CE220331 – PSoC 6 MCU with BLE Connectivity: BLE with User Interface (RTOS)

Objective

This code example demonstrates interfacing PSoC® 6 MCU with BLE Connectivity (PSoC 6 MCU) with user interface functions such as an RGB LED, and touch sensors based on self and mutual capacitance (CapSense® CSD and CSX). These functions provide bi-directional BLE connectivity between the PSoC 6 MCU and a PC running the CySmart™ BLE Host Emulation tool or a mobile device running the CySmart mobile application.

Overview

This code example demonstrates interfacing PSoC 6 MCU with BLE Connectivity with an RGB LED with color and intensity control, touch buttons based on mutual capacitance (CSX), and touch-slider-based on self-capacitance (CSD). This code example also shows connectivity between the PSoC 6 BLE (acting as a Peripheral and GATT Server) and a PC running the CySmart BLE Host Emulation tool or a mobile device running the CySmart mobile application (acting as a Central and GATT Client). Custom BLE services are used for CapSense touch sensing and LED control.

In more detail:

- RGB LED color and intensity control using configurable digital blocks of PSoC 6 MCU
- CapSense slider and buttons
- PSoC 6 MCU's ability to simultaneously scan touch sensors based on self-capacitance as well as mutual-capacitance
- BLE connectivity
  - Advertisement and connection with a Central device
  - Three custom services (CapSense Slider, CapSense Button, and RGB LED)
  - Data transfer over BLE using notifications, read, and write

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.

This code example uses FreeRTOS. See PSoC 6 101: Lesson 1-4 FreeRTOS training video to learn how to create a PSoC 6 FreeRTOS project with PSoC® Creator™. Visit the FreeRTOS website for documentation and API references of FreeRTOS.

Requirements

Tool: PSoC Creator™ 4.2; Peripheral Driver Library (PDL) 3.0.1

Programming Language: C (Arm® GCC 5.4.1)

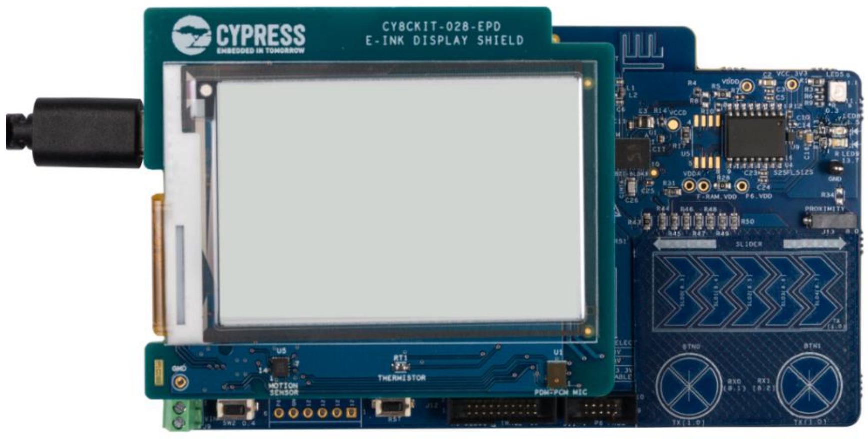
Associated Parts: All PSoC 6 MCUs with BLE Connectivity

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

Hardware Setup

Plug in the E-INK display shield on to the Pioneer board as Figure 1 shows.

Figure 1. 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_{DDD} / KitProg2	Back
J8	Installed	Back

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.

Note: For this code example, the CapSense Button service is not available in the CySmart iOS app. Use the host emulation tool or the Android app to evaluate this service.

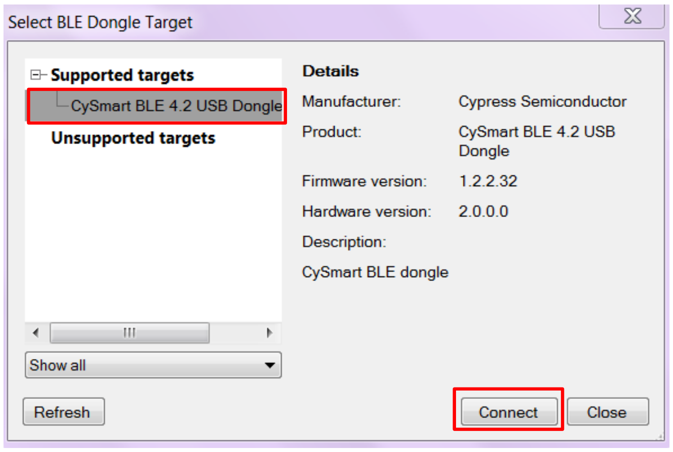
CySmart BLE Host Emulation Tool

To verify the *CE220331_BLE_UI_RTOS* 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 2. Connect to BLE Dongle



Details

Manufacturer:	Cypress Semiconductor
Product:	CySmart BLE 4.2 USB Dongle
Firmware version:	1.2.2.32
Hardware version:	2.0.0.0
Description:	

CySmart BLE dongle

3. Power the Pioneer Board through the USB connector **J10**.
4. Program the Pioneer Board with the *CE220331_BLE_UI_RTOS* project. See the Pioneer Kit guide for details on how to program firmware into the device.

   After programming, 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 3 shows. Table 2 lists all the LED indications corresponding to BLE states.

Figure 3. BLE Advertising

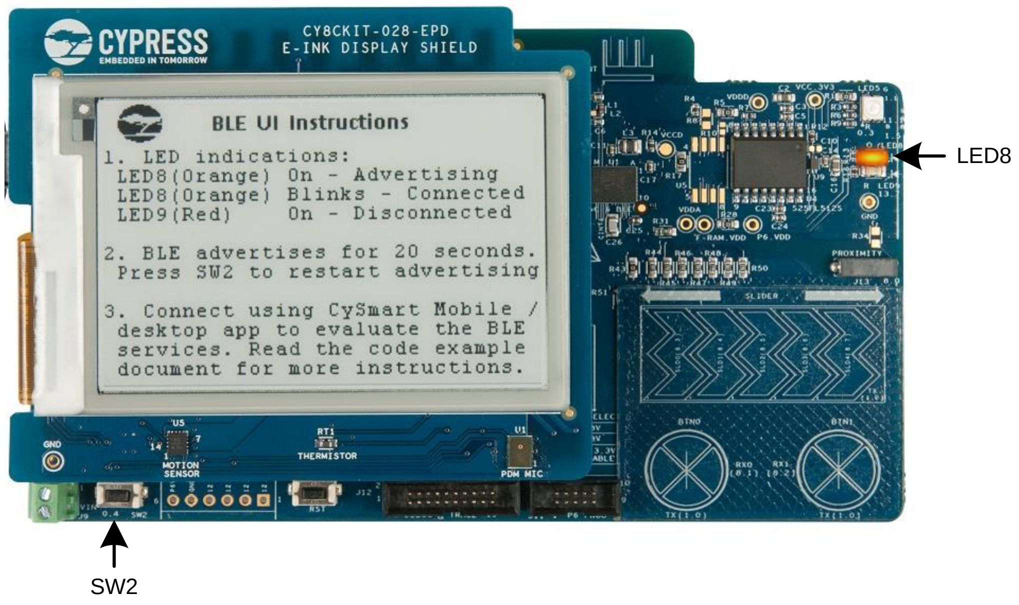
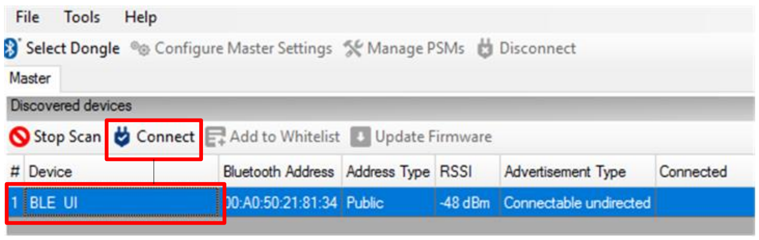

CYPRESS EMBEDDED IN TOMORROW
CY8CKIT-028-EPD
E-INK DISPLAY SHIELD
BLE UI Instructions
1. LED indications:
LED8(Orange) On - Advertising
LED8(Orange) Blinks - Connected
LED9(Red) On - Disconnected
2. BLE advertises for 20 seconds.
Press SW2 to restart advertising
3. Connect using CySmart Mobile / desktop app to evaluate the BLE services. Read the code example document for more instructions.
LED8
SW2

Table 2. LED Indications

BLE State	LED Indication	
Advertising	LED8 (Orange)	On
	LED9 (Red)	Off
Connected	LED8 (Orange)	Toggling continuously
	LED9 (Red)	Off
Disconnected	LED8 (Orange)	Off
	LED9 (Red)	Blinks once
Idle (following advertisement timeout or disconnection)	LED8 (Orange)	Off
	LED9 (Red)	Off

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 UI** device to connect, or click **BLE UI** and then click **Connect**. A successful connection is indicated by **LED8** continuously blinking at half second intervals.

