**ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL**

**FIEC**

**Laboratorio de Microcontroladores**

**Proyecto del Primer Parcial**

**Comparador de LED.**

**Mensaje en 2 Display 7-Segmentos**

**Alumno:**

**Jaime Tinoco Campuzano.**

**Paralelo: # 4**

**Grupo: # 1**

**Fecha de presentación:**

**11/07/2011**

**2011 – I TÉRMINO**

**Parte1**

**1.- Comparador de LED’s**

Colocar *LEDS* en el puerto C. Habrán 2 botones en el puerto B, uno será para incrementar un contador, y el otro lo decrementará. El máximo valor del contador es 10 y el mínimo 0. Se creará una variable dentro del programa, que comparará su valor con el del contador, y dependiendo de ello ocurrirá lo siguiente:

Variable = Contador : Parpadean *LEDs* en el puerto C a 1Hz.

Variable > Contador : Los *LEDs* rotan de izquierda a derecha a una frecuencia de 1 Hz.

Variable < Contador : Los *LEDs* rotan de derecha a izquierda a una frecuencia de 1 Hz.

**2.- Diagrama de Bloques**

PIC

16f887

8LED

Pulsadores

PORTB

**3.- Diagrama de Flujo funcional del Programa principal**



**4.- Diagrama de Flujo funcional de las Subrutinas**

**Subrutina de parpadeo (blink)**



**Subrutina de rotación de led a la derecha (der)**



**Subrutina de rotación de led a la izquierda (izq)**



**Subrutina de retardo 1seg. (unSegundo)**



**5.- Descripción del algoritmo o estrategia utilizada.**

1. Se comienza el programa configurando todo el puerto C como salidas digitales del programa y dos bits del puerto B como entradas.
2. Luego inicializamos las variables a utilizar o sea, el PORTB y counter1, counter2, counter3, var, cont, temp1, temp2, temp3.
3. La variable cont se compara con var a cada momento y dependiendo si es mayor, menor o igual a var realiza desplazamiento a la izquierda, delerecha o parpadeo de las salidas del PORTC.
4. La comparación de igualdad se la realiza mediante la operación xor entre cont y var que ocasiona que Z de STATUS se ponga en alto; luego para decidir si cont es mayor o menor que var se realiza una operación de resta en estas dos y probamos el bit mas significativo dado a que el PIC usa resta de complemento a 2
5. Si es mayor o menor setea un bit en el PORTC de acuerdo a la tabla1 o 2 le da un retardo de un segundo y continua preguntando por las entradas en el portB en caso de que estas hayan alterado el valor de cont
6. Si el valor de cont es igual a var avanza a una subrutina en la cual hace el seteo de todo el PORTC, le da retardo de un segundo; lo limpia de nuevo a PORTC le da retardo otra vez, y sigue consultando por las entradas de PORTB que manipula el valor de la variable cont.

**6.- Listado del programa fuente en lenguaje ensamblador con comentarios en las líneas de código que considere fundamentales**

**;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**; Comparador de LED's**

**;Colocar LEDS en el puerto C. Habrán 2 botones en el puerto B,**

**;uno será para incrementar un contador, y el otro lo decrementará.**

**;El máximo valor del contador es 10 y el mínimo 0.**

**;Se creará una variable dentro del programa,**

**;que comparará su valor con el del contador,**

**;y dependiendo de ello ocurrirá lo siguiente:**

**;**

**;Variable = Contador : Parpadean LEDs en el puerto C a 1Hz.**

**;Variable > Contador : Los LEDs rotan de izquierda a derecha**

**;a una frecuencia de 1 Hz.**

**;Variable < Contador : Los LEDs rotan de derecha a izquierda**

**;a una frecuencia de 1 Hz.**

**;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*HEADER\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**LIST p=16F887**

 **INCLUDE P16F887.INC**

 **\_\_CONFIG \_CONFIG1, \_CP\_OFF&\_WDT\_OFF&\_XT\_OSC**

 **errorlevel -302**

 **errorlevel -205**

 **errorlevel -203**

**;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*VARIABLES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**cblock 0x20**

**cont**

**var**

**counter1**

**counter2**

**counter3**

**temp**

**temp1**

**temp2**

**endc**

**;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*INICIO\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**org 0x00**

