

PSoC 4 BLE Lab4: CapSense Design with BLE Connectivity

Group: Gloria Bauman and Kervins Nicolas

Due: October 28th, 2016

Link to Github: <https://github.com/kervins/BLE-LAB-4>

Introduction:

Creating a CapSense Design and RGB LED to work with a BLE Connection. Student must use BLE Pioneer Kit to build and program. Using the Psoc software, individual must be able to configure code to work in order for CapSense and RGB LED to be implemented without any complications. Will understand some aspects of how CapSense Service can be used and ways that RGB LED can change due to slight modifications.

Description

This lab teaches you how 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.

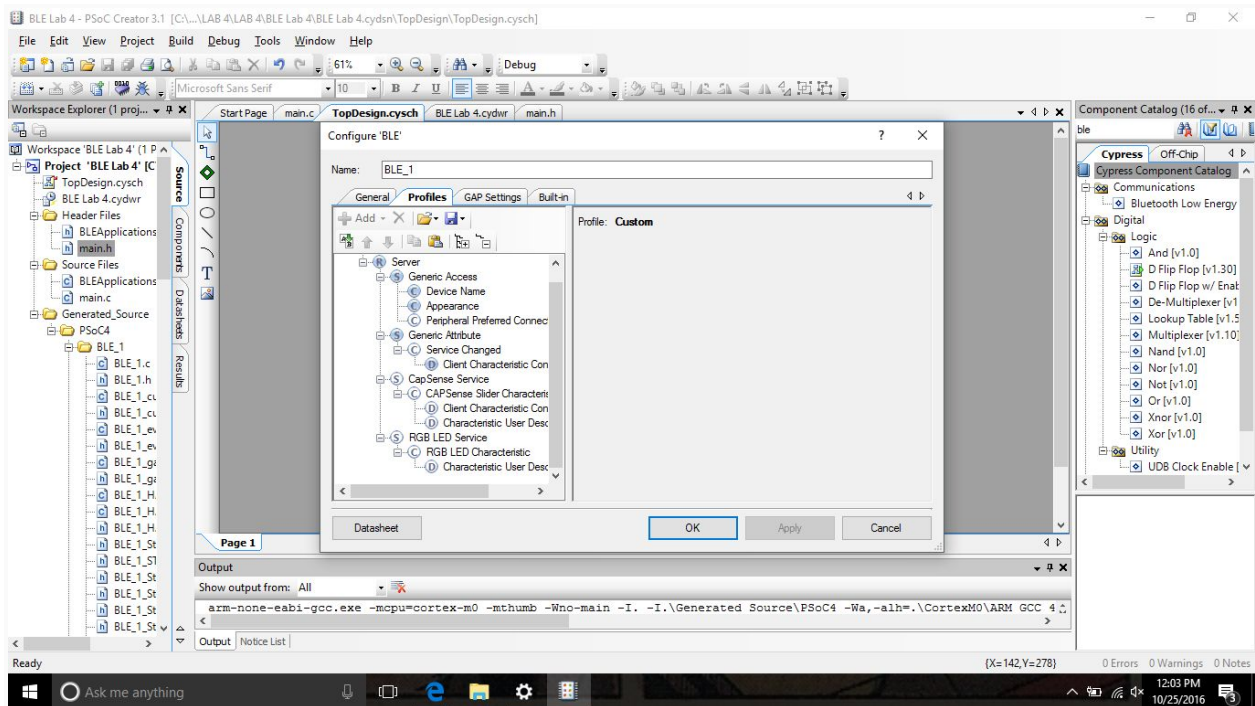
Objectives

1. Adjust RGB LED color and intensity using the PRiSM Component
2. Implement a custom BLE Profile with a custom Service to send RGB LED color and intensity over BLE
3. Implement a Custom Service to send CapSense slider data over BLE
4. Use the CySmart tool or mobile app to validate the operation

Process

- 1) Under the "TopDesign.cysch" from the "Workspace Explorer" drag and place a "Bluetooth Low Energy" Component from the "Component Catalog" under the communications category.
- 2) Configure the component to the appropriate parameters and settings.
 - Under the "General" Tab
 - Profile: "Custom"
 - Profile Role: "Server (GATT Server)"
 - GAP role: "Peripheral"
 - Under "Profiles Tab"
 - Server Tab:
 - "CapSense Service"
 - UUID: "CAB5 16-bit"
 - Service type: "Primary"
 - "Capsense Slider Characteristics"
 - Check off Notify
 - Delete Character User Description by right-clicking on it
 - "Characteristic User Description"
 - User Description: CapSense sliders
 - Add a new Custom Service to the "Profiles" page by right-clicking on the "Server" and then selecting "Add Service → Custom Service"
 - Rename the "Custom Service" to "RGB LED Service"
 - Set UUID (16-bit) to 0xCBBB
 - Rename "Custom Characteristic" to "RGB LED Characteristics"
 - Set UUID (16-bit) to 0xCBB1
 - New Field: uint8 array
 - Length: 4
 - Read: check
 - Write: Check
 - Delete the "Custom Descriptor"
 - Add the Characteristic User Description Descriptor
 - Value: "RGB LED Control"
 - Under "GAP Settings" Tab
 - "General"
 - Check "Silicon generated "Company assigned" part of the device address"
 - Device name: "BLE Lab 4"
 - Appearance: "Generic Watch"
 - "Peripheral role"- Advertisement Package
 - Local Name: enable
 - Service UUID

- CapSense Services: enable
- RGB LED Service: enable
- “Security”
 - Security mode: Select “Mode 1” security
 - Security level: Select “No Security (No Authentication, No encryption)”
 - I/O Capabilities: “No input No Output”
 - Bonding requirement: “No Bonding”

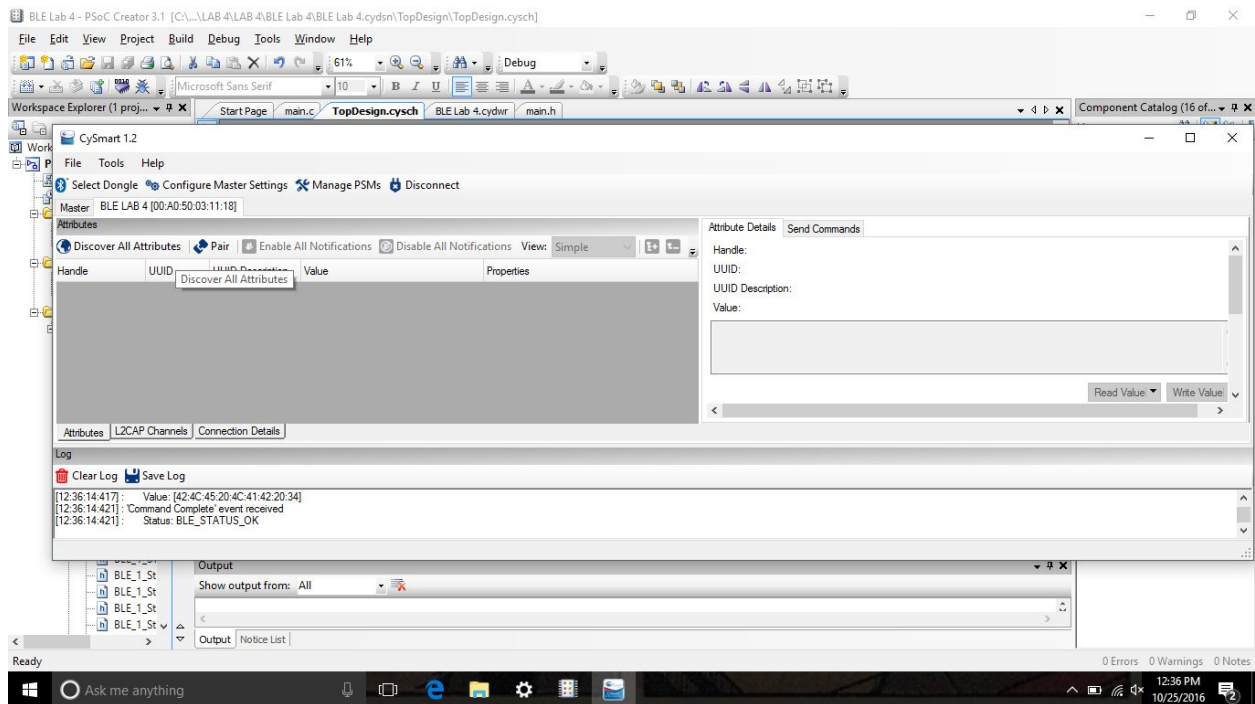


- 3) Add the “CapSense CSD Component” to the schematic.
- 4) Configure the CapSense CSD Component name to be “CapSense”
 - a) General Tab
 - i) Default Settings
 - b) Widgets Config Tab
 - i) Add “Linear Slider Button”
 - (1) Sliders Settings: Default
 - c) Scan Order Tab
 - i) Each element’s sensitivity can be changed hers
 - ii) Values 0:10 (Lower values= higher sensitivity)

