DIGITAL III:

MODULO 4: Interrupciones

Conceptos básicos:

Interrupción:
Una interrupción es un procedimiento desencadenado por un evento externo que produce una trasferencia impredecible del control de ejecución, desde la rutina que es interrumpida hacia una nueva rutina (denominada de atención de la interrupción).
Una interrupción se compone de 4 etapas: solicitud, reconocimiento, atención y retorno.

Solicitud de interrupción:
El proceso de interrupción comienza por una solicitud de interrupción que se cursa al micro a través de una señal asincrónica de entrada.

Reconocimiento:
Si están dadas ciertas condiciones (temporales y lógicas), el micro, reconoce el pedido y comienza con la atención de la interrupción. Por tanto, la transferencia del control de ejecución no es inmediata ni (en general) incondicional.

Atención de interrupción:
Es el proceso por el cual se ejecuta la rutina de atención.

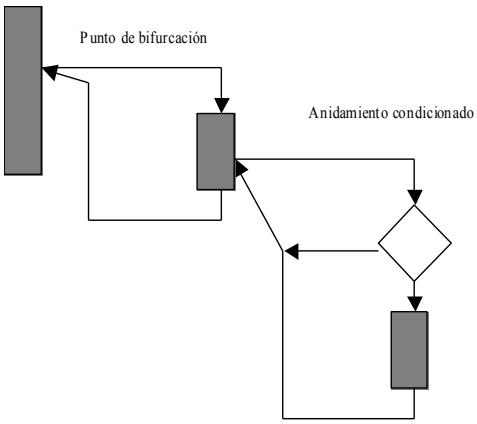
Retorno:
Una vez finalizada la ejecución de la rutina de atención, se produce el retorno del control de ejecución a la rutina que fue interrumpida, permitiendo al micro continuar con la ejecución de la misma. Esto se implementa a través de una instrucción especial, que se incluye en la rutina de atención.

Máscara de interrupción:
Una máscara de interrupción es un bit que actúa como condicionador para pasar de la etapa de solicitud al reconocimiento de la interrupción. La existencia o no de este bit, separa las interrupciones en dos grupos: enmascarables y no enmascarables.
En el 8088 existe un bit interno del micro (IF: interrupt enable flag bit) que se ubica en el Flag Register. Ese bit actúa como máscara de la interrupciones enmascarables. Si la máscara se encuentra activada (IF=0) la solicitud proveniente de una línea de solicitud, es ignorada. Si por el contrario IF=1, el pedido es aceptado y se dará inicio a el ciclo de reconocimiento.

Rutina de atención:
También denominada rutina de servicio, es el programa que se ejecuta como consecuencia de la atención de una interrupción. Este programa debe satisfacer el requerimiento que produce la interrupción y eliminar el nivel activo en la señal que ha producido la solicitud (si todavía permanece en el momento de su ejecución).

Anidamiento entre rutinas:
Dos rutinas se ejecutan en forma anidada si, durante la ejecución de una de ellas, hay una bifurcación en el flujo de ejecución (prevista o no, condicionada o no) hacia la otra

que (al finalizar su ejecución) retorna al punto de donde partió en la primera. Un ejemplo típico de anidamiento entre rutinas se da en la ejecución de subrutinas.



Anidamiento de interrupciones:
Una interrupción se anida en otra, cuando la primera interrumpe la ejecución de la rutina de atención de la segunda.
Esto puede evitarse si se cuenta con algún medio para evitar la atención de interrupciones, durante la ejecución de rutinas de atención de otras.

Sistemas de atención de dispositivos periféricos:
Para asignar tiempo de ejecución a las rutinas relacionadas a los dispositivos externos conectados a el micro, existen dos métodos:

- Método de Encuesta: (Polling)

El microprocesador pregunta de a un periférico por vez, si requiere atención. Si la necesita, ejecuta las acciones que le son solicitadas. Si no, continúa preguntando al siguiente dispositivo.
Cuando termina de ejecutar las tareas solicitadas (rutina de atención), vuelve a la rutina de encuesta y continua con otro periférico.
Estas "encuestas" se ejecutan dentro del programa principal y se implementan con saltos condicionados.
En este sistema, mientras el micro atiende a un dispositivo, ningún otro puede ser atendido, independientemente de la importancia de la atención de cada pedido.
Esta estructura no permite una buena implementación del anidamiento y de los esquemas de prioridades. Además consume los recursos de ejecución de forma ineficiente y resta tiempo para la ejecución de la rutina principal.
Tiene la ventaja de no requerir hardware adicional ya que todo se resuelve por soft.

- Método de Interrupciones:

Este método logra aprovechar los recursos del micro de manera más eficiente, ya que estos se utilizan para la ejecución del main y sólo se invierten en la atención de periféricos, cuando estos lo solicitan expresamente.
Este método se implementa a través de señales asincrónicas, que los dispositivos envían al microprocesador para comunicarle que necesitan atención. Cuando el micro las recibe, (en un momento totalmente impredecible), finaliza la ejecución de la instrucción en curso y comienza la ejecución de una rutina específica (rutina de atención de la interrupción en cuestión). Cuando finaliza esta rutina, el micro continúa con la ejecución de la instrucción siguiente a la última que ejecutó, cerrándose el anidamiento entre las rutinas.

Una interrupción es, entonces, un recurso de hard y soft que mejora la performance de ejecución del micro.

Nota: Es muy frecuente encontrar aplicaciones en donde las ventajas y desventajas aquí descriptas no se reflejen de esta manera o que los dos métodos mencionados se apliquen en forma mixta.

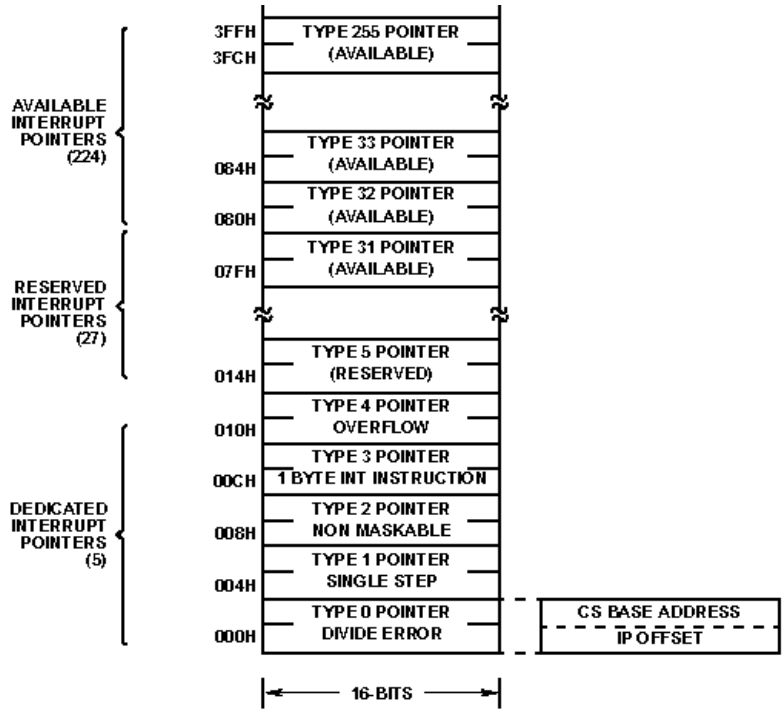
"Tipo" de interrupción:

En un sistema de micro cómputo, suele haber más de un dispositivo que solicita interrupciones al micro. Para que el micro pueda individualizar el dispositivo que ha solicitado la interrupción, podrían implementarse varias entradas de interrupción a la que conectaríamos un solo periférico. De esta manera no habría inconveniente en individualizar la fuente del pedido usando recursos de hardware.
Sin embargo este esquema sería muy caro y se agotaría fácilmente.
En el 8088, en cambio, hay una sola entrada de interrupción (por ahora ignoremos la NMI). Por tanto, una vez que el micro ha reconocido que se le pide una interrupción, debe también individualizar cuál es la fuente que la ha solicitado.
Para ello, entra en un ciclo especial (ciclo de reconocimiento de interrupción o de INTA) en donde, lee por el bus de datos un byte que entiende como el número de "Tipo" de la interrupción que ha sido solicitada. En el 8088, hay 256 "tipos de interrupciones posibles".

Este número actúa como puntero dentro de la IVT (Interrupt Vector Table) para determinar donde se encuentra la rutina de atención de ese "tipo". Si multiplicamos el "tipo" de interrupción por cuatro, obtenemos el desplazamiento dentro de la IVT, del primer byte que compone el vector.
Por ejemplo, a la interrupción del tipo 5, le corresponde el 6to vector de interrupciones cuyos 4 bytes se ubican el las direcciones 00014 H a la 00017 H.
En la 14 y la 15 se encuentra la IP y en la 16 y la 17 el CS.