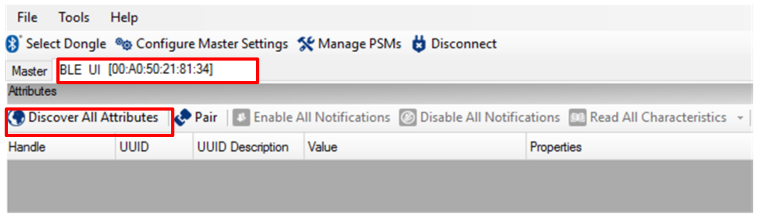
Figure 4. Connect to BLE Slider and LED Peripheral



#	Device	Bluetooth Address	Address Type	RSSI	Advertisement Type	Connected
1	BLE UI	00:A0:50:21:81:34	Public	-48 dBm	Connectable undirected	

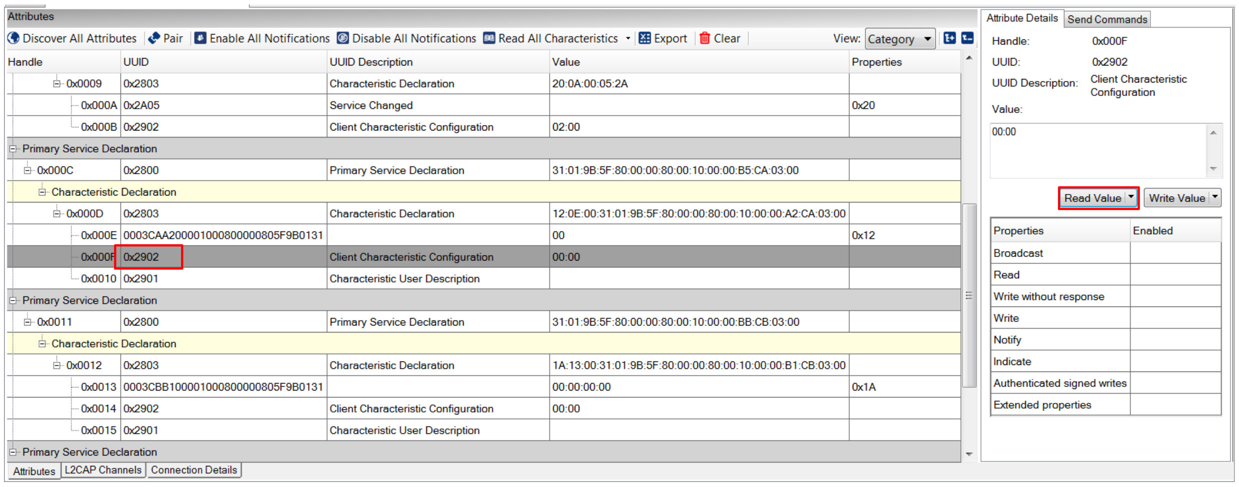
7. Click **Discover All Attributes** to find all attributes supported.

Figure 5. Discover All Attributes



8. Locate the attribute **Client Characteristic Configuration** descriptor (UUID 0x2902) under the CapSense Slider characteristic (**UUID 0x0003CAA2000010008000000805F9B0131**). Click **Read Value** to read the existing Client Characteristic Configuration Descriptor (CCCD) value as shown in Figure 6.

Figure 6. Read CCCD for CapSense Slider Characteristic



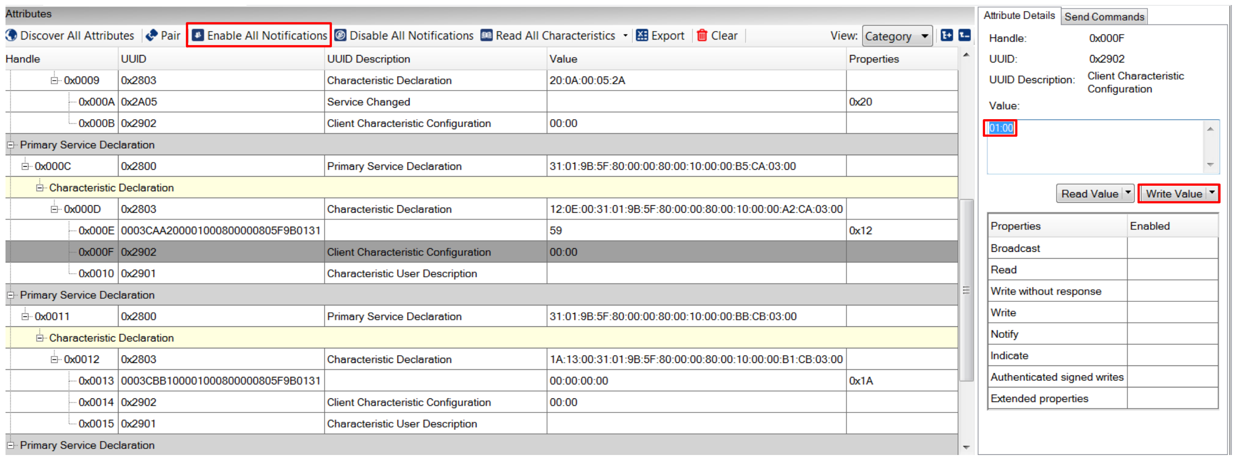
Handle	UUID	UUID Description	Value	Properties
0x0009	0x2803	Characteristic Declaration	20:0A:00:05:2A	
0x000A	0x2A05	Service Changed		0x20
0x000B	0x2902	Client Characteristic Configuration	02:00	
Primary Service Declaration				
0x000C	0x2800	Primary Service Declaration	31:01:9B:5F:80:00:00:80:00:10:00:00:B5:CA:03:00	
Characteristic Declaration				
0x000D	0x2803	Characteristic Declaration	12:0E:00:31:01:9B:5F:80:00:00:80:00:10:00:00:A2:CA:03:00	
0x000E	0003CAA2000010008000000805F9B0131		00	0x12
0x000F	0x2902	Client Characteristic Configuration	00:00	
0x0010	0x2901	Characteristic User Description		
Primary Service Declaration				
0x0011	0x2800	Primary Service Declaration	31:01:9B:5F:80:00:00:80:00:10:00:00:BB:CB:03:00	
Characteristic Declaration				
0x0012	0x2803	Characteristic Declaration	1A:13:00:31:01:9B:5F:80:00:00:80:00:10:00:00:B1:CB:03:00	
0x0013	0003CBB1000010008000000805F9B0131		00:00:00:00	0x1A
0x0014	0x2902	Client Characteristic Configuration	00:00	
0x0015	0x2901	Characteristic User Description		
Primary Service Declaration				

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

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

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

Figure 7. Write CCCD to Enable Notifications



Handle	UUID	UUID Description	Value	Properties
0x0009	0x2803	Characteristic Declaration	20:0A:00:05:2A	
0x000A	0x2A05	Service Changed		0x20
0x000B	0x2902	Client Characteristic Configuration	00:00	
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:A2:CA:03:00	
0x000E	0003CAA2000010008000000805F9B0131		59	0x12
0x000F	0x2902	Client Characteristic Configuration	00:00	
0x0010	0x2901	Characteristic User Description		
Primary Service Declaration				
0x0011	0x2800	Primary Service Declaration	31:01:9B:5F:80:00:00:80:00:10:00:00:BB:CB:03:00	
Characteristic Declaration				
0x0012	0x2803	Characteristic Declaration	1A:13:00:31:01:9B:5F:80:00:00:80:00:10:00:00:B1:CB:03:00	
0x0013	0003CBB1000010008000000805F9B0131		00:00:00:00	0x1A
0x0014	0x2902	Client Characteristic Configuration	00:00	
0x0015	0x2901	Characteristic User Description		
Primary Service Declaration				

Handle: 0x000F
UUID: 0x2902
UUID Description: Client Characteristic Configuration
Value:
01:00

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

10. Swipe your finger on the CapSense slider on the Pioneer Board, as shown in Figure 8 and see the notification values in the CapSense Slider value field, as shown in Figure 9.

Note: The sensor auto-reset feature is enabled for CapSense sensors. Pressing and holding a CapSense sensor for more than three seconds will reset the sensor to the inactive state. This feature will prevent stuck-on sensors under rapidly changing environments – see the CapSense Component datasheet for additional information.

Figure 8. CapSense Slider

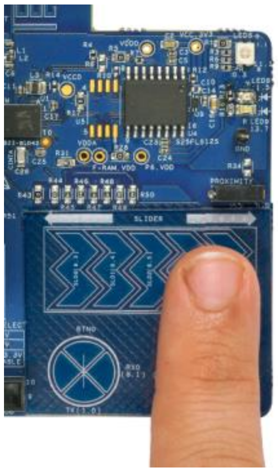
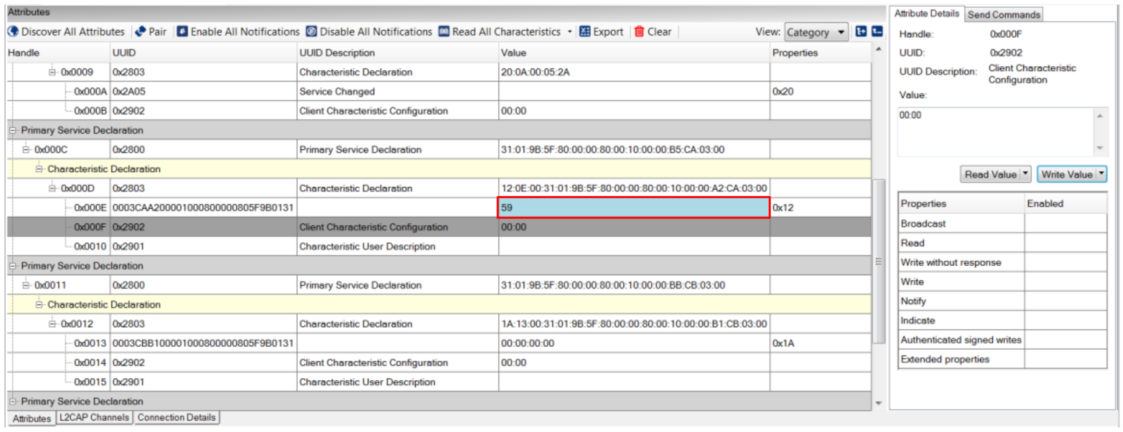


