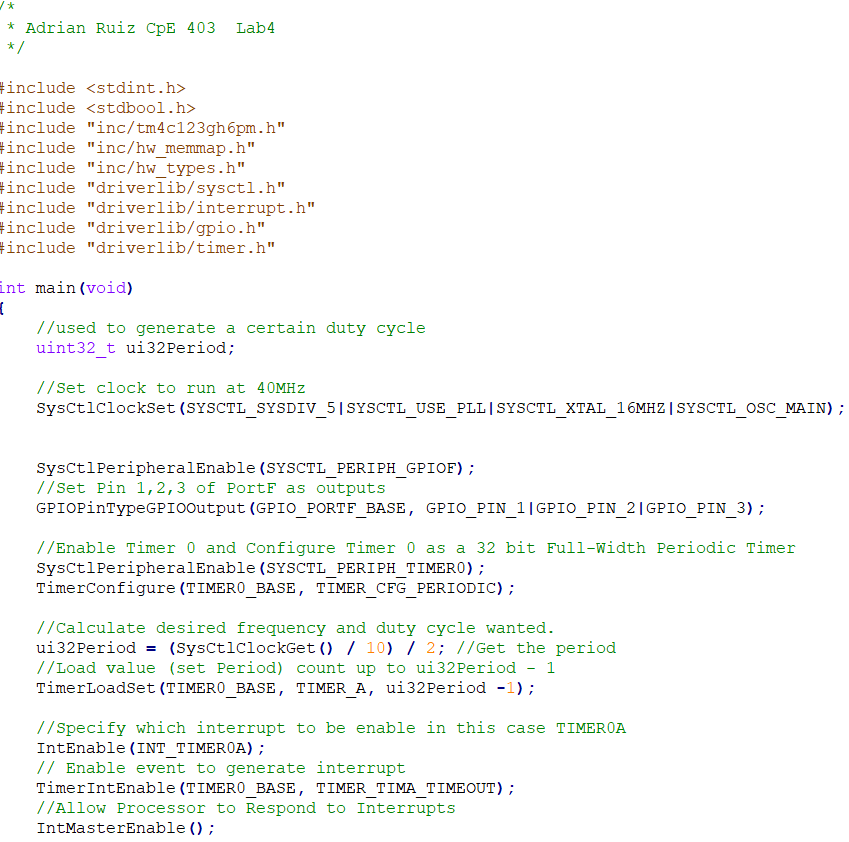
Adrian Ruiz

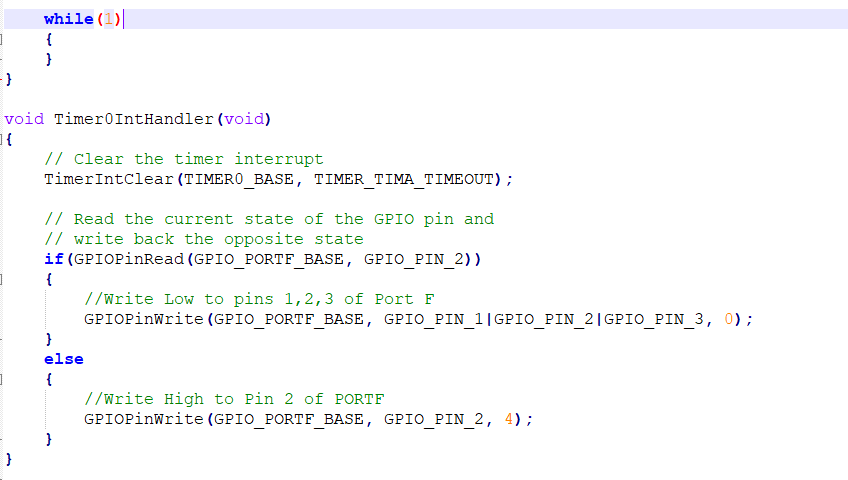
CPE 403-1001

Lab 4

Task 00:

Code:



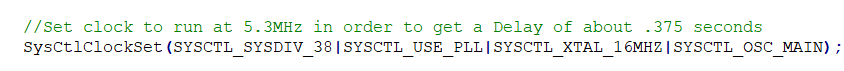


Task 01:

Snip of Altered code:



To generate a 2Hz clock, I divided the System clock by 2 and by 2 again to get a 2Hz 50% duty cycle.