Tabla de Vectores de Interrupción (IVT):

Es un sector del mapa de memoria que se encuentra en los primeros 1024 bytes del mapa. Va desde la dirección: 00000 H a la 003FF H.
Contiene 256 vectores de interrupción, de 4 bytes cada uno.



En la zona del mapa en donde se ubica la IVT, suele mapearse memoria RAM. Esto permite que la tabla sea re programada durante la ejecución, dándole gran flexibilidad al sistema.

En la mayoría de los sistemas, serán necesarias muchas menos que 256 interrupciones. En esos casos no sería necesario mapear memoria en toda la tabla. Sin embargo, es recomendable que, al menos durante la etapa de depuración, se asignen direcciones válidas a toda la IVT, de manera que las 256 interrupciones tengan una rutina a donde transferir el control.
A los "tipos" que no estén usados, se les carga un vector que deriva a una rutina que indique que se ha producido una interrupción no definida.

Vector de Interrupción:

Contiene la dirección de (segmento y desplazamiento) de inicio de la rutina de atención de la interrupción del "tipo" al que corresponde. Cada vector está compuesto por dos bytes para alojar el IP y dos para el CS.
El orden de almacenamiento de los datos, en sentido ascendente de las direcciones, es:
LSB – IP, MSB – IP, LSB – CS, MSB – CS.
Componiendo estos bytes, se genera la dirección de memoria en donde se encuentra la rutina de atención asociada al "tipo" de interrupción a la que corresponde dicho vector.

Clasificación de interrupciones.

Este es un ítem muy discutido entre los distintos autores.

Una clasificación posible es:

- Interrupciones predefinidas (0-31): que se usan para fines específicos
- Interrupciones definidas por el usuario (32-255): que no tienen función definida. Pueden ser de hard y de soft.

Las interrupciones predefinidas pueden ser disparadas por interrupciones definidas por usuario, tanto de hard como de soft.

Otra clasificación, la da Intel, que divide los 256 tipos de interrupciones en tres grupos (ver figura de la IVT):

- Dedicadas (0-4): que tienen una "fuente" preestablecida.
- Reservadas (5-31): que no recomiendan usar para futuras ampliaciones del esquema (en otros micros superiores, en esto "tipos" hay otras dedicadas).
- Disponibles (32-255): que el usuario puede designar a las fuentes que desee.

Nosotros consideraremos interrupciones, solo a las de hardware, independientemente del "tipo" que ejecuten. Dentro de estas ubicaremos a las enmascarables (cuyo "tipo" lo define el usuario) y a la no enmascarable (cuyo "tipo" está predefinido).

Las demás serán consideradas "Seudo interrupciones", ya que no cumplen con algún aspecto de la definición inicial de interrupción. Dentro de ellas ubicaremos las interrupciones de software, las de error y la de trampa (o Single Step).

Interrupciones de hardware:

Son las interrupciones propiamente dichas. Hay dos tipos.

Interrupción enmascarable:

La solicitud de interrupción enmascarable se realiza a través de la entrada INTR (Interrupt Request). INTR solicita una interrupción de "tipo" variable que se determina en el ciclo de reconocimiento.
INTR, es una entrada activa por nivel alto, que se muestrea durante el último ciclo de reloj de cada instrucción (hay instrucciones especiales en que esto no se cumple, ver nota). En ese momento se determina si el micro debe continuar ejecutando la próxima instrucción o debe ingresar en el ciclo de reconocimiento de interrupción.
Para que el pedido sea reconocido, INTR debe permanecer en alto (al menos) durante el último ciclo de reloj de la ejecución de la instrucción en curso. Luego INTR puede bajar en cualquier momento después del flanco descendente del primer pulso de INTA, dentro de la secuencia de reconocimiento.
Esta línea puede "enmascararse" a través de un bit interno del micro (IF: interrupt enable flag bit) que se ubica en el Flag Register. Si la máscara se encuentra activada (IF=0) la línea es ignorada. Si por el contrario IF=1, el pedido es aceptado y se dará inicio al ciclo de INTA.
La máscara permite condicionar la atención de las solicitudes de interrupción que ingresan a través de INTR. Esto es fundamental para permitir o no el anidamiento de rutinas de atención de interrupciones de esta clase.

Es importante ver que, cuando comienza la ejecución de la rutina de atención de la interrupción, IF debe pasar a 0 al menos hasta que la línea INTR pase a bajo. Si esto no fuera así, solo llegaría a ejecutarse la primera instrucción de esta rutina y el micro comenzaría nuevamente con otro ciclo de reconocimiento.

Conceptualmente: las interrupciones activas por nivel deben ser enmascarables.

Nota: Una excepción es la instrucción WAIT que espera un nivel bajo en la entrada TEST para continuar con el programa. Esta instrucción muestrea continuamente la entrada de interrupciones "durante" su ejecución, permitiendo la atención de interrupciones en el interior de la "espera". Cuando finaliza la rutina de atención, el micro retorna a ejecutar la instrucción WAIT.

Interrupción no enmascarable

La solicitud de interrupción no enmascarable se realiza a través de la entrada NMI (Non maskable Interrupt). NMI solicita una interrupción de "tipo" fijo igual a 2, por lo tanto no es necesario que se realice un ciclo de INTA.
NMI es una entrada asincrónica activa por flanco ascendente. El flanco ascendente en la entrada de NMI, se lachea en el interior del micro hasta que la instrucción en curso finaliza.
No es enmascarable por lo que su atención es inevitable. Esto hace que, independiente de la estructura que se monte para las interrupciones enmascarables, puede pensarse que, NMI es siempre la interrupción más prioritaria ya que puede interrumpir la ejecución de cualquier servicio generado por INTR.
Para que el pedido se haga efectivo, la entrada debe permanecer en alto, al menos dos ciclos de reloj, hasta que el micro la reconozca. La línea puede bajar antes, durante o después del servicio de NMI.
Para que el micro vuelva a reconocer otro pedido, la línea debe haber permanecido en bajo al menos dos ciclos de reloj después de haber bajado.
Debe tenerse especial cuidado en que no haya flancos espurios en el circuito de NMI que puedan generar interrupciones indeseables.
NMI, se usa para situaciones graves que requieran atención incondicional del micro. Un uso típico de NMI, es la ejecución de la rutina para caídas de alimentación del sistema o sistemas de watch - dog.

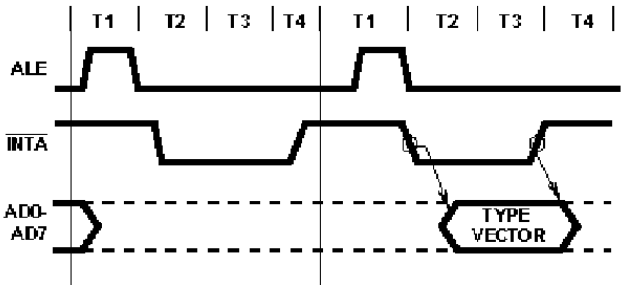
Es importante ver que, como NMI es no enmascarable, no podría ser activa por nivel, ya que sería imposible impedir que se repita el comienzo de la atención cuando se ejecuta la primera instrucción de la rutina de servicio.

Conceptualmente: las interrupciones no enmascarables deben ser activas por flanco.

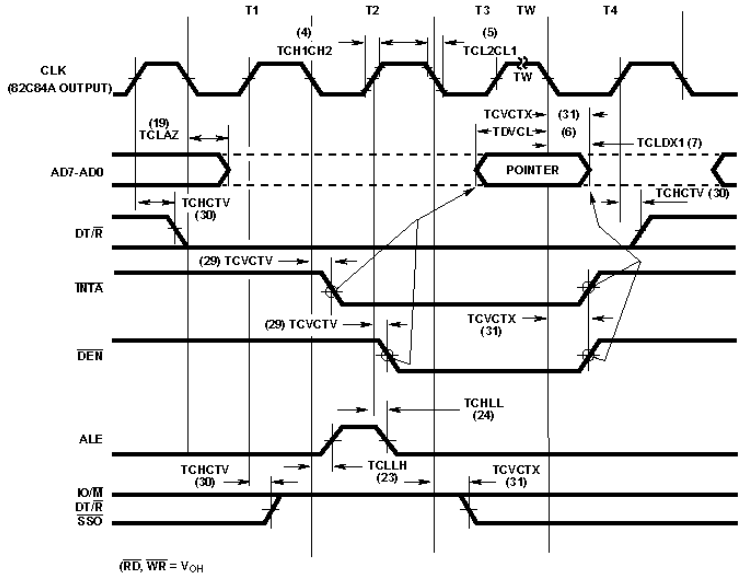
Ciclo de reconocimiento de interrupción:

Este ciclo, también llamado ciclo de INTA, se utiliza para detectar el "tipo" de interrupción que desea ejecutarse como resultado de la solicitud que ingresa por INTR. Esta secuencia de reconocimiento se genera solo como respuesta a una interrupción proveniente de INTR.
En realidad el ciclo de reconocimiento está formado por dos ciclos de INTA, cada uno de ellos es muy similar a un ciclo de lectura. En el segundo ciclo de INTA, se lee un byte del bus de datos que identifica el "tipo" de la interrupción.

