**EEET2250 – Week 2 Lab Task (due Week 3)**
**Accessing the OUSB board using functions**

**Total possible marks: 15 marks (3% of final mark)**

**Aim:**
The following lab tasks are aimed at helping you to build up code for Lab Test 1 which will be held during your week 4 lab session – Both the week 2 and 3 lab tasks are directly related to Lab Test 1.

This week's tasks require you to write several small sections of code that read and write values to the I/O ports on the Open USB-IO (OUSB) microcontroller board using **function-based code**. We will be working with the OUSB board throughout this course extensively; you may have encountered the board in some of that lab exercise in EEET2246. If not, you can familiarise yourself with the board by reading the information about the OUSB board available on the Canvas shell for EEET2250.

Specifically, you will be required to write a program that uses functions to connect to the OUSB board, read the current value of PORTB, and write an integer to PORTB on the OUSB board; PORTB is the LED array on the bottom left hand corner of the OUSB board. Composed of 8 LEDs, PORTB therefore has a limited range of acceptable values!

You must write (at least) two functions in your program to achieve the following behaviour:

1. main(…): the main entry and exit point of your program for interacting with the OUSB board.
2. A function that enables connection to the OUSB board: system calls are *not* allowed e.g., `system("ousb -r io portb 127")`. You must access the OUSB board using pipes.
3. A function that can read input from the OUSB board. This includes reading from PORTB as well as other I/O ports on the OUSB board (e.g., DIP switch, light-dependent resistor, trimpot).
4. A function that can write output to PORTB on the OUSB board. For the purpose of the lab task. Consider using command line arguments, as well as asking the user for input via the `cin` input stream class to take in one integer argument, then output this data to PORTB.

You may find it helpful to write all the code in main(...) first and then break out the code into functions; however, your code must be eventually function-based for assessment by your tutor.

Note that to connect, read data, and write data to the OUSB board can be achieved through writing a generic function that executes any command to the OUSB board i.e., the minimum number of functions required for your program is two (including main) – Note: *a generic function that executes any command to the OUSB board could be helpful for your lab test as it save you writing many specific functions that all have the standardized pipe code in them*!

The tasks specified below are aimed at helping you to break down the program into smaller functional 'modules': you should attempt to code in modules, checking the functionality of each module before continuing to code (i.e., do not write the whole program at once and expect it all to work!).

Once you have finished all of the tasks notify your tutor who will assess your work. The marking for these tasks is binary – it either works or does not.

Remember to use the lecture notes as a reference as well as the prescribed textbook and the OUSB reference material by Dr Pj Radcliffe on www.pjradcliffe.wordpress.com or the reference material available on EEET2250 Canvas shell.

**Tasks**

1. When you attempt each of the three tasks below, remember the six (6) steps of engineering design: draw flowcharts and write pseudocode to represent the logical flow of your program. (Hint: it might help to have a flowchart for the overall program as well as for each function!)

   [2.5 marks]

2. Write a function that connects to the OUSB board, and call this function from your main(...). You may find it helpful to write all the code in main(...) first and then break out the code into a function. You can use the example OUSB code provided to help you write this function (use pipes *not* system calls!), your function should print an error to the screen if the code cannot connect to the board (hint: check what the ousb.exe file returns when you try to connect to it). Remember to set a breakpoint at the end of your application to prevent the window from closing.

   [2.5 marks]

3. Write a function and/or use the function from task (2) to read the current value of PORTB from the OUSB board and return the value to main so it can be printed to the console. For this lab exercise you are only required to read from PORTB.

   [2.5 marks]

4. Update the function form the previous task so that it capable of reading from other I/O ports on the OUSB board (e.g., DIP switch, light-dependent resistor, trimpot) and output the reading from the I/O port onto PORTB (beware of the acceptable range of values for PORTB!). Make sure you show your lab tutor what references you used to find out how to communicate to the other ports on the OUSB board.

   [2.5 marks]

5. Write a function and/or use the function from tasks above to send an integer output to PORTB on the OUSB board. Print the new value of PORTB to the screen (and check that it is correct!). Make sure you check that the output is within the range that can be accepted by PORTB – what happens if the value written to PORTB is greater than 256? Consider using command line arguments, as well as asking the user for input via the `cin` input stream class to take in one integer argument, then output this data to PORTB. You may want to write this as 2 separate programs or call 2 different functions from main to handle each scenario.

   [2.5 marks]

6. The final task is to connect, read data, and write data to the OUSB board by writing a generic function that can pass any valid OUSB command to the OUSB board. You code should be able to do everything from tasks 1 through to 5 via a single function.

   [2.5 marks]

Once you have finished these tasks you are free to move on to working on the Week 3 Lab Tasks; however, please note that we will be covering file input/output in the Week 2 lecture, which is directly related to the Week 3 Lab Tasks. You should not leave the laboratory after completing the tasks on this sheet. There is certainly enough work to continue on with for later tasks.