**movlw .5**

**movwf var**

**movlw .0**

**movwf cont**

**banksel ANSEL**

**clrf ANSEL**

**clrf ANSELH**

**banksel TRISB**

**clrf TRISC ; configura el puertoC como salidas**

**clrf TRISA**

**movlw b'00000011' ;dos pines del puertoB como entradas**

**movwf TRISB**

**banksel PORTB**

 **movlw 0x00**

 **movwf temp1**

 **movwf temp2**

 **goto inicio**

**inicio movf cont,0**

 **movwf PORTA ;visualizar en A**

 **bcf STATUS,Z**

 **movf var,0**

 **xorwf cont,0**

 **btfss STATUS,Z**

 **goto otro**

 **goto blink**

**otro**

 **movf var,0**

 **subwf cont,0**

 **movwf temp ;variable usada para verificar si cont es**

 **bcf STATUS,C; mayor o menor que var**

 **btfss temp,7**

 **goto izq**

 **goto der**

**izq clrf PORTC**

 **movlw 0x01**

 **addwf temp1,0**

 **movwf temp1**

 **call TABLA**

 **movwf PORTC**

 **btfsc PORTC,7**

 **clrf temp1**

 **call unSegundo**

 **goto rutina1**

**der clrf PORTC**

 **movlw 0x01**

 **addwf temp2,0**

 **movwf temp2**

 **call TABLA1; tomo el valor correspondiente en la tabla1**

 **movwf PORTC;dependiendo de el valor de temp2**

 **btfsc PORTC,0**

 **clrf temp2**

 **call unSegundo**

 **goto rutina1**

**rutina1 movf cont,0**

 **xorlw 0x0B**

 **btfss STATUS,Z**

 **goto rutina11**

 **movlw 0x00**

 **movwf cont**

 **goto rutina11**

**rutina11 btfsc PORTB,0**

 **goto rutina2**

 **incf cont,1**

**hold btfss PORTB,0**

 **goto hold**

 **goto rutina2**

**rutina2 movf cont,0**

 **bcf STATUS,Z**

 **xorlw 0xFF**

 **btfss STATUS,Z**

 **goto rutina22**

 **movlw 0x0A**

 **movwf cont**

 **goto inicio**

**rutina22 btfsc PORTB,1**

 **goto inicio**

 **decf cont,1**

**hold1 btfss PORTB,1**

 **goto hold1**

 **goto inicio**

**blink call unSegundo**

 **movlw 0xFF**

 **movwf PORTC**

 **call unSegundo**

 **clrf PORTC**

 **goto rutina1**

**unSegundo**

 **movlw .250**

 **movwf counter1**

 **movwf counter2**

 **movlw .5**

 **movwf counter3**

**loop1**

 **decfsz counter1**

 **goto loop1**

 **decfsz counter2**

 **goto loop1**

 **decfsz counter3**

 **goto loop1**

 **return**

**TABLA addwf PCL,F**

 **retlw 0x00 ;0**

 **retlw 0x01 ;1**

 **retlw 0x02 ;2**

 **retlw 0X04 ;3**

 **retlw 0X08 ;4**

 **retlw 0X10 ;5**

 **retlw 0X20 ;6**

 **retlw 0X40 ;7**

 **retlw 0X80 ;8**

**TABLA1 addwf PCL,F**

 **retlw 0xFF ;0**

 **retlw 0x80 ;0**

 **retlw 0x40 ;1**

 **retlw 0x20 ;2**

 **retlw 0X10 ;3**

 **retlw 0X08 ;4**

 **retlw 0X04 ;5**

 **retlw 0X02 ;6**

 **retlw 0X01 ;7**

 **retlw 0X00 ;8**

**end**

**7.- Copia impresa del circuito armado en PROTEUS para la simulación en el momento de su ejecución**

****

****

**Parte2(Propuesto por el estudiante)**

**1.- Mensaje en dos Display’s de 7 segmentos**

En esta parte mediante el uso de subrutinas multiplexación haré la presentación del mensaje “HOLA CLASE” 2 display’s de 7 segmentos conectados al puerto C de manera consecutiva mediante RB0 se habilita mostrar el mensaje y con RB1 damos inicio a la presentación.