Por tanto, INTA (Interrupt Aknowledge) es la salida de reconocimiento de interrupción y se usa como pulso de lectura del "tipo" en el ciclo de INTA. Es activa en bajo y pasa a este valor durante T2, T3 (y Tw) de los ciclos de reconocimiento de interrupción. Su objetivo es avisar al dispositivo solicitante que su pedido de interrupción ha sido aceptado y que debe notificar al micro cuál es el "tipo" de interrupción que pide.



Segundo ciclo de INTA detallado:



Nota: Durante el ambos ciclos de INTA, el bus AD está flotante.

Atención de Interrupción:

Previo al comienzo de la ejecución de la rutina de servicio de interrupción, y como parte de la respuesta a cualquier interrupción (o pseudo interrupción) el micro realiza una serie de tareas que forman parte de la secuencia de ejecución de una interrupción. En los casos en donde exista (INTR), la secuencia de ejecución se realiza luego de la de reconocimiento.
Las acciones que forman esta secuencia son:

- PUSH del registro de banderas, el CS y el IP en la pila.
- Reset de los bits de trampa y de máscara para que , al menos por default, la rutina de ejecución de interrupción no sea ejecutada en modo paso a paso, ni sea interrumpida por interrupciones enmascarables (ver nota).
- Carga de los nuevos CS e IP (provenientes de la IVT).

Luego de estas acciones, se ejecuta la primera instrucción de la rutina de atención.

Nota: Estos dos bits pueden se re habilitados en el interior de la rutina de servicio de la interrupción, permitiendo la ejecución paso a paso de esa rutina (ojo no se puede hacer en la ejecución de la rutina de atención de paso a paso) o permitir que otra

interrupción enmascarable sea atendida durante la ejecución de una interrupción dada (anidamiento).

En resumen:
Cada vez que el micro finaliza la ejecución de una instrucción, comprueba (en este orden):

- ✓ Si está activa la trampa (TF=1)
- ✓ Si hubo pedido NMI
- ✓ Si hay overflow en el segmento del coprocesador
- ✓ Si hay INTR
- ✓ Si la próxima es una instrucción INT

Si se da alguna de estas condiciones, entonces se produce la siguiente secuencia:

- ✓ Si hay un INTR, se realiza la secuencia de reconocimiento de interrupción.
- ✓ Se salva el contenido del registro de banderas en pila
- ✓ Se desactivan la máscara de interrupción IF y el TF
- ✓ Se salva el contenido del CS en la pila
- ✓ Se salva el contenido del IP en la pila
- ✓ Se recupera el contenido del vector de interrupción y se carga el CS y el IP en con su contenido de manera que el micro continúe ejecutando donde ellos indiquen.

Tanto en la INTR como en la NMI, los tiempos de demora entre la activación del pedido y la ejecución de la primera instrucción de la rutina de servicio, dependen de la instrucción que se esté ejecutando en el momento de esa activación.
Los tiempos que si pueden determinarse en forma exacta son los de demora entre el final de la instrucción en curso y el comienzo de la primera instrucción de la rutina de servicio.

Por ejemplo, en el caso de una INTR, el retardo desde el final de la instrucción durante la cual ocurrió la interrupción, hasta el comienzo de la ejecución de la primera instrucción de la rutina de atención, es de 61 ciclos de reloj.
En las interrupciones generadas por soft no hay ciclos de INTA. Esto reduce el retardo a 51 ciclos de reloj (para INT nn y Single Step); 52 para INT3 y 53 para INTO.

Instrucciones relacionadas

- **STI:** setea el bit IF, permitiendo el acceso de interrupción INTR.

- **CLI:** resetea el bit IF, impidiendo el acceso de interrupción INTR.

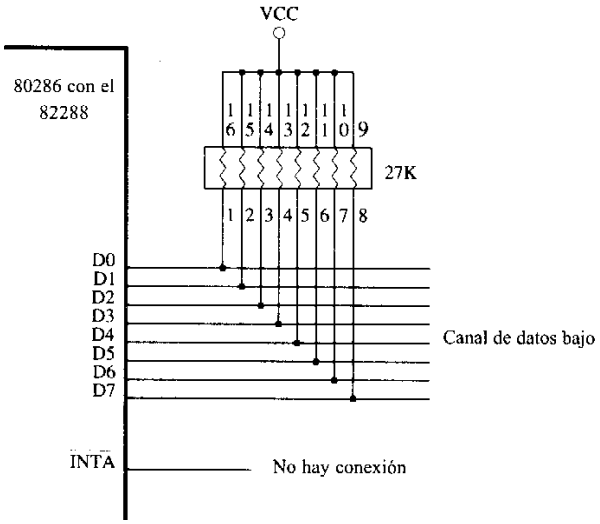
Nota: Para activar y desactivar el bit TF no hay instrucciones específicas, para hacerlo se usa una rutina especial.

- **IRET:** Interrupt return

Es la instrucción que finaliza las rutinas de atención de interrupciones, indicando al micro que debe recuperar el estado anterior y continuar ejecutando la siguiente instrucción previa al comienzo de la rutina de atención.
El procedimiento que se ejecuta, es el complementario a la secuencia mencionada en el item "Secuencia de la ejecución de una interrupción". Es decir:

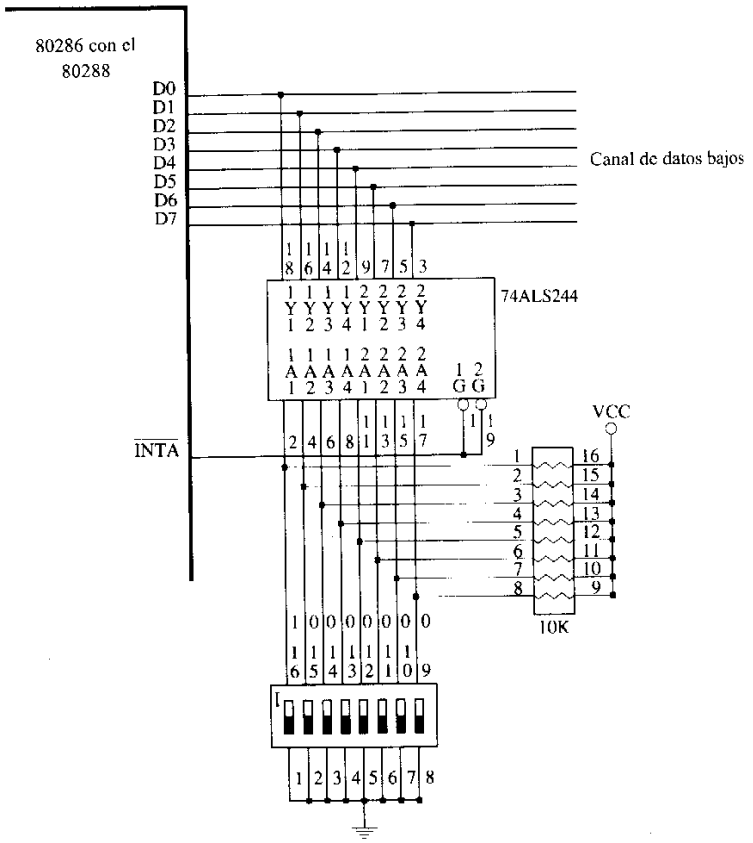
Se recuperan los dos bytes del IP de la pila
Se recuperan los dos bytes del CS de la pila
Se recuperan los dos bytes del registro de banderas
Al recuperar el IP el CS de la pila, luego de ejecutar IRET, el micro continúa ejecutando la instrucción que se encuentra luego de la última instrucción ejecutada antes de entrar en la rutina de atención de interrupción. Al recuperar el registro de banderas, se repone el estado anterior de IF y TF.

Circuito mínimo para generar una instrucción del tipo FF H (Tipo = 255):



Durante el ciclo de reconocimiento de interrupción, el bus de datos espera recibir el byte indicador del "tipo". En el momento en que el bus queda flotante, esperando recibir un dato, las resistencias de pull up, fuerzan las líneas a alto, imponiendo de esa manera el dato FF H, que será interpretado como que el pedido es del tipo = 255.
Si en el sistema hay una sola fuente de interrupciones, esta es la forma más simple de resolver el hardware necesario para su atención.

