 1](#).

Figure 1. BLE CapSense Proximity Service Configuration

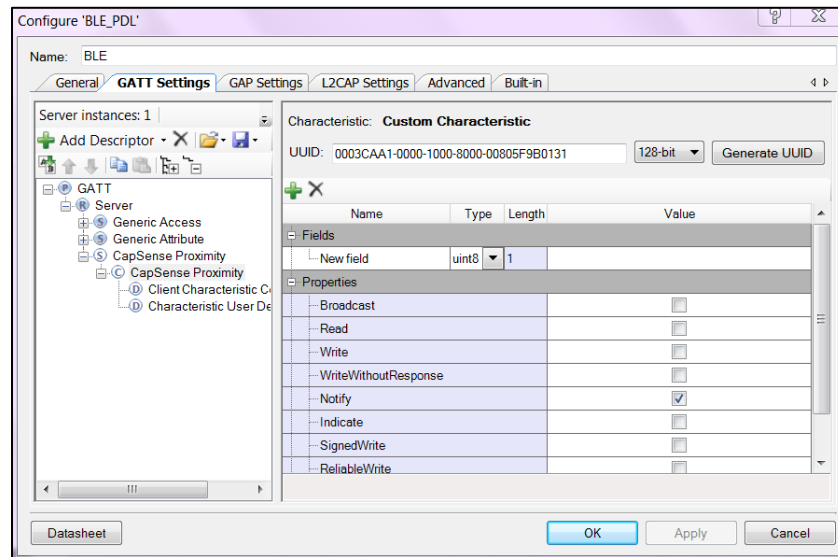


Figure 2, Figure 3, and Figure 4 show the TopDesign schematic of this code example.

