**10/2/2018:**

**Task 00: Execute provided code**

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

**Task 01:**

Youtube Link: https://www.youtube.com/watch?v=tJOz0sgp3QU

**Modified Code:**

**// initialize count**

**uint32\_t count = 0;**

**// Configure timer 0**

**SysCtlPeripheralEnable(SYSCTL\_PERIPH\_TIMER0);**

**TimerConfigure(TIMER0\_BASE, TIMER\_CFG\_PERIODIC);**

**// set period and set timer**

**ui32Period = (SysCtlClockGet() / 16);**

**TimerLoadSet(TIMER0\_BASE, TIMER\_A, ui32Period -1);**

**// set up interrupt**

**IntEnable(INT\_TIMER0A);**

**TimerIntEnable(TIMER0\_BASE, TIMER\_TIMA\_TIMEOUT);**

**IntMasterEnable();**

**// start timer**

**TimerEnable(TIMER0\_BASE, TIMER\_A);**

**while(1)**

**{**

**}**

**}**

**void Timer0IntHandler(void)**

**{**

**// Clear the timer interrupt**

**TimerIntClear(TIMER0\_BASE, TIMER\_TIMA\_TIMEOUT);**

**// turn LED on if count < 4 and inc count**

**if(count < 4) {**

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

**count++;**

**} else {**

**// turn LED off and reset count**

**GPIOPinWrite(GPIO\_PORTF\_BASE, GPIO\_PIN\_1|GPIO\_PIN\_2|GPIO\_PIN\_3, 0);**

**count = 0;**

**}**

**}**

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

**Task 02:**

Youtube Link: https://www.youtube.com/watch?v=y-XVBHZdiwc

**Modified Schematic (if applicable):**

**Modified Code:**

#include <stdint.h>

#include <stdbool.h>

#include "inc/tm4c123gh6pm.h"

#include "inc/hw\_memmap.h"

#include "inc/hw\_types.h"

#include "inc/hw\_gpio.h"

#include "driverlib/sysctl.h"

#include "driverlib/interrupt.h"

#include "driverlib/gpio.h"

#include "driverlib/timer.h"

// tracks if the switch is pressed

unsigned char volatile pressed;

int main(void)

{

uint32\_t ui32Period;

// set up the clock

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

// enable port F

SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOF);

// unlock the GPIOCR register for port F and free pin 0

HWREG(GPIO\_PORTF\_BASE + GPIO\_O\_LOCK) = GPIO\_LOCK\_KEY;

HWREG(GPIO\_PORTF\_BASE + GPIO\_O\_CR) = 0x01;

// set LEDs as output and switches as input

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

GPIOPinTypeGPIOInput(GPIO\_PORTF\_BASE, GPIO\_PIN\_0|GPIO\_PIN\_4);

// configure switches

GPIOPadConfigSet(GPIO\_PORTF\_BASE,GPIO\_PIN\_0,GPIO\_STRENGTH\_2MA,GPIO\_PIN\_TYPE\_STD\_WPU); // enable F0's pullup, the drive strength won't affect the input

// enable switches

SysCtlPeripheralEnable(SYSCTL\_PERIPH\_TIMER0);

SysCtlPeripheralEnable(SYSCTL\_PERIPH\_TIMER1);

// configure timer0

TimerConfigure(TIMER0\_BASE, TIMER\_CFG\_PERIODIC);

// calculate period and set timer0

ui32Period = (SysCtlClockGet() / 5) / 2;

TimerLoadSet(TIMER0\_BASE, TIMER\_A, ui32Period -1);

// enable interrupts

IntEnable(INT\_TIMER0A);

IntEnable(INT\_GPIOF);

GPIOIntTypeSet(GPIO\_PORTF\_BASE,GPIO\_PIN\_0,GPIO\_RISING\_EDGE);

TimerIntEnable(TIMER0\_BASE, TIMER\_TIMA\_TIMEOUT);

GPIOIntEnable(GPIO\_PORTF\_BASE, GPIO\_INT\_PIN\_0);

IntMasterEnable();

// enable timer

TimerEnable(TIMER0\_BASE, TIMER\_A);

while(1)

{

}

}

void Timer0IntHandler(void)

{

// Clear the timer interrupt

TimerIntClear(TIMER0\_BASE, TIMER\_TIMA\_TIMEOUT);

// Read the current state of the GPIO pin and

// write back the opposite state

if(pressed == 0) {

if(GPIOPinRead(GPIO\_PORTF\_BASE, GPIO\_PIN\_2))

{

GPIOPinWrite(GPIO\_PORTF\_BASE, GPIO\_PIN\_1|GPIO\_PIN\_2|GPIO\_PIN\_3, 0);

}

else

{

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

}

}

}

void Timer1IntHandler(void)

{

// Clear the timer interrupt

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

IntDisable(INT\_TIMER1A);

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

// disable timer1

TimerDisable(TIMER1\_BASE, TIMER\_A);

//reset to blinking

pressed = 0;

}

void SW2IntHandler(void)

{

// check if button is pressed

if(GPIOPinRead(GPIO\_PORTF\_BASE, GPIO\_PIN\_0))

{

pressed = 1;

// turn LED on

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

// calculate 1.5 seconds

uint32\_t ui32Period;

ui32Period = (SysCtlClockGet() \* 1.5);

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

//enable interupt for timer1

IntEnable(INT\_TIMER1A);

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

TimerEnable(TIMER1\_BASE, TIMER\_A);

}

// clear interrupt

GPIOIntClear (GPIO\_PORTF\_BASE, GPIO\_INT\_PIN\_0);

}

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