**2.- Diagrama de Bloques**

PIC

16f887

7

SEG

Pulsadores

PORTB

7

SEG

**3.- Diagrama de Flujo funcional del Programa principal**



**4.- Diagrama de Flujo funcional de las Subrutinas**

**Subrutina unSegundo (ver primera parte)**

**Subrutina mensaje**



**5.- Descripción del algoritmo o estrategia utilizada.**

1. Se comienza el programa configurando todo el puerto D como salidas digitales del programa, dos bits del puerto B como entradas y dos bits del puerto A como las encargadas de controlar la multiplexacion entre los display’s.
2. Pasamos a verificar la entrada RB0 que habilitara el mensaje hasta que le sea dada al programa.
3. Seguido pasamos a la subrutina del mensaje la cual empieza a la espera que le den inicio mediante RB1.
4. Pasa a cargarse todas las letras, una por una de manera sucesiva en los display’s de 7segmentos hasta que concluye el mensaje y se retorna a la espera nuevamente de RB0 sea habilitada.

**6.- Listado del programa fuente en lenguaje ensamblador con comentarios en las líneas de código que considere fundamentales**

**;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**; mensajes deslizante "hola clase"en displays de 7 segmentos**

**;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

 **list p=16f887**

 **include "p16f887.inc"**

 **errorlevel-302**

 **counter1 equ 0x24**

 **counter2 equ 0x25**

 **counter3 equ 0x26**

**;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* inicio del programa \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

 **org 0**

 **goto inicio**

 **inicio**

 **;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* seteo de bancos \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

 **bsf STATUS,5 ;banco1**

 **clrf PORTD**

 **clrf PORTA**

 **banksel ANSEL ;selecciono banco3**

 **clrf ANSEL**

 **clrf ANSELH**

 **banksel PORTB**

 **clrf PORTB ;inicializo el portb**

 **banksel TRISB**

 **movlw b'11111111' ;portb como entrada**

 **movwf TRISB ;**

 **clrf TRISA**

 **bcf STATUS,5**

 **clrf PORTD**

 **clrf PORTA**

 **clrf PORTB**

**;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* verificamos el pulsador \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**p2**

 **btfsc PORTB,0**

 **goto mensaje ;me mantengo preguntando por bit 0 de portb hasta q lo tenga**

 **goto p2**

**;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**; visualizacion del mensaje "HOLA CLASE"**

**;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*l\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**mensaje**

