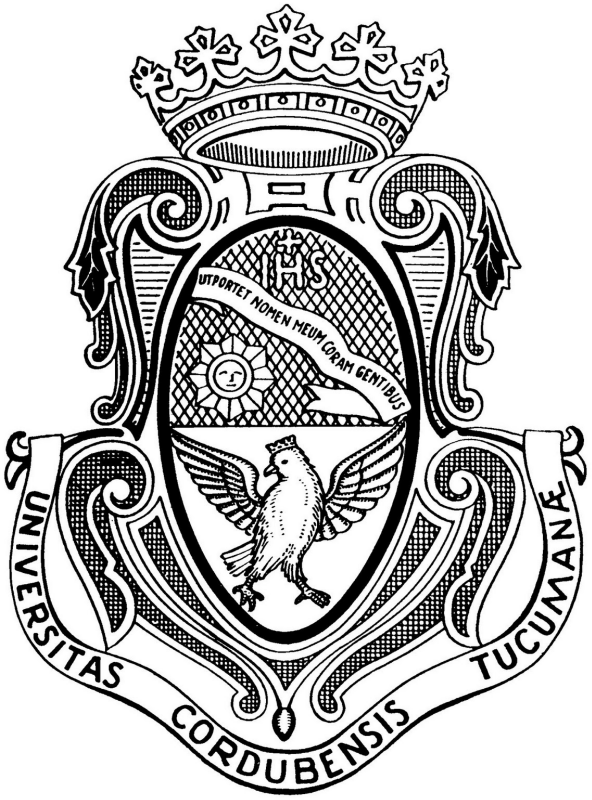
Universidad Nacional de Córdoba
Facultad de Ciencias Exatas, Físicas y Naturales



Trabajo Práctico Final

Electrónica Digital II

Docente: Ing. Martín Del Barco

Losano Quintana, Juan Cruz
Piñero, Tomás Santiago
Ingeniería en Computación

Año 2019

Índice

Índice **1**

1. Introducción **2**
Enunciado . . . 2

2. Desarrollo **2**
Cálculos realizados . . . 2
Resistencias para los segmentos . . . 2
Resistencias para la multiplexación . . . 3

3. Implementación **3**
Diagramas de flujo . . . 4
Programa principal . . . 4
Interrupciones . . . 6
Interrupción por RB0 . . . 6
Interrupción por *Timer0* . . . 6
Interrupción por *Timer1* . . . 6
Interrupción por *ADC* . . . 7
Delay . . . 7

4. Esquema del circuito **9**

1. Introducción

Este informe trata sobre el trabajo práctico final de la materia Electrónica Digital II. El tema es a elección de los estudiantes, por lo que se eligió realizar un sensor de temperatura para propósitos generales.

Enunciado

El sistema a realizar consiste en sensar la temperatura ambiente. La medición comienza cuando se presiona un botón. Una vez presionado se muestra el valor sensado en 4 (cuatro) *displays* de siete segmentos con el formato "XX°C" y se realiza un *log* en la computadora cada cinco segundos.

Para su realización se utilizaron los siguientes materiales:

- Microcontrolador PIC16F887,
- Cristal de 4 MHz,
- *Bridge USB to UART* CP2102,
- Sensor de temperatura LM35,
- Cuatro *displays* de siete segmentos cátodo común,
- Once transistores NPN XXXX,
- Resistencias de 1 kΩ, 330 Ω y 220 Ω,
- Capacitor de 22 pF,
- Un pulsador.

2. Desarrollo

Cálculos realizados

Resistencias para los segmentos

El color de los *displays* es rojo, por lo que cada segmento consume 1.7 [V] y 10 [mA], aproximadamente. Como la tensión de salida del PIC es $V_{OH} = 4{,}3\,[V]$, el cálculo queda:

$$algo = 0$$

$$R = \frac{algo}{algomas}$$

$$R = 330\,[\Omega] \tag{1}$$

Resistencias para la multiplexación

Al tratarse de *displays* de cátodo común, se necesitan transistores NPN para su multiplexación.

La multiplexación consiste en encender un *display* por vez, por lo que se necesita una frecuencia en la que el ojo sea capaz de ver *todos* los *displays* encendidos al mismo tiempo. Esta frecuencia es 50 [Hz].

Por lo tanto, la fórmula para el tiempo t de encendido de cada *display* es la siguiente:

$$t = \frac{20\,[ms]}{N} \tag{2}$$

Siendo N la cantidad de *displays* a utilizar.

