

1 seg.

MOV al, 1 ~ del 1 seg.
OUT COMP, al

PRÁCTICA 3 - RESOLUCIÓN

Interrupciones por Hardware

Objetivos: Comprender la utilidad de las interrupciones por hardware y el funcionamiento del Controlador de Interrupciones Programable (PIC). Utilizar el TIMER, F10, y el dispositivo de handshaking (HAND-SHAKE) mediante interrupciones para el intercambio de información entre el microprocesador y el mundo exterior.

Parte 1: PIC y Vector de Interrupciones

1) **Registros del PIC** Dados los siguientes registros del PIC indique su función. Además, para el valor de ejemplo, indique qué significa en términos del funcionamiento del programa

Dirección	Registro	Función	Valor de Ejemplo
20h	EOI	Fin de Interrupción. Solo para escribir, se debe mandar el comando de fin de interrupción cuando se terminó de servir la interrupción. (Valor 20h)	No aplica
21h	IMR	Registro de Mascara de Interrupciones. Permite enmascarar selectivamente las interrupciones que va a recibir el PIC. Cada bit de este registro representa una línea de interrupción (bit 0 se asocia a la entrada INT0 ... bit 7 se asocia a la entrada INT7). Si el valor es puesto en 1, esa interrupción estará inhibida.	1111 1101 Esta habilitada la interrupción INT1 (bit 1 en 0). Esa interrupción está conectada al Timer.
22h	IRR	Registro de Petición de Interrupciones. Almacena las interrupciones pendientes. Cada bit de este registro representa una línea de interrupción (bit 0 se asocia a la entrada INT0 ... bit 7 se asocia a la entrada INT7). Cuando la interrupción se satisface, el bit de dicha línea se pone a cero.	0000 0011 Se recibió y están pendientes de ser atendidas las interrupciones INT0 e INT1. Esa interrupción está conectada a la tecla F-10 y al Timer.
23h	ISR	Registro de Interrupción en	0000 0100

Parte 2:

Arquitectura de Computadoras

2024

F10	1111 1110	8	No importa	No importa	32
Timer	1111 1101	No importa	10	No importa	40
HANDSHAKE	1111 1011	No importa	No importa	5	20
Sin interrupciones	1111 1111	No importa	No importa	No importa	-
F10 y Timer	1111 1100	10	20	No importa	40 y 80
TIMER y HANDSHAKE	1111 1001	No importa	4	8	16 y 32
F10, TIMER y HANDSHAKE	1111 1000	5	10	15	20, 40, 60

Nota: En el caso de haber más de un dispositivo en uso, indicarlos en el mismo orden en las columnas "Dispositivos" y "Dirección Vector". En el caso en que una columna no deba configurarse, indicarla como "No importa".

Parte 2: Interrupciones con la tecla F10, Timer y Handshake

- 3) **Contador de pulsaciones** El siguiente programa cuenta el número de veces que se presiona la tecla F10 y acumula este valor en el registro DX. El programa nunca termina, ya que ejecuta un ciclo infinito. Completar las instrucciones faltantes para que el programa funcione correctamente.

```
EOI EQU 20H
IMR EQU 21H
INT0 EQU 24H

N_F10 EQU 15

ORG 60
IP_F10 DW RUT_F10
```

```
ORG 2000H
CLI
MOV AL, 0FEH
OUT IMR, AL
MOV AL, N_F10
OUT INT0, AL
MOV DX, 0
STI
LAZO: JMP LAZO
```

```
ORG 3000H
RUT_F10: PUSH AX
        INC DX
        MOV AL, EOI
        OUT EOI, AL
        POP AX
        IRET
END
```

Luego de completarlo, verifique su funcionamiento en el simulador, y explicar detalladamente:

- A. ¿Qué hacen las instrucciones CLI y STI? ¿Qué podría suceder si no están las mismas y la persona que usa el programa presiona F10 rápidamente durante la configuración del PIC?

La instrucción **CLI** inhibe las interrupciones enmascarables, la instrucción **STI** las vuelve a habilitar. Las interrupciones enmascarables son las interrupciones gestionadas por el PIC.

