Final Project

1. Introduction

- Microcontroller Board: Texas Instruments MSP430FR6989 LaunchPad
- This project will combine the knowledge of all the mini projects, sensor/adc, and LCD.
  _Button and LED
  _UART
  _ Various Interrupts
  _ LCD
  _ADC

2. Requirements:

Please use interrupts to realize the following functions:
1) The LCD displays the temperature in oF when button 1 is pressed.
2) The LCD displays the temperature in oC when button 2 is pressed.
3) Create an array of size 10 in FIFO structure to store the temperatures in oC sampled during past 10 seconds (one per second).
4) When computer sends a command (You can design the command as you like) through UART to the board, the cpu on the board will calculate the average of the 10 temperatures in oC collected in the past 10 seconds). The board will then send back the result to your computer.

If you finish the four functions correctly, your group will get full marks for this project.
If you could implement any further function to the system such as follows, you will get bonus.
Bonus tasks: (Will give your some examples soon…)
1) task scheduling:
2) ..
3)..

Make a video of your demo and merge it with your code into a zipped folder for submission.
Name this folder as "Firstname1_Firstname2_final".

Please first try to understand the two examples given at the end: one uses the LCD to display letters; one samples A10 Temperature and Convert it to oC and oF. (Will upload either a video or more explanation about the two examples)

(1) LCD example:

```
#include "msp430.h"

const unsigned char lcd_num[10] = {
    0xFC,        // 0
    0x60,        // 1
    0xDB,        // 2
    0xF3,        // 3
    0x67,        // 4
    0xB7,        // 5
    0xBF,        // 6
    0xE4,        // 7
    0xFF,        // 8
    0xF7,        // 9
};

int main(void)
{
    WDTCTL = WDTPW | WDTHOLD;               // Stop WDT
    // Disable the GPIO power-on default high-impedance mode
    // to activate previously configured port settings
    PM5CTL0 &= ~LOCKLPM5;

    Ini_LCD();

    // Display "4" on Segment 36,37
    LCDM19 = lcd_num[4];

    // Display "3" on " on Segment 28,29
    LCDM15 = lcd_num[3];

    // Display "0" on " on Segment 14,15
    LCDM8 = lcd_num[0];

    //Turn LCD on
    LCDCCTL0 |= LCDON; //If you move this before CDM19 = lcd_num[4]; digits will
appear in order instead of appear all at the same time.
}

void Ini_LCD() {
           PJSEL0 = BIT4 | BIT5;                   // For LFXT

           // Initialize LCD segments 0 - 21; 26 - 43
           LCDCPCTL0 = 0xFFFF;
           LCDCPCTL1 = 0xFC3F;
           LCDCPCTL2 = 0x0FFF;


           // Configure LFXT 32kHz crystal
           CSCTL0_H = CSKEY >> 8;                  // Unlock CS registers
           CSCTL4 &= ~LFXTOFF;                     // Enable LFXT
           do
           {
             CSCTL5 &= ~LFXTOFFG;                  // Clear LFXT fault flag
             SFRIFG1 &= ~OFIFG;
           }while (SFRIFG1 & OFIFG);               // Test oscillator fault flag
           CSCTL0_H = 0;                           // Lock CS registers

           // Initialize LCD_C
           // ACLK, Divider = 1, Pre-divider = 16; 4-pin MUX
           LCDCCTL0 = LCDDIV__1 | LCDPRE__16 | LCD4MUX | LCDLP;

           // VLCD generated internally,
           // V2-V4 generated internally, v5 to ground
           // Set VLCD voltage to 2.60v
           // Enable charge pump and select internal reference for it
           LCDCVCTL = VLCD_1 | VLCDREF_0 | LCDCPEN;

           LCDCCPCTL = LCDCPCLKSYNC;               // Clock synchronization enabled
```

