FACULTAD DE INGENIERÍA, UNAM

LABORATORIO DE MICROCOMPUTADORASS

SEMESTRE 2023-2

GRUPO 11

PREVIO PRÁCTICA 3

SISTEMA MÍNIMO MICROCONTROLADOR PIC16F877

NOMBRE DEL ALUMNO:

ARRIAGA MEJÍA JOSÉ CARLOS

PROFESOR

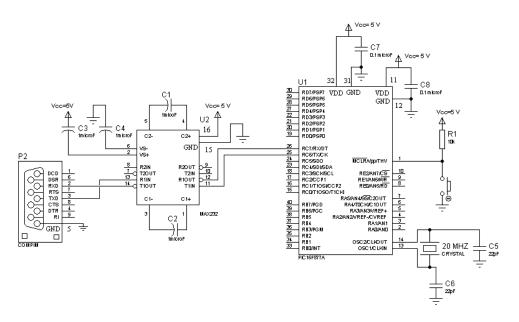
ING. ROMAN V. OSORIO COMPARAN

FECHA DE ENTREGA: 17 DE MARZO DE 2023 CALIFICACION

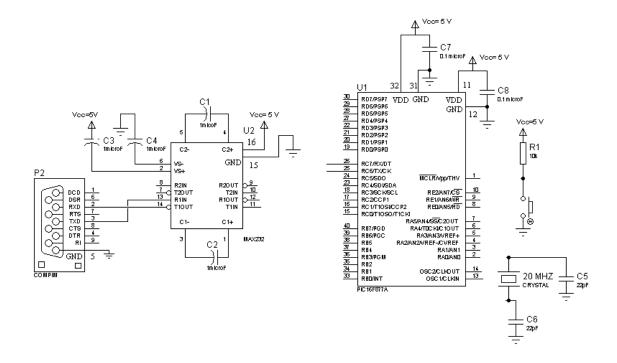
Objetivo

Desarrollar la habilidad de interpretación de esquemáticos. Conocer el diagrama del sistema mínimo del microcontrolador, el software de comunicación. Realizar aplicaciones con puertos paralelos en la modalidad de salida; ejecución de un programa en tiempo real.

1.- Revisar a detalle y en concordancia con el circuito 3.2, identificar las conexiones faltantes, discutir con sus compañeros y con su profesor(a) el impacto y función de estos.



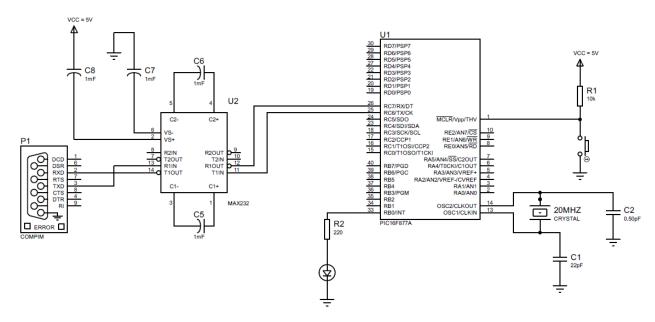
A.- Circuito 3.2



Podemos observar cómo falta la conexión entre el microcontrolador PIC16F877(A) y el circuito que permite la comunicación serie asíncrono, esta conexión se realza en los puertos 25 Y 26 del PIC y 11 y 12 también falta la conexión al pequeño circuito de reset, que es en puerto del Master Clean o MCLR. Por último, falta la conexión al reloj que genera la frecuencia de operación externa.

- 2.- Completar las conexiones faltantes, utilizando jumpers; cerciorar el alambrado correcto.
- 3.- Una vez resueltos las actividades anteriores, identificar la terminal PB0 del puerto B, realizar la conexión con la salida de una resistencia y un led.

Si realizamos las conexiones faltantes y conectando la resistencia y el LED nos quedaría de la siguiente manera



4.- Escribir, comentar e indicar que hace el siguiente programa.

processor 16f877	INICIO:BSF STATUS,RPO
include <p16f877.inc></p16f877.inc>	BCF STATUS,RP1
valor1 equ h'21'	MOVLW H'0'
valor2 equ h'22'	MOVWF TRISB
valor3 equ h'23'	BCF STATUS,RPO
cte1 equ 20h	CLRF PORTB
cte2 equ 50h	loop2 BSF PORTB,0
cte3 equ 60h	CALL retardo
ORG 0	BCF PORTB,0
GOTO INICIO	CALL retardo
ORG 5	GOTO loop2

