Ron Joshua Recrio
Github root directory: https://github.com/recrio/submissions.2/tree/master/Lab%205

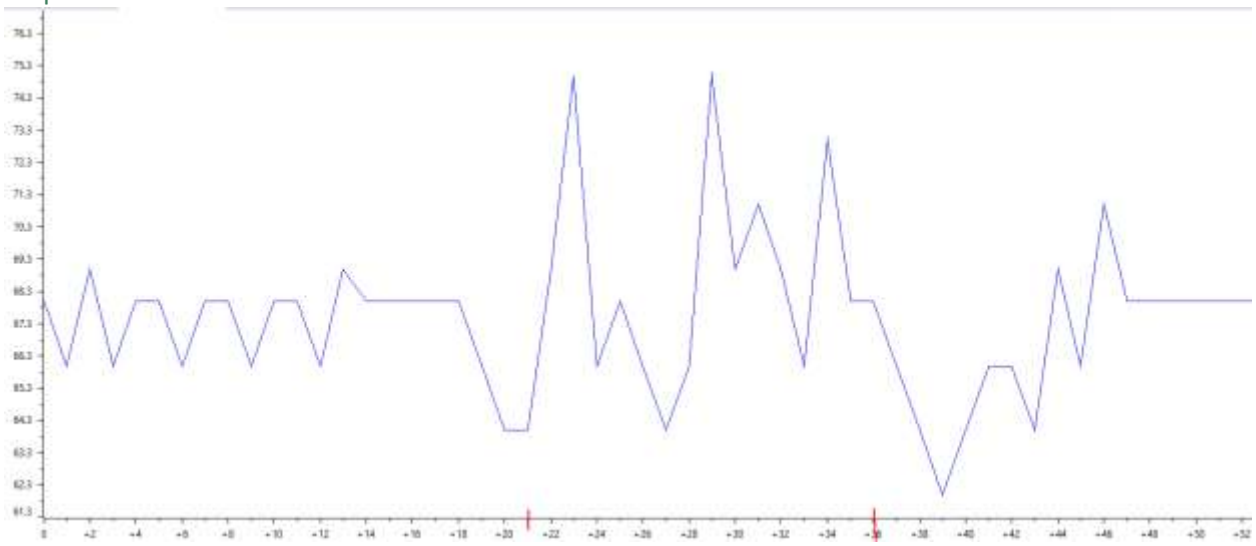
Date Submitted: 10/3/2019

Task 00: Execute provided code

Youtube Link:
https://youtu.be/RNnj99UDnbU

Task 01:

Used 67degF instead.
Graph:



Sample 21-36, my hand covered the system to generate some heat.
Youtube Link:
https://youtu.be/M0nYwIUk_Nk
Modified Schematic (if applicable):

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"

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

	SysCtlClockSet(SYSCTL_SYSDIV_5|SYSCTL_USE_PLL|SYSCTL_OSC_MAIN|SYSCTL_XTAL_16MH
Z);
```

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

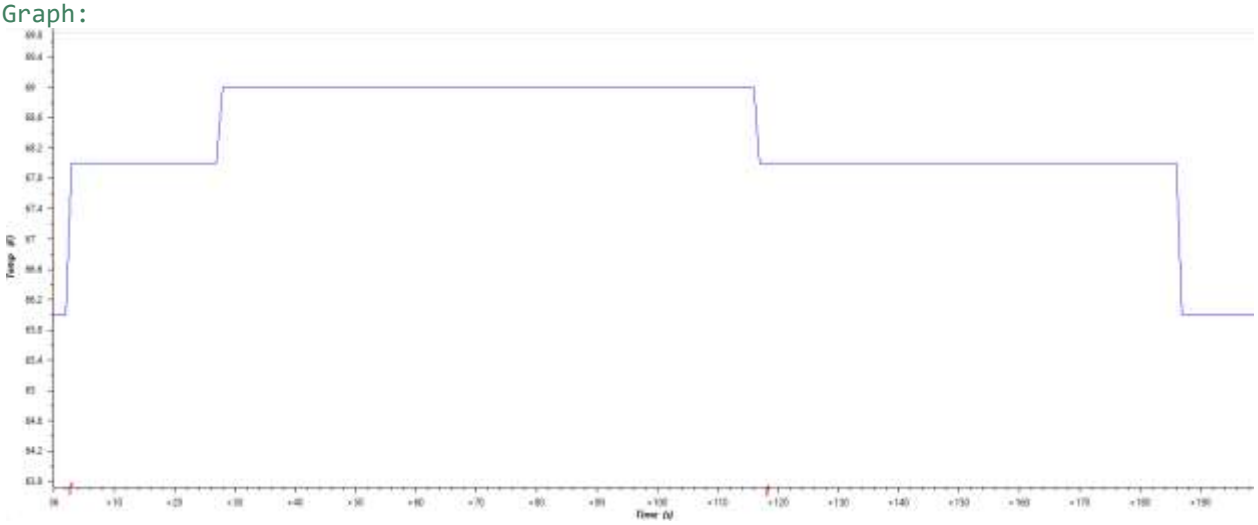
```
        SysCtlPeripheralEnable(SYSCTL_PERIPH_ADC0);

