INTERRUPCIONES EN LA ARQUITECTURA INTEL IA-32

José Miguel Blázquez Soriano – joblaso@inf.upv.es
Diciembre de 2004

Introducción

El presente proyecto pretende ofrecer un acercamiento teórico a las interrupciones en los procesadores de Intel®; se explican los procedimientos para el manejo de interrupciones en los procesadores Intel® modernos, así como el tratamiento de excepciones y análisis del hardware y software que da soporte a dichas operaciones.

1. Descripción general de Excepciones e Interrupciones

Las interrupciones y las excepciones son eventos que indican que existe una condición en algún lugar del sistema, o del programa en ejecución, que requiere la atención del procesador. Generalmente resultan en una trasferencia forzada del flujo de ejecución hacia una rutina denominada "manejador de interrupciones". Las interrupciones se asocian normalmente a eventos hardware, mientras que las excepciones se producen cuando se detectan ciertas condiciones durante la ejecución, como división por cero, fallos de página, violaciones de segmento, etc.
La arquitectura de errores de máquina del Pentium 4, Intel Xeon, familia P6, y Pentium, permite que se genere una excepción cuando se detectan errores de bus o errores internos de hardware.

El mecanismo para el manejo de excepciones e interrupciones en la arquitectura IA-32 permite que éstas sean manipuladas de manera transparente a los programas de aplicación y al mismo sistema operativo. Cuando se genera una interrupción o una excepción, el procedimiento en ejecución se suspende automáticamente mientras el procesador ejecuta el manejador correspondiente; cuando esta operación se termina, el procesador reanuda la ejecución de la tarea interrumpida. La reanudación del proceso sucede sin pérdida de la continuidad del programa, a menos que el retorno no sea posible o que el evento haya causado la terminación del programa.

2. Vectores de excepción e interrupción

Para ayudar en el manejo de las excepciones e interrupciones, a cada condición que puede causarlas y que requiera atención por parte del procesador, se le asigna un número de identificación denominado **vector**. El procesador emplea ese vector asignado como índice dentro de la Tabla de Descriptores de Interrupción (Interrupt Descriptor Table - IDT) para localizar la dirección de inicio de la rutina del manejador asociado.

El rango permitido para dichos números va de 0 a 255. Los vectores en el rango entre 0 y 31 están reservados para las interrupciones definidas por defecto para la arquitectura; no todos los vectores en ese rango tienen una función definida, algunos pueden estar vacíos y reservados para usos futuros. No se deben usar esos vectores reservados. El resto del rango (32-255) se designan para las interrupciones definidas para el usuario,

generalmente se asignan a dispositivos de E/S externos para que dichos dispositivos puedan interrumpir al procesador. La siguiente tabla muestra la asignación de los vectores tal y como hemos comentado, para cada excepción se muestra el tipo y se indica si el código de error se guarda en la pila, también se muestra su posible causa.

Table 5-1. Protected-Mode Exceptions and Interrupts

Vector No.	Mnemonic	Description	Type	Error Code	Source
0	#DE	Divide Error	Fault	No	DIV and IDIV instructions.
1	#DB	RESERVED	Fault/ Trap	No	For Intel use only.
2	—	NMI Interrupt	Interrupt	No	Nonmaskable external interrupt.
3	#BP	Breakpoint	Trap	No	INT 3 instruction.
4	#OF	Overflow	Trap	No	INTO instruction.
5	#BR	BOUND Range Exceeded	Fault	No	BOUND instruction.
6	#UD	Invalid Opcode (Undefined Opcode)	Fault	No	UD2 instruction or reserved opcode.[1]
7	#NM	Device Not Available (No Math Coprocessor)	Fault	No	Floating-point or WAIT/FWAIT instruction.
8	#DF	Double Fault	Abort	Yes (Zero)	Any instruction that can generate an exception, an NMI, or an INTR.
9		Coprocessor Segment Overrun (reserved)	Fault	No	Floating-point instruction.[2]
10	#TS	Invalid TSS	Fault	Yes	Task switch or TSS access.
11	#NP	Segment Not Present	Fault	Yes	Loading segment registers or accessing system segments.
12	#SS	Stack-Segment Fault	Fault	Yes	Stack operations and SS register loads.
13	#GP	General Protection	Fault	Yes	Any memory reference and other protection checks.
14	#PF	Page Fault	Fault	Yes	Any memory reference.
15	—	(Intel reserved. Do not use.)		No	
16	#MF	x87 FPU Floating-Point Error (Math Fault)	Fault	No	x87 FPU floating-point or WAIT/FWAIT instruction.
17	#AC	Alignment Check	Fault	Yes (Zero)	Any data reference in memory.[3]
18	#MC	Machine Check	Abort	No	Error codes (if any) and source are model dependent.[4]
19	#XF	SIMD Floating-Point Exception	Fault	No	SSE/SSE2/SSE3 floating-point instructions[5]
20-31	—	Intel reserved. Do not use.			
32-255	—	User Defined (Non-reserved) Interrupts	Interrupt		External interrupt or INT *n* instruction.

