PSoC 4 BLE – Custom Profile

Objective

This example demonstrates the use of BLE Component to design a Custom profile by implementing an RGB LED controller through BLE. It also demonstrates how to combine CapSense and BLE in a system, by designing a slider application.

Overview

This example implements a custom BLE profile with a RGB LED custom service to control the RGB LED color and intensity, and a CapSense Slider custom service to send the slider data over the BLE interface. The target PSoC 4 BLE device implements the CapSense slider and controls the RGB LED color and intensity using the PrISM component.

Requirements

Design Tool: PSoC Creator 4.0 Update 1, CySmart 1.2

Programming Language: C (GCC 4.9 – included with PSoC Creator)

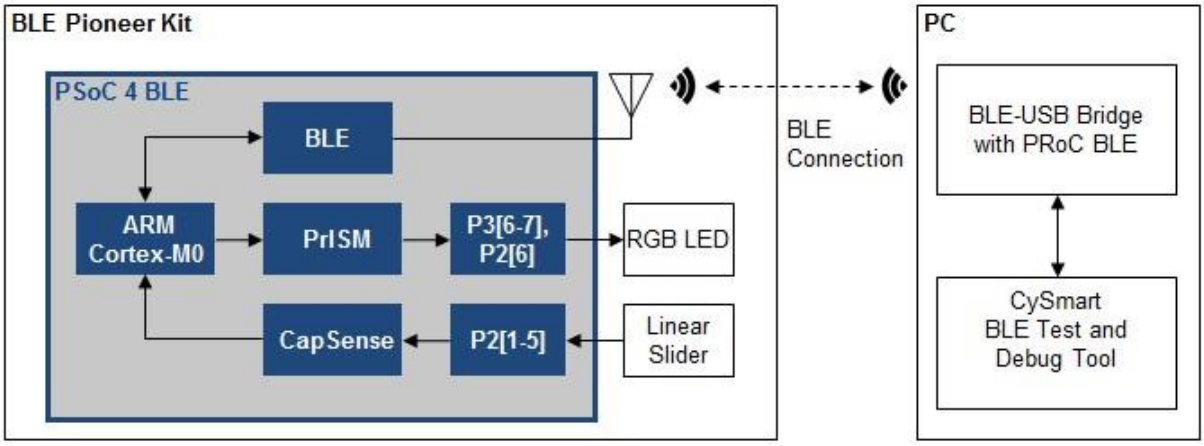
Associated Devices: All PSoC 4 BLE devices

Required Hardware: CY8CKIT-042-BLE Bluetooth® Low Energy (BLE) Pioneer Kit

Hardware Setup

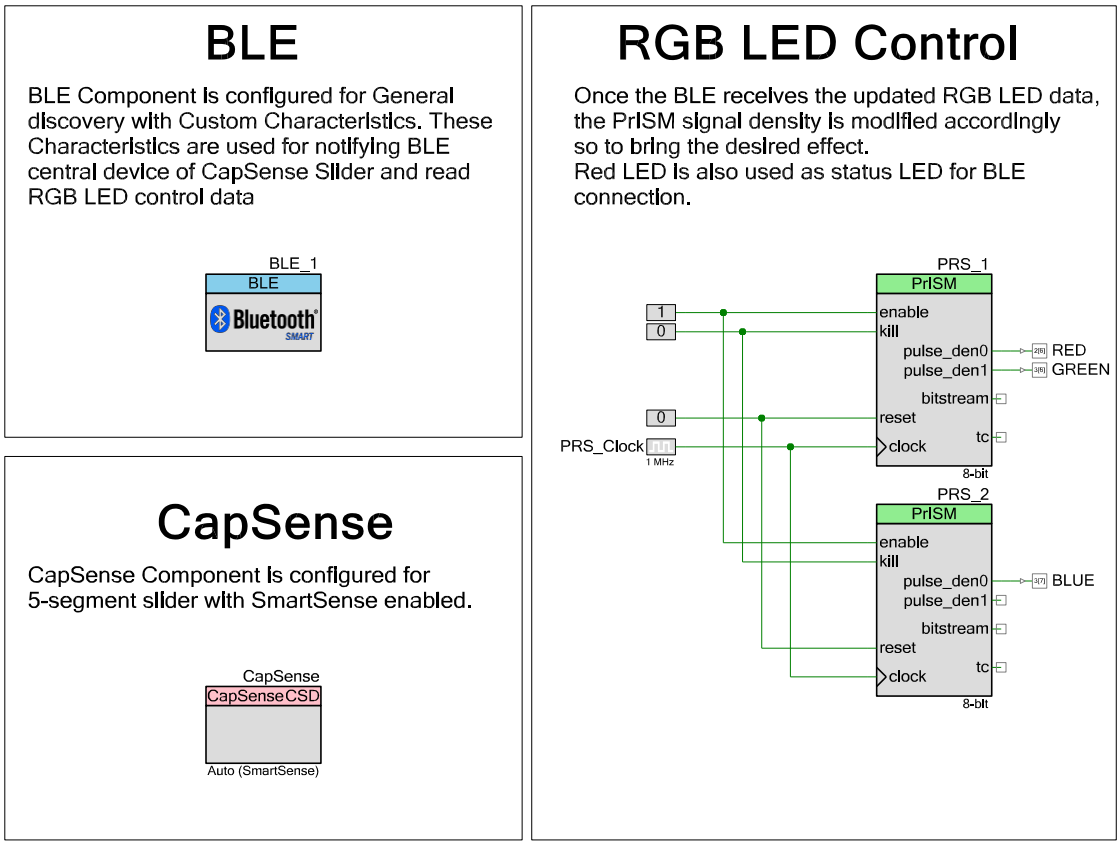
The BLE Pioneer Kit has all of the necessary hardware required for this example. Figure 1 shows the hardware setup for this application. For testing, hardware setup uses the BLE-USB Bridge, but user can also use CySmart Android/iOS app.

