Universidad del Valle de Guatemala

Electrónica Digital II

Ing. Pablo Mazariegos

Ing. Kurt Kellner

Sección 20

Rodrigo Díaz, 18265

Lab#7: Experimento 2, ISR, Timers y UART

# Pseudocódigo

**Parte 1**

Copiar y pegar los headers necesarios

**Parte 2**

Se configura el reloj

SysCtlClockSet(

SYSCTL\_SYSDIV\_5 | SYSCTL\_USE\_PLL | SYSCTL\_XTAL\_16MHZ

| SYSCTL\_OSC\_MAIN);

Se habilitan los periféricos en el puerto F

SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOF);

Se configuran los 3 leds del RGB como outputs

GPIOPinTypeGPIOOutput(GPIO\_PORTF\_BASE,

GPIO\_PIN\_1 | GPIO\_PIN\_2 | GPIO\_PIN\_3);

Se configura el TMR0 de 32 bits, periódico.

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

TimerEnable(TIMER0\_BASE, TIMER\_A);

**Parte 3**

Al saber la frecuencia de operación del sistema, calcular el número de ciclos para que se tenga un toggle a 1 Hz. Se debe tener un duty cycle de 50%. Si se tiene una frecuencia de 16MHz, cada 8M ciclos se cambia.

**Parte 4**

Habilitar las interrupciones en el TMR para que pasen cada 0.5Hz.

IntEnable(INT\_TIMER0A);

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

Habilitar todas las interrupciones.

IntMasterEnable();

Reemplazar el nombre del header en el archivo startup.

**Parte 5**

Habilitar el módulo UART

SysCtlPeripheralEnable(SYSCTL\_PERIPH\_UART0);

Habilitar los periféricos en el puerto A

SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOA);

Inicializar a 115200, 8 data bits, 1 stop bit, none parity

UARTConfigSetExpClk(

UART0\_BASE, SysCtlClockGet(), 115200,

(UART\_CONFIG\_WLEN\_8 | UART\_CONFIG\_STOP\_ONE | UART\_CONFIG\_PAR\_NONE));

**Parte 6**

Habilitar las interrupciones por UART

UARTIntEnable (UART0\_BASE, UART\_INT\_RX | UART\_INT\_RT);

Reemplazar el nombre del header en el archivo startup

**Parte 7**

En la interrupción, recopilar un dato.

If (dato=r) blinking de led roja

Lo mismo pasa para los otros dos leds

Si se repite, detener el blinking.