NOTAS:
1. La instrucción UD2 fue introducida en el procesador Pentium Pro.
2. Los procesadores IA-32 después del Intel386 no generan esta excepción.
3. Esta excepción se introdujo en el procesador Intel486.
4. Esta excepción se introdujo en el Pentium y en la familia P6.
5. Esta interrupción se introdujo en la familia Pentium III.

3. Fuentes de interrupciones

El procesador puede recibir interrupciones de dos fuentes:

1. Interrupciones externas (provocadas por hardware).
2. Interrupciones generadas por software.

Los procesadores 8088, 20286 y siguientes disponen de dos patillas para este servicio específico. Las designadas **INTR** y **NMI**, que sirven para atender las interrupciones enmascarables y no enmascarables respectivamente. A su vez, el procesador utiliza ciertas señales en algunas de sus patillas para generar un ciclo **INTA** ("Interrupt Acknowledge"), que sirve para notificar al **PIC** que ha recibido la interrupción.

3.1. Interrupciones externas

Estas interrupciones se reciben a través de los pines del procesador o a través del APIC (*Advanced Programmable Interrupt Controller – Controlador de interrupciones programable avanzado*) local. Los pines primarios de interrupción en el Pentium 4, Intel Xeon, familia P6 y Pentium son LINT[1:0], que están conectados al APIC local. Cuando el APIC local está habilitado, los pines LINT[1:0] pueden ser programados a través de la *tabla local de vectores del APIC* (LVT) para asociarse a alguno de los vectores de excepciones o interrupciones del procesador.

Cuando el APIC local está deshabilitado, esos pines se configuran como pines INTR y NMI respectivamente. Cuando el pin INTR envía señal al procesador cuando ocurre una interrupción externa, el procesador lee del bus del sistema el número del vector de interrupción proporcionado por un controlador de interrupciones externo como el 8259A. El pin NMI señaliza interrupciones no enmascarables (Non-maskable Interrupt), que se asigna al vector de interrupción 2, como podemos ver en la tabla anterior.

Los pines LINT[1:0] no están disponibles en los 486 ni en los primeros Pentium que no contenían un chip local de APIC. En su lugar tenían pines exclusivos dedicados a NMI y INTR. Con esos procesadores, las interrupciones externas se generan con un controlador de interrupciones (8259A), señaladas a través del pin INTR.

3.2. Interrupciones generadas por software

La instrucción INT *n* permite a las interrupciones ser generadas desde dentro del software utilizando el número del vector de interrupción como un operando. Por ejemplo, la instrucción INT 35 fuerza una llamada implícita al manejador de interrupciones para la interrupción 35.

Cualquiera de los vectores de interrupción desde el 0 al 255 se puede utilizar como parámetro de esta función. Las interrupciones generadas por software mediante la instrucción INT *n* no se pueden enmascarar por el flag IF del registro EFLAGS.

3.B Interrupciones hardware enmascarables / no-enmascarables

Cualquier interrupción externa que se entregue al procesador a través del pin INTR o a través del APIC local se denomina **interrupción hardware enmascarable,** además, bajo control software, el procesador es capaz de aceptar o ignorar dichas interrupciones. El procedimiento es el siguiente: se envía una señal a la patilla INTR, y en función del flag IF del registro FLAGS, la interrupción es atendida o ignorada.

