MECANISMOS DE PROTECCIÓN 10

10.1. ASPECTOS GENERALES

10.1.1 Necesidad de protección

Cuando se trabaja en Modo Real, el Pentium atiende a una sola tarea y el programa de instrucciones, confeccionado por el diseñador del software, se encarga de la problemática originada por el desarrollo de la aplicación.

En cambio, en Modo Protegido, el procesador atiende a varias tareas simultáneamente. El Pentium dispone de un hardware auxiliar, integrado en el chip, que se encarga de comprobar el cumplimiento de unas reglas que conforman el llamado mecanismo de protección y así permitir la ejecución de todas las tareas sin interferencias.

El objetivo fundamental del mecanismo de protección es caracterizar y defender las funciones vitales del sistema de explotación ante las posibles intrusiones procedentes de las aplicaciones, pero sin excluir una comunicación controlada.

En la construcción de un sistema lógico existen programas que son notablemente seguros por estar muy experimentados y probados; tienen un nivel de seguridad muy alto es decir, un elevado nivel de privilegio (PL). También existen programas, como los de aplicación del usuario, que son poco fiables al estar en la fase de evaluación y depuración; consecuentemente su nivel de privilegio es bajo.

Se debe evitar que programas poco seguros accedan a los datos o al código situado en niveles de privilegio superiores, porque podrían ocasionar la corrupción de estos últimos rebajando su nivel de privilegio al del programa que les accedió.

El mecanismo de protección hardware persigue la integridad del sistema, y las funciones vitales, sobre todo las intrusiones no permitidas y las comunicaciones entre las tareas.

Cuando el mecanismo de protección detecta una violación de las reglas escritas en el silicio, genera una excepción, deteniendo el procesamiento normal de la CPU. Como podría suceder al ejecutar una instrucción de división en la que el valor del divisor sea cero.

El núcleo del S.O. posee el grado de seguridad máximo, también son muy seguros los módulos de los servicios del S.O. Por el contrario, son poco fiables los programas de aplicación que tienen un notable riesgo de generar errores y fallos, además existen programas que necesitan de un nivel de seguridad intermedio.

Como se muestra en la figura 10.1, en el caso de la segmentación, se distinguen cuatro niveles de privilegio, numerados del 0 al 3 (el 0 el más privilegiado o de nivel jerárquico superior). Hay que tener en cuenta que el nivel numérico es inverso al jerárquico. Por ejemplo, un segmento con PL =2 es jerárquicamente mayor que otro con PL = 3, pero numéricamente es menor. Al comparar niveles de privilegio en la segmentación, se debe precisar si se hace referencia a su valor numérico o al jerárquico.

En el caso de la paginación, solamente se distinguen dos niveles de privilegio: el del supervisor, que es el que proporciona la máxima seguridad, y el del usuario, en el que el grado de seguridad es mínimo.

En la siguiente tabla se refleja la relación entre los niveles de protección en la segmentación y en la paginación:

SEGMENTACIÓN	PAGINACIÓN
PL = 0 Segmentos y programas del núcleo del S.O. : Seguridad alta	**Supervisor** Nivel máximo
PL = 1 Programas que necesitan seguridad media	
PL = 2 Programas que necsitan seguriadad media	
PL = 3 Programas de usuario : Seguridad baja.	**Usuario** Nivel mínimo

Figura 10.1 -- Niveles de protección en la segmentación y en la paginación.

La Unidad de Segmentación evalúa el cumplimiento de las reglas de acceso y manejo de los segmentos en primer lugar, y si está habilitada la paginación, son examinadas las reglas que afectan a las páginas en la Unidad de Paginación seguidamente.

En la figura 10.2, se incluye un ejemplo de un sistema que utiliza los cuatro niveles de privilegio. Maneja dos tareas, un espacio global compartido, existiendo segmentos de datos y de código en todos los niveles de privilegio. Hay que tener en cuenta que, siempre que exista un segmento de código debe haber uno de pila.

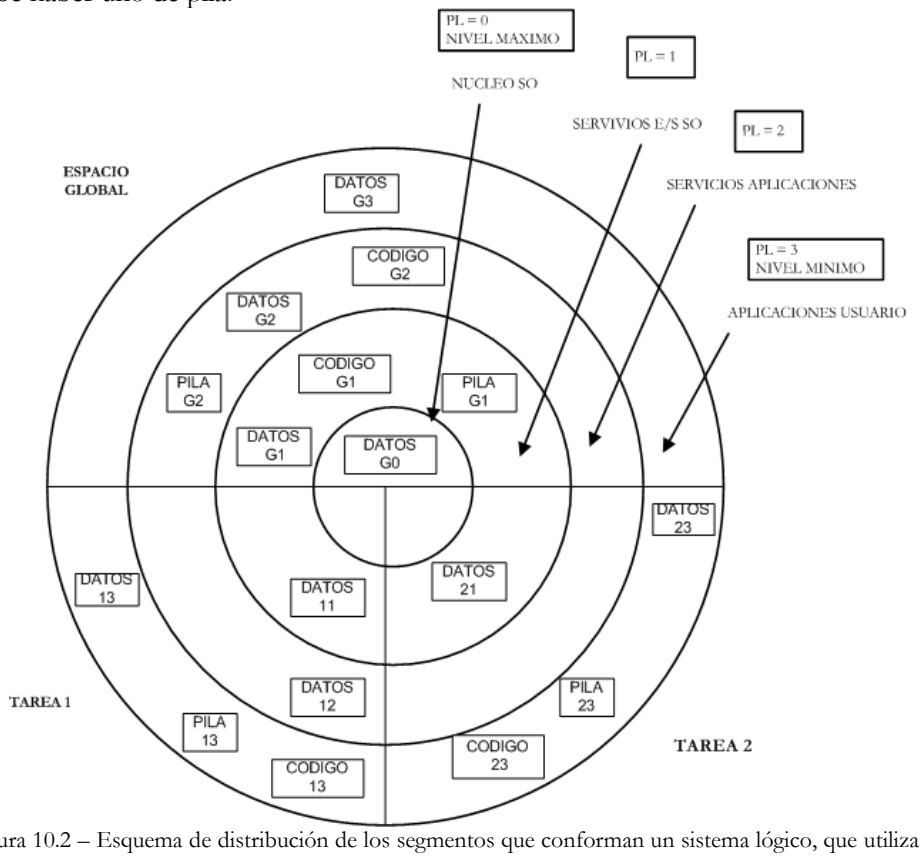

Figura 10.2 – Esquema de distribución de los segmentos que conforman un sistema lógico, que utiliza los cuatro niveles de privilegio.

