INTERRUPCIONES Y EXCEPCIONES

14

14.1. CONCEPTOS GENERALES

Las interrupciones y excepciones son acontecimientos externos o internos al procesador, que provocan una desviación en el flujo de control de la CPU. Estas anomalías son asíncronas, ya que no se saben cuando van a suceder.

Las **interrupciones** se originan por acontecimientos externos, como la activación en alguna patita del procesador o bien por la ejecución de alguna de las instrucciones especificas que tiene el procesador para generarlas.

Las **excepciones** se generan automáticamente como consecuencia de algún acontecimiento anormal, producido y detectado en el desarrollo del programa en curso de ejecución.

Para manejar las interrupciones y excepciones la CPU del Pentium dispone de una tabla de interrupciones, llamada **IDT** de 256 entradas, cada una de las cuales atiende a un tipo de interrupción o excepción diferente, y que mediante un mecanismo se especifica la dirección de comienzo de la rutina que atiende la causa que la ha provocado, solucionando de esta manera el problema.

La tabla IDT ocupa un segmento cuya base y límite están contenidos en el Registro de Tablas de Descriptores de Interrupciones **IDTR**, si el Pentium trabaja en Modo Protegido. En Modo Real, la operatividad de dicha tabla es diferente, aunque su objetivo es el mismo.

14.2. TIPOS DE INTERRUPCIONES

Las interrupciones que trata el procesador, pueden ser de dos tipos:

14.2.1. Interrupciones externas

Se trata de interrupciones activadas por componentes hardware externos, que provocan la activación de una de las patitas del procesador Pentium. Esta activación la detecta el Controlador de Interrupciones Programable Avanzado local (APIC) que dispone el procesador Pentium y sucesores

Cuando el APIC local no esta habilitada, las patitas de estas, se configuran como INTR y NMI. Cuando el APIC esta habilitado, las patitas se pueden configurar a través de la tabla de vectores para asociarlo con cualquier vector de interrupción o excepción del procesador.

Hay que tener en cuenta que la activación de otras patitas del procesador, son capaces de provocar una interrupción del proceso de la CPU. Sin embargo estas interrupciones no son manipuladas por el mecanismo de interrupción y excepción descritos por este capitulo. Esas patitas son: RESET#, FLUSH#, STPCLK#, SMI, R/S# e INIT#.

Las patitas que provocan estas interrupciones externas, son la **INTR** y **NMI**.

- **NMI**: es una interrupción NO enmascarable y que por lo tanto es siempre atendida por la CPU. esta interrupción es el resultado de un problema hardware serio, como el error de paridad de la memoria o un error del bus.

  Esta interrupción se activa por flanco ascendente de su patita y tras la su activación, se atiende mediante la entrada numero 2 de la tabla IDT.

- **INTR**: es una interrupción que se origina por la activación de su patita INTR en el procesador, por parte de un componente externo al procesador.

  Se trata de una interrupción enmascarable y su aceptación y puesta en marcha depende del estado del señalizador IF del registro E-FLAGS. Si IF=1, se atiende la interrupción enmascarable, pero si IF=0, no se tiene en cuenta la petición.

  Cuando la CPU acepta la petición de INTR, responde con dos ciclos de reconocimiento de interrupción, similares a los de lectura. Durante dichos ciclos, el componente externo que haya solicitado la interrupción del proceso del procesador, deberá introducir el valor de la entrada de la tabla IDT, que quiere que la interrupción procese. El valor de la entrada de la tabla IDT, vendrá dada por las 8 líneas de menos peso del bus de datos (D0-D7)

14.2.2. Interrupciones internas

Este grupo de interrupciones se origina como consecuencia de la ejecución de alguna instrucción especial, es decir, son interrupciones que se provocan a través del software. Su desarrollo funcional es igual al que corresponde a una excepción.

Las instrucciones que pueden provocar este tipo de interrupciones son **INT n** y **INTO**

- **INT n** es una interrupción NO enmascarable, generada por software. Siempre que se ejecuta esta instrucción, se salta a la rutina de la interrupción que indique el valor 'n' que apunta a la tabla de la IDT.

- **INTO** es una intrusión que salta al vector 4 de la IDT, siempre que el valor del bit OF del registro E-FLAGS valga 1.

14.3. TIPOS DE EXCEPCIONES

Las excepciones son provocadas automáticamente por el procesador al detectar alguna anomalía en el flujo de control.

Los tipos de excepciones son las siguientes:

- **Excepciones faltas o errores**: son aquellas excepciones que se encargan de corregir el error o la falta al intentar ejecutar una instrucción, retornando al lugar donde la CPU la dejo, tras la finalización de la excepción. De esta forma la instrucción que la había provocado, se puede realizar. Un ejemplo de este tipo de excepción es cuando la CPU ejecuta una operación matemática y aun no tiene todos los operandos que están involucrados en la instrucción.

- **Trampa**: son aquellas excepciones que se generan tras la finalización de la instrucción. Un ejemplo de este tipo de excepción son las interrupciones definidas por el usuario e incluidas en el programa.

- **Aborto**: son aquellas excepciones que no permiten la localización exacta de la instrucción que la origino. Se suele usar para indicar errores muy graves, como los que se originan del comportamiento del equipo físico o valores en las tablas que manipulan el sistema.

14.4. TABLA DE DESCRIPTORES DE INTERRUPCIONES (IDT)

En Modo Protegido, el procesador dispone de la Tabla de Descriptores de Interrupciones (IDT), que contiene los descriptores que asocia cada vector de excepción o interrupción con la puerta de descriptores para cada proceso de interrupción o excepción. Esta tabla consta de 256 entradas, de las cuales las veinte primeras (0 - 19) están reservadas exclusivamente para Intel. De las 20 a la 31 están reservados exclusivamente para la CPU y las demás entradas están a disposición del usuario.