Figure 2. TopDesign Schematic: BLE and Interrupts

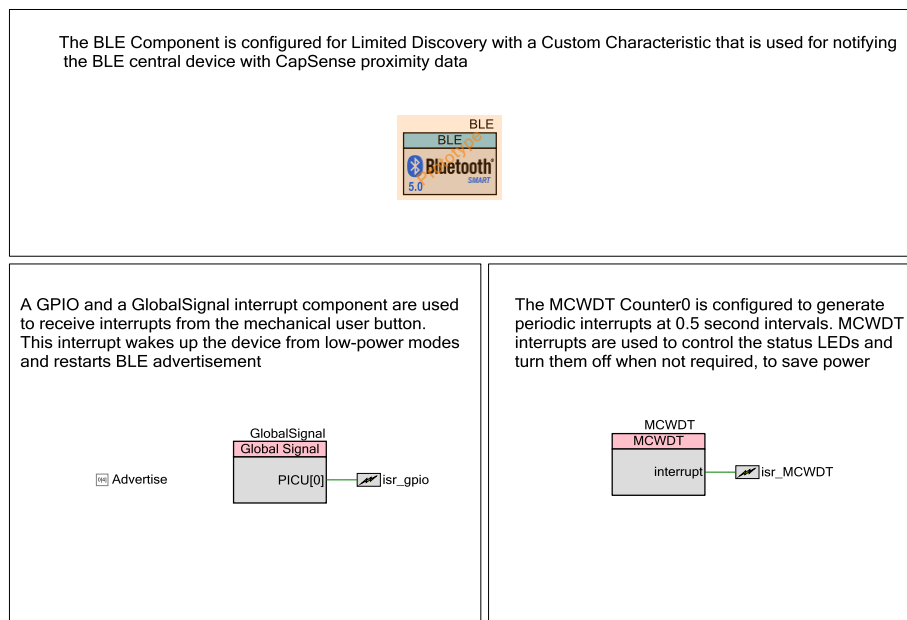


Figure 3. TopDesign Schematic: CapSense and LEDs

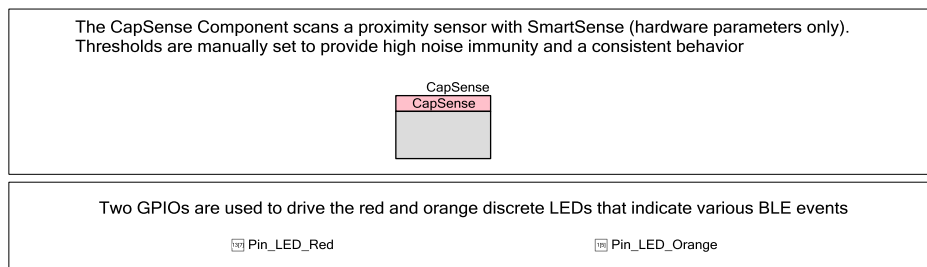
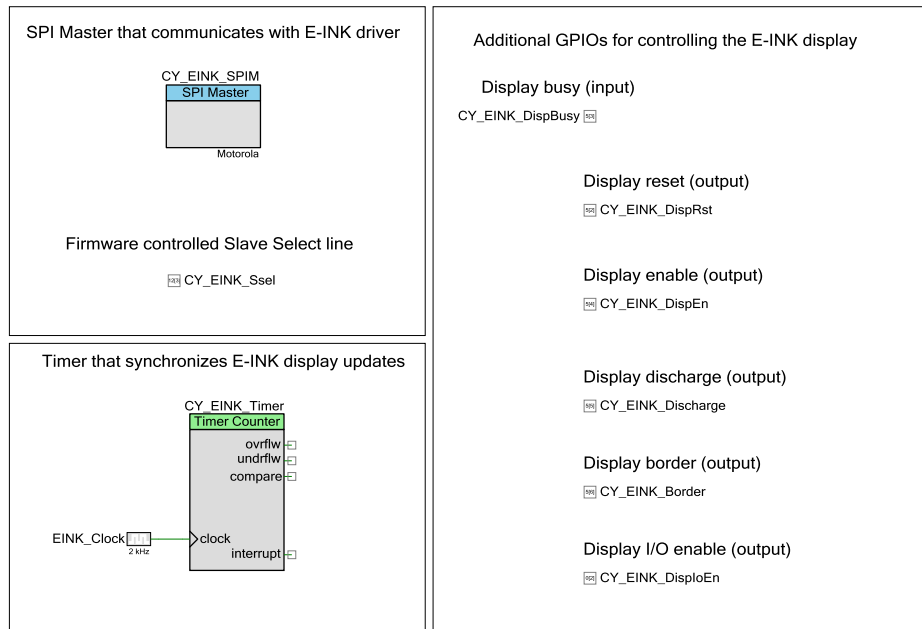


Figure 4. TopDesign Schematic: E-INK Display



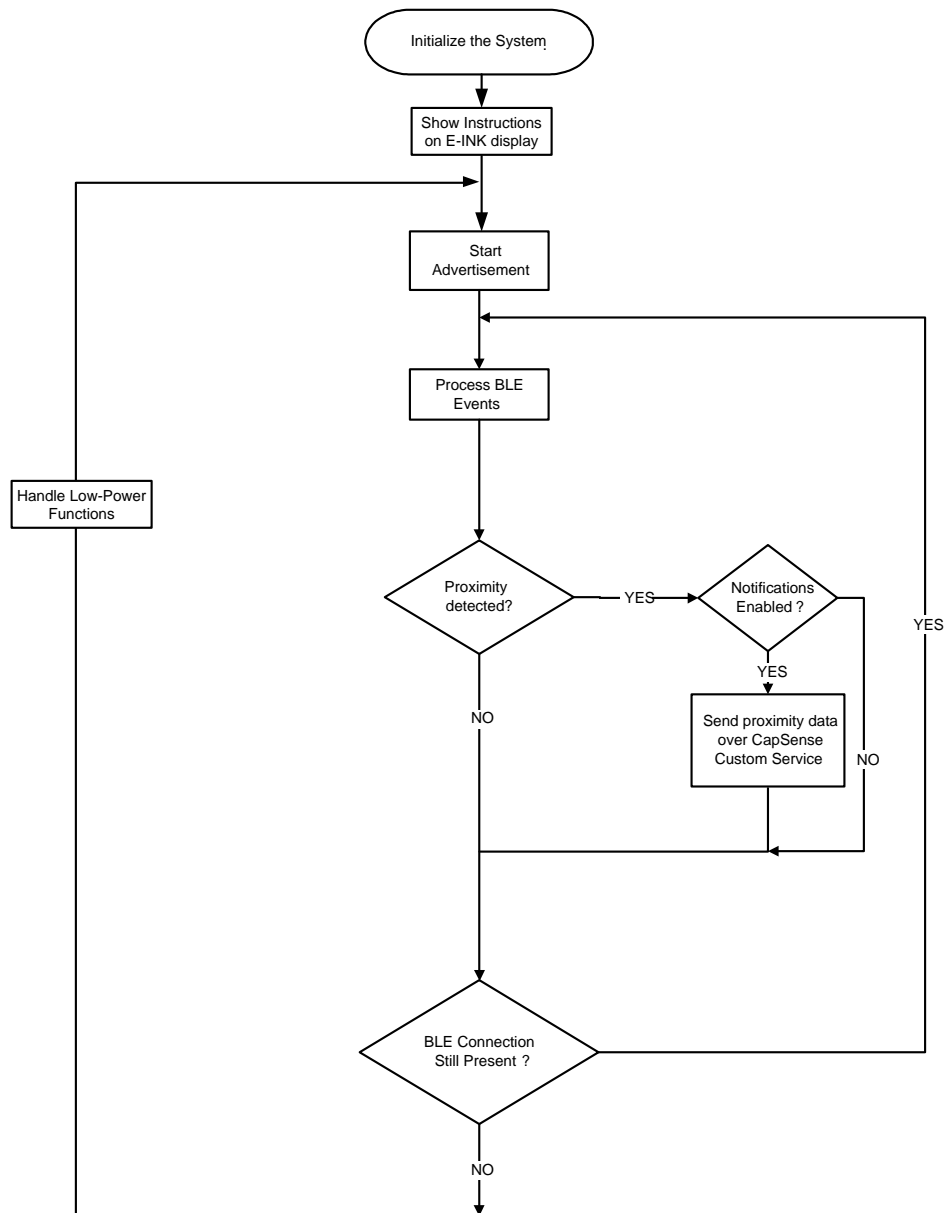
The code example consists of the following files:

- *main\_cm0p.c* contains functions that starts up the BLE controller, starts up CM4, and services BLE stack events.
- *main\_cm4.c* contains the main function, which is the entry point and execution of the firmware application. The main function calls the initializing and display functions, and continuously processes BLE and CapSense events.
- *ble\_application.c/h* contain all the macros and function definitions related to BLE communication and operation. They include the definition of the event callback function that is registered with the BLE Component at startup. The callback function is used to send BLE-related events from the BLE stack to the application layer for processing. These files contain functions to send CapSense notifications to the GATT Client device.
- *proximity.c/h* contain the functions that scan CapSense proximity sensor and process the data.
- *led.c/h* contain the functions that initialize and control status LEDs.
- *display.c/h* contain the functions that initialize the E-INK display and show the instructions to use this code example at startup<sup>1</sup>.
- *screen\_contents.c/h* contain the text and background images used by the display module.
- *low\_power.c/h* contain functions to make the system enter low-power modes and turn OFF status LEDs depending on system-level conditions.

Figure 5 shows the firmware flow of this code example.

<sup>1</sup> For a detailed list of files included in the E-INK Library, see the code example [CE218133 – PSoC 6 MCU E-INK Display with CapSense](#).

Figure 5. Firmware Flow



## Hardware Setup

Set the switches and jumpers on the Pioneer Board as shown in [Table 1](#).

Table 1. Switch and Jumper Selection

Switch/Jumper	Position	Location
SW5	3.3 V	Front
SW6	PSoC 6 BLE	Back
SW7	V <sub>DD</sub> /KitProg2	Back
J8	Installed	Back

Populate **J13** header with a proximity wire provided with the kit. Form a loop with the proximity wire as [Figure 6](#) shows for increased proximity range.

Figure 6. Hardware Setup



## Software Setup

Install the CY8CKIT-62-BLE PSoC 6 BLE Pioneer Kit software, which contains all the required software to evaluate this code example. No additional software setup is required.

## Operation

The code example can be verified using either of these methods: the CySmart BLE Host Emulation Tool and BLE dongle on a PC or the CySmart mobile application.

