**UNIVERSIDAD PEDAGÓGICA Y TECNOLÓGICA DE COLOMBIA**

**SEDE SECCIONAL SOGAMOSO**

**ESCUELA DE INGENIERÍA ELECTRÓNICA**

**MICROCONTROLADORES**

**COMPILACION, SIMULACION Y CREACION DE APLICACIONES BÁSICAS CON MICROCONTROLADORES**

Edgar Leonardo Mesa Camilo Andrés Puerto

e-mail: edgar.mesa@uptc.edu.co e-mail: camilo.puerto@uptc.edu.co

15 de marzo 2017

**RESUMEN:** *En esta práctica de laboratorio se realiza la utilización e implementación del microcontrolador PIC16F887 mediante ejercicios sencillos que permiten entender su funcionamiento*

**PALABRAS CLAVE**: microcontrolador, puertos, programador de PIC’s, rutina, set de instrucciones.

# 1. INTRODUCCIÓN

En la actualidad los microcontroladores son una herramienta muy utilizada para el desarrollo de la tecnología porque son un mecanismo que permite desarrollar cada una de las ideas que llegan a nuestra imaginación con el fin de cumplir una necesidad requerida para un proceso especifico.

Por lo tanto, en esta práctica se utiliza el set de instrucciones del PIC16F887 y el entorno de trabajo en el software Mplab y Proteus mediante una serie de pasos definidos en una guía de trabajo; cada ejercicio demuestra la utilización eficiente del microcontrolador para cualquier tipo de aplicación de la forma básica.

# 2. OBJETIVOS

* + - * Usar la aplicación de Microchip MPLAB en la creación, compilación, simulación y programación de micro controladores.
      * Usar la aplicación PROTEUS como herramienta de depuración y prueba de código realizado en la aplicación MPLAB.
      * Manejar el set de instrucciones de los microcontroladores PIC mediante un problema propuesto.

**3. MATERIALES Y EQUIPOS**

* Protoboard
* Resistencias
* Diodos LEDS
* Software MPLABX
* Software de simulación PROTEUS
* Fuente de alimentación 5V

**4. PROCEDIMIENTO**

**CONFIGURACION DE LA PALABRA DE CONTROL:**

Los bits de configuración pueden controlarse mediante el código obtenido en la herramienta del programa, o directamente escribiendo dicha disposición de bits en formato hexadecimal. Este formato obedece al conjunto definido de bits del registro CONFIG1 Y CONFIG2, y que puede observarse en el anexo 2.

Para el caso del registro CONFIG1, este consta de 16 bit a manipular, de los cuales concretamente modificaremos del cero al doce, dejando los restantes por defecto en 1. Cada uno de los bits comanda partes especificas del funcionamiento de microcontrolador así:

El bit 12 corresponde al voltaje bajo de programación LVP, que habilitado toma al pin RB3 como entrada durante le proceso de programación. Se deja deshabilitado. El siguiente es FCMEN, que activa o desactiva el reloj de respaldo en fallos, que se mantiene apagado.

El bit 10 (IESO) comanda el sintecho interno o externo del PIC, que se deja habilitado

El bit 9 (BOREN) comanda el reset de Brown out. Por datasheet se habilita su operación apagándose en un reset, lo que equivale a BOREN=10

CPD comanda la protección de la memoria de datos del código, que se deja habilitada con un 0.

CP es la protección del código que se habilita también con 0

MCLRE= comanda la forma como se hace el reset manual del equipo, que se ubica en 1, dejando a RE3 como entrada del reset.

PWRT comanda un contador de encendido del microcontrolador, que se deshabilita con un 1

WDTE hablita el perro guardián, con lo que para este caso se deja deshabilitado con un 0.

FOSCINTRCNOCLKOUT comanda el tipo y velocidad del oscilador a usar. Como el oscilador es interno, se configura con 100

Teniendo ya la cadena de binarios que debemos utilizar, se convierten en hexadecimal obteniendo:

La cadena obtenida es: 1110001000110100 que convertida en hexa es: E234

Para el caso de CONFIG\_2, los bits del 15 al 11 se leen como uno, WRT comanda la protección de escritura del pic, que corresponderá a 00ff y configurada como 10. El rango del reset se necesita de BOR21v, que equivale a 0. Por lo tanto, la cadena en binario es:

1111111011111111 y el valor en hexa es: FEFF

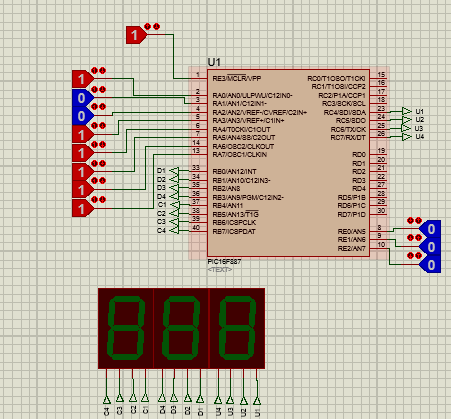
**CONTADOR/CONVERSOR**

En este ejercicio se debe realizar un contador de 0 a 255 con una conversión a BCD y la ubicación de las unidades decenas y centenas en diferentes nibble de puerto, al igual que un pin de precarga en el cual el valor que ingresa al PUERTO A se muestre por el PUERTO B Y C en su equivalencia en BCD.

Para realizar esto se definen tres registros (UNIDADES, DECENAS, CENTENAS), el valor de entrada es guardado en el registro UNIDADES luego a este se le restan 100, si el valor de esta resta es positivo se va incrementando el registro centenas pero si es negativo al registro UNIDADES se le vuelven a sumar 100 para volverlo positivo para restarlo después con un 10, si el valor es positivo se va incrementando el registro decenas pero si es negativo al registro UNIDADES se le vuelven a sumar 10 para volverlo positivo y obtener el valor de las unidades ingresadas o determinadas por el contador.

Para mostrar por el nibble alto del PUERTO B las centenas y por el nibble bajo las decenas, al registro centenas se le hace un intercambio de nibbles y se suma con lo que está en las decenas para después mostrar en el PUERTOB; se realiza un intercambio de nibbles de las unidades y se muestra en la parte alta del PUERTO C.

El diagrama de flujo de este ejercicio se muestra en el anexo



**Fig. 1.** Circuito conversor binario a BCD.

**SECUENCIAS DE LEDS**

Cuando el programa se ejecuta, va aumentando un contador Interno. Se aprovecha este hecho para comparar el conteo con un valor fijo, y testear esa comparación. Cuando el conteo acaba, la comparación obliga al programa a saltar a una rutina que va leyendo las posiciones de una tabla en el código, en donde están las secuencias de leds a implementar. Finalmente, cada vez que se termina la secuencia de leds, esta se repite para poder lograr el minuto de duración requisito del laboratorio.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Tabla 01:** Patrón de secuencia en binario y Hexadecimal | | | | | | | | | | |
|  | | | | | | | | | | |
|  |  | Salida Led | | | | | | | |  |
|  |  | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | **Hexa** |
| Serie | S0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | **0X81** |
| S1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | **0XC3** |
| S2 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | **0XE7** |
| S3 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | **0XFF** |
| S4 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | **0X7E** |
| S5 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | **0X3C** |
| S6 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | **0X18** |
| S7 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | **0X3C** |
| S8 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | **0X7E** |
| S9 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | **0XFF** |
| S10 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | **0XE7** |
| S11 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | **0XC3** |

**5. CONCLUSIONES**

* La utilización de diagramas de flujo ayuda a llevar un orden lógico de lo que se está haciendo en cualquier aplicación, evitando así cometer errores que comúnmente suelen ocurrir cuando se está implementando en un lenguaje de programación.
* Aunque la configuración de microcontrolador puede realizarse estableciendo los valores de los bits de los respectivos registros, CONFIG\_1 y CONFIG\_2 forzando su valor introduciendo un literal como el caso de un registro de propósito general MPLAB provee herramientas que simplifican el proceso, mediante l elaboración de la palabra de control por código, caso en el cual se es más consiente de la configuración de cada uno de los componentes utilizados por el PIC para su funcionamiento
* El software PROTEUS es una buena herramienta de simulación paso a paso que sirve para detectar errores en el funcionamiento cuando se está programando un PIC.
* Para el correcto funcionamiento al elaborar cualquier aplicación usando PIC’s es importante entender y conocer el microcontrolador, sus características, su estructura interna, así como las especificaciones de los elementos externos conectados al PIC.
* Es posible el uso de otros microcontroladores si se conoce la programación de una familia similar tal es el caso del pic 16f877

**6. REFERENCIAS**

* [1] http://www.mikroe.com/chapters/view/86/libro-de-a-progamacion-de-losmicrocontroladores-