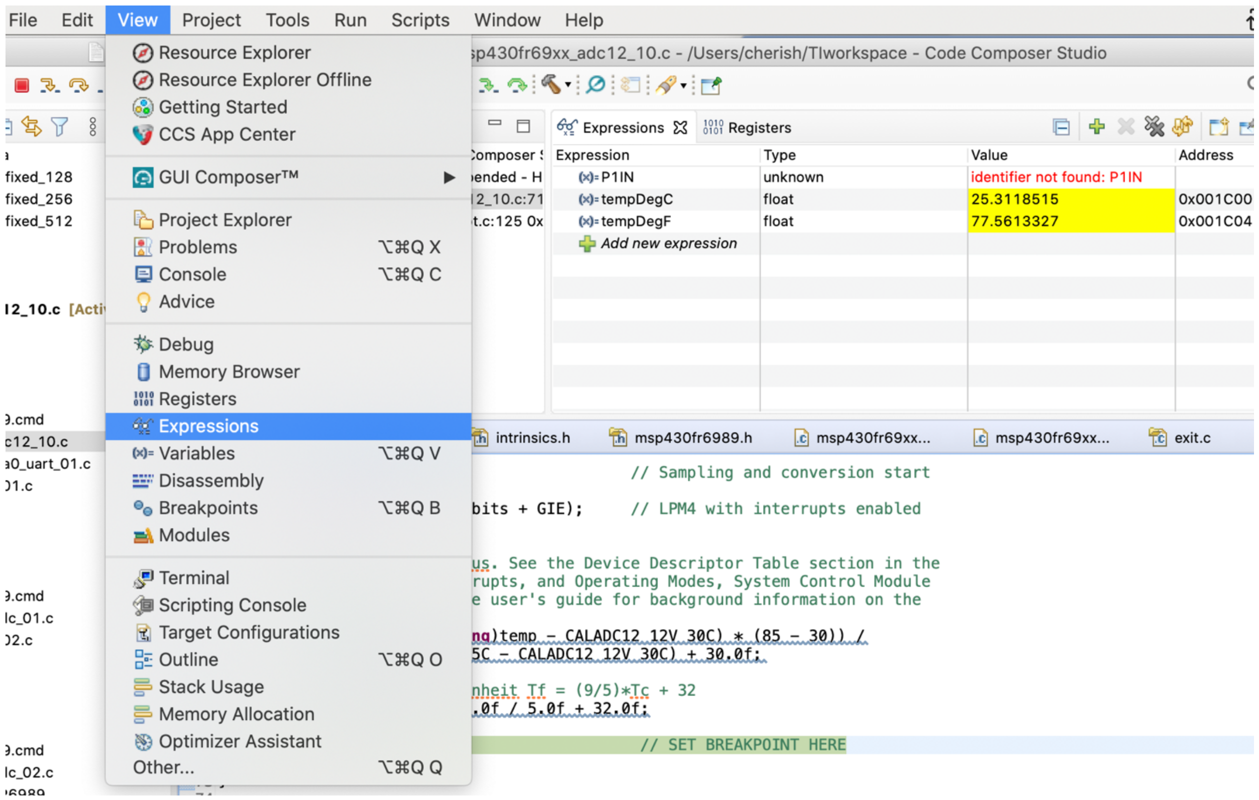
```
          LCDCMEMCTL = LCDCLRM;                   // Clear LCD memory


          return;
}
```

(2) ADC example with sensing temperature:

```
#include <msp430.h>

#define CALADC12_12V_30C  *((unsigned int *)0x1A1A)   // Temperature Sensor
Calibration-30 C
                                                      //See MSP430FR6989 datasheet
page 131 for TLV table memory mapping
#define CALADC12_12V_85C  *((unsigned int *)0x1A1C)   // Temperature Sensor
Calibration-85 C

unsigned int temp;
volatile float tempDegC;
volatile float tempDegF;

int main(void)
{
  WDTCTL = WDTPW + WDTHOLD;                   // Stop WDT

  // Initialize the shared reference module
  // By default, REFMSTR=1 => REFCTL is used to configure the internal reference
  while(REFCTL0 & REFGENBUSY);                // If ref generator busy, WAIT
  REFCTL0 |= REFVSEL_0 + REFON;               // Enable internal 1.2V reference

  /* Initialize ADC12_A */
  ADC12CTL0 &= ~ADC12ENC;                     // Disable ADC12 so that you could modify
the values in registers
  ADC12CTL0 = ADC12SHT0_8 + ADC12ON;          // Set sample time
  ADC12CTL1 = ADC12SHP;                       // Enable sample timer
  ADC12CTL3 = ADC12TCMAP;                     // Enable internal temperature sensor
  ADC12MCTL0 = ADC12VRSEL_1 + ADC12INCH_30;   // ADC input ch A30 => temp sense
  ADC12IER0 = 0x001;                          // ADC_IFG upon conv result-ADCMEMO

  while(!(REFCTL0 & REFGENRDY));              // Wait for reference generator
                                              // to settle
  ADC12CTL0 |= ADC12ENC;

  while(1)
  {
    ADC12CTL0 |= ADC12SC;                     // Sampling and conversion start

    __bis_SR_register(LPM0_bits + GIE);       // LPM4 with interrupts enabled
    __no_operation();

    // Temperature in Celsius. See the Device Descriptor Table section in the
    // System Resets, Interrupts, and Operating Modes, System Control Module
    // chapter in the device user's guide for background information on the
    // used formula.
    tempDegC = (float)(((long)temp - CALADC12_12V_30C) * (85 - 30)) /
            (CALADC12_12V_85C - CALADC12_12V_30C) + 30.0f;

    // Temperature in Fahrenheit Tf = (9/5)*Tc + 32
    tempDegF = tempDegC * 9.0f / 5.0f + 32.0f;
```

```
        __no_operation();                          // SET BREAKPOINT HERE
    }
}


#pragma vector=ADC12_VECTOR
__interrupt void ADC12ISR (void)
{
  if(ADC12IV & ADC12IV_ADC12IFG0)
  {
    // Vector 12:  ADC12MEM0 Interrupt
      temp = ADC12MEM0;                          // Move results, IFG is cleared
      __bic_SR_register_on_exit(LPM4_bits); // Exit active CPU
  }
}
```

In this example, you can watch the values of temperature in expressions window during debugging by clicking View<<Expressions as shown in the following figure (Set a checkpoint after the temperature is calibrated)



Expression	Type	Value	Address
P1IN	unknown	identifier not found: P1IN	
tempDegC	float	25.3118515	0x001C00
tempDegF	float	77.5613327	0x001C04
Add new expression			

```
                    // Sampling and conversion start
bits + GIE);        // LPM4 with interrupts enabled

us. See the Device Descriptor Table section in the
rupts, and Operating Modes, System Control Module
e user's guide for background information on the

ng)temp - CALADC12_12V_30C) * (85 - 30)) /
5C - CALADC12_12V_30C) + 30.0f;

nheit Tf = (9/5)*Tc + 32
.0f / 5.0f + 32.0f;
                     // SET BREAKPOINT HERE
```