PRÁCTICA 6

INTERRUPCIONES Y TEMPORIZADORES EN LENGUAJE C

**OBJETIVOS:**

* Familiarizarse con el entorno de programación en Lenguaje C
* Comprender el manejo de interrupciones en C
* Trabajar con temporizadores y el Perro Guardián (WDT)

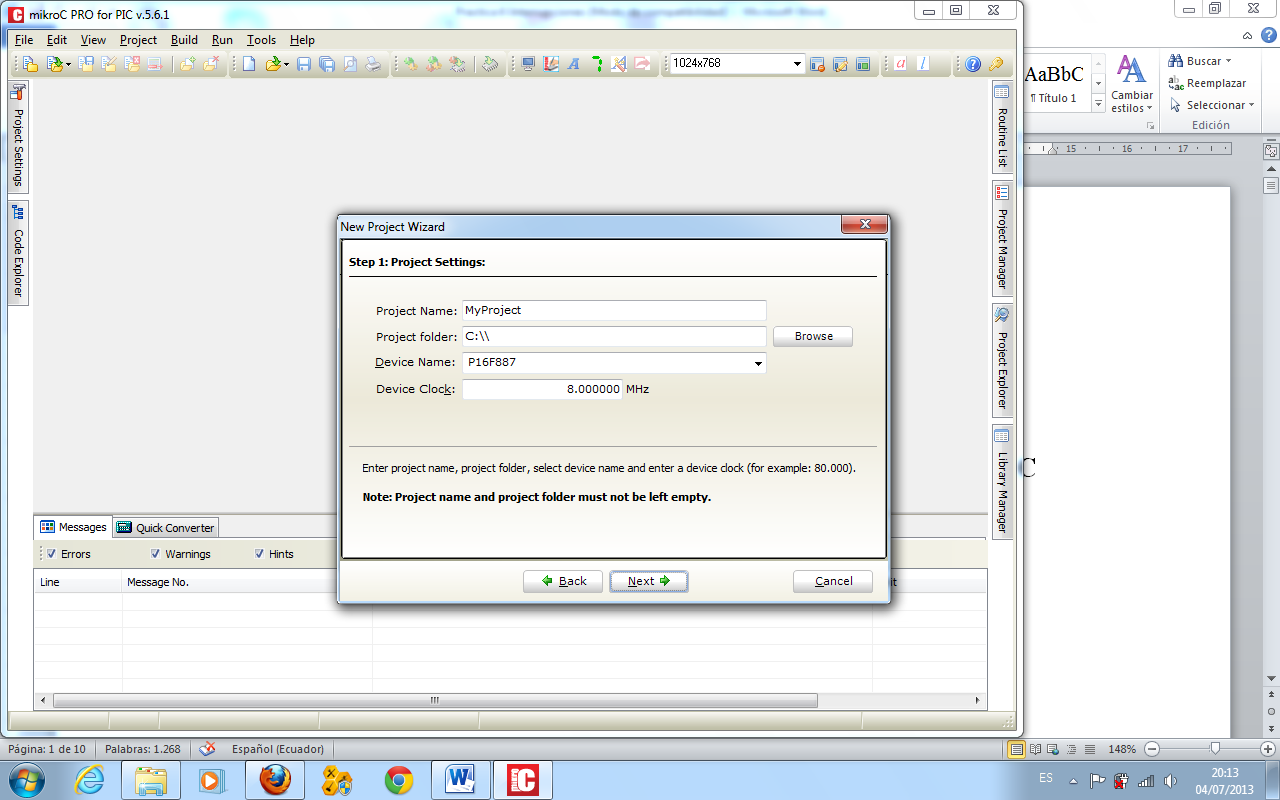
**CREACION DE UN PROYECTO NUEVO EN LENGUAJE C**

En esta práctica se hará uso del compilador mikroC de Mikroelectronica (www.mikroe.com). Para lo cual se deberán seguir las siguientes instrucciones:

