Teórica 4

Table of contents

Interrupciones

Fuentes de interrupción

Hardware

Señales eléctricas enviadas desde los dispositivos de hardware, generalmente concentradas en un controlador externo que las prioriza y envía a la CPU secuencialmente de acuerdo a este criterio.

- **Características**: Asincronicas. No determinísticas.

Software

Se producen cuando el software ejecuta INT n.

- **Característica**: Determinísticas.
  La instruccion *INT* se intercala donde se desea generar la interrupcion.

Internas

Se conocen como *excepciones*.
Son generadas por la propia CPU a consecuencia de una situacion que le impide completar la ejecucion de la instrucción en curso.
Por ejemplo, una division por cero, un page fault, una violación de proteccion, entre otras.

Identificación de la fuente de interrupción

- Es provisto por el hardware en interrupciones que ingresan por el pin INTR del procesador.
- Esta asociado a un pin del procesador, tal el caso de la Interrupción no enmascarable NMI que lleva asociado el tipo 2.
- Acompana al codigo de operacion en la instruccion *INT* type, para el caso de las interrupciones por software, por ejemplo INT 80h.
- Esta asociado a una instruccion (*INTO*,por ejemplo, genera una interrupcion de tipo 4).
- Las excepciones tienen asignado un tipo específico cada una.

Tipo	Mnemónico	Descripción	Clase	Código de Error	Fuente
0	#DE	Error de División	Fault	NO	Instrucciones DIV e IDIV
1	#DB	Reservada	Fault/Trap	NO	Solo para uso de Intel
2	-	NMI	Interrupción	NO	Interrupción No enmascarable. Pin NMI
3	#BP	BreackPoint	Trap	NO	Opcode 0xCC
4	#OF	Overflow	Trap	NO	Instrucción INTO
5	#BR	BOUND Range Exceeded	Fault	NO	Instrucción BOUND
6	#UD	Invalid Opcode	Fault	NO	Instrucción UD2 u Opcode Reservado
7	#NM	Coprocesador No disponible	Fault	NO	Instrucciones de Punto Flotante o WAIT / FWAIT
8	#DF	Doble Fault	Abort	SI (Cero)	Cualquier instrucción capaz de generar una excepción, una señal en NMI o en INTR
9		Coprocessor Segment Overrun (reservada)	Fault	SI	Instrucciones de Punto Flotante
10	#TS	TSS Inválido	Fault	SI	Task switch o acceso a un TSS
11	#NP	Segmento No Presente	Fault	SI	Carga o acceso a un registro de segmento

Vectorización en modo protegido

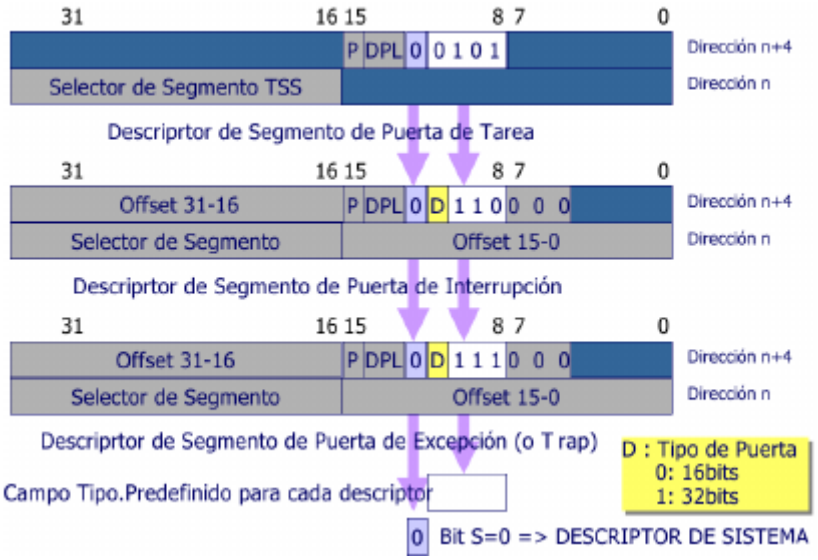
Interrupt Descriptor Table

Cuando el procesador trabaja en Modo Protegido debe definirse una tabla en memoria, llamada IDT (por Interrupt Descriptor Table), que a diferencia del funcionamiento en Modo Real, no almacena vectores de interrupción, sino que como su nombre lo indica almacena Descriptores.
Esta tabla tiene únicamente 256 entradas, ya que esa es la cantidad de tipos de interrupciones diferentes que maneja el microprocesador.