Figure 1: Block Diagram



PSoC Creator Schematic

Figure 2. PSoC Creator Schematic



Firmware Flow

The firmware consists of these high-level blocks:

1. **CapSense slider**: The CapSense slider on the kit is scanned periodically. If the finger position is different from the previous scan, a notification packet is sent over BLE. The scan happens only when the notifications for this CapSense Slider Characteristic are enabled by the GATT Client.
2. **RGB LED**: The PrISM blocks used to drive the RGB LED are configured based on the Attribute values written by the GATT Client. The value can be in a range of 0 and 255 for each LED individually. This value is converted into a percentage of intensity for that LED and the corresponding PrISM block is configured. The overall LED brightness is a separate input (one of the four bytes in that characteristic) which controls the final intensity of all LEDs.
3. **BLE**: The events generated by the BLE Stack are handled to keep track of advertisement, connection, and disconnection states. The Attribute Write request event is handled by first identifying the Attribute which was written to by the GATT Client.

If the Attribute is the CapSense Slider Characteristic's Client Characteristic Configuration Descriptor (CCCD), then the slider notifications are correspondingly enabled (CCCD = 1) or disabled (CCCD = 0). On the other hand, if the Attribute is the RGB LED Characteristic, then the LED is controlled accordingly.

4. **Application Layer**: A top-level application layer is written for the firmware.

The firmware flow is shown in Figure 3. Table 1 lists the files present in the firmware. This table describes the different functions defined in these files and their usage.

