EE447 Pre-Liminary Lab # 5

Question 1)

In this question we were tasked with supplying an analog signal and storing the signal in a digital form such as in a variable named $reading$.

In our code the hex code of 0xFFF represents the input analog signal of 0 volts while the maximum input voltage supplied which is 3.3 Volts is stored as hex code 0x000. The readings for the upper and lower bounds are shown in the diagrams below.

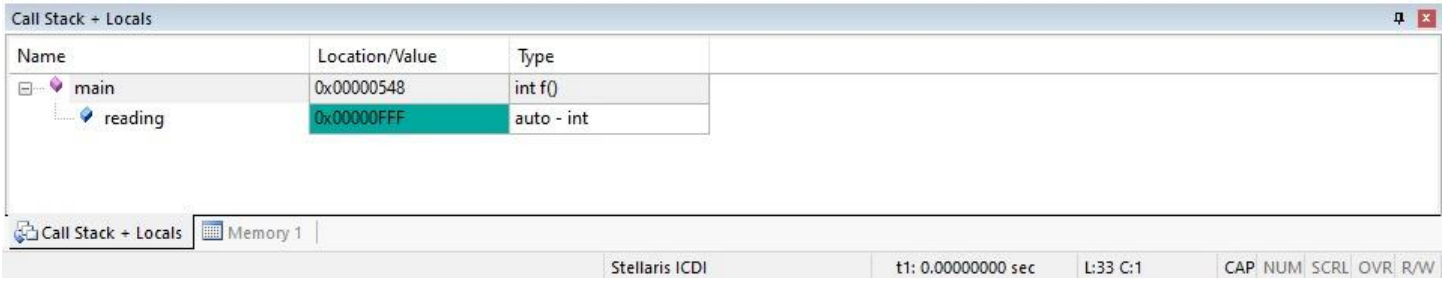

Fig 1.1 Input Analog = 0V, Output reading = 0xFFF

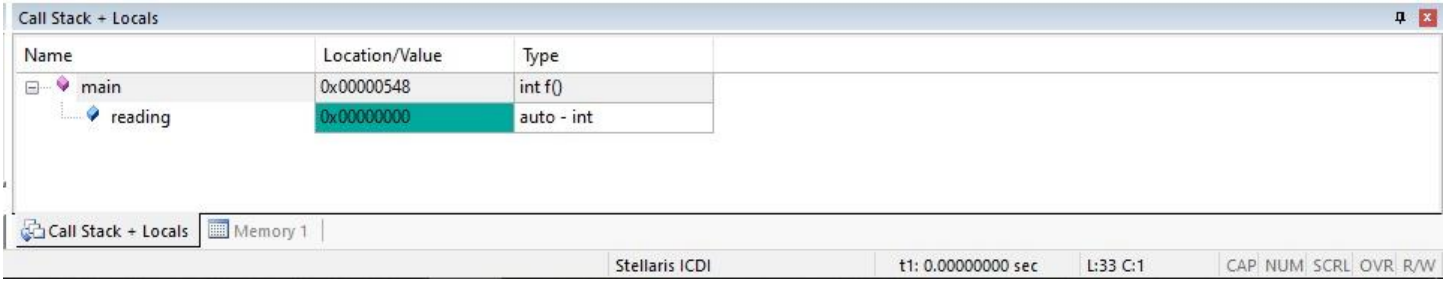

Fig 1.2 Input Analog = 3.3V, Output reading = 0x000

Question 2) After storing the analog values digitally in the reading variable, we will consider the fact that we have an offset of 1.65 DC Volts, therefore we will subtract the voltage reading in such a way that now the maximum input voltage which will be 1.64 V would be represented by 0x7FF and the minimum output voltage will be represented by 0xFFFFF800 which is a signed number corresponding to -1.64 in decimal.