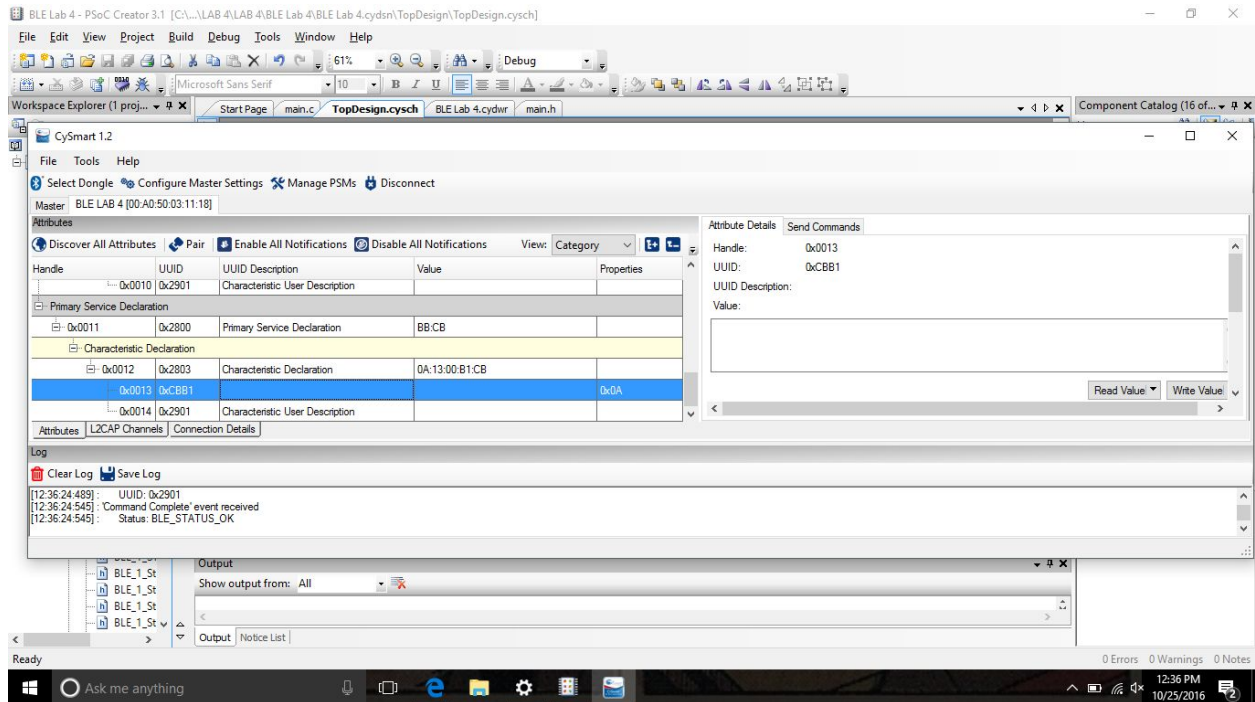
b) Build Project

Testing with CySmart Central Emulation Tool

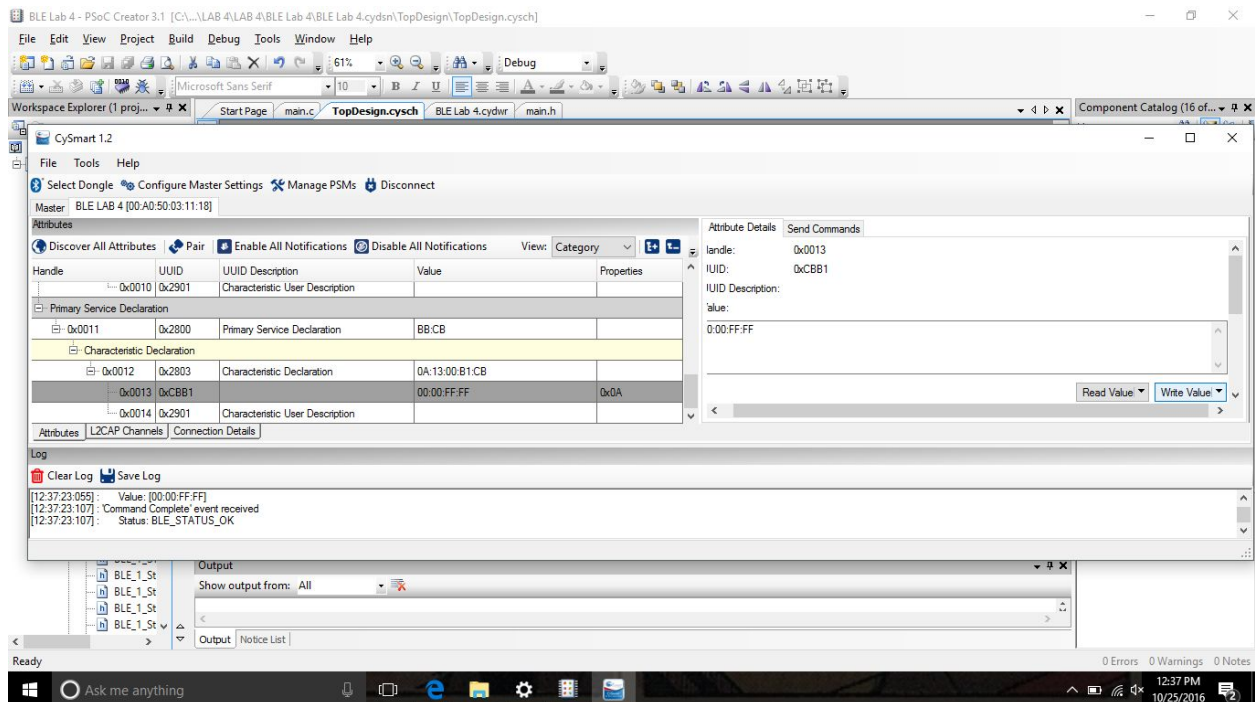
- 1) Open Cysmart 1.0 and connect it BLE-USB Bridge
- 2) Start Scan and Connect to your GATT Server Device
- 3) Discover All Attributes and then scroll down the attribute list to the RGB LED Characteristics



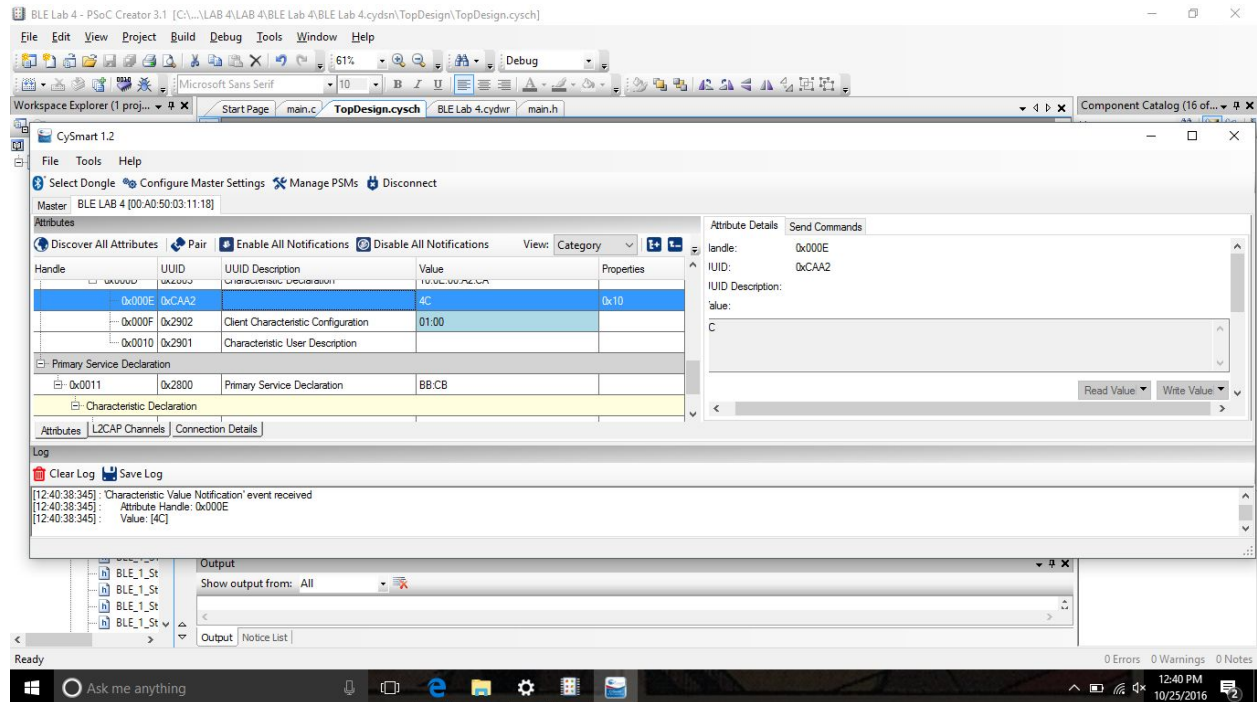
- 4) Write a 4 byte value to this characteristic on the right and notice the corresponding color and intensity of the RGB LED on the Kit. Byte 0 corresponds to the Red color, Byte 1 corresponds to the green color, byte 2 corresponds to the blue color and byte 3 corresponds to the intensity.



