Educational Objective:

The educational objective of this laboratory assignment is to investigate both the OLED LCD and Bluetooth modules on the TI-RSLK. This lab will primarily make use of provided library functions due to the complexity of these modules. At the completion of this lab, you will be able to use data received via Bluetooth to control the direction of the robot and display that data on the OLED LCD.

Background:

OLED LCD

The display module on the TI-RSLK is the SSD1306 128 x 64 dot matrix OLED (Organic Light-Emitting Diode) graphical display. You can find more information about this display in the datasheet posted on MyCourses (SSD1306.pdf). The display is hardwired to the MSP432 Launchpad via the following pins:

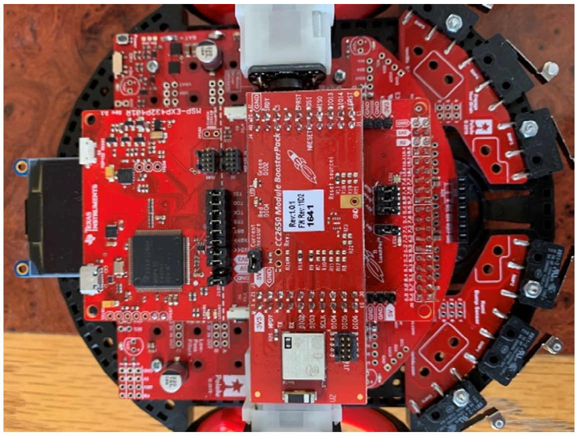
MSP432 P9.7 is SPI USC3SIMO, also called LCDMOSI (data)
MSP432 P9.6 is a GPIO, also called LCDDC (data/command)
MSP432 P9.5 is SPI UCA3CLK, also called LCDSCLK (clock)
MSP432 P9.4 is SPI UCA3STE, also called LCDCS (select)
MSP432 P9.3 is a GPIO, also called LCDRST (reset)

Notice that the data, clock and select lines are connect to MSP432 SPI specific pins for **eUSCI_A3**. The SSD1306_Init() function, which is provided to you, sets the P9SEL1 and P9SEL0 registers accordingly. Pins P9.3 and P9.6 are set to be regular digital outputs, and are also set in SSD1306_Init(). DC=1 means that the information being transmitted on LCDMOSI is data, and DC=0 means that the information being transmitted on LCDMOSI is a command. For the prelab, you will write two functions to send a byte of data to the OLED LCD. One function is to send a command (DC=0) and the other is to send data (DC=1). Notice the display does not run off power from the USB cable. Battery power on the TI-RSLK MAX robot must be active to operate the OLED.

Bluetooth Low Energy (BLE)

Bluetooth Low Energy is a ubiquitous protocol used to wirelessly send and receive data between devices in the same room. It is a complex protocol with a wide variety of features. In this module we have simplified BLE two ways. First, the low-level details of the radio and wireless communication are implemented on the CC2650 in a system called the Simple Network Processor (SNP). The high-level abstraction exists on the MSP432 as the Simple Application Processor (SAP). Second, this SAP-SNP system supports dozens of commands, but we will expose only the minimal set needed to establish a simple BLE link.

The TI-RSLK uses the CC2650 Module Booster Pack for BLE communication. When you install the module be extremely careful to orient it correctly on the Launchpad board. Use the power and ground pins to guide you and refer to the picture below:



The CC2650 board is connected to the MSP432 via UART1. The following table shows the pins used for the SNP-SAP system:

MSP432	SNP-SAP	CC2650	Description
P6.0 GPIO out	MRDY	DIO7	Master Ready
P2.5 GPIO in	SRDY	DIO8	Slave Ready
P6.7 GPIO out	NRESET	Reset	Reset to CC2650
P3.3 UART TxD	RX	DIO1 RXS	MSP432 -> CC2650
P3.2 UART RxD	TX	DIO0 TXD	CC2650 -> MSP432

The details of the GPIO interface are described in the provided GPIO.c file. The details of the UART interface are described in the provided UART1.c file.