retardo MOVLW cte1 GOTO uno

MOVWF valor1 DECFSZ valor2

tres MOVLW cte2 GOTO dos

MOVWF valor2 DECFSZ valor1

dos MOVLW cte3 GOTO tres

MOVWF valor3 RETURN

uno DECFSZ valor3 END

```
include <pl6f877.inc> ;Incluye biblioteca de la version del procesador
                                 ;Se crean 3 variables y 3 constantes
valor1 equ h'21'
valor2 equ h'22'
valor3 egu h'23'
ctel equ 20h
cte2 equ 50h
cte3 equ 60h
    ORG 0
                              Especifica el origen;
                           ;Manda al inicio del programa
     COTC INICIO
                              Especifica el origen del programa
     ORG 5
     CIO: ;etiqueta de inicio

BSF STATUS,RPO ;Coloca en 1 al bit RPO del registro STATUS

BCF STATUS,RP1 ;Coloca en 0 al bit RP1 del registro STATUS

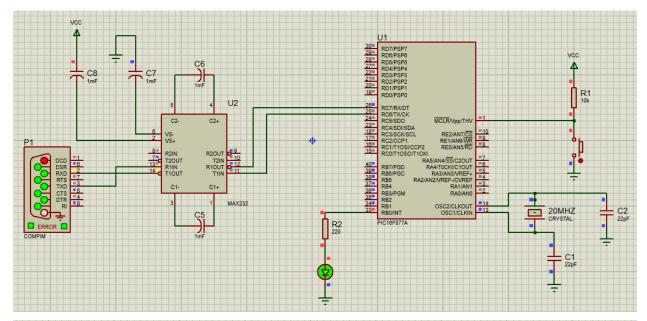
;banco 01
INICIO:
    MOVLW H'0' ;Carga 0 al registro W

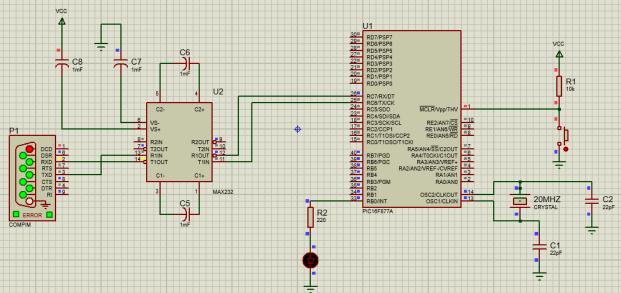
MOVWF TRISB ;Convierte el puerto B en pura salida, pasa el valor de W a los 8 bits del puerto B

BCF STATUS,RP0 ;Coloca en 0 al bit RP0 del registro STATUS

;banco 00
     CLRF PORTB
                               ;Limpia los datos del puerto B
loop2
                          ;Vuelve el bit 0 en entrada
;va a retardo
;Regresa el bit 0 a salida
     BSF PORTE, 0
     CALL retardo
     BCF PORTE, 0
     CALL retardo
                               ;va a retardo
                               ;se repite el loop
     COTC loop2
retardo
     MOVLW ctel
                                ; Carga a w la direccion de ctel
     MOVWF valor1
                                 ; carga a valorl el de w
     MOVLW cte2
                               ;Carga a w la direccion de cte3
     MOVWF valor2
                              ;carga a valor2 el de w
dos
                           ;Carga a w la direccion de cte3
     MOVLW cte3
     MOVWF valor3
                               ;carga a valor2 el de w
     DECFSZ valor3 ;Deciendo en 1 el valor de valor3
GOTC uno ;regresa a la etiqueta uno
DECFSZ valor2 ;Deciendo en 1 el valor de valor2
GOTC dos ;regresa a la etiqueta dos
DECFSZ valor1 ;Deciendo en 1 el valor de valor1
GOTC tres ;regresa a la etiqueta tres
RETURN ;retorna al ultimo llamado
     RETURN
                                ;retorna al ultimo llamado
     END
```

5.- Ensamblar y cargar el programa anterior en el microcontrolador; que es lo que puede visualizar.





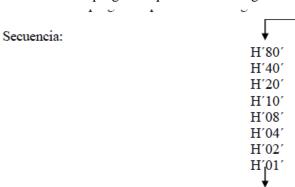
Al cargar el programa en Proteus podemos ver como nuestra salida ahora es una salida intermitente.