- 5) For the CapSense Slide “Enable All Notifications”
 - a) Write “1” to the CCCD Descriptor (UUID=0x2902)



- 6) Move your finger over the slider on the kit and observe the Characteristic changes in the CySmart tool, while the tool's log shows notification packets being received.



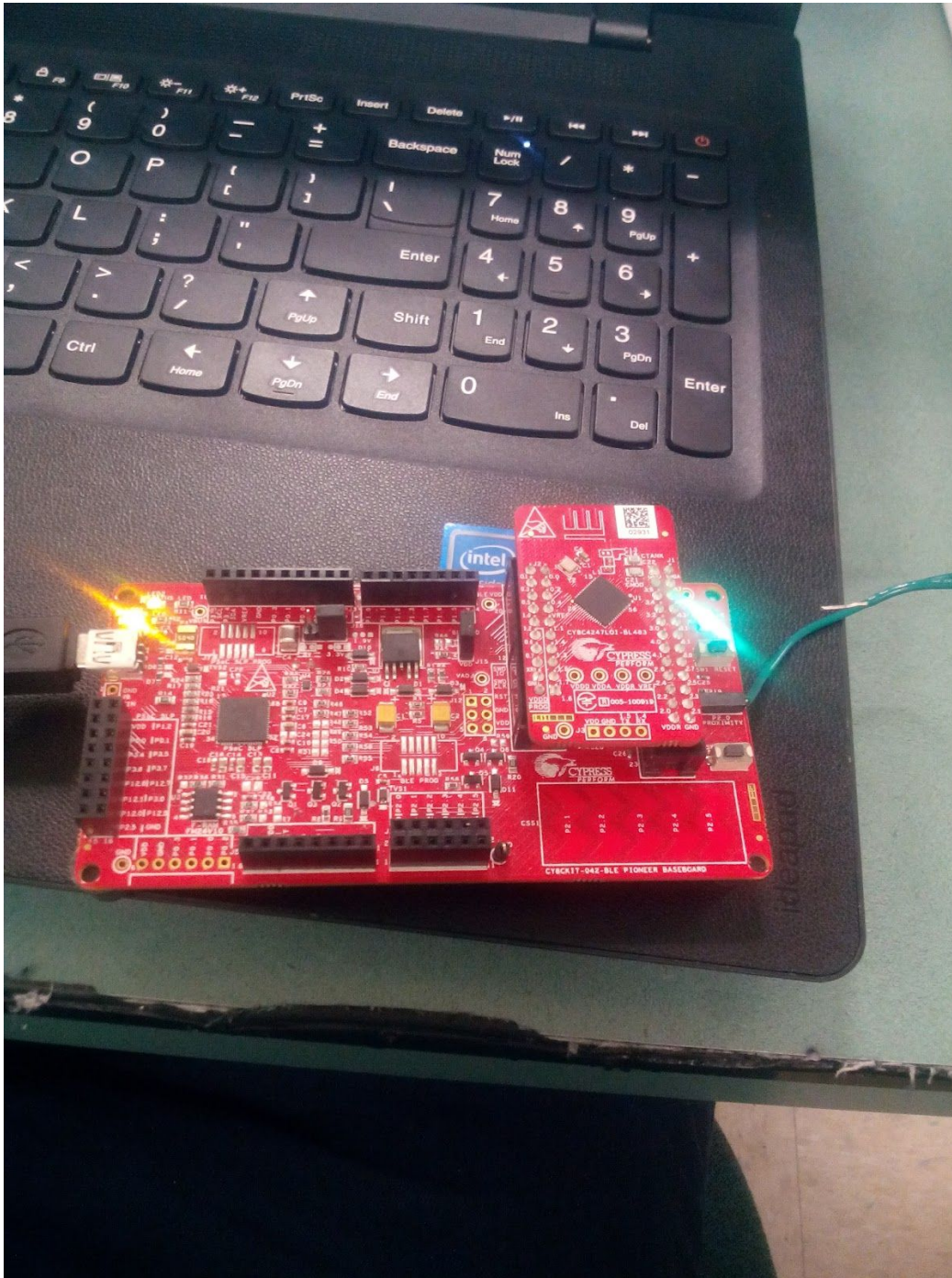
Testing with CySmart Mobile App

- 1) Open the "CySmart Mobile App" on your phone.
- 2) Connect your GATT Server device on the app.
- 3) Select the RGB LED Service. Tap anywhere on the color gamut to see the corresponding color on the RGB LED on the kit.

Testing RGB LED



- 4) Move the slider position on the app page to change the brightness level of the LED.

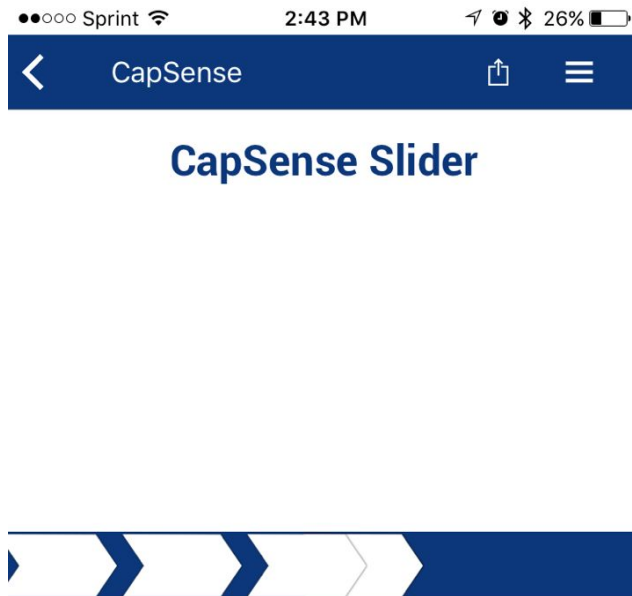


or

Color should change as you move your finger along the RGB LED Service screen

5) Now go back one page in the app and select the CapSense Slider Service

6) On the CapSense Slider page, you can move the slider with your finger.