Figure 3: Firmware Firmware

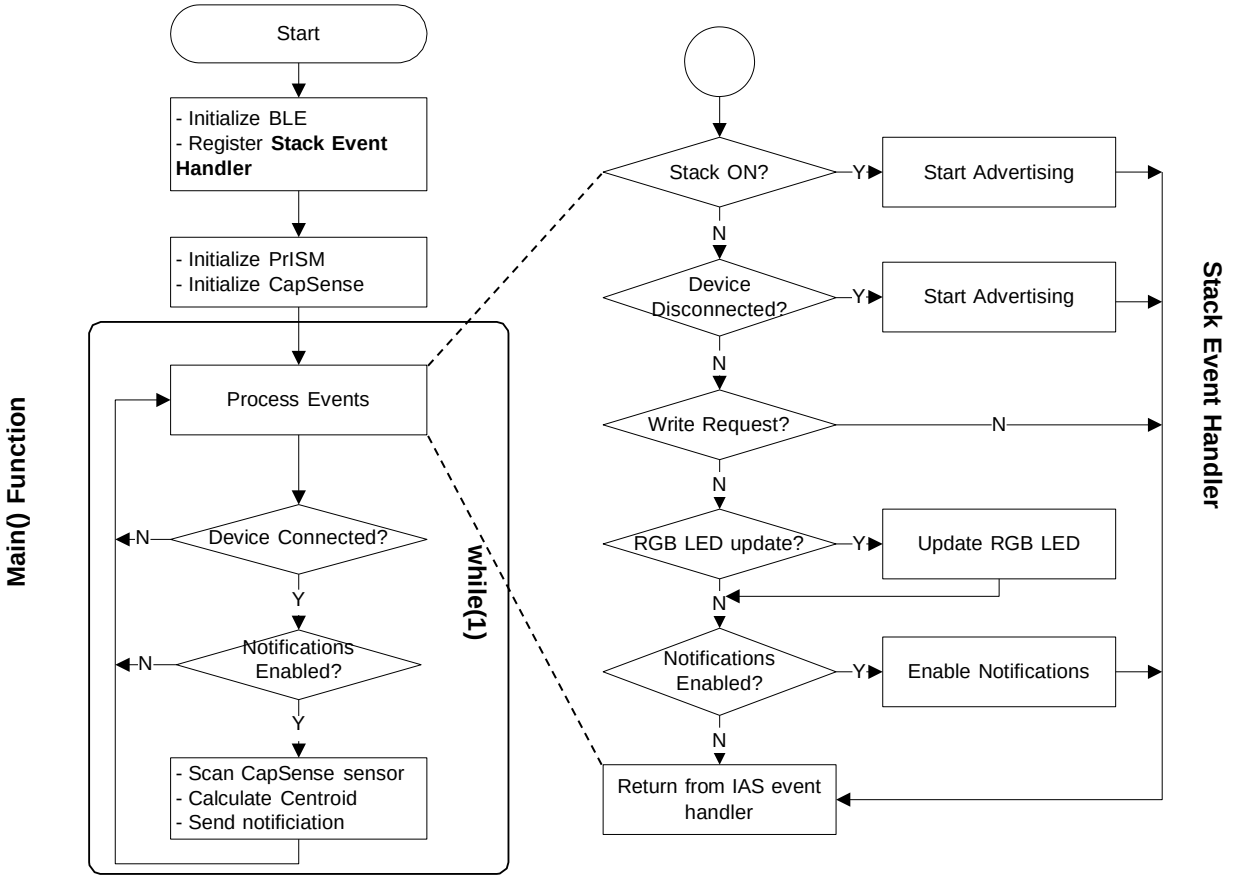


Table 1: Main Files Present in this project

File name	Details
main.c	This is the top level application file. It initializes the system and runs the main loop. It also handles the CapSense slider functionality. This file has three functions: ***main()*** – The main function for the application ***InitializeSystem()*** – Initializes all the blocks of the system

	HandleCapSenseSlider() – Scans the CapSense slider and finds the finger position on the slider. When the finger position changes relative to the previous scan, it sends the new position as a notification over BLE. The notifications are sent by calling the SendCapSenseNotification() function.
BLEApplications.c	This file handles the BLE specific functionality of the project. It handles the BLE events and notifications. The file has these functions: ***CustomEventHandler()***: Handles the events for BLE advertisement, connection, and disconnection. Also services the write requests to the Custom Characteristics and Descriptors. This function is a callback from the BLE Stack for all events. ***SendCapSenseNotification():*** Creates a CapSense Slider Characteristic notification packet and sends it. This function is called by HandleCapSenseSlider() function in main.c. ***UpdateRGBled():*** Configures the PrISM blocks to drive the RGB LED as per the latest data. Also updates the RGB LED Characteristic in the database with the latest data for a future read by the GATT Client. ***UpdateNotificationCCCD():*** Updates the CapSense Slider Characteristic's CCCD value from the GATT Client. The value written by the GATT Client goes to the Profile layer of the Stack, which is error-checked before being written to the GATT DB. For Standard Profiles, this is handled by the BLE Component. For Custom Profiles, the application firmware must do this.

Build and Program

1. **Build** your project to generate the hex file, and **Program** it to your kit.

Testing with CySmart BLE Test and Debug Tool

1. Open **CySmart 1.0** and **Connect** it to the **BLE-USB Bridge**.
2. **Start Scan** and **Connect** to your GATT Server device.
3. **Discover all Attributes** and then scroll down the Attribute list to the **RGB LED Characteristic** (it has the **UUID 0xCBB1**). See Figure 4.

