**Date Submitted: 10-10-2018**

**Task 00: Execute provided code (No submission required)**

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

**Task 01: Change the ADC Sequencer to SS3. Turn on the LED at PF2 if the temperature is greater**

**that 72 degF. Use internal temperature sensor for all SS2 sequence. Display the temperature in**

**the built-in graph tool.**

Youtube Link: <https://youtu.be/i7BvOY4EXXQ>

**Modified Code:**

//task01: Change the ADC Sequencer to SS3. Turn on the LED at PF2 if the

//temperature is greater that 72 degF. Use internal temperature sensor

//for all SS2 sequence

#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"

#define TARGET\_IS\_BLIZZARD\_RB1

#include "driverlib/rom.h"

#include "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**;**

SysCtlClockSet**(**SYSCTL\_SYSDIV\_5**|**SYSCTL\_USE\_PLL**|**SYSCTL\_OSC\_MAIN**|**SYSCTL\_XTAL\_16MHZ**);** //40MHz

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

SysCtlPeripheralEnable**(**SYSCTL\_PERIPH\_GPIOF**);** //enable GPIO peripherals

GPIOPinTypeGPIOOutput**(**GPIO\_PORTF\_BASE**,** GPIO\_PIN\_2**);** //set G as output

//ADCHardwareOversampleConfigure(ADC0\_BASE, 32); //hardware sampling

//configure ADC0 with temperature sensor

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

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**);** //enable sequence

**while(**1**){**

ADCIntClear**(**ADC0\_BASE**,** 2**);** //clear the ADC buffer

ADCProcessorTrigger**(**ADC0\_BASE**,** 2**);** //set off trigger to start convert

//wait for ADC to finish converting

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

**{**

**}**

ADCSequenceDataGet**(**ADC0\_BASE**,** 2**,** ui32ADC0Value**);** //grrab ADC value

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**);**

**}**

**else**

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

**}**

**}**

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

**{**

**}**

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

**Task 02: Introduce hardware averaging to 32. Using the timer TIMER1A conduct an ADC conversion**

**on overflow every 0.5 sec. Use the Timer1A interrupt. Display the temperature in the**

**built-in graph tool.**

Youtube Link: <https://youtu.be/kGATb86V7DA>

**Modified Code:**

//task02: Introduce hardware averaging to 32. Using the timer TIMER1A conduct

//an ADC conversion on overflow every 0.5 sec. Use the Timer1A interrupt.

#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"

#define TARGET\_IS\_BLIZZARD\_RB1

#include "driverlib/rom.h"

#include "driverlib/gpio.h"

#include "driverlib/timer.h"

#include "driverlib/interrupt.h"

#include "inc/tm4c123gh6pm.h"

#ifdef DEBUG

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

**{**

**}**

#endif

uint32\_t ui32ADC0Value**[**4**];**

volatile uint32\_t ui32TempAvg**;**

volatile uint32\_t ui32TempValueC**;**

volatile uint32\_t ui32TempValueF**;**

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

**{**

uint32\_t ui32Period**;**

SysCtlClockSet**(**SYSCTL\_SYSDIV\_5**|**SYSCTL\_USE\_PLL**|**SYSCTL\_OSC\_MAIN

**|**SYSCTL\_XTAL\_16MHZ**);**

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

SysCtlPeripheralEnable**(**SYSCTL\_PERIPH\_GPIOF**);** //enable GPIO peripherals

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

GPIOPinTypeGPIOOutput**(**GPIO\_PORTF\_BASE**,** GPIO\_PIN\_2**);** //set G as output

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

//configure timer type and period

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

ui32Period **=** **(**SysCtlClockGet**()** **/** 2**);**

TimerLoadSet**(**TIMER1\_BASE**,** TIMER\_A**,** ui32Period **-**1**);**

//configure ADC0 with temperature sensor

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

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**);** //enable sequence

IntEnable**(**INT\_TIMER1A**);**

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

IntMasterEnable**();**

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

**while(**1**){**

**}**

**}**

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

**{**

// Clear the timer interrupt

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

ADCIntClear**(**ADC0\_BASE**,** 2**);** //clear the ADC buffer

ADCProcessorTrigger**(**ADC0\_BASE**,** 2**);** //set off trigger to start convert

//wait for ADC to finish converting

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

**{**

**}**

ADCSequenceDataGet**(**ADC0\_BASE**,** 2**,** ui32ADC0Value**);** //grrab ADC value

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**);**

**}**

**else**

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

**}**