Jett Guerrero
Github root directory: https://github.com/guerrj1/Advanced-Embedded-Systems

Date Submitted: 10/13/2019

Task 00: Execute provided code

Youtube Link: https://youtu.be/VkZPO0LfBVY

Task 01:

Youtube Link: https://youtu.be/xfE7qpBla1c

Modified Schematic (if applicable):

Modified Code:

```
//Task 1
#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;

       //clock initialization
       SysCtlClockSet(SYSCTL_SYSDIV_5|SYSCTL_USE_PLL|SYSCTL_OSC_MAIN|SYSCTL_XTAL_16MHZ);

       SysCtlPeripheralEnable(SYSCTL_PERIPH_ADC0);
       //GPIO settings
       SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOF);
       GPIOPinTypeGPIOOutput(GPIO_PORTF_BASE, GPIO_PIN_1 | GPIO_PIN_2 | GPIO_PIN_3);

       //ADC settings
       ADCHardwareOversampleConfigure(ADC0_BASE, 64);

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

       while(1)
       {
              ADCIntClear(ADC0_BASE, 1);  //clears ADC
```

```
		ADCProcessorTrigger(ADC0_BASE, 1);

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

		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;

		//checks if temp is greater than 72
		if(ui32TempValueF > 72)
		{
			GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_2, 4);  //turn led on when
greater than 72
		}
		else
		{
			GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_2, 0); //turn led off when
less than 72
		}

	}
}
```

Task 02:

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

Modified Schematic (if applicable):

Modified Code:

```
//Task 2
#include <stdint.h>
#include <stdbool.h>
#include "inc/tm4c123gh6pm.h"
#include "driverlib/interrupt.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"

//global variables
uint32_t ui32ADC0Value[4];
volatile uint32_t ui32TempAvg;
volatile uint32_t ui32TempValueC;
volatile uint32_t ui32TempValueF;
```

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

	SysCtlPeripheralEnable(SYSCTL_PERIPH_ADC0);

	//GPIO settings
	//SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOF);
	GPIOPinTypeGPIOOutput(GPIO_PORTF_BASE, GPIO_PIN_1 | GPIO_PIN_2 | GPIO_PIN_3);

	//ADC settings
	ADCHardwareOversampleConfigure(ADC0_BASE, 32);  //API call for hardware averaging
32

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

	//Timer 1 initialization
	SysCtlPeripheralEnable(SYSCTL_PERIPH_TIMER1);
	IntMasterEnable();
	TimerConfigure(TIMER1_BASE, TIMER_CFG_PERIODIC);
	TimerLoadSet(TIMER1_BASE, TIMER_A, 0);
	IntEnable(INT_TIMER1A);
	TimerIntEnable(TIMER1_BASE, TIMER_TIMA_TIMEOUT);
	TimerEnable(TIMER1_BASE, TIMER_A);


	while(1)
	{
	}
}

void Timer1IntHandler(void)
{
	int32_t ui32Period = (SysCtlClockGet()) / 2; //0.5secs
	TimerIntClear(TIMER1_BASE, TIMER_TIMA_TIMEOUT);
	TimerLoadSet(TIMER1_BASE, TIMER_A, ui32Period);

	ADCIntClear(ADC0_BASE, 1);  //clears ADC
	ADCProcessorTrigger(ADC0_BASE, 1);

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

	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); //turn led on
        }
        else
        {
                GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_2, 0); //turn led off
        }
}
//------------------------------------------------------------------------------
```

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