Figure 4. CySmart - RGB LED Characteristic

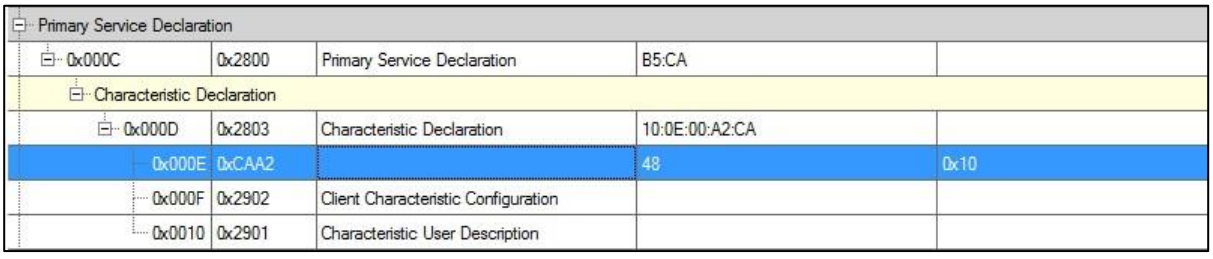
Primary Service Declaration				
0x0011	0x2800	Primary Service Declaration	BB:CB	
Characteristic Declaration				
0x0012	0x2803	Characteristic Declaration	0A:13:00:B1:CB	
0x0013	0xCBB1			0x0A
0x0014	0x2901	Characteristic User Description		

4. **Write** a 4 byte value to this Characteristic on the right and notice the corresponding color and intensity of the RGB LED on the kit. Byte 0 corresponds to the Red color, Byte 1 corresponds to the Green color, Byte 2 corresponds to the Blue color, and Byte 3 corresponds to the intensity. For example, writing 00:00:FF:FF to this Characteristic turns on the Blue LED with full intensity. See Figure 5.

Figure 5: CySmart - Write Attribute

5. Now locate the **CapSense Slider Characteristic (UUID = 0xCAA2)** and enable notifications for it, either by clicking **Enable All Notifications** or by writing **1** to its CCCD descriptor (UUID=0x2902).
6. Move your finger over the slider on the kit and observe that the value of the Characteristic changes in the CySmart tool, while the tool's log shows notification packets being received. See Figure 6.

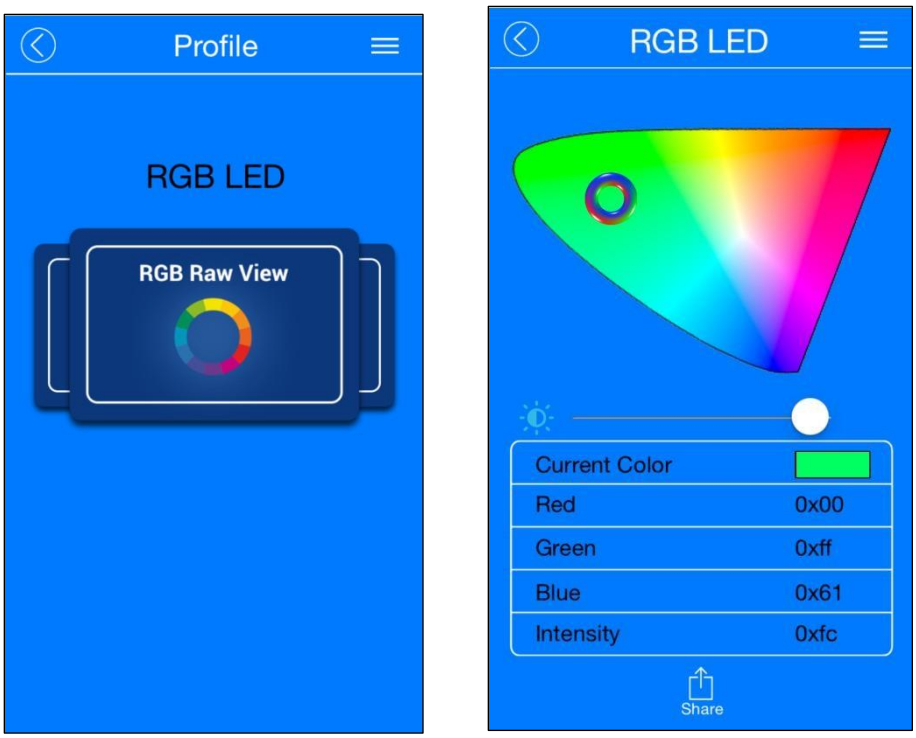
Figure 6: CySmart - Slider Notifications



