**Date Submitted: 10/01/2019**

**Task 00:**

**Youtube Link:**

<https://youtu.be/lpJu_-H1MIU>

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

**Task 01:**

Youtube Link:

<https://youtu.be/hs_3fmUfMWY>

**Modified Code:**

#include <stdint.h>

#include <stdbool.h>

#include "\ti\tivaware\_c\_series\_2\_1\_4\_178\inc\hw\_memmap.h"

#include "\ti\tivaware\_c\_series\_2\_1\_4\_178\inc\hw\_types.h"

#include "\ti\tivaware\_c\_series\_2\_1\_4\_178\driverlib\debug.h"

#include "\ti\tivaware\_c\_series\_2\_1\_4\_178\driverlib\sysctl.h"

#include "\ti\tivaware\_c\_series\_2\_1\_4\_178\driverlib\adc.h"

#include "\ti\tivaware\_c\_series\_2\_1\_4\_178\driverlib\gpio.h"

#ifdef DEBUG

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

{

}

#endif

int main(void)

{

uint32\_t ui32ADC0Value[4];

volatile uint32\_t ui32TempAvg;

volatile uint32\_t ui32TempValueC;

volatile uint32\_t ui32TempValueF;

// Setup Clock to 40MHz

SysCtlClockSet(SYSCTL\_SYSDIV\_5 | SYSCTL\_USE\_PLL | SYSCTL\_OSC\_MAIN | SYSCTL\_XTAL\_16MHZ);

// Enables ADC peripheral

SysCtlPeripheralEnable(SYSCTL\_PERIPH\_ADC0);

SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOF);

GPIOPinTypeGPIOOutput(GPIO\_PORTF\_BASE, GPIO\_PIN\_1 | GPIO\_PIN\_2 | GPIO\_PIN\_3);

ADCHardwareOversampleConfigure(ADC0\_BASE, 64);

// Hardware averaging. ( 2, 4, 8 , 16, 32, 64 )

// ADC Sequencer Configuration

ADCSequenceConfigure(ADC0\_BASE, 2, ADC\_TRIGGER\_PROCESSOR, 0);

// We want the processor to trigger the sequence and highest priority.

// Configure steps 0-2 on sequencer 1 to sample the temperature sensor.

ADCSequenceStepConfigure(ADC0\_BASE, 2, 0, ADC\_CTL\_TS);

// Sequencer Step 0: Samples Temperature Sensor

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

// Sequencer Step 1: Samples Temperature Sensor

ADCSequenceStepConfigure(ADC0\_BASE, 2, 2, ADC\_CTL\_TS);

// Sequencer Step 2: Samples Temperature Sensor

ADCSequenceStepConfigure(ADC0\_BASE, 2, 3, ADC\_CTL\_TS | ADC\_CTL\_IE | ADC\_CTL\_END); // Therefore, Temperature result is a result of 64 x 4 = 256 samples

ADCSequenceEnable(ADC0\_BASE, 2); // Enable ADC sequencer 1

while (1)

{

ADCIntClear(ADC0\_BASE, 2); // CLear ADC

ADCProcessorTrigger(ADC0\_BASE, 2);

// Triggers ADC conversion with software

while (!ADCIntStatus(ADC0\_BASE, 2, false))

// While conversion is not complete.

{

}

ADCSequenceDataGet(ADC0\_BASE, 2, ui32ADC0Value);

// Copies ADC values to a buffer, ui32ADC0Value[4]

ui32TempAvg = (ui32ADC0Value[0] + ui32ADC0Value[1] + ui32ADC0Value[2] + ui32ADC0Value[3] + 2) / 4;

// Average Temperature calc

ui32TempValueC = (1475 - ((2475 \* ui32TempAvg)) / 4096) / 10;

// Celsius Conversion

ui32TempValueF = ((ui32TempValueC \* 9) + 160) / 5; // Fahrenheit Conversion

if (ui32TempValueF >= 66) {

GPIOPinWrite(GPIO\_PORTF\_BASE, GPIO\_PIN\_2, 4);

}

else {

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

}

}

}

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

**Task 02:**

Youtube Link:

<https://youtu.be/1cQQoYABiHc>

**Modified Code:**

#include <stdint.h>

#include <stdbool.h>

#include "\ti\tivaware\_c\_series\_2\_1\_4\_178\inc\tm4c123gh6pm.h"

#include "\ti\tivaware\_c\_series\_2\_1\_4\_178\inc\hw\_memmap.h"

#include "\ti\tivaware\_c\_series\_2\_1\_4\_178\inc\hw\_types.h"

#include "\ti\tivaware\_c\_series\_2\_1\_4\_178\driverlib\sysctl.h"

#include "\ti\tivaware\_c\_series\_2\_1\_4\_178\driverlib\interrupt.h"

#include "\ti\tivaware\_c\_series\_2\_1\_4\_178\driverlib\gpio.h"

#include "\ti\tivaware\_c\_series\_2\_1\_4\_178\driverlib\timer.h"

#include "\ti\tivaware\_c\_series\_2\_1\_4\_178\driverlib\adc.h"

#include "\ti\tivaware\_c\_series\_2\_1\_4\_178\driverlib\debug.h"

#ifdef DEBUG

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

{

}

#endif

void configureTimer1A();

uint32\_t ui32ADC0Value[1];

volatile uint32\_t ui32TempAvg;

volatile uint32\_t ui32TempValueC;

volatile uint32\_t ui32TempValueF;

int main()

{

SysCtlClockSet(SYSCTL\_SYSDIV\_5 | SYSCTL\_USE\_PLL | SYSCTL\_OSC\_MAIN | SYSCTL\_XTAL\_16MHZ);

SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOF);

GPIOPinTypeGPIOOutput(GPIO\_PORTF\_BASE, GPIO\_PIN\_1 | GPIO\_PIN\_2 | GPIO\_PIN\_3);

SysCtlPeripheralEnable(SYSCTL\_PERIPH\_ADC0);

ADCHardwareOversampleConfigure(ADC0\_BASE, 32);

ADCSequenceConfigure(ADC0\_BASE, 3, ADC\_TRIGGER\_PROCESSOR, 0);

ADCSequenceStepConfigure(ADC0\_BASE, 3, 0, ADC\_CTL\_TS | ADC\_CTL\_IE | ADC\_CTL\_END);

configureTimer1A();

ADCSequenceEnable(ADC0\_BASE, 3);

ADCIntEnable(ADC0\_BASE, 3);

while (1)

{

}

}

void configureTimer1A()

{

int32\_t ui32PeriodHigh = (SysCtlClockGet() / 1);

SysCtlPeripheralEnable(SYSCTL\_PERIPH\_TIMER1);

IntMasterEnable();

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

TimerLoadSet(TIMER1\_BASE, TIMER\_A, 5 \* (SysCtlClockGet() / 10));

IntEnable(INT\_TIMER1A);

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

TimerEnable(TIMER1\_BASE, TIMER\_A);

}

void Timer1IntHandler(void)

{

int32\_t ui32PeriodHigh = 0.5 \* (SysCtlClockGet()); // Delay 0.5s

TimerIntClear(TIMER1\_BASE, TIMER\_TIMA\_TIMEOUT);

TimerLoadSet(TIMER1\_BASE, TIMER\_A, ui32PeriodHigh);

ADCIntClear(ADC0\_BASE, 3);

ADCProcessorTrigger(ADC0\_BASE, 3);

while (!ADCIntStatus(ADC0\_BASE, 3, false))

{

}

ADCSequenceDataGet(ADC0\_BASE, 3, ui32ADC0Value);

ui32TempValueC = (1475 - ((2475 \* ui32ADC0Value[0])) / 4096) / 10;

ui32TempValueF = ((ui32TempValueC \* 9) + 160) / 5;

if (ui32TempValueF > 66) {

GPIOPinWrite(GPIO\_PORTF\_BASE, GPIO\_PIN\_2, 4); // Turn on the LED

}

else {

GPIOPinWrite(GPIO\_PORTF\_BASE, GPIO\_PIN\_2, 0); // Turn off the blue LED

}

}

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