La IDT puede ubicarse en cualquier sitio dentro del espacio de direcciones lineales. Ya que el procesador localiza la IDT a través del registro IDTR. Este registro contiene 32 bits para marcar la dirección base y 16 bits para indicar el límite de la IDT.

Para usar la IDTR, el procesador utiliza dos instrucciones la LIDT y SIDT. La instrucción LIDT carga en la IDTR la dirección base y el límite. Esta instrucción puede ser ejecutada únicamente cuando el CPL=0. Este normalmente se emplea para la inicialización del código de un sistema operativo cuando se crea una IDT o también lo puede usar para pasar de una IDT a otra. La instrucción SIDT hace una copia en la memoria de los valores de la base y el límite almacenados en la IDTR. Esta instrucción se puede ejecutar en cualquier nivel de privilegio.

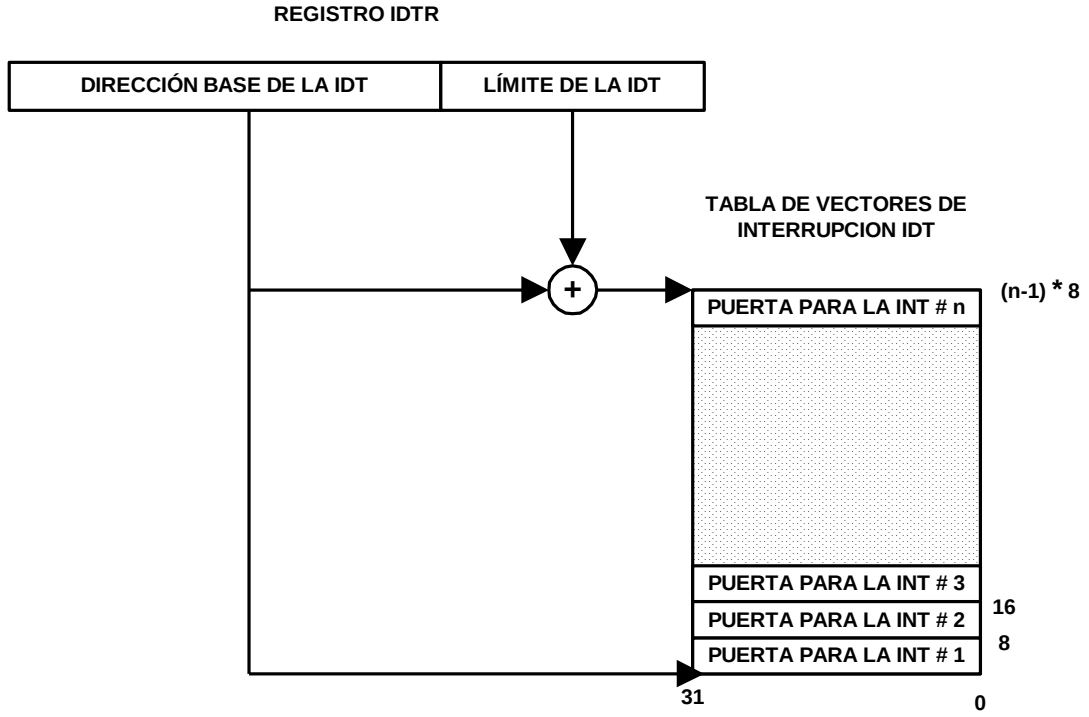

Figura 14.1 – Relaciones entre la IDTR y la IDT

En la tabla 14.2 se explican las entradas de la IDT, dando su vector de interrupción, descripción, instrucciones que puede causar, el código de error y su clase.

VECTOR	DESCRIPCIÓN	INSTRUCCIONES QUE PUEDE CAUSAR	CÓDIGO DE ERROR	CLASE
0	Error de división #DE	DIV IDIV	No	Falta
1	Excepción de depuración #DB	Cualquier código o dato de referencia	No	Falta Trampa
2	Interrupción no enmascarable NMI	Interrupción externa no enmascarable	No	NMI interrupción externa
3	Punto de ruptura #BP	INT 3	No	Trampa
4	Sobrepasamiento #OF	INT 0	No	Trampa
5	Comprobación de limites #BR	BOUND	No	Falta
6	Código OP no valido #UD	UD2 o código OP reservado	No	Falta
7	Coprocesador no disponible #NM	WAIT/FWAIT o coma flotante	No	Falta
8	Doble falta #DF	Instrucciones que origine una excepción, NMI o INTR	Si (cero)	Aborto
9	Sobrepasamiento del segmento por el coprocesador	Instrucciones de coma flotante	No	Falta
10	TSS no valida #TS	Acceso TSS o conmutación de tareas	Si	Falta
11	Segmento no presente #NP	Carga de registros del segmento o acceso a segmentos	Si	Falta
12	Excepción en la pila #SS	Operaciones de pila y carga registros SS	Si	Falta
13	Protección general #GP	Referencias a memoria y comprobación de protección	Si	Falta
14	Fallo de página #PF	Referencia a memoria	Si	Falta
15	Reservas de Intel	----	No	----
16	Error de coma flotante x87 FPU #MF	WAIT/FWAIT o coma flotante	No	Falta
17	Comprobación de alineamiento #AC	Cualquier dato referenciado en memoria	Si	Falta
18	Comprobación de la maquina #MC	Códigos de error y fuentes son modelos dependientes	No	Aborto
19	Excepción de coma flotante SIMD #XF	SSE SSE2	No	Falta
20-31	Reservas por Intel	----	----	----
32-255	Interrupción definidas por el usuario	Interrupción externa o INT n	----	Interrupción

Tabla 14.2 – Entradas de la tabla IDT

A continuación se describe cada vector de interrupción:

Vector 0: Error de división

Este error lo generan las instrucciones **DIV** o **IDIV** cuando tienen el divisor con valor 0 o cuando el cociente es muy grande y no cabe en el operando destino.

Los contenidos guardados en los registros CS y EIP apuntan a la instrucción que ha generado la excepción.