Si no están las mismas mientras configuramos el PIC, la CPU podría recibir un pedido de interrupción de un vector que aún no este configurado (en este caso el registro INT0), eso podría hacer que la CPU busque la rutina manejadora de la interrupción en la dirección equivocada y se termine ejecutando código en otra dirección de memoria.

- B. ¿Por qué se usa el valor 0FEH en el programa?

El valor 0FEH lo usamos para configurar el registro **IMR** del PIC, para habilitar solamente la interrupción **F10** del simulador.

- C. ¿Qué instrucciones le indican al PIC que la interrupción ha terminado?

Arquitectura de Computadoras

```

MOV BX, OFFSET TEXTO
MOV AL, OFFSET FIN_TEXTO - OFFSET TEXTO
INT 7
    INC TERMINA
    MOV AL, 0FFh
    OUT IMR, AL
SIGUE:
    MOV AL, EOI
    OUT EOI, AL
    MOV AL, 0
    OUT CONT, AL
    POP BX
    POP AX
    IRET

ORG 2000H
CLI
    MOV AL, 0FDH
    OUT IMR, AL
    MOV AL, N_TIMER
    OUT INT1, AL
    MOV AL, 10
    OUT COMP, AL
    MOV AL, 0
    OUT CONT, AL
    STI
LAZO:
    CMP TERMINA, 0
    JZ LAZO
    INT 0
END
    
```

- d) **Desafío:** Modificar a) para que el primer mensaje se imprima luego de 1 segundo, el segundo luego de 2 segundos, el tercero luego de 3, y así sucesivamente, hasta que se espere 255 en el último mensaje, y el programa termine

```

EOI EQU 20H
IMR EQU 21H
INT1 EQU 25h

CONT EQU 10h
COMP EQU 11h

N_TIMER EQU 20

ORG 80
IP_F10 DW RUT_TIMER

ORG 1000H
TEXTO DB "Vamos las interrupciones!"
FIN_TEXTO DB ?
ESPERA DB 1

ORG 3000H
RUT_TIMER: PUSH AX
    PUSH BX
    MOV BX, OFFSET TEXTO
    MOV AL, OFFSET FIN_TEXTO - OFFSET TEXTO
    INT 7
    
```

Arquitectura de Computadoras

```
IMR EQU 21h
INT2 EQU 26h
EOI EQU 20h
```

```
ORG 40
DW RUT_HAND
```

```
ORG 3000h
RUT_HAND: MOV AL, [BX]
          OUT DATO, AL
          INC BX
          DEC CL
          JNZ FIN
          IN AL, EST
          AND AL, 07FH
          OUT EST, AL
          MOV AL, 0FFh
          OUT IMR, AL
FIN:      MOV AL, EOI
          OUT EOI, AL
          IRET
```

```
ORG 1000h
NUM_CAR DB 5
CAR DB ?
```

```
ORG 2000h
; PROGRAMA PRINCIPAL
MOV BX, OFFSET CAR
MOV CL, NUM_CAR
LAZO:  INT 6
       INC BX
       DEC CL
       JNZ LAZO
       MOV BX, OFFSET CAR
       MOV CL, NUM_CAR
       CLI
       MOV AL, 0FBh
       OUT IMR, AL
       MOV AL, 10
       OUT INT2, AL
       IN AL, EST
       OR AL, 080h
       OUT EST, AL
       STI
```

```
ESPERA: CMP CL, 0
       JNZ ESPERA
       IN AL, EST
       AND AL, 07FH
       OUT EST, AL
       INT 0
```

```
END
```