        ADCSequenceConfigure(ADC0_BASE, 2, ADC_TRIGGER_PROCESSOR, 0); // Sequencer 2
        ADCSequenceStepConfigure(ADC0_BASE, 2, 0, ADC_CTL_TS);
        ADCSequenceStepConfigure(ADC0_BASE, 2, 1, ADC_CTL_TS);
        ADCSequenceStepConfigure(ADC0_BASE, 2, 2, ADC_CTL_TS);
        ADCSequenceStepConfigure(ADC0_BASE, 2, 3, ADC_CTL_TS|ADC_CTL_IE|ADC_CTL_END);
        ADCSequenceEnable(ADC0_BASE, 2);

        // Enable PF1 and PF2 LEDs
        SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOF);
        GPIOPinTypeGPIOOutput(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2);

        while(1)
        {
            ADCIntClear(ADC0_BASE, 2);
            ADCProcessorTrigger(ADC0_BASE, 2);

            while(!ADCIntStatus(ADC0_BASE, 2, false))
            {
            }

            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;

            // If greater than 67degF change to blue, else red.
            if (ui32TempValueF > 67)
            {
                GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2, 4);
            }
            else
            {
                GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2, 2);
            }
        }
}
```

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

Ron Joshua Recrio
Github root directory: https://github.com/recrio/submissions.2/tree/master/Lab%205

Task 02:

Graph:



*Held down on processor at red marks to generate heat.

Verification:

To find this, I simply turned off the LED after turning it on to check if the timer is working at 0.5s as intended.

Youtube Link:
https://youtu.be/IXKIg2ZsP6Y

Modified Schematic (if applicable):

Modified Code:

```
#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"

uint32_t ui32Period;
uint32_t ui32ADC0Value[4];
volatile uint32_t ui32TempAvg;
volatile uint32_t ui32TempValueC;
volatile uint32_t ui32TempValueF;

int main(void)
{
    SysCtlClockSet(SYSCTL_SYSDIV_5|SYSCTL_USE_PLL|SYSCTL_OSC_MAIN|SYSCTL_XTAL_16MHZ);
```

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

```
    SysCtlPeripheralEnable(SYSCTL_PERIPH_ADC0);
    ADCHardwareOversampleConfigure(ADC0_BASE, 32); // Hardware Averaging of 32
Samples
    ADCSequenceConfigure(ADC0_BASE, 2, ADC_TRIGGER_PROCESSOR, 0); // Sequencer 2
    ADCSequenceStepConfigure(ADC0_BASE, 2, 0, ADC_CTL_TS);
    ADCSequenceStepConfigure(ADC0_BASE, 2, 1, ADC_CTL_TS);
    ADCSequenceStepConfigure(ADC0_BASE, 2, 2, ADC_CTL_TS);
    ADCSequenceStepConfigure(ADC0_BASE, 2, 3, ADC_CTL_TS|ADC_CTL_IE|ADC_CTL_END);

    // Enable PF1 and PF2 LEDs
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOF);
    GPIOPinTypeGPIOOutput(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2);

    SysCtlPeripheralEnable(SYSCTL_PERIPH_TIMER1);
    TimerConfigure(TIMER1_BASE, TIMER_CFG_PERIODIC);

    ui32Period = SysCtlClockGet() / 2;
    TimerLoadSet(TIMER1_BASE, TIMER_A, ui32Period-1);
    IntEnable(INT_TIMER1A);
    TimerIntEnable(TIMER1_BASE, TIMER_TIMA_TIMEOUT);
    IntMasterEnable();
    TimerEnable(TIMER1_BASE, TIMER_A);

    ADCSequenceEnable(ADC0_BASE, 2); // 4 steps * 32 samples = 128 samples
    ADCIntEnable(ADC0_BASE, 2);

    while(1)
    {
    }
}

void Timer1AIntHandler(void)
{
    TimerIntClear(TIMER1_BASE, TIMER_A);
    ADCIntClear(ADC0_BASE, 2);
    ADCProcessorTrigger(ADC0_BASE, 2);

    while(!ADCIntStatus(ADC0_BASE, 2, false))
    {
    }

    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;

    // If greater than 67degF change to blue, else red.
    if (ui32TempValueF > 67)
    {
        GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2, 4);
    }
    else
    {
        GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2, 2);
```

```
    }
}
```

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