Lab 4

Task 1

```
//This program will give insight on the use of interrupts and timers on the TIVA C
board


#include <stdint.h>
#include <stdbool.h>
#include "inc/tm4c123gh6pm.h" //interrupt definitions and register assignments
#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 ()
{
	uint32_t ui32Period;

	//Sets clock to run at 40 MHz
	SysCtlClockSet(SYSCTL_SYSDIV_5|SYSCTL_USE_PLL|SYSCTL_XTAL_16MHZ|SYSCTL_OSC_MAI
N);
	//Sets the clock to 40MHz => 400MHz (PLL) / (5*2)

	//Configures the GPIO
	SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOF);
	GPIOPinTypeGPIOOutput(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3);

	//Configures Timer0
	SysCtlPeripheralEnable(SYSCTL_PERIPH_TIMER0);
	TimerConfigure(TIMER0_BASE, TIMER_CFG_PERIODIC);

	//Generates a period for the GPIO to run at 10MHz =>
(SystemClockSpeed/DesiredSpeed)/2
	ui32Period = (SysCtlClockGet() / 10) / 2;
	TimerLoadSet(TIMER0_BASE, TIMER_A, ui32Period -1); //subtract period by 1
since it starts at 0

	//Enables the interrupt for TIMER0
	IntEnable(INT_TIMER0A);
	TimerIntEnable(TIMER0_BASE, TIMER_TIMA_TIMEOUT);
	IntMasterEnable();

	//Enables TIMER
	TimerEnable(TIMER0_BASE, TIMER_A);

	while (1)
	{}

}

void Timer0IntHandler(void)
```

```
//This is the interrupt handler that will be called when the Timer reaches the value
specified

{
	// 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))
	//Checks if there is a value written to PORTF.2, write 0 if there is
	{
		GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3, 0);
	}
	else
	//else write 4 to PORTF to set PORTF.2 to 1
	{
		GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_2, 4);
	}
}
```

Task 2

```
//This program will give insight on the use of interrupts and timers on the TIVA C
board


#include <stdint.h>
#include <stdbool.h>
#include "inc/tm4c123gh6pm.h" //interrupt definitions and register assignments
#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 ()
{
	uint32_t ui32Period;

	//Sets clock to run at 40 MHz
	SysCtlClockSet(SYSCTL_SYSDIV_5|SYSCTL_USE_PLL|SYSCTL_XTAL_16MHZ|SYSCTL_OSC_MAI
N);
	//Sets the clock to 40MHz => 400MHz (PLL) / (5*2)

	//Configures the GPIO
	SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOF);
	GPIOPinTypeGPIOOutput(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3);

	//Configures Timer0
	SysCtlPeripheralEnable(SYSCTL_PERIPH_TIMER0);
	TimerConfigure(TIMER0_BASE, TIMER_CFG_PERIODIC);
	//Generates a period for the GPIO to run at 50MHz =>
(SystemClockSpeed/DesiredSpeed)/(duty cycle)
	ui32Period = (SysCtlClockGet() / 50) / 5;	//This is set to create a 50Hz
pulse with 20% duty cycle
	TimerLoadSet(TIMER0_BASE, TIMER_A, ui32Period -1); //subtract period by 1
since it starts at 0

	//Enables the interrupt for TIMER0
	IntEnable(INT_TIMER0A);
	TimerIntEnable(TIMER0_BASE, TIMER_TIMA_TIMEOUT);
	IntMasterEnable();

	//Enables TIMER
	TimerEnable(TIMER0_BASE, TIMER_A);

	while (1)
	{}

}

void Timer0IntHandler(void)
//This is the interrupt handler that will be called when the Timer reaches the value
specified

{
	// 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))
	//Checks if there is a value written to PORTF.2, write 0 if there is
	{
		GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3, 0);
	}
	else
	//else write 4 to PORTF to set PORTF.2 to 1
	{
		GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_2, 4);
	}
}
```