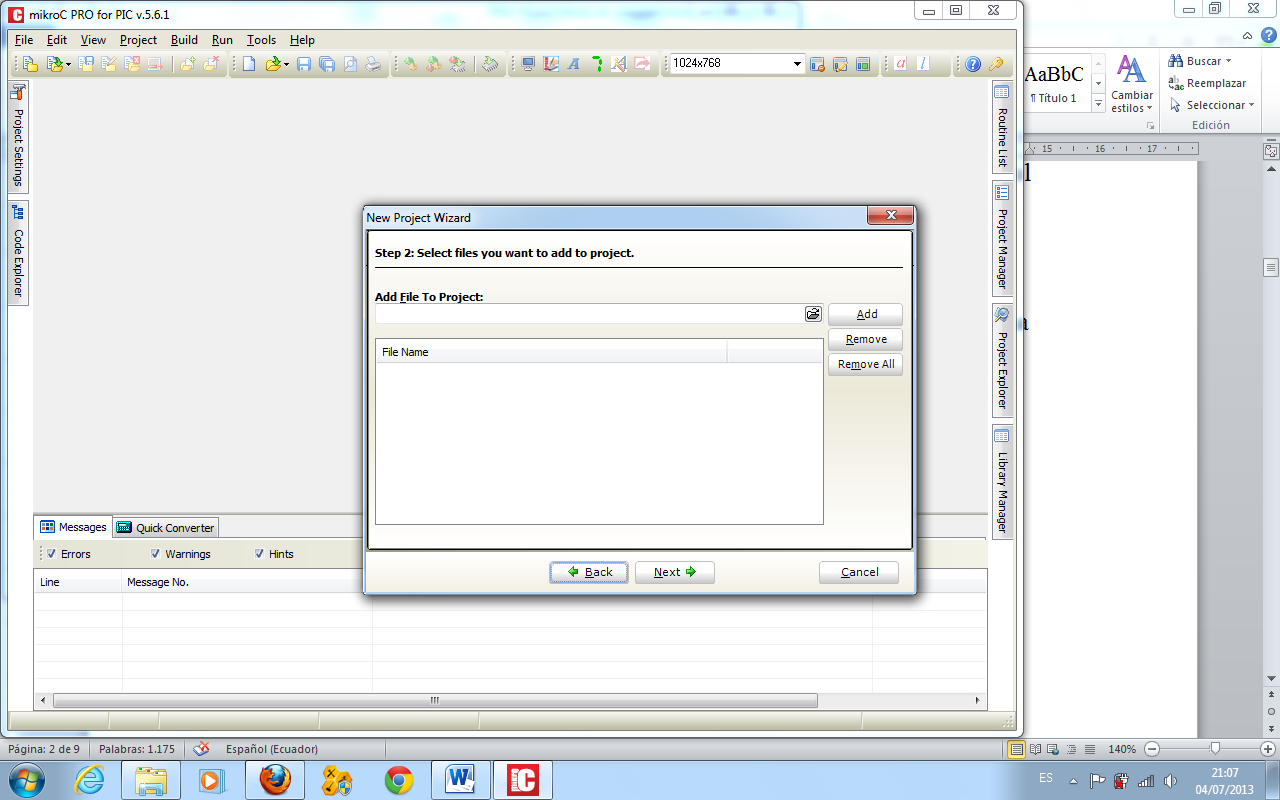
1. Abra el programa mikroC PRO for PIC
2. Cierre los proyectos abiertos
3. Abra un nuevo proyecto (Project🡪New Project) y aparece el Project Wizard que se muestra a continuación:



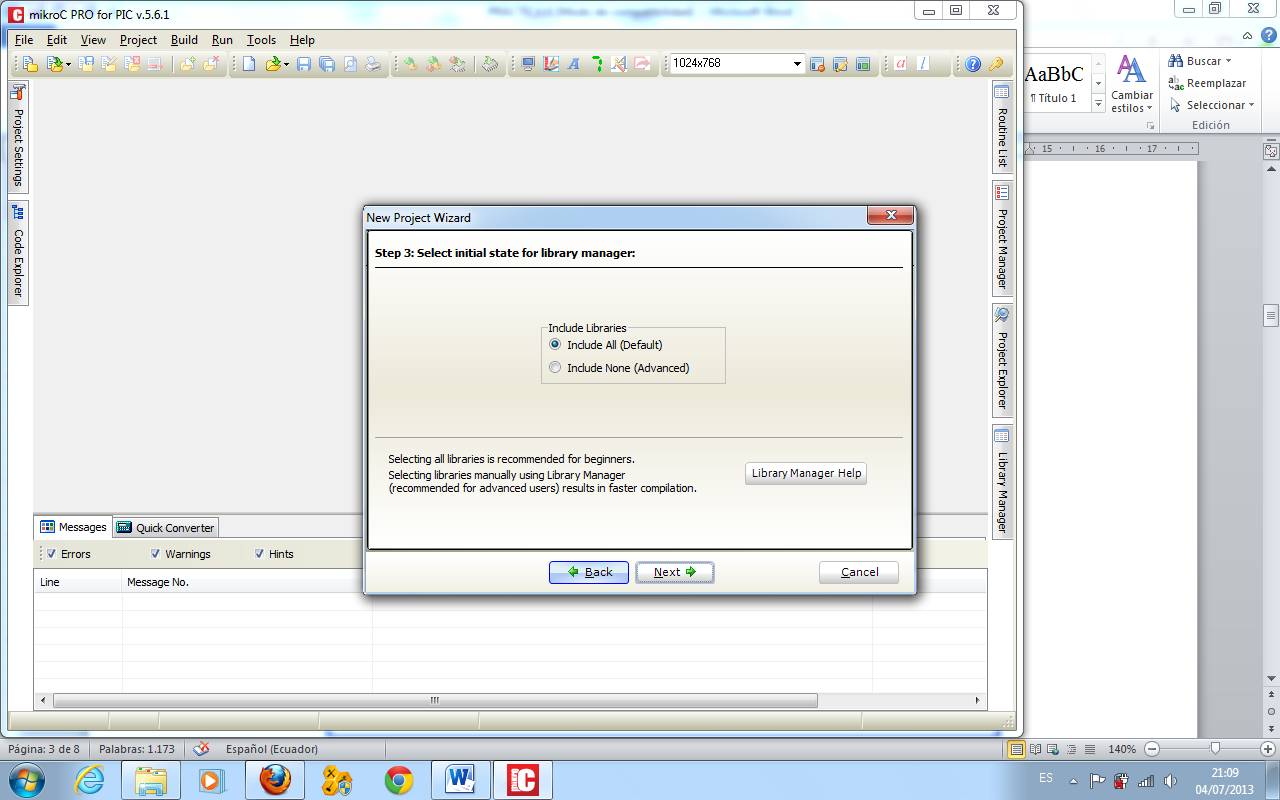
1. Se presiona **Next** y aparece una nueva pantalla donde se debe seleccionar el nombre para el proyecto, la carpeta o ubicación donde se guardará el proyecto, el tipo de microcontrolador (En este caso el 16F887) y por último el reloj de operación.