Circuito para generar una interrupción de "tipo" configurable por hardware:



Cuando línea INTA pulsa a bajo, el dato de entrada del latch pasa a la salida, entregando el tipo de interrupción a atender.
El dato se puede modificar a través de switches que ponen a masa o dejan flotantes las líneas. Las líneas que quedan flotantes son tomadas como altos, por la presencia de las resistencias de pull up.

Para generar una interrupción de "tipo" configurable por software, podía mapearse un registro al que se le pudiese grabar el dato del "tipo" para que lo entregue durante el ciclo de reconocimiento.

Pseudo Interrupciones

Interrupciones de software

Son instrucciones que provocan que el micro ejecute alguna de las rutinas de atención de interrupciones del "tipo" que estas le indiquen. Finalizada la ejecución de la rutina de atención, el micro continúa ejecutando la instrucción posterior a la instrucción de interrupción.
No son interrupciones propiamente dichas ya que al ser instrucciones, no son impredecibles.
Como son instrucciones, no respetan ningún esquema de prioridades y se anidan siempre que se incluyan dentro de una rutina de atención de interrupción.
Obviamente no son enmascarables.
Estas instrucciones pueden usarse para transferir el control a rutinas que son reubicables en memoria (en las que el programa que las llama no conoce la dirección de inicio de las mismas).

También salvan en el stack el registro de banderas.
No ejecutan el ciclo de INTA, pero si inhiben Single Step y IF durante su ejecución.

En el 8088 hay tres:

- INT3: Interrupción de soft de 1 byte de longitud.

La instrucción INT3, provoca una interrupción de Tipo 3.
Esta instrucción ocupa un solo byte ya que en el código de instrucción está implícito el tipo de interrupción que debe ejecutarse.
Se utiliza para insertar puntos de ruptura en rutinas que están siendo depuradas.

- INT n:

Es una instrucción que ejecuta el servicio de la interrupción del Tipo n.
Ocupa dos bytes, uno para el código de operación y otro para el para el dato del "tipo" a ejecutar. Se pude usar para simular cualquier tipo de interrupción y para depuración.

- Interrupción de sobre flujo:

La instrucción INTO interrumpe (tipo 4) el programa si existe un over flow señalado por la activación de la bandera de sobre flujo (OF).
Si OF = 0 la instrucción no ejecuta nada (se comporta como un NOP).
Esta instrucción permite corregir errores en el caso de over flow.

Interrupciones de error:

No son interrupciones propiamente dichas ya se producen por un motivo particular que en general es un error de software. Tampoco respetan ningún esquema de prioridades.
En el 8088 hay solo una:

- Tipo 0: Error al dividir

Ocurre cuando hay un sobre rango en el resultado de una división (como ocurre si se intenta dividir por 0). La dirección de retorno de la rutina de atención de este tipo, no apunta a la siguiente instrucción a ejecutar sino que, la dirección de retorno es igual a la de la instrucción que se terminó de ejecutar antes de empezar la ejecución de la rutina de atención. Esto permite corregir el error que provocó la interrupción en la rutina de atención y volver a empezar.
No es enmascarable.

Interrupciones de Trampa (Modo paso a paso):

Si el bit TF está activo, después de la ejecución de cada instrucción se produce una interrupción tipo 1. Al aceptar esta interrupción, se borra el TF de manera de ejecutar la rutina de atención en forma normal.
Se usa para usar la rutina de atención como programa de monitoreo en depuración de programas.
Comienza a ejecutarse una instrucción después de que se activó el bit TF.
No es enmascarable.

Interrupciones Simultáneas.

Para analizar casos de interrupciones simultáneas, debemos aplicar la siguiente guía:

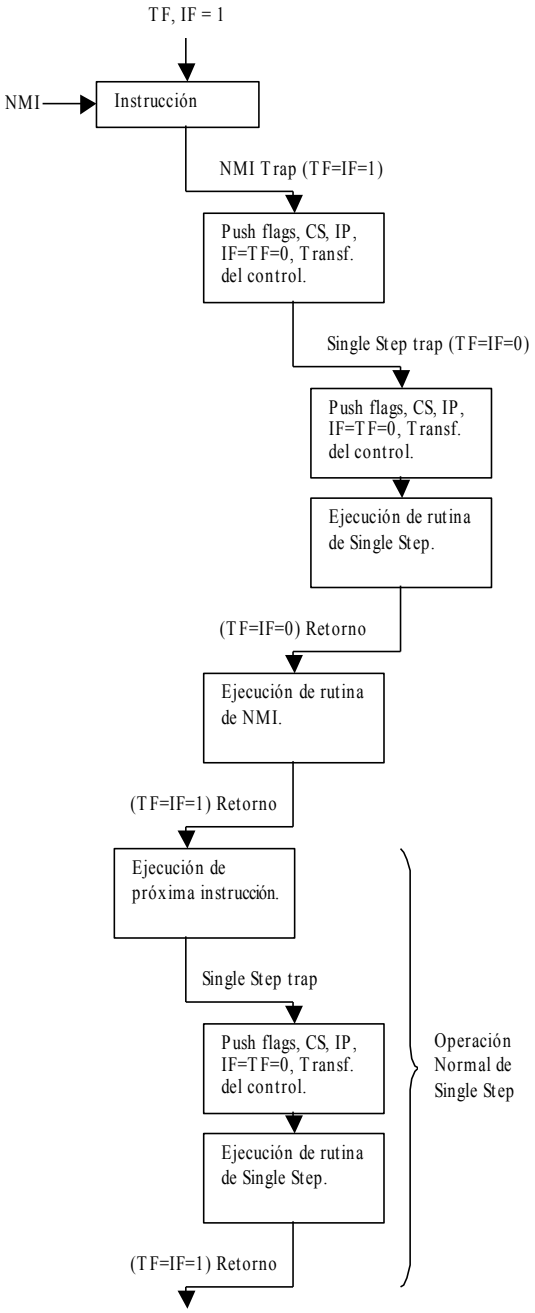
1. INTR es la única interrupción enmascarable y si se detecta en simultáneo con otras, su atención depende del estado de IF. Si las otras interrupciones mantienen (durante su ejecución) la mascara activa, entonces INTR no será atendida. Esto suele interpretarse como que INTR es la menos prioritaria de la interrupciones.
2. De las no enmascarables: En general: Single Step es la de mayor prioridad, luego NMI, y por último las de software. Esto se ve claramente en el orden en que el micro decide que hará después de la ejecución en curso (ver item Secuencia de la ejecución de una interrupción).
3. Una excepción a la regla expresada en 2, es si las tres interrupciones producen pedidos simultáneos. En ese caso, primero se ejecuta la de soft, luego la NMI, sin utilizar para su ejecución el modo paso a paso. Luego de la ejecución de la instrucción siguiente a la interrupción por soft, se ejecutará el single step.

Si se quiere evitar que single step tenga mayor prioridad que INTR, la rutina que está siendo ejecutada con single step, debe enmascarar el ingreso de interrupciones, y se deben habilitar solo durante la ejecución de la rutina de atención de la trampa.

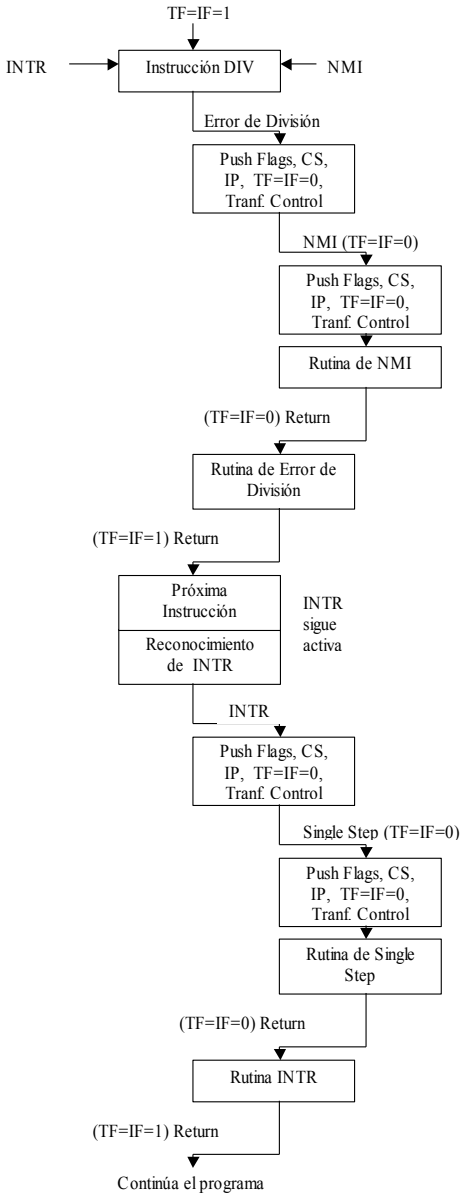
Para prevenir que el single step se atienda con más prioridad que NMI, uno de varios métodos que pueden usarse es siguiente: La rutina de single step debe verificar (en sus primeras instrucciones) el contenido de las últimas palabras almacenadas en el stack. Si esas palabras coinciden con la dirección de retorno de la NMI, significa que el single sep, ha interrumpido a una NMI y en ese caso, debemos culminar inmediatamente la ejecución de single sep, para que la NMI pueda ejecutarse.