Si se acepta, el procesador termina la instrucción que estuviera ejecutando y responde mediante una combinación de señales INTA; generalmente dos, la primera para avisar al PIC y la segunda para indicarle al PIC que debe colocar un byte en el bus de datos con el número de la interrupción, para que el procesador pueda interpretar el servicio solicitado y por tanto, el manejador al cual debe saltar. El valor que el procesador recibe se multiplica desplazando binariamente dos posiciones a la izquierda. Seguidamente, se salvan en la pila los valores del registro del contador de programa y de segmento de código, se deshabilita el flag IF y se ejecuta el servicio.

Interrupción no enmascarable significa que la interrupción no puede ser deshabilitada por software. Este tipo de interrupciones ocurren cuando se recibe una señal en la patilla **NMI** ("Nonmaskable Interrupt") del procesador; se reservan para casos en que es crítica la respuesta, por ejemplo que se detecte un error de paridad en la memoria. Además son de prioridad más alta que las enmascarables. La única forma de enmascarar estas interrupciones es a través de circuitos externos al procesador, por ejemplo a nivel del PIC. Cuando el procesador recibe una de estas instrucciones no se genera ningún ciclo de reconocimiento de la instrucción (INTA), y el procesador le asigna un 2 como número de excepción.

La penúltima instrucción de la rutina de servicio es enviar una señal para indicar el fin de la interrupción (EOI) para que el PIC pueda seguir enviando interrupciones (sino nos encontraríamos en un bucle infinito); a continuación se restauran los registros a su estado inicial (existente antes de que se produjera la interrupción).

4. Fuentes de excepciones

El procesador puede recibir excepciones mediante tres formas.

1. Errores de programa detectados por el procesador
2. Excepciones generadas por software
3. Excepciones de error de máquina.

Las primeras se generan como consecuencia de errores detectados en la ejecución de un programa de usuario o del propio sistema operativo. En la arquitectura IA-32 se definen unos números especiales para identificarlas, más adelante veremos su clasificación.

Las instrucciones INTO, INT 3 y BOUND permiten generar excepciones por software. Estas instrucciones permiten comprobar ciertas condiciones a la hora de determinar la excepción en puntos determinados del flujo de ejecución. Por ejemplo, la instrucción INT 3 genera una excepción por “breakpoint” en el flujo de ejecución.

La instrucción INT *n* puede ser utilizada para emular ciertas excepciones software.

La familia P6 y de procesadores Pentium incorporan internamente unos mecanismos para control que permiten verificar las operaciones del hardware de chips interno y las transacciones de bus. Estos mecanismos constituyen procedimientos de excepciones extendidos. Cuando se detecta un error de este tipo, el procesador emite la correspondiente excepción (vector 18) y devuelve un código de error.

5. Peticiones de Interrupción

Una petición de interrupción (IRQ – Interrupt Request) es una señal generada por un dispositivo hardware para solicitar la atención del procesador. La señal solicita al procesador que suspenda su actividad y atienda al dispositivo (por ejemplo un periférico) que la ha generado. Estas señales juegan un papel fundamental en el funcionamiento de los ordenadores actuales, y son la base del funcionamiento de sistemas operativos como Windows y Linux. Gracias a las interrupciones, el sistema evita el constante chequeo del estado de los dispositivos (prueba de estado), son los dispositivos los que avisan al procesador si lo necesitan, de este modo se ha conseguido aumentar el rendimiento de los procesadores.

En el caso de la prueba de estado, la UCP lleva la iniciativa accediendo al registro de estado del dispositivo para ver si está preparado, esta es la gran diferencia entre la sincronización por prueba de estado, y por interrupciones.

Líneas de petición de interrupción

El bus de control dispone de líneas especiales para el subsistema de interrupciones. En los PC XT existen 8, aunque solo hay disponibles 6 (las dos primeras están asignadas al temporizador y al teclado). En el momento de aparición del modelo AT se añadió un segundo procesador PIC colgado de la IRQ2, el cual ofrecía otras 8 IRQs.
A pesar de que se manejan 16 líneas, no todas tienen contacto con el bus externo, como la IRQ0, IRQ1 e IRQ8.