 **call teclaip ;espera a que se presione y suelte la tecla**

 **movlw 0x76 ;hexadecimal de la H**

 **movwf PORTD ;la muestra en display**

 **bsf PORTA,1**

 **bcf PORTA,0**

 **call unSegundo**

 **movlw 0x76**

 **movwf PORTD**

 **bsf PORTA,0**

 **bcf PORTA,1**

 **call unSegundo**

 **movlw 0x3F ;hexadecimal de la O**

 **movwf PORTD ;la muestra en display**

 **bsf PORTA,1**

 **bcf PORTA,0**

 **call unSegundo**

 **movlw 0x3F**

 **movwf PORTD**

 **bsf PORTA,0**

 **bcf PORTA,1**

 **call unSegundo**

 **movlw 0x38 ;hexadecimal de la L**

 **movwf PORTD**

 **bsf PORTA,1**

 **bcf PORTA,0**

 **call unSegundo**

 **movlw 0x38**

 **movwf PORTD**

 **bsf PORTA,0**

 **bcf PORTA,1**

 **call unSegundo**

 **movlw 0x77 ;hexadecimal de la A**

 **movwf PORTD**

 **bsf PORTA,1**

 **bcf PORTA,0**

 **call unSegundo**

 **movlw 0x77**

 **movwf PORTD**

 **bsf PORTA,0**

 **bcf PORTA,1**

 **call unSegundo**

 **movlw 0x00 ;hexadecimal para un espacio**

 **movwf PORTD**

 **bsf PORTA,1**

 **bcf PORTA,0**

 **call unSegundo**

 **movlw 0x00**

 **movwf PORTD**

 **bsf PORTA,0**

 **bcf PORTA,1**

 **call unSegundo**

 **movlw 0x39 ;hexadecimal de C**

 **movwf PORTD**

 **bsf PORTA,1**

 **bcf PORTA,0**

 **call unSegundo**

 **movlw 0x39**

 **movwf PORTD**

 **bsf PORTA,0**

 **bcf PORTA,1**

 **call unSegundo**

 **movlw 0x38 ;hexadecimal de L**

 **movwf PORTD**

 **bsf PORTA,1**

 **bcf PORTA,0**

 **call unSegundo**

 **movlw 0x38**

 **movwf PORTD**

 **bsf PORTA,0**

 **bcf PORTA,1**

 **call unSegundo**

 **movlw 0x77 ;hexadecimal de la A**

 **movwf PORTD**

 **bsf PORTA,1**

 **bcf PORTA,0**

 **call unSegundo**

 **movlw 0x77**

 **movwf PORTD**

 **bsf PORTA,0**

 **bcf PORTA,1**

 **call unSegundo**

 **movlw 0x6D ;hexadecimal de la S**

 **movwf PORTD**

 **bsf PORTA,1**

 **bcf PORTA,0**

 **call unSegundo**

 **movlw 0x6D**

 **movwf PORTD**

 **bsf PORTA,0**

 **bcf PORTA,1**

 **call unSegundo**

 **movlw 0x79 ;hexadecimal de la E**

 **movwf PORTD**

 **bsf PORTA,1**

 **bcf PORTA,0**

 **call unSegundo**

 **movlw 0x79**

 **movwf PORTD**

 **bsf PORTA,0**

 **bcf PORTA,1**

 **call unSegundo**

 **clrf PORTD**

 **goto p2**

**;\*\*\*\*\*\*\*\*\*\*\*\*\* presionar y soltar la habilitadora \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**teclaip btfsc PORTB,1 ;presionar pulsador**

 **goto teclaip**

**espera btfss PORTB,1 ;soltar pulsador**

 **goto espera**

 **return**

**;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**; Retardo de un segundo**

**;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**unSegundo**

 **movlw .250**

 **movwf counter1**

 **movwf counter2**

 **movlw .5**

 **movwf counter3**

**loop1**

 **decfsz counter1**

 **goto loop1**

 **decfsz counter2**

 **goto loop1**

 **decfsz counter3**

 **goto loop1**

 **return**

**end**

**7.- Copia impresa del circuito armado en PROTEUS para la simulación en el momento de su ejecución**

****

****

**8.- Conclusiones**

* El manejo de varias tareas mediante el uso de microcontroladores no se realiza de forma simultánea sino de hecho una por una, atendiendo alternadamente a todas, pero gracias a las altas velocidades de los osciladores se pueden resolver grandes cantidades de código en unos pocos milisegundos.
* El uso de tablas resulto de gran ayuda a la hora de resolver el problema de la rotación de los LED’s ya sea de izquierda a derecha o vice-versa, puesto que al principio considere usar las instrucciones RRF y RLF pero debido a la forma de mi algoritmo no me permitía su uso sin primero alterar la bandera de acarreo C de STATUS.
* Es posible la creación de programas multitarea mediante la un buen manejo de los diversos recursos del microcontrolador, tales como el uso de las tablas, subrutinas, manejo de los bit’s del registro de STATUS en el primer y segundo proyecto; como también mediante el uso de interrupciones temporizadores visto durante las prácticas y las clases. El límite es nuestra imaginación y ganas de hacerlo.
* Para hacer una verificación de si un número es mayor o menor que otro en un microcontrolador como el 16f887 el uso de la operación resta es de mucha utilidad porque, como usa el complemento a 2, convierte la resta en una suma y, cuando se le resta a un numero una cantidad más grande que él; el ultimo bit del registro donde realizamos la operación siempre queda lleno con un numero ‘1’ ; luego podremos preguntar por el mediante instrucciones como **btfsc o btfss.**
* Gracias a la multiplexacion podemos “economizar” salidas aprovechando así de una mejor manera las salidas del microcontrolador la técnica aprovecha la limitación en frecuencia que tiene la vista la cual esta acotada a unos 30Hz. En mi caso, en el proyecto 2, no fue mi intención hacer que parezca imperceptible el parpadeo de los display’s sino mas bien dar la sensación de que la palabra está moviéndose letra por letra en los dos display’s

**9.- Recomendaciones**

* El entormo MPLAB es muy poderoso conociendo bien el uso de sus recursos para desarrollar; en lo personal me fue muy útil en la implementación del primer y segundo proyecto.
* En caso de presentar errores en la construcción de proyectos en MPLAB, recomendamos verificar en la ventana de informe de compilación, y hacer clic en el número del error. Me fue de mucha ayuda a la hora del depurado de mis proyectos.
* Usar siempre el “include” del PIC a programar, de este modo se facilita la programación pero recuerde que la correcta escritura del nombre de cada registro de microcontrolador (los nombres de estos son sensibles a las mayúsculas).
* Se recomienda verificar en el datasheet cualquier duda sobre una instrucción, o registro, puesto que, es el documento provisto por el fabricante y dicta las reglas como se ha de realizar su trabajo de programación.