Ejemplos:

1- Single Step y NMI ocurren en forma simultánea:



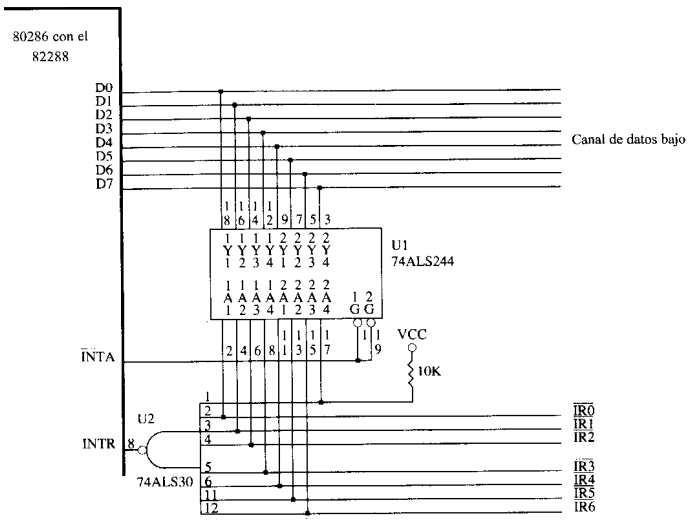
2 - NMI, INTR y un error de division, ocurren durante una instrucción de división con single sep.



Expansión de la estructura de interrupciones

Cuando en un sistema de micro cómputo con 8088, hay varias fuentes de solicitud de interrupciones, debo idear algún método para que el micro reciba los distintos pedidos, reconozca la fuente y ejecute el "tipo" correcto para cada fuente. Para esto hay diversos métodos.

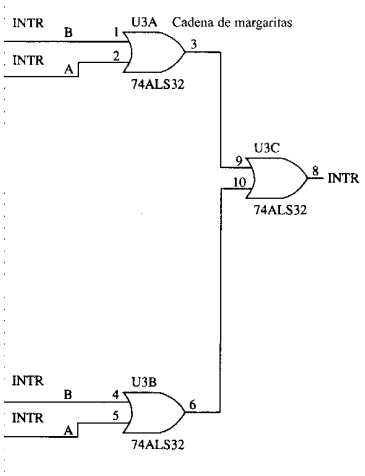
Por medio de un registro
Permite expandir el sistema hasta 6 entradas diferentes de interrupción, con 6 distintos "tipos" asociados.
Este método permite conectar varias fuentes de solicitud a una compuerta, de manera que todas piden a través de la pata INTR. Son las mismas líneas de pedido, las que se usan para determinar la palabra de "tipo".
Si cualquiera de las entradas IR se pone en 0, la salida de la NAND pasa a 1, y pide una interrupción por INTR. El tipo de la interrupción se obtiene dependiendo de los valores de las líneas en el momento de leer el dato de registro.



Este circuito permite, además, asignar distintos "tipos" de interrupción para cuando se suceden dos o más pedidos simultáneos. De esta manera, en la rutina de atención de estos tipos especiales, puedo determinar cuál de los pedidos simultáneos se atenderá primero. Así podemos decir que este circuito permite la asignación de prioridades entre los distintas entradas.

Circuito de "cadena de margaritas":

En este sistema todas las entradas de solicitud de interrupción, están asociadas al mismo "tipo". La rutina de atención deberá determinar cuál de los dispositivos conectados a la cadena, es el que pidió la interrupción.
Este método puede pensarse como una combinación de interrupciones y polling, ya que luego que la interrupción es reconocida, la determinación del dispositivo que pidió atención, se hace por encuesta.



PIC (Priority Interrupt Controller) 82C59A

Es un controlador diseñado para ampliar la estructura de interrupciones del 8088, permitiendo además, implementar una compleja y flexible estructura de prioridades entre las distintas entradas de pedido.

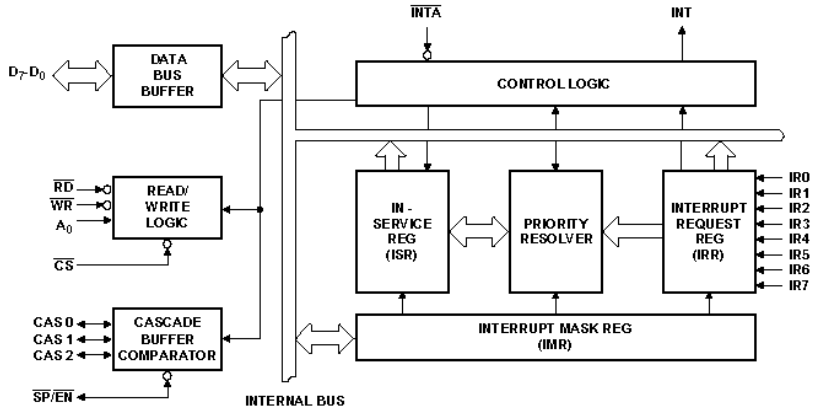
El PIC toma los pedidos de los distintos dispositivos, determina cual de ellos es el más prioritario (considerando también si actualmente hay una rutina en servicio) y finalmente le envía el pedido al micro.

Un 82C59A, puede manejar hasta 8 fuentes de interrupción, de diferentes "tipos". Pero con varios 82C59A, se puede implementar una estructura en cascada que puede manejar hasta 64 interrupciones (con 9 x 82C59A).

La asignación de prioridades para sus 8 entradas, es modificable por software. Para esto, posee diferentes modos de funcionamiento, por lo que el esquema de prioridades, puede cambiarse incluso dentro de una rutina de atención. Esto le da una gran flexibilidad de funcionamiento.

Nota: De todos los modos que tiene el 82c59, sólo veremos aquellos que conservan la estructura de prioridades, es decir el "Fully Nested" y el "Special Fully Nested". Las referencias que se hacen a los otros modos son sólo porque en algunos casos es necesario mencionarlos para una mejor comprensión de aquellos que damos o para tener una idea de las capacidades adicionales del 8259.

Bloques funcionales



- **Interrupt Request Register (IRR):**

Es un registro de 8 bit que indica con un 1, las entradas de IRx (Interrupt Request x) que requieren servicio y que están pendientes de ser reconocidos.
Cuando un determinado pedido es reconocido el bit correspondiente del IRR, es reseteado. Si siempre trabajamos con modos que respeten la estructura de prioridades, el bit que será reconocido siempre será el más prioritario de los que estén en 1.
Este registro no se afecta por el IMR.

- **In-Service Register (ISR):**

Es un registro de 8 bit que almacena las interrupciones que están siendo atendidas.
Este registro es actualizado cuando llega un EOI.

- **Priority Resolver**

Es un circuito lógico que determina las prioridades a partir de los bits del registro IRR.
En los ciclos de INTA, este bloque selecciona, el pedido de mayor prioridad y lo pasa al ISR.

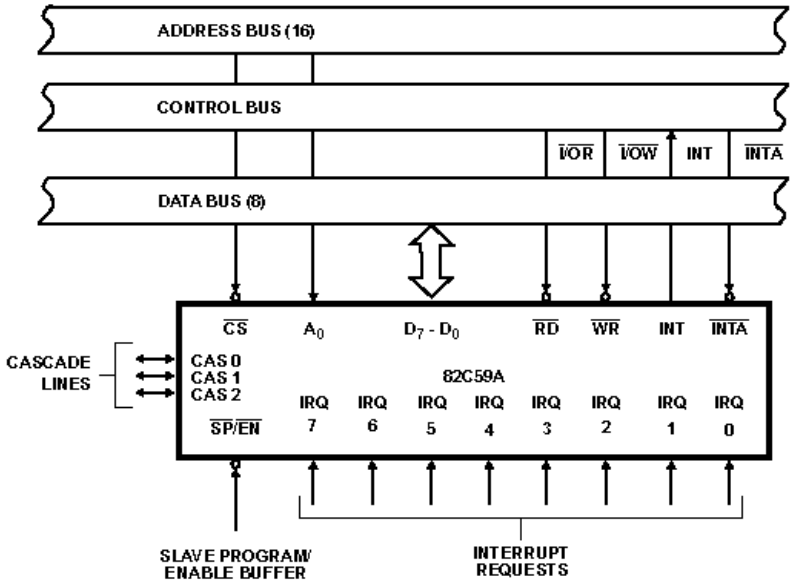
- **Interrupt Mask Register (IMR)**

Es un registro de 8 bit, que almacena los bits de máscaras de las líneas de entrada de solicitud IR. Opera en la salida del IRR.
Al activar la máscara de una línea en particular, no serán considerados los pedidos de esa entrada, independientemente de la estructura de prioridades.
Una interrupción enmasacarada, no afectará la atención de otras de menor prioridad que ella.

