Practica 2 - Interrupciones

9 de octubre de 2020

Interrupciones

- Mecanismo mediante el cual se puede interrumpir el procesamiento normal de la CPU
- Pueden ser de origen interno o externo
- Las interrupciones estan jerarquizadas por importancia
  - Algunas se pueden inhibir (enmascarables)
  - Otras no

Clasificacion

- Las interrupciones las podemos clasificar en
  - Interrupciones por Software
    - ★ Generalmente usadas para hacer llamadas a servicios del Sistema Operativo o monitor
    - ★ Tienen una instrucción específica para invocar la interrupción
    - ★ En el MSX88 usamos las interrupciones para pedirle servicios al programa monitor
  - Interrupciones por Hardware
    - ★ Pedidos de Interrupcion: Son generadas por dispositivos de Entrada / Salida
    - ★ Traps: Interrupciones creadas por el procesador en respuesta a ciertos eventos (ej. división por cero)

Vector de Interrupción

- El proceso de atención de las interrupciones en el simulador es vectorizado.
- Las interrupciones pueden ser de 256 tipos distintos (un byte): 0 a 255
- La tabla de vectores de interrupción es el nexo entre el tipo de interrupción y el procedimiento designado para atenderlo (manejador de interrupciones)
- Esta tabla esta alojada en la posición de memoria 0000h
- La tabla posee 256 entradas (una por cada tipo)
- Cada entrada ocupa 4 bytes:
  - 2 bytes para la dirección donde esta el manejador de ese tipo de interrupción
  - 2 bytes siempre en 00h (por compatibilidad)

Interrupciones por Software

- Para pedir una interrupción usamos la instrucción *INT n*. Donde n es el número de interrupcion a invocar.
- El simulador MSX88 nos provee 4 interrupciones por Software
  - *Tipo 0*: Finalizar el programa actual
  - *Tipo 3*: Hacer puntos de parada en nuestro programa
  - *Tipo 6*: Lectura de un Caracter
  - *Tipo 7*: Escribe en la pantalla de comandos un bloque de datos
- El resto de las interrupciones estan libres para ser usadas

Interrupcion por Software Tipo 0

- Con el llamado a esta interrupción damos por finalizado nuestro programa.
- Ahora disponemos de dos maneras de terminar un programa: *HLT* y esta interrupción. ¿Qué diferencia hay?
- No necesita ningun parámetro especial para usarlo

Ejemplo

```
ORG 2000H
MOV AX, 2515h
INT 0
```

Interrupcion por Software Tipo 3

- Con el llamado a esta interrupción podemos detener la ejecución del programa
- El simulador detiene la ejecucion del programa y nos da el control para ejeuctar otros comandos
- Podemos ver que valores tienen los registros, o la memoria.
- Con el comando *g* podemos continuar la ejecución
- Esto puede ser util para encontrar errores o ver detalles de la ejecución

Ejemplo

```
...
MOV AL, NUM1
MOV BL, NUM2
INT 3  ; detengo aca y puedo ver que esta pasando
...
```

Interrupcion por Software Tipo 6

- Espera para leer un carácter del teclado y lo almacena en la posición de memoria indicada por el registro BX
- Antes de llamar a la interrupción debemos cargar en BX la dirección de memoria donde queremos que se guarde el carácter

Ejemplo

```
ORG 1000H
CAR DB ?
...
ORG 2000H
MOV BX, OFFSET CAR
INT 6
INT 0
```

Interrupcion por Software Tipo 7

- Escribe en la pantalla de comandos un bloque de datos. La dirección de comienzo del bloque se deberá cargar en el registro BX y el número de datos que componen el bloque en el registro AL.