En la figura 10.3, puede observarse que todos los segmentos de las tareas tienen asignado un nivel de privilegio. El campo DPL del descriptor que referencia al segmento, es el que indica dicho nivel de privilegio.

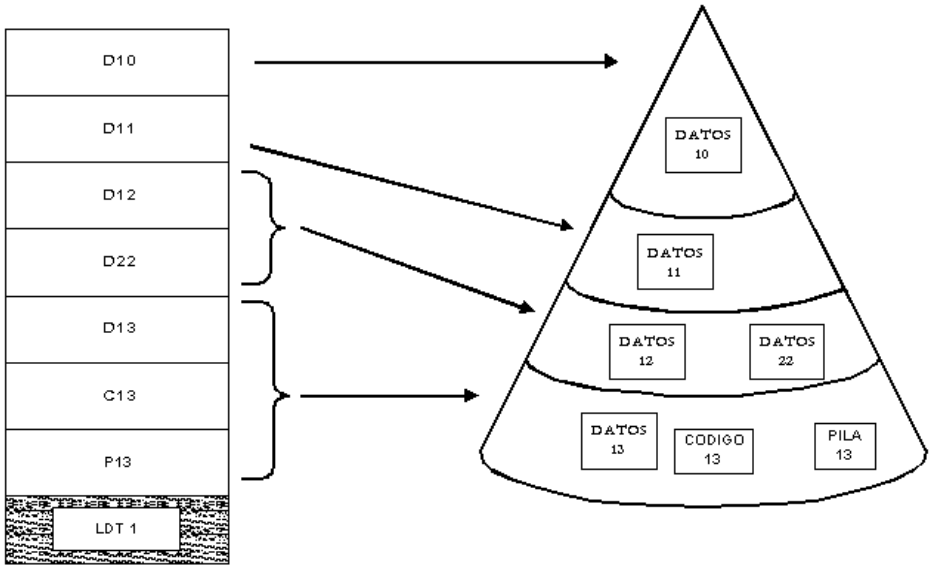

Figura 10.3 -- Cada segmento de tarea tiene asignado un nivel de privilegio, indicado en el campo DPL del descriptor que referencia.

El mecanismo ideal de protección es el proporcionado conjuntamente por la segmentación y la paginación: segmentación para la protección y paginación para manejar la memoria.

Como resultado de esta combinación ciertas páginas pueden ser escritas (las páginas son una parte del segmento) y otras leídas.

El mecanismo de protección del Pentium cubre los siguientes niveles:

1. Protección de tareas
2. Protección de los segmentos
3. Protección de las páginas
4. Protección de las instrucciones

10.1.2. Activación y desactivación de la protección de segmentos y páginas

Para que el procesador cambie a Modo Protegido y éste active el mecanismo de protección de segmentos, se debe activar el flag EP (enable protection) del registro de control CR0.

No existe ningún bit de control para desactivar este mecanismo de protección estando en Modo Protegido, pero asignando el nivel de protección 0 (mayor privilegio) a todos los segmentos y descriptores de segmento se pueden desactivar las reglas de protección de segmentos basadas en los niveles de privilegio. Así se deshabilitan las barreras de protección del nivel de privilegio entre los segmentos, pero todavía siguen activas las reglas de protección del límite y del acceso permitido a lectura/escritura.

Al activar la paginación, poniendo el flag PG del registro CR0 a 1, se activa automáticamente la protección en la paginación.

Si se quiere desactivar el mecanismo de protección en la paginación, se debe poner el flag WP del registro CR0 a 0. Posteriormente, se ponen a 1 los flags R/W (read/write) y U/S (usuario/supervisor) en el Directorio de Páginas y en la Tabla de Páginas. Así las páginas pasan a ser escribibles y pertenecientes al nivel Supervisor, quedando desactivado el mecanismo de protección en la paginación.

10.1.3. Campos y flags usados para la protección entre segmentos y páginas

El mecanismo de protección debe tener en cuenta ciertos campos y señalizadores para controlar el acceso a las páginas y segmentos.

A continuación, se describen los campos necesarios para realizar dicho control.

1. **Flag tipo de segmento (S)**: bit 12 en la segunda doble palabra de un descriptor de segmento. Determina si el descriptor de segmento describe un segmento del sistema o un segmento normal de código o datos.
2. **Campo TIPO**: bits del 8 al 11 en la segunda doble palabra de un descriptor de segmento. Determina si el segmento es de código, de datos o del sistema.
3. **Campo del límite**: bits del 0 al 15 de la primera doble palabra y bits del 16 al 19 de la segunda doble palabra del descriptor de segmento. Determina el tamaño del segmento, estando relacionado con el flag G y el flag E (para segmentos de datos).
4. **Flag G**: bit 32 en la segunda doble palabra del descriptor de segmento. Define el tamaño del segmento, estando relacionado con el límite del segmento y el flag E (para segmentos de datos).
5. **Flag E**: bit 10 en la segunda doble palabra del descriptor de segmento. Define el tamaño del segmento, estando relacionado con el límite del segmento y el flag G.
   Si está a 1 será un segmento de código (ejecutable o de sólo lectura), sin embargo si está a 0 se trata de un segmento de datos.
   Junto con el bit G se puede saber de que tamaño se trata si de 4MB ó 4KB y con el campo límite se observa cual es el límite de dicho segmento.
6. **Campo descriptor del nivel de privilegio (DPL)**: bits 13 y 14 en la segunda doble palabra del descriptor de segmento. Determina el nivel de privilegio del segmento.
7. **Campo del nivel de privilegio del peticionario (RPL)**: bits 0 y 1 de cualquier selector de segmento.
8. **Campo del nivel de privilegio de la tarea en curso (CPL)**: bits 0 y 1 del registro de segmento CS. Se refiere al procedimiento o programa que se está ejecutando en ese momento.
9. **Bit de Tamaño (SIZ)**: Bit 7 de una entrada directa del Directorio de Páginas o de la Tabla de Páginas. Indica si se trata de páginas de 4KB si es 0 ó sin embargo si se tarta de páginas de 4MB si es 1. No se debe olvidar que el Pentium acepta páginas de 4MB.
10. **Bit Sucio (D):** Bit 6 de una entrada directa del Directorio de Páginas o de la Tabla de Páginas. Si D=1 se ha escrito a la página. Sirve para avisar al Sistema Operativo que antes de machacar la página hay que actualizarla en la Memoria Virtual.