- **The Cascade Buffer/Comparator**

Este bloque, funciona cuando se conectan varios en cascada. Su función es la de almacenar y comparar las identificaciones de todos los 82C59A, usados en el sistema.
CAS0, CAS1 y CAS2 se comportan como salidas cuando el 82C59A se configura como el "maestro" de un sistema en cascada. Por el contrario, son entradas cuando es "esclavo".
El 82C59A "maestro" envía, a través de las líneas CAS, la identificación del "esclavo" que será atendido.

Conexión a el sistema con 8088:



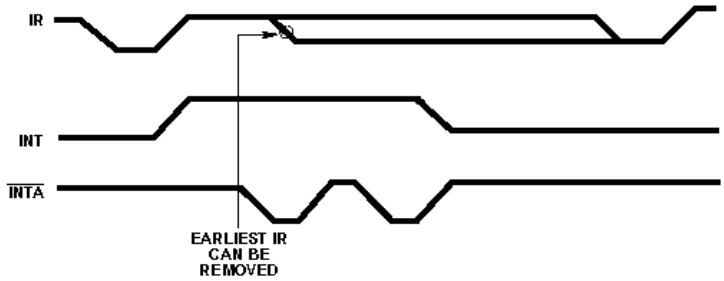
Secuencia de funcionamiento:

1. Una o más entradas de IR0 a IR7 se activan pasando a alto, seteando los respectivos bits del IRR.
2. El 82C59A evalúa los pedidos entrantes y las interrupciones que están siendo atendidas, dentro del "Priority Resolver" comparando los datos del IRR y el ISR.
3. Si la interrupción de mayor prioridad que solicita servicio a través del IRR, tiene mayor prioridad que la de mayor prioridad de las que está siendo atendida, el 8259 pide una interrupción al micro a través de INT. Caso contrario, almacena el pedido hasta que llegue el próximo EOI. Si se ha cursado el pedido la secuencia continúa así:
4. Si el micro tiene habilitadas las interrupciones enmascarables, reconoce el pedido y responde con el ciclo de INTA.
5. Durante el primer pulso de INTA, el bus permanece inactivo.
6. Durante el segundo pulso de INTA, el bit correspondiente del ISR se setea y el correspondiente bit de IRR, se resetea. Las salidas de datos del 82C59A ponen el respectivo dato de 8-bit para que lo lea el 8088.

En el modo AEOI, el bit del ISR se resetea, al final del segundo pulso de INTA. En los otros modos, el bit del ISR, permanece seteado hasta que llega el comando EOI apropiado al final de la rutina de atención de interrupción.

Si nivel activo de IR no permanece hasta el paso 5 de la secuencia anterior, el 82C59A asume que el pedido fue de IR7. Si en esa entrada hay programado un esclavo, las líneas CAS, permanecen inactivas, y el byte de "tipo" lo impone el maestro.

Ciclo de INTA del 8259:



El primer ciclo de la secuencia de reconocimiento de INTA, se usa para congelar internamente, el estado de las interrupciones, para resolver las prioridades, y para que el maestro (si existe), coloque el código de identificación del esclavo en las líneas CAS.
En el segundo ciclo de la secuencia de INTA, el maestro (o el esclavo si corresponde), envía el byte del "tipo".

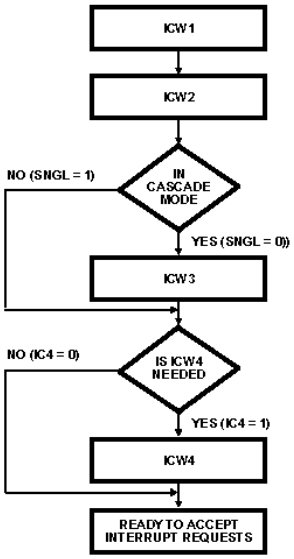
Programación del 82C59A:

La programación del 8259 es muy particular. Para hacerlo se escriben determinados "comandos" en sus registros internos. Lo particular del caso, es que estos registros no se acceden de forma directa, ya que no están mapeados en diferentes direcciones del mapa de I/O. En cambio, varios registros se encuentran mapeados en la misma dirección y se accede a ellos a través de una secuencia particular, controlada por la señal de WR.
No debemos perder de vista que esto es sólo una particularidad de implementación de este dispositivo y no tiene relación conceptual con su funcionalidad.
El 8259 tiene dos grupos de palabras (o comandos) de programación. Los ICW y los OCW.

Comandos de inicialización. (lCWs)
Antes de comenzar con la operación, todos los 82C59As que integren el sistema, deben ser configurados, grabando los ICWs. Estos se graban a través de una secuencia de 2 a 4 ciclos de escritura.
Cuando el 8259 detecta un ciclo de escritura en el cual, la entrada A0 = 0 y cuyo dato contiene el bit D4 = 1, ese dato entrante se interpreta como una lCW1. La grabación de lCW1 comienza la secuencia de inicialización durante la cual sucederá, en forma automática, lo siguiente:

a. Se resetea el circuito de detección de flanco, por lo que luego de la secuencia de inicialización, la entrada IR quedará programada como activa por flanco ascendente. *Para que quede programada como nosotros la usamos, es decir, como activa por nivel, debemos modificar este estado a través de ICW1.*
b. El IMR se borra.
c. A la entrada lR7 se le asigna el menor nivel de prioridad (7). Luego el resto de las entradas tendrán un nivel de prioridad inverso al número de IR.
d. Se borra el Special Mask Mode y el Status Read se setea apuntando al lRR.
e. Si el bit 0 del la palabra ICW1, llamado lC4, es igual a 0, entonces todos los bits del lCW4 se ponen en cero y el 8259 queda configurado así: Non-Buffered mode (ver nota), no Auto-EOI, modo 8080/85. *Para usar el dispositivo con un 8088, deberemos siempre grabar la ICW4.*

Nota: El bit 2 de la ICW4 (Master/Slave), se usa sólo en el buffered mode. En los demás casos, el comportamiento como maestro o esclavo, se configura por medio de una entrada. *No utilizaremos el buffered mode.*



ICW1:

ICW1

A_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	X	X	X	1	LTIM	X	SNGL	IC4

1 = ICWR needed
0 = No ICW4 needed

1 = Single
0 = Cascade Mode

1 = Level triggered mode
0 = Edge triggered mode

- ✓ LTIM: Si LTIM = 1, las entradas IR, son activas por nivel (tal como la entrada INTR del micro). Si LTIM = 0, las entradas IR son activas por flanco ascendente. *Usaremos siempre LTIM = 1.*
- ✓ SNGL: Indica si el 8259 es el único que hay en el sistema. Si SNGL = 1, ICW3 se elimina de la secuencia de programación de las ICWs.
- ✓ IC4: Si es igual a 0, entonces se elimina a ICW4 de la secuencia de secuencia de programación de las ICWs.

ICW2:

ICW2

A_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	T_7	T_6	T_5	T_4	T_3	X	X	X

T_7 - T_3 of interrupt vector address

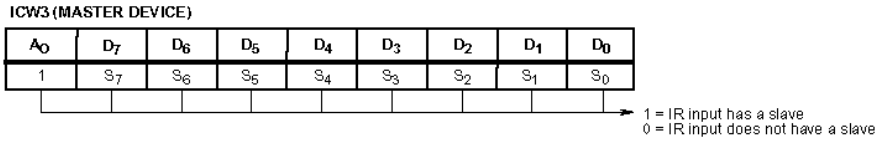
En los bits T7 – T3 se programan los 5 bits más significativos del byte de "tipo". El 8259 pone los otros tres bits menos significativos de acuerdo con el número de la entrada IR a la que corresponde.