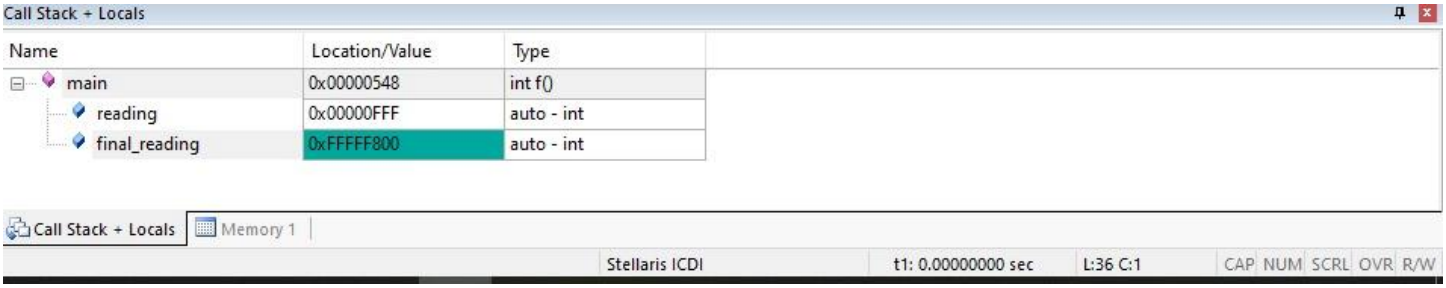

Fig 1.3 Input Analog = 0V, Analog Input After Offset = -1.65, Output reading = 0xFFFFF800

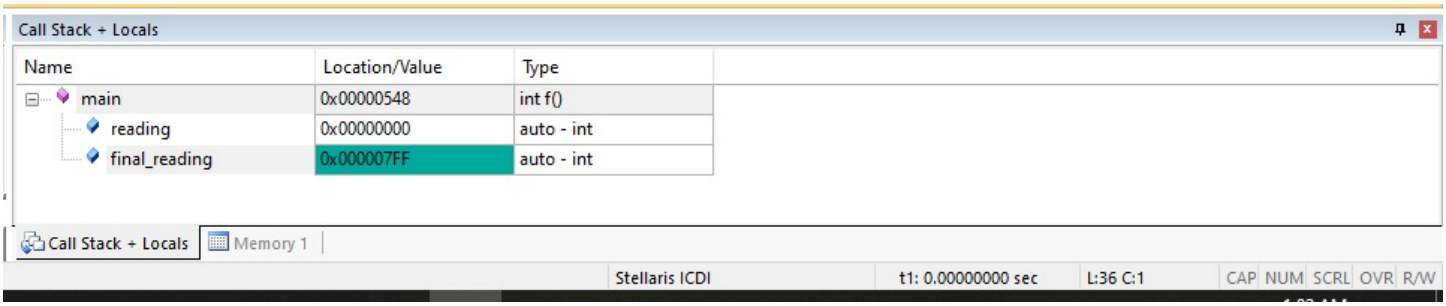

Fig 1.4 Input Analog = 3.3V, Analog Input After Offset = 1.65, Output reading = 0x7FF

Syed Muhammad Farrukh Aijaz-2417152 Mubeen Ahmad Fayyaz-2415842

Question 3 & 4)

After the DC offset value has been calculated we are now printing the values of the analog signal converted in to digital values within every second using a delay function and calling the ATD function continuously to print the values. The stored values for question 3 can be seen in the images below.

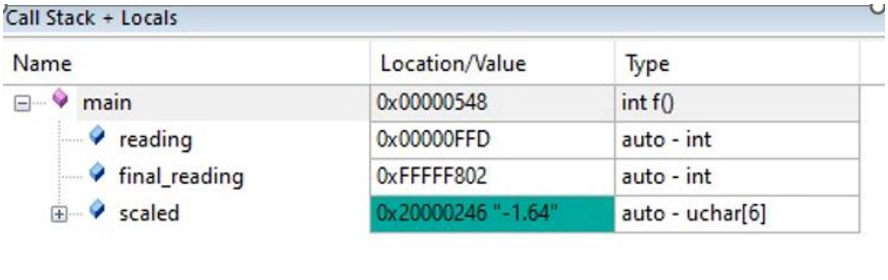

Fig 1.5 Input Analog = 0V, Analog Input After Offset = -1.65, Scaled reading = -1.64