Las otras tiene conexión directa con el zócalo y pueden ser empleadas para la asignación de nuevos dispositivos, aunque algunas de ellas están reservadas a dispositivos estándar, como la IRQ7 de la impresora primaria; y otras para los puertos COM, etc.

En la tabla adjunta se puede apreciar la descripción de la asignación de algunas IRQs.

Nombre	Int (hex)	XT: Descripción	AT: Descripción
NMI	--	Paridad	Paridad
IRQ0	8	Temporizador	Temporizador
IRQ1	9	Teclado	Teclado
IRQ2	A	Reservado	IRQ 8 a 15
IRQ3	B	Puerto serie COM2	Puerto serie COM2
IRQ4	C	Puerto serie COM1	Puerto serie COM1
IRQ5	D	Disco duro	Puerto LPT2

IRQ6	E	Disquete	Disquete
IRQ7	F	Puerto LPT1	Puerto LPT1
IRQ8	70	--	Reloj de tiempo real
IRQ9	71	--	No asignada
IRQ10	72	--	No asignada
IRQ11	73	--	No asignada
IRQ12	74	--	No asignada
IRQ13	75	--	Coprocesador 80287
IRQ14	76	--	Disco duro
IRQ15	77	--	No asignada

Ejemplo de la asignación de IRQs en un sistema Windows 98:

IRQ 0	Cronómetro del sistema	OK
IRQ 1	Teclado estándar de 101/102 teclas o MS Natural Keyboard	OK
IRQ 2	Controlador programable de interrupciones	OK
IRQ 3	Puerto de comunicaciones (COM2)	OK
IRQ 4	Puerto de comunicaciones (COM1)	OK
IRQ 5	Emulación de Creative SB16	OK
IRQ 6	Controlador estándar de disquetes	OK
IRQ 7	EPSON Printer Port (LPT1)	OK
IRQ 8	Sistema CMOS/reloj en tiempo real	OK
IRQ 9	SCI IRQ utilizada por el bus ACPI	OK
IRQ 10	Controladora de host universal VIA Tech 3038 PCI a USB	OK
IRQ 10	Creative SB Live!	OK
IRQ 10	Marcador de IRQ ACPI para manejo de IRQ PCI	OK
IRQ 10	Marcador de IRQ ACPI para manejo de IRQ PCI	OK
IRQ 11	3D Blaster GeForce Pro	OK
IRQ 11	Realtek RTL8139/810X Family PCI Fast Ethernet NIC	OK
IRQ 11	Marcador de IRQ ACPI para manejo de IRQ PCI	OK
IRQ 12	Puerto de mouse compatible con PS/2	OK
IRQ 13	Procesador de datos numéricos	OK
IRQ 14	Controlador VIA Bus Master PCI IDE	OK
IRQ 14	Controlador primario IDE (FIFO doble)	OK
IRQ 15	Controlador VIA Bus Master PCI IDE	OK
IRQ 15	Controlador secundario IDE (FIFO doble)	OK

Cuando se realiza la instalación de un nuevo dispositivo que pueda requerir la atención del procesador, se le debe asignar una IRQ disponible. Actualmente, esto se deja en manos del sistema operativo, que gracias a tecnologías como Plug&Play, se consigue la asignación del recurso de manera rápida y sencilla.

El Controlador PIC

Evidentemente, toda la parafernalia anterior debe ser gobernada por un chip, en este caso se le conoce como PIC ("Programmable Interrupt Controller"). El que manejaba

originalmente 8 IRQ era conocido como 8259A, la cual se le añadió otro para poder manejar otras 8 más.

En el sistema, las peticiones que antes se atienden son las de número más bajo (lo cual indica su prioridad). De modo que las primeras que se atienden son la IRQ0 y 1, (correspondientes al cronómetro del sistema y al teclado), después de la IRQ8 a la IRQ15 del controlador añadido (esto es así, porque el controlador añadido cuelga directamente de la IRQ2, y por tanto hereda su prioridad, antes que la IRQ3 y sucesivas) y finalmente de la IRQ3 a la IRQ7 del PIC maestro. Como podemos observar, las prioridades mas bajas corresponden a la disquetera y a la impresora.