Vector 1: Excepción de depuración

Esta interrupción se produce por causa de que se haya detectado una o más de las condiciones de excepción de depuración. En función de si es una falta o una depuración la condición será diferente, como se puede apreciar en la figura 14.2:

Condiciones de excepción	Clase de condición
Instrucción de Captura del punto de ruptura	Falta
Lectura de datos o escritura del punto de ruptura	Trampa
Lectura de E/S o escritura del punto de ruptura	Trampa
Condición de defecto general	Falta
Paso único	Trampa
Cambio de tarea	Trampa
Ejecución de la instrucción INT 1	Trampa

Tabla 14.3 – Condiciones de excepción de depuración y sus clases de excepción correspondientes.

Si se trata de una falta, los contenidos guardados en los registros CS y EIP apuntan a la instrucción que ha generado la excepción, mientras que si es una depuración, apuntan a la **siguiente** instrucción a la que ha generado la excepción.

Vector 2: NMI

Esta interrupción la provoca la patita de entrada del **NMI** del procesador y es siempre atendida. Esta interrupción provoca la llamada al manipulador de interrupciones de la **NMI**.

Vector 3: Punto de ruptura (#BP)

Esta interrupción salta cuando de ha ejecutado una instrucción de punto de ruptura, **INT 3**, saltando a una rutina de depuración. Además la excepción de punto de ruptura puede también generarse ejecutando la instrucción **INT n** con el valor 3 en 'n'. La acción de esta instrucción (INT n) es ligeramente diferente que la instrucción **INT 3.**

Los contenidos guardados de los registros CS y EIP apuntan a la instrucción siguiente a la instrucción **INT 3**.

Vector 4: Desbordamiento (#OF)

Esta interrupción salta cuando se ejecuta una instrucción **INTO** y el valor de OF del registro E-FLAGS es igual a 1. Hace que se produzca una depuración de desbordamiento.

Este tipo de interrupciones se ejecutan cuando se comprueba directamente el señalizador OF del registro E-FLAGS o mediante la instrucción INTO, tras la ejecución de una instrucción aritmética como la ADD o SUB. La ventaja de utilizar la instrucción INTO, consiste en que si la excepción por desbordamiento es detectada, un manipulador de excepciones puede ser llamado automáticamente para manipular la condición de desbordamiento.

Los contenidos guardados de los registros CS y EIP apuntan a la instrucción siguiente a la instrucción INTO.

Vector 5: Excepción por sobrepasamiento del rango (Bound #BR)

Esta excepción surge cuando se ha intentado leer o escribir fuera de los límites de un segmento. La genera la instrucción BOUND al comprobar que un índice de array con signo no está entre el límite inferior y superior de un array ubicado en memoria.

Los contenidos guardados de los registros CS y EIP apuntan a ka instrucción BOUND que generó la excepción.

Vector 6: Excepción por cogido de operación no valido

Esta excepción surge cuando el procesador:

- Intenta ejecutar un código de operación no valido o reservado.
- Intenta ejecutar un código de operación con tipos de operandos no validos para la instrucción.
- Intenta ejecutar una instrucción MMX, SSE o SS2 en un procesador IA-32 que no soporta esa tecnología o el señalizador EM del registro de control CR0 esta activo.
- Intenta ejecutar una instrucción SSE o SSE2 en el procesador IA-32 que produce una excepción de coma flotante SIMD cuando el bit OSXMMEXCPT del registro de control CR4 esta inactivo o cuando el bit OSFXSR del registro de control CR4 está inactivo.
- Intenta ejecutar una instrucción LLDT, SLDT, LTR, LSL, LAR, VERR, VERW o ARPL mientras se está en Modo Real o en Modo Virtual 8086.
- Intenta ejecutar la instrucción RSM cuando no está en Modo SMM.
- Ha detectado un prefijo LOCK que precede a una instrucción que puede no estar cerrada o una que puede estar cerrada pero que el operando destino no esta localizado en memoria.
- Ha ejecutado la instrucción UD2, que garantiza la generación de un código de operación no valido.

Los contenidos guardados de los registros CS y EIP apuntan a la instrucción que generó la excepción.

Vector 7: Excepción por coprocesador no disponible (#NM)

Esta excepción salta cuando el procesador detecta que no tiene disponible el coprocesador para realizar operaciones con **coma flotante** o **WAIT** / **FWAIT**.

Los contenidos guardados de los registros CS y EIP apuntan a la instrucción de coma flotante o a la instrucción WAIT/FWAIT que generó la excepción.

Vector 8: Excepción por doble fallo (#DF)

Indica que el procesador detecta una excepción secundaria mientras está llamando a un manipulador de excepciones para una excepción anterior. Normalmente, cuando el procesador detecta otra excepción mientras está intentando llamar al manipulador de excepciones, las dos excepciones pueden ser manipuladas seriamente. Si en cambio el procesador no puede manipularlas, señaliza una excepción de doble falta. Para determinar cuando dos faltas necesitan ser señaladas como doble falta, el procesador divide las excepciones en tres clases: excepciones benignas, excepciones contribuyentes y faltas de página.

Clase	Numero de vector	Descripción
Excepciones benignas	1	Depuración
	2	Interrupción NMI
	3	Punto de ruptura.
	4	Desbordamiento
	5	Rango BOUDED excedido
	6	Código OP no valido.
	7	Coprocesador no disponible
	9	Segmento de estado de tarea no valido
	16	Error en coma flotante.
	17	Comprobación de alineamiento.
	18	Comprobación de maquina.
	19	Coma flotante SIMD
	All	INT n
	All	INTR
Excepciones contribuyentes	0	Error de división.
	10	TSS o valida.
	11	Segmento no presente
	12	Falta de pila
	13	Protección gerenal
Falta de páginas	14	Falta de página

Tabla 14.4 – Clases de interrupciones y excepciones.

Cualquier tipo de faltas generadas mientras el procesador intenta transferir el control a un manipulador de faltas adecuado podría dejar secuencias de doble falta.

