ME4405 - Fundamentals of Mechatronics (Spring 2020)

Lab Assignment Six
Data Acquisition on the MSP432

Due Thursday, March 5th, 2020

Objective: This purpose of this lab is to learn how to use onboard flash memory to save data acquired through analog to digital conversion or other means. A secondary objective is to learn how to print binary data in ASCII format using the sprintf() function.

Deliverables and Grading:

To get credit for this lab assignment you must:

1. Demonstrate proper operation of your code to the TA or instructor during lab or by the end of Wednesday office hours. **(50 points)**
2. Submit the commented final version of your code on Canvas (main.c plus any header or additional files needed to run your code). **(Pass/Fail)**
3. Lab quiz will be held Friday, March 6th. **(10 pts)**

Setup:

This lab requires Code Composer Studio, the MSP432, and the temperature sensor circuit built in Lab 1. The lab uses onboard features of the device such as flash memory, ADC, and UART. The MSP432 has 256KB of flash main memory and 16KB of flash information memory. This lab makes use of main memory to store data acquired by the ADC. Sector number 31 of main memory is chosen arbitrarily to store data.

Problem Statement:

The goal of this lab is to use the microcontroller as a data acquisition device. The program you write will have two modes: data acquisition mode, and data output mode. In data acquisition mode, the program will collect a total of 30 data samples from the ADC at 1 sec intervals and store it in flash memory. In data output mode, the program will print out the stored data from flash memory into the putty terminal window.

Background:

Flash Memory:

The memory space on the MSP432 is a 4GB address space divided into 512 MB zones. The first 512MB zone is called the Code zone. The Code zone contains the ROM, SRAM and Flash memory

regions. The flash memory is a 4MB region that starts at address 0x0000_0000, i.e. at the beginning of the entire memory region. This flash memory is then divided into Main Memory, Information Memory and Reserved Memory. The main memory starts at address 0x0000_0000. It is a 256KB region that holds the code and data for user applications. The information memory is a 16KB region starting at 0x0020_0000 which is mainly reserved for use by the device. For more details on the memory map and different memory zones, please read the device datasheet and slas826a.pdf (product overview) available on TI website.

Main flash memory is further divided into two parts, called Banks, each of 128KB. Both banks are further subdivided into 4KB memory spaces or sectors. So main flash memory contains 64 sectors divided into 2 banks, with 32 sectors in each part. These sectors are numbered 0-31.

Each sector in main flash memory can be individually programmed and erased. Driverlib API's are available for programming and erasing flash memory sectors, as discussed in the lecture on flash memory. The output of these API's are true if the operation is successful and false if the operation fails. If the operation fails, it is a good practice to trap the microcontroller (after reporting the error) in an infinite loop which does nothing. It is rare that failure occurs but nevertheless this is good practice.

Sectors must first be unprotected in order to be erased and programmed. Likewise, sectors should be protected once they are programmed. See the class lecture on flash memory for more details.

Hardware:

The circuit assembled/built in Lab 1 should be interfaced with the MSP432 similar to the setup in Lab 5. Adjust the potentiometer used to tune the gain of the op-amp to set the gain such that the **output voltage is always less than 2.5V for the full temperature range**. Connect the output of the temperature sensor signal conditioning circuit to the pin corresponding to the ADC module used. Furthermore, **connect a ground wire from the temperature sensing circuit to the MSP432**.

Software:

Your program should proceed in the following steps:

1. First, you should wait in a loop until the user presses one of the buttons. Pressing button S1 should place the system in data acquisition mode, while pressing S2 should place the system in data output mode.
2. If data acquisition mode is selected, the MCU should collect ADC measurements from the temperature sensor at a rate of 1 Hz (as in Lab 5). It should store the data in a standard array of floating point values. After it has collected a total of 30 measurements, it should load (program) the collected data into flash memory. Use main flash memory, bank 1, sector 31. After programming is complete, the program should then enter an infinite loop which does nothing, and an LED can light up.

3. If the user chooses data output mode, the data from main flash memory sector 31 should be printed to the putty terminal via the UART interface from Lab 4. Use the sprintf() function to convert the stored binary numbers into ASCII values. When you print the data, print the temperature as a floating point number with one decimal place (this can be achieved using the format descriptor "%.1f"). After you print the data, enter an infinite while loop that does nothing.

Note: If you reprogram the board between acquiring the data and printing it out, all data in flash memory will be lost (since the programmer erases it). **Instead, use the RST button on the board (S3) to reboot it when you want to switch modes.**

Helpful Tips:

- Your clock frequency affects two things: the parameters you use to set up your baud rate, and the interrupt frequency for Timer_A. If you choose to use a single clock frequency, be sure to recalculate what the parameters need to be; if you prefer to change clock frequencies, you can switch DCO and SMCLK frequency at any time. If you choose to use two different frequencies (for the two different modes), make sure to set up the clocks after the user selects the mode.
- Make sure to enable the floating point module using FPU_enableModule() at the beginning of your main loop along with halting the watchdog timer.
- Place the ADC in Manual Iteration mode (rather than Automatic Iteration mode) and place the toggle conversion function inside the timer ISR, as in Lab 5.
- After you reach 30 samples and are ready to write to flash memory, disable all interrupts to avoid the ADC continuing to sample.
- Store your data in flash memory as an array of floating-point numbers.
- The memory address of main flash memory, bank 1, sector 31 is 0x0003F000. Use a macro to define this address as follows:

  ```
  #define MAIN_1_SECTOR_31    0x0003F000
  ```

- The only interrupt you should use in this program is the Timer A capture compare interrupt, as in the previous ADC lab.

Requirements:

1. Successfully demonstrate the outcome of the program and all the required functionality to the Instructor or the TA.
2. Submit the commented final version of the code on Canvas.

Note: **There is no report for this lab**. You will earn points based solely on the functionality of your program and hardware.