La IDT puede contener 3 tipos de descriptores solamente:

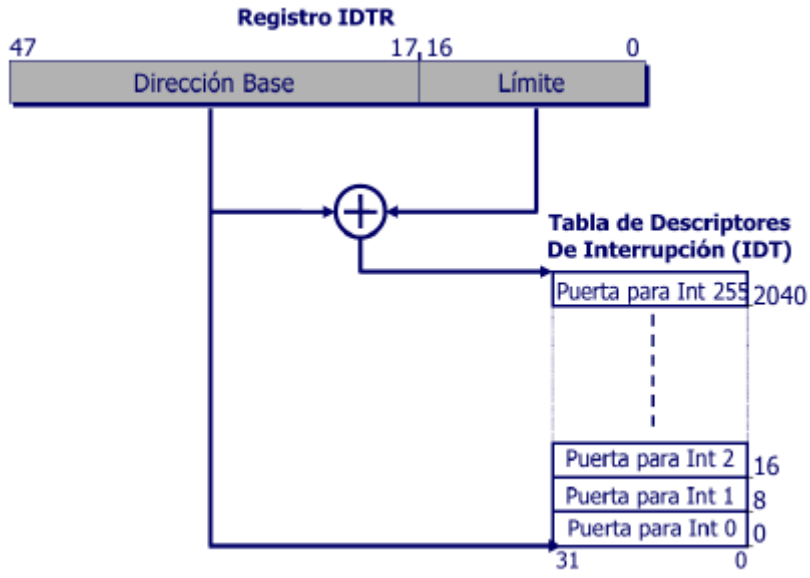
1. **Interrupt Gate** (de 16, 32, o 64 bits)
2. **Trap Gate** (de 16, 32, o 64 bits)
3. **Task Gate** (solo en el modo protegido de 32 bits!!, en 64 bits no existen)

- En los tres casos se trata de descriptores de Sistema (Bit S=0 en el descriptor)
  No se debe definir en la IDT un descriptor de segmento de datos ni de código, ni otro tipo de Descriptor de Sistema (S=0) diferente de los tres enunciados anteriormente en modo protegido de 32 bits, o de los dos pirmeros en el modo 64 bits. El procesador generará invariablemente una excepción de protección general al intentar vectorizar esa interrupción.