Ejemplo

```
ORG 1000H
SALUDO DB 'HOLA'
FIN_SALUDO DB ?
...
ORG 2000H
MOV BX, OFFSET SALUDO
MOV AL, OFFSET FIN_SALUDO - OFFSET SALUDO
INT 7
INT 0
```

Ejemplo

Veamos un ejemplo en el simulador

Entrada Salida

- Para probar las interrupciones por Hardware del simulador necesitamos entender que dispositivos tiene disponible el MSX88 y como acceder a ellos.
- MSX88 utiliza un espacio de direcciones diferente para el acceso a la Entrada/Salida que para acceder a la memoria.
- Vamos a necesitar aprender más instrucciones para poder acceder a los registros de los diferentes dispositivos.
- Los dispositivos exponen registros en direcciones especificas que nos permiten configurarlos y obtener información de los mismos

Entrada Salida - Continuación

- Los registros de los dispositivos están disponibles para que la CPU los pueda leer o escribir
- Y esos registros tienen direcciones para referenciarlos, pero no son direcciones de memoria.
- Por ese motivo, para comunicarse con los dispositivos de entrada / salida en esta arquitectura debemos usar instrucciones nuevas.
- Las direcciones de memoria y de entrada salida no tienen relación entre sí.
- Las direcciones de entrada salida las solemos llamar puertos de entrada salida

Instrucciones para Entrada / Salida

- Para leer el valor de un puerto, usamos la instrucción *IN*
  - Recibe 2 parámetros. El primero es donde vamos a guardar el valor leido y el segundo la dirección
  - Ejemplo: *IN AL, DX*
- Para escribir un valor en un puerto, usamos la instrucción *OUT*
  - Recibe 2 parámetros. El primero es la dirección donde se va a escribir el valor y el segundo el valor a escribir
  - Ejemplo: *OUT DX, AL*

Interrupciones por Hardware

- El simulador utiliza un Controlador de Interrupciones (conectado a la Entrada/Salida) para manejar las interrupciones
- Para comunicarnos con el PIC y otros dispositivos vamos a usar las instrucciones de entrada / salida
- También vamos a necesitar más instrucciones para controlar las interrupciones

Instrucciones para manejo de Interrupciones

- El simulador puede desactivar la atención de interrupciones enmascarables
- Normalmente esto se utiliza mientras estamos configurando los dispositivos
- Es raro mantener las interrupciones desactivadas por mucho tiempo
- Se utilizan 2 instrucciones para activar o desactivar las interrupciones enmascarables
  - *STI* - Activamos las interrupciones enmascarables
  - *CLI* - Desactivamos las interrupciones enmascarables

Instrucciones para manejo de Interrupciones

- Si configuramos un dispositivo para que nos genere una interrupción es porque estamos interesados en hacer algo cuando ocurre el evento.
- Ejemplo: Si queremos tener un contador, podemos usar el dispositivo *TIMER* para mantener la cuenta
- Se llama manejador de interrupciones a esta subrutina.
- La dirección de inicio de esta subrutina se carga en el vector de interrupción correspondiente
- Hay una instrucción especial para volver desde el manejador de interrupciones. Esta instrucción es *IRET*

Controlador de Interrupciones

- El encargado de administrar las interrupciones es el Controlador Programable de Interrupciones (PIC)
- Los dispositivos que pueden interrumpir al procesador se conectan directamente al PIC, el PIC se conecta a la CPU.
- El PIC puede manejar 8 peticiones de interrupción independentes, numeradas de 0 a 7 (INT0 a INT7)
- El PIC ademas de la señal de interrupción, coloca en el bus de datos un identificador de la interrupción (vector)
- Se atiende una interrupción a la vez. El número de interrupción es la prioridad, teniendo prioridad los números más bajos.
- El manejador de la interrupción llamado deberá indicarle al PIC cuando termina de procesar la interrupción.
- Hasta que no se termine de atender a esta interrupción no se podrá atender a la siguiente interrupción.

