**Date Submitted: 09/28/19**

**Task 00:**

**Youtube Link:**

<https://www.youtube.com/watch?v=JRWcje1kyH4>

**Execute provided code**

**#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; //unsigned 32 bit integer

//Clock runs at 40MHz

**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**() / 10) / 2;//10Hz and 50% DC

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

//Enabling Interrupt

**IntEnable**(INT\_TIMER0A);

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

**IntMasterEnable**();

//Will start the timer and interrupts will begin triggering on the timeouts

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

//The toggling of the GPIO will happen in the interrupt service routine

**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))

{

//turn off LEDs

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

}

**else**

{

//lights blue LED

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

}

}

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

**Task 01:**

Youtube Link:

<https://www.youtube.com/watch?v=u0CFcncRoSI>

**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** "driverlib/sysctl.h"

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

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

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

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

{

uint32\_t ui32Period; //unsigned 32 bit integer

//Clock runs at 40MHz

**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**() / 10)/2;//10Hz and 50% DC

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

//Enabling Interrupt

**IntEnable**(INT\_TIMER0A);

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

**IntMasterEnable**();

//Will start the timer and interrupts will begin triggering on the timeouts

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

//The toggling of the GPIO will happen in the interrupt service routine

**while**(1)

{

}

}

**void** **Timer0IntHandler**(**void**)

{

uint32\_t ui32Period1, ui32Period2; //unsigned 32 bit integer

// 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))

{

//turn off LEDs

ui32Period1 = (**SysCtlClockGet**() / 10) \* .57;//10Hz and 57% DC

**TimerLoadSet**(TIMER0\_BASE, TIMER\_A, ui32Period1 -1);

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

}

**else**

{

//lights blue LED

ui32Period2 = (**SysCtlClockGet**() / 10) \* .43;//10Hz and 43% DC

**TimerLoadSet**(TIMER0\_BASE, TIMER\_A, ui32Period2 -1);

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

}

}

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

**Task 02:**

Youtube Link:

<https://www.youtube.com/watch?v=3O5Fmm4Ijf8>

**Modified Schematic (if applicable):**



**Modified Code:**

**#define** SW2 GPIO\_PIN\_0

**#include** <stdint.h>

**#include** <stdbool.h>

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

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

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

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

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

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

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

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

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

uint32\_t secDelay;

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

{

uint32\_t ui32Period; //unsigned 32 bit integer

//Clock runs at 40MHz

**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);

//Unlocking SW2

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

HWREG(GPIO\_PORTF\_BASE + GPIO\_O\_CR) = 0x1;

//Switch Interrupt

**GPIOPinTypeGPIOInput**(GPIO\_PORTF\_BASE, SW2); //enabling switch as an inout

**GPIOPadConfigSet**(GPIO\_PORTF\_BASE, SW2, GPIO\_STRENGTH\_2MA, GPIO\_PIN\_TYPE\_STD\_WPU); //turn weak pull-ups on

**GPIOIntTypeSet**(GPIO\_PORTF\_BASE,SW2,GPIO\_RISING\_EDGE); //sets interrupt to rising edge on GPIO

**GPIOIntEnable**(GPIO\_PORTF\_BASE, GPIO\_INT\_PIN\_0); //enables a specific event within the GPIO to generate an interrupt

//Timer 0 enabling and config

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

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

//Timer 1 enabling and config

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

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

//Timer0 value

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

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

//Timer1 value

secDelay = **SysCtlClockGet**();

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

**TimerLoadSet**(TIMER1\_BASE, TIMER\_A, secDelay);

//Enabling Interrupt

**IntEnable**(INT\_TIMER0A);

**IntEnable**(INT\_TIMER1A);

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

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

**IntMasterEnable**();

//Will start the timer and interrupts will begin triggering on the timeouts

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

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

//Port F Interrupt for SW1

**IntEnable**(INT\_GPIOF); //enables the specific vector associated with GPIO

//Will allow interrupts to occur without other occurrences in code

**while**(1)

{

}

}

**void** **Timer0IntHandler**(**void**)

{

uint32\_t ui32Period1, ui32Period2; //unsigned 32 bit integer

// 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))

{

//turn off LEDs

ui32Period1 = (**SysCtlClockGet**() / 10) \* .57;//10Hz and 57% DC

**TimerLoadSet**(TIMER0\_BASE, TIMER\_A, ui32Period1 -1);

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

}

**else**

{

//lights blue LED

ui32Period2 = (**SysCtlClockGet**() / 10) \* .43;//10Hz and 43% DC

**TimerLoadSet**(TIMER0\_BASE, TIMER\_A, ui32Period2 -1);

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

}

}

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

{

**TimerIntClear**(TIMER1\_BASE,TIMER\_A); //Required to launch next interrupt

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

}

**void** **IntPortFHandler**(**void**)

{

**TimerDisable**(TIMER0\_BASE, TIMER\_A); //stop timer0

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

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

}

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