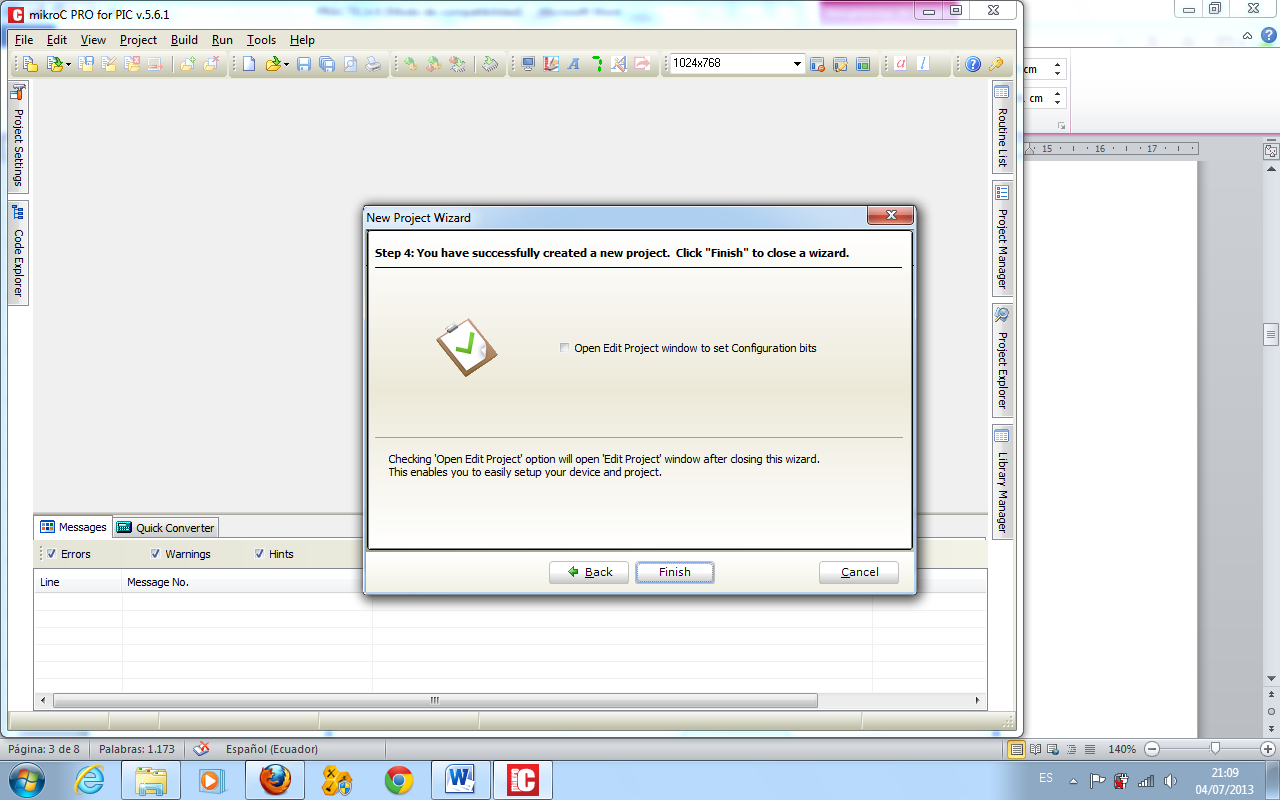
1. Con **Next** aparece una nueva pantalla:



1. Se presiona **Next** y aparece:



1. Apriete **Next** nuevamente y aparece:



1. Presione **Finish** para finalizar la creación del nuevo proyecto.

**EJERCICIOS SOBRE INTERRUPCIONES Y TEMPORIZADORES EN LENGUAJE C**

Utilizando el método para la creación de proyectos indicado proceda a la creación de los proyectos indicados a continuación.

**Proyectos:**

/\*

\* Nombre del Proyecto:

P6a\_t0.c

\* Nombre del Autor:

(c) Mikroelektronika, 2009.

