Evaluación 3er Parcial

Departamento de Ciencias de la Computación

Asignatura:

"Organización Computacional"

Profesor:

Luis Alejandro Flores Oropeza

Alumno:

Luis Pablo Esparza Terrones
ID: *182563*
Juan Francisco Gallo Ramírez
ID: *232872*
Luis Ángel Soto Alderete
ID: *339104*
Pablo Emilio Soto Parada
ID: *284961*

Ingeniería en Computación Inteligente

PROGRAMA: *Péndulo cíclico.*

Programa del "PÉNDULO" para desplazar un bit de izquierda a derecha de forma cíclica con un tiempo de aproximado de un segundo entre cada desplazamiento.

```
;============================ PUNTO DE INICIO ==============
INICIO:
       MOV A, #10000000B
       MOV P2, A
       MOV R4, #7
;-------------- Recorrida a derecha ------
DERECHA:
       CALL DELAY
       RR A
       MOV P2, A
       DJNZ R4, DERECHA

       MOV R4, #7
;-------------- Recorrida a izquierda ----
IZQUIERDA:
       CALL DELAY
       RL A
       MOV P2, A
       DJNZ R4, IZQUIERDA

       JMP INICIO
;============================ CÓDIGO PARA DELAY 1 SEG. =====
DELAY:
       MOV R2, #0FFH
CICLO3:       MOV R1, #0FFH
CICLO2: MOV R0, #8
CICLO1: DJNZ R0, CICLO1
       DJNZ R1, CICLO2
       DJNZ R2, CICLO3
       RET
END
```

PROGRAMA: *Contador cíclico (00-99).*

Programa que mediante los puertos de salida controla dos displays de 7 segmentos para hacer un conteo ascendente de 0 al 99 de forma cíclica. El conteo de los números será de un segundo en un segundo aproximadamente.

```
;========================== PUNTO DE INICIO ==============
;---------------- Cuenta de Decenas ------
CUENTA_DECENAS:
        MOV P0, #11000000B
        CALL CUENTA_UNIDADES
        MOV P0, #11111001B
        CALL CUENTA_UNIDADES
        MOV P0, #10100100B
        CALL CUENTA_UNIDADES
        MOV P0, #10110000B
        CALL CUENTA_UNIDADES
        MOV P0, #10011001B
        CALL CUENTA_UNIDADES
        MOV P0, #10010010B
        CALL CUENTA_UNIDADES
        MOV P0, #10000010B
        CALL CUENTA_UNIDADES
        MOV P0, #11111000B
        CALL CUENTA_UNIDADES
        MOV P0, #10000000B
        CALL CUENTA_UNIDADES
        MOV P0, #10010000B
        CALL CUENTA_UNIDADES
        JMP CUENTA_DECENAS

;---------------- Cuenta de Unidades ------
CUENTA_UNIDADES:
        MOV P2, #11000000B
        CALL DELAY
        MOV P2, #11111001B
        CALL DELAY
        MOV P2, #10100100B
        CALL DELAY
        MOV P2, #10110000B
        CALL DELAY
        MOV P2, #10011001B
        CALL DELAY
        MOV P2, #10010010B
        CALL DELAY
        MOV P2, #10000010B
        CALL DELAY
        MOV P2, #11111000B
        CALL DELAY
        MOV P2, #10000000B
        CALL DELAY
        MOV P2, #10010000B
        CALL DELAY
        RET

;========================== CÓDIGO PARA DELAY 1 SEG. =====
DELAY:
        MOV R2, #8
CICLO3: MOV R1, #0FFH
CICLO2: MOV R0, #0FFH
CICLO1: DJNZ R0, CICLO1
        DJNZ R1, CICLO2
        DJNZ R2, CICLO3
        RET
END
```

