UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

Práctica 4: Puertos Paralelos E/S

INTEGRANTES DE EQUIPO Y GRUPO DE TEORÍA

López Becerra Ricardo - 420053710 - GRUPO 3
Navarrete Zamora Aldo Yael - 317242409 - GRUPO 4

ASIGNATURA

Laboratorio de Microcomputadoras

GRUPO DE LABORATORIO

8

FECHA DE REALIZACIÓN

16 de marzo del 2023

FECHA DE ENTREGA

30 de marzo del 2023

1. Desarrollo

Para cada uno de los siguientes ejercicios, realizar los programas solicitados y comprobar el funcionamiento de ellos.

1. Empleando dos puertos paralelos del microcontrolador PIC, uno de ellos configurado como entrada y el otro como salida; realizar un programa que de acuerdo al valor del bit menos significativo del puerto A, se genere la acción indicada en el puerto B.

Valor PA0	Acción puerto B
0	00000000
1	11111111

Algoritmo

Para el ejercicio partimos de la base de la práctica anterior, en donde configuramos el PORT B como salida, la diferencia es, que, para este caso, ahora también tenemos que configurar el PORT A como puerto de entrada.
Cabe mencionar que, hay que configurar el puerto ADCON1, este puerto nos especifica si nuestras entradas están definidas como digitales, en este caso así será. Dentro del programa, se empleará un loop infinito, en el que se verifica si el bit menos significiativo del PORT A (es decir, el primero de derecha a izquierda) se encuentra encendido o apagado, dependiendo del valor de este, el programa se dirigirá ya sea a 'APAGAR_LEDS' o 'PRENDER_LEDS', cada una de estas etiquetas mandará al PORT B un conjunto de bits encendidos o apagados para mostrar en la salida.

Código comentado

```
processor 16f877
#include <p16f877.inc>
; REGISTROS DEFINIDOS PARA
; ALMACENAR LOS CONTADORES.
valor1 equ h'21' ;
valor2 equ h'22' ;
valor3 equ h'23' ;
;CONSTANTES QUE DEFINEN EL RETARDO
cte1 equ h'ff'   ;
cte2 equ 50h     ;
cte3 equ 60h     ;
        ORG 0
GOTO INICIO
        ORG 5
;INICIO DEL PROGRAMA
INICIO:
    ;LIMPIAMOS EL PORTA
        CLRF PORTA
    ; CAMBIAMOS AL BANCO 1
        BSF STATUS,RP0
        BCF STATUS,RP1
```

```
;TRISB = SALIDA (8 BITS)
    MOVLW H'00'   ;
    MOVWF TRISB       ;
; ENTRADAS DIGITALES.
; 06H,07H <- DIGITALES
    MOVLW 06H
    MOVWF ADCON1
; TRISA = ENTRADA (6 BITS)
    MOVLW 3FH         ;
    MOVWF TRISA
; CAMBIAMOS AL BANCO 0
    BCF STATUS,RP0
;LIMPIAMOS EL PORTA
    MOVLW B'00000000';
    MOVWF PORTB

loop1:
;CHECAMOS EL PORTA, EN EL BIT 0
    BTFSC PORTA, 0
;PRENDIDO? -> VAMOS A 'PRENDER_LEDS'
    GOTO PRENDER_LEDS
;CASO CONTRARIO, VAMOS A 'APAGAR_LEDS'
    GOTO APAGAR_LEDS
;REPETIMOS LOOP.
    goto loop1

PRENDER_LEDS:
    MOVLW B'11111111'       ;
    MOVWF PORTB             ; MANDAMOS A PORTB PUROS BITS ENCENDIDOS
    GOTO loop1              ; REGRESAMOS AL LOOP

APAGAR_LEDS
    CLRF PORTB              ; APAGAMOS LOS BITS DEL PORTB
    GOTO loop1              ;

retardo
    MOVLW cte1              ;
    MOVWF valor1            ;
tres
    MOVLW cte2              ;
    MOVWF valor2            ;
dos
    MOVLW cte3              ;
    MOVWF valor3            ;
uno
    DECFSZ valor3           ;
    GOTO uno                ;
    DECFSZ valor2           ;
    GOTO dos                ;
    DECFSZ valor1           ;
```

```
GOTO tres                ;
RETURN                   ;
END                      ; CODIGO DEL RETARDO.
```

2. Escribir un programa, el cuál realice las siguientes acciones de control indicadas, para lo cuál requiere trabajar un puerto de entrada y otro puerto de salida, usar los sugeridos en el ejercicio anterior; generar retardos de $\frac{1}{2}$ seg., en las secuencias que lo requieran.

   Para realizar este programa nos basamos en el realizado en el ejercicio anterior, ya que dos de los estados de este programa ya se programaron en el anterior.