Figure 9. CapSense Slider Notification Received



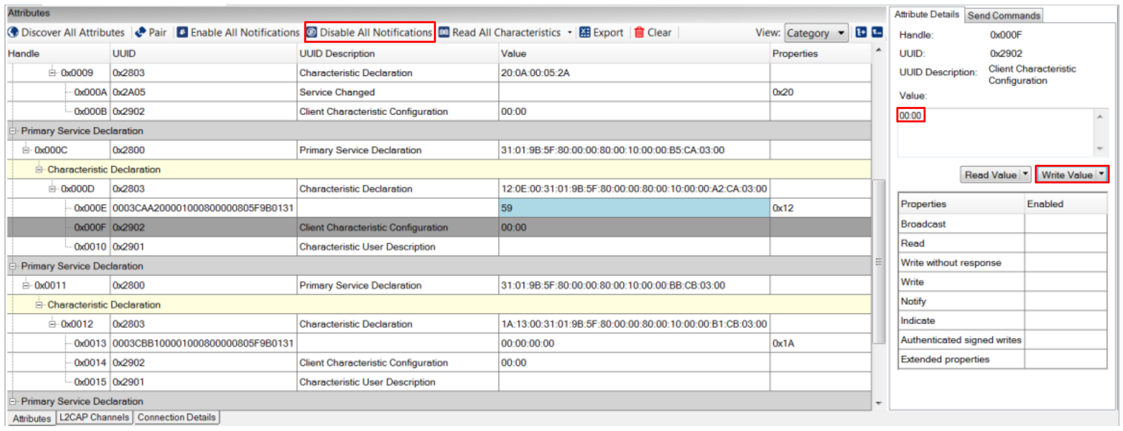
Handle	UUID	UUID Description	Value	Properties
0x0009	0x2803	Characteristic Declaration	20:0A:00:05:2A	
0x000A	0x2A05	Service Changed		0x20
0x000B	0x2902	Client Characteristic Configuration	00:00	
Primary Service Declaration				
0x000C	0x2800	Primary Service Declaration	31:01:9B:5F:80:00:00:80:00:10:00:00:B5:CA:03:00	
Characteristic Declaration				
0x000D	0x2803	Characteristic Declaration	12:0E:00:31:01:9B:5F:80:00:00:80:00:10:00:00:A2:CA:03:00	
0x000E	0003CAA2000010008000008 05F9B0131		59	0x12
0x000F	0x2902	Client Characteristic Configuration	00:00	
0x0010	0x2901	Characteristic User Description		
Primary Service Declaration				
0x0011	0x2800	Primary Service Declaration	31:01:9B:5F:80:00:00:80:00:10:00:00:BB:CB:03:00	
Characteristic Declaration				
0x0012	0x2803	Characteristic Declaration	1A:13:00:31:01:9B:5F:80:00:00:80:00:10:00:00:B1:CB:03:00	
0x0013	0003CBB1000010008000008 05F9B0131		00:00:00:00	0x1A
0x0014	0x2902	Client Characteristic Configuration	00:00	
0x0015	0x2901	Characteristic User Description		
Primary Service Declaration				

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

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 the **Disable All Notifications** button to disable the notifications of all services.

Figure 10. Disable Notifications



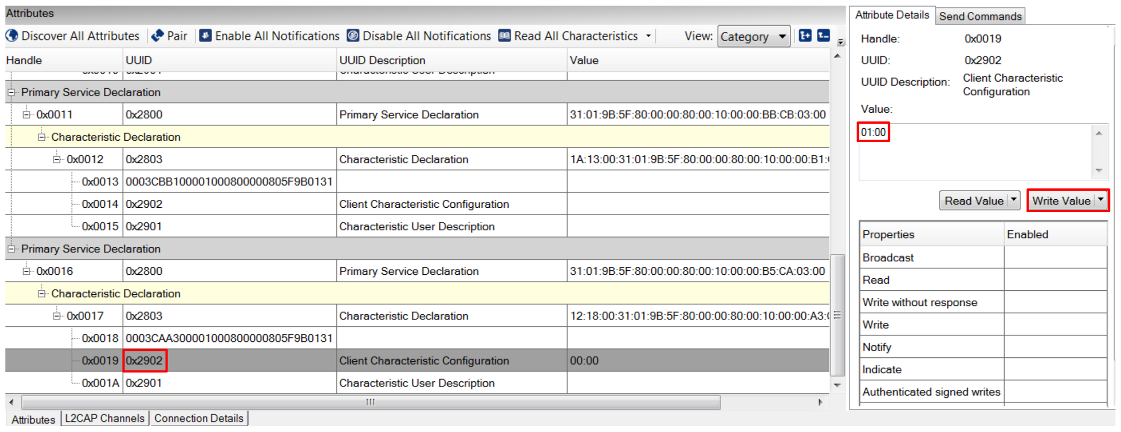
Handle	UUID	UUID Description	Value	Properties
0x0009	0x2803	Characteristic Declaration	20:0A:00:05:2A	
0x000A	0x2A05	Service Changed		0x20
0x000B	0x2902	Client Characteristic Configuration	00:00	
Primary Service Declaration				
0x000C	0x2800	Primary Service Declaration	31:01:9B:5F:80:00:00:80:00:10:00:00:B5:CA:03:00	
Characteristic Declaration				
0x000D	0x2803	Characteristic Declaration	12:0E:00:31:01:9B:5F:80:00:00:80:00:10:00:00:A2:CA:03:00	
0x000E	0003CAA2000010008000008 05F9B0131		59	0x12
0x000F	0x2902	Client Characteristic Configuration	00:00	
0x0010	0x2901	Characteristic User Description		
Primary Service Declaration				
0x0011	0x2800	Primary Service Declaration	31:01:9B:5F:80:00:00:80:00:10:00:00:BB:CB:03:00	
Characteristic Declaration				
0x0012	0x2803	Characteristic Declaration	1A:13:00:31:01:9B:5F:80:00:00:80:00:10:00:00:B1:CB:03:00	
0x0013	0003CBB1000010008000008 05F9B0131		00:00:00:00	0x1A
0x0014	0x2902	Client Characteristic Configuration	00:00	
0x0015	0x2901	Characteristic User Description		
Primary Service Declaration				

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

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

12. Locate the attribute **Client Characteristic Configuration** descriptor (UUID 0x2902) under CapSense Button characteristic (**UUID 0x0003CAA3000010008000000805F9B0131**), read the value and enable the notification as described in steps 8 and 9.

Figure 11. Enable CapSense Button Notification



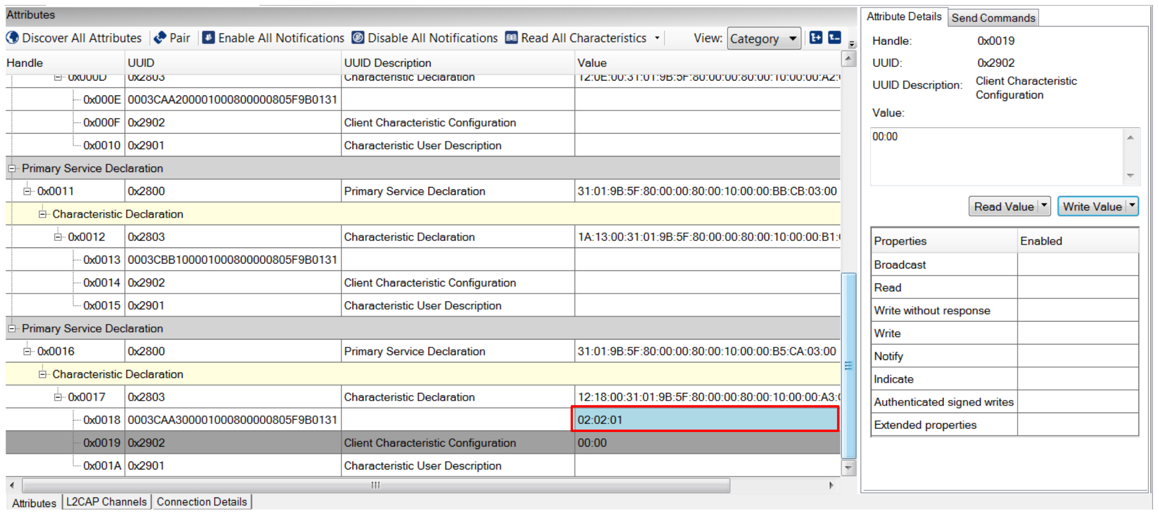
Handle	UUID	UUID Description	Value
Primary Service Declaration			
0x0011	0x2800	Primary Service Declaration	31:01:9B:5F:80:00:00:80:00:10:00:00:BB:CB:03:00
Characteristic Declaration			
0x0012	0x2803	Characteristic Declaration	1A:13:00:31:01:9B:5F:80:00:00:80:00:10:00:00:B1:
0x0013	0003CBB1000010008000000805F9B0131		
0x0014	0x2902	Client Characteristic Configuration	
0x0015	0x2901	Characteristic User Description	
Primary Service Declaration			
0x0016	0x2800	Primary Service Declaration	31:01:9B:5F:80:00:00:80:00:10:00:00:B5:CA:03:00
Characteristic Declaration			
0x0017	0x2803	Characteristic Declaration	12:18:00:31:01:9B:5F:80:00:00:80:00:10:00:00:A3:
0x0018	0003CAA3000010008000000805F9B0131		
0x0019	0x2902	Client Characteristic Configuration	00:00
0x001A	0x2901	Characteristic User Description	

Handle: 0x0019
UUID: 0x2902
UUID Description: Client Characteristic Configuration
Value:
01:00

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

13. Touch the CapSense buttons on the Pioneer Board, and see the notification values in the CapSense Button value field, as shown in Figure 12. The LSB (byte 0) indicates the button mask – 0 when no buttons are active, 1 when BTN0 is active, 2 when BTN1 is active, and 3 when both buttons are active. See step 11 for instructions to disable notifications.