We will be using a smart device to communicate with the robot. If you do not have a smart device, you may share one from a friend or borrow one from the lab. Initially we will be using an app called **LightBlue,** which is available for both iOS and Android devices. <u>You should download this app before lab.</u>

Pre-Laboratory:

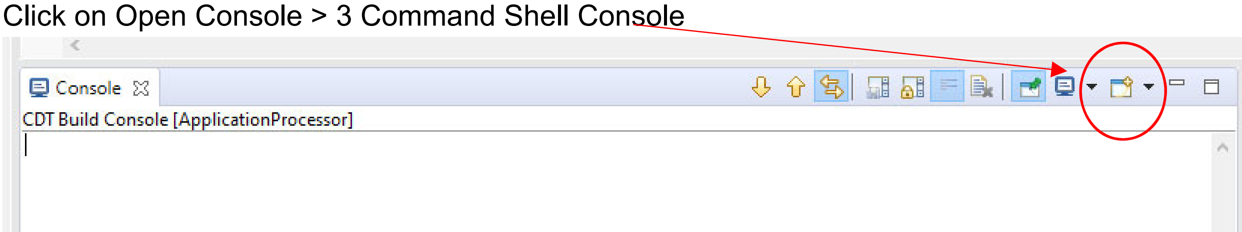
1. Read and understand the requirements for this lab. **If you have not done so, go back and read the background information.**
2. Create a new project in Code Composer called Lab6_OLED. Download the Lab6_OLEDmain.c and SSD1306.c files from MyCourses and put them in the project. Also copy the Clock.c file to your project.
3. In the SSD1306.c file, review the **SSD1306_Init()** function to better understand the configuration of SPI for use with the OLED LCD. We are using eUSCI_A3.
4. Using the lecture notes as a guide, complete the code for the **commandwrite()** and **datawrite()** functions in SSD1306.c. Comments are included to help guide you.
5. You will know your functions are correct if you compile and run the Lab6_OLEDmain code, turn on the battery power and see characters on the display after pressing the MSP432 Reset button.
6. Submit your functions to the Lab 6 prelab dropbox in MyCourses.

The second part of this prelab will establish that you are able to create a BLE communication link between your robot and a smart device (no coding required).

