**Date Submitted: 10-04-2018**

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

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

**Task 01: Toggle of the GPIO at 2 Hz using Timer0 with 75% duty cycle.**

Youtube Link: <https://youtu.be/NIeZBCqkJB0>

**Modified Code:**

//task01: toggle of the GPIO at 2 Hz using Timer0 with 75% duty cycle

#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"

#include "driverlib/pwm.h"

#include "driverlib/pin\_map.h"

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

**{**

uint32\_t ui32Period**;**

SysCtlClockSet**(**

SYSCTL\_SYSDIV\_5 **|** SYSCTL\_USE\_PLL **|** SYSCTL\_XTAL\_16MHZ

**|** SYSCTL\_OSC\_MAIN**);** //40MHz clk

SysCtlPWMClockSet**(**SYSCTL\_PWMDIV\_1**);** //PWM match system clock

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

SysCtlPeripheralEnable**(**SYSCTL\_PERIPH\_PWM1**);** //enable PWM peripheral

SysCtlPeripheralEnable**(**SYSCTL\_PERIPH\_TIMER0**);** //enable timer0 peripheral

TimerConfigure**(**TIMER0\_BASE**,** TIMER\_CFG\_PERIODIC**);** //32-bit periodic mode timer0

GPIOPinConfigure**(**GPIO\_PF2\_M1PWM6**);** //configure PF2(B) as PWM6

GPIOPinTypePWM**(**GPIO\_PORTF\_BASE**,** GPIO\_PIN\_2**);** //set PF2 pin type as PWM

PWMGenConfigure**(**PWM1\_BASE**,** PWM\_GEN\_3**,**

PWM\_GEN\_MODE\_DOWN **|** PWM\_GEN\_MODE\_NO\_SYNC**);** //countdown non-synchronous mode

ui32Period = (SysCtlClockGet() / 2) / 2; //2Hz period

TimerLoadSet(TIMER0\_BASE, TIMER\_A, ui32Period - 1); //account for zero count

PWMGenPeriodSet(PWM1\_BASE, PWM\_GEN\_3, 100); //set period to 100

PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_6, 75); //75% duty cycle

PWMGenEnable(PWM1\_BASE, PWM\_GEN\_3); //enable PWM generator 3 for PWM6/7

IntEnable(INT\_TIMER0A); //enable vector associated w/ timer0a

TimerIntEnable(TIMER0\_BASE, TIMER\_TIMA\_TIMEOUT); //enable timer interrupt

IntMasterEnable(); //master interrupt enable

TimerEnable(TIMER0\_BASE, TIMER\_A); //enable timer

while (1)

{

}

}

void Timer0IntHandler(void)

{

TimerIntClear(TIMER0\_BASE, TIMER\_TIMA\_TIMEOUT); // Clear the timer interrupt

// Read the current state of the GPIO pin and write back the opposite state

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

{

PWMOutputState(PWM1\_BASE, PWM\_OUT\_6\_BIT, false); //turn off LED with PWM

}

else

{

PWMOutputState(PWM1\_BASE, PWM\_OUT\_6\_BIT, true); //turn on LED with PWM

}

}

void PFSW2IntHandler(void)

{

}

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

**Task 02a: toggle of the GPIO at 2 Hz using Timer0 with 75% duty cycle, include a GPIO Interrupt from switch SW2 to turn ON and the LED for 1.5s.**

Youtube Link: <https://youtu.be/Y4E9fMacTvs>

**Modified Code:**

//task02: toggle of the GPIO at 2 Hz using Timer0 with 75% duty cycle,include

//a GPIO Interrupt from switch SW2 to turn ON and the LED for 1.5s.