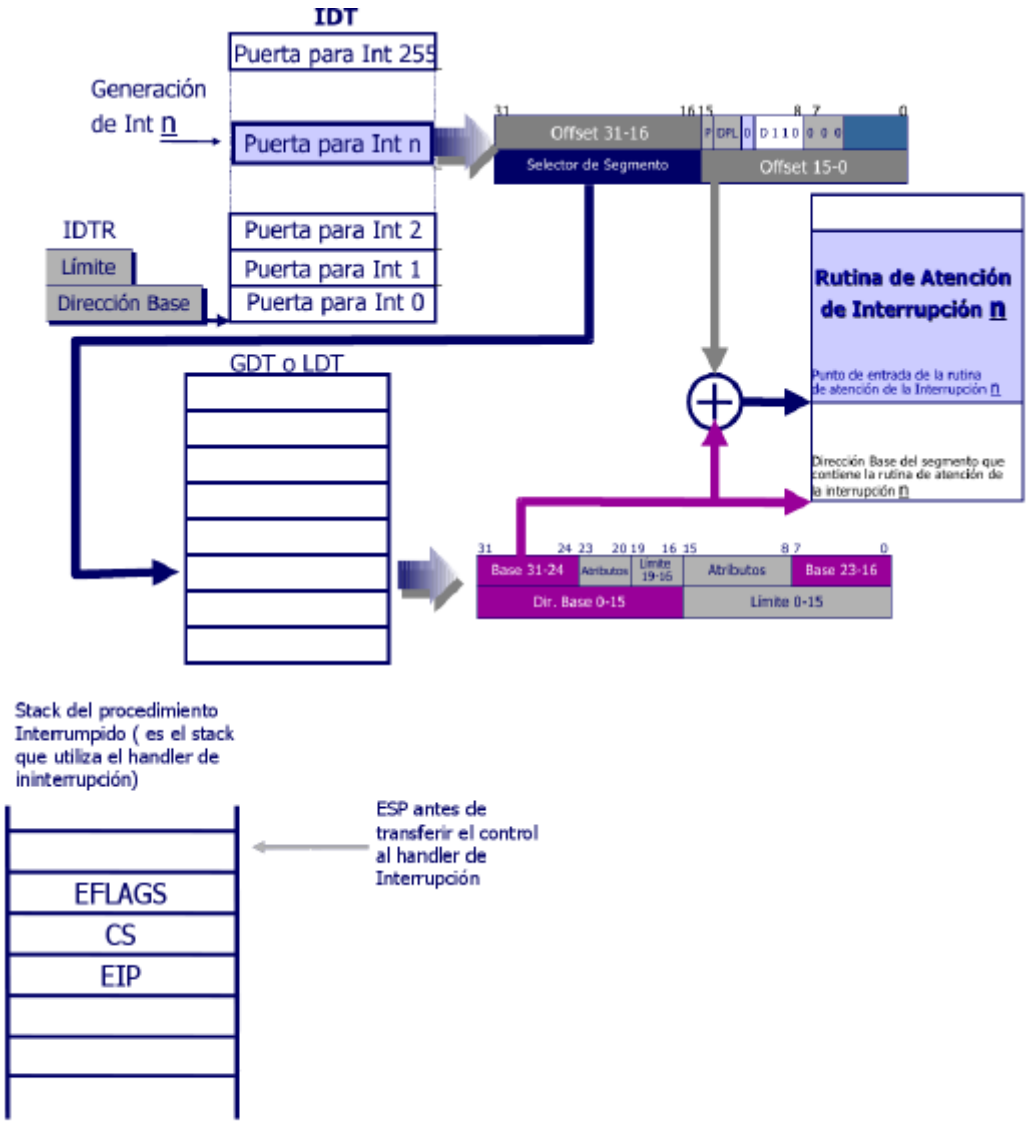


> Los descriptores no definen un segmento, definen la direccion lógica donde estará la rutina de atención de la interrupción.

IDTR



Vectorización de la interrupción



Excepciones

La excepciones se dividen en tres tipos

1. Fault: Excepción que puede corregirse permitiendo al programa retomar la ejecución de esa instrucción sin perder continuidad. El procesador guarda en la pila la dirección de la instrucción que produjo la falla.
2. Traps: Excepción producida inmediatamente a continuación de una instrucción de trap. Algunas permiten al procesador retomar la ejecución sin perder continuidad. Otras no. El procesador guarda en la pila la dirección de la instrucción a ejecutarse luego de la instrucción trapeada.
3. Aborts: Excepción que no siempre puede determinar la instrucción que la causó, ni permite recuperar la ejecución de la tarea que la causó. Reporta errores severos de hardware o inconsistencias en tablas del sistema.

Código de error

Una excepción identificada con un determinado tipo (por ejemplo # GP), puede obedecer a numerosas causas, todas ellas de diferente origen.

Una vez vectorizado el handler de la interrupción, necesitamos determinar cual ha sido la causa concreta de la excepción para poder resolverla.
Por ello, en excepciones que pueden tener causas múltiples, se provee en la pila una palabra de 4 bytes denominda Código de Error.
El Capítulo 6 del Vol 3A de Intel es muy detallado en este aspecto ya que allí se describen cada interrupción con sus características, de modo que es necesario prestar mucha atención a esta documentación cuando se diseña un handler de Interrupción.

1. **EXT:** External Event (bit 0): Se setea para indicar que la excepción ha sido causada por un evento externo al procesador.
2. **IDT:** Descriptor Location (bit 1): Cuando está seteado indica que el campo Segment Selector Index se refiere a un descriptor de puerta en la IDT: Cuando está en cero indica que dicho campo se refiere a un descriptor en la GDT o en la LDT de la tarea actual.
3. **TI:** GDT-LDT (bit 2): Tiene significado cuando el bit anterior está en cero. Indica a que tabla de descriptores corresponde el selector del campo Indice: 0 GDT, 1 LDT.

Cambio a nivel de privilegio

En un sistema Multitarea, el kernel es quien maneja las Interrupciones, y Excepciones. ¡SIEMPRE!.
Si al generase la excepción o la Interrupción se estaba ejecutando código del Kernel (KERNEL Mode), el resto de esta sección no aplica.
Si la interrupción o excepción se ha producido en medio del código de una aplicación (Modo USER), será el Kernel quien tome el control.
Conclusión: No hay cambio de contexto, pero si hay un cambio de Nivel de privilegio de ejecución.
Entonces debemos tener presentes las reglas de protección del procesador.

En particular:

> Un segmento de pila puede ser accedido por segmentos de código de mismo nivel de privilegio (Atributo DPL de sus descriptores respectivos).

- Los datos que se terminan de guardar, se han almacenado en la pila de la aplicación, ya que la interrupción no cambia el contexto de ejecución actual. Es decir la interrupción no produce un task switch.

> Por este motivo, necesariamente el procesador debe transferir los datos resguardados en la pila de la aplicación de USER Mode, a la pila de la aplicación de KERNEL Mode.

Prioridad	Descripción
1 (Maxima)	Reset y Machine check
2	Trap en Conmutacón de tarea. Flag T de TSS = 1
3	Eventos de Hardware Externos -FLUSH -STOPCLK -SMI -INIT
4	Trap en la instrucción previa -Breackpoint, o Debug Trap Exception. -TF=1 o breackpoint de dato o E/S
5	NMI, Non Maskable Interrupt
6	INTR, Interrupciones enmascarables
7	Code Breakpoint Fault
8	Fault en el ciclo de Fectch de la siguiente instrucción. -Violación del límite del segmento de código -Code-Page Fault)
9	Faults en la decodificación de la siguiente instrucción. -Instruction length mayor a 15 bytes -Invalid Opcode -Coprocessor Not Available
10 (Minima)	Faults en la ejecución de una instrucción. -Overflow -Bound error -Invalid TSS -Segment Not Present -Stack fault -General Protection -Data Page Fault -Alignment Check -x87 FPU Floating-point except -SIMD floating-point exception