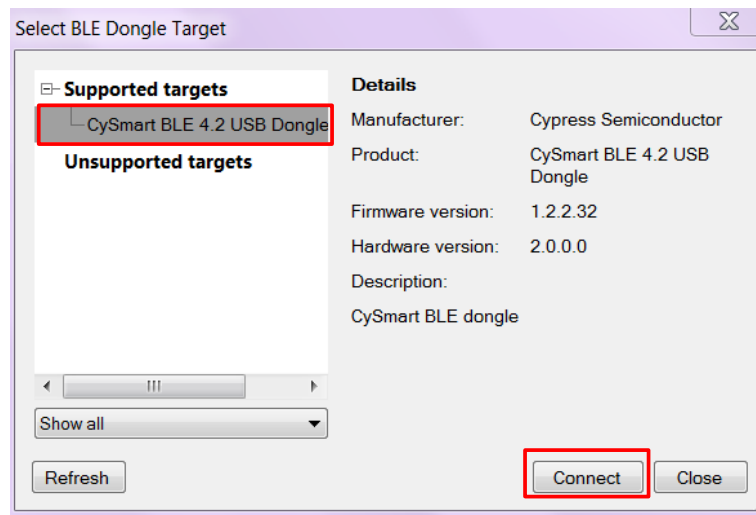
### CySmart BLE Host Emulation Tool

To verify the *CE218135\_BLE\_Proximity* code example using the CySmart BLE Host Emulation tool, follow these steps:

**Note:** See the [CySmart BLE host emulation tool documentation](#) to learn how to use the tool.

1. Connect the BLE dongle to one of the USB ports on the computer.
2. Start the CySmart BLE Host Emulation tool on the computer by going to **Start > All Programs > Cypress > CySmart <version> > CySmart <version>**. You will see a list of BLE dongles connected to it. If no dongle is found, click **Refresh**. Select the BLE dongle and click **Connect**.

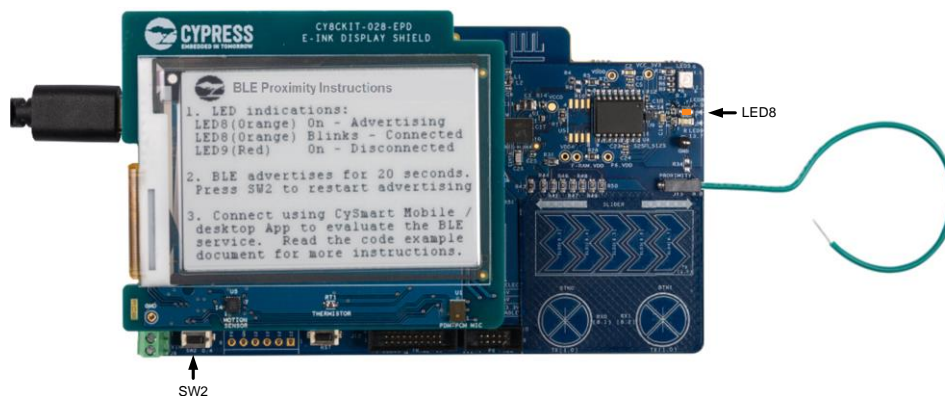
Figure 7. Connect to BLE Dongle



3. Power the Pioneer Board through the USB connector **J10**.
4. Program the Pioneer Board with the `CE218135_BLE_Proximity` project. See the [Pioneer Kit guide](#) for details on how to program firmware into the device.

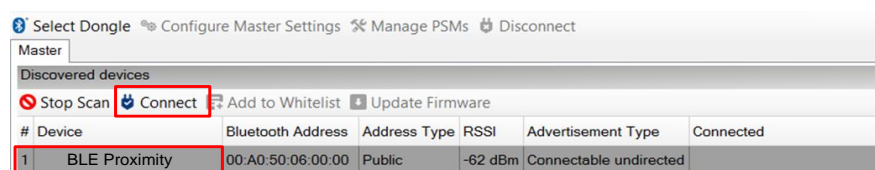
After programming successfully, the E-INK display will refresh and show the instructions to use this project and the BLE will start advertising. The advertising timeout is configured to be 20 seconds. The orange LED (**LED8**) remains ON during this period to indicate the BLE advertising state as [Figure 8](#) shows.

Figure 8. BLE Advertising



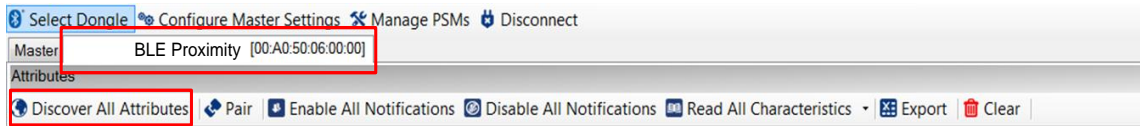
5. If the BLE advertisement has timed out (**LED8** is OFF), press **SW2** to restart advertisement.
6. On the CySmart Host Emulation tool, click **Start Scan** to see the list of available BLE Peripheral devices. Double-click the **BLE Proximity** device to connect, or click **BLE Proximity** and then click **Connect**. A successful connection is indicated by **LED8** continuously blinking at half-second intervals.