Primera excepción	Segunda excepción		
	Benigna	**Contribuyente**	**Falta de página**
Benigna	Manipula Excepciones en serie	Manipula Excepciones en serie	Manipula Excepciones en serie
Contribuyente	Manipula Excepciones en serie	Genera una doble falta	Manipula Excepciones en serie
Falta de página	Manipula Excepciones en serie	Genera una doble falta	Genera una doble falta

Tabla 14.5 – Condiciones para generar una doble falta

Los contenidos guardados de los registros CS y EIP están indefinidos.

Vector 9: Sobrepasamiento del segmento por el procesador

Indica que un sistema basado en el Intel386 con procesador matemático del Intel387 detecta la violación de una página o segmento mientras se transfería una parte del operando del coprocesador matemático del Intel387.

Vector 10: Excepción del segmento de estado de tarea no valido

Indica que se ha intentado realizar una conmutación de tareas y que la información no valida es detectada por la TSS para la tarea correspondiente.

Índice del código de error	Condición no valida
Índice del selector del segmento TSS.	El límite del segmento TSS menor que 67H para 32 bits o menor que 2CH para 16 bits.
Índice del selector del segmento LDT.	LDT no valida o LDT no presente.
Índice del selector del segmento de pila.	El selector del segmento de pila excede del límite de la tabla de descriptores.
Índice del selector del segmento de pila.	El segmento de pila no se puede escribir.
Índice del selector del segmento de pila.	El segmento de pila DPL ≠ CPL.
Índice del selector del segmento de pila.	El selector de segmento de pila RPL ≠ CPL.
Índice del selector del segmento de código.	El selector del segmento de código excede del límite de la tabla de descriptores.
Índice del selector del segmento de código.	Segmento de código no es ejecutable.
Índice del selector del segmento de código.	Segmento de código no ajustable.
Índice del selector del segmento de código.	Segmento de código ajustable DPL mejor que CPL.
Índice del selector del segmento de datos.	El selector de segmento de datos excede del límite de la tabla de descriptores.
Índice del selector del segmento de datos.	Segmento de datos no leíble.

Tabla 14.6 – Condiciones TSS no válidas.

Si la condición de excepción se detecta antes del cambio de tarea, los contenidos guardados de los registros CS y EIP apuntan a la instrucción que invocó el cambio de tarea. Si la condición de excepción se detecta después del cambio de tarea, los contenidos guardados de los registros CS y EIP apuntan a la primera instrucción de la nueva tarea.

Vector 11: Segmento no presente

Esta excepción surge cuando el procesador intenta acceder a un descriptor de segmento o una puerta y está inactivo. El procesador puede generar esta excepción durante cualquiera de las siguientes operaciones:

- Mientras se intenta cargar los registros CS, DS, ES, FS o GS. Esta situación puede ocurrir mientras se realiza el cambio de tarea.
- Mientras se intenta cargar el LDTR empleando la instrucción LLDT.
- Mientras se ejecuta la instrucción LTR y la TSS es señalada como no presente.
- Mientras se intenta usar un descriptor de puertas o la TSS es señalada como segmento no presente, pero a pesar de todo es valida.

Los contenidos guardados de los registros CS y EIP normalmente apuntan a la instrucción que ha generado el cambio de tarea. Si la excepción se produce mientras se carga los descriptores del segmento para los selectores del segmento de la nueva TSS, los registros CS y EIP apuntan a la primera instrucción en la nueva tarea. Si la excepción se produce mientras se accede al descriptor de la puerta, los registros CS y EIP apuntan a la instrucción que invoca el acceso.

Vector 12: Excepción por falta de pila

Esta excepción surge cuando hay algún problema al manejar la pila. Estos problemas pueden ser:

- Se detecta una violación de límite durante una operación que se refiere al registro SS. Las provocan las instrucciones orientadas al manejo de la pila como puedes ser POP, PUSH, CALL, RET, IRET, ENTER y LEAVE, así como otras referenciadas a memoria que implícita el empleo del registro SS.

- Se detecta un segmento de pila no presente cuando se intenta cargar el registro SS. Esto puede ser debido al cambio de tarea, la ejecución de la instrucción CALL o un retorno a un nivel de privilegio diferente, la ejecución de la instrucción LSS, MOV o POP al registro SS.
- Es posible la recuperación de esta falta extendiendo el límite del segmento de pila o cargando el segmento de pila perdido en memoria.

Los contenidos guardados de los registros CS y EIP normalmente apuntan a la instrucción que ha generado la excepción. Sin embargo cuando la excepción resulta del intento de carga el segmento de pila no presente durante un cambio de tarea, los contenidos guardados de los registros CS y EIP apuntan a la primera instrucción de la nueva tarea.

Vector 13: Excepción por protección general

Esta excepción surge cuando hay alguna clase de violación de protección, como pueden ser:

- Exceso del segmento límite cuando se accede a los segmentos CS, DS, ES, FS o GS.
- Exceso del segmento límite cuando se hace referencia a la tabla de descriptores.
- Transferencia de la ejecución a un segmento que no es ejecutable.
- Escritura de un segmento de código o de datos que es solo de lectura.
- Lectura de un segmento de solo ejecución.
- Carga del registro SS con un selector de segmento para un segmento de solo lectura.
- Carga de los registros SS, DS, ES, FS o GS con un selector de segmento para un segmento del sistema.
- Carga de los registros DS, ES, FS o GS con un selector de segmento para un segmento de código de sólo ejecución.
- Carga del registro SS con un selector de segmento de un segmento ejecutable o un selector de segmento nulo.
- Carga del registro CS con un selector de segmento para un segmento de datos o un selector de segmento nulo.
- Acceso a memoria empleando los registros DS, ES, FS o GS cuando contiene un selector se segmento nulo.
- Cambia de una tarea ocupada durante una llamada o salto al TSS.
- Cambiar a una tarea no ocupada durante la ejecución de una instrucción IRET.
- Empleo de un selector de segmento en un cambio de tarea que apunta a un descriptor TSS en la LDT actual. Los descriptotes TSS pueden únicamente estar en la GDT.
- Violación de cualquiera de las reglas de privilegio.
- Exceso del límite de la longitud de ka instrucción (15 bytes).
- Carga del registro CR0 con un señalizador de página PG activo y un señalizador PE inactivo.
- Carga del registro CR0 con un señalizador activo y un señalizador CD inactivo.
- Referencia a una entrada en la IDT que no es ni una interrupción, trampa ni puerta de tarea.
- Intento de acceso a un interrupción o excepción a través de una interrupción o puerta de depuración del Modo Virtual 8086 cuando el segmento de código DPL del manipulador es mejor que 0.
- Intento de escritura de '1' en un bit reservado del registro CR4.
- Intento de ejecución de una instrucción privilegiada cuando el CPL no es igual a 0.
- Escritura de un bit reservado en una MSR.
- Acceso a una puerta que contiene un selector de segmento nulo.
- Ejecución de la instrucción INT n cuando el CPL es mejor que el DPL de la interrupción, trampa o puerta de tarea correspondiente.

- El selector de segmento en una llamada, interrupción o puerta de depuración no apunta a un código de segmento.
- El operando del selector del segmento en la instrucción LLDT es un tipo local o no apunta al descriptor del segmento del tipo LDT.
- El operando del selector del segmento en la instrucción LTR es local o apunta a una TSS que no es valida.
- El selector del segmento de código preciso para una llamada, salto o retorno es nulo.
- Si el señalizador PAE y/o PSE del registro de control está activo y el procesador detecta bits reservados en una entrada de la tabla de punteros de directorio de página, se activa a 1. Esos bits son comprobados durante una escritura de los registros de control CR0, CR3 y CR4 que hace que la entrada a la tabla de punteros del directorio de páginas se vuelva a cargar.
- Intento de escritura con un valor que no sea cero en los bit reservados del registro MXCSR.
- Ejecución de la instrucción SE o SSE2 que intenta acceder a una posición de memoria de 128 bits que no está alineada en un límite de 16 bytes cuando la instrucción se requiere un alineamiento de 16 bytes. Esta condición también se aplica al segmento de pila.

Los contenidos guardados de los registros CS y EIP normalmente apuntan a la instrucción que ha generado la excepción.

Vector 14: Excepción de falta de página

Indica que, con la paginación habilitada, el procesador detecta una de las siguientes condiciones mientras se emplea el mecanismo de traslación de página para trasladar la dirección lineal a física:

- El señalizador P (presencia) en un dirección de página o en la entrada de la tabla de página necesitado para la traslación de la dirección, esta puesto a 0, indicando que una tabla de página o la página que contiene el operando no esta presente en memoria física.
- El proceso no tiene suficiente privilegio para acceder a la página indicada.
- El código de ejecución en modo usuario intenta escribir en una página de solo lectura.
- Uno o más bits reservados en la entrada de directorio de páginas son activados a 1.

Vector 16: Error en coma flotante x87 FPU

Indica que el x87 FPU ha detectado un error en coma flotante. El señalizador NE en el registro CR0 debe ser activado por una interrupción 16.

NOTA: las excepciones en coma flotante (#XF) están señaladas a través de la interrupción 19.

Vector 17: Excepción de comprobación de alineamiento

Esta excepción surge cuando el procesador antes de acceder a un segmento, detecta cuando una palabra almacenada en dirección de byte más antiguo, o una doble palabra almacenada en la dirección que no es un entero múltiplo de 4.

Vector 18: Excepción de comprobación de maquina (#MC)

Esta excepción surge cuando el procesador detecta un error interno de la maquina o un error del bus o que un agente externo detecta un error del bus, indicando dicho error por las patitas BIBIT# y MCERR#.

Vector 19: Excepción de coma flotante SIMD (#XF)

Esta excepción salta cuando el procesador detecta algún error en las instrucciones **SSE** o **SSE2** al ejecutar operaciones flotantes SIMD. Las razones que se produzcan la ejecución de esta excepción puede ser alguna de las siguientes:

- Operación invalida (#I)
- Divisor por cero (#Z)
- Operandos desnormalizado (#D)
- Desbordamientos numéricos (#O)
- Desbordamiento inferior numérico (#U)
- Resultado inexacto (precisión) (#P)

Vectores 32-255: Interrupciones definidas por el usuario

Indica que el procesador hace alguna de las siguientes cosas:

- Se ejecuta una instrucción **INT n** donde el operando de la instrucción es uno de los números se los vectores, del 32 al 255.
- Se devuelve la petición de interrupción a la patita INTR o del APIC local cuando el numero del vector de interrupción asociado con la petición es del 32 al 255.

14.5. CÓDIGO DE ERROR

Cuando el procesador esta atendiendo una interrupción o excepción y se detecta una nueva interrupción o excepción, genera un código de error que es apilado en la pila del gestor de la excepción. De esta forma se marca que se ha producido un error y antes de terminar el proceso de la interrupción o excepción con al IRET, se deberá desapilar los errores que se han producido y atenderlos. Después de hacerlo ejecutara la instrucción IRET para volver a la tarea principal.

El formato del código de error es el siguiente:

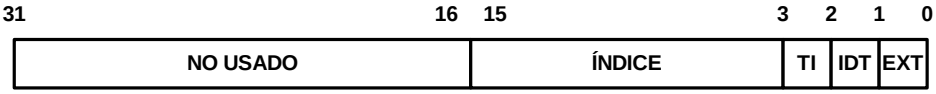

Figura 14.7 – Formato del código de error que se introduce en la pila del gestor de la excepción

EXT es un bit que indica el origen del error. Si vale 1, indica que el error viene desde el exterior y por lo tanto es una interrupción. Si por lo contrario es 0, indica que el procesador ha detectado una anomalía y por lo tanto se trata de una excepción.

