PSoC 4 BLE Lab #4: CapSense Design with BLE Connectivity

## ECEG 721-61

Embedded Systems

November 10, 2016

Prepared for: Dr. Hussein

Prepared By: Manisha Agrawal

Michael Surovich

# Introduction

This lab teaches students to create 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. A Custom Profile will be used in this lab. A custom profile can contain standard BLE service as well as custom services that the user can define. A Custom service is user-defined where the user can define their own Characteristics and Descriptors. A Custom Services for the RGB LED and the CapSense Slider are created where the UUIDs are defined. The objectives of this lab are to adjust the RGB LED color and intensity using the PRiSM Component, implement a custom BLE Profile with a custom Service to send RGB LED color and intensity over BLE, implement a Custom Service to send CapSense slider data over BLE, and use the CySmart tool or mobile app to validate the operation.

# Results

A template project is provided by Dr. Hussein and is used to start the project. That template file is opened and used throughout the lab. The schematic editor is opened and the provided schematic is opened. A “Bluetooth Low Energy” Component is placed on the sheet. The component is located under the “Communications” category. The setting for the BLE must be configured. Under the General Tab, the profile is set to Custom, the Profile Role is set to Server (GATT Server), and the GAP Role is set to Peripheral. These settings are shown in Figure 1.

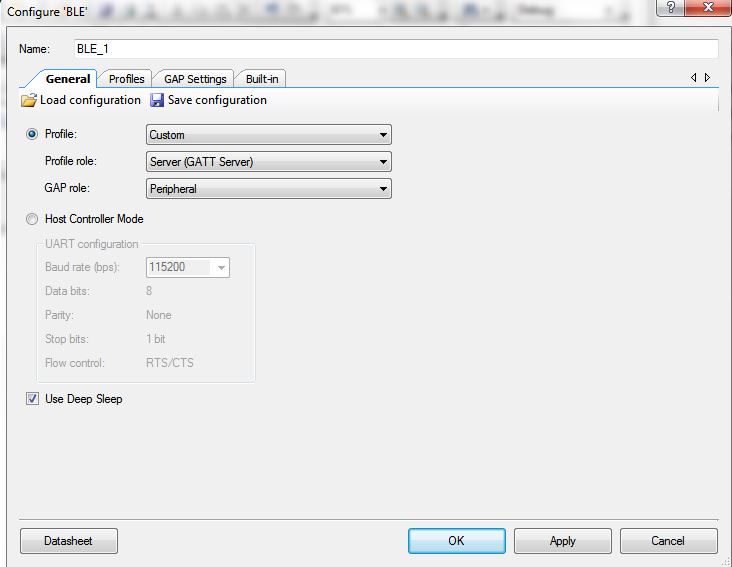


Figure 1 - Configuration of BLE (General Tab)

Under the Profiles Tab, the CapSense Slider and an RGB LED Services are configured. The Custom Service already listed will be used for the CapSense Slider. The Custom Service is renamed to CapSense Service. The UUID length is set to 16-bit and the value is 0xCAB5, which is the value defined by Cypress for this Service. The Custom Characteristic is renamed to CapSense Slider Characteristic and the UUID length is set to 16-bit and the value is set to 0xCAA2. Notify is enabled in the Properties in the Characteristic. The Custom Descriptor is deleted. A Characteristic User Description Descriptor is added and the value is set to CapSense Slider. The Service, Characteristic, and Descriptor settings are shown in Figure 2, Figure 3, and Figure 4, respectively.

