Task 01: Change the toggle of the GPIO at 50Hz and at 75% duty cycle and verify.

The following line was the only change from the provided code.

ui32Period = (SysCtlClockGet() / 50) \* .75;

//Calculate period by getting number of clock cycles div by desired

//frequency (50Hz) and then multiply by (75%) duty cycle

Task 02: Include a GPIO Interrupt to Task 02 from switch SW2 to turn ON and the LED for 2

sec. Use a Timer1 to calculate the 2 sec delay. The toggle of the GPIO is suspended when executing the interrupt.

...

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

//40MHz system clock

SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOF);

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;

//unlocks use of sw2

GPIODirModeSet(GPIO\_PORTF\_BASE, GPIO\_PIN\_0, GPIO\_DIR\_MODE\_IN);

//configures pf 4 as inputs

GPIOPadConfigSet(GPIO\_PORTF\_BASE, GPIO\_PIN\_0, GPIO\_STRENGTH\_2MA, GPIO\_PIN\_TYPE\_STD\_WPU);

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

//Enable GPIO peripherals and configure pins connected to LEDs

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

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

IntEnable(INT\_GPIOF);

SysCtlPeripheralEnable(SYSCTL\_PERIPH\_TIMER0);

//Enable clock to peripheral

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

//Timer 0 as 32 bit timer in periodic mode

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

ui32Period = (SysCtlClockGet() / .5);/// 50) \* .75;

//Calculate period by getting number of clock cycles div by desired frequency div and 75% duty cycle

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

//Load period(minus 1) into Timers Interval LOAD Reg

IntMasterEnable();

//Enable API for all interrupts

while(1)

{

}

}

void Timer0IntHandler(void)

{

//Clear the timer interrupt

TimerIntClear(TIMER0\_BASE, TIMER\_TIMA\_TIMEOUT);

TimerIntDisable(TIMER0\_BASE, TIMER\_TIMA\_TIMEOUT);

GPIOPinWrite(GPIO\_PORTF\_BASE, GPIO\_PIN\_1, 0);

}

void PortEPin0IntHandler(void)

{

//Clear the GPIO interrupt

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

GPIOPinWrite(GPIO\_PORTF\_BASE, GPIO\_PIN\_1, 2);

TimerEnable(TIMER0\_BASE, TIMER\_A);

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

IntEnable(INT\_TIMER0A);

//Start timer and interrupts will begin triggering at timeouts

}