11. **Bit A (Accedido):** Bit 5 de una entrada directa del Directorio de Páginas o de la Tabla de Páginas. El Sistema Operativo pone un contador para cada página. Cada vez que se accede a esa página el bit A se pone automáticamente a 1. El Sistema Operativo cada cierto tiempo mira los bits A incrementando los respectivos contadores. Al final cuando se llena la memoria con páginas el S.O sustituye el que menos tiene en su contador.
12. **Bit PCD:** Bit 4 de una entrada directa del Directorio de Páginas o de la Tabla de Páginas. Indica si está a 1 que se trata e una página cacheable (se puede meter en la caché).
13. **Bit PWT:** Bit 3 de una entrada directa del Directorio de Páginas o de la Tabla de Páginas. Indica si está a 1 que la página es de escritura obligada y cacheable.
14. **Usuario/Supervisor (U/S)**: bit 2 de una entrada directa del Directorio de Páginas o de la Tabla de Páginas. Indica el tipo de página: Usuario o Supervisor.
15. **Lectura/escritura (R/W)**: bit 1 de una entrada del Directorio de Páginas o de la Tabla de Páginas. Indica el tipo de acceso permitido a esta página: sólo lectura o lectura y escritura.
16. **Bit de Presencia (P)**: Bit 0 de una entrada directa del Directorio de Páginas o de la Tabla de Páginas. Indica si está a 1 que la página está presente en la memoria principal y si está a 0 que no lo está.

En las figuras 10.4 y 10.5 se muestran los campos de un Descriptor de Segmento y de una entrada de la Tabla de Páginas, respectivamente.

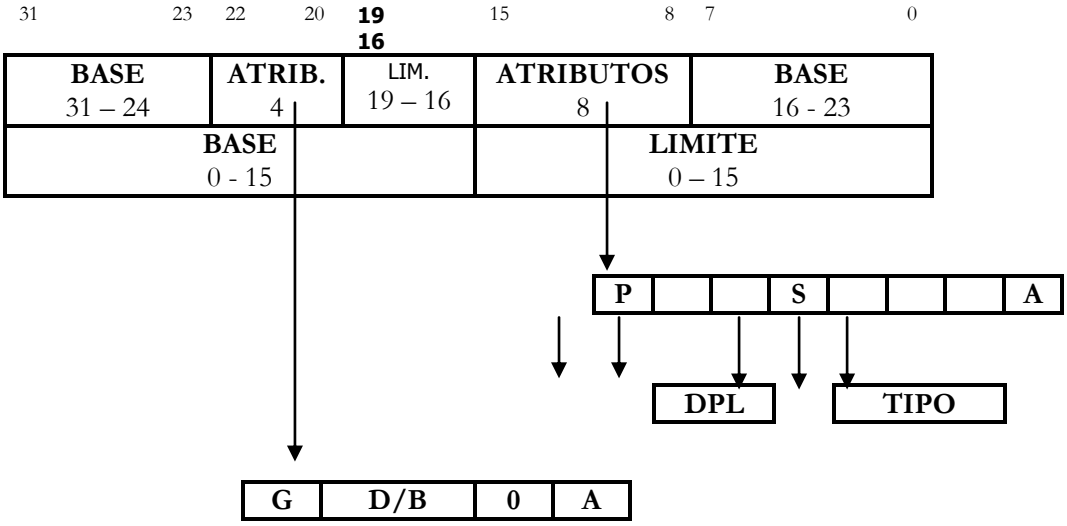

Figura 10.4 Estructura del descriptor de segmento

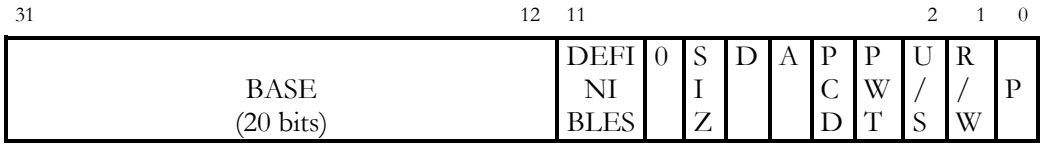

Figura 10.5 Formato de las entradas a las Tablas de Páginas

10.2. NIVELES DE PROTECCIÓN

10.2.1. Protección entre tareas

Los descriptores de segmento de la GDT, en todo momento, hacen referencia a objetos del espacio global (comunes a todas las tareas). Y los descriptores de segmento de la LDT, hacen referencia a los segmentos de la tarea en curso.

Nunca es posible que, ejecutando una tarea se realice un acceso prohibido a un segmento de otra tarea, ya que a la CPU sólo se le permite el acceso a los descriptores de la LDT activa, ésta de forma indirecta, es apuntada por el registro LDTR.

Lo que sí es posible, es acceder desde una tarea a segmentos referenciados por los descriptores de la GDT.

Para ejecutar una tarea nueva, se cambia el contenido del registro de tabla LDTR y así se referencia a la LDT de la tarea nueva y se activan los descriptores de los segmentos de la nueva tarea.

10.2.2. Protección de segmentos

Cuando se carga el contenido de un descriptor en el registro caché ultrarrápido asociado a un registro de segmento, se almacena la base (32 bits), el límite (20 bits) y los atributos (12 bits).

Cada vez que se accede a memoria, la Unidad de Segmentación es la encargada de comprobar las normas de protección y en caso de fallo, se genera una excepción.

10.2.2.1. Protección del límite

Para llevar a cabo la comprobación del límite de los segmentos se debe tener en cuenta el flag de granularidad (G). Para segmentos de datos, también depende del flag ED (dirección de expansión descendente) y del B (tamaño por defecto del puntero a pila).

Si G = 0 (granularidad de byte), el límite efectivo, medido en bytes, viene marcado por los 20 bits del límite indicado en el descriptor de segmentos. En este caso el límite está situado en el rango de 0 a 1MB (FFFFF H). En cambio, si G = 1 (granularidad de páginas de 4KB), el procesador escala el valor del campo límite con un factor de 2^{12}, el rango límite efectivo se encuentra comprendido entre 4KB (FFF H) y 4GB (FFFFFFFF H). En el caso de G = 1, los 12 bits menos significativos del desplazamiento no se comprueban con respecto al límite.