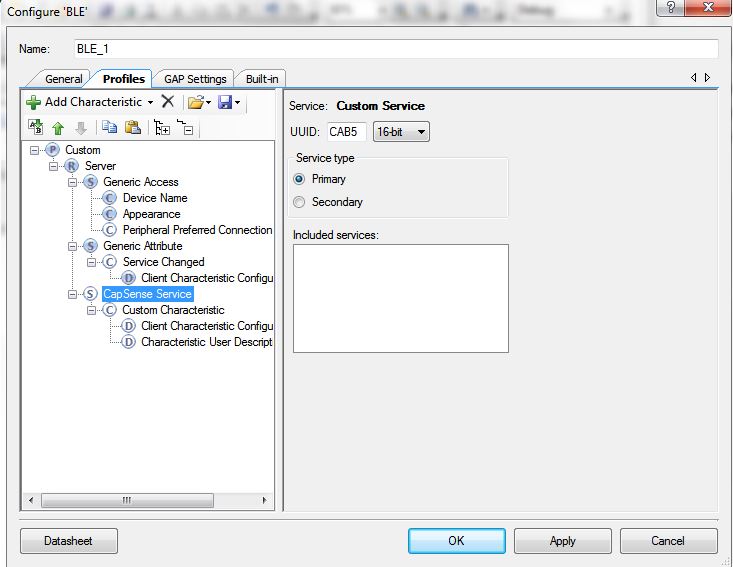


Figure 2 - CapSense Service Configuration

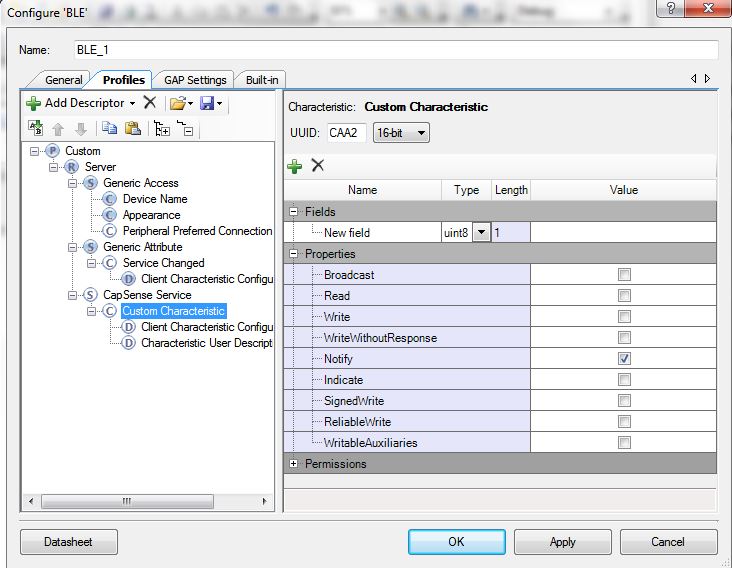


Figure 3 - CapSense Characteristic Configuration

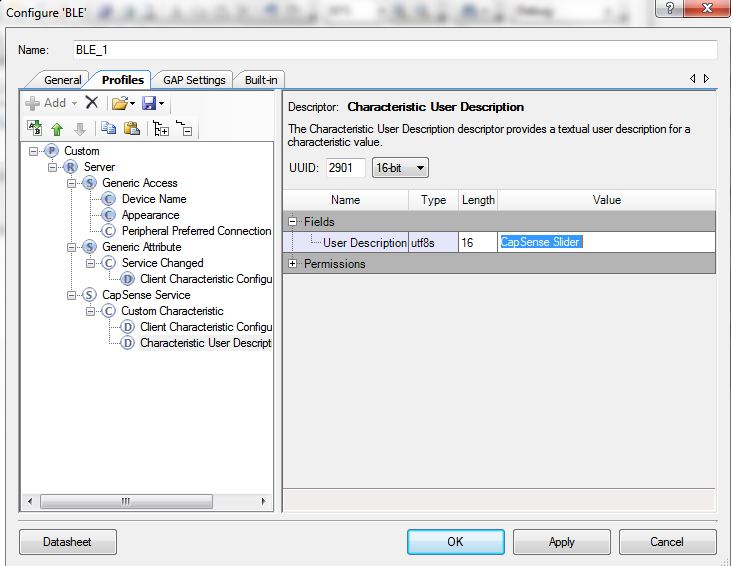


Figure 4 - CapSense Descriptor Configuration

A new custom service is added for the RGB LED. The service is renamed to RGB LED Service. The UUID length is 16-bit and the value is set to 0cCBBB. The Custom Characteristic has a UUID length of 16-bit and the value is set to 0xCBB1. The Type of New Field under the Fields column is set to uint8 array and the Length to 4. The Read and Write is enabled in the properties. The Custom Descriptor is deleted and the Characteristic User Description Descriptor. The value is given as RGB LED Control. The Service, Characteristic, and Descriptor settings are shown in Figure 5, Figure 6, and Figure 7, respectively.

