```c
/*
 * sample.c
 *
 *  Created on: 16-Jan-2023
 *      Author: Rajat Sankhla
 */

//----------------------------------------
// TivaWare Header Files
//----------------------------------------
        "myLib.h" // invoking user defined header files
        "lcd.h"   // invoking user defined header files(contains prototypes of user defined functions)


//---------------------------------------
// Globals
//---------------------------------------
    i;
    a;
    b;

uint32_t ui32ADC0Value[4]; // storing samples of ADC0
        uint32_t ui32TempAvg; // average of 4 samples(keeps on changing as ADC values changes hence qualifier volatile is added)
        uint32_t ui32TempValueC; // stores calculated value of temp in Celcius.
    digit[3];
//-------------------------------------------------------------------------
// main()
//-------------------------------------------------------------------------
    main(    )

{

    //.............................LCD MAIN.................................//
    lcd_init(); // clock is initialized inside this
    //lcd_cursor(1,1);
    //lcd_string("RAJ-DRU-JAN");
    //lcd_string("Temperature : ");
    //lcd_cursor(2,1);
    //lcd_string("IIT BOMBAY");
    lcd_cursor(1,1);
    lcd_string("Temperature :");
    //while(1); // while is used for first part of program

    //.............................ADC MAIN.................................//
    ADC_config(); // configuring ADC

        (1)
    {

        ADCIntClear(ADC0_BASE,1); // clears the flag which marks the end of sampling conversion

        ADCProcessorTrigger(ADC0_BASE,1); // Start the conversion of 4 samples to their digital equivalent also raise interrupt flag when conversion is complete
            (!ADCIntStatus(ADC0_BASE,1,false)); // waits until conversion is not complete
```

```
54          ADCSequenceDataGet(ADC0_BASE, 1, ui32ADC0Value); // copies the converted data from
  FIFO buffer to ui32ADC0Value
55
56          ui32TempAvg = (ui32ADC0Value[0] + ui32ADC0Value[1] + ui32ADC0Value[2] + ui32ADC0Value
  [3])/4;
57          ui32TempValueC = (1475 - ((2475 * ui32TempAvg)) / 4096)/10; // contains the analog
  value of temperature
58
59          lcd_print(1,14, ui32TempValueC);
60          lcd_char(" ");
61      }
62
63  }
64
65
66  //-------------------------------------------------------------------------
67
68       lcd_portconfig(     )
69  {
70      //Set CPU Clock to 40MHz. 400MHz PLL/2 = 200 DIV 5 = 40MHz
71      SysCtlClockSet(SYSCTL_SYSDIV_5|SYSCTL_USE_PLL|SYSCTL_XTAL_16MHZ|SYSCTL_OSC_MAIN);
72
73       //* * * * * * Setting Ouput for LCD * * * * * * * *
74        SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOB);
75        GPIOPinTypeGPIOOutput(GPIO_PORTB_BASE, 0xFF);
76
77        // * * * * * * LCD_Control Pin* * * * * * *
78        SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOC);
79        GPIOPinTypeGPIOOutput(GPIO_PORTC_BASE, 0x70);
80    }
81
82
83       lcd_init()
84  {
85      lcd_portconfig();
86
87      lcd_reset_RW(); //writing to the lcd
88      lcd_reset_EN(); // low at start of data transfer
89
90      // transferring of data starts
91
92      lcd_command(0x38); // function set command
93      lcd_command(0x0f); // display switch command
94      lcd_command(0x06); // increment cursor to the right after one character is sent
95      lcd_command(0x01); // clear screen command
96      lcd_command(0x80); // Set cursor to second line starting
97  }
98
99
100      lcd_command(     cmd)
101 {
102     lcd_put_data(cmd);
103     lcd_reset_RS(); // pulled low for command transfer
104     lcd_delay;
105     lcd_set_EN();// setting enable high means command transfer is complete
106     lcd_delay;
107     lcd_reset_EN(); //setting enable to zero for future command transfer
```

```
108     lcd_delay;
109 }
110
111 /*
112  *lcd_char()
113  *Description: Print single character
114  *Example lcd_char('A'); prints letter A
115  */
116
117      lcd_char(     data)
118 {
119     /* TASK 3 : Write the code here
120        Hint: Somehow Similar to the function lcd_command()*/
121
122     lcd_set_RS(); //enabling data transfer
123     lcd_delay;
124     lcd_reset_EN(); // pulling enable low for data transfer
125     lcd_delay;
126     // RW is already pulled low in lcd_init();
127     lcd_put_data(data); // putting the data
128     lcd_delay;
129     lcd_set_EN(); // marks the completion of data transfer
130     lcd_delay;
131
132
133 }
134
135 /*
136  * lcd_string()
137  * Description: Print string
138  * Example: lcd_string("Hello World");
139  */
140      lcd_string(     *name){
141
142              (*name){
143             // means while pointer is pointing to the string
144             lcd_char(*name); // keep sending the characters of the string one by one to the
    lcd_char(); function.