Algoritmo

Se concluyó que podríamos trabajar este ejercicio mediante la comparación en un ciclo infinito de los valores que se presenten en el PORTA, podríamos hacer las 4 posibles comparaciones con la instrucción **SUBWF**, y dependiendo de qué opción sea la elegida, nos iremos a alguna u otra etiqueta.
En uno de los siguientes casos, si la entrada es:

- 0: En esta opción el puerto B simplemente se limpiará. Esto indica que a la salida no veremos ningún LED encendido.
- 1: Para este caso será lo contrario al anterior, moveremos un 0xFF al PORTB, indicando que todos los LEDs estarán prendidos.
- 2: Para este caso la idea fue crear un ciclo que se repite mientras la entrada sigue siendo 2 y mientras no se complete un ciclo de corrimiento. Dentro de este loop se hacen las verificaciones de las condiciones anteriores y si no se cumplen, se realiza un corrimiento a la derecha en el puerto B.
- 3: El corrimiento a la izquierda es muy similar al anterior. También hay un ciclo que se repite mientras la entrada sigue siendo 3 y mientras no se complete un ciclo de corrimiento. Dentro de este loop se hacen las verificaciones de las condiciones y si no se cumplen se realiza un corrimiento a la izquierda en el puerto B.
- 4: Para el zig-zag lo que se hizo fue crear dos subrutinas, una parecida al corrimiento a la derecha y otra parecida al corrimiento a la izquierda. Cuando uno de los corrimientos termina, se llama al otro y así sucesivamente hasta que dentro de alguno de los dos se detecte un cambio del modo de ejecución del programa.
- 5: El código para este caso es muy similar al ya realizado en prácticas anteriores. En un loop primero se encienden todos los bits del puerto B, luego se hace un retardo, luego se apagan todos los bits del puerto B y finalmente se hace un segundo retardo.