- d) Modificar b) de modo que cuando se presione F10 se cancele la impresión. En tal caso, deben desactivarse las interrupciones del HANDSHAKE para evitar que se envíen más caracteres a imprimir.

```
IMR EQU 21H
EOI EQU 20H
INT0 EQU 24H
INT2 EQU 26H

HAND EQU 40H
EST EQU 41H
DATO EQU 40H
N_HND EQU 10
N_F10 EQU 15

ORG 40
DW RUT_HND

ORG 60
DW RUT_F10

ORG 1000H
MSG DB "UNIVERSIDAD NACIONAL DE LA PLATA"
FIN DB ?
TERMINO DB 0

ORG 2000H
MOV BX, OFFSET MSG
MOV CL, OFFSET FIN - OFFSET MSG
CLI
MOV AL, 0FAH
OUT IMR, AL
MOV AL, N_F10
OUT INT0, AL
MOV AL, N_HND
OUT INT2, AL
MOV AL, 80H
OUT EST, AL
STI
LAZO: CMP TERMINO, 0
JZ LAZO
INT 0

ORG 3000H
RUT_HND: PUSH AX
MOV AL, [BX]
OUT DATO, AL
INC BX
DEC CL
JNZ FINAL
MOV AL, 0FFH
OUT IMR, AL
IN AL, EST
AND AL, 7FH
OUT EST, AL
MOV TERMINO, 1
FINAL: MOV AL, EOI
OUT EOI, AL
POP AX
```

Arquitectura de Computadoras

IRET

ORG 3200H

RUT_F10: PUSH AX

MOV AL, 0FFH

OUT IMR, AL

IN AL, EST

AND AL, 7FH

OUT EST, AL

MOV TERMINO, 1

MOV AL, EOI

OUT EOI, AL

POP AX

IRET

END

Impresora MSJ 255 MHz, cl 1 seg
 similar a a) $0x9,00004$: MOV al, 1 mod 1 seg
 { MOV al, 0 : OUT COMR, al
 OUT COMR, al

Arquitectura de Computadoras

```

OUT CB, AL
; inicializo strobe en 0
IN AL, PA
AND AL, 11111101b
OUT PA, AL

MOV CL, LARGO
MOV BX, OFFSET CADENA
; espero que busy=0
IN AL, PA
AND AL, 01h ; 1000 0000
JNZ POLL
; se que busy es 0, mandar caracer
MOV AL, [BX]
OUT PB, AL
; mandar flanco ascendente de strobe
IN AL, PA
OR AL, 00000010b
OUT PA, AL
; strobe en 0
IN AL, PA
AND AL, 11111101b
OUT PA, AL

INC BX
DEC CL
JNZ POLL
INT 0
  
```

- 2) **F10 y Luces** ★★ Escribir un programa que al presionar F10 encienda todas las luces, y al presionarlo nuevamente las apague. El programa nunca termina.

```

PA EQU 30h
PB EQU 31h
CA EQU 32h
CB EQU 33h

IMR EQU 21h
EOI EQU 20h
INT0 EQU 24h
N_F10 EQU 10

ORG 40
DW RUT_F10

ORG 1000h
ESTADO DB 0

ORG 3200h
RUT_F10: PUSH AX
        XOR ESTADO, 0FFh
        MOV AL, ESTADO
        OUT PB, AL
        MOV AL, EOI
  
```

Arquitectura de Computadoras

2024

```
OUT EOI, AL
POP AX
IRET

ORG 2000h
CLI
MOV AL, 0 ; puerto de datos todo salida
OUT CB, AL
OUT PB, AL
MOV AL, 0FEh
OUT IMR, AL
MOV AL, N_F10
OUT INT0, AL
STI
LOOP: JMP LOOP
END
```

- 3) **Ruleta, F10 y azar** ★★ Escribir un programa que permita seleccionar un dígito al azar para jugar a la ruleta. Para eso, el programa principal debe iterar continuamente, cambiando un valor de un registro desde el '0' hasta el '9' (y volviendo al '0' luego del '9'). Cuando se presiona F10, la letra queda "seleccionada" en base al valor del registro, y debe mostrarse de inmediato en la pantalla. Luego el programa termina.

```
IMR EQU 21h
EOI EQU 20h
INT0 EQU 24h
N_F10 EQU 10

ORG 40
DW RUT_F10

ORG 1000h
FINALIZADO DB 0
NUM DB ?

ORG 3200h
RUT_F10: PUSH AX
        MOV FINALIZADO, 1
        MOV AL, 0FFh
        OUT IMR, AL
        MOV AL, EOI
        OUT EOI, AL
        POP AX
        IRET

ORG 2000h
        CLI
        MOV AL, 0FEh
        OUT IMR, AL
        MOV AL, N_F10
        OUT INT0, AL
        STI
        MOV DL, '0'
LOOP:   CMP FINALIZADO, 1
        JZ ELEGIDO
        INC DL
```