Figure 12. CapSense Button Notification Received



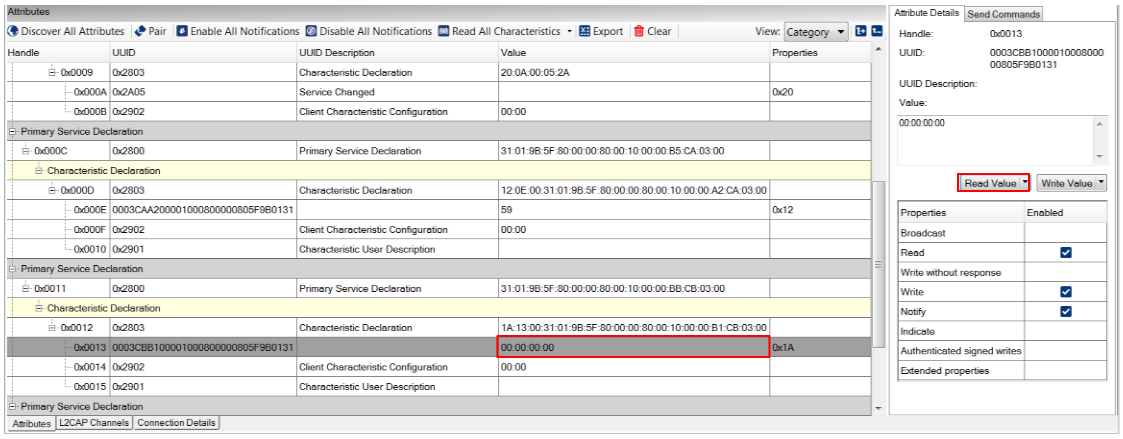
Handle	UUID	UUID Description	Value
0x000E	0003CAA2000010008000000805F9B0131		
0x000F	0x2902	Client Characteristic Configuration	
0x0010	0x2901	Characteristic User Description	
Primary Service Declaration			
0x0011	0x2800	Primary Service Declaration	31:01:9B:5F:80:00:00:80:00:10:00:00:BB:CB:03:00
Characteristic Declaration			
0x0012	0x2803	Characteristic Declaration	1A:13:00:31:01:9B:5F:80:00:00:80:00:10:00:00:B1:
0x0013	0003CBB1000010008000000805F9B0131		
0x0014	0x2902	Client Characteristic Configuration	
0x0015	0x2901	Characteristic User Description	
Primary Service Declaration			
0x0016	0x2800	Primary Service Declaration	31:01:9B:5F:80:00:00:80:00:10:00:00:B5:CA:03:00
Characteristic Declaration			
0x0017	0x2803	Characteristic Declaration	12:18:00:31:01:9B:5F:80:00:00:80:00:10:00:00:A3:
0x0018	0003CAA3000010008000000805F9B0131		02:02:01
0x0019	0x2902	Client Characteristic Configuration	00:00
0x001A	0x2901	Characteristic User Description	

Handle: 0x0019
UUID: 0x2902
UUID Description: Client Characteristic Configuration
Value:
00:00

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

14. Locate the **RGB LED Control** characteristic (**UUID 0x0003CBB1-0000-1000-8000-00805F9B0131**). Click **Read Value** to read the existing 4-byte onboard RGB LED color information, as shown in Figure 13. The four bytes indicate red, green, blue, and the overall intensity, respectively.

Figure 13. Read RGB LED Control Characteristic Value



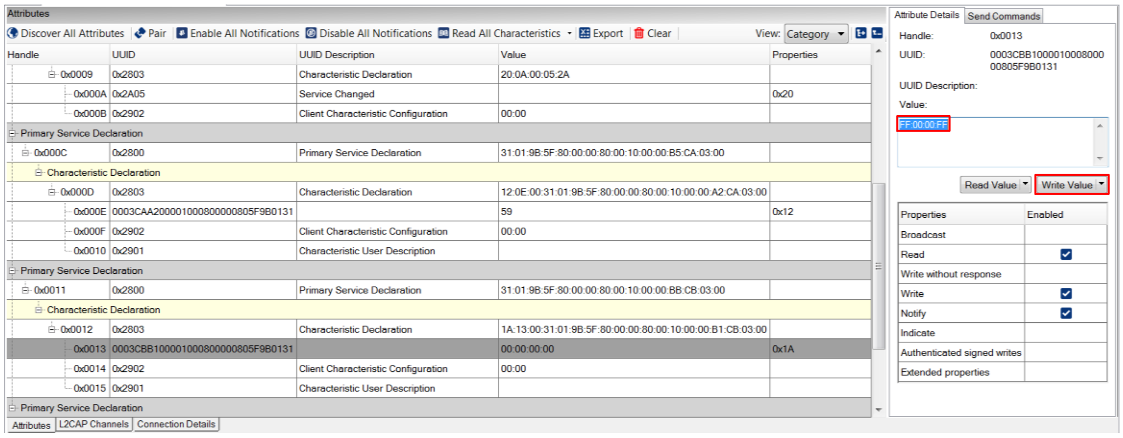
Handle	UUID	UUID Description	Value	Properties
0x0009	0x2803	Characteristic Declaration	20:0A:00:05:2A	
0x000A	0x2A05	Service Changed		0x20
0x000B	0x2902	Client Characteristic Configuration	00:00	
Primary Service Declaration				
0x000C	0x2800	Primary Service Declaration	31:01:9B:5F:80:00:00:80:00:10:00:00:B5:CA:03:00	
Characteristic Declaration				
0x000D	0x2803	Characteristic Declaration	12:0E:00:31:01:9B:5F:80:00:00:80:00:10:00:00:A2:CA:03:00	
0x000E	0003CAA2000010008000000805F9B0131		59	0x12
0x000F	0x2902	Client Characteristic Configuration	00:00	
0x0010	0x2901	Characteristic User Description		
Primary Service Declaration				
0x0011	0x2800	Primary Service Declaration	31:01:9B:5F:80:00:00:80:00:10:00:00:BB:CB:03:00	
Characteristic Declaration				
0x0012	0x2803	Characteristic Declaration	1A:13:00:31:01:9B:5F:80:00:00:80:00:10:00:00:B1:CB:03:00	
0x0013	0003CBB1000010008000000805F9B0131		00:00:00:00	0x1A
0x0014	0x2902	Client Characteristic Configuration	00:00	
0x0015	0x2901	Characteristic User Description		
Primary Service Declaration				

Handle: 0x0013
UUID: 0003CBB1000010008000 00805F9B0131
UUID Description:
Value:
00:00:00:00

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

15. Modify the four bytes of data in the **Value** field to **FF:00:00:FF** and click **Write Value**, as shown in Figure 14. You will see the corresponding change in the color (Red) and intensity (full intensity) of the RGB LED on the Pioneer Board as shown in Figure 15.

Figure 14. Write RGB LED Control Characteristic Value

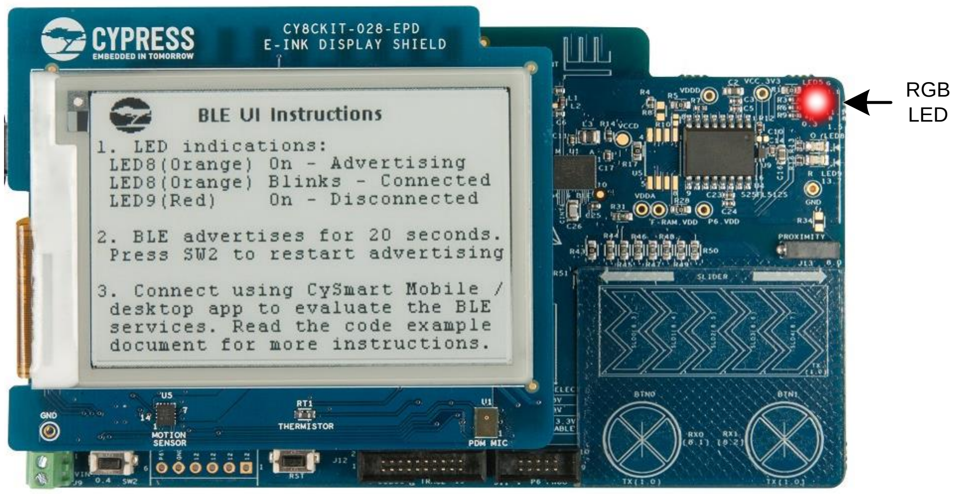


Handle	UUID	UUID Description	Value	Properties
0x0009	0x2803	Characteristic Declaration	20:0A:00:05:2A	
0x000A	0x2A05	Service Changed		0x20
0x000B	0x2902	Client Characteristic Configuration	00:00	
Primary Service Declaration				
0x000C	0x2800	Primary Service Declaration	31:01:9B:5F:80:00:00:80:00:10:00:00:B5:CA:03:00	
Characteristic Declaration				
0x000D	0x2803	Characteristic Declaration	12:0E:00:31:01:9B:5F:80:00:00:80:00:10:00:00:A2:CA:03:00	
0x000E	0003CAA2000010008000000805F9B0131		59	0x12
0x000F	0x2902	Client Characteristic Configuration	00:00	
0x0010	0x2901	Characteristic User Description		
Primary Service Declaration				
0x0011	0x2800	Primary Service Declaration	31:01:9B:5F:80:00:00:80:00:10:00:00:BB:CB:03:00	
Characteristic Declaration				
0x0012	0x2803	Characteristic Declaration	1A:13:00:31:01:9B:5F:80:00:00:80:00:10:00:00:B1:CB:03:00	
0x0013	0003CBB1000010008000000805F9B0131		00:00:00:00	0x1A
0x0014	0x2902	Client Characteristic Configuration	00:00	
0x0015	0x2901	Characteristic User Description		
Primary Service Declaration				