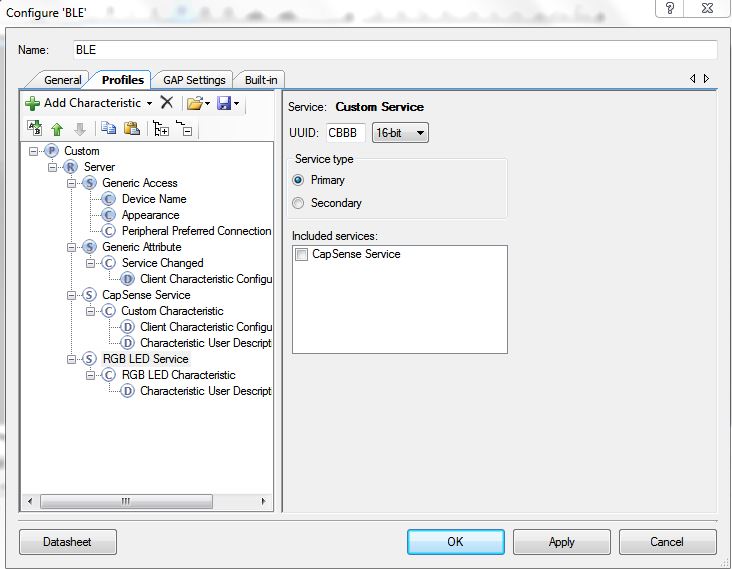


Figure 5 - RGB LED Service Configuration

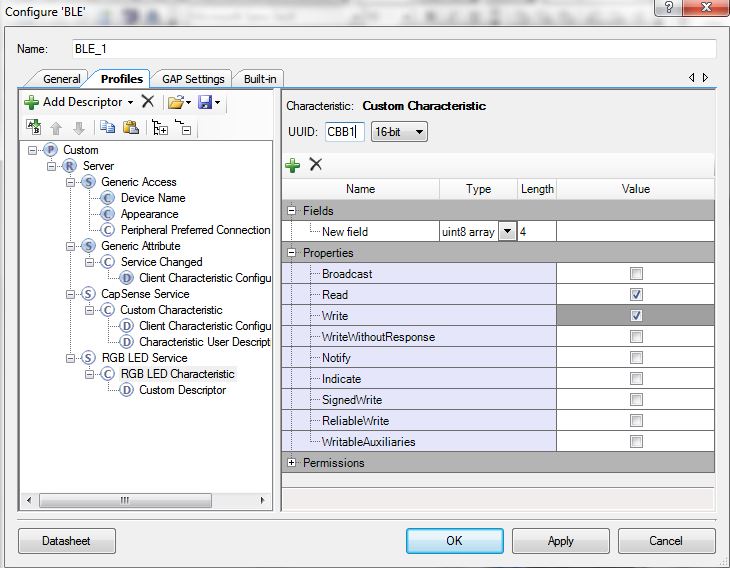


Figure 6 - RGB LED Characteristic Configuration

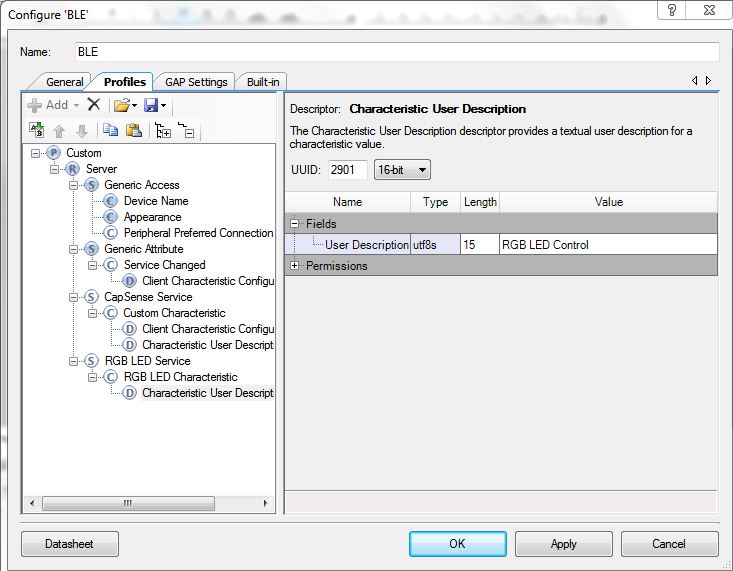


Figure 7 - RGB LED Descriptor Configuration

Under the GAP Settings Tab, the Device Address, Device Name, and Appearance are set as shown in Figure 8. The Advertising Settings are left as the default settings. The Advertisement Packet settings are changed to enable the Local Name, as shown in Figure 9. The Scan Response Packet is left as the default settings. The Security Settings are changed as shown in Figure 10.