	D7	D6	D5	D4	D3	D2	D1	D0
IR7	T7	T6	T5	T4	T3	1	1	1
IR6	T7	T6	T5	T4	T3	1	1	0
IR5	T7	T6	T5	T4	T3	1	0	1
IR4	T7	T6	T5	T4	T3	1	0	0
IR3	T7	T6	T5	T4	T3	0	1	1
IR2	T7	T6	T5	T4	T3	0	1	0
IR1	T7	T6	T5	T4	T3	0	0	1
IR0	T7	T6	T5	T4	T3	0	0	0

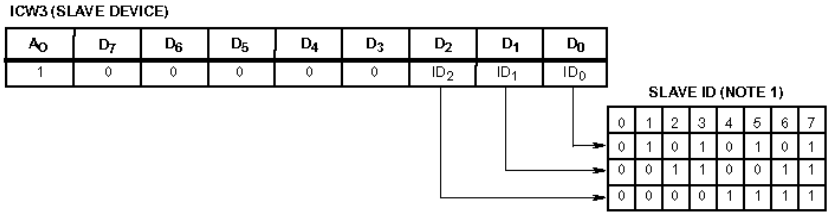
ICW3:

Esta palabra se omite en la secuencia de inicialización cuando el 8259 es el único en el sistema. En cambio si SNGL = 0, el registro cumple distinta función dependiendo si el 8259 es el maestro o el esclavo.

a. Si el 8259 está configurado como master (SP = 1 o si M/S = 1 en el buffered mode) en esta palabra se coloca un "1" en cada bit correspondiente a cada entrada IR en donde hay un esclavo conectado.

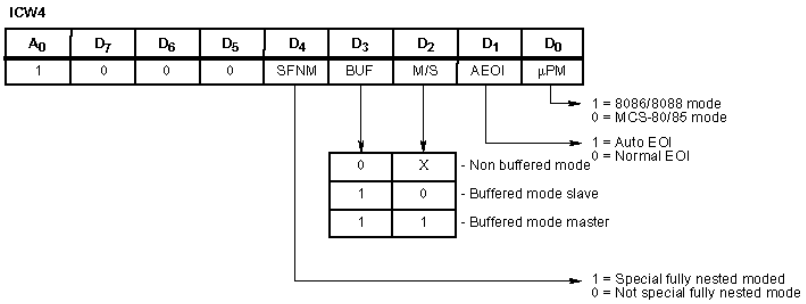


b. Si el 8259 está configurado como esclavo (SP = 0 o si M/S = 0 en el buffered mode), los bits 2 - 0 identifican el esclavo. En la operación normal, el esclavo compara el valor de las líneas de CAS con el contenido de estos bits y si coinciden, el esclavo coloca el byte en el bus en la secuencia de INTA.



Nota: La dirección del esclavo debe corresponderse con el número de la línea a la que está conectada al master.

ICW4:



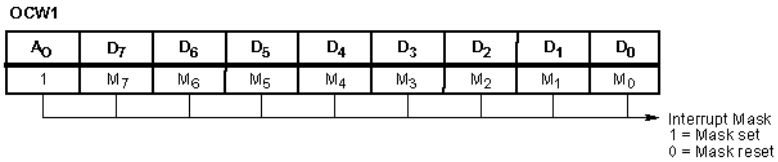
- ✓ SFNM: Si es igual a 1, el 8259 se programa en "Special Fully Nested mode".
- ✓ BUF: Si es igual a 1, el 8259 se programa en "Buffered Mode". En este modo, la entrada SP/EN se transforma en una salida de habilitación y la configuración de maestro / esclavo se hace a través de bit M/S.
- ✓ M/S: En el buffered mode selecciona: M/S = 1 master, M/S = 0 slave. Si BUF = 0, M/S no tiene función.
- ✓ AEOI: Si es igual a 1, se activa el modo "Automatic End of Interrupt".
- ✓ mPM: Selecciona el Microprocessor mode: Si mPM = 0 el 82C59A se configura para funcionar con el 8080/85. Si mPM = 1 el 82C59A queda configurado para funcionar con el 80C86/88/286. *Obviamente a nosotros solo nos interesará este último modo de funcionamiento.*

Palabras de Operación (OCWs)

El hecho de escribir estos registros, actúa como comandos que se envían al 8259 para controlar su funcionamiento.

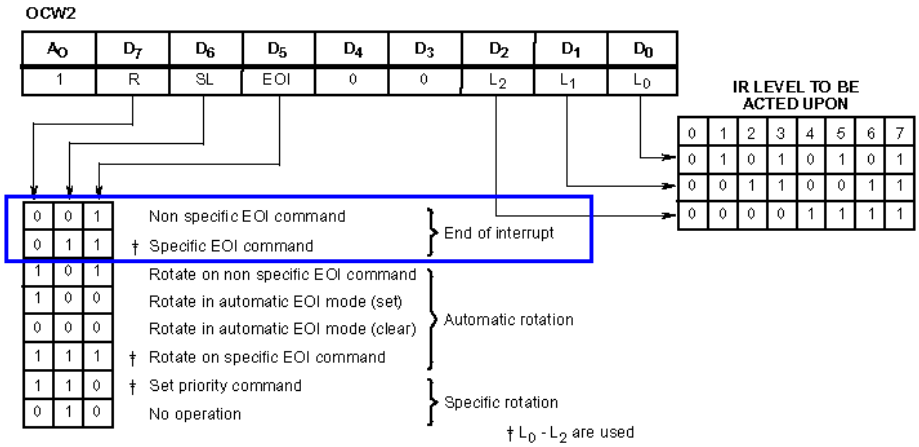
Las OCW's pueden ser escritas en cualquier momento luego de la secuencia de inicialización ya que luego de la programación de las ICW's, el dispositivo está listo para recibir pedidos de interrupción en sus entradas.
Durante la operación, el 8259 puede cambiar su modo de funcionamiento a través de las OCW's. Por ejemplo, se puede cambiar entre: Fully nested mode, Rotating priority mode, Special mask mode y Polled mode.

OCW1:



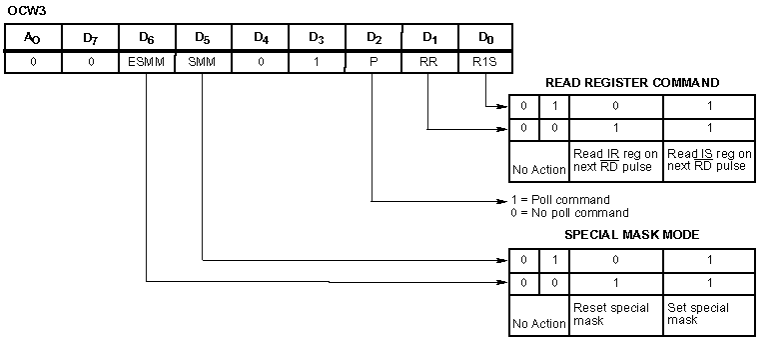
Setea o borra los bits de mascara en el Interrupt Mask Register (IMR). M7 - M0 representan los 8 bits de mascara.
Si Mx = 1 la línea IRx está inhibida, si Mx = 0 la línea IRx está habilitada.

OCW2:



- ✓ R, SL, EOI : Las combinaciones de estos tres bits generan los commandos EOI y los modos de rotación automática y específica. *Usaremos sólo los EOI.*
- ✓ L2, L1, L0 : Estos tres bits determinan el número de IR que será reseteado en el ISR dentro de un EOI específico.

OCW3:



- ✓ ESMM - Enable Special Mask Mode.

Cuando este bit es = 1, este habilita el bit SMM para setear o resetear el "Special Mask Mode". Cuando ESMN = 0, el bit SMM no es considerado.

- ✓ SMM - Special Mask Mode.

Si ESMM = 1 y SMM = 1, el 82C59A está en "Special Mask Mode". Si ESMM = 1 y SMM = 0, el 82C59A está en "Normal Mask Mode". Si ESMM = 0, SMM no es considerado.

- ✓ P = Si P= 1, se activa el Poll command. *No estudiaremos este modo.*
- ✓ RR y R1S: se utilizan para la lectura de los registros internos del 8259.

Modos de Funcionamiento:

Fully Nested Mode

Este modo se activa por defecto, luego de la inicialización.

En él, se le asigna un orden de prioridad a las líneas IR inversa a su número de orden, (IR0 = mayor prioridad, IR7 = menor prioridad.

En este modo, cuando es reconocida una interrupción, se determina el pedido de mayor prioridad y su vector es colocado en el bus de datos. Además, el bit correspondiente del ISR se setea. Este bit permanece en 1, hasta que el micro envía un comando End of Interrupt (EOI). Este comando, que no es más que una escritura

en OCW2, se envía antes de retornar de la rutina de servicio de la interrupción respectiva.

Nota: Recordar que en el modo AEOI (Automatic End of Interrupt) el bit permanece seteado hasta el último flanco del segundo pulso de INTA del ciclo de reconocimiento de la interrupción.

Mientras el bit del ISR de una determinada entrada IR, está seteado, todas las demás interrupciones de igual o menor prioridad, están inhibidas. Si hay pedidos de interrupciones de mayor prioridad, estas serán reconocidas solo si la máscara interna de micro ha sido re habilitada luego de ingresar en la rutina de atención de la interrupción en curso y si no están enmascaradas a través del IMR.

Las prioridades pueden ser cambiadas por medio de los métodos de rotación de prioridades o vía el comando de seteo de prioridades. *No veremos estos modos.*

End of Interrupt (EOI)

Cuando el modo AEOI, está activado, el bit del ISR asociado a una determinada entrada IR, se resetea automáticamente con el último flanco del segundo pulso de INTA de la secuencia de reconocimiento de interrupción.