\* Description:

(Explicación del ejercicio)

\* Test configuration:

MCU: PIC16F887

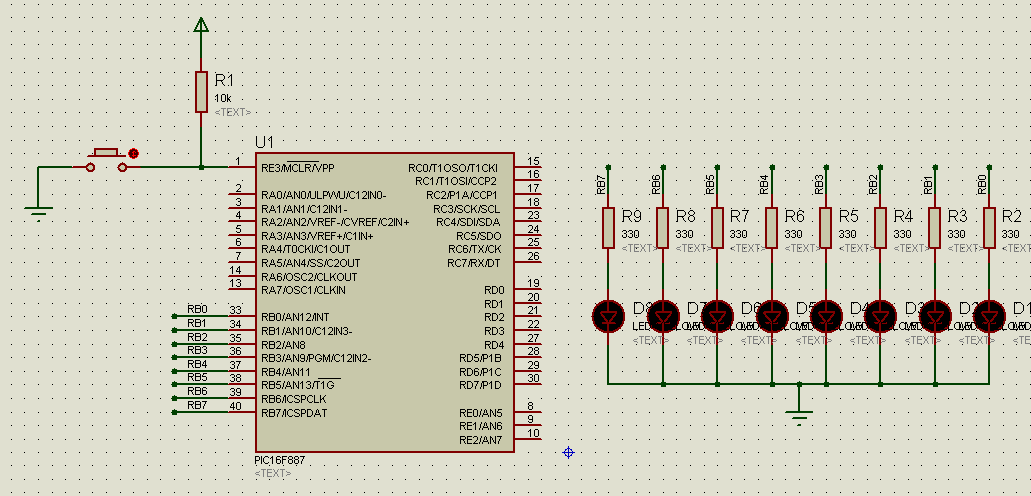
Oscillator: HS, 08.0000 MHz

SW: mikroC PRO for PIC

\* NOTES:

Cada vez que el TMR0 se desborda ocurre una interrupción

\*/



/\*Header\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

unsigned cnt; // Define variable cnt

void interrupt() {

cnt++; // Interrupt causes cnt to be incremented by 1

TMR0 = 96; // Timer TMR0 is returned its initial value

INTCON = 0x20; // Bit T0IE is set, bit T0IF is cleared

}

void main() {

OPTION\_REG = 0x84; // Prescaler is assigned to timer TMR0

ANSEL = 0; // All I/O pins are configured as digital

ANSELH = 0;

TRISB = 0; // All port B pins are configured as outputs

PORTB = 0x0; // Reset port B

TMR0 = 96; // Timer T0 counts from 96 to 255

INTCON = 0xA0; // Enable interrupt TMR0

cnt = 0; // Variable cnt is assigned a 0

do { // Endless loop

if (cnt == 400) { // Increment port B after 400 interrupts

PORTB = PORTB++; // Increment number on port B by 1

cnt = 0; // Reset variable cnt

}

} while(1);