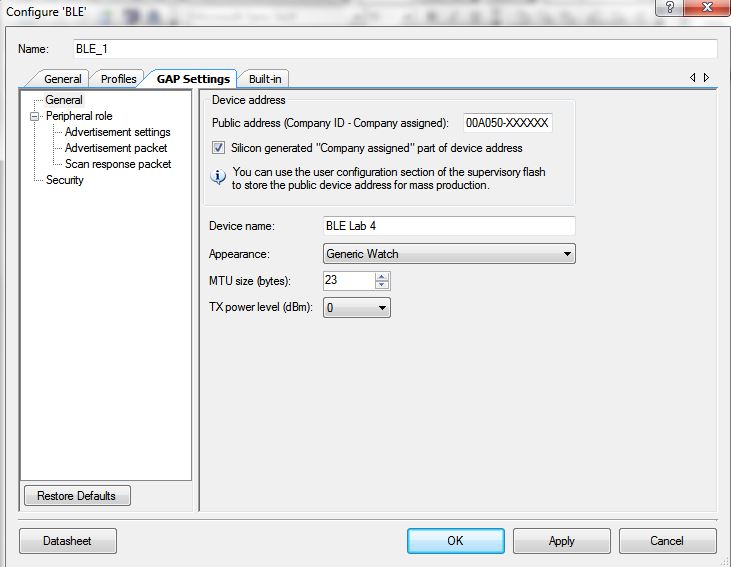


Figure 8 - GAP Settings (General)

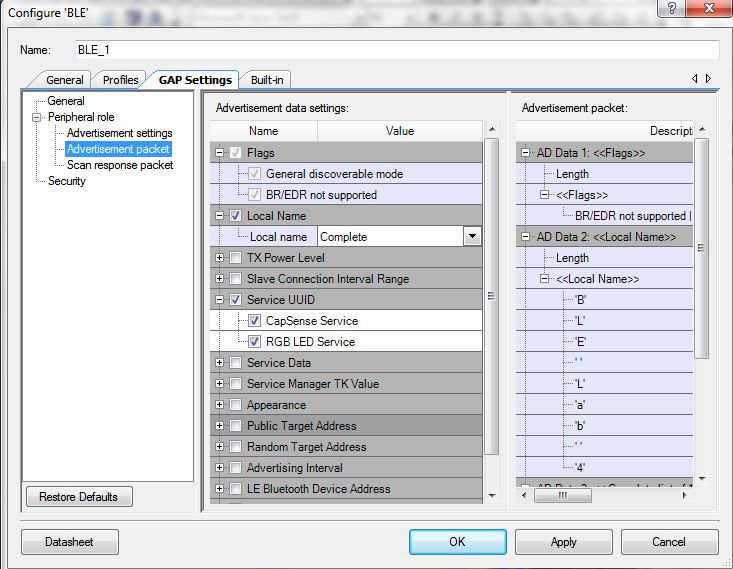


Figure 9 - GAP Settings (Advertisement Packet)

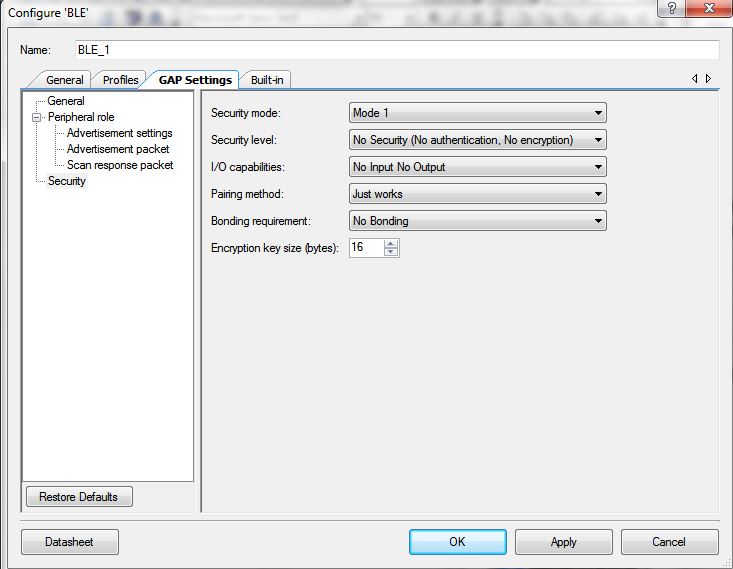


Figure 10 - GAP Settings (Security)

