Homework 3 Report

EEL 4742C

T TR 9:00-10:15

Group Members: Justin Bates, Kelsey Cameron

**Objective:** Write a program that asynchronously takes commands from UART interface and respond to them. Periodically wake up the microcontroller to perform a temperature read and record in a buffer including a timestamp. This is to occur every 5 minutes. The program must also show the current time of the system, be able to set the current time of the system, show the oldest temperature reading and timestamp. For extra credit pressing down the microcontroller button will also trigger the command that shows the current system time followed by the temperature readings and their timestamps.

**Procedure:** The first step in completing homework 3 was to solder the 32.768kHz crystal to the MSP430 Launchpad. The crystal enables very accurate time keeping through the chip, and using a timer, a Real-Time Clock (RTC) can be created. After checking the frequency of the crystal with an oscilloscope with the provided code we began programing the assignment in C.

The UART code to trigger interrupts whenever a character was received was used verbatim from homework 2.

To configure the Analog to Digital Converter (ADC) which enables on-chip temperature readings, we took several sample readings from the temperature sensor, then used the voltage temperature constants to subtract the different voltages levels from the reading. Then, when the temperature interrupt is triggered, the data is stored in a global variable called temp, which is then used later to store and display the temperature readings.

For printing temperature readings, and time stamps to a UART interface we needed to convert the hexadecimal value received on the chip to the appropriate ASCII character value that would show up to the user. Since we printed the time in the format of ‘hh:mm:ss’ and there were two digits for hours, minutes, and seconds we created a convert to character function. With the parameter ‘num’ our convert to character function divides the first digit by 10 and using modulus 10 to get the second digit. This technique was used both for displaying any timestamp in this program. In order to set the time for the system the ‘s’ key needs to be pressed through the UART terminal. Once this triggers the user will enter the 6 time digits. And the six digits will follow the technique previously described. If an ‘o’ is entered the oldest temperature reading is shown. This is done by iterating through the 32 saved readings (32 being the maximum amount of saved readings) and finding the one with the smallest timestamp. A similar function is used for the display all temperature readings and timestamps except all of the 32 iterations are printed to the UART terminal. (When ‘l’ is pressed)

A series of if statements were used to count up seconds and minutes to find the multiples of five, enabling the program to know when a temperature reading should be grabbed and posted to the terminal for the user to see.

**Conclusion:** In conclusion, this assignment helped us further understand our use of interrupts, and the implementation of printing data to a UART terminal. We also got experience soldering a peripheral onto our chip which will surely be good practice for senior design. Lastly, we learned a bit about the usage of the Analog to Digital Converter to do things like access the on-chip temperature sensor. This assignment was a good opportunity to test the skills we have learned so far in class, in the lab, and on our own.