Universidad del Valle de Guatemala
Digital 2
Kurt Kellner
Pablo Rene Arellano Estrada

EXPERIMENTO 2 Tiva_C TIVAWARE

PSEUDOCODIGO

```
int main(void)
{
    volatile uint32_t ui32Loop;

    // 1. Reloj del Sistema: Principal | 16 MHz | (400 MHz/2) = 200 MHz |  (200MHz/5)
= 40 MHz
    SysCtlClockSet(SYSCTL_OSC_MAIN | SYSCTL_XTAL_16MHZ |    SYSCTL_USE_PLL    |
SYSCTL_SYSDIV_5  );

    // 2a. Enable del reloj Puerto F (Leds)
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOF);

    // 2b. Verifica si hay acceso a periferico de Puerto B
    while(!SysCtlPeripheralReady(SYSCTL_PERIPH_GPIOF))
    {
    }

    // 2c. Enable GPIO para el LED (PF3). Pines de salida y digitales
    GPIOPinTypeGPIOOutput(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3);

    // 2d.  Enable del reloj del Timer0
    SysCtlPeripheralEnable(SYSCTL_PERIPH_TIMER0);

    // 2e. Verifica si hay acceso a periferico del Timer
    while(!SysCtlPeripheralReady(SYSCTL_PERIPH_TIMER0))
    {
    }

    // 2f. Timer periodico
    TimerConfigure(TIMER0_BASE,TIMER_CFG_PERIODIC);

    // 2g. Timer de 40 millones/2 = 20 millones - 1 = 19.99 millones  para que sea de
1 segundo
    TimerLoadSet(TIMER0_BASE, TIMER_A, ((SysCtlClockGet())/2)-1);

    // 2h. Interrupcion Timer0
    IntEnable(INT_TIMER0A);

    // 2i. Se habilita el Timer
    TimerEnable(TIMER0_BASE, TIMER_A);
```

```
    // 4a. Se establece que exista la interrupción por Timeout
    TimerIntEnable(TIMER0_BASE, TIMER_TIMA_TIMEOUT);

    // 4b. Se habilitan las interrupciones Globales
    IntMasterEnable();

    // 5a. Enable reloj Puerto A
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOA);

    // 5b. Enable reloj del UART
    SysCtlPeripheralEnable(SYSCTL_PERIPH_UART0);

    // 5c. Inicio Modulo UART: 115200, 8 bits de datos, 1 stop bit, No Paridad
    UARTConfigSetExpClk(UART0_BASE, SysCtlClockGet(), 115200,(UART_CONFIG_WLEN_8 |
UART_CONFIG_STOP_ONE | UART_CONFIG_PAR_NONE));

    // 5c. Enable los pines para ser perifericos
    // GPIOPinTypeUART(GPIO_PORTA_BASE, GPIO_PIN_0 | GPIO_PIN_1);
    // UARTCharPut(UART0_BASE, 'H');

    // 6a. Interrupcion para modulo UART
    IntEnable(INT_UART0);

    // 6b. Interrupcion para modulo UART
    UARTIntEnable(UART0_BASE, UART_INT_RX | UART_INT_RT);

    // Loop
    while(1)
    {

    }
}
```

```
//*****************************************************************************
//
//  3 y 4. Handler de la interrupción del TIMER 0 - Interrupcion a 0.5Hz
//
//*****************************************************************************

void Timer0IntHandler(void)
{
    // 3a. Encendido - lo apago
    if (GPIOPinRead(GPIO_PORTF_BASE, GPIO_PIN_2))
    {
        GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1 | GPIO_PIN_2 | GPIO_PIN_3, 0);
    }

    // 3b. Apagado - lo enciendo
    else
    {
        GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_2, 4);
    }

    // 3c. Clear the timer interrupt
    TimerIntClear(TIMER0_BASE, TIMER_TIMA_TIMEOUT);
}

//*****************************************************************************
//
//  6 y 7. Handler de la interrupción del TIMER 0 - Interrupcion a 0.5Hz
//
//*****************************************************************************

void UARTIntHandler(void)
{
}
```