9/14

mymin MSJ 255 news, cl 1 seg.

nilon a a)
 { ORG 2000H
 { MOV al, 0
 { OUT CONT, al
 { MOV al, 1 mod 1 seg
 { OUT COMP, al
 ORG 3000H

Arquitectura de Computadoras

```
CMP DL, '9'
JNZ LOOP
MOV DL, '0'
JMP LOOP
ELEGIDO: MOV BX, OFFSET NUM
          MOV [BX], DL
          MOV AL, 1
          INT 7
          INT 0
END
```

- 4) **F10, lectura e impresora con PIO** ★★★★★ Escribir un programa que debe mostrar en pantalla el mensaje "PRESIONE F10 PARA COMENZAR" y una vez que el usuario presiona F10, leer de teclado un mensaje de 10 caracteres. Este mensaje debe luego ser enviado a través de la pila a una subrutina para imprimirse en la impresora mediante el PIO. La configuración del PIO también debe hacerse en una subrutina aparte.

```
PA EQU 30h
PB EQU 31h
CA EQU 32h
CB EQU 33h

EOI EQU 20h
IMR EQU 21h
INT0 EQU 24h

N_F10 EQU 10

ORG 40
DW RUT_F10

ORG 1000h
MSJ_EMPEZAR DB "PRESIONE F10 PARA COMENZAR"
FIN_MSJ_EMPEZAR DB ?
EMPEZAR DB 0
MENSAJE DB ?

ORG 3000h
CONFIG_PIO: MOV AL, 0FDh
             OUT CA, AL
             MOV AL, 0
             OUT CB, AL
             ; inicializo strobe en 0
             IN AL, PA
             AND AL, 11111101b
             OUT PA, AL
             RET

ORG 3200h
RUT_F10: PUSH AX
          MOV EMPEZAR, 1
          MOV AL, EOI
          OUT EOI, AL
          POP AX
          IRET
```

Arquitectura de Computadoras

2024

ORG 3400h
LEER_CADENA: INT 6
 INC BX
 DEC CL
 JNZ LEER_CADENA
 RET

ORG 3600h
IMPRIMIR: MOV BX, SP
 ADD BX, 2
 MOV CL, [BX]
 ADD BX, 2
 MOV BX, [BX]

POLL: IN AL, PA
 AND AL, 01h ; 1000 0000
 JNZ POLL

 MOV AL, [BX]
 OUT PB, AL

 ; mandar flanco ascendente de strobe

 IN AL, PA
 OR AL, 00000010b
 OUT PA, AL

 ; strobe en 0

 IN AL, PA
 AND AL, 11111101b
 OUT PA, AL

 INC BX
 DEC CL
 JNZ POLL
 RET

ORG 2000h
 MOV BX, OFFSET MSJ_EMPEZAR
 MOV AL, OFFSET FIN_MSJ_EMPEZAR - OFFSET MSJ_EMPEZAR
 INT 7
 CLI
 MOV AL, 0FEh
 OUT IMR, AL
 MOV AL, N_F10
 OUT INT0, AL
 STI
ESPERA: CMP EMPEZAR, 0
 JZ ESPERA

 CALL CONFIG_PIO

 MOV BX, OFFSET MENSAJE
 MOV CL, 10
 CALL LEER_CADENA

 MOV BX, OFFSET MENSAJE

24
x4
96

abrutim

Imprimin M8J 255 veces, c/ 1 seg.