PROGRAMA: *Comparador de 8 bits.*

Escribir un programa que lea los datos de dos puertos en binario y coloque el MAYOR DE LOS DOS NÚMEROS (en binario) en un tercer puerto, en caso de que los dos números sean iguales, entonces todos los LEDS del puerto de salida PARPADEARÁN hasta que los números de entrada en los puertos cambien.

```
;========================== PUNTO DE INICIO ==============
INICIO:
       CLR C
       MOV A, P1
       MOV B, P3

;--- Resta para hacer comparación ------
       SUBB A, B

       JZ IGUAL

       JNC MAYOR_P1
       JC MAYOR_P3

;-------- Código cuando son iguales ------
IGUAL:
       MOV P2, #11111111B
       CALL DELAY
       MOV P2, #0
       CALL DELAY
       JMP INICIO

;-------- Código cuando P1 es mayor ------
MAYOR_P1:
       MOV P2, P1
       JMP INICIO

;-------- Código cuando P3 es mayor ------
MAYOR_P3:
       MOV P2, P3
       JMP INICIO

;========================== CÓDIGO PARA DELAY 1 SEG. =====
DELAY:
       MOV R2, #0FFH
CICLO3:      MOV R1, #0FFH
CICLO2: MOV R0, #4
CICLO1: DJNZ R0, CICLO1
       DJNZ R1, CICLO2
       DJNZ R2, CICLO3
       RET
END
```

PROGRAMA: *Sumador de 8 bits.*

Escribir un programa que lea los datos de dos puertos en binario y coloque la SUMA DE LOS DOS NÚMEROS (en binario) en un tercer puerto, en caso de que la suma no se pueda expresar en 8 bits, entonces todos los LEDS del puerto de salida PARPADEARÁN hasta que los números de entrada en los puertos cambien y no superen un resultado de 8 bits.

```
;========================== PUNTO DE INICIO ==============
INICIO:
       CLR C
       MOV A, P1
       MOV B, P3

;----- Suma ----------------------------------------
       ADD A, B
       JC MAYOR

       MOV P2, A

       JMP INICIO

;----- Código cuando la suma es mayor a 8 bits ------
MAYOR:
       MOV P2, #11111111B
       CALL DELAY
       MOV P2, #0
       CALL DELAY
       JMP INICIO

;========================== CÓDIGO PARA DELAY 1 SEG. =====
DELAY:
       MOV R2, #0FFH
CICLO3:      MOV R1, #0FFH
CICLO2: MOV R0, #4
CICLO1: DJNZ R0, CICLO1
       DJNZ R1, CICLO2
       DJNZ R2, CICLO3
       RET
END
```