Para todos los segmentos (menos para los segmentos de datos de expansión decreciente), la última dirección a la que se puede acceder (límite efectivo), es el tamaño del segmento (en bytes) menos 1. Cada vez que se intente acceder a las siguientes direcciones en un segmento, el procesador genera una excepción de protección general:

1. Un byte más arriba del límite efectivo
2. Una palabra más arriba del límite efectivo (-1)
3. Una doble palabra más arriba del límite efectivo (-3)

4. Una cuádruple palabra más arriba del límite efectivo (-7)

Para segmentos de Pila (segmentos de datos de expansión decreciente), el límite depende del valor de B. Si B=0, el rango va de (límite efectivo + 1) a 4KB. Si B=1, el rango válido va de (límite efectivo + 1) a 4GB. Cuando el límite es 0, un segmento de expansión decreciente tiene el máximo tamaño.

Esta comprobación genera errores, que sino se detectan se puede llegar a sobrescribir el código o datos de otro segmento.

El procesador también debe comprobar los límites de las tablas de descriptores. Para evitar que los programas seleccionen un descriptor de segmento de fuera de sus tablas de descriptores, se usan los valores de los registros GDTR e IDTR (16 bits). Para evitar que se acceda fuera de las actuales LDT y TSS, se utilizan los valores del límite de segmento (32 bits) de los registros LDTR y los de tareas TR.

10.2.2.2. Protección del tipo

Para evitar el uso de un segmento o puerta de forma incorrecta se tiene en cuenta el tipo de los descriptores de segmento que viene determinado por el campo tipo y el flag S, tal y como puede observarse en la figura 10.6.

<table>
<tr><td rowspan="2">S = 1</td><td>E = 1</td><td>C</td><td>R</td><td rowspan="2">A</td></tr>
<tr><td>E = 0</td><td>ED</td><td>W</td></tr>
</table>

Figura 10.6. Formato del campo TIPO

Si S = 1, se trata de un segmento normal de código, datos o pila creado por el programa. Si S = 0, es un segmento especial creado por el programador de sistemas.

Si el bit E = 1, el segmento es ejecutable o de código. En este caso los bits restantes que componen el campo tipo son C, que indica si es ajustable (1) o no (0), y el bit R, que indica si el segmento es leíble (1) o no (0).

En caso de que E = 0, se trata de un segmento de datos no ejecutable, siendo los bits ED y W los que forman el campo tipo. ED indica si el segmento tiene o no expansión de direcciones decreciente. Si ED = 1 se trata de un segmento de pila, en caso contrario es un segmento de datos. El flan W indica si el segmento es escribible.

En cada acceso, se comprueban todas las características del segmento relativas al campo Tipo. Las diferentes excepciones generadas por el mecanismo de protección, son las siguientes:

1. Si se intenta escribir en un segmento de código (E=1)
2. Si se intenta leer un segmento de código con R=0 (prohibición de lectura)
3. Si se intenta cargar CS (registro de segmento de código) con el valor de un selector que corresponda a un descriptor con E=0. Se estaría intentando ejecutar un segmento de datos
4. Si se intenta escribir un segmento de datos con W=0 (prohibición de escritura)
5. Si se intenta cargar SS (registro del segmento de pila) con el valor de un selector, cuyo descriptor asociado esté definido como no escribible (W=0).

La información es examinada por el procesador en varias ocasiones según se opere con segmentos selectores o segmentos descriptores. A continuación se muestran algunos ejemplos de operaciones típicas en las que se comprueba el campo TIPO:

1. Cuando un segmento selector es cargado en un registro de segmento: ciertos registros de segmento sólo pueden contener ciertos tipos de descriptores, por ejemplo:

- El registro CS solo se puede cargar con el selector de un segmento de código.
- Selectores de segmento de segmentos de código que no son leíbles o de segmentos del sistema no pueden ser cargados en registros de segmentos de datos (DS, ES, FS, GS).
- Sólo selectores de segmentos de datos escribibles pueden ser cargados en registros de pila (SS).

2. Cuando un selector de segmento es cargado en LDTR o un registro de tareas:

- En LDTR sólo se puede cargar el selector de una LDT.
- En el registro de tareas TR sólo se puede cargar un selector de segmento de un TSS.

3. Cuando instrucciones acceden a segmentos cuyos descriptores ya han sido cargados en registros de segmento. Ciertos segmentos pueden ser utilizados por instrucciones sólo en ciertas predefinidas formas, por ejemplo:

- Ninguna instrucción debería escribir en un segmento ejecutable
- Ninguna instrucción debería escribir en un segmento de datos si éste no es escribible.
- Ninguna instrucción debería leer un segmento ejecutable a menos que el flag de permiso de lectura esté a 1.

4. Cuando un operando de una instrucción contiene un selector de segmento: ciertas instrucciones pueden acceder a segmentos o puertas de tan solo un tipo en particular. Por ejemplo:

- Una instrucción JMP o CALL sólo puede acceder a un descriptor de segmento de un segmento de código conforming , no conforming, puerta de llamada, puerta de tareas o TSS.
- La instrucción LLDT debe referenciar a un descriptor de segmento de una LDT
- La instrucción LTR debe referenciar a un descriptor de segmento de una TSS
- La instrucción LAR debe referenciar a un descriptor de segmento o puerta de una LDT, TSS, puerta de llamada, CS o segmento de datos.
- La instrucción LST debe referenciar a un descriptor de segmento de una LDT, TSS, segmento de código o segmento de datos.
- Las entradas IDT deben ser interrupciones, excepciones o puertas de llamada.

5. Durante una determinada operación interna. Por ejemplo:

- En una llamada (CALL) o salto (JMP) lejano, el procesador determina el tipo de transferencia de control que debe llevarse a cabo (call o jump a otro segmento de código, a través de una puerta o un cambio de tareas) comprobando el campo TIPO en el descriptor de segmento (o puerta) apuntado por el selector de segmento (o puerta) dado como operando en la CALL o JMP. Si el descriptor es para un CS o puerta de llamada, la llamada o salto a otro CS está indicado. Si el descriptor es para una TSS o puerta de tarea, es indicado un cambio de tareas.
- En una llamada o salto a través de una puerta de llamada , el procesador comprueba automáticamente que el descriptor de segmento al que apunta por medio de la puerta es un segmento de código.
- En una llamada o salto a una nueva tarea a través de la puerta de tareas, el procesador comprueba automáticamente que el descriptor de segmento apuntado por la puerta de rateas es una TSS.
- En una llamada o salto a una nueva tarea mediante una referencia directa a una TSS, el procesador comprueba automáticamente que el descriptor de segmento al que se apunta por medio de la instrucción CALL o JUMP es una TSS.
- Volviendo de una tarea anidada (iniciada por una instrucción IRET),el procesador comprueba que el campo de unión de la tarea precedente en la actual TSS apunta a una TSS.