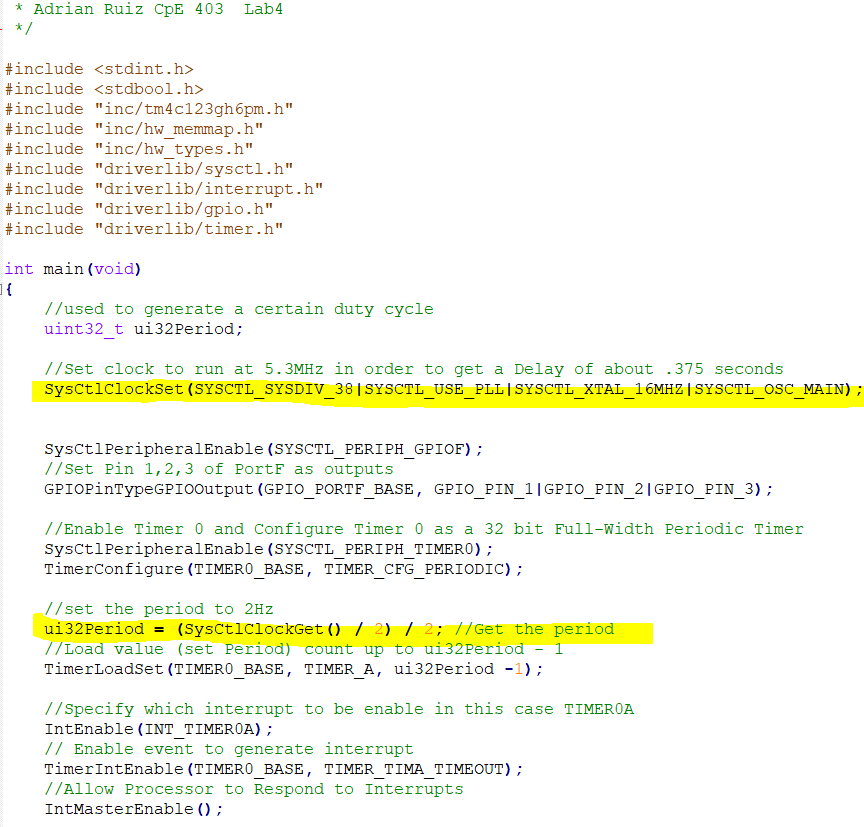


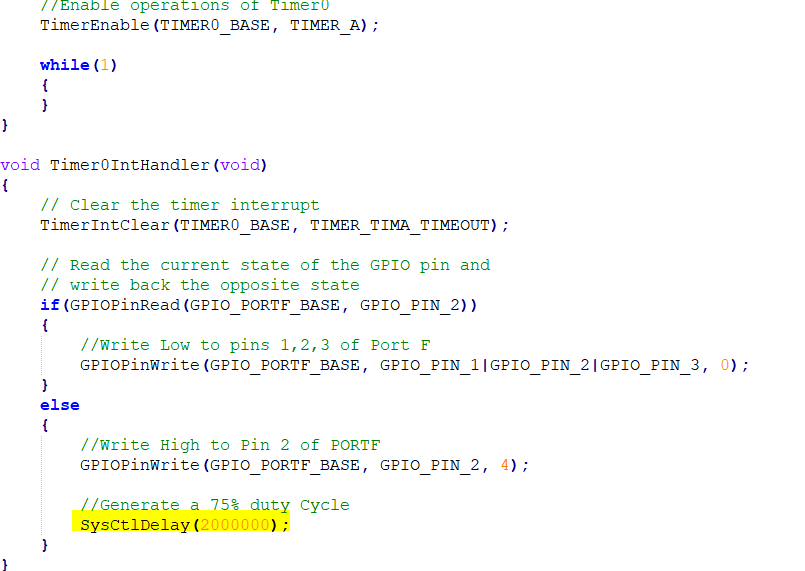
To get a .375 delay need for the duty cycle, I changed the clock to run at 5.3 Mhz in order to get StyCtlDelay(2000000) to equal about .375.



