ELEC 335 - Lab #6

Objective

The objective of this lab is to

- practice C language, and write basic programs,
- work with analog to digital converters,
- work with pulse width modulation,
- work with serial communication protocols.

Setup

Install STM32CubeIDE and implement problems on the breadboard. For that, use the **`blinky`** project from the g0 project repo (`https://github.com/fcayci/stm32g0`) and follow `https://youtu.be/UYk_NAnDJlw`. Do not use **standalone** for any problem / lab. That is just for demonstration purposes. You can use one of the undergraduate laboratories for using the scope and measuring the necessary voltages if needed.

- **For each coding problem**, you are required to create a flowchart that shows software operation and a block/connection diagram that shows how things are connected and what modules are in use. You can use online tools to do that. One example is lucidchart (`https://lucid.app`)
- **All codes should be written in C.**
- If a specific pin number is not stated, you are free to pick any appropriate pin to connect your components. Make sure to include this in your block/connection diagram.

Submission

You should submit the following items organized in a folder:

- **Lab report** - Written in English. PDF file. Should include
  - a cover page
  - for each problem
    - a flow chart
    - a block/connection diagram
    - pictures/photos if any/requested
    - code listing
    - at least one paragraph explanation/comment about that problem, code listing
  - final lab conclusion about what you've learned.
- **Source files** - should have proper comments and pin connections if there are any external components.
- **Elf files** - generated elf file with debug information.

Problems

Problem 1 [50 pts]. In this problem, you will be working on implementing **a signal follower**. Attach a signal to one of the pins, capture its value and replay it back using PWM. You should see the original signal back on the oscilloscope.

- You can use a function generator to send the signal.
- The signal frequency depends on how fast you can go.

Problem 2 [50 pts]. In this problem, you will be working with reading and logging MPU6050 IMU sensor data utilizing Timer, I2C, and UART modules and using MPU6050, and 24LC512 EEPROM.

- Write your I2C routines to read / write multiple data. You should have four functions: single read, single write, multi read, multi write. Multi read and write deal with multiple bytes.
- Write a data structure to hold the sensor data.
- To ensure the data is correct, read all sensor data and send them all over UART to PC as the example below:

  ```
  AX: 0.12, AY: 0.53, ..., GX: 1.32, ...
  ```

- Sample the sensors every 10 ms, and write the data values to EEPROM. You should first start with writing and reading single bytes. Once the operation is completed successfully, work on your way to write and read multiple bytes.
- EEPROM and MPU6050 should be sharing the same I2C bus. Check the IMU board for pull-up resistors. If it includes pull-up resistors, you should not need to add another set of pull-up resistors.
- Once you press an external button, data collection should start, and once it collects 10 seconds of data, it should stop, and an LED should light up to display data that is ready on EEPROM.
- When the LED is on (meaning there is data on the EEPROM) pressing the button will transmit all the data over UART to your PC.
- You save this to a file and plot your results using Python or MATLAB.