Handle: 0x0013
UUID: 0003CBB1000010008000 00805F9B0131
UUID Description:
Value:
FF:00:00:FF

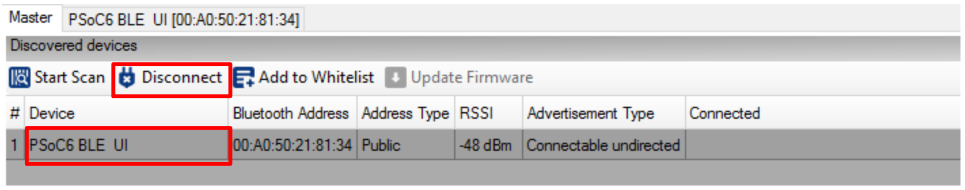
Properties	Enabled
Broadcast	
Read	✓
Write without response	
Write	✓
Notify	✓
Indicate	
Authenticated signed writes	
Extended properties	

Figure 15. RGB LED Control with BLE


CYPRESS
EMBEDDED IN TOMORROW
CY8CKIT-028-EPD
E-INK DISPLAY SHIELD
BLE UI Instructions
1. LED indications:
LED8(Orange) On - Advertising
LED8(Orange) Blinks - Connected
LED9(Red) On - Disconnected
2. BLE advertises for 20 seconds.
Press SW2 to restart advertising
3. Connect using CySmart Mobile /
desktop app to evaluate the BLE
services. Read the code example
document for more instructions.
RGB LED

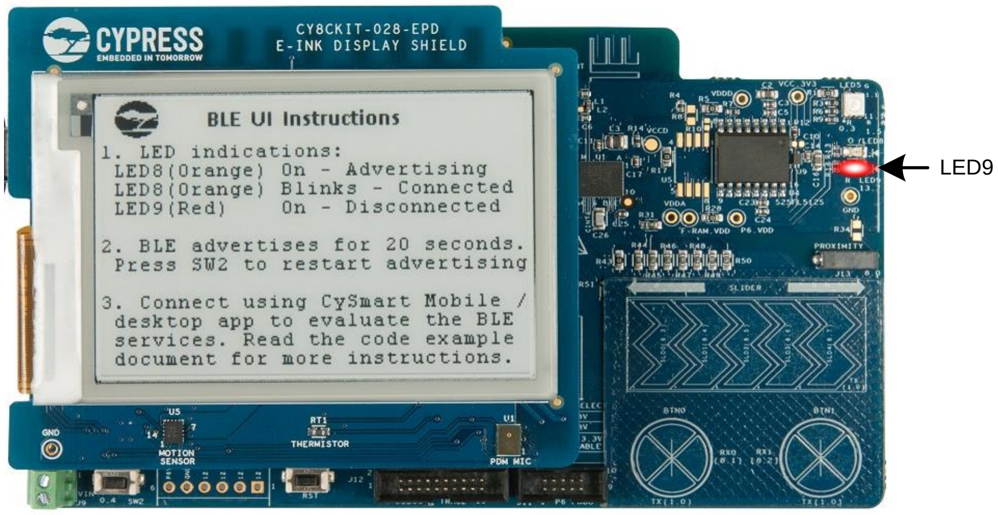
16. To disconnect from the device, click **Disconnect**, as shown in Figure 16. The red LED (**LED9**) will turn ON for three seconds to indicate a disconnect event. Press **SW2** to restart the advertisement, if required.

Figure 16. Disconnect from the Device



#	Device	Bluetooth Address	Address Type	RSSI	Advertisement Type	Connected
1	PSoC6 BLE UI	00:A0:50:21:81:34	Public	-48 dBm	Connectable undirected	

Figure 17. Disconnect Indication


CYPRESS
EMBEDDED IN TOMORROW
CY8CKIT-028-EPD
E-INK DISPLAY SHIELD
BLE UI Instructions
1. LED indications:
LED8(Orange) On - Advertising
LED8(Orange) Blinks - Connected
LED9(Red) On - Disconnected
2. BLE advertises for 20 seconds.
Press SW2 to restart advertising
3. Connect using CySmart Mobile /
desktop app to evaluate the BLE
services. Read the code example
document for more instructions.
LED9

CySmart Mobile Application

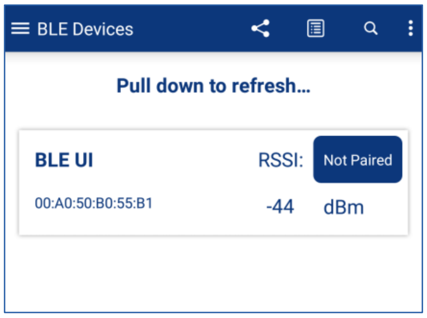
To verify this code example using the CySmart mobile application (refer to the CySmart Mobile App webpage), follow these steps:

Note: For this project, the CapSense Button service is not available in the CySmart iOS app. Use the Android app to evaluate this service.

1. Install the CySmart app.
2. Power the Pioneer Board through the USB connector **J10**.
3. Program the Pioneer Board with the *CE220331_BLE_UI_RTOS* project. See the Pioneer Kit guide for details on how to program firmware into the device.

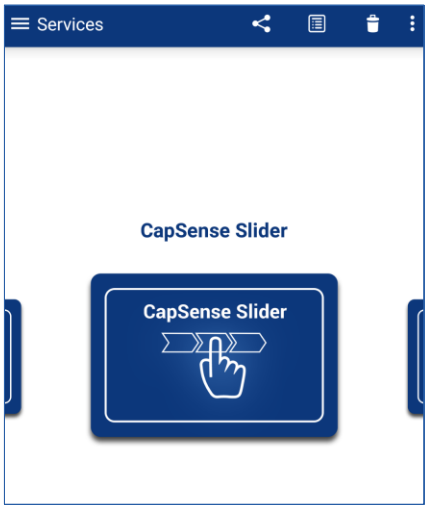
   After programming, the E-INK display refreshes and shows the instructions to use this project; BLE starts advertising. The advertising timeout is configured to be 20 seconds. The orange LED (**LED8**) remains ON during this period to indicate the BLE advertising state.
4. If the BLE advertisement has timed out (**LED8** is OFF), press **SW2** to restart advertisement. See the figures in the earlier 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 prompt you to enable it.
6. After Bluetooth is enabled, the CySmart mobile application automatically searches for available devices and lists them. Select the **BLE UI** peripheral as shown in Figure 18. A successful connection is indicated by **LED8** continuously blinking at half-second intervals.

Figure 18. BLE UI Peripheral



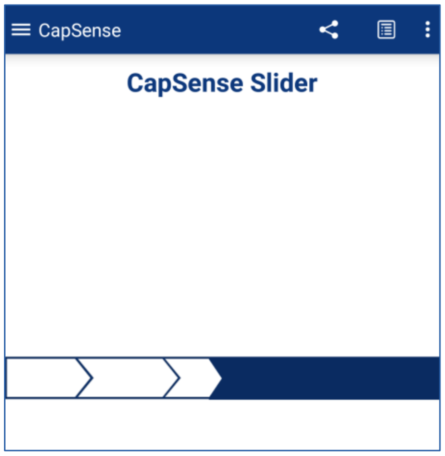
7. When connected, the CySmart mobile application lists the services supported by the device. Scroll and select the CapSense Slider icon, as shown in Figure 19.

Figure 19. CapSense Slider Service Page



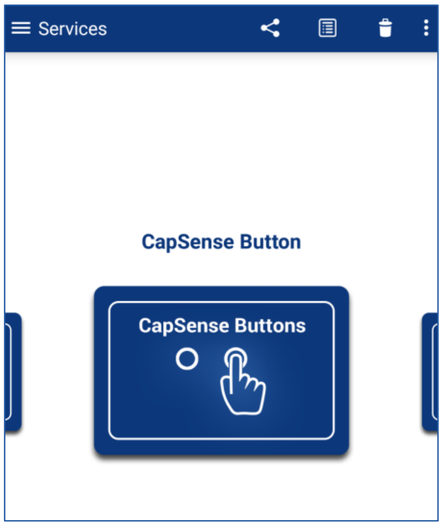
8. Swipe your finger on the CapSense slider on the Pioneer Board and see a similar response on the CapSense Slider page in the CySmart application (see Figure 20).

Figure 20. CapSense Slider



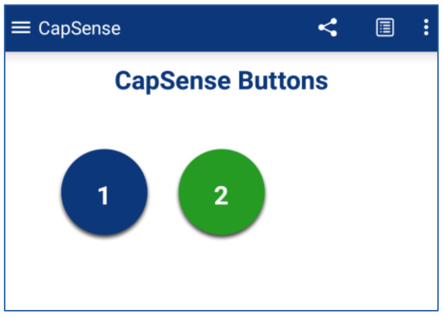
9. Press the back button to return to the service selection page. Scroll and tap on the CapSense Button service.