The BLE Configurations are complete and the window is closed. The CapSense CSD Component is added to the schematic. The CapSense CSD Component’s name is changed to CapSense. On the General tab, the default settings are left as the defaults. On the Widgets Configuration tab, a linear slider is added by clicking Linear sliders and then the "add linear slider" button. The slider’s settings are left as the default and the configuration can be closed. The CapSense Settings are shown in Figure 11.

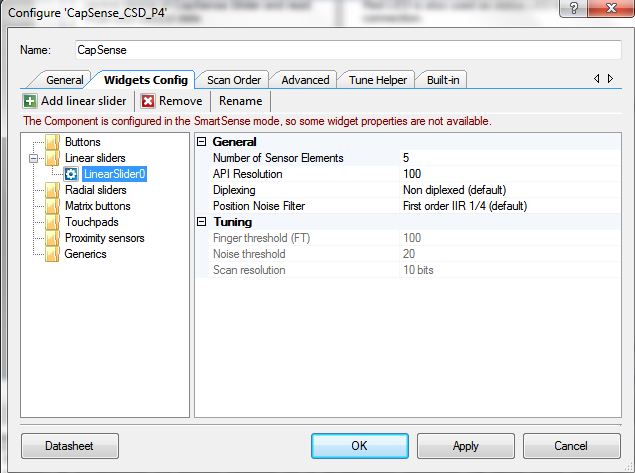


Figure 11 - CapSense Component Configuration

All of the components have been added and the parameters for each component are set to the required settings. The complete schematic is shown in Figure 12.

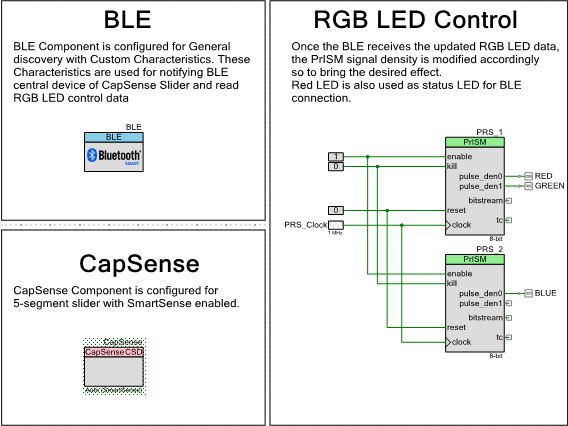


Figure 12 - Completed Schematic

The DWR Settings must be conjured. On the Pins tab, the segments of the linear slider are set to pins P2[1], P2[2], P2[3], P2[4], and P2[5] in increasing order. The CMOD capacitor is set to P4[0]. The three LEDs are already set to the correct device pins.

