**Date Submitted: 11/30/19**

**Task 00: Execute provided code**

Youtube Link: **https://youtu.be/Zyh6cONgi8Q**

**#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"**

**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\_16MHZ);**

**SysCtlPeripheralEnable(SYSCTL\_PERIPH\_ADC0);**

**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);**

**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;**

**}**

**}**

**------------------------------------------------------------------------------------**

**Task 01:**

Youtube Link: https://youtu.be/2l7GTxZR5cs

**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\_16MHZ);**

**SysCtlPeripheralEnable(SYSCTL\_PERIPH\_ADC0);**

**//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);**

**SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOF);**

**GPIOPinTypeGPIOOutput(GPIO\_PORTF\_BASE, GPIO\_PIN\_2);**

**while(1)**

**{**

**ADCIntClear(ADC0\_BASE, 1);**

**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;**

**// Turn on the LED at PF2 if the temperature is greater than 72 degF.**

**if(ui32TempValueF > 72) {GPIOPinWrite (GPIO\_PORTF\_BASE,GPIO\_PIN\_2,4); } // 4 = BLUE\_LED**

**else {GPIOPinWrite(GPIO\_PORTF\_BASE,GPIO\_PIN\_2,0);} // Keep LED off**

**}**

**}**

**------------------------------------------------------------------------------------**

**Task 02:**

Youtube Link: https://youtu.be/lVZq\_mjsswM

**Modified Code:**

**#include <stdint.h>**

**#include <stdbool.h>**

**#include "inc/hw\_memmap.h"**

**#include "inc/hw\_types.h"**

**#include "inc/tm4c123gh6pm.h"**

**#include "driverlib/interrupt.h"**

**#include "driverlib/timer.h"**

**#include "driverlib/debug.h"**

**#include "driverlib/sysctl.h"**

**#include "driverlib/adc.h"**

**#include "driverlib/gpio.h"**

**uint32\_t tPeriod;**

**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); // system clock run at 40MHz**

**SysCtlPeripheralEnable(SYSCTL\_PERIPH\_ADC0);// enable the ADC0 peripheral**

**ADCHardwareOversampleConfigure(ADC0\_BASE, 32); // hardware averaging**

**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);// enable ADC sequencer 1**

**SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOF);**

**GPIOPinTypeGPIOOutput(GPIO\_PORTF\_BASE, GPIO\_PIN\_2); // Enable PF2**

**tPeriod = SysCtlClockGet()/2;**

**configTimer1A();**

**IntMasterEnable();**

**ADCIntEnable(ADC0\_BASE,2);**

**while(1)**

**{**

**}**

**}**

**void configTimer1A()**

**{**

**SysCtlPeripheralEnable(SYSCTL\_PERIPH\_TIMER1);**

**TimerConfigure(TIMER1\_BASE, TIMER\_CFG\_PERIODIC);**

**TimerLoadSet(TIMER1\_BASE, TIMER\_A, tPeriod-1); // counts up to sec\_delay**

**TimerIntEnable(TIMER1\_BASE, TIMER\_TIMA\_TIMEOUT);**

**IntEnable(INT\_TIMER1A);**

**TimerEnable(TIMER1\_BASE, TIMER\_A);**

**}**

**Timer1IntHandler(void)**

**{**

**TimerIntClear(TIMER1\_BASE, TIMER\_A);**

**ADCIntClear(ADC0\_BASE, 1);**

**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); } // 4 = BLUE\_LED**

**else {GPIOPinWrite(GPIO\_PORTF\_BASE,GPIO\_PIN\_2,0);} // Keep LED off**

**}**