Testing with CySmart Mobile App

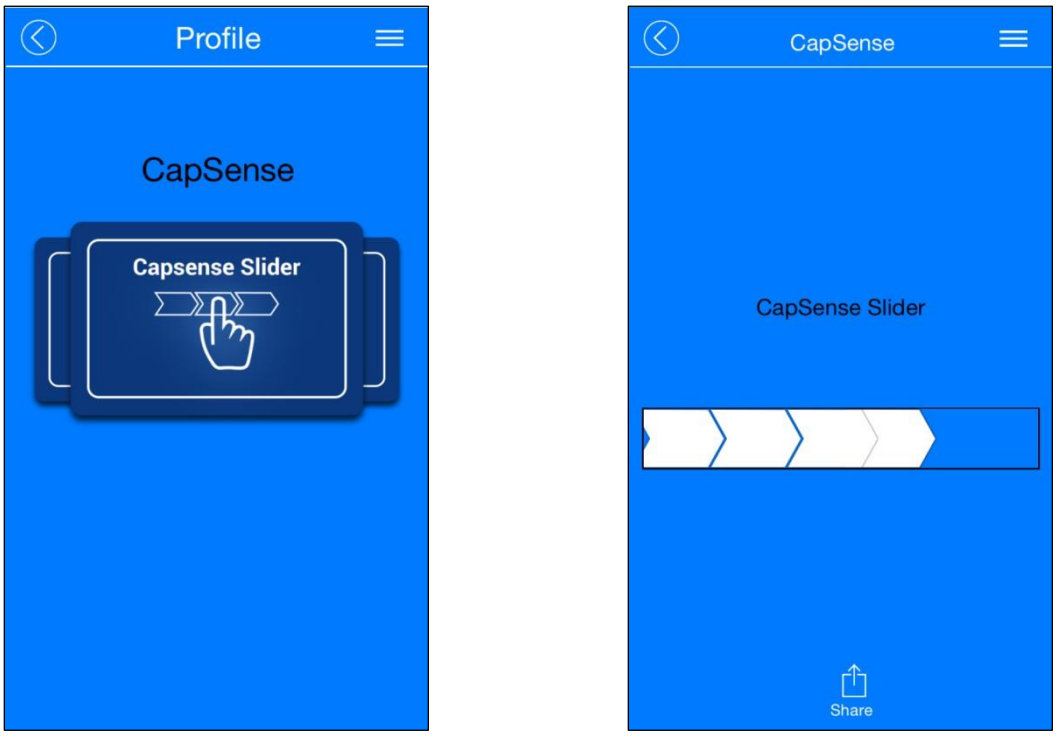
1. Open the **CySmart Mobile App** on your phone. If you do not have Bluetooth switched on already, the app will ask you to do it.
2. Connect to your GATT Server device on the app. Once connected, the app shows you all the Services exposed by the GATT Server. It automatically detects the Custom Services for RGB LED and CapSense Slider and lists them respectively.
3. Select the RGB LED Service. You will see that a color gamut is available. Tap anywhere on the gamut to see the corresponding color on the RGB LED on the kit. See Figure 7.

Figure 7: CySmart iOS Mobile App – RGB LED Control



4. Move the slider position on the app page to change the brightness level of the LED.
5. Now go back one page in the app and select the CapSense Slider Service.
6. Once on the CapSense Slider page, move your finger on the slider on the kit. You will see a corresponding slider update on the app page. See Figure 8.

Figure 8: CySmart iOS Mobile App – CapSense Slider



Related Documents

Table 2 lists all relevant application notes, code examples, knowledge base articles, device datasheets, and Component / user module datasheets.

Table 2. Related Documents

Document	Title	Comment
AN91267	Getting Started with PSoC 4 BLE	Provides an introduction to PSoC 4 BLE device that integrates a Bluetooth Low Energy radio system along with programmable analog and digital resources.
AN91445	Antenna Design Guide	Provides guidelines on how to design an antenna for BLE applications.