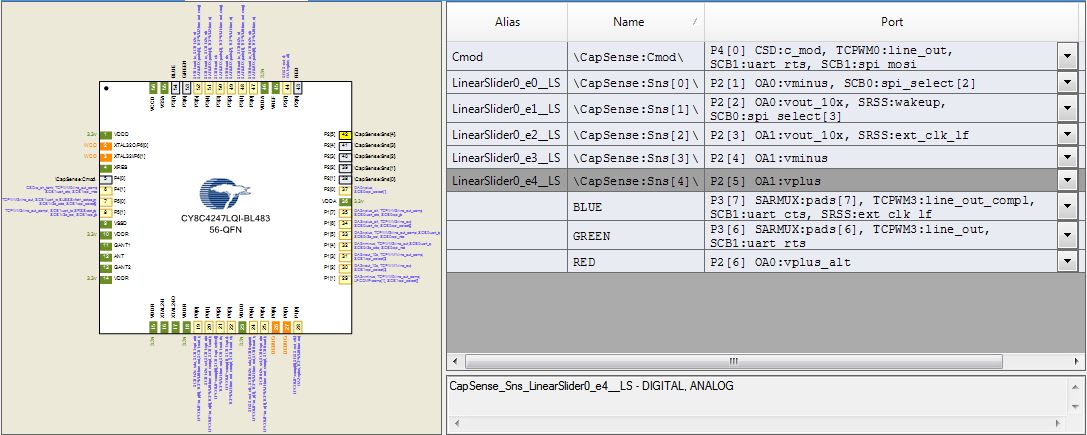


Figure 13 - Design Wide Resources and Pin Assignments

The Project is now built and the component source files are generated. All of the firmware is supplied for this lab and is thoroughly explained in the document supplied to the students. The project is programmed to the board. The board is now programmed.

The system is tested using the CySmart Mobile App. The app shows the RGB LED and CapSense service. The RGB LED Service is opened and the color gamut is available. Some test color selections and pictures of the board are shown in Figure 14. The CapSense test is shown in Figure 15. That completes the first potion of the lab.

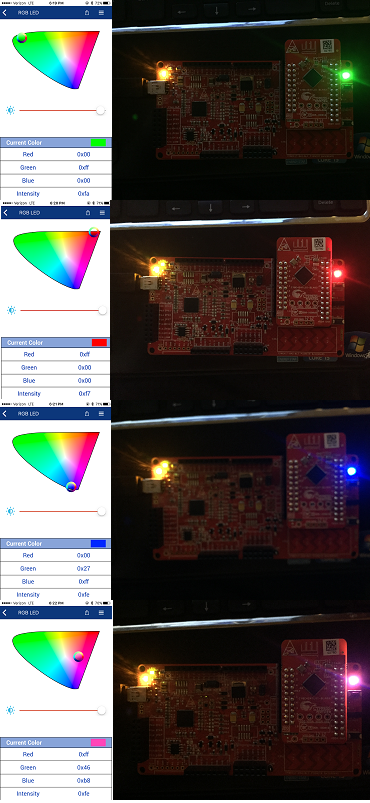


Figure 14 - RGB Test Color Selections and Results

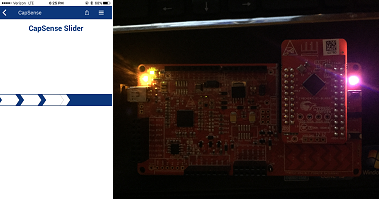


Figure 15 - CapSense Test and Results

# Additional Exercises

## Replace CapSense Slider with CapSense Proximity

The CapSense Slider can be replaced with the CapSense Proximity sensor. In order to perform this task, some changes had to be made to the original lab settings.

The first change made was to change the UUID of the CapSense Service to CAA1. The result is shown in Figure 16. The proximity sensor was enabled using the API Function “CapSense\_EnableWidget(CapSense\_PROXIMITYSENSOR0\_PROX)”. This new function is shown in Figure 17. The next change made is to replace the “CapSense\_GetCentroidPos(CapSense\_LINEARSLIDER0\_LS)” function with “CapSense\_GetDiffCountData(CapSense\_PROXIMITYSENSOR0\_PROX)” to extract the proximity sensor value. The result is shown in Figure 18. Finally, slight modifications are made to the code to always send the CapSense Proximity sensor Notification data.

