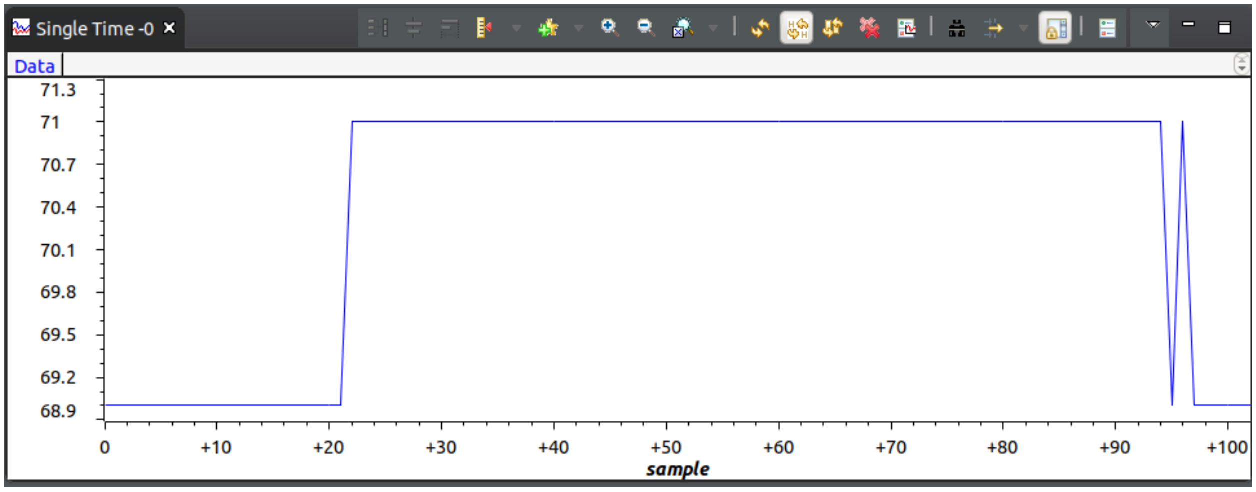
Brian West
Github root directory: https://github.com/westbrian2/Fall2019

**Date Submitted:10/25/2019**

## Task 01:



**Modified Code:**
```
#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"
#include "driverlib/gpio.h"
#define TARGET_IS_BLIZZARD_RB1
#include "driverlib/rom.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);

    //Configuring LED
    ROM_SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOF);
    ROM_GPIOPinTypeGPIOOutput(GPIO_PORTF_BASE,GPIO_PIN_2);



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

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

    while(1)
    {
        ROM_ADCIntClear(ADC0_BASE, 1);
        ROM_ADCProcessorTrigger(ADC0_BASE, 1);

       while(!ROM_ADCIntStatus(ADC0_BASE, 1, false))
       {
       }

       ROM_ADCSequenceDataGet(ADC0_BASE, 1, ui32ADC0Value);
       ui32TempAvg = (ui32ADC0Value[0] + ui32ADC0Value[1] + ui32ADC0Value[2] +
ui32ADC0Value[3] + 2)/4;
       ui32TempValueC = (1475 - ((2475 * ui32TempAvg)) / 4096)/10;
       ui32TempValueF = ((ui32TempValueC * 9) + 160) / 5;
       if(ui32TempValueF>72){
           GPIOPinWrite(GPIO_PORTF_BASE,GPIO_PIN_2,4);
       }
       else{
           GPIOPinWrite(GPIO_PORTF_BASE,GPIO_PIN_2,0);
       }
    }
}
```

------------------------------------------------------------------------------------

## Task 02:

Youtube Link: https://youtu.be/WYWZZcQ7GyM

```
#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"
#include "driverlib/gpio.h"
#include "driverlib/timer.h"
#include "driverlib/interrupt.h"
#define TARGET_IS_BLIZZARD_RB1
#include "driverlib/rom.h"

//uint32_t ui32period;

void configureTimer1A(void){
    SysCtlPeripheralEnable(SYSCTL_PERIPH_TIMER1);
    IntMasterEnable();
    TimerConfigure(TIMER1_BASE, TIMER_CFG_PERIODIC);
    //ui32period = (SysCtlClockGet()/2);
    TimerLoadSet(TIMER1_BASE, TIMER_A, 20000000-1);

    IntEnable(INT_TIMER1A);
    TimerIntEnable(TIMER1_BASE,TIMER_TIMA_TIMEOUT);
    TimerEnable(TIMER1_BASE, TIMER_A);
}


int main(void)
{



ROM_SysCtlClockSet(SYSCTL_SYSDIV_5|SYSCTL_USE_PLL|SYSCTL_OSC_MAIN|SYSCTL_XTAL_16MHZ);

    //Configuring LED
    ROM_SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOF);
    ROM_GPIOPinTypeGPIOOutput(GPIO_PORTF_BASE,GPIO_PIN_2);

    configureTimer1A();

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

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

    while(1){
    }
}

void Timer1IntHandler(void){
    TimerIntClear(TIMER1_BASE, TIMER_TIMA_TIMEOUT);
    uint32_t ui32ADC0Value[4];
    volatile uint32_t ui32TempAvg;
    volatile uint32_t ui32TempValueC;
    volatile uint32_t ui32TempValueF;

    ROM_ADCIntClear(ADC0_BASE, 1);
    ROM_ADCProcessorTrigger(ADC0_BASE, 1);

    while(!ROM_ADCIntStatus(ADC0_BASE, 1, false)){
    }

           ROM_ADCSequenceDataGet(ADC0_BASE, 1, ui32ADC0Value);
           ui32TempAvg = (ui32ADC0Value[0] + ui32ADC0Value[1] + ui32ADC0Value[2] +
ui32ADC0Value[3] + 2)/4;
```

```
            ui32TempValueC = (1475 - ((2475 * ui32TempAvg)) / 4096)/10;
            ui32TempValueF = ((ui32TempValueC * 9) + 160) / 5;
            if(ui32TempValueF>72){
                GPIOPinWrite(GPIO_PORTF_BASE,GPIO_PIN_2,4);
            }
            else{
                GPIOPinWrite(GPIO_PORTF_BASE,GPIO_PIN_2,0);
            }

}
```

**Grading scheme:** 30% Coding, 30% Documentation, 40% Execution/Video.