Código comentado

```
processor 16f877
include <p16f877.inc>
valor1 equ h'21'
valor2 equ h'22'
valor3 equ h'23'
cte1 equ h'ff'
cte2 equ 50h
cte3 equ 60h
```

```
        ORG 0
GOTO INICIO
        ORG 5
INICIO:
        ; Limpia los bits del puerto A
        CLRF PORTA
        ;Selecciona el banco de memoria 1
        BSF STATUS,RP0
        BCF STATUS,RP1
        ;Programa el puerto B como salida
        MOVLW H'00'
        MOVWF TRISB
        ;Se indica que que se usaran entradas
        ;digitales
        MOVLW 06H
        MOVWF ADCON1
        ;Se configura al puerto A como entrada
        MOVLW 3FH
        MOVWF TRISA
        ;Se regresa al banco 0
        BCF STATUS,RP0
        MOVLW B'00000000'
        ;Se limpia la memoria del puerto B
        MOVWF PORTB


;En este loop se verifica a que caso se quiere
;entrar
loop1:

        ;Verifica si se quiere apagar los leds
        MOVLW h'00'
        SUBWF PORTA, W
        BTFSC STATUS, Z
        GOTO APAGAR_LEDS
        ;Verifica si se quiere prender todos los
        ;leds
        MOVLW H'01'
        SUBWF PORTA, W
        BTFSC STATUS, Z
        GOTO PRENDER_LEDS
        ;Verifica si se quiere realizar corrimiento
        ;a la derecha
        MOVLW H'02'
        SUBWF PORTA, W
        BTFSC STATUS, Z
        GOTO RIGHT_SHIFT
        ;Verifica si se quiere realizar corrimiento
        ; al a izquierda
        MOVLW H'03'
```

```
        SUBWF PORTA, W
        BTFSC STATUS, Z
        GOTO LEFT_SHIFT
        ; Verifica si se quiere realizar zig-zag.
        MOVLW H'04'
        SUBWF PORTA, W
        BTFSC STATUS, Z
        GOTO ZIG_ZAG_1
        ;Verifica si se quiere comportamiento
        ;intermitente
        MOVLW H'05'
        SUBWF PORTA, W
        BTFSC STATUS, Z
        GOTO INTERMITENTE

goto loop1

;Coloca todos los bits del puerto B en 1
PRENDER_LEDS:
        MOVLW B'11111111'
        MOVWF PORTB
        GOTO loop1

;Coloca todos los bits del puerto B en 0
APAGAR_LEDS:
        CLRF PORTB
        GOTO loop1


RIGHT_SHIFT:
        ;Se prende el ultimo bit del puerto
        ; B
        MOVLW B'10000000'
        MOVWF PORTB
        ;Limpia el carry para evitar errores
        BCF STATUS, C
LOOP_RS:
        ;Verifica si el contedio del puerto B es cero,
        ; De ser el caso termino el corrimiento, por lo
        ; que debe empezar de nuevo.
        MOVLW H'00'
        SUBWF PORTB
        BTFSC STATUS, Z
        GOTO RIGHT_SHIFT
        ; LLama a retardo para hacer visible la accion
        CALL retardo
        ;verfica si se quiere acceder a otro modo. Si es
        ;asi regresa al loop principal
        MOVLW H'02'
        SUBWF PORTA, W
```

```
        BTFSS STATUS, Z
        GOTO loop1
        ;limpa el contenido del carry
        BCF STATUS, C
        ;Realiza un corrimiento a la derecha
        RRF PORTB

        goto LOOP_RS

LEFT_SHIFT:
        ;Se prende el primer bit del puerto
        ; B
        MOVLW B'00000001'
        MOVWF PORTB
LOOP_LS:
        ;Verifica si el contedio del puerto B es cero,
        ; De ser el caso termino el corrimiento, por lo
        ; que debe empezar de nuevo.
        MOVLW H'00'
        SUBWF PORTB
        BTFSC STATUS, Z
        GOTO LEFT_SHIFT
        ; LLama a retardo para hacer visible la accion
        CALL retardo
        ;verfica si se quiere acceder a otro modo. Si es
        ;asi regresa al loop principal
        MOVLW H'03'
        SUBWF PORTA, W
        BTFSS STATUS, Z
        GOTO loop1
        ;limpa el contenido del carry
        BCF STATUS, C
        ;Realiza un corrimiento a la izquierda
        RLF PORTB
        goto LOOP_LS

; Es casi identico al corrimento a la derecha, pero
; en lugar de ejecutarse a si mismo cuando termina
; el corrimiento, ejecuta a ZIG_ZAG_2 que es casi
;identico al corrimiento a la izquierda
ZIG_ZAG_1:
        MOVLW B'10000000'
        MOVWF PORTB
        BCF STATUS, C
LOOP_ZZ_1:
        MOVLW H'00'
        SUBWF PORTB
        BTFSC STATUS, Z
        GOTO ZIG_ZAG_2
        CALL retardo
```

```
        MOVLW H'04'
        SUBWF PORTA, W
        BTFSS STATUS, Z
        GOTO loop1
        BCF STATUS, C
        RRF PORTB
        goto LOOP_ZZ_1

; Es casi identico al corrimento a la izquierda, pero
; en lugar de ejecutarse a si mismo cuando termina
; el corrimiento, ejecuta a ZIG_ZAG_1 que es casi
;identico al corrimiento a la derecha
ZIG_ZAG_2:
        MOVLW B'00000001'
        MOVWF PORTB
        ;BCF STATUS, C
LOOP_ZZ_2:
        MOVLW H'00'
        SUBWF PORTB
        BTFSC STATUS, Z
        GOTO ZIG_ZAG_1
        CALL retardo
        MOVLW H'04'
        SUBWF PORTA, W
        BTFSS STATUS, Z
        GOTO loop1
        BCF STATUS, C
        RLF PORTB
        goto LOOP_ZZ_2

;Hace parpadear todos los leds de manera intermitente
INTERMITENTE:
        ;Verifica si se quiere acceder a otro modo y si es
        ; asi regresa al loop principal
        MOVLW H'05'
        SUBWF PORTA, W
        BTFSS STATUS, Z
        GOTO loop1
        ;Prende todos los leds
        MOVLW H'FF'
        MOVWF PORTB
        CALL retardo
        ;Apaga todos los leds
        CLRF PORTB
        CALL retardo
        goto INTERMITENTE


retardo
        MOVLW cte1
```

```
        MOVWF valor1
tres
        MOVLW cte2
        MOVWF valor2
dos
        MOVLW cte3
        MOVWF valor3
uno
        DECFSZ valor3
        GOTO uno
        DECFSZ valor2
        GOTO dos
        DECFSZ valor1
        GOTO tres
        RETURN
        END
```

2. Conclusiones

- López Becerra Ricardo: Esta práctica fue complementaria de lo visto en la anterior ya que utilizamos conceptos similares. Lo que aprendimos fue a programar uno de los puertos como entrada para poder controlar el flujo del programa desde afuera. Esto lo realizamos con ayuda del registro ADCON que nos ayuda a señalar que usaremos entradas digitales y con el registro TRISX, que nos ayuda a indicar en que modo queremos utilizar un puerto. Con estos conocimientos, pudimos hacer un programa más complejo que combina varios de otros programas realizados en prácticas anteriores.
- Navarrete Zamora Aldo Yael: En este reporte puedo rescatar información aprendida de la práctica anterior, como la configuración y empleo del puerto b como salida. A diferencia de la práctica anterior, en esta práctica configuramos de forma exitosa al puerto a como entrada, esta entrada en el PIC16F877A representaban los dipswitches.
  El primer ejercicio fue de mucha ayuda para poder enfrentar al segundo, en este último ejercicio reciclamos las soluciones vistas en la práctica anterior para hacer el corrimiento de bits a la derecha y a la izquierda.