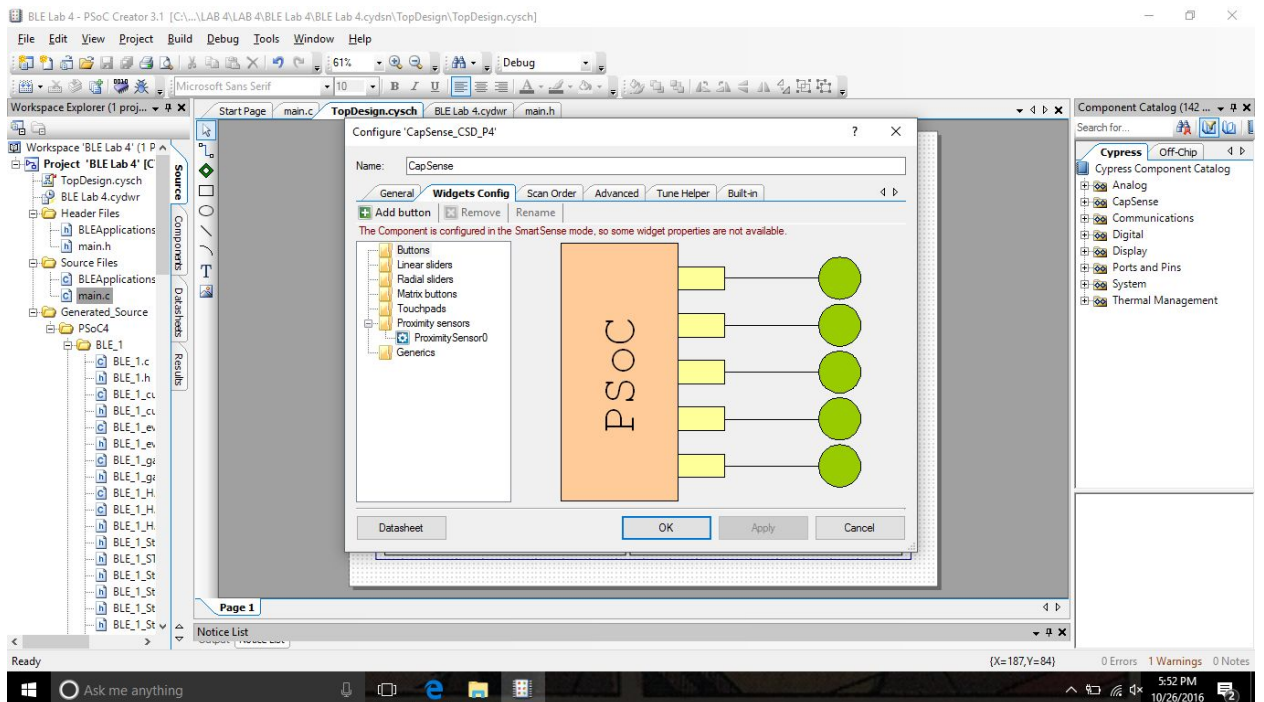
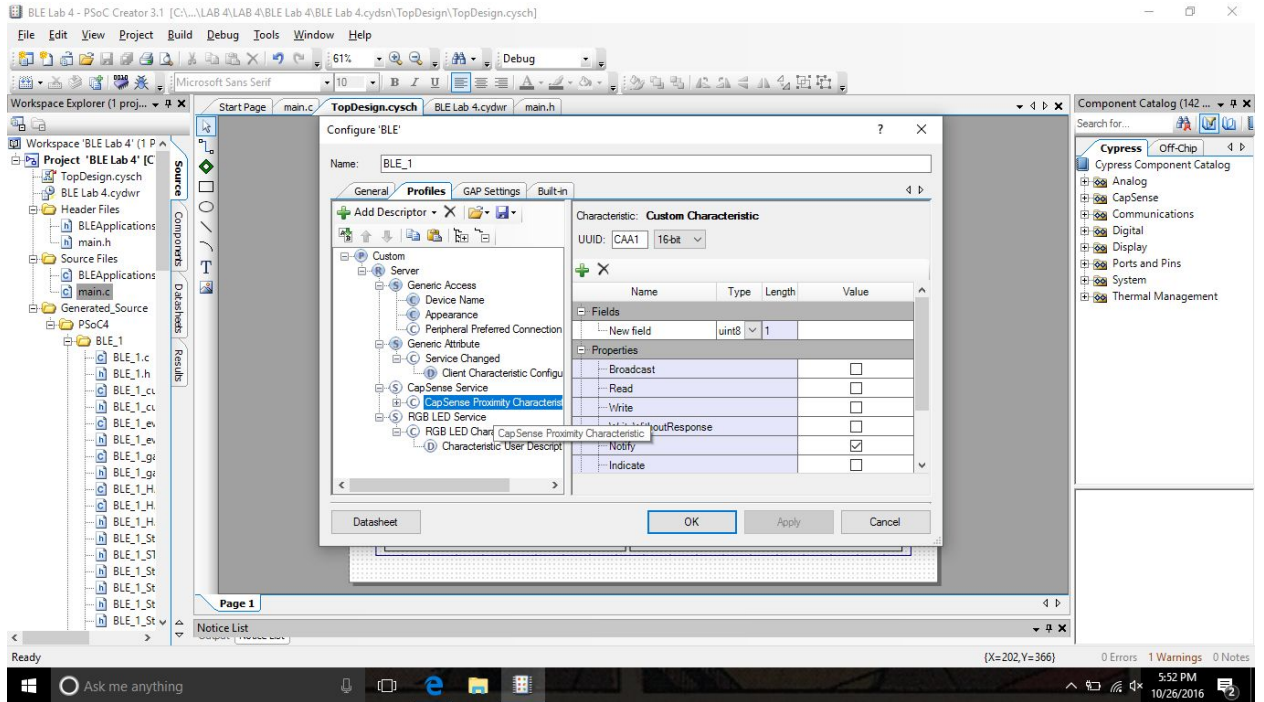
CapSense Slider screen image on your phone should be align with user's fingers on the PSoc Device

Additional Exercises

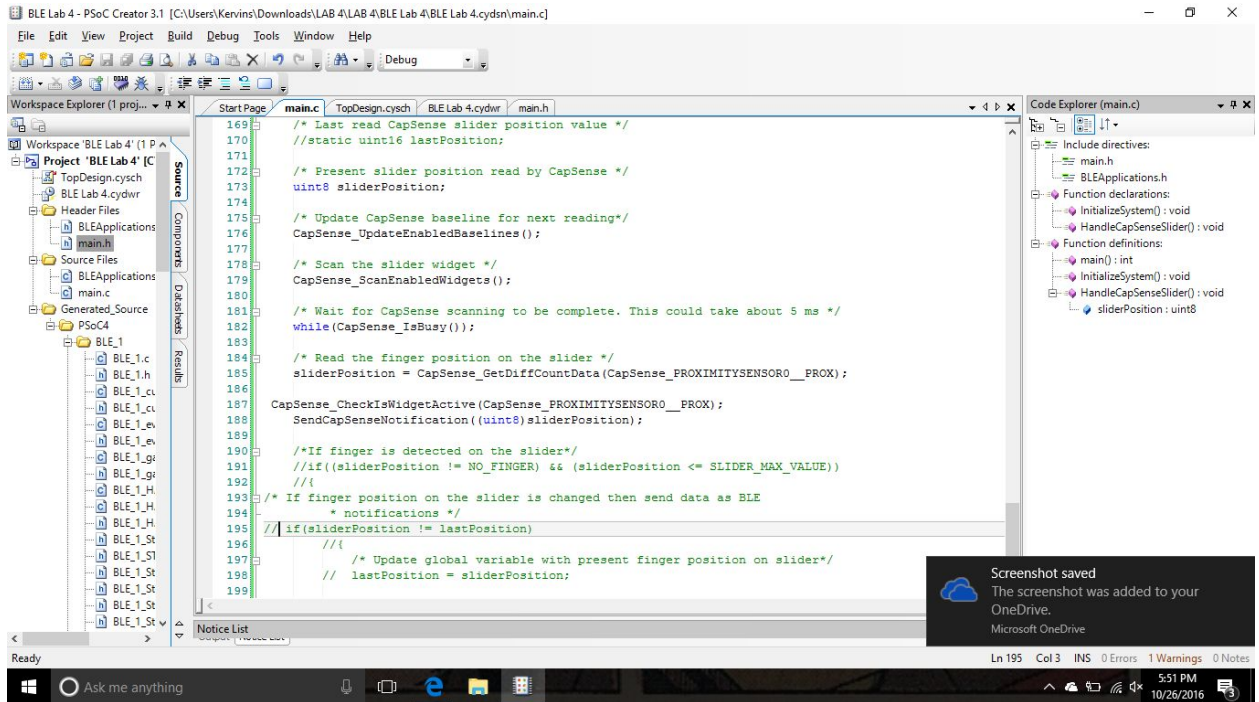
1) Replace the CapSense Slider with a CapSense Proximity Sensor.

Hints:

a) Use CAA1 as the UUID for CapSense Proximity Service.

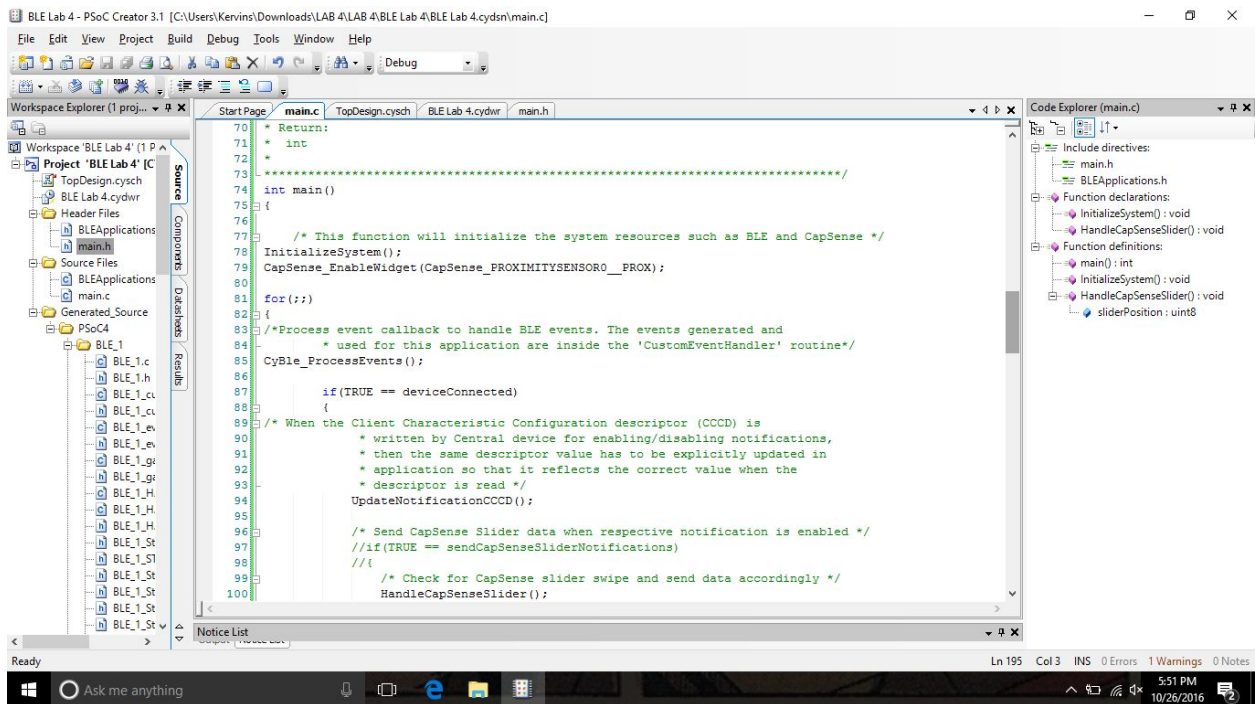


- c) Use the API function
CapSense_GetDiffCountData(CapSense_PROXIMITYSENSOR0__PROX)
instead of CapSense_GetCentroidPos(CapSense_LINEARSLIDER0__LS) to extract
the proximity value