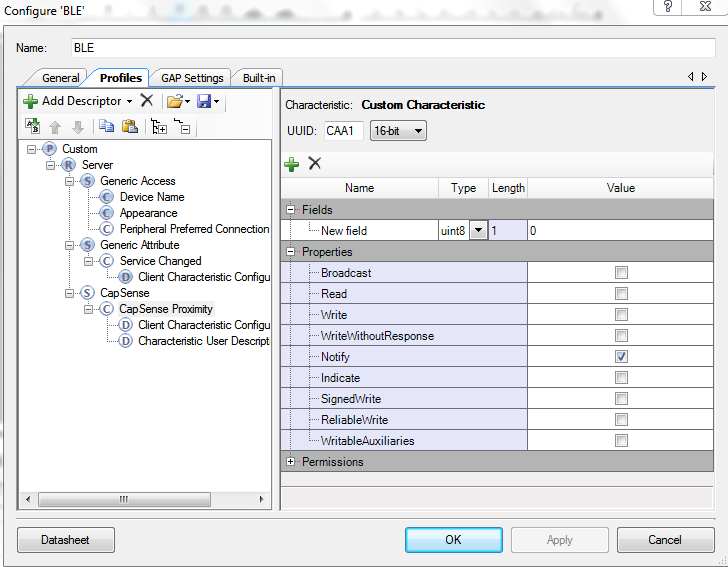


Figure 16 - CapSense Proximity Sensor UUID

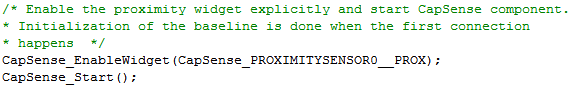


Figure 17 - Code to Enable Proximity Sensor



Figure 18 - Extract Proximity Sensor Value

In order to test the system, a wire is inserted into the P2[0] terminal. When the screen application is opened on the phone a blank bar graph is shown on the screen. As a person moves their finger closer to the tip of the wire, the line rises on the bar graph. Screen shots of these two events are shown in Figure 19 and Figure 20, respectively.