#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"

#include "driverlib/pwm.h"

#include "driverlib/pin\_map.h"

#include "driverlib/rom.h"

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

**{**

uint32\_t ui32Period**;**

SysCtlClockSet**(**

SYSCTL\_SYSDIV\_5 **|** SYSCTL\_USE\_PLL **|** SYSCTL\_XTAL\_16MHZ

**|** SYSCTL\_OSC\_MAIN**);** //40MHz clk

SysCtlPWMClockSet**(**SYSCTL\_PWMDIV\_1**);** //PWM match system clock

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

SysCtlPeripheralEnable**(**SYSCTL\_PERIPH\_PWM1**);** //enable PWM peripheral

SysCtlPeripheralEnable**(**SYSCTL\_PERIPH\_TIMER0**);** //enable timer0 peripheral

// unlock the GPIOLOCK register for PF0. (pg. 70 workbook)

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

HWREG**(**GPIO\_PORTF\_BASE **+** GPIO\_O\_CR**)** **|=** 0x01**;** //unlock PF0(SW2)

HWREG**(**GPIO\_PORTF\_BASE **+** GPIO\_O\_LOCK**)** **=** 0**;**

TimerConfigure(TIMER0\_BASE, TIMER\_CFG\_PERIODIC); //32-bit periodic mode timer0

GPIOPinTypeGPIOInput(GPIO\_PORTF\_BASE, GPIO\_PIN\_0); //set SW2 as input

GPIOPadConfigSet(GPIO\_PORTF\_BASE, GPIO\_PIN\_0, GPIO\_STRENGTH\_2MA,

GPIO\_PIN\_TYPE\_STD\_WPU); //enable weak pullup resistor for PF4

GPIOPinConfigure(GPIO\_PF2\_M1PWM6); //configure PF2(B) as PWM6

GPIOPinTypePWM(GPIO\_PORTF\_BASE, GPIO\_PIN\_2); //set PF2 pin type as PWM

PWMGenConfigure(PWM1\_BASE, PWM\_GEN\_3,

PWM\_GEN\_MODE\_DOWN | PWM\_GEN\_MODE\_NO\_SYNC); //countdown non-synchronous mode

ui32Period = (SysCtlClockGet() / 2) / 2; //2Hz period

TimerLoadSet(TIMER0\_BASE, TIMER\_A, ui32Period - 1); //account for zero count

PWMGenPeriodSet(PWM1\_BASE, PWM\_GEN\_3, 100); //set period to 100

PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_6, 75); //75% duty cycle

PWMGenEnable(PWM1\_BASE, PWM\_GEN\_3); //enable PWM generator 3 for PWM6/7

IntEnable(INT\_TIMER0A); //enable vector associated w/ timer0a

IntEnable(INT\_GPIOF); //enable interrupt vector associated w/ GPIO

TimerIntEnable(TIMER0\_BASE, TIMER\_TIMA\_TIMEOUT); //enable timer interrupt

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

IntMasterEnable(); //master interrupt enable

TimerEnable(TIMER0\_BASE, TIMER\_A); //enable timer

while (1)

{

}

}

void Timer0IntHandler(void)

{

TimerIntClear(TIMER0\_BASE, TIMER\_TIMA\_TIMEOUT); // Clear the timer interrupt

// Read the current state of the GPIO pin and write back the opposite state

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

{

PWMOutputState(PWM1\_BASE, PWM\_OUT\_6\_BIT, false); //turn off LED with PWM

}

else

{

PWMOutputState(PWM1\_BASE, PWM\_OUT\_6\_BIT, true); //turn on LED with PWM

}

}

void PFSW2IntHandler(void)

{

TimerIntDisable(TIMER0\_BASE, TIMER\_TIMA\_TIMEOUT); //disable timer interrupt

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

PWMOutputState(PWM1\_BASE, PWM\_OUT\_6\_BIT, true); //turn on LED with PWM//turn on LED

SysCtlDelay(20000000); //delay 1.5s (1.5=20e6\*3\*[1/40e6])

TimerIntEnable(TIMER0\_BASE, TIMER\_TIMA\_TIMEOUT); //enable timer interrupt

}