Phillip Sortomme
Github root directory:

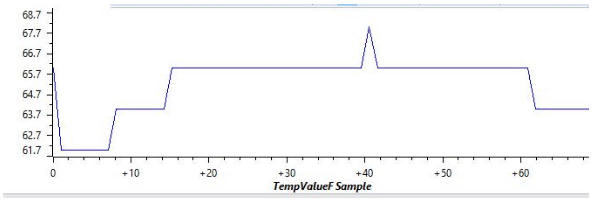
Date Submitted: **10/9/2018**

Task 01:

Youtube Link: https://youtu.be/PbEmzqD9Nc0

```
#include <stdint.h>
#include <stdbool.h>
#include "inc/hw_memmap.h"
#include "inc/hw_types.h"
#include "driverlib/debug.h"
#include "driverlib/sysctl.h"
#include "driverlib/adc.h"
#define TARGET_IS_BLIZZARD_RB1
#include "driverlib/rom.h"
#include "driverlib/gpio.h"

int main(void){
    uint32_t ui32ADC0Value[4];
    volatile uint32_t ui32TempAvg;
    volatile uint32_t ui32TempValueC;
    volatile uint32_t ui32TempValueF;

    ROM_SysCtlClockSet(SYSCTL_SYSDIV_5|SYSCTL_USE_PLL|SYSCTL_OSC_MAIN|SYSCTL_XTAL_16MHZ);

    ROM_SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOF);
    ROM_GPIOPinTypeGPIOOutput(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3);

    ROM_SysCtlPeripheralEnable(SYSCTL_PERIPH_ADC0);
    ROM_ADCHardwareOversampleConfigure(ADC0_BASE, 64);

    ROM_ADCSequenceConfigure(ADC0_BASE, 2, ADC_TRIGGER_PROCESSOR, 0);
    ROM_ADCSequenceStepConfigure(ADC0_BASE, 2, 0, ADC_CTL_TS);
    ROM_ADCSequenceStepConfigure(ADC0_BASE, 2, 1, ADC_CTL_TS);
    ROM_ADCSequenceStepConfigure(ADC0_BASE, 2, 2, ADC_CTL_TS);
    ROM_ADCSequenceStepConfigure(ADC0_BASE, 2, 3, ADC_CTL_TS|ADC_CTL_IE|ADC_CTL_END);
    ROM_ADCSequenceEnable(ADC0_BASE, 2);

    while(1){
        ROM_ADCIntClear(ADC0_BASE, 2);
        ROM_ADCProcessorTrigger(ADC0_BASE, 2);
       while(!ROM_ADCIntStatus(ADC0_BASE, 2, false)){}
       ROM_ADCSequenceDataGet(ADC0_BASE, 2, ui32ADC0Value);
       ui32TempAvg = (ui32ADC0Value[0] + ui32ADC0Value[1] + ui32ADC0Value[2] + ui32ADC0Value[3] + 2)/4;
       ui32TempValueC = (1475 - ((2475 * ui32TempAvg)) / 4096)/10;
       ui32TempValueF = ((ui32TempValueC * 9) + 160) / 5;
       // check if the temp is 72 or higher.
       if(ui32TempValueF >= 72 )
           ROM_GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_2, 4); // turn on LED
       else
           ROM_GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_2, 0); // turn off LED
    }
}
```

CCS output graph



Task 02:

Youtube Link: https://youtu.be/-8uDnJ-Yziw

```
#include <stdint.h>
#include <stdbool.h>
#include "inc/tm4c123gh6pm.h"
#include "inc/hw_memmap.h"
#include "inc/hw_types.h"
#include "driverlib/debug.h"
#include "driverlib/sysctl.h"
#include "driverlib/adc.h"
#define TARGET_IS_BLIZZARD_RB1
#include "driverlib/rom.h"
#include "driverlib/gpio.h"
#include "driverlib/interrupt.h"
#include "driverlib/timer.h"

#ifdef DEBUG
void__error__(char *pcFilename, uint32_t ui32Line)
{
}
#endif

    volatile uint32_t ui32TempAvg;
    volatile uint32_t ui32TempValueC;
    volatile uint32_t ui32TempValueF;

int main(void)
{
    ROM_SysCtlClockSet(SYSCTL_SYSDIV_5|SYSCTL_USE_PLL|SYSCTL_OSC_MAIN|SYSCTL_XTAL_16MHZ);
    ROM_SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOF);
    ROM_GPIOPinTypeGPIOOutput(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3);
    //set timer1 peripheral then configure timer1
    ROM_SysCtlPeripheralEnable(SYSCTL_PERIPH_TIMER1);
    ROM_TimerConfigure(TIMER1_BASE, TIMER_CFG_PERIODIC);

    ROM_SysCtlPeripheralEnable(SYSCTL_PERIPH_ADC0);
    ROM_ADCHardwareOversampleConfigure(ADC0_BASE, 32);

    ROM_ADCSequenceConfigure(ADC0_BASE, 2, ADC_TRIGGER_PROCESSOR, 0);
    ROM_ADCSequenceStepConfigure(ADC0_BASE, 2, 0, ADC_CTL_TS);
    ROM_ADCSequenceStepConfigure(ADC0_BASE, 2, 1, ADC_CTL_TS);
    ROM_ADCSequenceStepConfigure(ADC0_BASE, 2, 2, ADC_CTL_TS);
    ROM_ADCSequenceStepConfigure(ADC0_BASE, 2, 3, ADC_CTL_TS|ADC_CTL_IE|ADC_CTL_END);
    ROM_ADCSequenceEnable(ADC0_BASE, 2);
    // set timer to 2Hz (0.5s)
    ROM_TimerLoadSet(TIMER1_BASE, TIMER_A, (SysCtlClockGet()/2));
    ROM_IntEnable(INT_TIMER1A);
    ROM_TimerIntEnable(TIMER1_BASE, TIMER_TIMA_TIMEOUT);
    ROM_IntMasterEnable();

    ROM_TimerEnable(TIMER1_BASE, TIMER_A);

    while(1)  {}
}

void Timer1IntHandler(void)
{
    uint32_t ui32ADC0Value[4];

    // Clear the timer interrupt
    ROM_TimerIntClear(TIMER1_BASE, TIMER_TIMA_TIMEOUT);
    ROM_ADCIntClear(ADC0_BASE, 2);
    ROM_ADCProcessorTrigger(ADC0_BASE, 2);

    while(!ROM_ADCIntStatus(ADC0_BASE, 2, false)){}
    ROM_ADCSequenceDataGet(ADC0_BASE, 2, ui32ADC0Value);
    ui32TempAvg = (ui32ADC0Value[0] + ui32ADC0Value[1] + ui32ADC0Value[2] + ui32ADC0Value[3] + 2)/4;
    ui32TempValueC = (1475 - ((2475 * ui32TempAvg)) / 4096)/10;
    ui32TempValueF = ((ui32TempValueC * 9) + 160) / 5;
    // reload timer1 to begin 0.5s delay
    ROM_TimerLoadSet(TIMER1_BASE, TIMER_A, SysCtlClockGet()/2);
}
```