IDT es el bit que indica que el error se ha producido en la tabla IDT, si vale 1 y en caso de que valga 0, el error se ha producido en la GDT o LDT.

TI es el bit que indica si el error proviene de la GDT, si tiene valor '0' o de la LDT si tiene valor '1'.

ÍNDICE, indica el selector donde se produjo el error.

14.6. INTERRUPCIONES Y EXCEPCIONES EN MODO REAL

Los Pentium como los demás procesadores de Intel, cuando arrancan por primera vez, trabajan en Modo Real y el sistema operativo debe tener ya en la memoria principal la tabla IDT, para funcionar. En la inicialización o Reset del Pentium, hay que ubicar la tabla de vectores de interrupción en el mismo sitio que el 8086 y para ello se carga la base del IDTR con el valor 0000 0000H.

Como en Modo Protegido, la IDT del Modo Real dispone de 256 entradas para las interrupciones. La diferencia entre el como protegido y el Modo Real, reside en el tipo de entras que dispone cada uno. En Modo Real, la tabla contiene descriptores, que son de 64 bits. En cambio las entradas en Modo Real son de 32 bits cada una. En cada entrada de la IDT, se encuentra la dirección del segmento de código al que a de saltar para ejecutar la interrupción, dado por el registro CS y el desplazamiento dentro de ese segmento de código, que viene dado por el registro IP.

Al ser las entradas de la IDT de 32 bits, el tamaño máximo que puede tener la tabla de interrupciones y excepciones es de un 1KB (256 entradas x 4bytes/entrada).

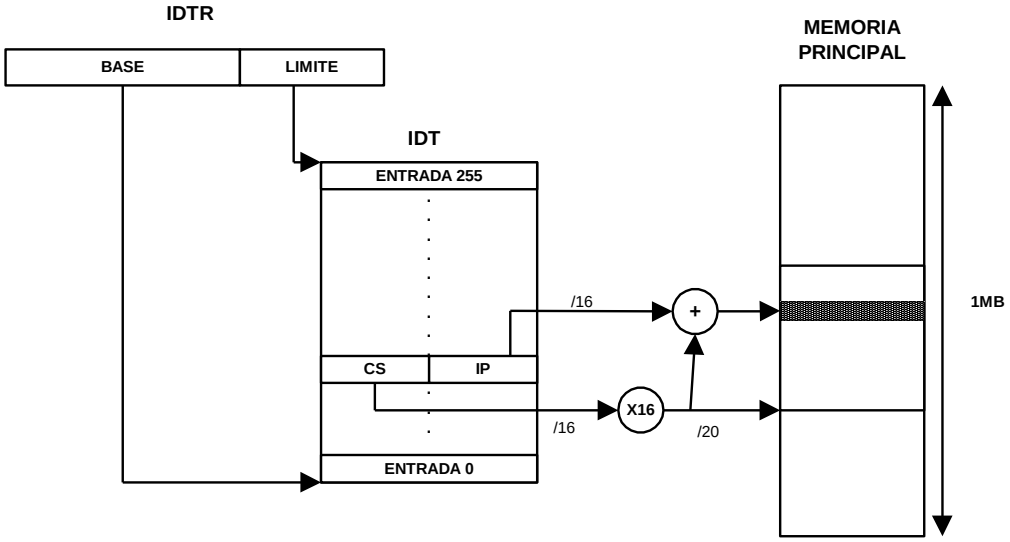

Figura 14.8 – Acceso a las interrupciones y excepciones en Modo Real.

En Modo Real y en Modo Protegido, además de las diferencias la tabla IDT, tienen algunas diferencias mas. En Modo Real dispone menos vectores de interrupción ya que no dispone de los mecanismos de protección y funcionamiento de la memoria virtual. Los vectores comunes son los siguientes:

Función	Numero de vector
Error de división.	0
Interrupción paso a paso.	1
Interrupción NMI.	2
Interrupción de puno de parada.	3
Excepción por sobrepasamiento.	4
Sobrepasamiento de los límites.	5
Código OP invalido.	6
Coprocesador no disponible.	7
Reservadas (Modo Protegido)	8 – 15
Error de coprocesador.	16
Reservadas INTEL	17 – 31
Definidas por el usuario	32 – 255

Tabla 14.9 – Relación de vectores comunes entre el Modo Real y Modo Protegido.

14.7. INTERRUPCIONES Y EXCEPCIONES EN MODO PROTEGIDO

Los Pentium como los demás procesadores de Intel, cuando arrancan por primera vez, trabajan en Modo Real. Para pasar a Modo Protegido solo hay que cambiar el valor del bit PE de registro de estados CR0 a 1. Pero antes el sistema operativo debe haber creado la tabla IDT en memoria principal nada mas empezar el Modo Protegido.

Esta tabla IDT dispone de 256 entradas para el tratamiento de las posibles interrupciones. Sin embargo a diferencia del Modo Real, la tabla está cargada por descriptores de puertas, que ocupan 8 Bytes cada uno, haciendo que el tamaño máximo de la IDT sea de 2K