Tareas realizadas por el PIC:

- Puesto que existen muchos dispositivos que pueden solicitar interrupciones, es responsabilidad del **PIC** priorizarlas cuando existen varias IRQ's simultáneas.
- Después de enviar una solicitud de interrupción, debe enviar un número de interrupción (número de vector) cuando el procesador indica que está listo para atender la petición.
- Mantiene un registro de que se está procesando una interrupción; cuando esto sucede, no envía más peticiones al procesador hasta que este le responde con una señal EOI ("End Of Interrupt"), indicando que la rutina de servicio precedente ha terminado o puede aceptar otra interrupción.
- Puede enmascarar de forma selectiva cualquiera de las 8 IRQ's que tiene conectadas.

6. Prioridad en las interrupciones

Si más de una excepción / interrupción se activan al mismo tiempo, el procesador elige una de ellas atendiendo a la prioridad detallada en la siguiente tabla. El resto quedan clasificadas atendiendo al mismo criterio:

Priority	Descriptions
1 (Highest)	Hardware Reset and Machine Checks - RESET - Machine Check
2	Trap on Task Switch - T flag in TSS is set
3	External Hardware Interventions - FLUSH - STOPCLK - SMI - INIT
4	Traps on the Previous Instruction - Breakpoints - Debug Trap Exceptions (TF flag set or data/I-O breakpoint)

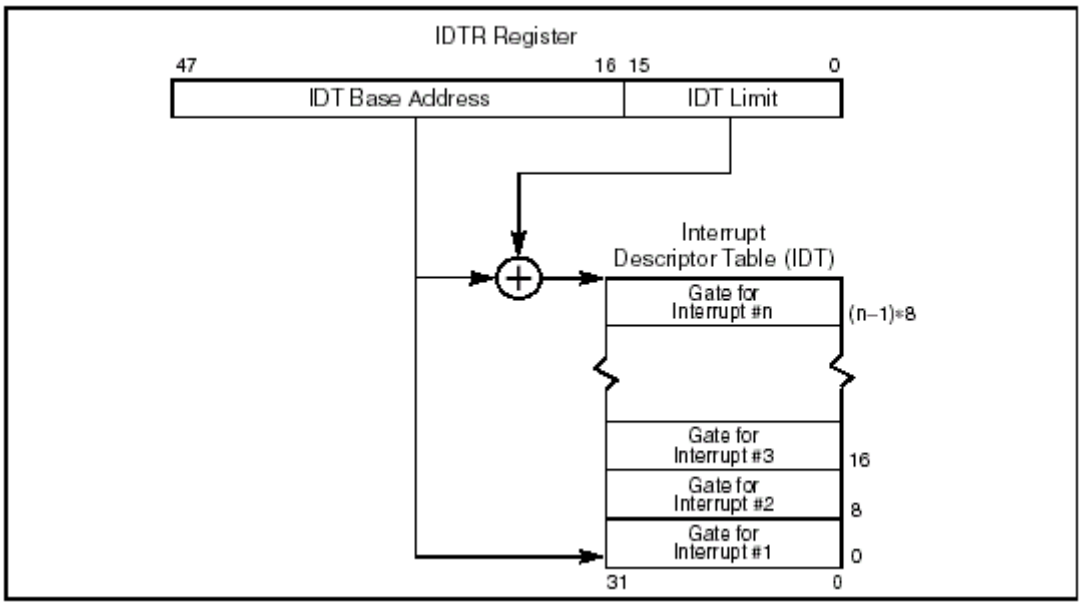
Priority	Descriptions
5	External Interrupts - NMI Interrupts - Maskable Hardware Interrupts
6	Code Breakpoint Fault
7	Faults from Fetching Next Instruction - Code-Segment Limit Violation - Code Page Fault
8	Faults from Decoding the Next Instruction - Instruction length > 15 bytes - Invalid Opcode - Coprocessor Not Available
9 (Lowest)	Faults on Executing an Instruction - Overflow - Bound error - Invalid TSS - Segment Not Present - Stack fault - General Protection - Data Page Fault - Alignment Check - x87 FPU Floating-point exception - SIMD floating-point exception

