ELEC335 Lab6

December 23, 2020

The objective of this lab is to get you communicate with MCU from PC, and utilize bidirectional data transmission and parsing.
You will use C language for the problems unless some parts require *inline assembly*. Use *blinky* project from stm32g0 repo as the starting point for your problems.

Submission

You should submit the following items organized in a folder:

- **Lab report** - Written in English. Proper cover page, intro, problems, flow charts, block diagrams, schematic diagrams, comments, and theoretical, mathematical work, simulation vs.
- **Source files** - should have proper comments and pin connections if there are any external components.
- **Video link** - A link to video demonstration of your lab (this video also should show yourself.)
- **Elf files** - generated elf file with **debug information**.

Compress the folder to a zip file, and submit that way. Each problem should be in a separate folder. Example folder structure is given below.

```
name_lastname_number_labX/
  name_lastname_report_labX.pdf
  video_link.txt
  problem1/
    problem1.s/c
    problem1.elf
    ...
  problem2/
    problem2.s/c
    problem2.elf
    ...
```

Problems

Problem 1

In this problem, you will work on implementing a simple tone generator utilizing Timer, PWM and External Interrupt modules and use a keypad, a speaker, and 7SDs.

- Connect a keypad to your microcontroller, pick a block of notes, and assign a tone for each key on the keypad.

    - One of the keys should be silence (rest).
- Tones will be played using PWM by changing the period at 50% duty cycle.
- Design an amplifier, and connect a speaker with adjustable gain using a pot.
- When a key is pressed, the relevant tone should play indefinitely.
- Connect your 7SD to display the tone that is being played. You can display the frequency being played. (440, 480, ...) OR you can display the tone being played (A4, B4, C4, ...).

Problem 2

In this problem, you will be working with reading and logging MPU6050 IMU sensor data utilizing Timer, I2C, and UART modules and use 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, send your results over UART to PC with You should read all sensor data and send them all as the example below:

```
1  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 resisitors. 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 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 can optionally save this to a file and/or plot your results using Python or Matlab.