PROGRAMA: *Interrupciones con display.*

Colocar 2 INTERRUPCIONES al programa de conteo cíclico ascendente del 0 al 99, se tratará de una interrupción de baja prioridad con la que el programa principal dejará de contar y pasará a hacer parpadear por 5 segundos a todos los leds de los display de 7 segmentos mientras que la segunda interrupción, de alta prioridad, hará que el microcontrolador deje de hacer lo que esté haciendo para mostrar en los displays de 7 segmentos las letras, A, B, C, D, E y F en intervalos de un segundo entre ellas. Luego que se dejen de atender las interrupciones, el programa principal del microcontrolador se deberá seguir ejecutando en el punto que estaba antes de atender a la o las interrupciones.

```
ORG 0000H
       LJMP PRINCIPAL
ORG 0003H
       LJMP INT_0
ORG 0013H
       LJMP INT_1

;========================== PUNTO DE INICIO ==============
PRINCIPAL:
       MOV IP, #04H
       MOV IE, #85H

;---------------- Cuenta de Decenas ------
       CUENTA_DECENAS:
              MOV P0, #11000000B
              MOV R4, #11000000B
              CALL CUENTA_UNIDADES
              MOV P0, #11111001B
              MOV R4, #11111001B
              CALL CUENTA_UNIDADES
              MOV P0, #10100100B
              MOV R4, #10100100B
              CALL CUENTA_UNIDADES
              MOV P0, #10110000B
              MOV R4, #10110000B
              CALL CUENTA_UNIDADES
              MOV P0, #10011001B
              MOV R4, #10011001B
              CALL CUENTA_UNIDADES
              MOV P0, #10010010B
              MOV R4, #10010010B
              CALL CUENTA_UNIDADES
              MOV P0, #10000010B
              MOV R4, #10000010B
              CALL CUENTA_UNIDADES
              MOV P0, #11111000B
              MOV R4, #11111000B
              CALL CUENTA_UNIDADES
              MOV P0, #10000000B
              MOV R4, #10000000B
              CALL CUENTA_UNIDADES
              MOV P0, #10010000B
              MOV R4, #10010000B
              CALL CUENTA_UNIDADES
              JMP CUENTA_DECENAS

;--------------- Cuenta de Unidades ------
       CUENTA_UNIDADES:
              MOV P2, #11000000B
              MOV R5, #11000000B
              CALL DELAY
              MOV P2, #11111001B
              MOV R5, #11111001B
              CALL DELAY
              MOV P2, #10100100B
              MOV R5, #10100100B
```

```
                CALL DELAY
                MOV P2, #10110000B
                MOV R5, #10110000B
                CALL DELAY
                MOV P2, #10011001B
                MOV R5, #10011001B
                CALL DELAY
                MOV P2, #10010010B
                MOV R5, #10010010B
                CALL DELAY
                MOV P2, #10000010B
                MOV R5, #10000010B
                CALL DELAY
                MOV P2, #11111000B
                MOV R5, #11111000B
                CALL DELAY
                MOV P2, #10000000B
                MOV R5, #10000000B
                CALL DELAY
                MOV P2, #10010000B
                MOV R5, #10010000B
                CALL DELAY
                RET

;========================== INTERRUPCIÓN #1 ==============
INT_0:
        MOV A, R4
        MOV R6, A
        MOV B, R5
        MOV R7, B

        MOV A, #5

;------------- Parpadeo del display ------
        PARPADEO:
                MOV P0, #11111111B
                MOV P2, #11111111B
                MOV R4, #11111111B
                MOV R5, #11111111B
                CALL DELAY_MEDIO
                MOV P0, #00000000B
                MOV P2, #00000000B
                MOV R4, #00000000B
                MOV R5, #00000000B
                CALL DELAY_MEDIO
                DJNZ ACC, PARPADEO

        MOV A, R6
        MOV R4, A
        MOV P0, R4
        MOV A, R7
        MOV R5, A
        MOV P2, R7

        RETI

;========================== INTERRUPCIÓN #2 ==============
INT_1:

;------------- Letra "A" -----------------
        MOV P0, #10001000B
        MOV P2, #10001000B
        CALL DELAY
;------------- Letra "B" -----------------
        MOV P0, #10000011B
        MOV P2, #10000011B
        CALL DELAY
;------------- Letra "C" -----------------
        MOV P0, #11000110B
        MOV P2, #11000110B
        CALL DELAY
;------------- Letra "D" -----------------
        MOV P0, #10100001B
        MOV P2, #10100001B
        CALL DELAY
;------------- Letra "E" -----------------
        MOV P0, #10000110B
        MOV P2, #10000110B
```

```
        CALL DELAY

        MOV P0, R4
        MOV P2, R5
        RETI

;============================ CÓDIGO PARA DELAY 1 SEG. =====
DELAY:
        MOV R2, #8
CICLO3: MOV R1, #0FFH
CICLO2: MOV R0, #0FFH
CICLO1: DJNZ R0, CICLO1
        DJNZ R1, CICLO2
        DJNZ R2, CICLO3
        MOV R2, #2
        MOV R1, #0FFH
        MOV R0, #0FFH
        RET

;========================= CÓDIGO PARA DELAY 0.5 SEG. =====
DELAY_MEDIO:
        MOV R2, #4
CICLO3_M: MOV R1, #0FFH
CICLO2_M: MOV R0, #0FFH
CICLO1_M: DJNZ R0, CICLO1_M
          DJNZ R1, CICLO2_M
          DJNZ R2, CICLO3_M
         MOV R2, #2
          MOV R1, #0FFH
          MOV R0, #0FFH
        RET
END
```