Controlador de Interrupciones

- El PIC tiene conectados los siguientes dispositivos a sus lineas de petición de interrupción:
  - *INT0* Conectado a la tecla F-10
  - *INT1* TIMER, Contador de Eventos
  - *INT2* HANDSHAKE, USART (lo vemos mas adelante)
  - *INT3* Controlador de DMA, USART (lo vemos mas adelante)

Registros del Controlador de Interrupciones

- El PIC posee varios registros donde podemos configurar u obtener su estado (con las instrucciones OUT e IN)
- EL PIC está conectado al bus de Entrada Salida a partir de la dirección 20h.
- Tiene 12 registros en total. Los registros se encuentran ubicados en forma consecutiva a partir de la dirección base (20h).
- Todos los registros son de 8 bits.

Registros del Controlador de Interrupciones

- *EOI* (20h) Fin de Interrupción. Solo para escribir, se debe mandar el comando de fin de interrupción cuando se termino de servir la interrupción. (Valor 20h)
- *IMR* (21h) Registro de Mascara de Interrupciones. Permite enmascarar selectivamente las interrupciones que va a recibir el PIC. Cada bit de este registro representa una linea de interrupción. Si el valor es puesto en 1, esa interrupción estará inhibida.
- *IRR* (22h) Registro de Peticion de Interrupciones. Almacena las interrupciones pendientes. Cada bit de este registro representa una linea de interrupción. Cuando la interrupción se satisface, el bit de dicha línea se pone a cero.

Registros del Controlador de Interrupciones

- *ISR* (23h) Registro de Interrupción en Servicio. El bit que representa la línea de interrupción que se está atendiendo estará en 1. El resto en 0.
- *INT0..7* (24h-2Bh) Cada uno de estos registros contiene le valor del *vector de interrupción* correspondiente a la entrada del mismo nombre.

¿Como programamos el PIC?

- Primero desactivamos las interrupciones
- Configuramos el registro *IMR* para tener activas solo las interrupciones que nos interesen
- Configuramos el o los vectores de interrupcion elegidos en los registros INT0 a INT7 (según corresponda)
- Por último, volvemos a activar las interrupciones

¿Como programo el manejador de interrupción?

- Tenemos que poner la dirección del manejador de interrupciones en el vector correspondiente.
- Para saber la dirección multiplicamos por 4 al vector de interrupción (recordar que cada entrada ocupa 4 bytes).
- El resultado es la posición de memoria dentro de la tabla de vectores de interrupción.
- En esa posición de memoria debo guardar el *offset* de la subrutina que va a manejar la interrupción

Ejemplo

```
ORG 40          ; Desplazamiento 40
                ; Vector de Interrupción 10
IP_F10      DW RUTINA_F10
```

¿Como programo el manejador de interrupción?

El controlador de interrupciones debe tener las siguientes condiciones

- Debe terminar con la instrucción *IRET*.
- Se debe enviar el comando de fin de interrupción al PIC para que pueda
- Normalmente no deberíamos cambiar ningún registro, ya que estamos interrumpiendo al programa en cualquier momento.

¿Como programo el manejador de interrupción?

Ejemplo

```
ORG 3000H
; Manejador de F-10
RUTINA_F10: INC  CONTADOR
            PUSH AX
            MOV AL, 20h
            OUT 20h, AL
            POP AX
            IRET
```

Tecla F-10

- Es la interrupción mas simple que tiene el simulador
- La interrupción se genera cada vez que apretamos la tecla F-10.
- Ver el ejemplo en el simulador

Timer

- Es un dispositivo contador de eventos.
- Posee dos registros (de 8 bits)
  - *CONT*: Registro contador, que se incrementa cada 1 segundo
  - *COMP*: Registro de comparación, cuando *COMP* y *CONT* son iguales, se genera una interrupción.
- El registro *CONT* está en la dirección 10h y el registro *COMP* en la dirección 11h
- El registro *CONT* no vuelve a cero cuando se genera una interrupción.

Timer

Para utilizar el TIMER:

- Habilitar la interrupción en el PIC
- Configurar el registro *COMP* para que nos llame en el intervalo deseado
- Poner a cero el registro *CONT*
- Programar el manejador de interrupción.