**Date Submitted: 10/4/18**

**Task 00: Execute provided code**

Youtube Link: <https://www.youtube.com/watch?v=6SY7EZpk408>

**#include** <stdint.h>

**#include** <stdbool.h>

**#include** "inc/tm4c123gh6pm.h"

**#include** "inc/hw\_memmap.h"

**#include** "inc/hw\_types.h"

**#include** "driverlib/sysctl.h"

**#include** "driverlib/interrupt.h"

**#include** "driverlib/gpio.h"

**#include** "driverlib/timer.h"

**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);

/\* GPIO running at 10MHz

\* (Clock Speed / Desired Clock Speed) \*( duty cycle)

\*/

ui32Period = (**SysCtlClockGet**() / 10 ) / 2;

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

**IntEnable**(INT\_TIMER0A);

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

**IntMasterEnable**();

**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**(**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);

}

}

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

**Task 01: Toggle of GPIO at 2Hz w/ 75% duty cycle**

Youtube Link: <https://www.youtube.com/watch?v=t9jexBASO2E>

**#include** <stdint.h>

**#include** <stdbool.h>

**#include** "inc/tm4c123gh6pm.h"

**#include** "inc/hw\_memmap.h"

**#include** "inc/hw\_types.h"

**#include** "driverlib/sysctl.h"

**#include** "driverlib/interrupt.h"

**#include** "driverlib/gpio.h"

**#include** "driverlib/timer.h"

**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);

/\* GPIO running at 10MHz

\* (Clock Speed / Desired Clock Speed) \*( duty cycle)

\*/

ui32Period = (**SysCtlClockGet**() / 2) \* 3 / 4;

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

**IntEnable**(INT\_TIMER0A);

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

**IntMasterEnable**();

**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**(**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);

}

}

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

**Task 02:**

Youtube Link: <https://www.youtube.com/watch?v=kAZ0VSxhEgc>

**#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"

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

{

uint32\_t ui32Period;

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

**SysCtlPeripheralEnable**(SYSCTL\_PERIPH\_GPIOF); // enable port F

**SysCtlPeripheralEnable**(SYSCTL\_PERIPH\_TIMER0); // Enable Timer0

**SysCtlPeripheralEnable**(SYSCTL\_PERIPH\_TIMER1); // Enable Timer1

/\* Reconfigure SW2 \*/

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

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

/\* enable SW2 as an interrupt\*/

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

**GPIOPinTypeGPIOInput**(GPIO\_PORTF\_BASE, GPIO\_PIN\_0);

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

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

**IntEnable**(INT\_GPIOF);

/\*TIMER0 INTERRUPT\*/

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

/\* GPIO running at 2MHz

\* (Clock Speed / Desired Clock Speed) \*( duty cycle)

\*/

ui32Period = (**SysCtlClockGet**() / 2 )\* 3 / 4;

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

**IntEnable**(INT\_TIMER0A);

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

**IntMasterEnable**();

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

**while**(1)

{

}

}

**void** **Timer1DelayIntHandler**(**void**)

{

**TimerDisable**(TIMER1\_BASE, TIMER\_A); // Turnoff Timer1A

**TimerIntClear**(TIMER1\_BASE,TIMER\_TIMA\_TIMEOUT); // Clear Timer1A

**IntDisable**(INT\_TIMER1A); // Turn off interrupt Timer1A

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

**IntEnable**(INT\_GPIOF); // Turn on GPIO Port F interrupt

**IntEnable**(INT\_TIMER0A); // Turn on interrupt Timer0A

}

**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);

}

}

**void** **PortFPin0IntHandler**(**void**)

{

**IntDisable**(INT\_GPIOF); // Disable Port F interrupt

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

**IntDisable**(INT\_TIMER0A); // Disable Timer0A interrupt

**IntEnable**(INT\_TIMER1A); // Enable Timer1A interrupt

**TimerLoadSet**(TIMER1\_BASE, TIMER\_A, **SysCtlClockGet**()\*1.5);// Set Delay to 1.5s

**IntEnable**(INT\_TIMER1A); // Turn on interrupt Timer1A

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

**TimerEnable**(TIMER1\_BASE, TIMER\_A); // Turn on Timer1A

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

}

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