145             name++;
146         }
147         lcd_char(" ");
148     }
149
150 /*
151  * Name: lcd_cursor (row, column). For setting cursor position in 16 by 2 lcd
152  * Description: Position the LCD cursor at "row", "column"
153  * row: 1,2
154  * column: 1 to 16
155  * Example: lcd_cursor(2,14) - Places cursor at 2nd line 14th column
156 */
157
158      lcd_cursor (    row,    column)
159 {
160     /* TASK 4 : Write the code to set the cursor position*/
161
162
163         (row==1){
```

```
164
165             a = 0x80;
166             b = 0x80 + column-1;
167         }
168                 (row==2){
169             a = 0xC0;
170             b= 0xC0 + column-1;
171         }
172
173  lcd_command(a);
174  lcd_command(b);
175
176
177 }
178
179      lcd_integer(    integer){
180     //integer = ui32TempValueC;
181         r=0;
182         sizee;
183         q=integer;
184         i=0;
185         j;
186          (q!=0){
187         r=q%10;
188         digit[i]=r;
189         q=q/10;
190         i++;
191     }
192     sizee=i;
193
194        (j=sizee-1 ; j>=0 ; j--)
195      {
196         lcd_char(digit[j] + 0x30);
197      }
198
199 }
200
201 /*
202  * Name: lcd_print (value, digit). Print value (a numeric number).
203  * Description: Print number
204  * value: Numeric number
205  * digit: number of digits in number. Maximum allowed digit is 7
206  * Example: lcd_print(456,3) - Print 456 on LCD
207 */
208
209
210
211      lcd_print (     row,      column, uint32_t value)
212 {
213
214     /* row    : Input Cursor position for the row position
215        column : Input Cursor position for the column position
216        value  : Input integer value to be displayed on LCD
217        digits : Input number to depict the number of decimal places u want to show*/
218
219     // To set the LCD cursor position
220
```

```
221     lcd_cursor(row,column);
222
223     /* TASK 5 : Write the code to convert the integer number into ASCII format so
224        that can passed to the LCD for display */
225     /* You can declare your own variables also*/
226
227     lcd_integer(value);
228
229
230 }
231
232
233 //.....................................ADC configuration.....................//
234
235      ADC_config(){
236
237     SysCtlPeripheralEnable(SYSCTL_PERIPH_ADC0);
238     // ADC_TRIGGER_PROCESSOR is a trigger generated by the processor to start taking the
    samples. 0 is the highest priority given to sampling sequencer
239     ADCSequenceConfigure(ADC0_BASE,1,ADC_TRIGGER_PROCESSOR,0);
240
241     // we know that sample sequencer 1 will take 4 samples(steps) we neeed to configure each
    of these steps below
242
243     ADCSequenceStepConfigure(ADC0_BASE,1,0,ADC_CTL_TS); // configuring step 0 of sample
    sequencer to take sample from internal temp sensor.
244     ADCSequenceStepConfigure(ADC0_BASE,1,1,ADC_CTL_TS);//  configuring step 1 of sample
    sequencer to take sample from internal temp sensor.
245     ADCSequenceStepConfigure(ADC0_BASE,1,2,ADC_CTL_TS);//  configuring step 2 of sample
    sequencer to take sample from internal temp sensor.
246     // configuring the last step of sample sequencer to take sample from internal temp sensor
    and generate an interrupt when sampling is complete and mark the ending of sample sequencing
247     ADCSequenceStepConfigure(ADC0_BASE,1,3,ADC_CTL_TS|ADC_CTL_IE|ADC_CTL_END); // all three
    are flags which sets to high.
248
249     ADCSequenceEnable(ADC0_BASE,1); // Enabling the ADC
250
251 }
252
253
```