Figure 9. Connect to BLE Proximity Peripheral



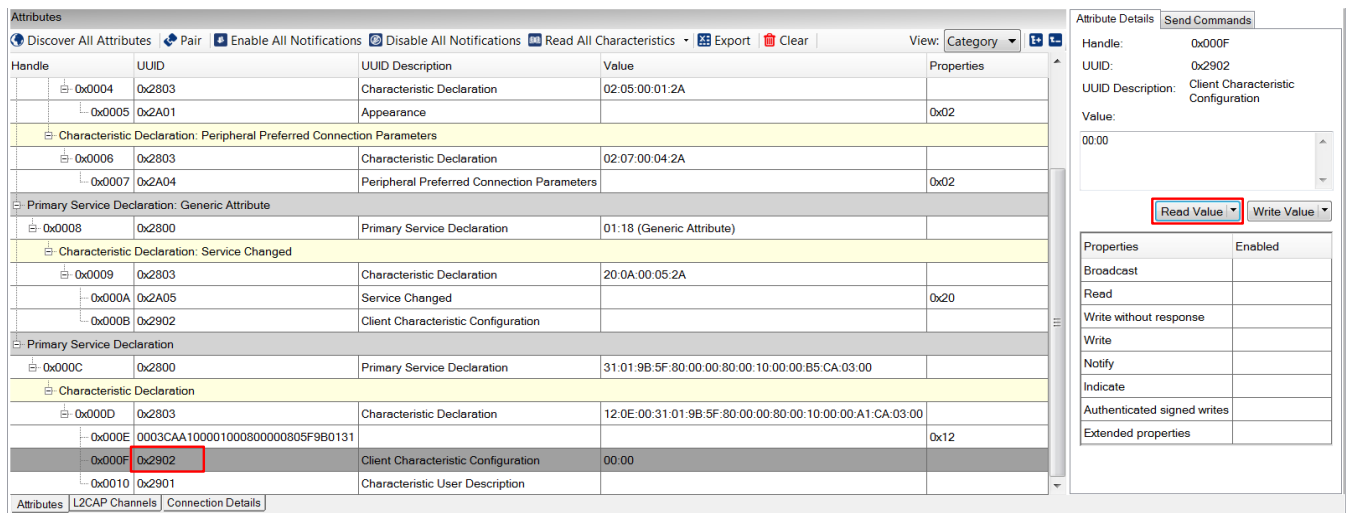
7. Click **Discover All Attributes** to find all attributes supported.

Figure 10. Discover All Attributes



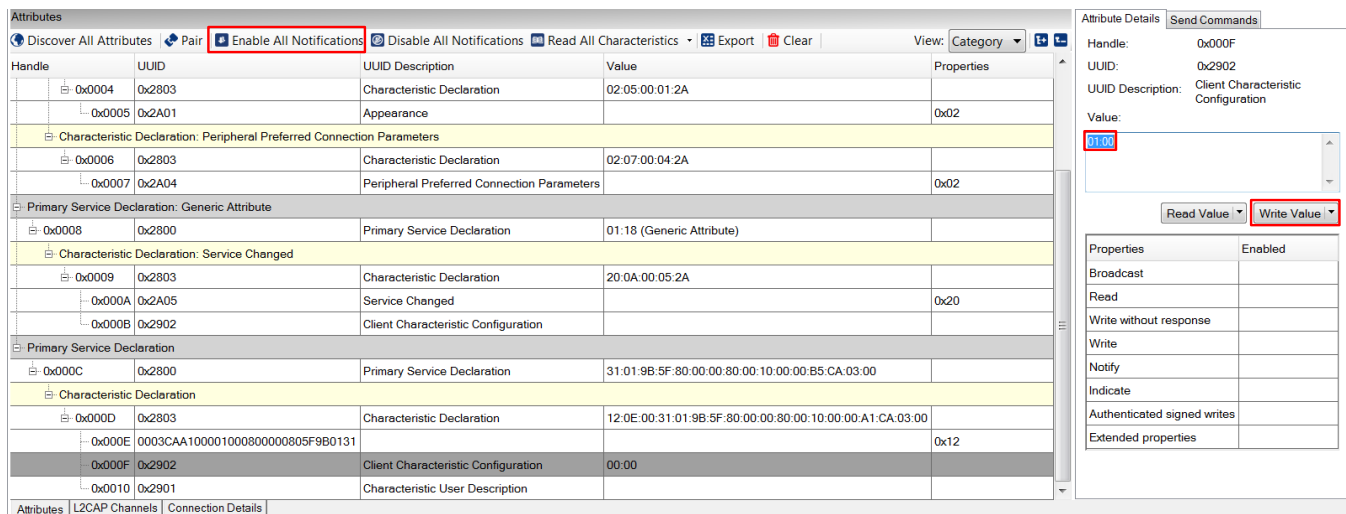
8. Locate the attribute **Client Characteristic Configuration** descriptor (UUID 0x2902) under the CapSense Proximity characteristic (UUID 0x0003CAA200001000800000805F9B0131). Click **Read Value** to read the existing Client Characteristic Configuration Descriptor (CCCD) value as shown in Figure 11.