- 6.- En el programa, modifique el valor de cte1 a 8h, ensamblar y programar; que sucede y porque? La intermitencia es mas rápida, debido a que el retraso creado ha sido disminuido, ya que tiene que recorrer menos veces el valor 1 que paso a ser h'20' a h'8'
- 7.- Modifique cte1 a 80h; ensamblar y programar, ¿existe algún cambió?
- Si, la intermitencia es mucho más lenta.
- 8.- Modificar el programa anterior, para que ahora se actualice el contenido de todos los bits del puerto B y se genere una rutina de retardo de un segundo.

```
processor 16f877
#include <pl6f877a.inc>
valor1 equ h'21'
valor2 equ h'22'
valor3 equ h'23'
ctel equ 0xe0
cte2 equ 50h
cte3 equ 60h
   ORG 0
    GOTC INICIO
    ORG 5
INICIO:
   BSF STATUS, RPO ; Nos movemos al banco 01
   BCF STATUS, RP1 ; Para cambiar los TRIS
    MOVLW H'0' ; Se declara B como salidas
   MOVWF TRISB
   BCF STATUS, RPO
   CLRF PORTE ; Se limpia lo que haya en B
loop2
   MOVLW 0x00 ; Movemos 0s a todo PORTB, apagado
   MOVWE PORTB
    CALL retardo ; Retardo de 1 segundo
   MOVLW Oxff ; Movemos 1s a todo PORTB, encendido
   MOVWE PORTB
   CALL retardo ; Retardo de 1 segundo
    GOTC loop2 ; Se repite todo
retardo
    MOVLW ctel
    MOVWF valor1
    MOVLW cte2
    MOVWF valor2
   MOVLW cte3
    MOVWF valor3
   DECFSZ valor3
    GOTC uno
    DECFSZ valor2
    GOTC dos
    DECFSZ valor1
    GOTC tres
    RETURN
    END
```

Para cambiar el tiempo que dura el retraso modificamos la cte1 y además para la salida de los 8 Led tenemos que activar todos los bits del puerto B

9.- Realizar un programa que muestre la siguiente secuencia en el puerto B con retardos de ½ segundo.



El circuito empleado es el mismo que en el ejercicio anterior.

```
processor 16f877
#include <pl6f877a.inc>
valor1 equ h'21'
valor2 equ h'22'
valor3 equ h'23'
ctel equ 90h
cte2 equ 50h
cte3 equ 60h
   ORG 0
   COTC INICIO
   ORG 5
INICIO:
   BSF STATUS, RPO
   BCF STATUS, RP1
   MOVLW H'0'
   MOVWF TRISB
   BCF STATUS, RP0
   CLRF PORTB
EJER:
   BCF STATUS, C ; Limpia carry
   MOVLW 0x80 ; W <- 0x80
   MOVWF PORTE ; ROTA <- (W)
   CALL retardo
ES_CERO:
   BTFSC PORTE, 0 ; checa bit 0 de PORTB
   GOTC EJER ; sigue con corrimiento
   RRF PORTE, 1
   CALL retardo
   goto ES_CERO
retardo
   MOVLW ctel
   MOVWF valor1
tres
    MOVLW cte2
   MOVWF valor2
dos
   MOVLW cte3
   MOVWF valor3
   DECFSZ valor3
   GOTC uno
   DECFSZ valor2
    GOTO dos
    DECFSZ valor1
    GOTO tres
    RETURN
```

10.- Realizar un programa que controle el funcionamiento de dos semáforos; cada estado tendrá una duración de 2 segundos.



Estado	Salida
1	V1, R2
2	A1, R2
3	R1, V2
4	R1, A2

```
processor 16f877
#include <pl6f877a.inc>
valor1 equ h'21'
valor2 equ h'22'
valor3 equ h'23'
ctel equ 0xe0
cte2 equ 0xa0
cte3 equ 60h
   ORG 0
   COTC INICIO
   ORG 5
INICIO:
   BSF STATUS, RPO
   BCF STATUS, RP1
   MOVLW H'0'
   MOVWF TRISB
   BCF STATUS, RPO
   CLRF PORTB
loop2
   MOVLW b'01000001' ;Semáforo 1 Verde, Semaforo 2 Rojo
   MOVWE PORTB
   CALL retardo ; Retardo de 2 seg
   MOVLW b'00100001' ; Semáforo 1 Amarillo, Semaforo 2 Rojo
   MOVWF PORTB
   CALL retardo ; Retardo de 2 seg
   MOVLW b'00010100' ; Semáforo 1 Rojo, Semaforo 2 Verde
   MOVWF PORTB
   CALL retardo ; Retardo de 2 seg
   MOVLW b'00010010' ; Semaforo 1 Rojo, Semaforo 2 Amarillo
   MOVWE PORTE
   CALL retardo ; Retardo de 2 seg
   GOTC loop2 ; Se repite
reterdo
retardo
   MOVLW ctel
   MOVWF valor1
   MOVLW cte2
   MOVWF valor2
dos
   MOVLW cte3
   MOVWF valor3
uno
   DECFSZ valor3
    GOTC uno
    DECFSZ valor2
    GOTC dos
    DECFSZ valor1
    GOTO tres
    RETURN
    END
```