Al tratase de cuatro *displays* la ecuación 2 queda:

$$t = \frac{20\,[ms]}{4}$$

$$t = 5\,[ms] \tag{3}$$

3. Implementación

El PIC tiene como frecuencia de reloj un cristal de 4 MHz, por lo que el ciclo de instrucción se realiza con una frecuencia de 1 MHz debido a que cada 4 ciclos del reloj se realiza una instrucción. Esto se debe a que para ejecutar la instrucción indicada, el PIC debe ejecutar cuatro acciones:

1. Buscar la instrucción en la memoria principal.
2. Decodificar la instrucción.
3. Ejecutar la instrucción.
4. Almacenar los resultados.

Esto es importante para el cálculo de la subrutina de retardo, ya que depende de la frecuencia de reloj que se utilice.

Diagramas de flujo

En esta sección se muestran los diagramas de flujo del programa principal y las subrutinas que utiliza.

Programa principal

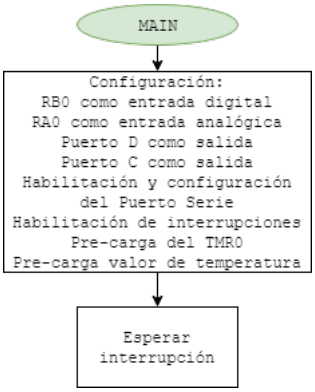


Figura 1: Diagrama de flujo del programa.

Primero se realiza la configuración de los puertos:

- Puerto A: A0 como entrada analógica.
- Puerto B: RB0 como entrada digital.
- Puerto C como salida digital.
- Puerto D como salida digital.

Una vez configurados los puertos se configuran los módulos a utilizar.

- *Timer0*: (Diagrama 3) se utiliza para la multiplexación de los *displays*, por lo que se lo configura en modo temporizador para contar los 5 [ms]. El tiempo máximo que puede contar el *timer0* con el *prescaler* en 256 y la frecuencia de máquina en 1 [MHz] es:

$$t = (256 - precarga) * 256 * 1\,[\mu s] \tag{4}$$
$$t_{max} = 256 * 256 * 1\,[\mu s]$$

$$t_{max} \approx 65\,[ms] \tag{5}$$

Por lo tanto, para 5 [ms] la ecuación 4 queda:

$$5\,[ms] = (256 - precarga) * 256 * 1\,[\mu s]$$

$$precarga = 256 - \frac{5\,[ms]}{256 * 1\,[\mu s]}$$

$$precarga = 236 \tag{6}$$

- *Timer1*: (Diagrama 4) este módulo también se utiliza en modo contador ya que se encarga de contar los cinco segundos. Al cumplirse este tiempo, se realiza una nueva conversión, actualizando el valor sensado, y lo envía al puerto serie.

  El tiempo máximo que puede contar este *timer* con el *prescaler* en 8 y la frecuencia de máquina en 1 [MHz] es:

$$t = (65536 - precarga) * 8 * 1\,[\mu s]$$
$$t_{max} = 65536 * 8 * 1\,[\mu s]$$

$$t_{max} \approx 524\,[ms] \tag{7}$$

Al no poder alcanzar los cinco segundos, se utiliza el *timer1* para contar 500 [ms] y una variable auxiliar que se encargue de verificar que el *timer1* se haya desbordado diez veces, consiguiendo así los cinco segundos necesarios. Para contar los Consecuentemente, la ecuación 7 queda, para 500 [ms]:

$$500\,[ms] = (65536 - precarga) * 8 * 1\,[\mu s]$$

$$precarga = 65536 - \frac{500\,[ms]}{8 * 1\,[\mu s]}$$

$$precarga = 3036 \tag{8}$$

Esta precarga debe realizarse en dos registros distintos debido a que el *timer1* es de 16 bits. La parte alta del registro (TMR1H) se precarga con un valor de 0x0B, mientras que la parte baja del registro (TMR1L) se precarga con un valor de 0xDC, formando así un total de 3060 en decimal y 0x0BDC en hexadecimal.

Por último se precarga el valor que de temperatura “00°C” que mostrará el programa mostrará el valor hasta que se presione el botón conectado en RB0.

Interrupciones

Interrupción por RB0 Esta interrupcion hace esto

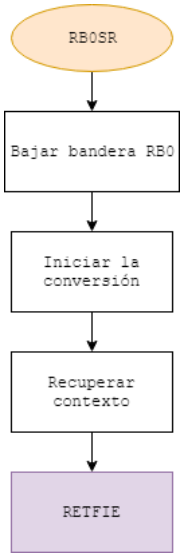


Figura 2: Diagrama de flujo de la interrupción por RB0.