Para los segmentos del sistema en los que el bit S = 0, los otros cuatro bits del campo tipo definen la clase de segmento de que se trata, de acuerdo con la siguiente relación de códigos:

CÓDIGO	TIPO
0	No definido
1	TSS disponible del 80286
2	LDT
3	TSS ocupado del 80286
4	Puerta de llamada del 80286
5	Puerta de tarea
6	Puerta de interrupción del 80286
7	Puerta de excepción del 80286
8	No definido
9	TSS disponible del 386
A	No definido
B	TSS ocupado del 386
C	Puerta de llamada del 386
D	No definido
E	Puerta de interrupción del 386
F	Puerta de excepción del 386

Figura 10.7. Los códigos del campo tipo cuando S es 1

Comprobación del selector del segmento nulo:

Si se intenta cargar un selector de segmento nulo en CS o SS se provoca una excepción.

Este selector sólo puede cargarse en los registros DS, ES, FS o GS, pero su acceso, una vez cargados con valor cero, genera una excepción.

Éste es un método para detectar accesos a registros de segmentos que no se usan y evitar accesos no deseados a segmentos de información.

10.2.2.3. Protección según el nivel de privilegio

El campo DPL de los atributos de un descriptor, consta de dos bits que expresan el nivel de privilegio del descriptor, o sea, del segmento al que selecciona. Para identificar el nivel de privilegio de un segmento, hay que estudiar el campo DPL de los atributos de un descriptor.

Los niveles de privilegio son usados por el procesador para evitar accesos indebidos a los mismos, como puede ser que un programa de un nivel bajo de seguridad acceda a los segmentos de niveles superiores. En caso de que esto ocurra, el procesador genera una excepción de protección general (GP).

La mezcla de segmentos de mayor nivel de seguridad con otros segmentos de menor nivel de seguridad por parte del usuario no se puede permitir ya que, si sucediera, se degradaría al nivel de privilegio del más seguro.

Siempre que se quiera acceder a un segmento debe ser a través de una instrucción ejecutada en el segmento de código en curso, su nivel de privilegio se denomina Nivel de Privilegio en Curso (CPL).

Mediante la ejecución de una instrucción, es decir, desde el segmento de código en curso de procesamiento se realiza el acceso a cualquier segmento. Es imposible acceder a un segmento desde uno que no sea de código. Para realizar la selección del segmento en curso se hace mediante el selector cargado en CS: sus 13 bits de más peso actúan como índice de la tabla GDT o LDT, pudiendo así encontrar el descriptor del segmento de código cuyo nivel de privilegio es definido por el campo DPL.

El nivel de privilegio del segmento de código en curso recibe el nombre de Nivel de Privilegio en Curso, CPL y a partir del valor del CPL se determinan las reglas de acceso a otros segmentos, ya que son las instrucciones que contiene las que determinan los mismos.

Para localizar un operando, una instrucción puede necesitar acceder a un segmento de datos, por ejemplo, MOV EAX, 5FFFH. Esta instrucción accede al dato localizado en la posición 5FFFH del segmento de datos DS (DS:5FFFH) para cargarlo en EAX.

También se puede acceder a un segmento de pila con una instrucción como PUSH EAX. Esta instrucción lo que hace es introducir un nuevo dato en la pila.

Las instrucciones del tipo JMP y CALL. La JMP: realiza un salto incondicional a otro segmento de código, por otro lado, la CALL: llama a una rutina que al terminar (RET) vuelve a la posición en la que se encontraba al hacer la CALL. Ésta última guarda en una pila parámetros del segmento de código en curso, por lo que siempre debe haber un segmento de pila en el mismo PL que haya un segmento de código.

En el caso de Pentium, siempre que haya un acceso a un segmento, se comprueba la relación entre el CPL y el DPL del segmento que será accedido.

Para controlar el acceso a los distintos tipos de segmentos existen unas reglas básicas que se describen a continuación:

1ª REGLA: ACCESO A SEGMENTOS DE CÓDIGO

Sólo se puede acceder a segmentos de código que tengan el mismo PL que el segmento de curso peticionario.

Si se quisiera acceder a otro de mayor PL, no está permitido porque éste se "rebajaría" a nuestro nivel, que es menos seguro. Si por el contrario, se desease acceder a uno de menor PL, el segmento peticionario sería el que se "rebajara" a dicho nivel inferior.

Las instrucciones que permiten estas llamadas o saltos directos son JMP, CALL y RET. El CPL del segmento de código peticionario debe ser igual al DPL del segmento a acceder directamente con la instrucción JMP,CALL o RET.

En la figura 10.8. se muestran los accesos permitidos entre segmentos de código y accesos no permitidos entre estos. Para que se puedan realizar dichos accesos los segmentos de código tienen que tener igual nivel de privilegio.

