ECEN 5823
Assignment 2 - Managing Energy Mode Assignment
Fall 2020

Objective: Become familiar with the Silicon Labs' Simplicity development system as well as learn the different Blue Gecko energy modes and how to manage these energy modes.

Note: This assignment will begin with the completed Assignment #1 - Simplicity Studio Exercise Assignment.

Instructions:

1. Start by creating your assignment repository using the assignment link at https://classroom.github.com/a/meH-HlcT . You will not need to clone this from the GitHub repo you just created into a new local directory since you will be starting with your code from the previous assignment. Instead, follow these instructions and run these commands with git bash within your assignment directory. **It is recommended that you close Simplicity Studio before performing the git merge operations listed in the instructions above.** This will help avoid problems as the project file will be updated during the merge. Ensure you are setup to use SSH keys before proceeding, see the instructions from github at https://help.github.com/articles/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent/ and be sure to use Git Bash For Windows or Cygwin if using a Windows environment. (Cygwin .ssh directory will be different than Git Bash.)

    a. On your local PC, duplicate the folder created in Assignment #1 to a new folder, name it something like ecen5823-assignment2-<username> where <username> is your GitHub username.
    b. Change your current directory to this new folder ecen5823-assignment2-<username> and execute the following git commands.
    c. git remote remove origin
    d. git remote add origin <url>
        i. Where <url> is the URL for the repository created with the link above
        ii. This adds the new submission repository created in the link above as your new origin.
    e. git remote add assignments-base https://github.com/CU-ECEN-5823/ecen5823-f20-assignments.git
        i. ecen5823-f20-assignments is a repo that contains branches with new files for each assignment
    f. git fetch assignments-base

    g. git merge assignments-base/assignment2
        i. This merges in new files for the current assignment into your local directory on your PC
    h. git push origin master
        i. This pushes your local code to the master branch of the newly created repository, including previous assignment source and questions merged from assignments-base to your working repository for this assignment. After running these commands you should see the Assignment2-ManagingEnergyModes.md file in the questions directory within your ecen5823-assignment2-<username> repository and you should see your code from the first assignment in this repository as well.
    i. Import ecen5823-assignment2-<username> into Simplicity Studio
        i. Remember to perform a **Clean…** immediately after you import the project into Simplicity Studio.
2. Develop timer code to configure a periodic timer with interrupt. This timer will be used to turn an LED on/off with a fixed period, configured via #define macro, and will operate in two possible modes, a low power mode (used with EM3 sleep) and a higher power mode (used with EM1/EM2 sleep states).
    a. The timer code should use LETIMER0
    b. LETIMER0 should use one of two possible oscillators, depending on configuration
        i. In low power mode (EM3) the ULFRCO (1 KHz) should be used.
        ii. In higher power modes (EM1/EM2) the LFXO (38.4 KHz) should be used.
    c. The default setting for the interrupt handler timing should configure the interrupt for an LED0 175ms on time with a period of 2.25 seconds. Use this setting for your measurements in the questions file.
    d. Two #define statements should exist in the timer code to set the LED on time and total blink period in milliseconds. The #define values should be configurable, and should allow for settings of up to 7 seconds for the blink period and up to 1 second for on time.
3. Use the EMLIB SLEEP functions to sleep the system at specific power levels based on a single const value set at compile time within your appMain() function in main.c, to allow you to change your code easily and test/profile different power levels. For instance:
    const SLEEP_EnergyMode_t sleep_mode_blocked=sleepEM4;
    would result in your code using sleep mode EM3 within the main loop.
4. Answer the questions within the questions directory for assignment 2 (Assignment2-ManagingEnergyModes.md) and include with your submission.

Deliverables:

- Submission via **github classroom** and your repository setup via the classroom link at https://classroom.github.com/a/meH-HlcT . Use the following git command to submit your assignment:
  git push origin master

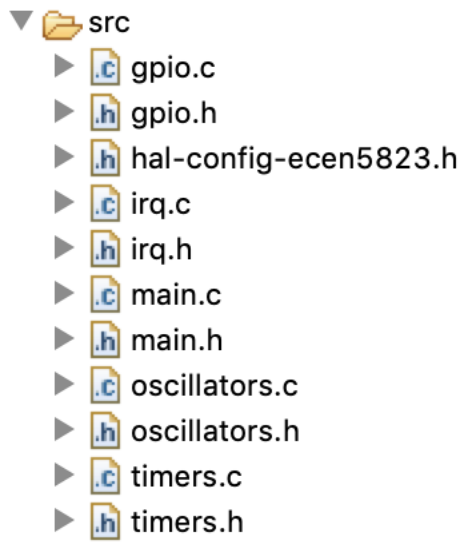
- Verify your ecen5823-assignment2-<username> repository contains your latest code and question responses before the deadline.
- In **Canvas**, submit the URL to your github repository to complete your assignment submission

Approach/Guidance:

Develop code that uses LETIMER0 to generate periodic interrupts. Place the MCU into energy modes EM0, EM1, EM2 and EM3 in the main while (1) loop while waiting for the timer interrupt(s). You shall use the IDE's Energy Profiler to measure current in each of the 4 energy modes.

1. Create a #define label that will set the lowest energy mode allowed, example: #define LOWEST_ENERGY_MODE *sleepEM3*
2. Create a #define that controls whether any sleep modes are enabled at all, example:
    a. #define ENABLE_SLEEPING 1
    b. if ENABLE_SLEEPING is not defined, call __WFI() (wait for interrupt) instead
    c. This may help you debug your interrupt code without having to be concerned about the sleep modes. Once you have your program working, ENABLE_SLEEPING = 1 should be defined when profiling the energy modes.
3. Create all of your top-level defines in a new file main.h and include that in main.c. Create a new pair of files to configure oscillators called: oscillators.c/.h. Create a new pair of files to configure timers called: timers.c/.h. Create a pair of files hold all of the code for interrupts: irq.c/.h.

Files in your src/ directory should look like:



Order of development:

1. Develop code to configure oscillators, clock trees and get the timer running without interrupts. After you believe you have the timer running, write some test code that reads the timer value to confirm it is running. Test both oscillators.
2. Develop your IRQ service routine for LETIMER0 and configure the timer to generate interrupts. Test that code using both oscillators.
3. Lastly, develop the code that places the MCU into each of the 4 energy states, once for each energy mode per compile.
4. You will run your code 4 times, once in each energy mode and make current measurements using the Energy Profiler.