}

**RESPONDER:**

1. Verifique el tiempo que demora en ocurrir una interrupción
2. ¿Cuánto demoran las 400 interrupciones?

PARTE 2

/\*

**\* Nombre del Proyecto:**

**P6b\_t1.c**

**\* Nombre del Autor:**

**(c) Mikroelektronika, 2009.**

**\* Description:**

**(Explicación del ejercicio)**

**\* Test configuration:**

**MCU: PIC16F887**

**Oscillator: HS, 08.0000 MHz**

**SW: mikroC PRO for PIC**

**\* NOTES:**

**Cada vez que el TMR1 (TMR1H, TMR1L) se desborda ocurre una interrupción**

\*/

***/\*Header\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/***

**unsigned short cnt; *// Define variable cnt***

**void interrupt() {**

**cnt++ ; *// Interrupt causes cnt to be incremented by 1***

**PIR1.TMR1IF = 0; *// Reset bit TMR1IF***

**TMR1H = 0x80; *// TMR1H and TMR1L timer registers are returned***

**TMR1L = 0x00; *// their initial values***

**}**

**void main() {**

**ANSEL = 0; *// All I/O pins are configured as digital***

**ANSELH = 0;**

**PORTB = 0xF0; *// Initial value of port B bits***

**TRISB = 0; *// Port B pins are configured as outputs***

**T1CON = 1; *// Set timer TMR1***

**PIR1.TMR1IF = 0; *// Reset bit TMR1IF***

**TMR1H = 0x80; *// Set initial value for timer TMR1***

**TMR1L = 0x00;**

**PIE1.TMR1IE = 1; *// Enable interrupt on overflow***

**cnt = 0; *// Reset variable cnt***

**INTCON = 0xC0; *// Enable interrupt (bits GIE and PEIE)***

**do { *// Endless loop***

**if (cnt == 76) { *// Change port B state after 76 interrupts***

**PORTB = ~PORTB; *// Number in port B is inverted***

**cnt = 0; *// Reset variable cnt***

**}**

**} while (1);**

**}**

**RESPONDER:**

1. Cada qué tiempo se interrumpe con el Timer 1
2. ¿Cuánto demoran las 76 interrupciones?

PARTE 3

/\*

**\* Nombre del Proyecto:**

**P6c\_t2.c**

**\* Nombre del Autor:**

**(c) Mikroelektronika, 2009.**

**\* Description:**

**(Explicación del ejercicio)**

**\* Test configuration:**

**MCU: PIC16F887**

**Oscillator: HS, 08.0000 MHz**

**SW: mikroC PRO for PIC**

**\* NOTES:**

**Cada vez que el TMR2 se desborda ocurre una interrupción**

**\*/**

***/\*Header\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/***

**unsigned short cnt; *// Define variable cnt***

**void Replace() {**

**PORTB = ~PORTB; *// Define new function ‘Replace’***

**} *// Function inverts port state***

**void interrupt() {**

**if (PIR1.TMR2IF) { *// If bit TMR2IF = 1,***

**cnt++ ; *// Increment variable cnt by 1***

**PIR1.TMR2IF = 0;*// Reset bit and***

**TMR2 = 0; *// reset register TMR2***

**}**

**}**

***// main***

**void main() {**

**cnt = 0; *// Reset variable cnt***

**ANSEL = 0; *// All I/O pins are configured as digital***

**ANSELH = 0;**

**PORTB = 0b10101010; *// Logic state on port B pins***

**TRISB = 0; *// All port B pins are configured as outputs***

**T2CON = 0xFF; *// Set timer T2***

**TMR2 = 0; *// Initial value of timer register TMR2***

**PIE1.TMR2IE = 1; *// Enable interrupt***

**INTCON = 0xC0; *// Set bits GIE and PEIE***

**while (1) { *// Endless loop***

**if (cnt > 30) { *// Change PORTB after more than 30 interrupts***

**Replace(); *// Function Replace inverts the port B state***

**cnt = 0; *// Reset variable cnt***

**}**

**}**

**}**

**RESPONDER:**

1. Cada qué tiempo se interrumpe con el Timer 2
2. ¿Cuánto demoran 31 interrupciones?

PARTE 4

/\*

**\* Nombre del Proyecto:**

**P6d\_wdt.c**

**\* Nombre del Autor:**

**(c) Mikroelektronika, 2009.**

**\* Description:**

**(Explicación del ejercicio)**

**\* Test configuration:**

**MCU: PIC16F887**

**Oscillator: HS, 08.0000 MHz**

**SW: mikroC PRO for PIC**

**\* NOTES:**

**Para que este programa funcione correctamente es necesario habilitar el uso del perro guardián con Project🡪Edit Project**

**\*/**

***/\*Header\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/***

**void main() {**

**OPTION\_REG = 0x08; *// Prescaler is assigned to timer WDT (1:64)***

**WDTCON = 1;**

**PORTB = 0x0F; *// Initial value of the PORTB register***

**TRISB = 0; *// All port B pins are configured as outputs***

**Delay\_ms(300); *// 30mS delay***

**PORTB = 0xF0; *// Porta B value different from initial***

**while (1)**

**{**

**asm CLRWDT; *// Assembly command to reset WDT timer***

***}***

**}**

**RESPONDER:**

1. Quite la instrucción: **asm CLRWDT** y explique lo que ocurre.