```
#include "msp.h"
#include <stdio.h>
#include <stdbool.h>
#include <stdint.h>
#include <stdlib.h>


void GPIO_Init(void){




    //Initialize P9.4 as output for potentiometer
      P9->SEL0 &= ~BIT4;
      P9->SEL1 &= ~BIT4;
      P9->DIR |= BIT4;//
      P9->OUT |= BIT4;//set as output high

      //set as analog input
        P5->SELC |= BIT1;

        P2->SEL0 |= BIT4;
        P2->SEL1 &= ~BIT4;
        P2->DIR &= ~BIT4;

        P2->SEL0 |= BIT5;
        P2->SEL1 &= ~BIT5;









}
```

```
//Global Variables
    uint16_t result;
    float Vin;
    float Vcm;
    float C1;
    float C2;




   void simple_clock_init(void)
   {
            //Set power level for the desired clock frequency
       while (PCM->CTL1 & 0x00000100) {;} //wait for PCM to become not busy
       uint32_t key_bits = 0x695A0000; //0x695A is the key code
       uint32_t AM_LDO_VCORE1_bits = 0x00000100; //AMR Active Mode Request - 01b =
AM_LDO_VCORE1
       PCM->CTL0 = key_bits | AM_LDO_VCORE1_bits; //unlock PCM register and set power
mode
       while (PCM->CTL1 & 0x00000100); //wait for PCM to become not busy
   PCM->CTL0 &= 0x0000FFFF; //lock PCM register again

           //Flash read wait state number change
       FLCTL->BANK0_RDCTL &= ~(BIT(12) | BIT(13) | BIT(14) | BIT(15)); //reset bits
       FLCTL->BANK0_RDCTL |= BIT(12); //bit 12~15: wait state selection. 0001b=1 wait states
       FLCTL->BANK1_RDCTL &= ~(BIT(12) | BIT(13) | BIT(14) | BIT(15)); //reset bits
       FLCTL->BANK1_RDCTL |= BIT(12); //bit 12~15: wait state selection. 0001b = 1 wait states

        //Clock source: DCO, nominal DCO frequency: 48MHz
       CS->KEY = 0x695A; //unlock CS registers
       CS->CTL0 |= BIT(18) | BIT(16); //bit 16~18 DCORSEL frequency range select 101b =
48Mhz
       CS->CTL0 |= BIT(23); //bit 23 - DCOEN, enables DCO oscillator

        //clock module that uses DCO: MCLK
       CS->CTL1 &= ~(BIT(16) | BIT(17) | BIT(18)); //source divider = 1
       CS->CTL1 &= ~(BIT0 | BIT1 | BIT2 ); //reset all bits
       CS->CTL1 |= 0x3; //bits 0 to 2 - SELM, selects MCLK source. 011b = DCOCLK (by default)
       CS->CLKEN |= BIT1; //bit 1 - MCLK_EN, enable MCLK (by default)

        //clock module that uses DCO: SMCLK
        //SMCLK source divider = 4 (12Mhz) (010b)
        CS->CTL1 &= ~(BIT(30) | BIT(28));
```

```
        CS->CTL1 |= BIT(29);
        //Select SMCLK source (011b)
        CS->CTL1 &= ~(BIT6);
        CS->CTL1 |= (BIT5 | BIT4);
        //bit 3 - SMCLK_EN, enable SMCLK
        CS->CLKEN |= BIT3;
        //lock CS registers
        CS->KEY = 0x0;

       while (!(CS->STAT & BIT(25))){} //while MCLK isn't steady
    }









void adc_init(void){

    ADC14->CTL0 &= ~BIT1;//ENC
        ADC14->CTL0 |= (1<<26);//sample and hold pulse source
        ADC14->CTL0 |= (1<<30);//predivder clck source
        ADC14->CTL0 |= (1<<19);//clck source select, MCLK
        ADC14->CTL0 |= (1<<20);
        ADC14->CTL0 |= BIT8;//0011b sample and hold pulse
        ADC14->CTL0 |= BIT9;
        ADC14->CTL0 |= BIT4;//adc on
        ADC14->MCTL[0] &= ~(0x1F); //reset to all zeros
        ADC14->MCTL[0] |= 1 << 2;//choose A4
        ADC14->CTL0 |= BIT1;//enable conversion

}



void timer_init(void){

    TIMER_A0->CTL |= BIT9;//select SMCLK



    TIMER_A0->CTL |= BIT4;//up-mode
```

```
    TIMER_A0->CCTL[1] |= BIT6;//PWM mode 2 for CCR1?
    TIMER_A0->CCTL[2] |= BIT6;//PWM mode 2 for CCR2?
    TIMER_A0->CCR[0] = 1200;//TA1CCR0 is the period of the timer, SMLCK is 12MHz from
simple clock init

}

//adc read function
void read(void){
    ADC14->CTL0 |= BIT0;
    while(ADC14->CTL0 & ADC14_CTL0_BUSY){

    }
    result=ADC14->MEM[0];


}

void convert(void){
    //result converted into a float
    Vin= result;
    Vin= (Vin/16384)*(3.3f);

}

void main(void){
    //call initialize function
    simple_clock_init();
    GPIO_Init();

    timer_init();
    adc_init();




    while(1){
        read();
        convert();
        Vcm=18.2f*(Vin-1.65f);
        C1=(Vcm*20.0f)+599.0f;
        C2=-1.0f*(Vcm*20.0f)+599.0f;
        TIMER_A0->CCR[1] = C1;
        TIMER_A0->CCR[2] = C2;
```

```
    }

}
```