Initializes API function in order for CapSense can understand the data it is being received

d) Modify the code to always send the CapSense Proximity sensor Notification data



```
70 * Return:
71 * int
72 *
73 .....*/
74 int main()
75 {
76
77 /* This function will initialize the system resources such as BLE and CapSense */
78 InitializeSystem();
79 CapSense_EnableWidget(CapSense_PROXIMITYSENSOR0__PROX);
80
81 for(;;)
82 {
83 /*Process event callback to handle BLE events. The events generated and
84 * used for this application are inside the 'CustomEventHandler' routine*/
85 CyBle_ProcessEvents();
86
87 if(TRUE == deviceConnected)
88 {
89 /* When the Client Characteristic Configuration descriptor (CCCD) is
90 * written by Central device for enabling/disabling notifications,
91 * then the same descriptor value has to be explicitly updated in
92 * application so that it reflects the correct value when the
93 * descriptor is read */
94 UpdateNotificationCCCD();
95
96 /* Send CapSense Slider data when respective notification is enabled */
97 //if(TRUE == sendCapSenseSliderNotifications)
98 //{}
99 /* Check for CapSense slider swipe and send data accordingly */
100 HandleCapSenseSlider();
101 }
```

Must enable the proximity with this code, implements the .cydwr

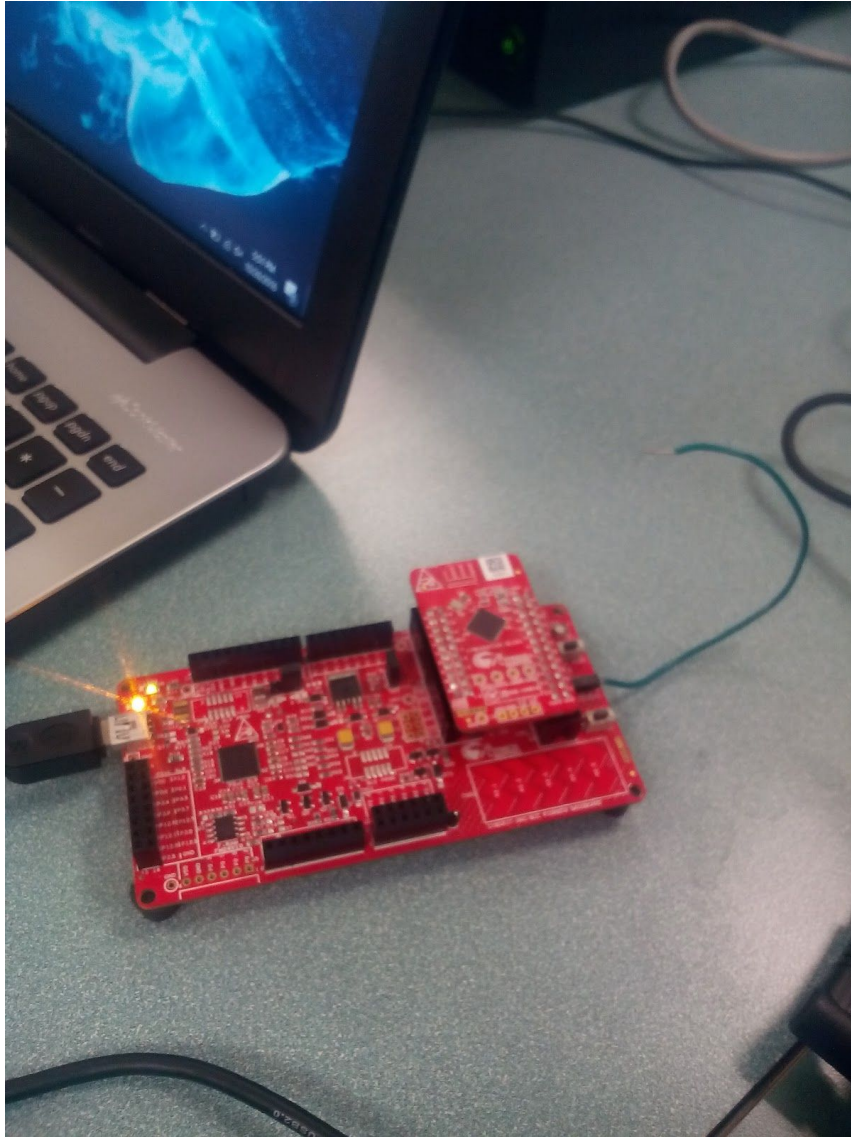
Once program is built and programmed unto the board:

Find the proximity sensor on your BLE Pioneer Kit

P2[0] 0A0: vplus, SCBO:spi_select[1]

/Implementing a wire into the device can help detect movement/

(Doesn't need to be added)



Using your phone

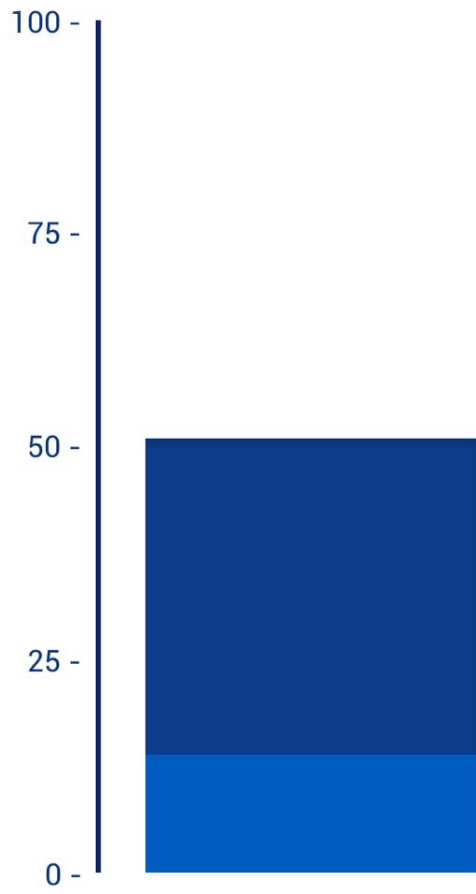
Find the proximity sensor in the CySmart Application

As you move your hand above the device

You should see device detecting any objects near its sensor

/As you get closer to the sensor you should hear a beeping sound detecting movement/

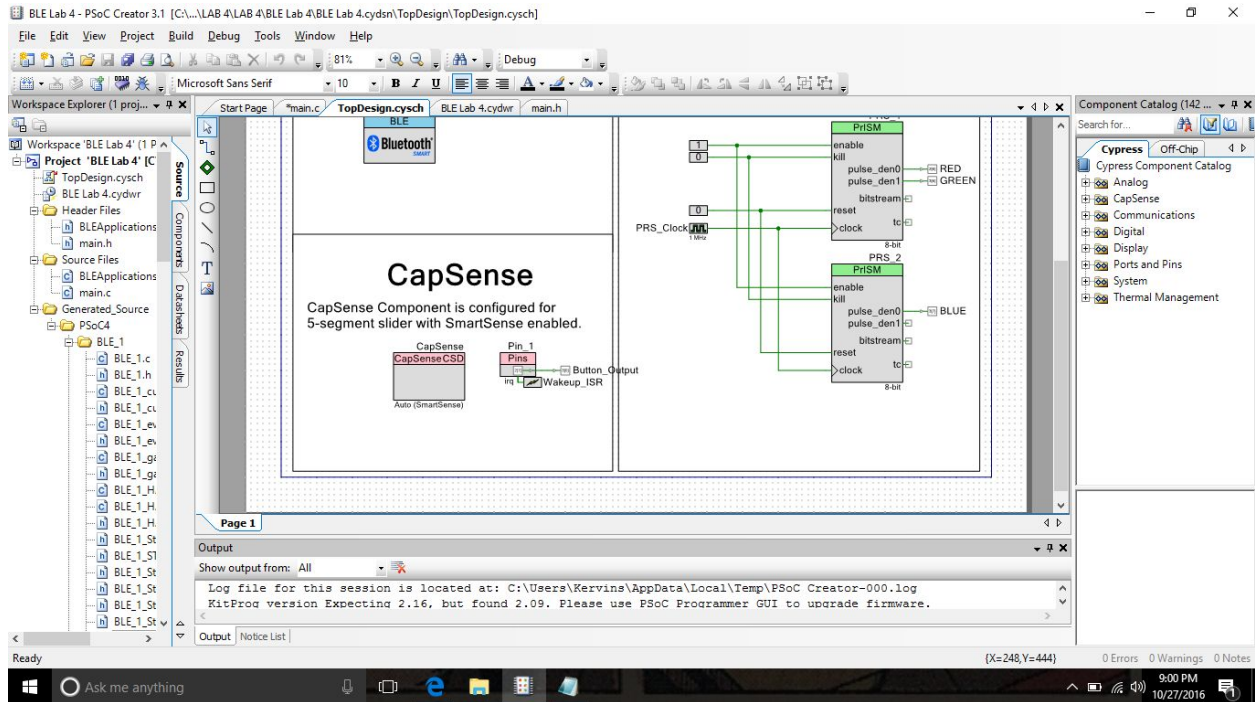
CapSense Proximity



2) Implement the low-power operation for Lab 4.

