**Date Submitted:**

**Task 00: Execute provided code**

**Youtube Link: https://youtu.be/AKoXC8E-CZ0**

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

**Task 01:**

Youtube Link: https://youtu.be/ZH45VZsYGY8

**Modified Schematic (if applicable):**

**Modified Code:**

ui32Period = (SysCtlClockGet() / 2) / 3;

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

**Task 02:**

Youtube Link:

**Modified Schematic (if applicable):**

**Modified Code:**

#include <stdint.h>

#include <stdbool.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 "inc/tm4c123gh6pm.h"

void timer1A\_delaySec(int ttime);

void PortEPin0IntHandler(void);

int main(void)

{

uint32\_t ui32Period;

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

SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOF);

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

SysCtlPeripheralEnable(SYSCTL\_PERIPH\_TIMER0);

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

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

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

IntEnable(INT\_TIMER0A);

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

IntMasterEnable();

TimerEnable(TIMER0\_BASE, TIMER\_A);

SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOE);

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

HWREG(GPIO\_PORTE\_BASE+GPIO\_O\_CR) |= GPIO\_PIN\_7;

GPIOPinTypeGPIOInput(GPIO\_PORTE\_BASE,GPIO\_PIN\_0);

//enables a specific event within the GPIO to generate an interrupt.

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

//sets interrupt to rising edge on GPIO

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

//enables the specific vector associated with GPIOF.

IntPrioritySet(GPIO\_INT\_PIN\_0, 0);

IntEnable(INT\_GPIOF);

while(1)

{

PortEPin0IntHandler();

}

}

void PortEPin0IntHandler(void)

{

// Clear the GPIO interrupt

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

// Read the current state of the GPIO pin and

// write back the opposite state

if(GPIOPinRead(GPIO\_PORTE\_BASE, GPIO\_PIN\_0) != GPIO\_PIN\_0)

{

GPIOPinWrite(GPIO\_PORTF\_BASE, GPIO\_PIN\_1|GPIO\_PIN\_2|GPIO\_PIN\_3, 14);

timer1A\_delaySec(3);

}

else

{

GPIOPinWrite(GPIO\_PORTF\_BASE, GPIO\_PIN\_1, 2);

timer1A\_delaySec(3);

}

}

void timer1A\_delaySec(int ttime)

{

int i;

SYSCTL\_RCGCTIMER\_R |= 2;

TIMER1\_CTL\_R = 0;

TIMER1\_CFG\_R = 0x04;

TIMER1\_TAMR\_R = 0x02;

TIMER1\_TAILR\_R = 64000 - 1;

TIMER1\_TAPR\_R = 250 - 1;

TIMER1\_ICR\_R = 0x1;

TIMER1\_CTL\_R |= 0x01;

for( i = 0; i < ttime; i++){

while((TIMER1\_RIS\_R & 0x1) == 0);

TIMER1\_ICR\_R = 0x1;

}

}

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(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);

}

}

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