7. Tabla de Descriptores de Interrupción

Como ya se comentó en un principio, la Tabla de Descriptores de Interrupción (IDT – Interrupt Descriptor Table) asocia cada vector de excepción o interrupción con un descriptor en una posición de la tabla que indica la tarea a usar en el caso de que se tenga que servir a dicha excepción o interrupción. La tabla está formada por un vector de descriptores de 8 bytes (en modo protegido). Para indexar la IDT el procesador escala el vector de 8 en 8 (bytes por descriptor). Puesto que sólo hay 256 vectores de interrupción, la IDT contendrá 256 posiciones como máximo, de hecho suele contener menos, ya que sólo se necesitan los descriptores de las interrupciones o excepciones que ocurran cada cierto tiempo. Todos los descriptores que no tienen asignación (están vacíos), tienen el flag puesto a 0.

La dirección base de la IDT debe estar alineada a un límite de 8 bytes para maximizar las prestaciones del uso de las líneas de la caché. El valor límite se expresa en bytes y se añade a la dirección base para obtener la dirección del último byte válido. Un valor límite de 0, resulta en un byte válido. Puesto que las entradas de la IDT son siempre de 8 bytes de largo, el límite siempre debe ser uno menor que un múltiplo integral de 8 (es decir, 8N-1).

La IDT puede residir en cualquier lugar del espacio lineal de direcciones. Como vemos en la siguiente figura, el procesador localiza la IDT gracias al registro IDTR. Este registro mantiene tanto una dirección base de 32 bits, como un límite de 16 bits para la IDT.



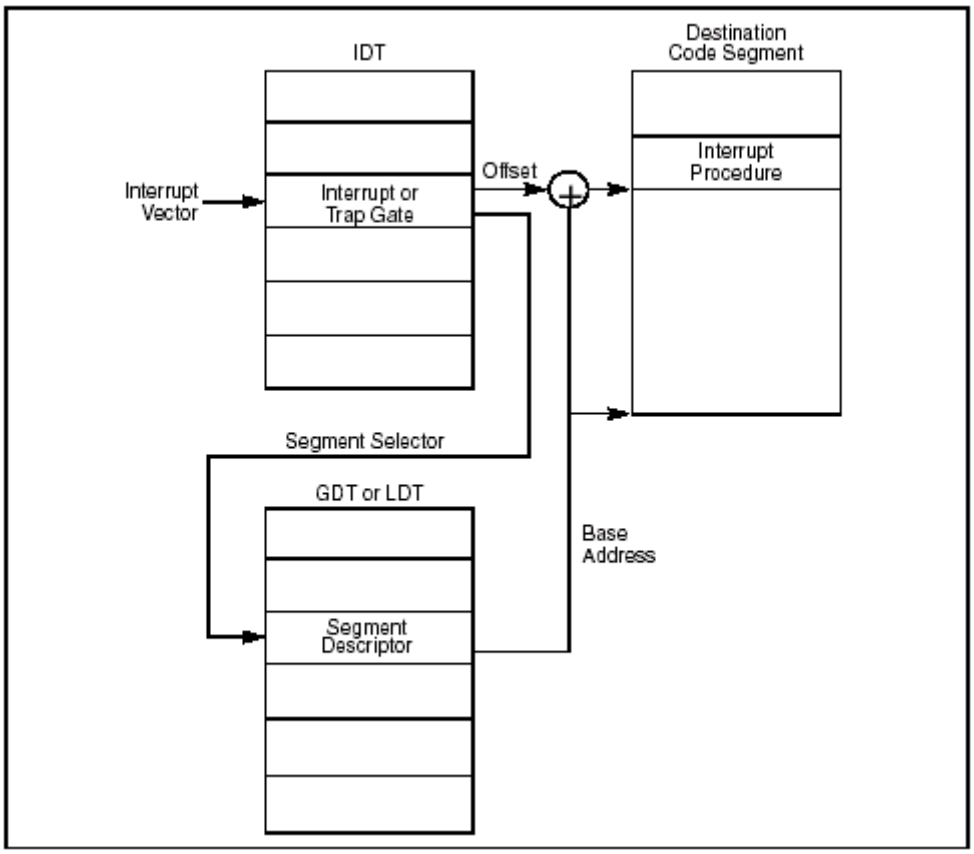
Las instrucciones LIDT (Load IDT register) y SIDT (Store IDT register), cargan y almacenan los contenidos del registro IDTR, respectivamente. La instrucción LIDT carga el registro IDTR con la dirección base y el límite contenido en un operando de memoria. Esta instrucción sólo puede ser ejecutada cuando el CPL es 0. Generalmente es usada por el código de inicialización de un sistema operativo cuando se crea la IDT.
El sistema operativo también la puede usar para cambiar de una IDT a otra. La instrucción SIDT copia la base y el límite almacenados en el registro IDTR a la memoria. Esta instrucción puede ser ejecutada sin privilegios.
Si un vector referencia un descriptor más allá del límite de la IDT, se genera una excepción de protección general (#GP).