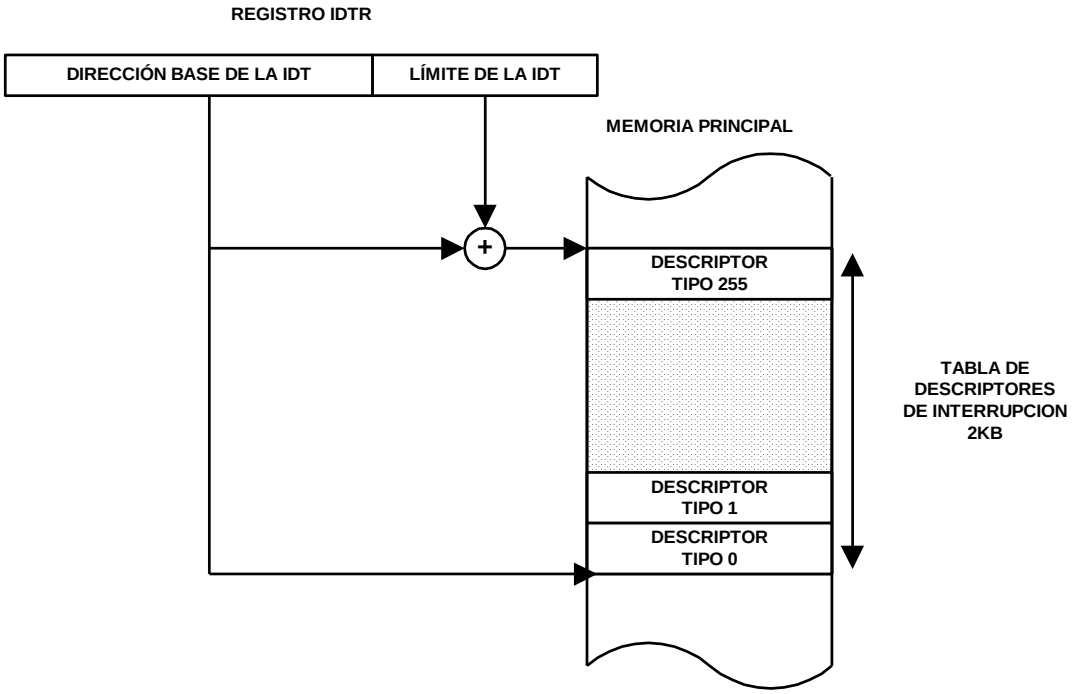

Figura 14.10 – Acceso a las interrupciones y excepciones en Modo Protegido

Los descriptores de la IDT responden a tres tipos de puertas, que son los siguientes:

- **Puertas de Tarea:** en la entrada de la IDT puedes encontrar puertas de tareas (PT) y eso quiere decir que la rutina donde esta localizada la interrupción está en otra tarea distinta a la tarea en curso. Esto tiene la ventaja de que hay una independencia entre tareas, pero como inconveniente hay que decir que una conmutación de tarea consume mucho tiempo en realizarla.

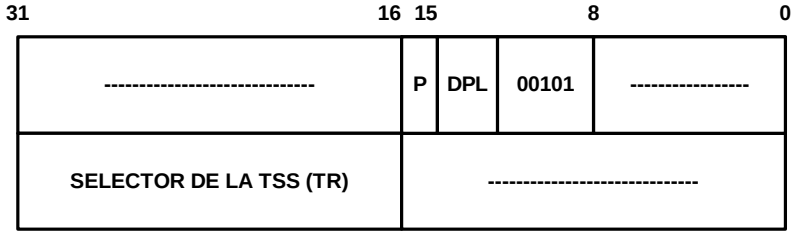

Figura 14.11 – Puerta de Tarea.

Al producirse la interrupción, se direcciona una entrada de la IDT, en la que reside el descriptor de una puerta de tarea que la va a atender. Este tipo de descriptor da lugar a una conmutación de tarea, que implica un total aislamiento de la nueva tarea en curso con respecto a la tarea que estaba ejecutando el procesador antes de la interrupción, dejando el señalizador NT con valor a 1. La tarea de la interrupción finalizará con la ejecución de la instrucción IRET, que devolverá el control a la tarea previa.

Si durante la ejecución de la tarea de la interrupción, se producen nuevas excepciones, se introducirán en la pila de la tarea de la interrupción y antes de terminar la tarea de la interrupción hay que eliminar las excepciones apiladas de la tarea.

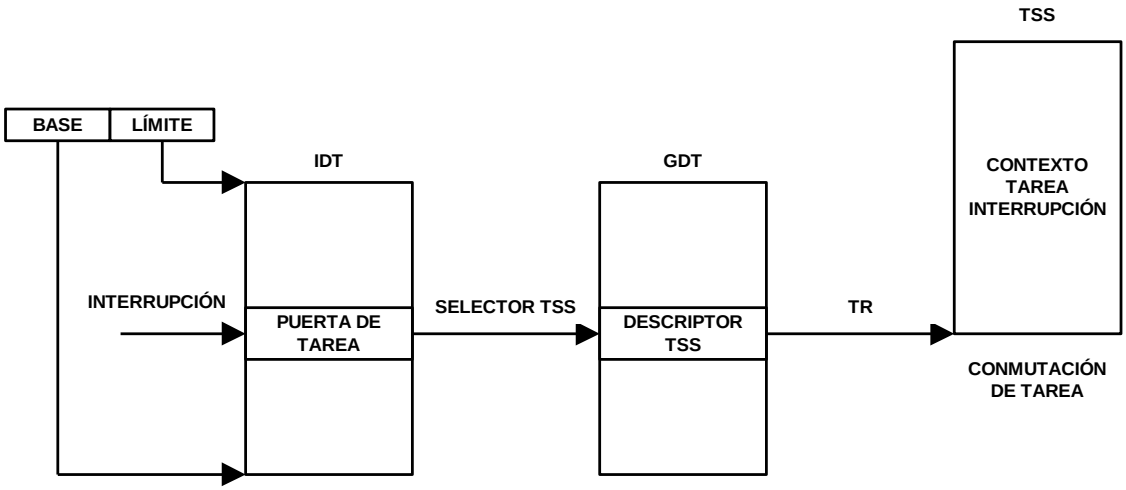

Figura 14.12 – Mecanismo de funcionamiento de una puerta de tarea.

- **Puertas de Interrupciones y Excepciones:** actúan de la misma forma que las puertas de llamadas. Estas puertas no hacen una conmutación de tareas, sino que lo único que se hace es cambiar de segmento de código dentro de una tarea.