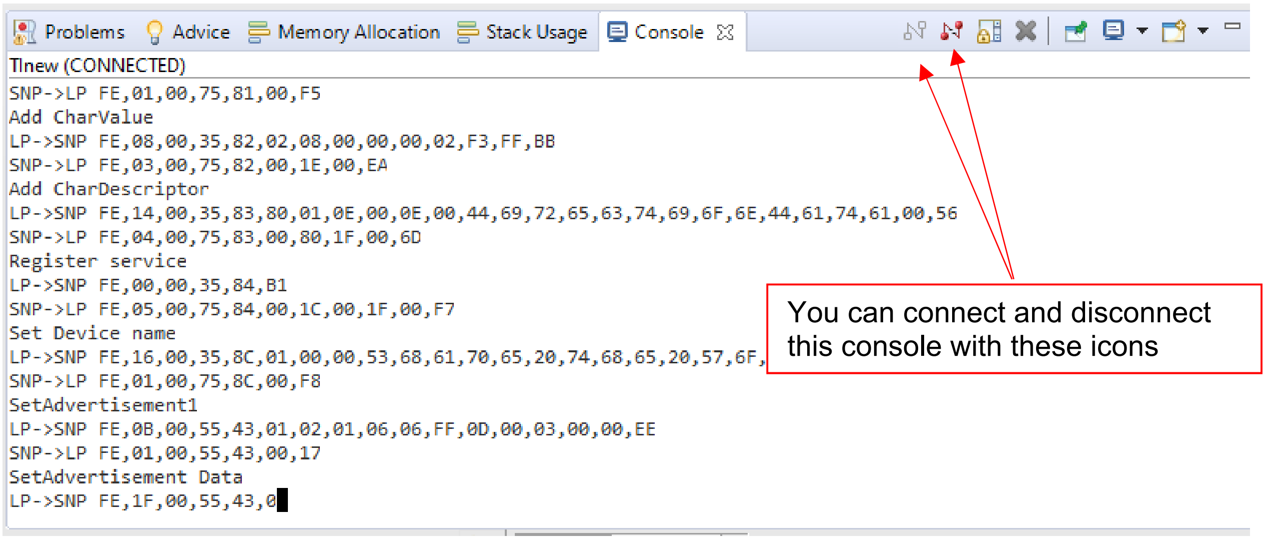
7. Create a new project in Code Composer called Lab6_BLE. Download the VerySimpleBLEmain.c, AP.c, UART0.c files from MyCourses and put them in the project. Also copy the Clock.c, CortexM.c, GPIO.c, UART1.c and LaunchPad.c files from the inc folder to the project. You will need to edit the include paths in GPIO.c and UART1.c to point to the .h files in the inc folder.
8. Build and run the code. When you open the LightBlue app on the smart device, you should see “Shape the World” as an available device. This is the CC2650 module on your robot. You may need to flash the board and move away from your classmates to verify you are connecting to the correct robot.
9. Select Shape the World and then scroll down in the app until you see Data, Switches, LEDs and Count. Select LEDs and then ‘write new value’. Enter a value between 1 and 7 and see how LED2 on the LaunchPad is affected. Feel free to experiment with the other options.
10. Be prepared to show your app connection and control of LED2 at the start of your lab section for prelab credit.
11. If you do not have a laptop to bring to lab, put your code on a flash drive.

Procedure:

1. Download the Lab6_BLEmain.c file from MyCourses and add it to your project. Also copy motors.c from lab 5 and SSD1306.c from the prelab to this project. In VerySimpleBLE.c, change **main(void)** to **main1(void)** so that it is no longer the program that is run.
2. In AP.c, find **NPI_SetAdvertisementDataJacki[ ]** and change the name from “Micros rocks” to your own name.
3. Compile and run the program. On the LightBlue app you should see your device. Select your device and scroll down to see ‘DirectionData’. DirectionData is a write characteristic which you can use to send direction commands to the robot.
4. Open a console window so you can see the data being passed between the CC22650 and the MSP432.
    a. Click on Open Console > 3 Command Shell Console



    b. Select ‘Serial Port’ for Connection Type
    c. Click on New…. In the popup window, assign a new name. To find the correct COM port, open Device Manager > Ports(COM & LPT) and then identify the COMn for UART1. Choose COMn for Serial Port. All other parameters can stay the same. Click finish and okay.
    d. Your new console should be added and look similar to the figure below:



5. As you write new values to DirectionData on the app, you should see them appear in the console window. Verify the data is being received and printed to the console before continuing.
6. Complete the **MoveRobot()** function so that each value (1-5), received from the app, controls a different motor function (forward, backward, right turn, left turn, and stop). You can use your motor functions from previous labs and you do not need a state machine. The robot will continue in the direction it was heading until it receives a new command from the Bluetooth controller.
7. Using the Lab6_OLEDmain.c code as a guide, display the robot direction to the OLED LCD.
8. Demonstrate your Bluetooth controlled robot for a signoff.

Post Lab:

Take the Post lab quiz in MyCourses before your next lab section.

Signoffs and Grade:

Name:___________________________________

Component	Signoff	Date
OLED LCD Displays Direction		
Bluetooth controlled movement		

==

Component	Received	Possible
Prelab		20
Signoffs		60
Post-lab Quiz		20
Penalties • after the start of lab session -10pts • after 1 week late -25 pts • no signoffs after 2 weeks late	-	
Total		100