8. Manejo de Excepciones e interrupciones

El procesador maneja las llamadas a las excepciones e interrupciones de manera similar a como lo hace con la instrucción CALL a un procedimiento o tarea. Cuando se responde a una excepción o interrupción, el procesador emplea el vector asociado, como un índice para la IDT.

Procedimientos para el manejador de excepciones o interrupciones

Una puerta de interrupción o trampa referencia a un procedimiento de manejo de excepción o interrupción que se ejecuta en el contexto de la tarea actualmente en ejecución. El selector de segmento apunta a un descriptor de segmento para un segmento de código ejecutable en GDT o LDT. El campo offset del descriptor de puerta apunta al inicio del manejador de la excepción/interrupción.



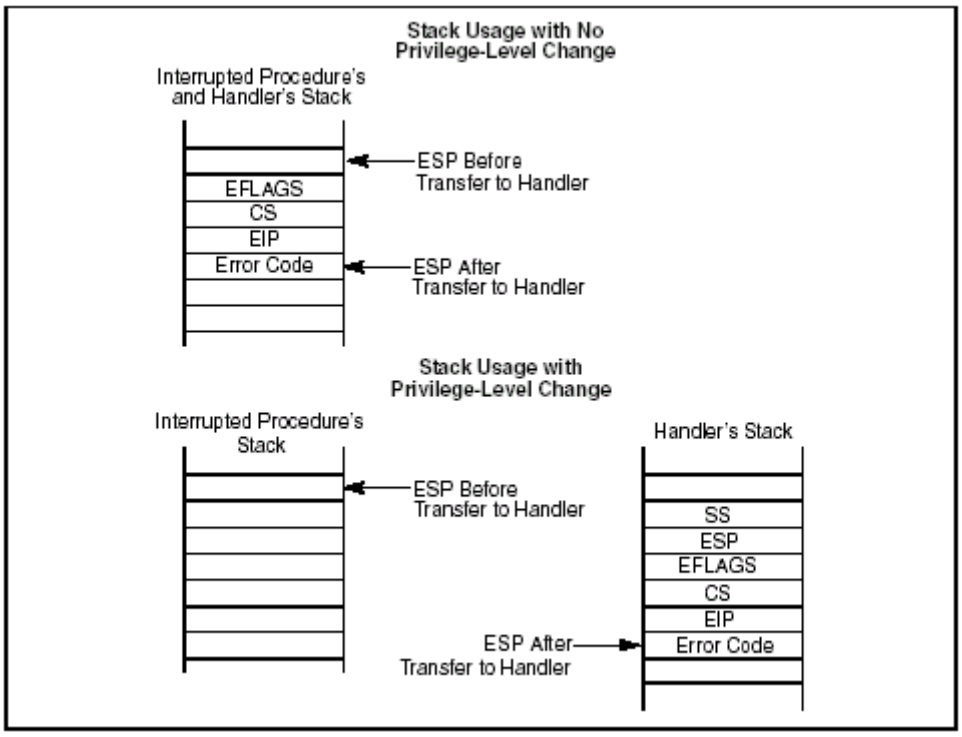
Cuando el procesador lleva a cabo una llamada al procedimiento del manejador de excepciones o interrupciones:

- Si el manejador se va a ejecutar con un nivel de privilegio numéricamente bajo, ocurre un cambio de pila. Cuando ocurre este cambio:

    1. El selector de segmento y el puntero de pila para la pila que va a usar el manejador, se obtienen del TSS para la tarea actualmente en ejecución. En esta nueva pila, el procesador apila el selector de segmento de pila y el puntero de pila del procedimiento interrumpido.
    2. Entonces, el procesador guarda el estado actual de los EFLAGS, CS y EIP en la nueva pila.
    3. Si una excepción provoca que un código de error se guarde, se apila en la nueva pila después del valor EIP.

- Si el manejador se va a ejecutar al mismo nivel de privilegios que el procedimiento interrumpido:

    1. El procesador guarda el estado actual de los registros EFLAGS, CS y EIP en la pila actual.
    2. Si una excepción provoca que un código de error se guarde, se apila en la pila actual después del valor EIP.



Para retornar de un manejador de excepciones o interrupciones, el manejador debe utilizar la instrucción IRET (o IRETD). La instrucción IRET es similar a la instrucción RET sólo que la primera restaura los flags guardados en el registro EFLAGS. El campo IOPL del registro EFLAGS es restaurado sólo si el CPL es menor o igual que el IOPL. Si se produce un cambio de pila cuando se llama a un manejador, la instrucción IRET cambia de nuevo a la pila del procedimiento interrumpido cuando se retorna.

8.B. Protección de los procedimientos de manejo de excepciones e interrupciones.

La protección por nivel de privilegios para los procedimientos de manejo de excepciones e interrupciones es similar a la empleada para las llamadas a procedimientos ordinarios a través de una "Call Gate". El procesador no permite la transferencia del flujo de ejecución a un procedimiento de manejo de excepciones o de interrupciones, en un segmento de código menos privilegiado que el CPL.

Un intento de violar esta regla da lugar a una excepción de protección general (#GP). El mecanismo de protección para los procedimientos de manejo de excepciones o interrupciones se diferencia por:

- Porque los vectores de excepciones e interrupciones no disponen de RPL, el RPL no se controla en las llamadas implícitas a los manejadores de excepciones o int.
- El procesador comprueba el DPL de la interrupción sólo si la excepción o interrupción se han generado con las instrucciones INT *n*, INT 3 o INTO. El CPL debe ser menor o igual al DPL de la puerta. Esta restricción previene que las aplicaciones o procedimientos ejecutándose a nivel 3 de privilegios utilicen una interrupción por software para acceder a manejadores de excepción críticos,

como el manejador de fallos de página, teniendo en cuenta que dichos manejadores se sitúan en segmentos de código más privilegiados (menor número de privilegio). Para la interrupciones generadas por hardware y las excepciones detectadas por el procesador, el procesador ignora el DPL de la interrupción.

Puesto que el suceso de una excepción o interrupción no se puede predecir, estas normas de privilegios imponen restricciones a los niveles de privilegios a los que los manejadores de excepciones e interrupciones puede ejecutarse. Cualquiera de las siguientes técnicas se puede emplear para evitar las violaciones de nivel de privilegios:

- El manejador de excepciones e interrupciones puede situarse en un segmento de código que le permita acceder a los datos disponibles en la pila (excepciones de error de división, etc). Si el manejador necesita datos de un segmento de datos, dicho segmento debe ser accesible desde el nivel 3 de privilegios, con lo cual queda desprotegido.
- El manejador puede situarse en un segmento de código con nivel 0 de privilegios. Dicho manejador siempre se ejecutará, sin importar el CPL al que el programa interrumpido estaba ejecutándose.

Bibliografía de referencia

- **"IA-32 Intel(R) Architecture Software Developer's Manual, Volume 3 System Programming Guide"** – Intel Corporation ©2004

- **"IA-32 Intel(R) Architecture Software Developer's Manual Volume 1 Basic Architecture"** – Intel Corporation ©2004

- http://www.zator.com/Hardware/H2_4.htm - 2.4. Interrupciones