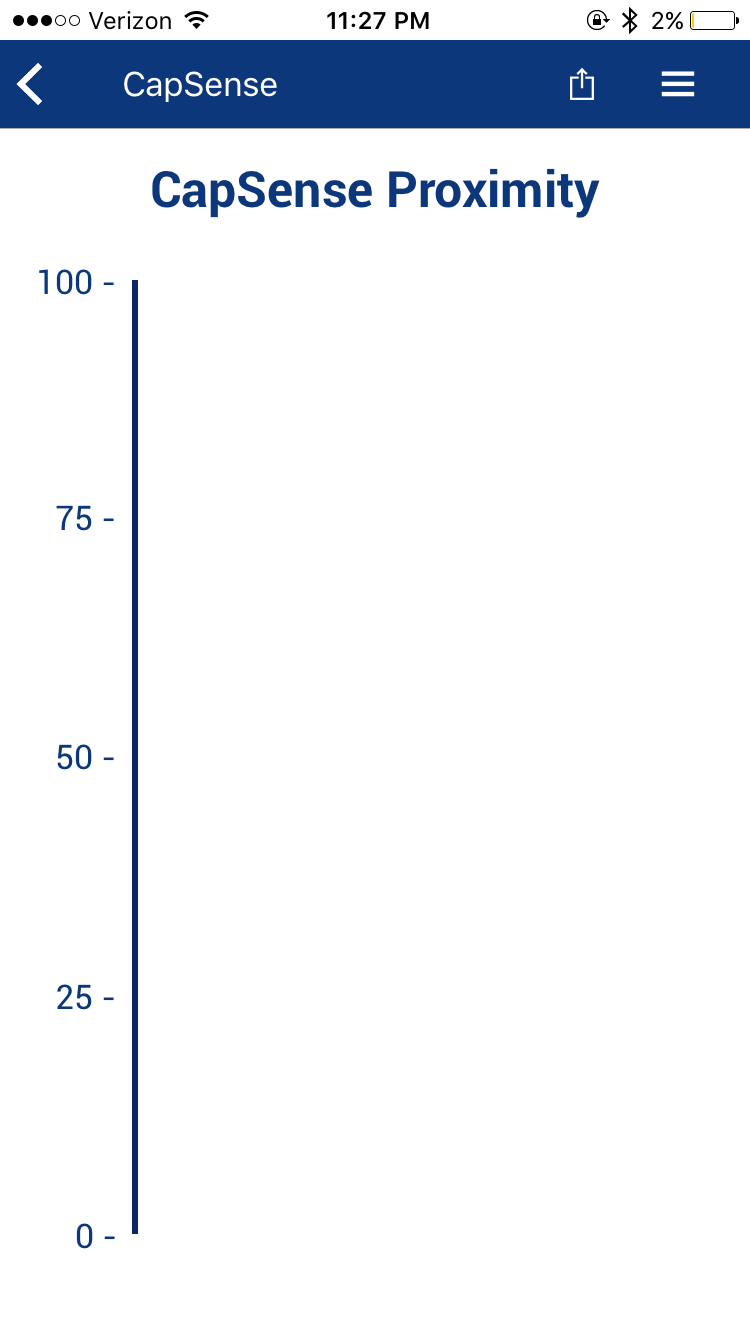


Figure 19 - CapSense at Baseline Value

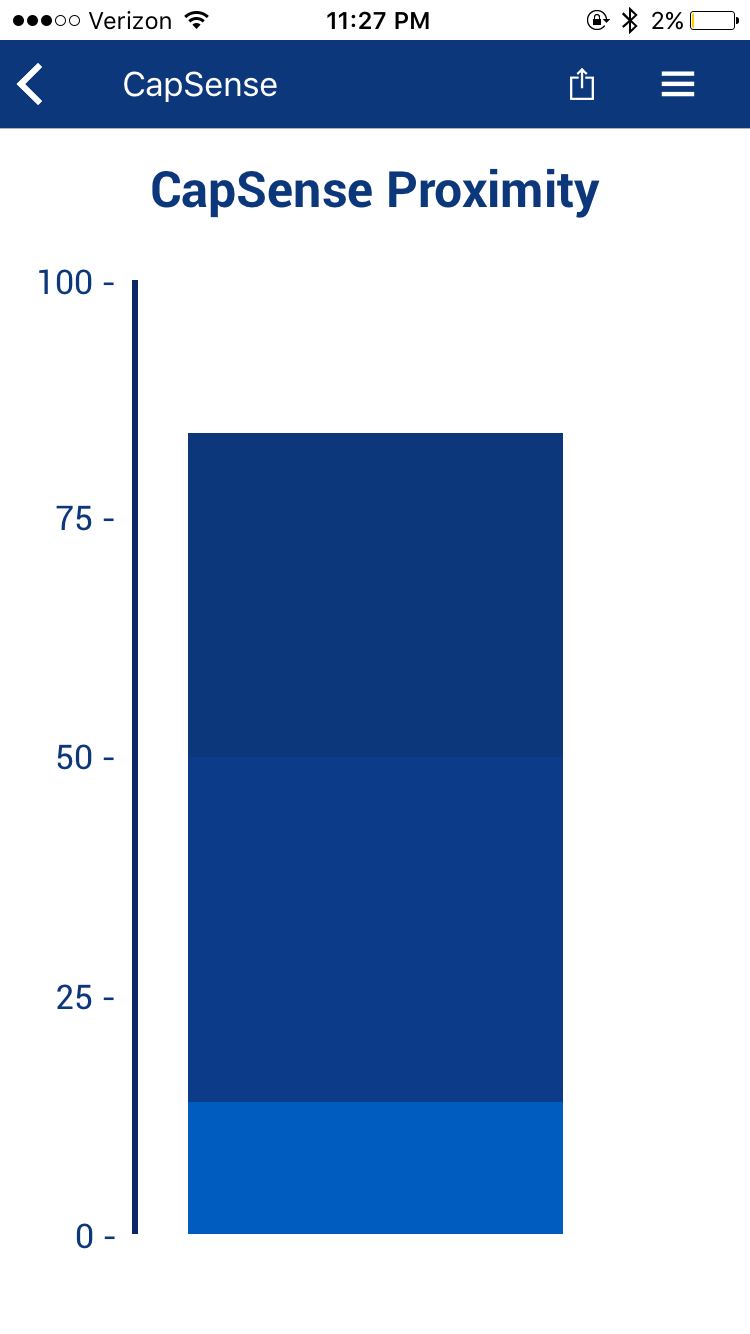


Figure 20 - CapSense with Human Interaction

## Implement Low-Power Operation

The final portion of the lab is to implement low power operation for this lab. This portion is done following the same procedure as the PSoC BLE Lab 3. A switch to wake-up the device from hibernate mode is added to the design. Since, the PrISM Component is not active during the Deep-Sleep mode, the code for putting the device in the Deep-Sleep mode is commented out.

# Conclusion

This lab taught students to create a Custom Profile by implementing an RGB LED controller through BLE. It also demonstrated how to combine CapSense and BLE in a system, by designing a slider application. A Custom Profile is used in this lab. The objectives of this lab were to adjust the RGB LED color and intensity using the PRiSM Component, implement a custom BLE Profile with a custom Service to send RGB LED color and intensity over BLE, implement a Custom Service to send CapSense slider data over BLE, and use the CySmart tool or mobile app to validate the operation.

# GitHub Links

Lab 4: <https://github.com/msurovich22/Lab-4>

Lab 4 Additional Exercises: <https://github.com/manishahg/PSOC_CapSenseProximity>