Figure 11. Read CCCD for CapSense Proximity Characteristic



9. Modify the **Value** field of the CCCD to '01:00' and click **Write Value**. This enables the notifications on the CapSense Proximity characteristic. Alternatively, you can press **Enable All Notifications** to enable the notifications for all services.

Figure 12. Write CCCD to Enable Notifications



10. Bring your hand close to the proximity sensor, as shown in Figure 13 and see the notification values in the CapSense Proximity value field, as shown in Figure 14.



Figure 13. CapSense Proximity Testing

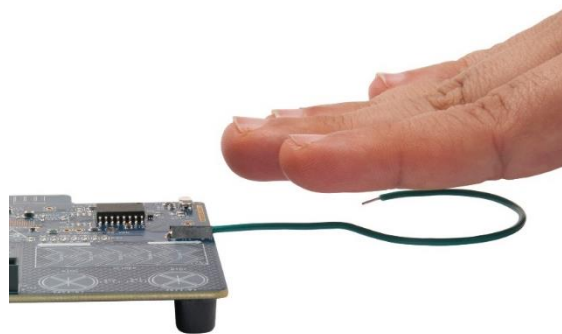


Figure 14. CapSense Proximity Notification Received

Attributes

Discover All Attributes | Pair | Enable All Notifications | Disable All Notifications | Read All Characteristics | Export | Clear | View: Category

Handle	UUID	UUID Description	Value	Properties
0x0004	0x2803	Characteristic Declaration	02:05:00:01:2A	
0x0005	0x2A01	Appearance		0x02
Characteristic Declaration: Peripheral Preferred Connection Parameters				
0x0006	0x2803	Characteristic Declaration	02:07:00:04:2A	
0x0007	0x2A04	Peripheral Preferred Connection Parameters		0x02
Primary Service Declaration: Generic Attribute				
0x0008	0x2800	Primary Service Declaration	01:18 (Generic Attribute)	
Characteristic Declaration: Service Changed				
0x0009	0x2803	Characteristic Declaration	20:0A:00:05:2A	
0x000A	0x2A05	Service Changed		0x20
0x000B	0x2902	Client Characteristic Configuration		
Primary Service Declaration				
0x000C	0x2800	Primary Service Declaration	31:01:9B:5F:80:00:00:80:00:10:00:00:B5:CA:03:00	
Characteristic Declaration				
0x000D	0x2803	Characteristic Declaration	12:0E:00:31:01:9B:5F:80:00:00:80:00:10:00:00:A1:CA:03:00	
0x000E	0003CAA100001000800000805F9B0131		EC	0x12
0x000F	0x2902	Client Characteristic Configuration	00:00	
0x0010	0x2901	Characteristic User Description		

Attribute Details | Send Commands

Handle: 0x000F  
 UUID: 0x2902  
 UUID Description: Client Characteristic Configuration  
 Value: 00:00

Read Value | Write Value

Properties	Enabled
Broadcast	
Read	
Write without response	
Write	
Notify	
Indicate	
Authenticated signed writes	
Extended properties	

11. To disable notifications, modify the **Value** field of the **Client Characteristic Configuration** descriptor to '00:00' and click **Write Value**. Alternatively, you can press **Disable All Notifications** to disable the notifications of all services.