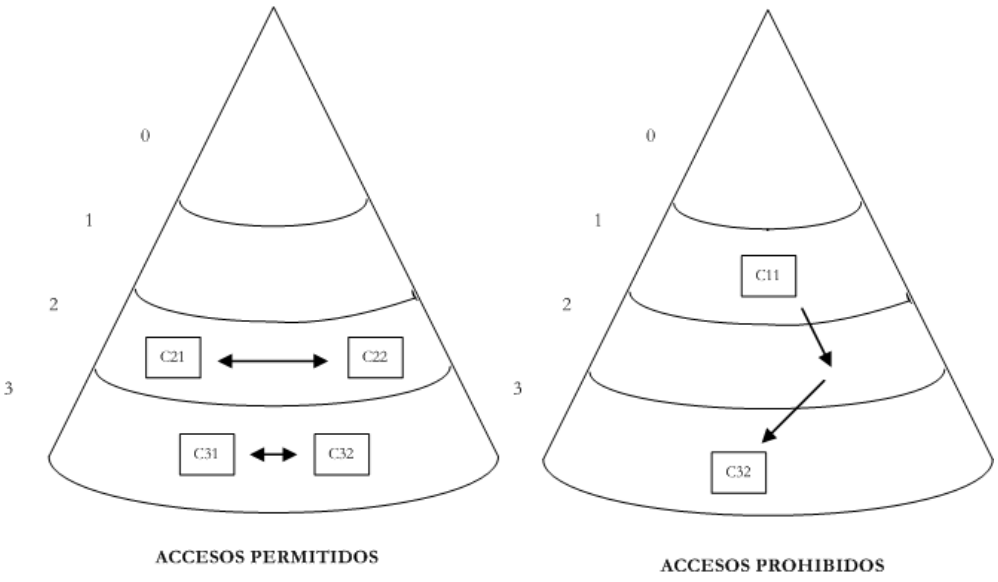

Figura 10.8. El acceso entre segmentos de código

El selector de segmento del destinatario debe ser cargado en el CS para realizar la transferencia del control del programa de un segmento de código a otro. Las comprobaciones en cuanto al límite, tipo y privilegio son examinadas por parte del procesador como parte de este proceso de carga. Si son satisfactorias, lo que implica que el registro CS es cargado, el control del programa es transferido al nuevo segmento de código y empiezan a ejecutarse las instrucciones apuntadas por el registro EIP.

Las transferencia del control del programa las llevan a cabo las instrucciones JMP, CALL, RET, INTn, IRET, SYSENTER y SYSEXIT, al igual que lo hacen los mecanismos de excepción e interrupción. Las dos últimas instrucciones son especiales y se usan para ejecutar procedimientos ó para realizar llamadas rápidas o para regresar del S.O.

El Pentium dispone de un recurso llamado Puerta de Llamada creada por el programador de sistema ya que es normal que desde un segmento de código de bajo PL se quiera acceder a rutinas del S.O. (PL superior).

2ª REGLA DE ACCESO A SEGMENTOS DE DATOS

Sólo está permitido el acceso desde segmentos de código con un PL a otros de datos de igual o menor PL. Para ello se usan instrucciones tipo MOV o similares.

Un segmento de datos se puede leer y escribir. Para evitar corromper la seguridad de los segmentos de datos se usa esta regla, ya que obliga a ser igual o mayor el nivel de privilegio del segmento de código que les intenta acceder. La escritura se realiza con la seguridad correspondiente al segmento de código. Se produciría una degradación del nivel de privilegio del segmento de datos si se accediese desde un segmento de código con menor PL (Figura 10.9).

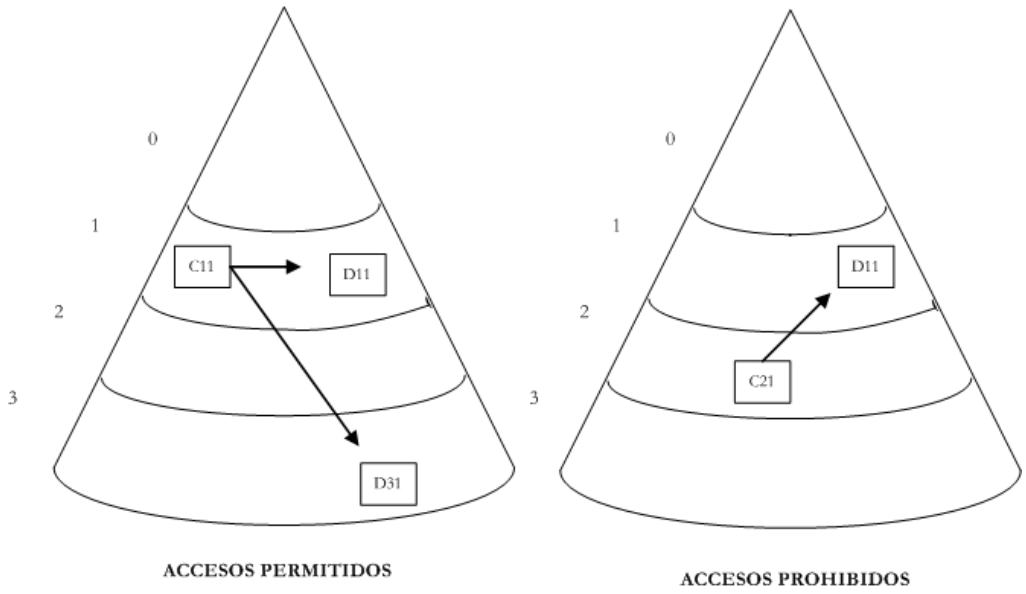

Figura 10.9. El acceso entre segmentos de datos

A través de una puerta de llamada se puede aumentar el PL de un segmento de código, pudiendo así, pasar a la pila un parámetro que sería usado como selector al ser cargado en un registro de segmento.

Esto ocurre si se comparan los valores, solamente, de CPL y DPL; para solucionarlo se compara el DPL del segmento al que se quiere acceder con el EPL (Nivel de Privilegio Efectivo).

3ª REGLA DE ACCESO A SEGMENTOS DE PILA

Sólo se permite acceder a segmentos de pila con el mismo PL que el del segmento de código que los solicita.

El CPL, el RPL del selector de segmento y el DPL del selector del segmento de pila deben ser iguales. En caso de restricción se genera una excepción de protección general (GP).

El procesador debe tener en cuenta los siguientes tipos de niveles de privilegio para llevar a cabo todas las comprobaciones:

1. Nivel de privilegio actual (CPL): PL del programa o tarea en curso, es el mismo que del segmento de código del que se realizan actualmente las instrucciones.

Al realizar una transferencia de control internivel, el procesador cambia el CPL.

El CPL puede encontrarse en los dos bits de menos peso de los registros CS y SS.

2. Nivel de privilegio del descriptor (DPL): PL de un segmento o puerta. Se almacena en el campo DPL del descriptor del segmento ó de la puerta.

Dependiendo del tipo de puerta o segmento al que se quiere acceder, se puede interpretar el DPL de diferentes maneras:

- Segmento de datos: el DPL indica el menor nivel de privilegio que un programa o tarea puede tener para que pueda acceder al segmento.
- Segmento de código, que no puede ser accedido a través de una puerta de llamada: indica el PL que se debe tener para poder acceder al segmento.
- Segmento de código accedido a través de una puerta de llamada: indica el mayor PL que se debe tener para acceder al segmento.
- Puerta de llamada: el DPL es el menor PL que se le permite tener a u segmento para acceder a la puerta. Esta norma es igual que la de los segmentos de datos.
- TSS: indica el menor PL que el programa o tarea, en estado de ejecución, puede tener para realizar el acceso. Esta norma es igual que la de los segmentos de datos.

