ELEC 335 - Lab #1

The objective of this lab is to get familiarized with the development board, understand all the ICs and their connections with the microprocessor. Practice assembly language, and write basic programs. Work with the debugging tools for analysing program flow.

Problem 0. Download and install Keil with G0 pack, create a new project and add the assembly code given here https://gist.github.com/fcayci/66d8d260ae67ca08c140e075905698fb. The steps are also explained here: https://youtu.be/DgIrZvg_KLY
Note 1: This is just so that you can run your code in the simulator and see the register contents to make sure everything is working as expected.
Note 2: If you have the board, and if you want, you can do the lab using STM32CubeIDE with the board plugged in - and even implement problems on the breadboard- For that, use the g0 project repo and follow https://youtu.be/UYk_NAnDJlw

Problem 1. In the Nucleo G031K8 board, identify all the ICs and explain their usage. Also explain all the connected peripherals and their pin connections with the microcontroller.

Problem 2. Implement assembly code that will light up 1 LED connected to pin PA8. For this problem (and lab) you do not need to worry about clocking the peripherals (and that is fine if you don't even know what that means).

- Just assume an LED is connected to the given port/pin. A represents the port GPIOA, and 8 represents the pin 8.
- Get the port address from RM0444. Look for GPIOA base address from the memory map table (Section 2) Then find the offset for ODR register in GPIO peripheral (Section 6).
- To turn on an LED, just write a 1 to the corresponding bit location in the ODR register memory address.
    - For example, to turn on an LED on PE5, you need to write 1 to the GPIOE ODR register bit 5.
- Remember in Keil, defining a word is different from GNU assembly. Check the lecture slides (or book).

Problem 3. Implement assembly code that will light up 4 LEDs connected to pins PA11, PA12, PB4, PB5.

Problem 4. Implement a delay routine and have 1 LED described on Problem 2 toggle in roughly 1 second intervals. For this assume the clock is running at 16 Mhz. (It will be impossible to single step through 1 second delay routine, so utilize breakpoints)