Interrupción por *Timer0* Esta interrupcion hace esto

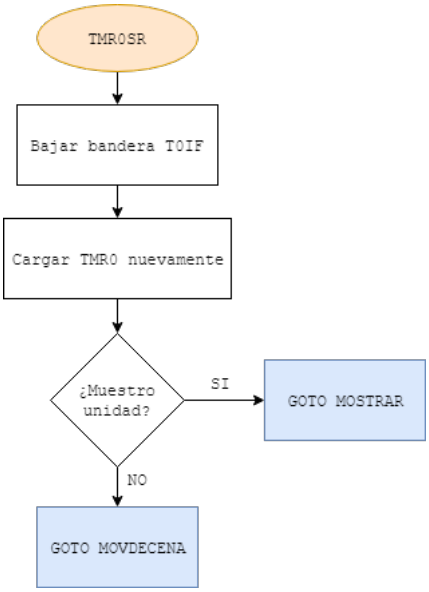


Figura 3: Diagrama de flujo de la interrupción por *Timer0*.

Interrupción por *Timer1* Esta interrupcion hace esto

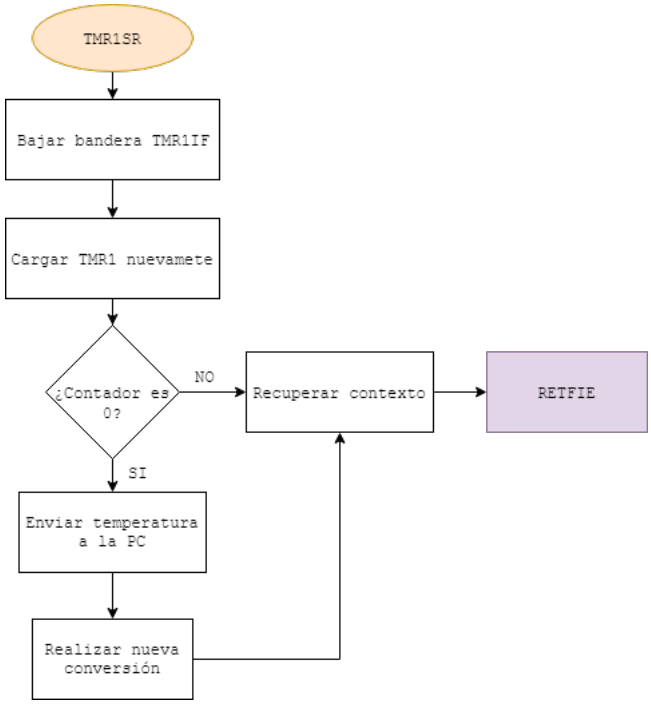


Figura 4: Diagrama de flujo de la interrupción por *Timer1*.

Interrupción por *ADC* Esta interrupcion hace esto

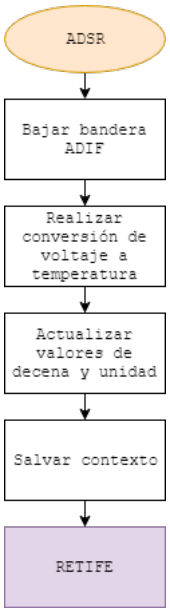


Figura 5: Diagrama de flujo de la interrupción por *ADC*.

Delay

Esta sub–rutina se encarga de darle al conversor los tiempos necesarios para la adquisición y conversión de los datos.

El tiempo T que demora este *delay* es el siguiente:

$$T = 4\,\mu s + 3\,\mu s * (DLAY - 1)$$
$$T = 4\,\mu s + 3\,\mu s * (250 - 1)$$

$$T = 751\,\mu s \tag{9}$$

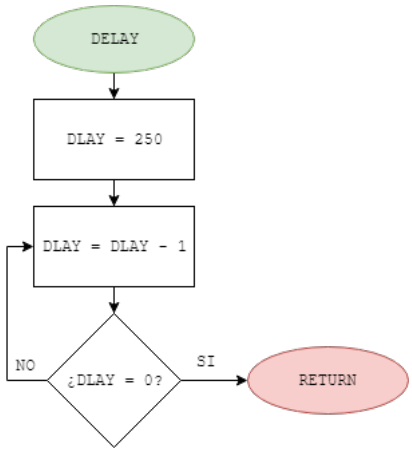


Figura 6: Diagrama de flujo del *delay*.

4. Esquema del circuito