**10/09/2018**

**Task 00: Execute provided code**

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

Modified code highlighted in yellow

**Task 01:**

Youtube Link: <https://youtu.be/EHXwowkUnkw>

**Modified Code:**

**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);

**SysCtlPeripheralEnable**(SYSCTL\_PERIPH\_GPIOF); // Enabling GPIO for LED

**GPIOPinTypeGPIOOutput**(GPIO\_PORTF\_BASE, GPIO\_PIN\_2); // Enable Pin2 for Blue LED

**ADCHardwareOversampleConfigure**(ADC0\_BASE, 64);

**ADCSequenceConfigure**(ADC0\_BASE, 2, ADC\_TRIGGER\_PROCESSOR, 0); // Changed to 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);

**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** ( ui32TempValueF > 64 ) // This is the threshold of my environment temperature to detect change

{

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

}

**else**

{

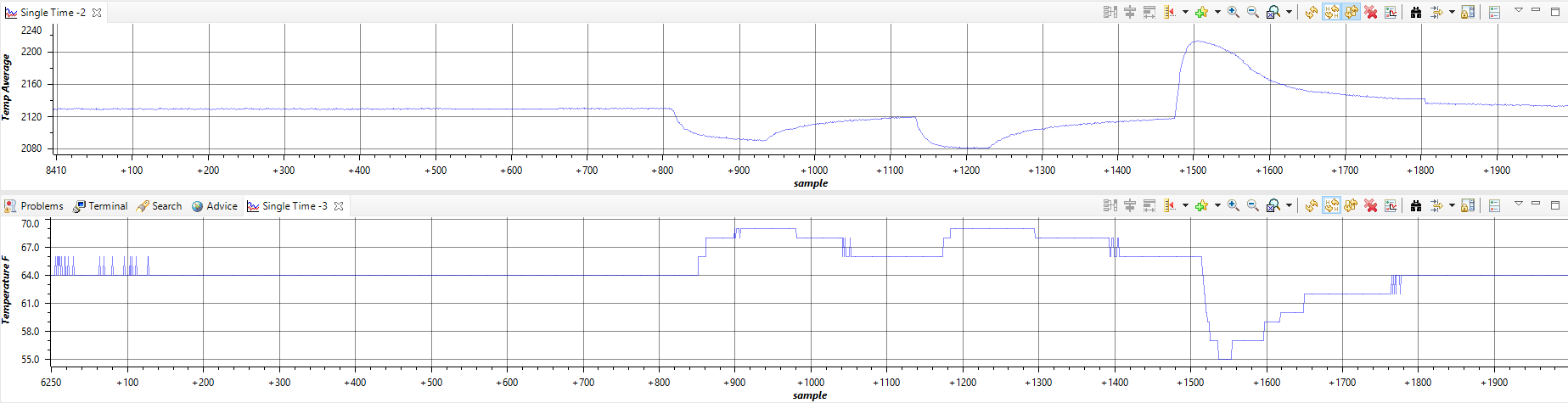
**GPIOPinWrite**(GPIO\_PORTF\_BASE,GPIO\_PIN\_2, 0);

}

}

}

**Displaying temperature using built-in graphing tool**



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

**Task 02:**

Youtube Link: <https://youtu.be/gmuuoUaAxJ8>

**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"

**#define** TARGET\_IS\_BLIZZARD\_RB1

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

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

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

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

**#include** "driverlib/rom\_map.h"

**#include** "driverlib/pin\_map.h"

**#ifdef** DEBUG

**void\_\_error\_\_**(**char** \*pcFilename, uint32\_t ui32Line)

{

}

**#endif**

// Globals

uint32\_t ui32Period;

uint32\_t ui32ADC0Value[4];

**volatile** uint32\_t ui32TempAvg;

**volatile** uint32\_t ui32TempValueC;

**volatile** uint32\_t ui32TempValueF;

**int** **main**(**void**)

{

// Clock configuration

ROM\_SysCtlClockSet(SYSCTL\_SYSDIV\_5|SYSCTL\_USE\_PLL|SYSCTL\_OSC\_MAIN|SYSCTL\_XTAL\_16MHZ); //Clock configuration

// LED configuration

ROM\_SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOF); // Enabling GPIO for LED

ROM\_GPIOPinTypeGPIOOutput(GPIO\_PORTF\_BASE, GPIO\_PIN\_2); // Enable Pin2 for Blue LED

// ADC configuration

ROM\_SysCtlPeripheralEnable(SYSCTL\_PERIPH\_ADC0); // ADC peripheral enabled

ROM\_ADCHardwareOversampleConfigure(ADC0\_BASE, 32); // Introducing hardware averaging of 32

ROM\_ADCSequenceConfigure(ADC0\_BASE, 2, ADC\_TRIGGER\_PROCESSOR, 0); // ADC trigger mode

ROM\_ADCSequenceStepConfigure(ADC0\_BASE, 2, 0, ADC\_CTL\_TS); // Sequences configuration

ROM\_ADCSequenceStepConfigure(ADC0\_BASE, 2, 1, ADC\_CTL\_TS);

ROM\_ADCSequenceStepConfigure(ADC0\_BASE, 2, 2, ADC\_CTL\_TS);

ROM\_ADCSequenceStepConfigure(ADC0\_BASE, 2, 3, ADC\_CTL\_TS|ADC\_CTL\_IE|ADC\_CTL\_END);

// Timer 1 configuration

ROM\_SysCtlPeripheralEnable(SYSCTL\_PERIPH\_TIMER1); // Timer 1 enabled

ROM\_TimerConfigure(TIMER1\_BASE, TIMER\_CFG\_PERIODIC); // Timer in periodic mode

ui32Period = ROM\_SysCtlClockGet()/2; // Period of 0.5s 2Hz

ROM\_TimerLoadSet(TIMER1\_BASE, TIMER\_A, ui32Period -1); // setting timer to expire at 0.5 seconds

ROM\_IntEnable(INT\_TIMER1A); // Interrupt for Timer1A enabled

ROM\_TimerIntEnable(TIMER1\_BASE, TIMER\_TIMA\_TIMEOUT);

// Enable interrupts

ROM\_IntMasterEnable();

ROM\_ADCSequenceEnable(ADC0\_BASE, 2);

ROM\_TimerEnable(TIMER1\_BASE, TIMER\_A);

**while** (1) //wait forever

{

}

}

**void** **Timer1IntHandler**(**void**)

{

// Clear the timer interrupt

ROM\_TimerIntClear(TIMER1\_BASE, TIMER\_TIMA\_TIMEOUT);

ROM\_ADCIntClear(ADC0\_BASE, 2);

ROM\_ADCProcessorTrigger(ADC0\_BASE, 2);

**while**(!ROM\_ADCIntStatus(ADC0\_BASE, 2, false))

{

}

ROM\_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** ( ui32TempValueF > 68 ) // This is the threshold of my environment temperature to detect change

{

ROM\_GPIOPinWrite(GPIO\_PORTF\_BASE,GPIO\_PIN\_2, GPIO\_PIN\_2); // turn it on

}

**else**

{

ROM\_GPIOPinWrite(GPIO\_PORTF\_BASE,GPIO\_PIN\_2, 0); // turn it off

}

}