Hint:

- Follow the firmware implementation from PSoC 4 BLE Lab 3.
- Add a switch to wake-up the device from Hibernate mode.



Add Pins

Must select:

Digital input/ Hardware connection

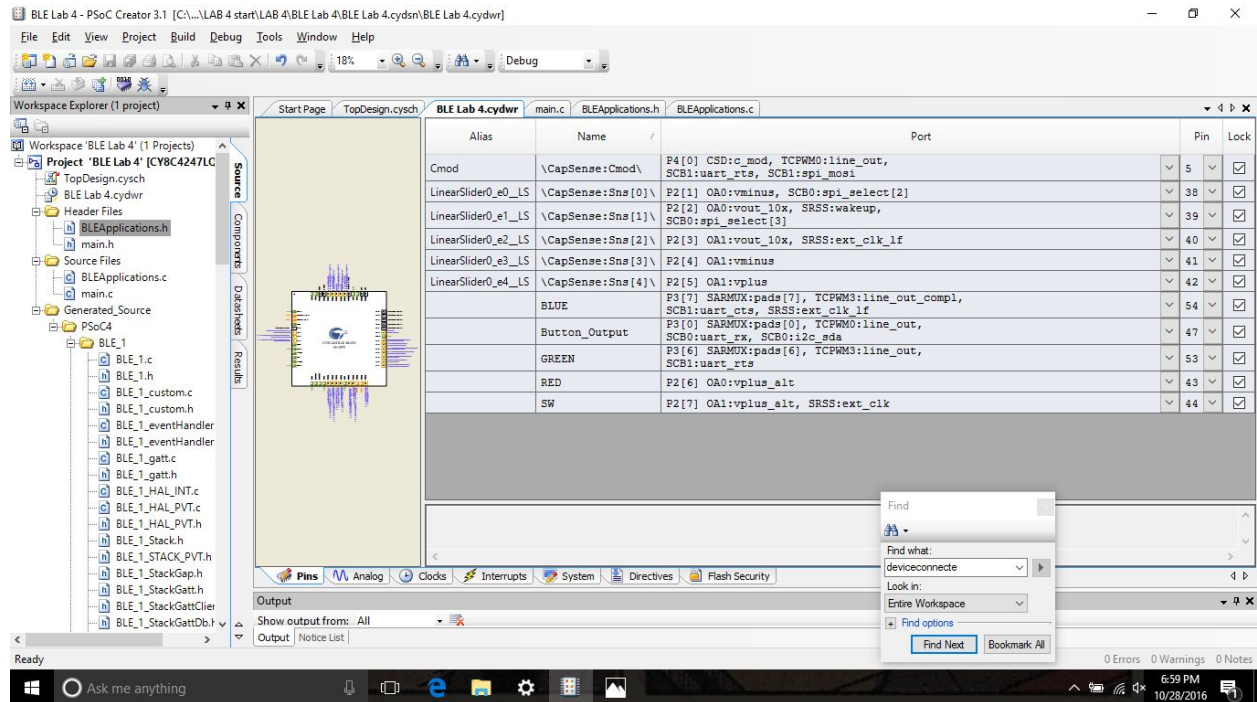
Drive mode must be in Resistive Pull up

Must add an Interrupt

Interrupt is connected to Wakeup_ISR

Wakes up Device

Add Output Pin (Button_Output)

