Saul Alejandro Mendoza Guzman
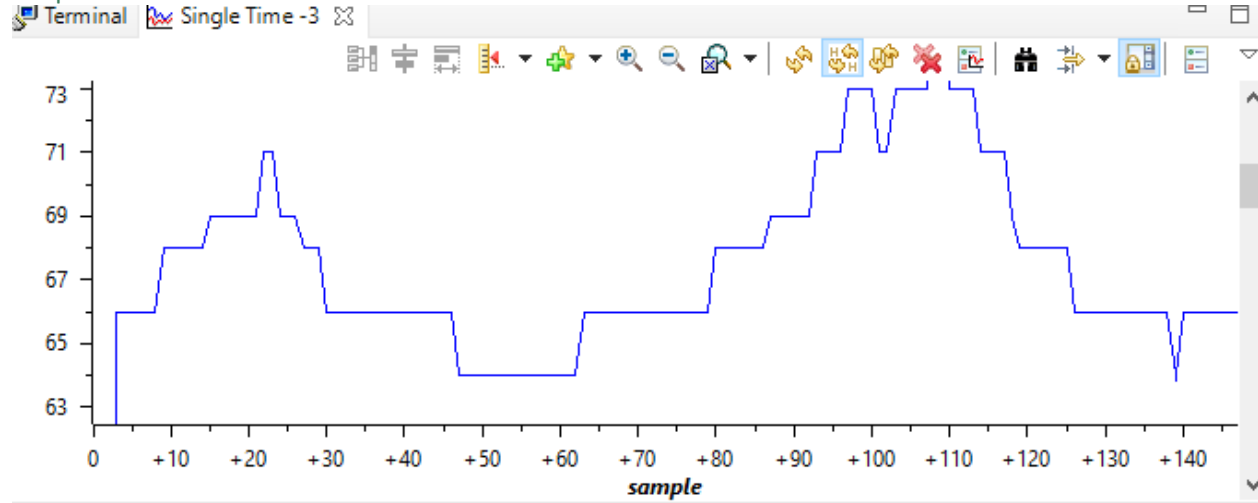Github root directory: https://github.com/mendos1/CPE_403/Lab5

Date Submitted: 10/10/2019

Task 00: Execute provided code

Youtube Link:
https://www.youtube.com/watch?v=WD3YLFheLXM

Task 01:

Graph:

Youtube Link:
https://www.youtube.com/watch?v=9LCvm5uinTg
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_16MZ
);

    SysCtlPeripheralEnable(SYSCTL_PERIPH_ADC0);

    ADCSequenceConfigure(ADC0_BASE, 2, ADC_TRIGGER_PROCESSOR, 0); // Sequencer 2
    ADCSequenceStepConfigure(ADC0_BASE, 2, 0, ADC_CTL_TS);
```

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

```
    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 68degF change to red, else blue.
        if (ui32TempValueF > 68) {
            GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2, 2);
        }
        else {
            GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2, 4);
        }
    }
}
```
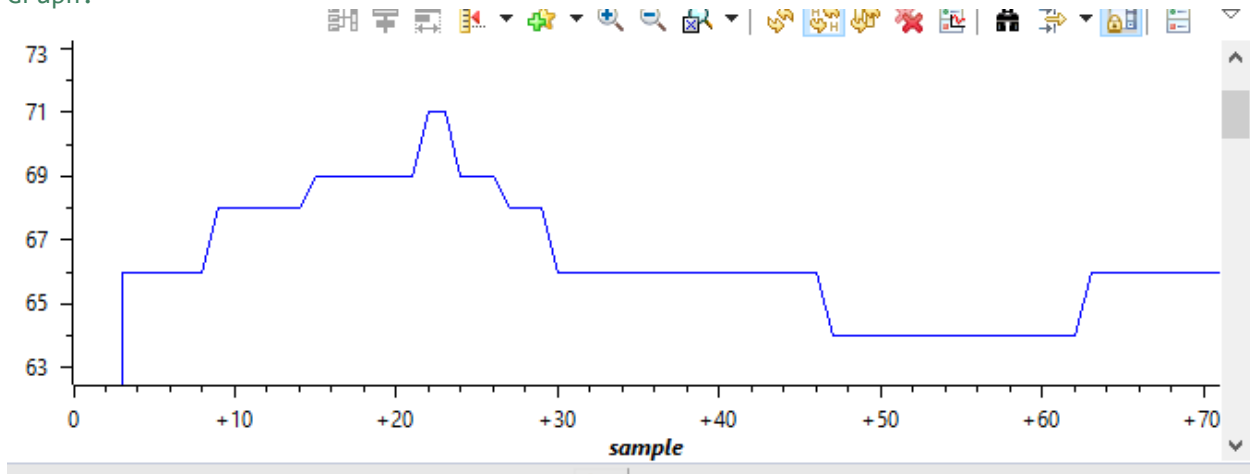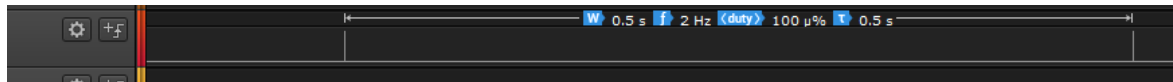

Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.

Task 02:

Graph:

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

Verification of timer:

Youtube Link:

https://www.youtube.com/watch?v=nnmxvSmT4kE

Modified Schematic (if applicable):

Modified Code:

```c
#include <stdint.h>
#include <stdbool.h>
#include "inc/hw_types.h"
#include "driverlib/debug.h"
#include "driverlib/sysctl.h"
#include "driverlib/adc.h"
#include "driverlib/gpio.h"
#include "inc/hw_memmap.h"
#include "inc/tm4c123gh6pm.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);

    SysCtlPeripheralEnable(SYSCTL_PERIPH_ADC0);
    ADCHardwareOversampleConfigure(ADC0_BASE, 32); // Hardware Averaging of 32
Samples
    ADCSequenceConfigure(ADC0_BASE, 2, ADC_TRIGGER_PROCESSOR, 0); // Sequencer 2
```

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

```
    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);

    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); // 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 68degF change to blue, else red.
    if (ui32TempValueF > 68){
        GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2, 2);
    }
    else{
        GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2, 4);
    }

}
```

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