Figure 21. CapSense Button Service



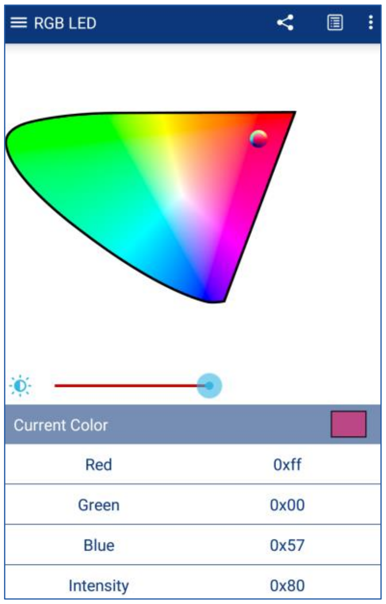
10. Touch CapSense buttons on the Pioneer Board and see a similar response on the CapSense Button page in the CySmart application.

Figure 22. CapSense Buttons



11. Press the back button to return to the service selection page. Scroll and tap on the RGB LED service.
12. On the RGB LED service page, select a color on the color gamut to see a similar color response on the Pioneer Board RGB LED. The slider below the color gamut controls the intensity of the RGB LED color.

Figure 23. RGB LED Control with CySmart Mobile Application



Red	0xff
Green	0x00
Blue	0x57
Intensity	0x80

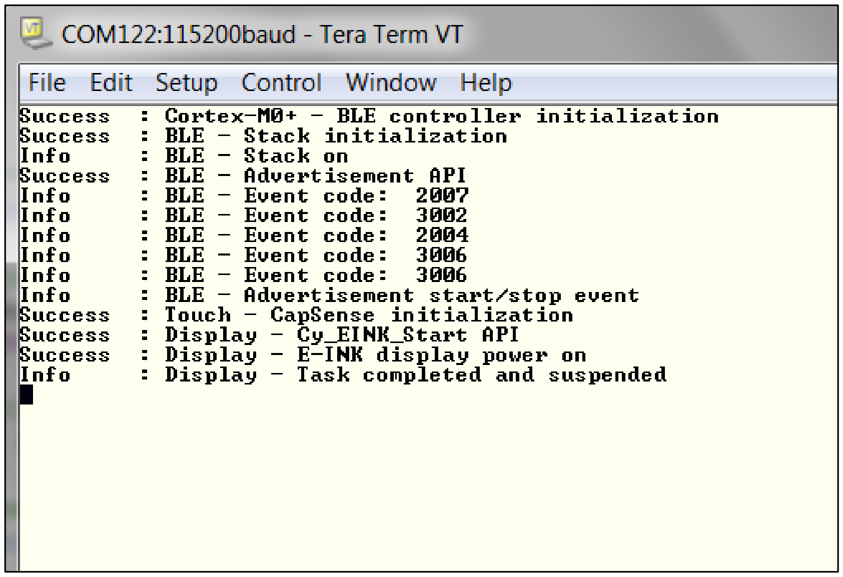
On the service selection page, there is also a "GATT DB" selection, which allows you to examine the GATT database directly. From this page, you can read and write characteristics as well as enable and disable notifications.

13. 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.

Viewing Debug Messages

This code example allows you to view debug messages from various tasks and functions using a serial port terminal emulator such as Tera Term or HyperTerminal as Figure 24 shows.

Figure 24. Viewing Debug Messages using UART


COM122:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
Success   : Cortex-M0+ - BLE controller initialization
Success   : BLE - Stack initialization
Info      : BLE - Stack on
Success   : BLE - Advertisement API
Info      : BLE - Event code:  2007
Info      : BLE - Event code:  3002
Info      : BLE - Event code:  2004
Info      : BLE - Event code:  3006
Info      : BLE - Event code:  3006
Info      : BLE - Advertisement start/stop event
Success   : Touch - CapSense initialization
Success   : Display - Cy_EINK_Start API
Success   : Display - E-INK display power on
Info      : Display - Task completed and suspended
```

This feature is disabled by default for higher performance and power efficiency. You can The UART debug can be enabled by right clicking the UART_DEBUG component in the TopDesign schematic and selecting "enable". In addition, `UART_DEBUG_ENABLE` macro in uart_debug.h file should be set to “true”.

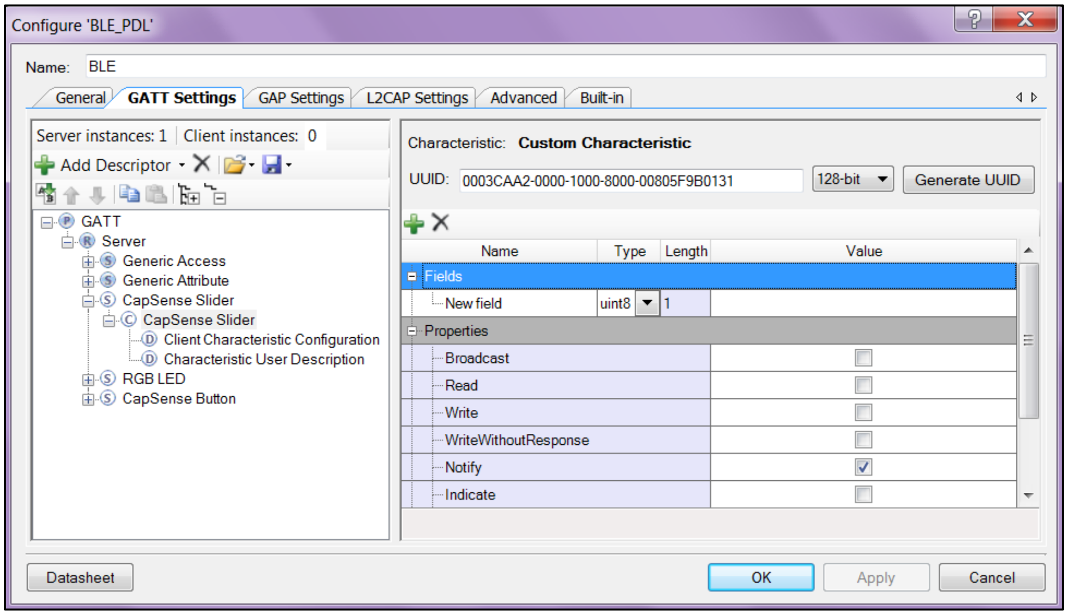
After re-building the projects and re-programming PSoC 6 MCU with the updated project, you should set up a serial port terminal emulator with these settings to view the debug information:

- Baud rate : 115200
- Data size : 8-bit
- Parity : None
- Stop : 1-bit
- Flow Control : None

Design and Implementation

The BLE profile in this code example consists of three BLE custom services: CapSense Slider, CapSense Button, and RGB LED. The two CapSense services consist of custom characteristics that are used to send data as notifications to the GATT client device. The notification data consists of the finger location read by the CapSense Component on the slider and the ON/OFF status of the two CapSense buttons. These characteristics support notification, which allows the GATT server to send data to the connected client device whenever new data is available. The RGB LED service consists of one custom characteristic called RGB LED Control. This characteristic supports three operations (read, write, and notify) through which the connected GATT client device can read data as well as write a new value to the characteristic. This data has four single-byte values indicating red, green, blue, and the intensity to control the onboard RGB LED. The properties for the custom service/characteristics are configured in the BLE Component under the **GATT Settings** tab. As an example, Figure 25 shows the configuration of the CapSense Slider Service.

Figure 25. BLE CapSense Slider Service Configuration


Configure 'BLE_PDL'

Name: BLE

Server instances: 1 | Client instances: 0

