```
#include <msp430.h>

extern int IncrementVcore(void);
extern int DecrementVcore(void);

int avgVolt, avgTemp, sampTime, slope, ref30, ref85, calcTime;
int timer;
unsigned int refTemp30, refTemp85;

int count;
int i;
int time;
                      //          10        20        30        40        50
                      //012345678901234567890123456789012345678901234567890123456
unsigned char msg[] = {"000. The temperature is xx C. Running time is 0:00   "};


void main(void) {

msg[51] = 0x0a;
msg[52] = 0x0d;

UCSCTL6 = 0x0000; // default XTLCLK on

P7SEL |= BIT0 + BIT1;         //stabilizing crystal
while (UCSCTL7 & XT1LFOFFG) {
UCSCTL7 &= ~XT1LFOFFG;
}


IncrementVcore();
IncrementVcore();


UCSCTL1 = DCORSEL_5;    //
UCSCTL2 = 518;    //17MHz
UCSCTL3 = 0x0000; //XT1CLK
UCSCTL4 = SELM__DCOCLK + SELS__DCOCLKDIV + SELA__XT1CLK; //DCOCLK for MCLK


P11DIR = BIT2 + BIT1; //output at SMCLK and MCLK
P11SEL |= BIT2 + BIT1; //SMCLK and MCLK

TA0CTL = TASSEL__ACLK + TACLR; //  timer clear, ACLK source, clock on for Timer A0,
and continous mode
TA0CCTL0 |= CCIE;   //capture compare interrupt enabled
TA0CCR0 = 32768; // count to 1 sec

//set up rest of timer A
```

```
//button setup

P2DIR &= ~BIT6; //set button as input
P2SEL &= ~BIT6; //I/O function for button
P2IE |= BIT6;    //interrupt enabled
P2IES |= BIT6;  //triggers on falling edge
P2REN |= BIT6;  //pullup resistor for button
P2OUT |= BIT6;

//reference voltage set up


REFCTL0 = REFON + REFOUT + REFMSTR; //reference voltage set to 1.5 and set as output

//P5DIR |= BIT0;
P5SEL = BIT0;  //output reference voltage
P5OUT = BIT0;

//ADC setup
ADC12CTL0 |= ADC12ON + ADC12MSC + ADC12SHT0_12; //has multiple samples and
conversions in sequence mode, sample/hold time is 1024, and ADC12_A is on
ADC12CTL1 |= ADC12SHP + ADC12DIV_1 + ADC12SSEL_3 + ADC12CONSEQ_1; //SAMPCON sourced
from sampling timer, clock divided by 2, SMCLK clock is source, sequence-of-channels
mode
ADC12CTL2 |= ADC12REFOUT;   // reference output on (for the REFCTL0 set up earlier)
ADC12IE = ADC12IE7; //enable interrupt on 7 after the 8 measurements (0-7)

ADC12MCTL0 = ADC12SREF_1 + ADC12INCH_10;  //setting up memory control register 0 to
use Vref as reference and to read the temperature diode
ADC12MCTL1 = ADC12SREF_1 + ADC12INCH_10;
ADC12MCTL2 = ADC12SREF_1 + ADC12INCH_10;
ADC12MCTL3 = ADC12SREF_1 + ADC12INCH_10;
ADC12MCTL4 = ADC12SREF_1 + ADC12INCH_10;
ADC12MCTL5 = ADC12SREF_1 + ADC12INCH_10;
ADC12MCTL6 = ADC12SREF_1 + ADC12INCH_10;
ADC12MCTL7 = ADC12SREF_1 + ADC12INCH_10 + ADC12EOS; //EOS is end of sequence for
measurement 8

ADC12CTL0 |= ADC12ENC; //enable conversions

//temperature conversion and slope of voltage vs temp values

refTemp30 = *((unsigned int*)0x01A1A);        //extracting value for the reference
temp sensor at 30 degrees (TLV table in data sheet(pg 92))
refTemp85 = *((unsigned int*)0x01A1C);        //reference temp sensor at 85 degrees
ref30 = refTemp30;
ref85 = refTemp85;
slope = (float)(ref85 - ref30)/(85 - 30);     //slope of graph from difference of
voltage values over temp values.
```

```
//UART setup

UCA1CTL0 = UCPEN + UCPAR + UC7BIT;      //parity enabled, even parity, 7-bit data,
UCSPB optional 2 points

UCA1CTL1 |= UCSWRST;           //reset enabled
UCA1CTL1 |= UCSSEL_2;          //SMCLK source
UCA1MCTL |= UCOS16;           //over sampling

//set values for baud rate (same as lab 4)

UCA1BR0 = 27;  //first remainder or prescaler value
UCA1BR1 = 0;    // need to set to 0 cause no default value
UCA1MCTL |= UCBRF_11 + UCBRS_6; //following remainders
//UCA1IFG = 0;

UCA1CTL1 &= ~UCSWRST; //reset disabled


P5SEL |= BIT6 + BIT7;

TA0CTL |= MC__UP;

_EINT();

LPM0;

}

void buttonTimer(void)__interrupt[PORT2_VECTOR] {

if(P2IV == P2IV_P2IFG6){
timer ^= BIT0;

  count = count+1;  //counts how many times button is pressed
  msg[0]=count/100 +'0';  //set the digits corresponding to the count in msg to
count value
  msg[1]=(count%100)/10 + '0';      //middle one
  msg[2]=count%10 + '0';



 msg[46] = (time/60) + '0';  //minutes
msg[48] = (time%60)/10 + '0';  //tens of seconds
msg[49] = time%10 +'0';       //seconds

 UCA1TXBUF = msg[0];

ADC12CTL0 |= ADC12SC; //start sample and conversion
}
//P2IFG = 0;  //interrupt is not pending
}

void tempAverage(void)__interrupt[ADC12_VECTOR] {
//sampTime = TA0R; //capture sampling time
//TA0CTL |= TACLR;



if (ADC12IV == ADC12IV_ADC12IFG7){
avgVolt = (ADC12MEM0 + ADC12MEM1 + ADC12MEM2 + ADC12MEM3 + ADC12MEM4 + ADC12MEM5 +
ADC12MEM6 + ADC12MEM7)/8;//average of the 8 memory values
avgTemp = (avgVolt - refTemp30)/slope + 30; //converting to temp

  msg[24] = avgTemp/10 + '0';
 msg[25] = avgTemp%10 + '0';
 UCA1IE |= UCTXIE; //enable USCI_A1 TX interrupt


}
calcTime = TA0R;
}

void USCI_A1_ISR(void)__interrupt[USCI_A1_VECTOR] {

switch(UCA1IV) {
case 0: break;                //no interrupt
case 2:

  break;
case 4:                   // TXIFG

//while (!(UCA1IFG&UCTXIFG));     //RXIFG
         //check if TX buffer is ready then RX read character
   i = i+1;
 if (i<=52) {

  UCA1TXBUF = msg[i]; //prints array

  }
  else {
  i = 0;
  }


default: break;
}
}
```

```
void timerA(void)__interrupt[TIMER0_A0_VECTOR] {

time = time+1;
}
```