3. Nivel de privilegio del peticionario (RPL): es el PL más importante y es almacenado en los bits 0 y 1 del selector de segmento.

   El CPL y el RPL son comprobados por el procesador para ver si se permite el acceso ó, por el contrario, es un acceso no permitido.

   Para asegurarse que las instrucciones privilegiadas no son accedidas por segmentos de un programador de aplicaciones a menos que el programa en sí tenga privilegios de acceso a este segmento se usa el RPL .

10.2.2.4. Acceso a segmentos con el bit ajustables (conforming) a cero

Para accesos a segmentos de código con C = 0, el CPL del segmento que quiere hacer el acceso, igual del DPL del segmento de código destino, sino se daría excepción de protección general (GP).

Al cargar un selector de segmento de un segmento de código en un registro CS, el campo del nivel de privilegio no varía aún cuando el RPL del segmento selector sea diferente del CPL.

El RPL debe ser jerárquicamente mayor o igual que el CPL de la rutina, por lo que se dice que el RPL tiene un efecto limitado en la comprobación de privilegio.

10.2.2.5. Acceso a segmentos con el bit ajustables (conforming) a uno

Cuando se accede a segmentos de código con C = 0, el CPL de la rutina que hace la llamada, puede ser igual o menor que el DPL del segmento de código destino, sino se daría excepción de protección general (GP).

En el caso de C = 1, no se tiene en cuenta el RPL del selector de segmento del destino. En estos casos el DPL representa numéricamente el de menor PL que puede tener una rutina para poder hacer una llamada.

El CPL no cambia al transferir el control del programa a un segmento de código conforming (aunque el DPL del destino sea mayor que el CPL), por lo que no hay intercambio de pilas. Es la única situación en la que el CPL puede ser distinto del DPL del segmento de código actual.

Los segmentos ajustables se usan para módulos de código como por ejemplo, librerías matemáticas y manejadores de excepciones, que aunque formando parte del S.O. o del software ejecutable pueden ser ejecutados en niveles de menos privilegio.

Para los segmentos de código nonconforming, que son la mayoría, a menos que la transferencia se lleve a cabo mediante una puerta de llamada, el programa de control puede ser transferido sólo a segmentos de código con el mismo nivel de privilegio.

10.2.3.Protección de las páginas

Los niveles existentes en la paginación son el Usuario y el Supervisor como se muestra en la figura 10.10. Las reglas de paginación se aplican tras aplicar las reglas en segmentación.

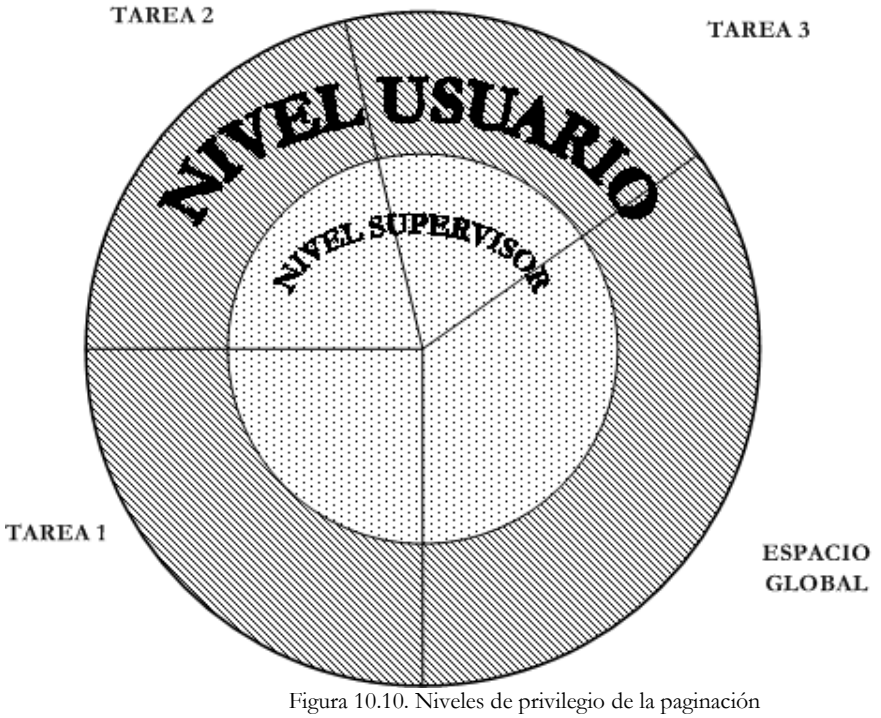

Figura 10.10. Niveles de privilegio de la paginación

Para definir el acceso el único bit que se usa el W/R, el cual se encuentra en las entradas de las Tablas de Páginas y en las del Directorio de Páginas. En caso que éste sea 0, puede leerse, si es 1 puede tanto leerse como escribirse.

Puede establecerse una relación entre estos niveles y los de la segmentación: en lo referente al nivel supervisor, éste equivale al los niveles 0, 1 o 2 en los que se realizan la ejecución de instrucciones y programador del sistema.

En cuanto al nivel usuario, que equivale al nivel 3 se realizan las aplicaciones de usuario y del programador de aplicaciones. En este tipo de protección se tienen en cuenta dos reglas:

1. Desde una página ubicada en nivel Usuario sólo se puede acceder a páginas de dicho nivel.

2. Desde una página del nivel de Supervisor se puede acceder a todas.

Este mecanismo de protección se usa al nivel de Directorio de Páginas y al nivel de Tabla de Páginas, es decir en la traducción de direcciones lineales a físicas. Puede darse que haya diferencias en ambas entradas respecto a la especificación de página, por lo que se coge como nivel de privilegio y tipo de acceso el más restrictivo.

