Nathan Hanuscin
Github root directory: https://github.com/nhanuscin/HappyFunStuff

Date Submitted: 9/27/19

Task 01:

Youtube Link: https://www.youtube.com/watch?v=g-GprItkVHA

Modified Schematic (if applicable): N/A

Modified Code:

```
int main(void)
{
    uint32_t ui32Period;
    //Set Clock to 40MHz
    SysCtlClockSet(SYSCTL_SYSDIV_5|SYSCTL_USE_PLL|SYSCTL_XTAL_16MHZ|SYSCTL_OSC_MAIN);
    //Enable PortF
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOF);
    GPIOPinTypeGPIOOutput(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3);
    //Enable Timer0A
    SysCtlPeripheralEnable(SYSCTL_PERIPH_TIMER0);
    TimerConfigure(TIMER0_BASE, TIMER_CFG_PERIODIC);
    //Set timer to 10Hz 43% duty cycle
    ui32Period =  (43*SysCtlClockGet()/10) / 100;
    TimerLoadSet(TIMER0_BASE, TIMER_A, ui32Period -1);
    //Enable timer interrupt
    IntEnable(INT_TIMER0A);
    TimerIntEnable(TIMER0_BASE, TIMER_TIMA_TIMEOUT);
    IntMasterEnable();

    TimerEnable(TIMER0_BASE, TIMER_A);

    while(1)
    {
    }
}

void Timer0IntHandler(void)
{
    uint32_t ui32Period2;
    // 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 LED and set 57% duty cycle
        GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3, 0);
        ui32Period2 =  (57*SysCtlClockGet()/10) / 100;
        TimerLoadSet(TIMER0_BASE, TIMER_A, ui32Period2 -1);
    }
    else
```

Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.

```
    {
        //Turn on LED and set 43% duty cycle
        GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_2, 4);
        ui32Period2 =  (43*SysCtlClockGet()/10) / 100;
        TimerLoadSet(TIMER0_BASE, TIMER_A, ui32Period2 -1);
    }
}
```

Task 02:

Youtube Link: https://www.youtube.com/watch?v=-HwVGsCC02s

Modified Schematic (if applicable): N/A

Modified Code:

```
void timer1A_delaySec(int ttime)
{
    int i;
    SYSCTL_RCGCTIMER_R |= 2; //Enable clock to Timer Block 1
    TIMER1_CTL_R = 0;        //Disable Timer before initialization
    TIMER1_CFG_R = 0x04;     //16-bit mode
    TIMER1_TAMR_R = 0x02;    //Period mode and down counter
    TIMER1_TAILR_R = 40000-1;//TimerA interval load value reg
    TIMER1_TAPR_R = 250-1;   //TimerA Prescaler 20MHz/250 = 80000Hz
    TIMER1_ICR_R = 0x1;      //Clear TimerA timeout flag
    TIMER1_CTL_R |= 0x01;    //Enable Timer A after initialization
    for(i = 0; i < ttime*2; i++) //Every Loop = 1 second
    {
        while((TIMER1_RIS_R & 0x1) == 0)
        {
            //wait for timer timeout
        }
        TIMER1_ICR_R = 0x1;     //Clear the timeout flag
    }
}


int main(void)
{
    //uint32_t period;
    //Set Clock to 20MHz

SysCtlClockSet(SYSCTL_SYSDIV_10|SYSCTL_USE_PLL|SYSCTL_XTAL_16MHZ|SYSCTL_OSC_MAIN);
    //SysCtlDelay(3);
    //period = SysCtlClockGet();
    //Enable PortF pin 4 for interrupt with weak pullup
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOF);
    GPIOPinTypeGPIOInput(GPIO_PORTF_BASE, GPIO_PIN_4);
    GPIOPadConfigSet(GPIO_PORTF_BASE, GPIO_PIN_4, GPIO_STRENGTH_2MA,
GPIO_PIN_TYPE_STD_WPU);
    GPIOPinTypeGPIOOutput(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3);
```

```
    GPIOIntEnable(GPIO_PORTF_BASE, GPIO_INT_PIN_4);
    GPIOIntTypeSet(GPIO_PORTF_BASE, GPIO_INT_PIN_4, GPIO_RISING_EDGE);
    //Clear LEDs initially
    GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3, 0);
    IntEnable(INT_GPIOF);

    while(1)
    {
        //wait for button to be pressed
    }
}

void PortFPin4IntHandler(void)
{
        // Clear the GPIO interrupt
        GPIOIntClear(GPIO_PORTF_BASE, GPIO_INT_PIN_4);
        SysCtlDelay(3);
        // Disable the Interrupt
        IntDisable(INT_GPIOF);
        GPIOIntDisable(GPIO_PORTF_BASE, GPIO_INT_PIN_4);
        // Turn on the LED
        GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_2, 4);
        timer1A_delaySec(1); //call delay function for timer1A
        //turn off LED
        GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_2, 0);
        //Enable the interrupt
        GPIOIntEnable(GPIO_PORTF_BASE, GPIO_INT_PIN_4);
        IntEnable(INT_GPIOF);
}
--------------------------------------------------------------------------------
```