Similón a a) $\begin{cases} \text{ORG } 2000\text{h} \\ \text{MOV AL, 0} \\ \text{OUT CONT, AL} \end{cases}$ $\begin{cases} \text{MOV AL, 1 mod 1 seg} \\ \text{OUT COMP, AL} \end{cases}$
 $\text{ORG } 3000\text{h}$

Arquitectura de Computadoras

2024

```
PUSH BX
MOV CL, 10
PUSH CX
CALL IMPRIMIR
POP CX
POP AX
INT 0
END
```

- 5) **Timer y Luces** ★★ Escribir un programa que debe leer el estado de las llaves y enviarlo a una subrutina. La subrutina debe llamarse **LUCES12** y recibe el estado por referencia a través de la pila. Si el estado indica que la llave 7 (la de más a la izquierda) está prendida entonces debe encender todas las luces por 12 segundos y luego apagar todas las luces. En caso contrario no debe hacer nada.

```
PA EQU 30h
PB EQU 31h
CA EQU 32h
CB EQU 33h

ORG 1000h
ESTADO_LUCES DB ?

ORG 3000h
LUCES12: MOV BX, SP
        ADD BX, 2
        MOV BX, [BX]

        MOV AL, [BX]
        AND AL, 080h
        JZ FIN_LUCES12
        MOV AL, 0FFh
        OUT PB, AL
FIN_LUCES12: RET

ORG 2000h
        MOV AL, 0FFh
        OUT CA, AL
        MOV AL, 0
        OUT CB, AL

        IN AL, PA
        MOV ESTADO_LUCES, AL
        MOV AX, OFFSET ESTADO_LUCES
        PUSH AX
        CALL LUCES12
        POP AX
        INT 0
END
```

- 6) **F10, impresora con HS y dígitos** ★★ Escribir un programa para imprimir en la impresora un conteo regresivo en base a un dígito ingresado por teclado. Por ejemplo, si el usuario ingresa "3", se debe imprimir "3 2 1 0". El programa comienza leyendo un carácter dígito con la subrutina **LEER_DIGITO**. Luego espera a que el usuario presione F10 para comenzar la impresión, y llama a la subrutina **DESCENDER** que la implementa.

LEER_DIGITO muestra en pantalla el mensaje "INGRESE UN NUMERO DEL 1 AL 9:" y lee un carácter de teclado. Si el carácter ingresado no corresponde al número solicitado, se debe volver a leer un carácter hasta que el usuario ingrese efectivamente un número del 1 al 9.

Imprimin H8J 255 News, cl 1 seg.

milor a a)
 ORG 2000H
 MOV AL, 0
 OUT CONT, AL
 ORG 3000H
 MOV AL, 1
 OUT COMP, AL

Arquitectura de Computadoras

```

RET
ORG 2000H
IN AL, EST
AND AL, 7fh
OUT EST, AL
CALL LEE_DIGITO
CLI
MOV AL, 0FEH
OUT IMR, AL
MOV AL, N_F10
OUT INT0, AL
STI
MOV SEGUIR_ESPERANDO, 1
ESPERA: CMP SEGUIR_ESPERANDO, 1
JZ ESPERA
PUSH DX
CALL DESCENDER
POP DX
INT 0
END
  
```

- 7) **Dispositivo nuevo con PIO** ★★★ Escribir un programa para VonSim que envíe la cadena de caracteres "Hola!" a un dispositivo nuevo, conectado a los 8 bits del puerto PB. Este dispositivo recibe la cadena de a un carácter a la vez, esperando 5 segundos entre cada carácter. Para indicar al dispositivo que finalizó el envío de datos, se debe enviar el código ascii 0 luego de enviar todos los caracteres del string. El programa debe finalizar cuando se han enviado todos los caracteres de la cadena, o cuando se presiona la tecla F10, cancelando el envío cancelando el envío de los caracteres que resten.
- Ejemplo** para enviar la cadena "ASDF": Envío de la "A" → 5 segundos de espera → Envío de la "S" → 5 segundos de espera → Envío de la "D" → 5 segundos de espera → Envío de la "F" → 5 segundos de espera → Envío de 0

```

PB EQU 31h
CB EQU 33h

EOI EQU 20h
IMR EQU 21h
INT0 EQU 24h
INT1 EQU 25h

CONT EQU 10h
COMP EQU 11h

N_CLK EQU 10
N_F10 EQU 15

ORG 40
DW RUT_CLK

ORG 60
DW RUT_F10

ORG 1000H
CADENA DB "Hola!"
FIN CADENA DB ?
  
```