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**)

{

}