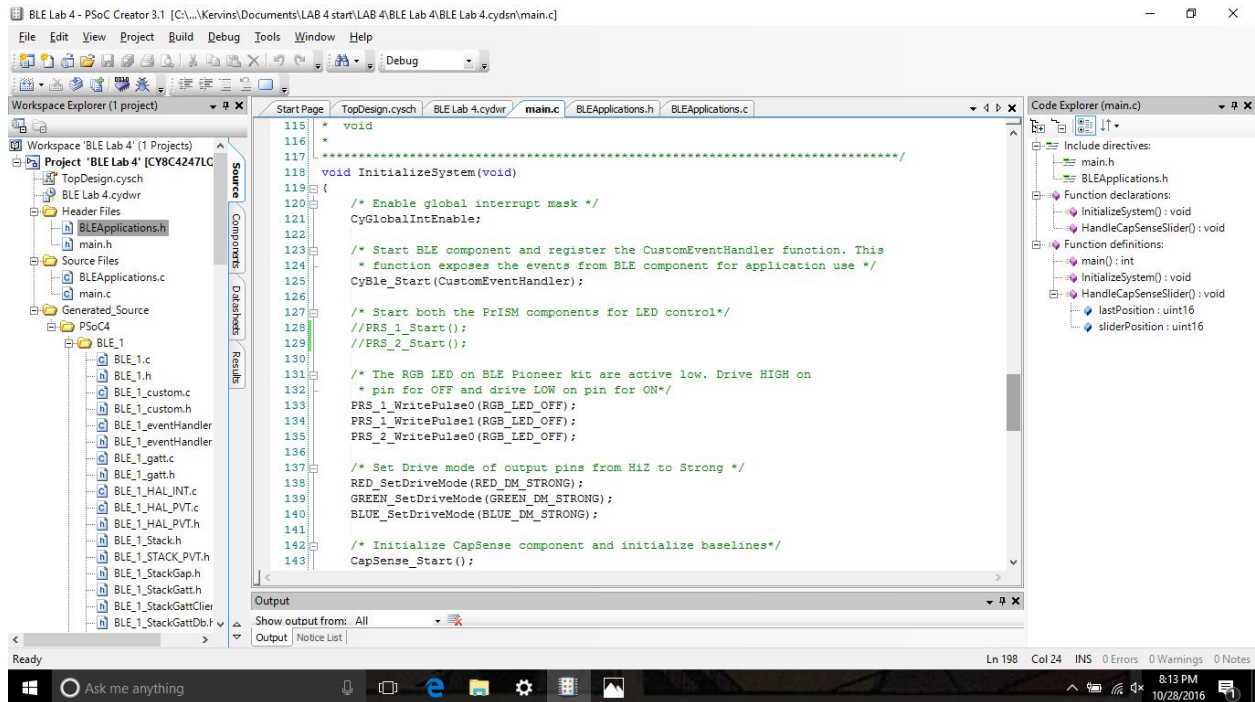


Must assign Pins on the DWR file

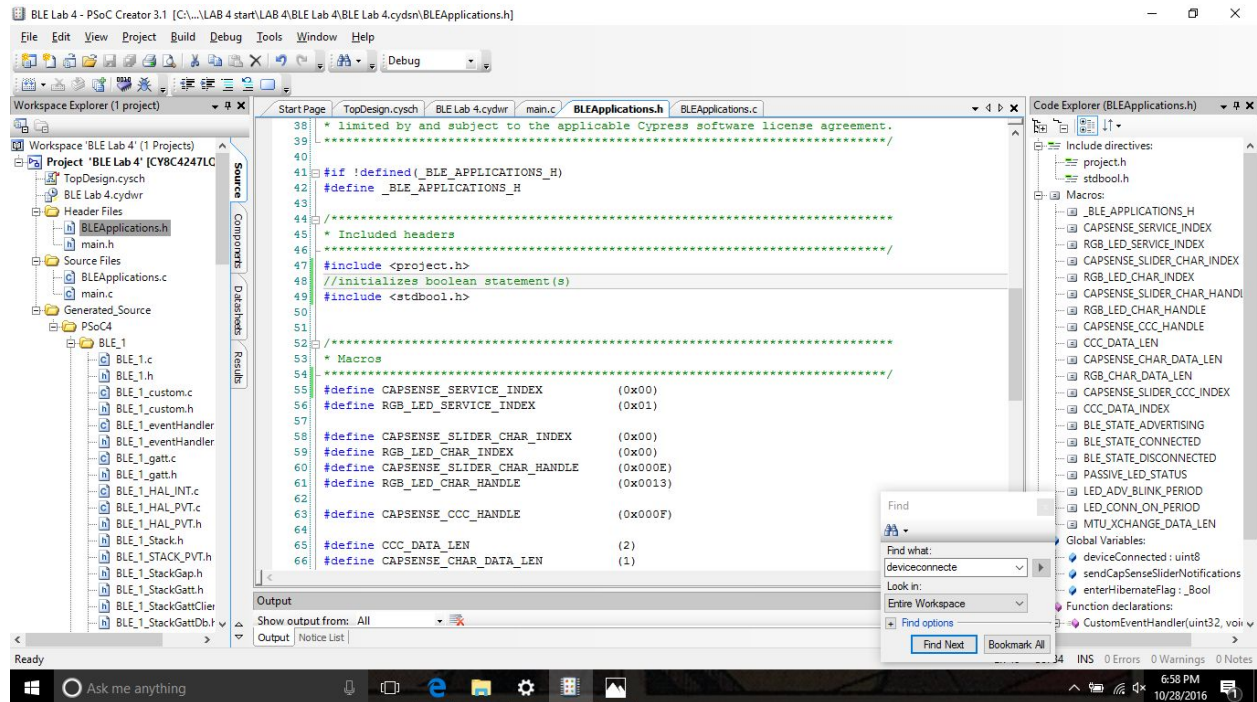
Pin (Sw) = P2[7]

Pin (Button_Output) = P3[0]

- c) The PRISM Component is not active during the Deep-Sleep mode, so comment-out the code for putting the device in the Deep-Sleep mode.



Prism is commented out of main.c but still enabled in TopDesign



Initialize boolean statement in order to add:

Add in BleApplication.h

Extern bool enterHibernateFlag;

Hibernate puts device into Deep Sleep mode

Add in BleApplication.c

Initialize hibernate mode to be true in order for code to work

enterHibernateFlag = true;

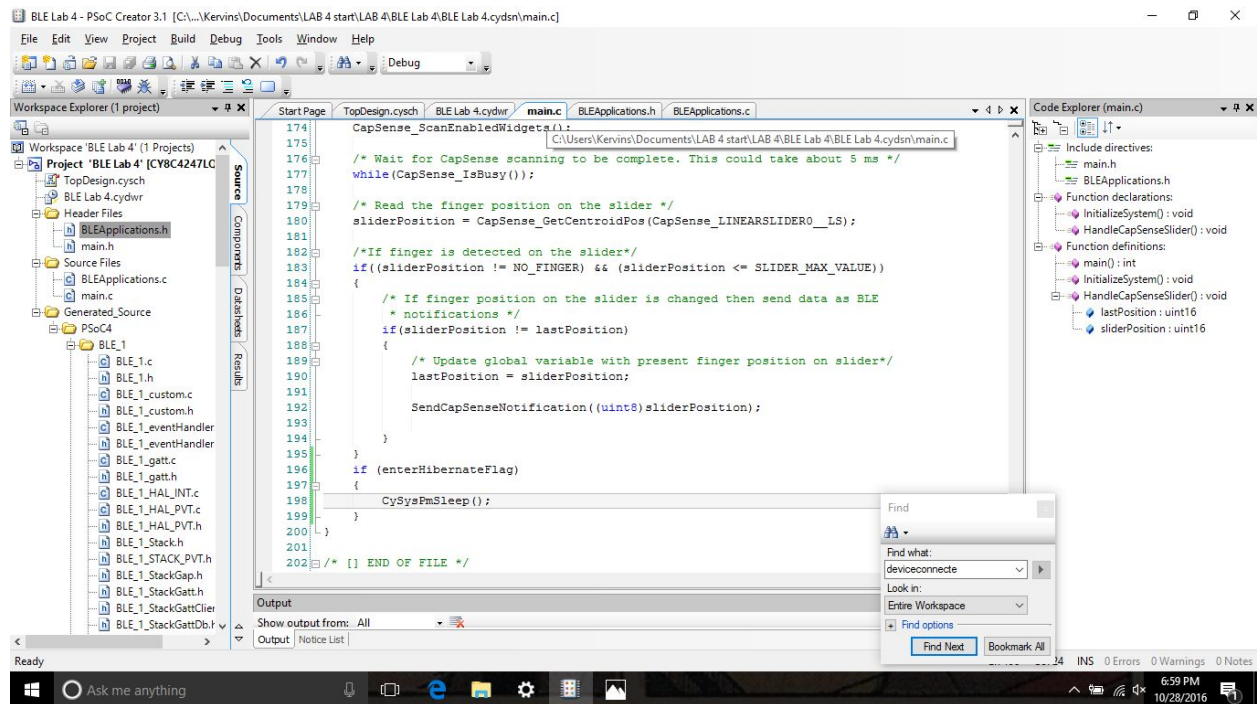
Once this is done must initialize the device hibernation mode in main.c file

if(enterHibernateFlag)

```

{
    CySysPmSleep();
}

```



Once the program enters Hibernation mode, the Deep Sleep is initialized and causes the device to go to sleep

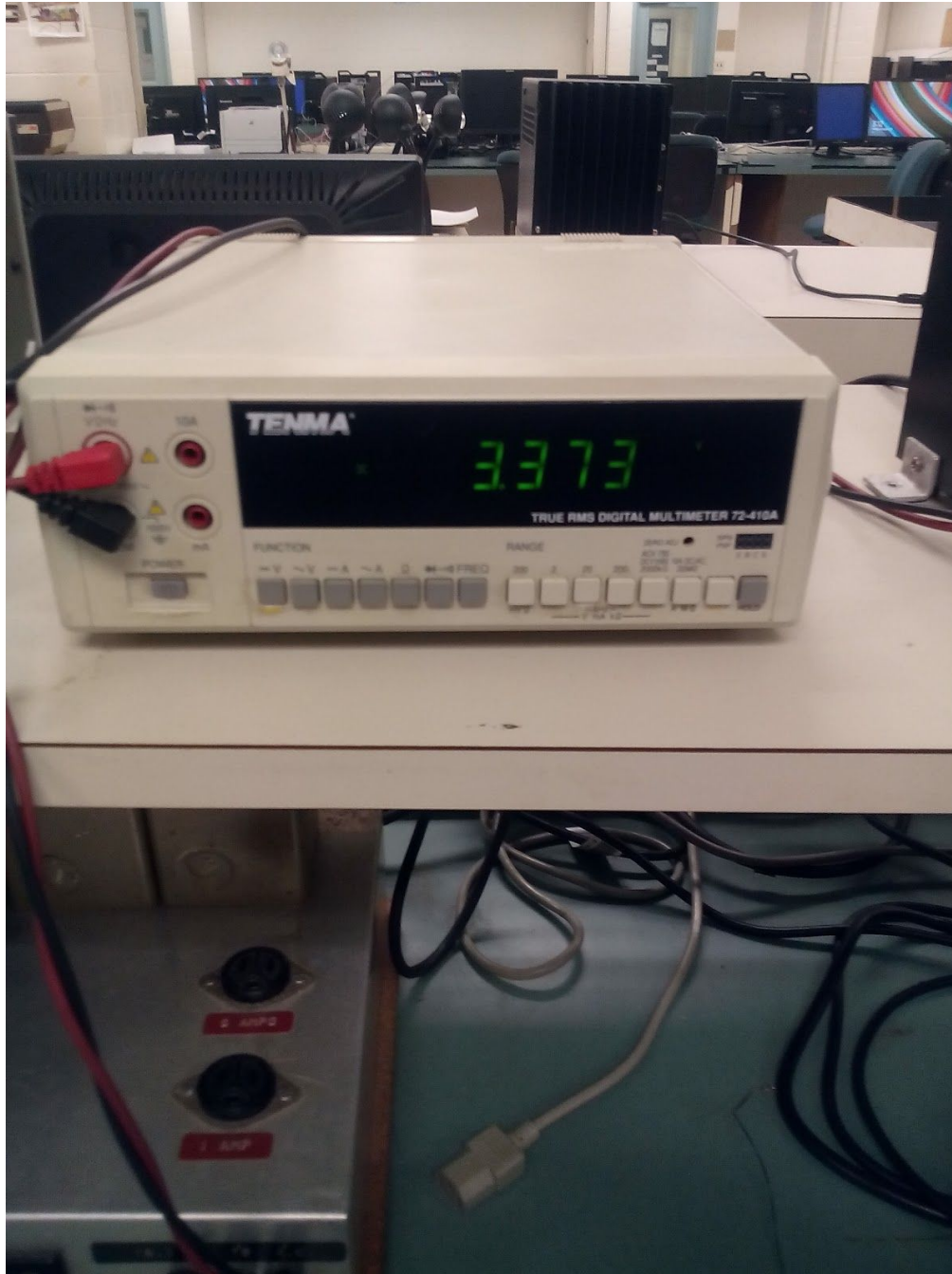
When built and programmed:

Device starts in Sleep mode

Voltage should be low

When Switch 2 is pressed on the BLE Device the voltage of the device

Voltage is increased



Device inputs:

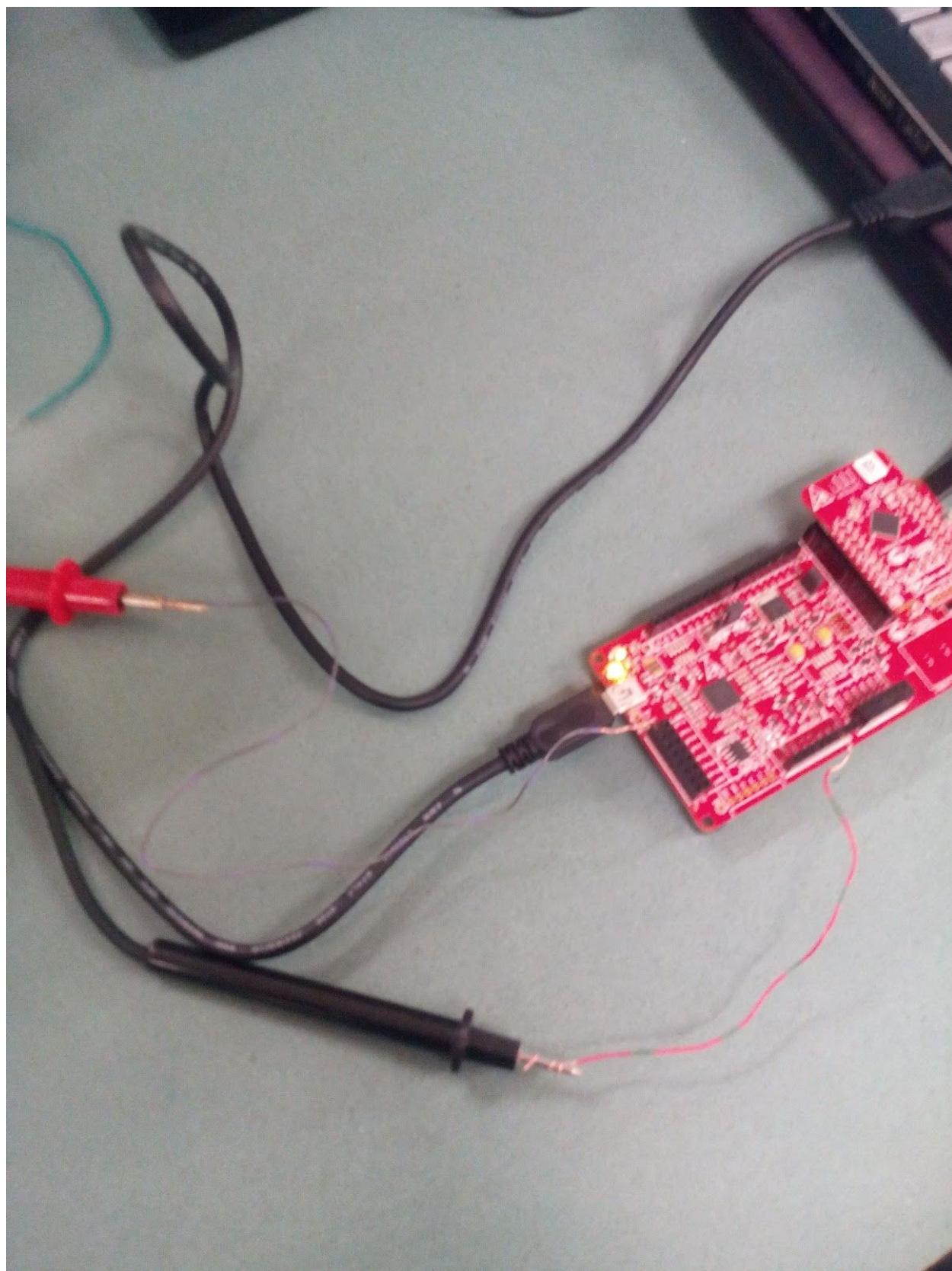
Can be tested by connecting a wire into the power

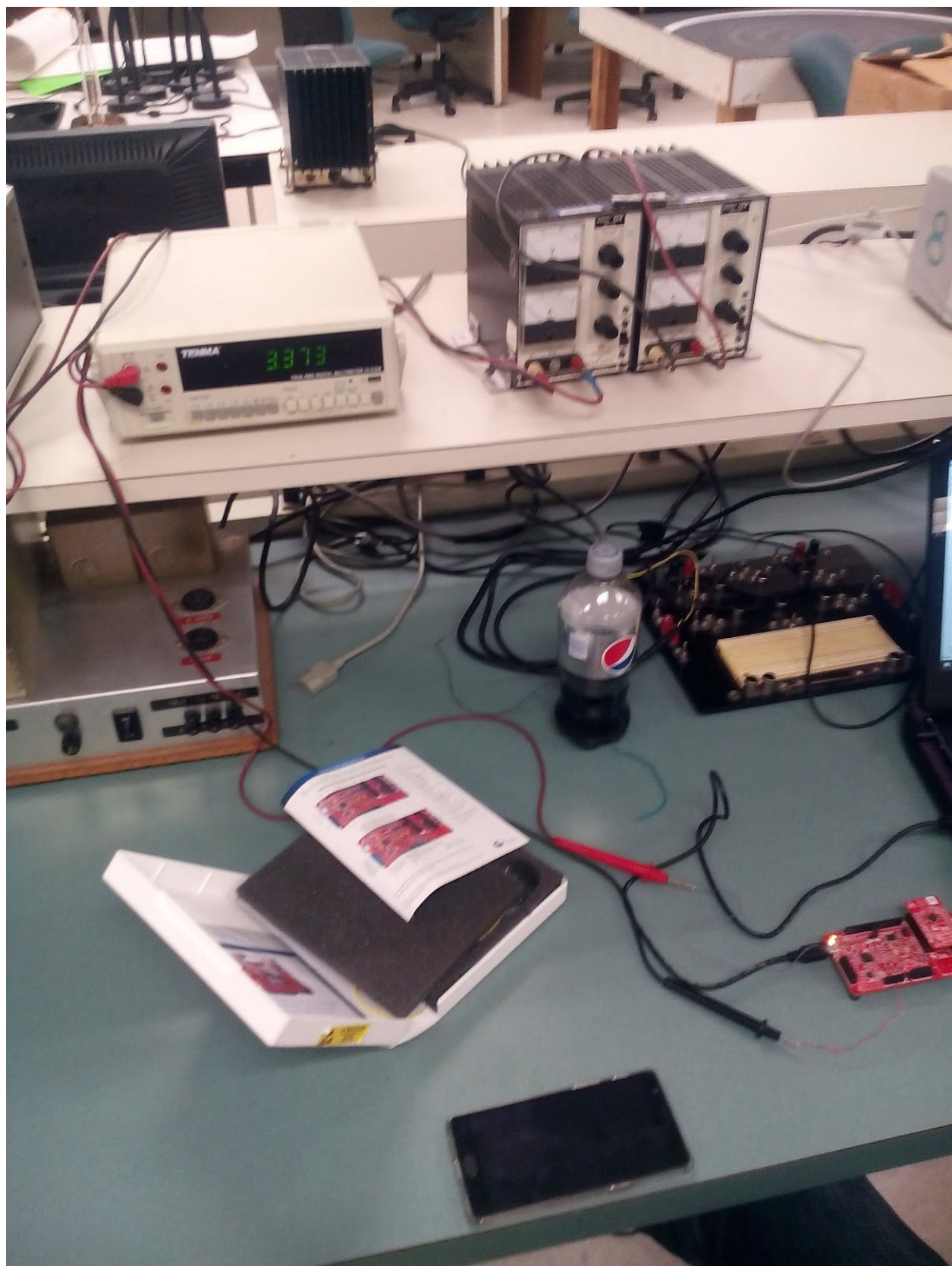
VDD P1.2

And a wire into the Ground source

GND

To test the voltage.





Conclusion:

This lab gave the user's knowledge about CapSense and the way it is implemented using BLE Connectivity. Students were able to pick which CapSense CSD Component they would like to use. Configuring its characteristics in the BLE and Enabling the properties so the project can be implemented. These Characteristics are then added into the DWR file in order for the pins to be initialized. Building and Programming the code leads you to codes that the user can manipulate in order to activate all the aspects of the CapSense. Also understand how to configure the code and Psoc in order for the device to work in different modes. Making the BLE, CapSense and RGB LED to work on various devices was a way to get the student to think about the extent of the CapSense Design and Bluetooth connection.