Figure 15. Disable Notifications

Attributes

Discover All Attributes | Pair | Enable All Notifications | **Disable All Notifications** | Read All Characteristics | Export | Clear | View: Category

Handle	UUID	UUID Description	Value	Properties
0x0004	0x2803	Characteristic Declaration	02:05:00:01:2A	
0x0005	0x2A01	Appearance		0x02
Characteristic Declaration: Peripheral Preferred Connection Parameters				
0x0006	0x2803	Characteristic Declaration	02:07:00:04:2A	
0x0007	0x2A04	Peripheral Preferred Connection Parameters		0x02
Primary Service Declaration: Generic Attribute				
0x0008	0x2800	Primary Service Declaration	01:18 (Generic Attribute)	
Characteristic Declaration: Service Changed				
0x0009	0x2803	Characteristic Declaration	20:0A:00:05:2A	
0x000A	0x2A05	Service Changed		0x20
0x000B	0x2902	Client Characteristic Configuration		
Primary Service Declaration				
0x000C	0x2800	Primary Service Declaration	31:01:9B:5F:80:00:00:80:00:10:00:00:B5:CA:03:00	
Characteristic Declaration				
0x000D	0x2803	Characteristic Declaration	12:0E:00:31:01:9B:5F:80:00:00:80:00:10:00:00:A1:CA:03:00	
0x000E	0003CAA100001000800000805F9B0131		00	0x12
0x000F	0x2902	Client Characteristic Configuration	00:00	
0x0010	0x2901	Characteristic User Description		

Attribute Details | Send Commands

Handle: 0x000F  
 UUID: 0x2902  
 UUID Description: Client Characteristic Configuration  
 Value: 00:00

Read Value | **Write Value**

Properties	Enabled
Broadcast	
Read	
Write without response	
Write	
Notify	
Indicate	
Authenticated signed writes	
Extended properties	



12. To disconnect from the device, click **Disconnect**, as shown in Figure 16. The red LED (**LED9**) will turn ON for three seconds to indicate a disconnect event. Press **SW2** to restart the advertisement, if required.

Figure 16. Disconnect from the Device

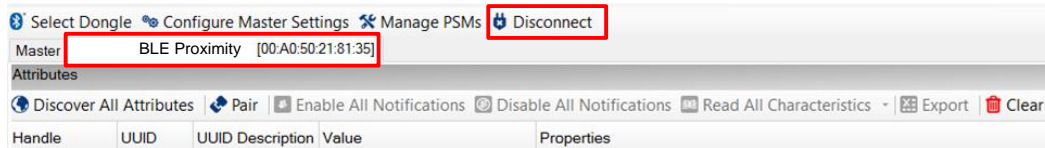


Figure 17. Disconnect Indication

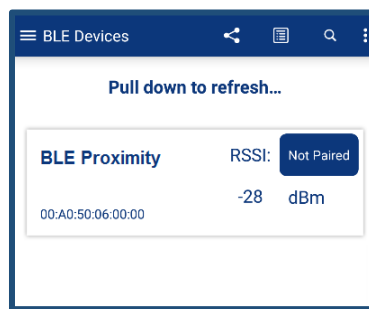


## CySmart Mobile Application

To verify the *CE218135\_BLE\_Proximity* code example using the CySmart mobile application (See the [CySmart Mobile App webpage](#)), follow these steps:

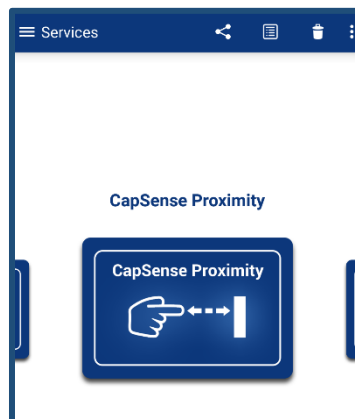
1. Install the CySmart app.
2. Power the Pioneer Board through the USB connector **J10**.
3. Program the Pioneer Board with the *CE218135\_BLE\_Proximity* project. See the [Pioneer Kit guide](#) for details on how to program firmware into the device.  
 After programming successfully, the E-INK display will refresh and show the instructions to use this code example and the BLE will start advertising. The advertising timeout is configured to be 20 seconds. The orange LED (**LED8**) remains ON during this period to indicate the BLE advertising state.
4. If the BLE advertisement has timed out (**LED8** is OFF), press **SW2** to restart advertisement. See the figures in the prior section for LED and switch locations.
5. Open the CySmart app on the mobile device. If Bluetooth is not enabled on the device, the application will ask to enable it.
6. After Bluetooth is enabled, the CySmart mobile application will automatically search for available devices and will list them. Select the **BLE Proximity** peripheral as shown in [Figure 18](#). A successful connection is indicated by **LED8** continuously blinking at half-second intervals.