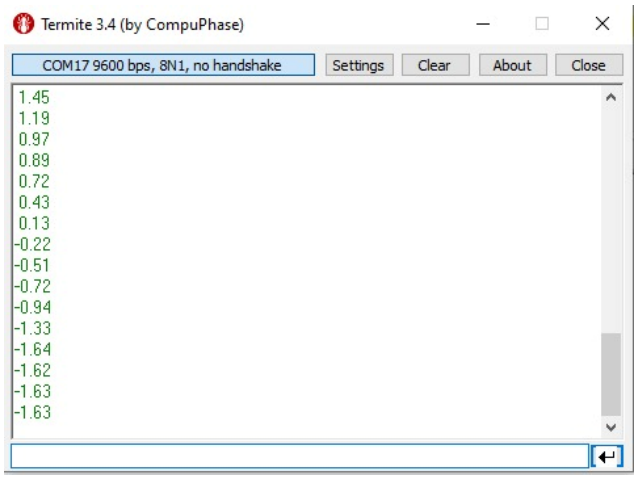

Fig 1.6 Termite Output display by varying potentiometer

Question 5) Screenshot of the codes are given below:

```
#include "TM4C123GH6PM.h"


#define SAMPLE_MASK 0xFFF
#define DCOFFSET 0x00

int high = 27;
int low = 18;
int state = 0;

void TIMER0A_Handler (void);
void pulse_init(void){
    volatile int *NVIC_EN0 = (volatile int*) 0xE000E100;
    volatile int *NVIC_PRI4 = (volatile int*) 0xE000E410;
    SYSCTL->RCGCGPIO |= 0x20; // turn on bus clock for GPIOF
    __ASM("NOP");
    __ASM("NOP");
    __ASM("NOP");

  GPIOF->DIR            |= 0x08; //set PF2 as output
```

```
  GPIOF->AFSEL      &= (0xFFFFFFFB);  // Regular port function
    GPIOF->PCTL         &= 0xFFFF0FFF;  // No alternate function
    GPIOF->AMSEL        =0; //Disable analog
    GPIOF->DEN          |=0x08; // Enable port digital



    SYSCTL->RCGCTIMER   |=0x01; // Start timer0
    __ASM("NOP");
    __ASM("NOP");
    __ASM("NOP");
    TIMER0->CTL         &=0xFFFFFFFE; //Disable timer during setup
    TIMER0->CFG         =0x04;  //Set 16 bit mode
    TIMER0->TAMR        =0x02; // set to periodic, count down
    TIMER0->TAILR       =low; //Set interval load as LOW
    TIMER0->TAPR        =63; // Divide the clock by 16 to get 1us
    TIMER0->IMR         =0x01; //Enable timeout intrrupt

    //Timer0A is interrupt 19
    //Interrupt 16-19 are handled by NVIC register PRI4
    //Interrupt 19 is controlled by bits 31:29 of PRI4
    *NVIC_PRI4 &=0x00FFFFFF; //Clear interrupt 19 priority
    *NVIC_PRI4 |=0x40000000; //Set interrupt 19 priority to 2

    //NVIC has to be neabled
    //Interrupts 0-31 are handled by NVIC register EN0
    //Interrupt 19 is controlled by bit 19
    *NVIC_EN0 |=0x00080000;

    //Enable timer
    TIMER0->CTL          |=0x03; // bit0 to enable and bit 1 to stall on debug
    return;
}


void delay(){
for (int i = 0; i <=160000;i++){
;
}
}


int scaledval(int reading){
    int s_integer = ((reading*40)/4095); //this is to scale the reading to 1000
    return s_integer;
}
void setpwm(int val){
    high = val;
    low = 45 - val;
}
```

```
void initfunc(){
/*CLOCK CONFIG*/
SYSCTL->RCGCGPIO |= (1<<1);
SYSCTL->RCGCADC |= 0x01;

/*Port configurations*/
GPIOB->DIR &= ~(1<<4);
GPIOB->DEN &= ~(1<<4);
GPIOB->AFSEL |= (1<<4);
GPIOB->AMSEL |= (1<<4);

/*ATD configuration*/
ADC0->ACTSS &= ~(1<<3);
ADC0->EMUX &= ~(0xF000);
ADC0->SSMUX3 = 0x0A;
ADC0->SSCTL3 |= ((1<<1)|(1<<2));
ADC0->PC = 0X01;
ADC0->ACTSS |= (1<<3);
}

int main(){
    initfunc();
    pulse_init();
    int reading;
    int final_reading;
    int scaled;

    while(1){
    ADC0->PSSI |= (1<<3);
    while ((ADC0->RIS & (1<<3)) == 0){}
    reading = ADC0->SSFIFO3 & SAMPLE_MASK;
  final_reading = reading;
    scaled = scaledval(final_reading);
    setpwm(scaled);
    ADC0->ISC |= (1<<3);
    delay();
    }
}


void TIMER0A_Handler (void){
    if (state == 0){
    GPIOF->DATA  &= ~0x08;  //toggle PF3 pin
    TIMER0->TAILR = low;

        state =1;
    }
    else{
    GPIOF->DATA  |= 0x08;  //toggle PF3 pin
    TIMER0->TAILR   = high;
```

```
        state = 0;
    }
    TIMER0->ICR |=0x01; //Clear the interrupt
    return;
}
```