```
GATT
  Server
    Generic Access
    Generic Attribute
    CapSense Slider
      CapSense Slider
        Client Characteristic Configuration
        Characteristic User Description
    RGB LED
    CapSense Button
```

Characteristic: **Custom Characteristic**

UUID: 0003CAA2-0000-1000-8000-00805F9B0131 | 128-bit

Name	Type	Length	Value
Fields			
New field	uint8	1	
Properties			
Broadcast			☐
Read			☐
Write			☐
WriteWithoutResponse			☐
Notify			☑
Indicate			☐

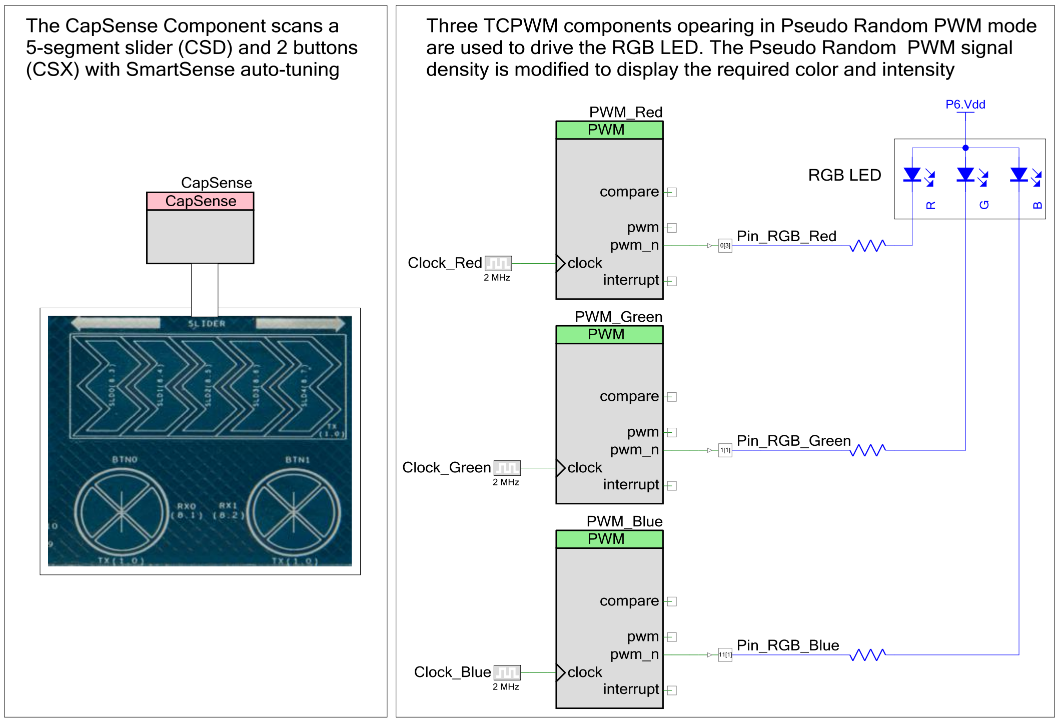
Figure 26, Figure 26, Figure 28 and Figure 29 show the TopDesign schematic of this code example.

The 5-element CapSense CSD slider (self-capacitance), and two CapSense CSX buttons (mutual-capacitance) are scanned with SmartSense™ auto-tuning. See AN85951 - PSoC 4 and PSoC 6 MCU CapSense Design Guide for details of CapSense touch sensing technology and to design capacitive touch sensing applications with PSoC 6 MCU.

Three TCPWM components operating in Pseudo Random PWM mode are used to drive the RGB LED. The Pseudo Random PWM signal density is modified to display the required color and intensity per the data received over BLE.

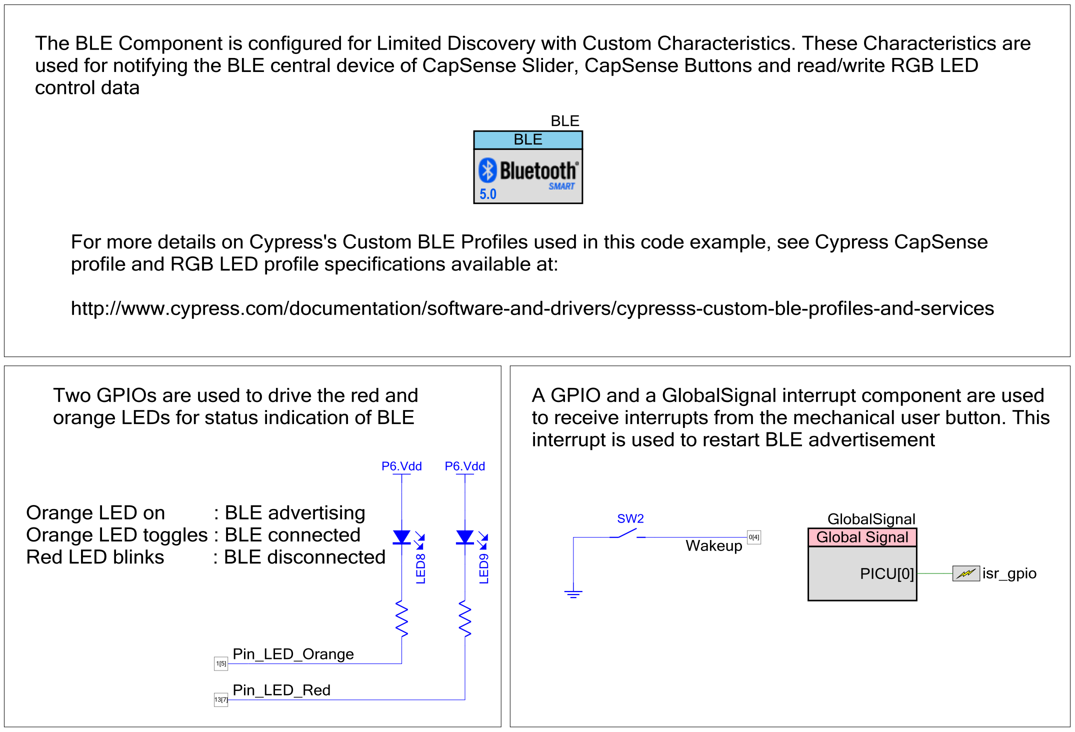
The E-INK display shows the instructions to use this code example at startup and is then turned OFF to save power. E-INK displays consume no power to retain the display. For more details on E-INK display, see the code example CE218133 – PSoC 6 MCU E-INK Display with CapSense.

Figure 26. TopDesign Schematic: User Interface


The CapSense Component scans a 5-segment slider (CSD) and 2 buttons (CSX) with SmartSense auto-tuning

Three TCPWM components opearing in Pseudo Random PWM mode are used to drive the RGB LED. The Pseudo Random PWM signal density is modified to display the required color and intensity

Figure 27. TopDesign Schematic: BLE


The BLE Component is configured for Limited Discovery with Custom Characteristics. These Characteristics are used for notifying the BLE central device of CapSense Slider, CapSense Buttons and read/write RGB LED control data

For more details on Cypress's Custom BLE Profiles used in this code example, see Cypress CapSense profile and RGB LED profile specifications available at:

http://www.cypress.com/documentation/software-and-drivers/cypresss-custom-ble-profiles-and-services

Two GPIOs are used to drive the red and orange LEDs for status indication of BLE

Orange LED on	: BLE advertising
Orange LED toggles	: BLE connected
Red LED blinks	: BLE disconnected

A GPIO and a GlobalSignal interrupt component are used to receive interrupts from the mechanical user button. This interrupt is used to restart BLE advertisement

Figure 28. TopDesign Schematic: E-INK Display

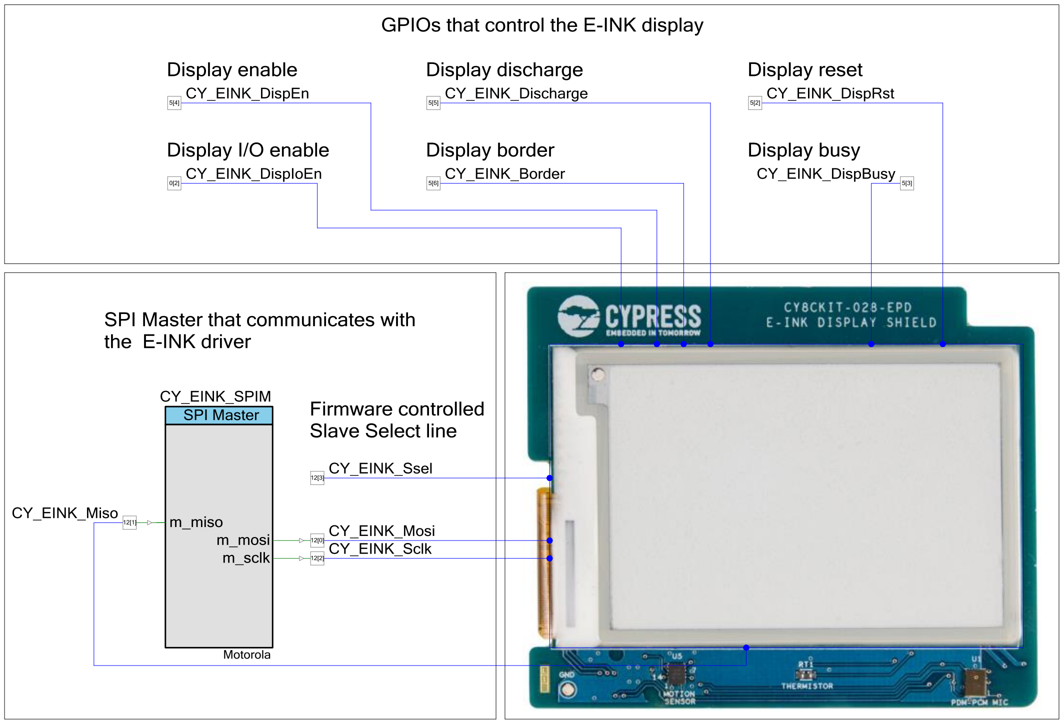
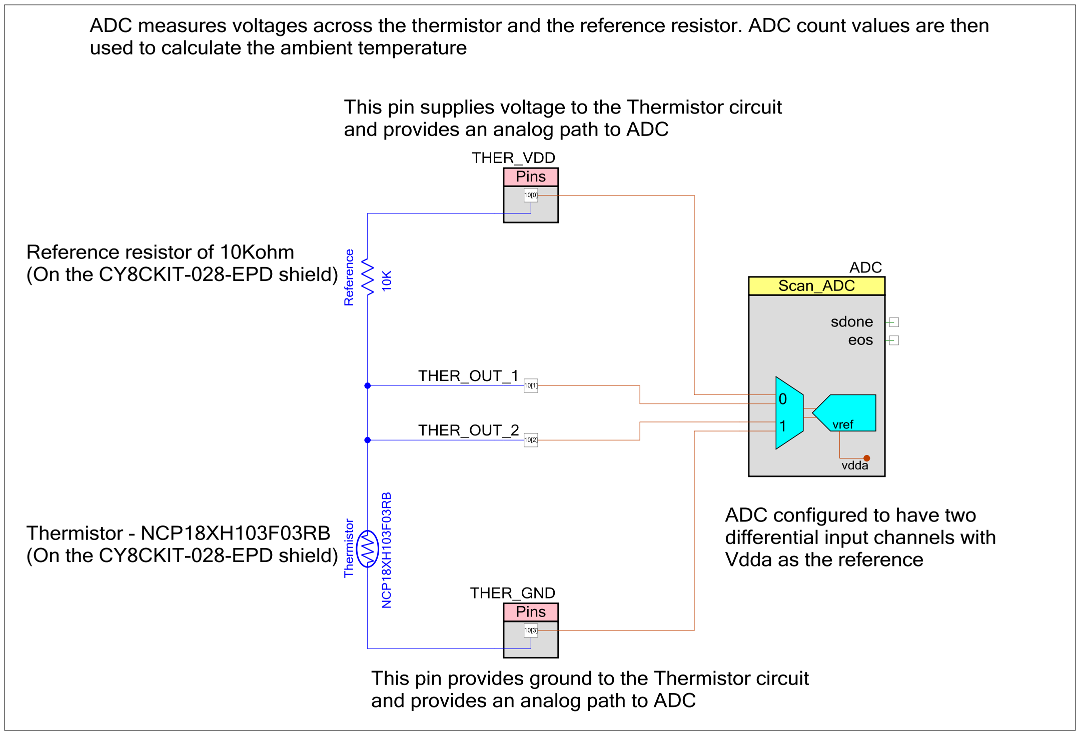


Figure 29. TopDesign Schematic: Temperature Compensation for E-INK Display


ADC measures voltages across the thermistor and the reference resistor. ADC count values are then used to calculate the ambient temperature
This pin supplies voltage to the Thermistor circuit and provides an analog path to ADC
THER_VDD
Pins
Reference resistor of 10Kohm (On the CY8CKIT-028-EPD shield)
Reference
10K
ADC
Scan_ADC
sdone
eos
THER_OUT_1
THER_OUT_2
0
1
vref
vdda
ADC configured to have two differential input channels with Vdda as the reference
Thermistor - NCP18XH103F03RB (On the CY8CKIT-028-EPD shield)
Thermistor
NCP18XH103F03RB
THER_GND
Pins
This pin provides ground to the Thermistor circuit and provides an analog path to ADC

The code example consists of the following files:

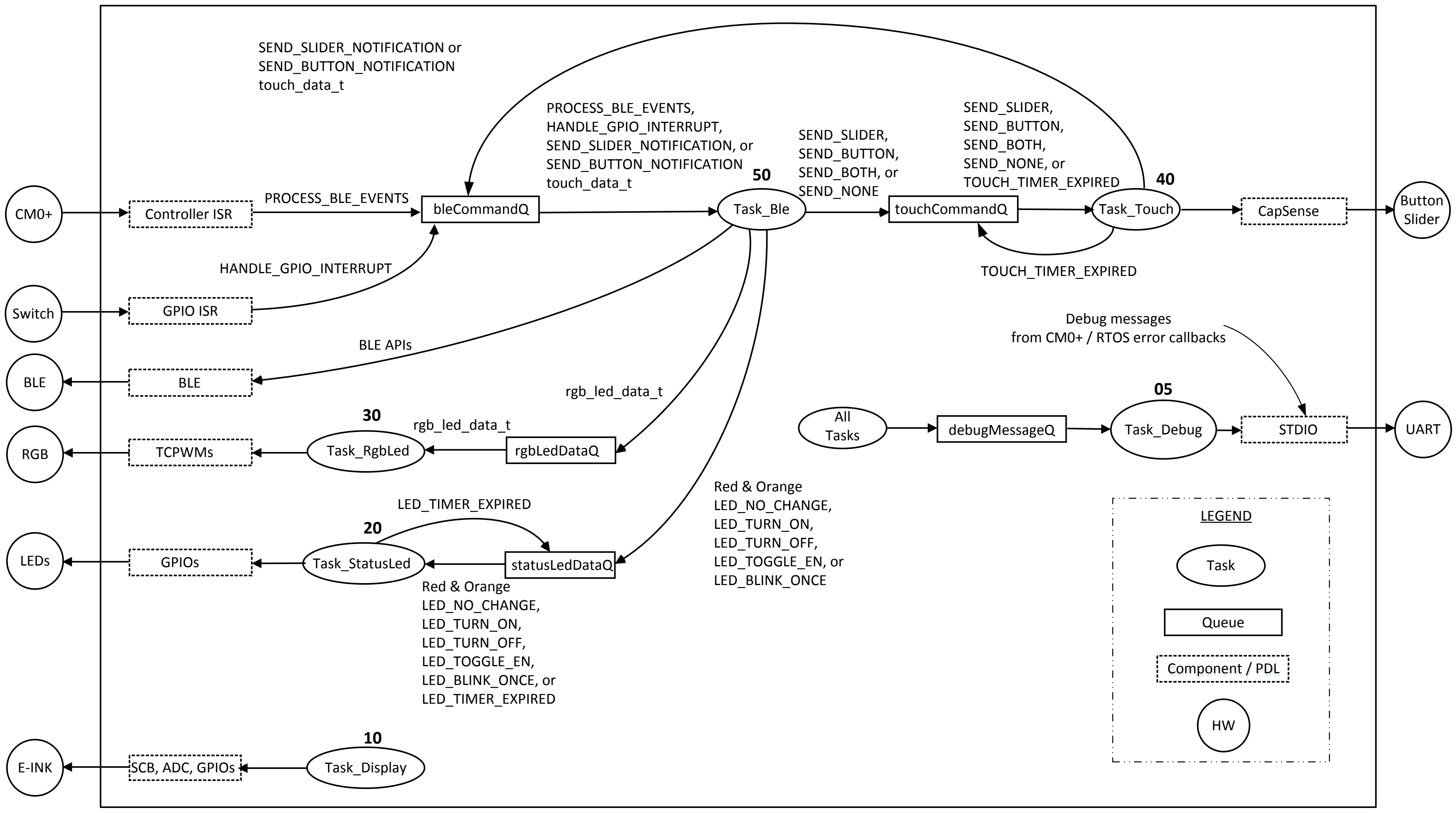
- *FreeRTOSConfig.h* contains the FreeRTOS settings and configuration. Non-default settings are explained with in-line comments.
- *main_cm4.c* contains the main function, which is the entry point and execution of the firmware application. The main function sets up user tasks and then starts the RTOS scheduler.
- main_cm0p.c contains functions that starts up the BLE controller, starts up the CM4, and continuously services BLE stack events.
- *ble_task.c/.h* contain the task and functions related to BLE communication and operation.
- *ble_custom_service_config.h* contains the macros and datatypes used for the three custom BLE services