Figure 18. BLE Proximity Peripheral



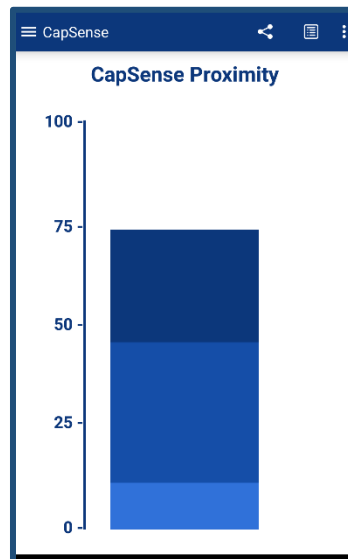
7. When connected, the CySmart mobile application will list the services supported by the device. Scroll and select the CapSense Proximity icon, as shown in [Figure 19](#).

Figure 19. CapSense Proximity Service Page



8. Bring your hand close to the proximity sensor, as shown in [Figure 13](#), and see a similar response on the CapSense Proximity bar graph in the CySmart application (see [Figure 20](#)).

Figure 20. CapSense Proximity Response



9. On the service selection page, there is also a "GATT DB" selection, which allows you to examine the GATT database directly. From this page, you can read and write characteristics as well as enable and disable notifications.
10. If the CySmart app is closed, or Bluetooth is turned OFF, the red LED (**LED9**) will turn ON for three seconds to indicate a disconnect event. Press **SW2** to restart the advertisement, if required.

## Components

Table 2. List of PSoC Creator Components

Component	Instance Name	Function
BLE	BLE	The BLE Component is configured for Limited Discovery with a Custom Characteristic that is used for notifying the BLE Central device of CapSense Proximity data.
CapSense	CapSense	The CapSense Component scans a proximity sensor with SmartSense (hardware parameters only). Thresholds are manually set to provide high noise immunity and a consistent behavior.
MCWDT	MCWDT	The MCWDT Counter0 is configured to generate periodic interrupts at 0.5-second intervals. MCWDT interrupts are used to control status LEDs and turn them OFF when not required to save power.
Digital Output Pin	Pin_LED_Red	These GPIOs are configured as firmware-controlled digital output pins that control status LEDs.
	Pin_LED_Orange	
	Pin_RGB_Red	
	Pin_RGB_Blue	
Digital Input Pin	Pin_RGB_Green	These GPIOs are configured as digital output pins with hardware connections. These pins route PWM signals to RGB LED.
	Advertise	This pin is configured as a digital input pin that is used to generate interrupts when the user button ( <b>SW2</b> ) is pressed.
Global Signal Reference	GlobalSignal	The global signal component is configured to extract interrupts from <b>Advertise</b> pin.

**Note:** See the code example [CE218133 – PSoC 6 MCU E-INK Display with CapSense](#) for more details on components used by E-INK library.

See the PSoC Creator project for more details of PSoC Component configurations and design-wide resource settings.

## Related Documents

Application Notes	
<a href="#">AN210781 – Getting Started with PSoC 6 MCU with Bluetooth Low Energy (BLE) Connectivity</a>	Describes PSoC 63 with Bluetooth Low Energy (BLE) Connectivity and how to build your first PSoC Creator project.
PSoC Creator Component Datasheets	
<a href="#">Bluetooth Low Energy</a>	Facilitates designing applications requiring BLE connectivity.
<a href="#">CapSense</a>	Provides guidelines to use the CapSense component.
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: CE218135 – PSoC 6 MCU with BLE Connectivity: BLE with Proximity

Document Number: 002-18135

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	5654180	NIDH	03/22/2017	New code example.
*A	5861843	NIDH	08/23/2017	Initial public release version
*B	5890173	NIDH	09/20/2017	Public release version
*C	6001000	NIDH	12/13/2017	Updated template and minor text edits. Updated project to PSoC Creator 4.2 Beta.

## 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

[Forums](#) | [WICED IOT 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, 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. 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.