**Date Submitted: 9/28/2019**

**Task 00: Execute provided code**

**Youtube Link:**

<https://www.youtube.com/watch?v=x36VptR0c30>

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

**Task 01:**

Verification:



Youtube Link:

<https://www.youtube.com/watch?v=1Mmxl5n4F08>

**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;**

**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;**

**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)**

**{**

**uint32\_t ui32Period\_high,ui32Period\_low;**

**// 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))**

**{**

**ui32Period\_low = (SysCtlClockGet() / 10) \* 0.57;**

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

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

**}**

**else**

**{**

**ui32Period\_high = (SysCtlClockGet() / 10) \* 0.43;**

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

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

**}**

**}**

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

**Task 02:**

Could not use SW2 because of NMI default making it locked. Tried to unlock it but it ended up making the code not work so I used SW1 instead.

Verification:



Youtube Link:

<https://www.youtube.com/watch?v=2OL3c6F08yE>

**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"**

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

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

**#include "driverlib/pin\_map.h"**

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

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

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

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

**uint32\_t ui32High;**

**uint32\_t ui32Low;**

**uint32\_t ui32Delay\_1s;**

**int main(void){**

**uint32\_t ui32Period; // This variable is used to set the timer**

**SysCtlClockSet(SYSCTL\_SYSDIV\_5|SYSCTL\_USE\_PLL|SYSCTL\_XTAL\_16MHZ|SYSCTL\_OSC\_MAIN); // 40MHz System clock ((400MHz/(default 2))/ 5(SYSCTL\_SYSDIV\_5))**

**SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOF); //enable port F (PF0...PF4)**

**// The following three lines of code unlock the GPIOLOCK register for PF0 using direct Register Programming**

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

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

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

**GPIOPinTypeGPIOInput(GPIO\_PORTF\_BASE, GPIO\_PIN\_0); // set PF0 (switch 2)**

**GPIOPadConfigSet(GPIO\_PORTF\_BASE ,GPIO\_PIN\_0,GPIO\_STRENGTH\_2MA,GPIO\_PIN\_TYPE\_STD\_WPU); // disables pull of resistor of PF0 with a 2mA output drive strength**

**GPIOIntTypeSet(GPIO\_PORTF\_BASE,GPIO\_PIN\_0,GPIO\_FALLING\_EDGE); // sets PF0 as falling edge**

**//GPIOIntRegister(GPIO\_PORTF\_BASE,PortFIntHandler); // registers an interrupt handler for a GPIO port, that is, calls the second parameter function**

**GPIOIntEnable(GPIO\_PORTF\_BASE, GPIO\_INT\_PIN\_0); // enables interupts from PF0 (switch 2)**

**GPIOPinTypeGPIOOutput(GPIO\_PORTF\_BASE, GPIO\_PIN\_1|GPIO\_PIN\_2|GPIO\_PIN\_3); // set GPIO\_PIN\_1,GPIO\_PIN\_2,GPIO\_PIN\_3 as base (2 clock cycles needed to toggle) output pins**

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

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

**TimerConfigure(TIMER0\_BASE, TIMER\_CFG\_PERIODIC); // configure timer0 as periodic (Full-width periodic timer)**

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

**ui32Delay\_1s = (SysCtlClockGet());**

**ui32Period = (SysCtlClockGet() / 10) / 2; // 40MHz/10/2 gives 2MHz timer (500ns)**

**TimerLoadSet(TIMER0\_BASE, TIMER\_A, ui32Period -1); // set the timer load value of timer A**

**TimerLoadSet(TIMER1\_BASE, TIMER\_A, ui32Delay\_1s);**

**IntEnable(INT\_TIMER0A); // Enable Timer0A interrupts**

**IntEnable(INT\_TIMER1A);**

**TimerIntEnable(TIMER0\_BASE, TIMER\_TIMA\_TIMEOUT); // only enables Timer A timeout interrupt**

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

**IntMasterEnable();**

**TimerEnable(TIMER0\_BASE, TIMER\_A);// enables timer A**

**IntEnable(INT\_GPIOF);**

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

**while(1){**

**}**

**}**

**// make sure Timer 0 subtimer A default handler is replaced with "Timer0IntHandler" in startup\_ccs.c**

**// also make sure you declare "extern void Timer0IntHandler(void);" bellow "extern void \_c\_int00(void);" in startup\_ccs.c**

**void Timer0IntHandler(void){**

**uint32\_t ui32Period\_high,ui32Period\_low;**

**// 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)){**

**ui32Period\_low = (SysCtlClockGet() / 10) \* 0.57;**

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

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

**}**

**else{**

**ui32Period\_high = (SysCtlClockGet() / 10) \* 0.43;**

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

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

**}**

**}**

**void Timer1IntHandler(void){**

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

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

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

**}**

**void PortFIntHandler(void){**

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

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

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

**}**