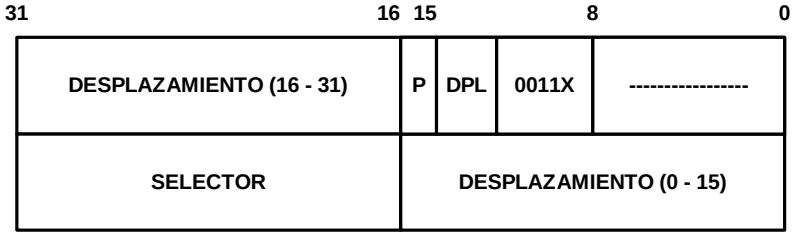

Figura 14.13 – Puerta de Interrupción o Excepción.

El bit '**X**' representa el tipo de puerta. Si adquiere el valor 1, indica que la puerta es del tipo excepción. Esto conlleva a poner el bit IF con valor 1, permitiendo que la subrutina a la que accede pueda ser parada por otra interrupción, como por un dispositivo de entrada y salida. Si en cambio el bit 'X' adquiere el valor 0, indica de que se trata de una interrupción y por lo tanto pone el bit IF con valor 0.

Al producirse la interrupción o excepción, se direcciona una entrada de la IDT, en la que reside el descriptor de una puerta de interrupción o de excepción que la va a atender. Este tipo de descriptor funciona como la puerta de llamada, atendiendo a los periféricos externos y usando los mecanismos de protección.

La puerta realiza una indirección dentro del espacio virtual de la tarea, guardando en la pila las direcciones de retorno (CS y EIP), los valores SS y ESP en caso de menor nivel de privilegio y el registro E-FLAGS. Una vez salvados estos datos, si la puerta es de interrupción, los señalizadotes TF e IF, se pondrán a cero, prohibiendo la aceptación de nuevas interrupciones enmascarables, hasta que se complete la que se halla en curso de procesamiento. Si la puerta es de excepción, el señalizador IF no se pone a 0, para poder atender a los periféricos.

Con la instrucción IRET, ultima de la rutina de servicio, se recupera CS, EIP, SS, ESP y el registros E-FLAGS, para continuar con lo que el procesador estaba haciendo antes de la interrupción o excepción.

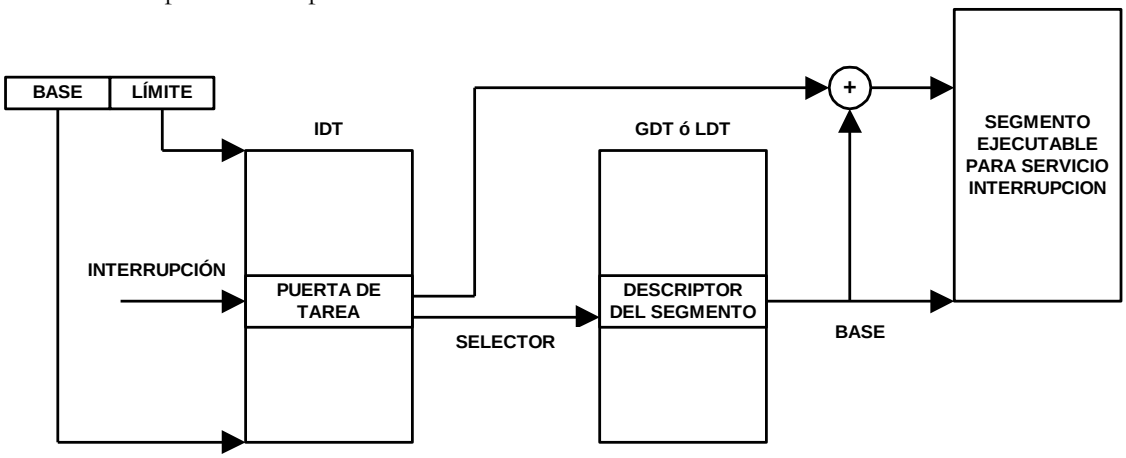

Figura 14.14 – Mecanismo de funcionamiento de una puerta de interrupción o excepción.

14.8. REGLAS DE ATENCIÓN DE UNA INTERRUPCIÓN O EXCEPCIÓN

Cuando se produce una interrupción o una excepción. Los pasos que hace la CPU para tratarla son los siguientes:

- Analizar si hay más de una interrupción pendiente. En caso de haberla, se selecciona aquella solicitud con mayor prioridad. Las prioridades son las siguientes:

Prioridad	Descripciones
1 (Mayor prioridad)	Reset del hardware y comprobación de la maquina. • RESET • Comprobación de la maquina.
2	Depuración en el cambio de tarea. • Se activa el señalizador T en TSS.
3	Intervenciones de hardware externos. • FLUSH • STOPCLK • SMI • INIT
4	Depuración de la instrucción previa. • Punto de ruptura. • Eliminación de fallos en la depuración de excepciones.
5	Interrupciones externas. • Interrupciones NMI. • Interrupciones hardware enmascarables.

6	Faltas procedentes de la captura de las instrucciones siguientes • Falta del código de punto de ruptura. • Violación límite del código de segmento. • Falta de código de página.
7	Falta procedente de la codificación de las instrucciones siguientes: • Longitud de instrucción > 15 bytes. • Código OP ilegal. • Coprocesador no valido.
8 (Menor prioridad)	Faltas en la ejecución de una instrucción • Desbordamiento. • Error bound • TSS invalida • Segmento no presente • Falta de la pila • Protección general • Falta de la página de datos • Comprobación del alineamiento. • Excepción de coma flotante x87 FPU • Excepción de coma flotante SIMD

Tabla 14.15 – Prioridad entre interrupciones y excepciones simultaneas.

- Se salva en la pila el contenido del CS, IP y el registro de estado E-FLAGS. Además se pone a 0 los bits TF e IF.

- Busca el vector predefinido, mirando en la IDT. Si no viene predefinido, viene dado por la instrucción INTR que esta en los bits de menos peso del bus de datos (D0 - D7).

- Al finalizar la rutina de la interrupción, con la instrucción IRET, se saca de la pila los datos antes salvados, CS, IP y E-FLAGS para continuar con lo que antes estaba haciendo le CPU.