- *touch_task.c/h* contain the task that scan CapSense sensors and process the data.
- *rgb_led.c/.h* contain the task that initialize and control the RGB LED and intensity.
- *status_led_task.c/h* – contain the task that controls status LED indications.
- *display_task.c/.h* contain the functions that initialize the E-INK display and show the instructions to use code example at startup[1].
- *uart_debug.c/h* contain the task and functions that enabled UART based message printing
- *screen_contents.c/h* contain the text and background images used by the display module.
- *temperature_eink.c/h* contain functions that measure ambient temperature for E-INK display compensation

See the corresponding header / source file for more details.

Figure 30 shows the firmware flow of this code example.

[1] 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 30. RTOS Firmware Flow



Components

Table 3. List of PSoC Creator Components

Component	Instance Name	Purpose
BLE	BLE	The BLE Component is configured for Limited Discovery with custom characteristics. These characteristics are used for notifying the BLE Central device of CapSense Slider, CapSense Buttons and read/write RGB LED control data.
CapSense	CapSense	The CapSense Component scans a 5-segment slider (CSD) and 2 buttons (CSX) with SmartSense auto-tuning.
Digital Output Pin	Pin_LED_Red Pin_LED_Orange	These GPIOs are configured as firmware controlled digital output pins that control status LEDs.
	Pin_RGB_Red Pin_RGB_Blue Pin_RGB_Green	These GPIOs are configured as digital output pins with hardware connections. These pins route PWM signals to RGB LED.
Digital Input Pin	Pin_Advertise	This pin is configured as a digital input pin that is used to generate interrupts when the user button (**SW2**) is pressed.
Global Signal Reference	GlobalSignal	The global signal component is configured to extract interrupts from **Pin_Advertise** pin.
PWM	PWM_Red PWM_Blue PWM_Green	These three TCPWMs are configured in PWM mode to control the color of the RGB LED.
UART	DEBUG_UART	UART is used to transmit debug information to a terminal (disabled by default)

Note: See the code example CE218133 – PSoC 6 MCU E-INK Display with CapSense for more details on components used by E-INK library and temperature compensation.

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

Related Documents

Application Notes	
AN210781 – Getting Started with PSoC 6 MCU with Bluetooth Low Energy (BLE) Connectivity	Describes PSoC 6 MCU with BLE Connectivity devices and how to build your first PSoC Creator project
AN85951 – PSoC 4 and PSoC 6 MCU CapSense Design Guide	Describes how to design Capacitive touch sensing applications with PSoC 6 MCU
AN215656 – PSoC 6 MCU: Dual-Core CPU system Design	Describes the dual-core CPU architecture in PSoC 6 MCU, and shows how to build a simple dual-core design
AN219434 – Importing PSoC Creator Code into an IDE for a PSoC 6 MCU Project	Describes how to import the code generated by PSoC Creator into your preferred IDE
PSoC Creator Component Datasheets	
Pins	Supports connection of hardware resources to physical pins
Timer Counter (TCPWM)	Supports fixed-function Timer/Counter implementation
Clock	Supports local clock generation
Interrupt	Supports generating interrupts from hardware signals
Bluetooth Low Energy	Supports BLE connectivity.
CapSense	Supports touch sensing
Device Documentation	
PSoC 6 MCU: PSoC 63 with BLE Datasheet	PSoC 6 MCU: PSoC 63 with BLE Architecture Technical Reference Manual
Development Kit Documentation	
CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit	
Training Videos	
PSoC 6 101: Lesson 1-4 FreeRTOS	

Document History

Document Title: CE220331 – PSoC 6 MCU with BLE Connectivity: BLE with User Interface (RTOS)

Document Number: 002-20331

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**		NIDH		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	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Internet of Things	cypress.com/iot
Memory	cypress.com/memory
Microcontrollers	cypress.com/mcu
PSoC	cypress.com/psoc
Power Management ICs	cypress.com/pmic
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless Connectivity	cypress.com/wireless

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

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