```c
#include <stdint.h>
#include <stdbool.h>
#include "inc/hw_types.h"
#include "inc/hw_memmap.h"
#include "driverlib/sysctl.h"
#include "driverlib/gpio.h"
#include "inc/hw_gpio.h"
#include "inc/hw_ints.h"
#include "inc/hw_sysctl.h"
#include "driverlib/timer.h"


void swInt()
{
    int a=GPIOIntStatus(GPIO_PORTF_BASE,true);   //kesmenin hangi pinden geldiğini anlamak için
    GPIOIntClear(GPIO_PORTF_BASE, GPIO_PIN_0|GPIO_PIN_4);  //kesmeyi temizler

    if(a==GPIO_PIN_4){
        GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3, 8);
        SysCtlDelay(2000000);
        GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3, 0);
    }
    else if(a==GPIO_PIN_0){
        GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3, 4);
        SysCtlDelay(2000000);
        GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3, 0);
        }
}

int status=1;
void Timer0IntHandler(void)
{
    // Clear the timer interrupt
    TimerIntClear(TIMER0_BASE, TIMER_TIMA_TIMEOUT);
    GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1, 2);
    if(status==1){  //timer kesmesi geldikçe kare dalga üretecek.yani kırmızı ledi yakıp söndürecek
         GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3, 2);
         status=0;
     }
     else if(status==0){
         GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3, 0);
         status=1;
         }

}

int main(void)
 {

     // kesmede her zaman 5 fonksiyon ile ayar yapacağız.
    IntMasterEnable(); // global ayar 1

    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);


    //alttaki 3 satır kod pin 0 daki kiliti kaldırmak ve butonu kullanabilmek için
    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|GPIO_PIN_4);
    GPIOPadConfigSet(GPIO_PORTF_BASE, GPIO_PIN_0|GPIO_PIN_4, GPIO_STRENGTH_4MA, GPIO_PIN_TYPE_STD_WPU);
//pull down veya pull up dirençlerini ayarlamak için
```

```
    GPIOIntRegister(GPIO_PORTF_BASE,swInt); // çevre birimi ayar 1  //swInt buton kesmi geldiğinde işlemcinin gideceği fonskiyondur.
    GPIOIntTypeSet(GPIO_PORTF_BASE, GPIO_PIN_0|GPIO_PIN_4,GPIO_FALLING_EDGE); // çevre birimi ayar 2

    GPIOIntClear(GPIO_PORTF_BASE, GPIO_PIN_0|GPIO_PIN_4);
    GPIOIntEnable(GPIO_PORTF_BASE, GPIO_PIN_0|GPIO_PIN_4); // çevre birimi ayar 3

    IntEnable(INT_GPIOF); // global ayar 2

    GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3, 0);

        SysCtlPeripheralEnable(SYSCTL_PERIPH_TIMER0);
        TimerConfigure(TIMER0_BASE, TIMER_CFG_PERIODIC);

        uint32_t ui32Period = SysCtlClockGet()/8;   //sistem 40milyon dalgayı 1 saniye de saydığı için kare dalga için de 1/8 periyot gerektiğinden sistem saatini 8/e böldük.
        TimerLoadSet(TIMER0_BASE, TIMER_A, ui32Period - 1);   //sysctl fonksiyonu 1den itibaren saydığı için bir çıkartarak 0 a kaydırdık.
        TimerMatchSet(TIMER0_BASE, TIMER_A, 0);

        IntEnable(INT_TIMER0A);
        TimerIntEnable(TIMER0_BASE, TIMER_TIMA_TIMEOUT);
        TimerIntRegister(TIMER0_BASE, TIMER_A, Timer0IntHandler);
        IntMasterEnable();

        TimerEnable(TIMER0_BASE, TIMER_A);

    while(1)
    {

    }
}
```