Computing for Engineers (EE108FZ)
Assignment 1 – functions

Objectives: This assignment aims to further develop the use of functions in C

Learning outcomes: same as for lab

Instructions

1. The assignment is for individuals working independently and builds on the lab exercises;
2. Create new sketches as required, naming them as specified in each section;
3. Create a single plain text submission file (.txt) for the assignment using a plain text editor (e.g., NotePad++). Copy all the sketches you write and any other answers required for the assignment into the submission text file. *Name the file "108_A1_firstname_surname.txt", using your actual name. Put your name and assignment number at the top of the submission file also. Clearly label everything in the file. Submissions without names or clearly labelled sections will have to be marked down or (in the worst case) not marked at all.*
4. Upload your submission file to relevant Moodle section.

Marking

For all code sections, marks will be deducted for bad structure, communication and style (e.g., repetitive or highly inefficient code, inappropriate global variables, missing or mismatching comments, poor variable names, bad indentation, etc.), incorrect behaviour, and failure to compile.

General marks will also be lost if the submission document requirements are not followed

NOTE: plagiarism and collusion will not be tolerated. You may be interviewed in detail on any work submitted and any evidence that the work is not your own will have the appropriate consequences.

1 Bits to Bar LEDs – save sketch as "A1_BitsToBarLeds"

Background: There are 10 Bar LEDs on the daughterboard and 16 bits in an unsigned integer. Therefore we can use the value (0/1) of each of the least significant 10 bits in an unsigned integer to specify the individual on/off state of each of the 10 Bar LEDs at once. Let's call this unsigned integer `ledStates` and consider the following example:

- Assume that `ledStates` has the hex value `0x0185` which corresponds to binary `0000 0001 1000 0101`.
- We would interpret this to mean that all Bar LEDs should be off except for the LEDs where the corresponding bits are 1.
- Bit 0 always refers to the least significant bit (on the right as written above). Therefore, examining the value of ledStates in this example, all bar LEDs should be off except for the bar LEDs corresponding to those bits that are 1, i.e. bit 0, bit 2, bit 7, and bit 8. The corresponding bar LEDs to switch would be bar LEDs 1, 3, 8, and 9 respectively (since bit 0 of ledStates specifies the on/off state of Bar LED 1 and so on for subsequent bits in ledStates).

Remember that the Arduino `bitRead(x, n)` function allows you to get the value of bit number `n` from value `x`. Note also that it is possible to iterate over all the bits in a number simply by incrementing the argument that you pass in for `n` within a loop.

Sketch requirements:

- Each time the `loop` function runs execute the following steps
  - light Bar LEDs 1 and 2 only for 300ms and print the binary ledStates value that corresponds to these LEDs
  - light Bar LEDs 1, 3, 5, 6, 7, 8, and 10 only for 600ms and print the binary ledStates value that corresponds to these LEDs
  - Blink Bar LEDs 9 and 10 on and off twice, on for 50 ms, off for 250 ms and print the binary ledStates value that corresponds to these LEDs
  - Choose a completely random set of Bar LEDs (different each time the loop function runs), and blink those LEDs on and off 5 times, on for 10 ms, off for 140 ms. Print the binary ledStates value that corresponds to these LEDs. (Note that a random set of bar LEDs means as few as one and as many as 10 LEDs might be lighting).
- The output of each execution of the loop function (printing the 10 least significant bits of ledStates) should be something like:

```
---------
ledStates: 0 0 0 0 0 0 0 0 1 1
ledStates: 0 0 0 0 1 1 1 1 0 0
ledStates: 1 0 0 1 0 0 0 0 0 0
ledStates: 1 0 1 1 1 1 0 0 0 1
---------
…
```

- **You must use functions** that you declare and define to minimize code repetition. **You must not have a different function for each different set of leds** as that would not reduce code repetition and wouldn't deal with random LEDs. (For information: my model solution makes use of 3 functions). The functions should be focused on doing one thing well so don't mix

printing and lighting LEDs in a single function for example. Note that functions can call other functions and this can also reduce code repetition. Finally note that your functions should be flexible – E.g. do not hard code LED numbers, repetition counts, on or off times inside your functions – these kind of values should always be passed as parameters.

Copy the sketch into your answer document.

2 Reflection

Q1: What was the most important thing you learned in assignment 1?

Q2: What was the hardest aspect of this assignment and why?