By calling this inside the Timer interrupt when the clock cycle is high, the duty cycle gets shifted to about 75%.

Code:





Task 02:

Altered Code:

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

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

This unlocks the pin 0 from portF, SW2, in order to be used as an input.

//GPIO\_PIN\_4, SW2, configured to be an input

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

//Enable switch interrupt

IntEnable**(**INT\_GPIOF**);**

//How the interrupt will be called and enabling event for interrupt

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

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

This snippet sets SW2 to be an input and enables the GPIO Interrupt. SW2 becomes a the trigger for the interrupt.

void GPIO\_PF\_IntHandler**(**void**)**

**{**

IntMasterDisable**();**

//Clear the GPIOPin0

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

//Enable Timer1 and configure it as a 32 periodic timer

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

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

TimerLoadSet**(**TIMER1\_BASE**,** TIMER\_A**,** SysCtlClockGet**()** **-**0.5**);**

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

//Write Low to all LEDs and turn on a single LED

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

GPIOPinWrite**(**GPIO\_PORTF\_BASE**,** GPIO\_PIN\_3**,** 8**);**

//creat a 1.5 sec delay

int i **=** 0**;**

**for(**i **=** 0**;** i**<** 3**;** **++**i**)**

**{**

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

**while(**1**)**

**{**

**if(**TimerIntStatus**(**TIMER1\_BASE**,**true**)&**TIMER\_TIMA\_TIMEOUT**==**TIMER\_TIMA\_TIMEOUT**)**

**{**

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

**break;**

**}**

**}**

**}**

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

IntMasterEnable**();**

**}**

This is the GPIO Interrupt Handler. When the button is pressed, it disables the blinking LED and makes the Green LED blink for about 1.5 seconds.

Code:

/\*

\* Adrian Ruiz CpE 403 Lab4

\* I left the blinking part in. The driver libraries wouldn't

\* link properly if i tried taking it out

\*/

#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\_gpio.h"

#include "driverlib/pin\_map.h"

#include "driverlib/rom\_map.h"

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

**{**

//used to generate a certain duty cycle

uint32\_t ui32Period**;**

//Set clock to run at 40MHz

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

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

//Set Pin 1,2,3 of PortF as outputs

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

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

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

//GPIO\_PIN\_4, SW2, configured to be an input

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

//Enable switch interrupt

IntEnable**(**INT\_GPIOF**);**

//How the interrupt will be called and enabling event for interrupt

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

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

//Enable Timer 0 and Configure Timer 0 as a 32 bit Full-Width Periodic Timer

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

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

//set the period to 2Hz

ui32Period **=** **(**SysCtlClockGet**()** **/** 2**)** **/** 2**;** //Get the period

//Load value (set Period) count up to ui32Period - 1

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

//Specify which interrupt to be enable in this case TIMER0A

IntEnable**(**INT\_TIMER0A**);**

// Enable event to generate interrupt

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

//Allow Processor to Respond to Interrupts

IntMasterEnable**();**

//Enable operations of Timer0

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**))**

**{**

//Write Low to pins 1,2,3 of Port F

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

**}**

**else**

**{**

//Write High to Pin 2 of PORTF

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

**}**

**}**

void GPIO\_PF\_IntHandler**(**void**)**

**{**

IntMasterDisable**();**

//Clear the GPIOPin0

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

//Enable Timer1 and configure it as a 32 periodic timer

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

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

TimerLoadSet**(**TIMER1\_BASE**,** TIMER\_A**,** SysCtlClockGet**()** **-**0.5**);**

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

//Write Low to all LEDs and turn on a single LED

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

GPIOPinWrite**(**GPIO\_PORTF\_BASE**,** GPIO\_PIN\_3**,** 8**);**

//creat a 1.5 sec delay

int i **=** 0**;**

**for(**i **=** 0**;** i**<** 3**;** **++**i**)**

**{**

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

**while(**1**)**

**{**

**if(**TimerIntStatus**(**TIMER1\_BASE**,**true**)&**TIMER\_TIMA\_TIMEOUT**==**TIMER\_TIMA\_TIMEOUT**)**

**{**

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

**break;**

**}**

**}**

**}**

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

IntMasterEnable**();**

**}**