Las distintas combinaciones quedan reflejadas en la tabla de la figura 10.11:

Entrada al Directorio de Páginas		**Entrada a la Tabal de Páginas**		**Efecto Combinado**	
Privilegio	Tipo de acceso	Privilegio	Tipo de acceso	Privilegio	Tipo de acceso
Usuario	Solo lectura	Usuario	Solo lectura	Usuario	Solo lectura
Usuario	Solo lectura	Usuario	lectura-escritura	Usuario	Solo lectura
Usuario	lectura-escritura	Usuario	Solo lectura	Usuario	Solo lectura
Usuario	lectura-escritura	Usuario	lectura-escritura	Usuario	lectura/escritura
Usuario	Solo lectura	Supervisor	Solo lectura	Supervisor	lectura/escritura*
Usuario	Solo lectura	Supervisor	lectura-escritura	Supervisor	lectura/escritura*
Usuario	lectura-escritura	Supervisor	Solo lectura	Supervisor	lectura/escritura*
Usuario	lectura-escritura	Supervisor	lectura-escritura	Supervisor	lectura/escritura
Supervisor	Solo lectura	Usuario	Solo lectura	Supervisor	lectura/escritura*
Supervisor	Solo lectura	Usuario	lectura-escritura	Supervisor	lectura/escritura*
Supervisor	lectura-escritura	Usuario	Solo lectura	Supervisor	lectura/escritura*
Supervisor	lectura-escritura	Usuario	lectura-escritura	Supervisor	lectura/escritura
Supervisor	Solo lectura	Supervisor	Solo lectura	Supervisor	lectura/escritura*
Supervisor	Solo lectura	Supervisor	lectura-escritura	Supervisor	lectura/escritura*
Supervisor	lectura-escritura	Supervisor	Solo lectura	Supervisor	lectura/escritura*
Supervisor	lectura-escritura	Supervisor	lectura-escritura	Supervisor	lectura/escritura

Figura 10.11. Tabla de niveles de privilegio

(*) si el flag WP de CR0 está a 1, el tipo de acceso lo determina R/W de las entradas del Directorio de Páginas y de la Tabla de páginas

Se debe tener en cuenta que no todas las instrucciones pueden ejecutarse desde cualquier nivel de privilegio, el sistema de protección el Pentium resuelve este problema.

Cabe destacar la existencia de dos grupos de instrucciones especiales que exigen ser ejecutadas con un CPL mínimo: el de instrucciones protegidas y el de instrucciones privilegiadas

10.2.4.1. Instrucciones protegidas

Son aquéllas que sólo pueden ejecutarse desde objetos de código situados en un nivel de privilegio igual o mayor que el indicado en el campo IOPL (Nivel de privilegio de E/S) del registro EFLAGS, éste es determinado por el programador del sistema.

El campo IOPL puede ser controlado dinámicamente por el S.O. para decidir los módulos que tienen acceso a los periféricos, este hecho se muestra en la figura 10.12.

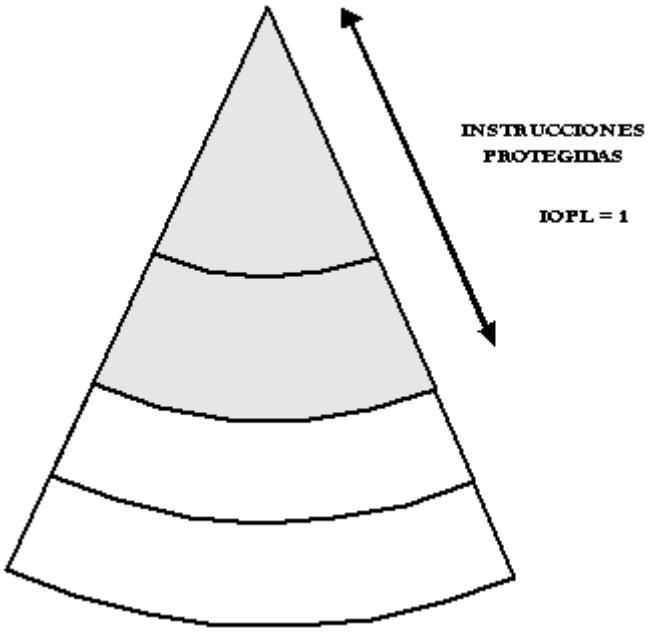

Figura 10.12. Si IOPL = 1, el grupo de instrucciones protegidas sólo se puede ejecutar desde segmentos de código con nivel de privilegio 0 y 1

Se incluyen las siguientes instrucciones para manejar las operaciones del espacio de E/S periféricos:

1. IN: puerta por la que toma un valor para cargar el acumulador
2. OUT: puerta en la que se deposita el valor del acumulador (EAX)
3. INS, OUTS: como las anteriores, pero manejan una cadena de caracteres
4. CLI: pone el flan de interrupción IF del registro de señalizadotes a 0, no permitiendo instrucciones mascarables
5. STI: pone el flan de interrupción IF del registro de señalizadotes a 1, permitiendo instrucciones mascarables.

10.2.4.2.Instrucciones privilegiadas

Están protegidas de los programas de aplicaciones, estas instrucciones controlan las funciones del sistema (como por ejemplo la carga de los registros del sistema).

Sólo se pueden ejecutar desde el nivel de mayor de privilegio (CPL=0), sino se da una excepción de protección general (GP).

Grupos en los que se clasifican:

1. Instrucciones que pueden modificar el IOPL

   POPF: carga los 32 bits de la cima de la pila en el registro de flags afectando al IOPL

   RET: retorno de interrupción, además de retornar al programa original, carga una parte en el registro de estado

2. Instrucciones que escriben los registros que controlan las tablas del sistema al operar en modo protegido

3. Instrucciones que afectan al contenido de la palabra de estado

4. Instrucción de paro HTL.

A continuación se muestra una lista de las instrucciones privilegiadas:

- LGDT: carga el registro GDT
- LLDT: carga el registro LDT
- LTR: carga el registro de tareas
- LIDT: carga el registro IDT
- MOV (registros de control):carga y almacena los registros de control
- LMSW: carga la palabra de estado de la máquina
- CLTS: pone a cero el flan de conmutación de tareas del registro CR0
- MOV (registro debug): carga y almacena registros de debug
- INVD: invalida la caché, sin actualizar la memoria principal
- WBINVD: invalida la caché, actualizando la memoria principal
- INVLPG: invalida la TLB
- HLT: para temporalmente el procesador.
- RDMSR: Read Mode-Specific Registers : instrucción de leer modelo y registro específico.
- WDMSR: Write Mode-Specific Registers : instrucción de escribir modelo y registro específico.
- RDPMC: Read Preformase-Monitoring Counter: leer contadores de monitoreo de desempeño.
- RDTSC: Read Time-Stamp Counter: leer contador de tiempo estampado