Cuando AEOI está desactivado, para borrar el bit del ISR, el programador debe enviar al 8259, un comando EOI, antes de finalizar la rutina de atención de la interrupción en cuestión.

En el modo cascada, se debe enviar un comando EOI al esclavo, para borrar el IS de la línea que pidió la interrupción y otro EOI al maestro para borrar el bit IS de la línea en donde está conectado el esclavo a través del cuál entró la interrupción.

Hay dos formas de comando EOI: Specific EOI y Non-Specific EOI.
Cuando el 82C59A opera en modos que preservan la estructura de prioridades, el 8259 puede determinar cuál es el bit del ISR, que debe borrar cuando finaliza la rutina de servicio. Esto es así pues en estos modos, la única rutina de atención que puede estar ejecutándose es la de mayor prioridad de las que figuran en el ISR.
En estos casos, puedo mandar un EOI "Non-Specific" ya que el 8259 va a resetear automáticamente el bit del ISR de mayor prioridad de todos los que están seteados.
Un EOI nonspecific se envía escribiendo en OCW2: (EOl = 1, SL = 0, R = 0).

Cuando se usa un modo que puede perturbar la estructura de prioridades *(todos los que no vemos),* el 82C59A no puede saber cuál es la última IR atendida.
En estos casos, debe enviarse un Specific EOI. Este comando incluye como parte del mismo la información de cuál es el bit del ISR que debe ser reseteado.
Un Specific EOl se envía escribiendo en OCW2 (EOI = 1, SL = 1, R = 0, y en LO - L2 se pone el numero del bit del ISR que se quiere borrar).
Nota: como no vemos estos modos, no nos interesa este comando.

Nota: Un bit del lRR que está enmascarado por el lMR, no será borrado por medio de un non-specific EOI si el 82C59A está en Special Mask Mode.

En el modo AEOI, el 82C59A realiza un EOI no-específico automático en el último flanco del segundo pulso de INTA. Este modo solo se puede usar cuando no se requiere una estructura de prioridades. *Nosotros no lo usaremos.*

Automatic Rotation (Equal Priority Devices)
No estudiaremos este modo.

Specific Rotation (Specific Priority)
No estudiaremos este modo.

Special Mask Mode
No estudiaremos este modo.

Poll Command
No estudiaremos este modo.

Lectura del estado del 82C59A
Para tomar información que puede ser útil en la operación del sistema podemos leer el estado de los registros internos del 8259. Estos pueden ser leídos por medio del OCW3 (el lRR y el ISR) o por medio de OCW1 (el lMR).

El IRR puede ser leído cuando, previo al ciclo de lectura, se escribe en OCW3 un comando de lectura del tipo: (RR = 1, R1S = 0).

El ISR puede leído cuando, previo al ciclo de lectura, se escribe en OCW3 un comando de lectura del tipo: (RR = 1, RIS = 1).

No es necesario escribir una OCW3 antes de cada operación de lectura, mientras el estado de ICW3. Esto no vale cuando se usa el poll mode.
Luego de la inicialización, el 8259 queda seteado para el IRR.

Para leer el IMR, no se necesita de OCW3. Esta operación de lectura se efectuará, cada vez que RD se active y A0=1 (OCW1).

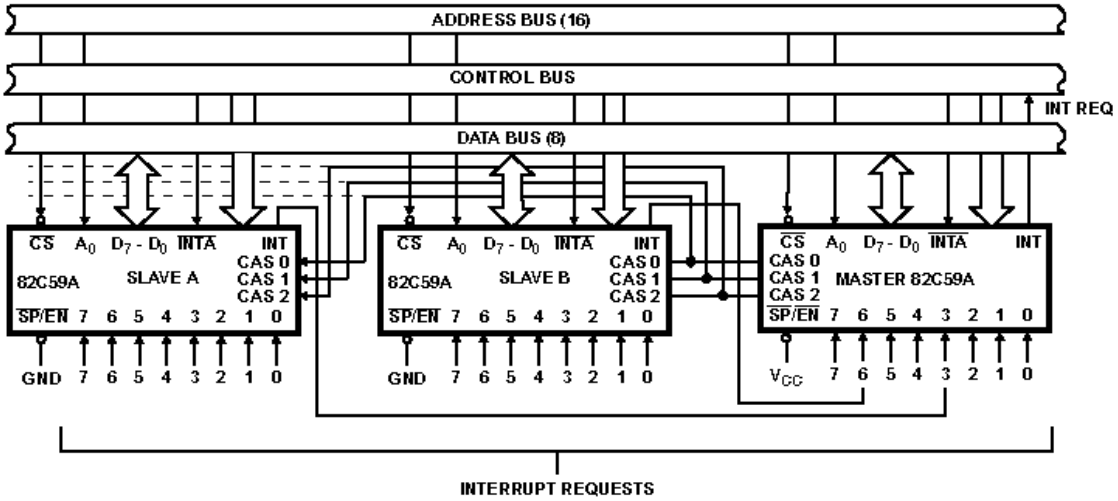
Edge and Level Triggered Modes
Siempre usaremos el modo activo por nivel para las entradas IR, de modo que se comporten como la entrada INTR del micro.

Buffered Mode
No estudiaremos este modo.

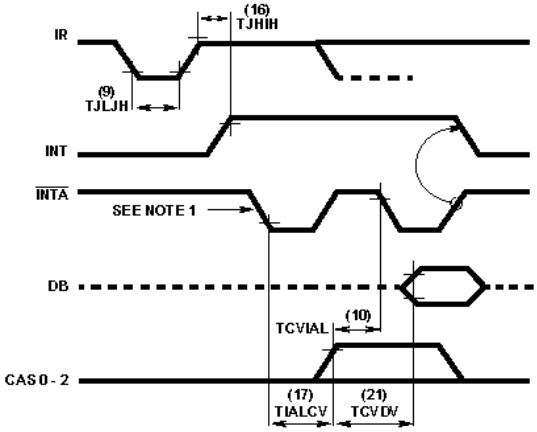
Cascade Mode
El 82C59A puede ser interconectado en un sistema compuesto por un maestro y hasta 8 esclavos permitiendo el manejo de hasta 64 entradas de IR.
El maestro controla a los esclavos a través de 3 líneas (CAS2 - 0) que los seleccionan. Estas líneas forman el bus de cascada.
En esta configuración, las salidas INT de cada esclavo, se conectan a las líneas de entrada IR del maestro.
Cuando un esclavo solicita una interrupción a través de su línea IR, y luego de ser reconocida, el maestro habilitará al esclavo correspondiente para permitirle escribir el "tipo" en el bus de datos durante la secuencia de INTA.

Las líneas del bus de cascada, están normalmente en 0. Contendrán la dirección del esclavo, desde el flanco ascendente del primer puso de INTA hasta el ascendente del segundo.

Dijimos que el bus de cascada, cuando está inactivo, permanece en 0, por lo que la dirección 0 de esclavo, solo debe usarse cuando todas las otras están siendo usadas.



Temporización del bus de cascada:



Nota 1: IR debe permanecer alto hasta el flanco ascendente del primer pulso de INTA.

Normal Fully Nested Mode:
Al colocar en cascada un 8259 como maestro y uno o más 8259 como esclavos, sin activar el Special Fully Nested, el sistema queda en modo Normal.
En este modo, un esclavo queda enmascarado por el maestro, cuando un servicio proveniente de él, está siendo atendido. Por tanto se conservan las prioridades asignadas al maestro, pero no las prioridades dentro de los esclavos.
Para que se conserven esas prioridades se debe usar el Special Fully Nested mode.

Special Fully Nested Mode
Este modo se usa cuando tengo un sistema en cascada y se quiere conservar la estructura de prioridades dentro de cada esclavo.

Para esto se debe programar este modo en el maestro a través de lCW4.
Este modo es similar al "Normal Nested mode" con las siguientes exenciones:

a. Cuando un pedido de interrupción de un esclavo está en servicio, este esclavo no estará bloqueado por el maestro y otras interrupciones de mayor prioridad a la que se está atendiendo y que provengan del mismo esclavo, podrán interrumpir su servicio.

b. Cuando se pretende salir de la rutina que está siendo atendida, la rutina de servicio tiene que chequear que la interrupción de la que se sale, sea la única que está en servicio (en el esclavo de la que provino). Esto se hace por medio del envío de un "Non-specific EOI" al esclavo, luego se lee el ISR del esclavo y se chequea que sea igual a cero. Si esto es así, se envía un "Non-Specified EOI" al maestro. Si no, no se envía otro EOI al maestro.

Ejemplo de un sistema de dos esclavos conectados a un maestro.