UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Microprocesadores

Registros

Dra. Claudia Silvia Litvak
Ing. Silvana Panizzo
Ing. María de la Paz Colla

Programador de sistemas y programador de aplicaciones

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Programador de aplicaciones

- Es el encargado de **codificar programas** para los usuarios finales.
- En general suelen **desarrollar aplicaciones compuestas** por uno o más programas en un lenguaje de alto nivel comercial.
- Contempla la **CPU como un conjunto de registros de trabajo** que le permiten confeccionar dichas aplicaciones del usuario.
- Con la **visión limitada** que tiene **del procesador** puede **manipular las instrucciones del lenguaje** que aplica, datos y direcciones, además de los elementos necesarios que le permitirán desarrollar sus programas de aplicación.

Programador de sistemas y programador de aplicaciones

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Programador de aplicaciones

- Para este programador **resultan transparentes los recursos de la CPU** empleados para llevar a cabo la **tarea de aplicación en coordinación con otras distintas** y de acuerdo con un **mecanismo de protección** que controla los accesos evitando las transgresiones entre tareas.
- Los conocimientos que el programador de aplicaciones debe tener sobre la máquina son los imprescindibles para obtener el **máximo rendimiento de las instrucciones** usadas para resolver las **aplicaciones**.
- En el caso de emplear el **lenguaje máquina**, deberá conocer los registros internos accesibles para la manipulación de datos y direcciones, el repertorio básico de instrucciones y los modos de direccionamiento.

Programador de sistemas y programador de aplicaciones

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Programador de sistemas

- Es el encargado de **desarrollar programas y utilidades del sistema operativo**, se necesita mayor cualificación que para los programas de aplicación ya que este programador es el que controla a los programadores de aplicaciones.

- La misión de este programador es **construir un sistema de explotación óptimo** que sea **capaz de soportar todas las aplicaciones previstas.**

- Entre sus funciones más destacadas están:

  - **Organizar el sistema** para el correcto tratamiento de las tareas pertenecientes a los diferentes usuarios.
  - Confección de **objetos para sistemas operativos, depuradores, compiladores**, etc.
  - Asignar a cada tarea su **nivel de privilegio** y un **sistema de protección adecuado**.
  - **Organizar toda la memoria** y **el procesador** para que de esta forma las tareas consigan un mejor rendimiento.

Programador de sistemas y programador de aplicaciones

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Programador de sistemas

- Es indispensable que **conozca profundamente la arquitectura detallada de la CPU** para así **optimizar** todos los recursos, obteniendo en su funcionamiento la máxima potencia, seguridad y rendimiento.
- Debe conocer también **las prestaciones de la memoria virtual, las características de protección del entorno, los mecanismos que posibilitan la conmutación de tareas, el tratamiento de interrupciones y excepciones**, etc.
- Los microprocesadores Pentium **disponen de una serie de registros y recursos especiales**, **denominados «del sistema»,** que se encargan de gestionar el funcionamiento general, aprovechando todas las prestaciones.
- La complejidad de estos microprocesadores ha hecho necesaria **la construcción de herramientas para el desarrollo de software.**
- Estas herramientas varían **dependiendo del tipo de programador.**

Programador de sistemas y programador de aplicaciones

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Herramientas especificas

- Intel inicialmente diseñó una herramienta lógica, denominada **BINDER**, **encargada de la generación de aplicaciones**, sencilla de manejo, pero que no permite el acceso a los mecanismos del sistema, haciéndola idónea para el programador de aplicaciones.
- El **BUILDER,** por el contrario, es otra poderosa **herramienta destinada a la construcción de sistemas**, que posibilita el **control de todos los recursos de la CPU**, por lo que requiere ser usada por los **programadores de sistemas**, **capaces de describir con detalle la estructura global que se precisa implantar.**

Registros internos para el programador de aplicaciones

El Pentium dispone de 32 registros en su arquitectura interna de los cuales la mitad, es decir, 16, son para uso del programador de aplicaciones:

Se clasifican en cuatro grandes grupos:

- Registros de propósito general.
- Registro Puntero de Instrucciones (EIP).
- Registro de Estado o de Señalizadores (EFLAGS).
- Registros de Segmento.

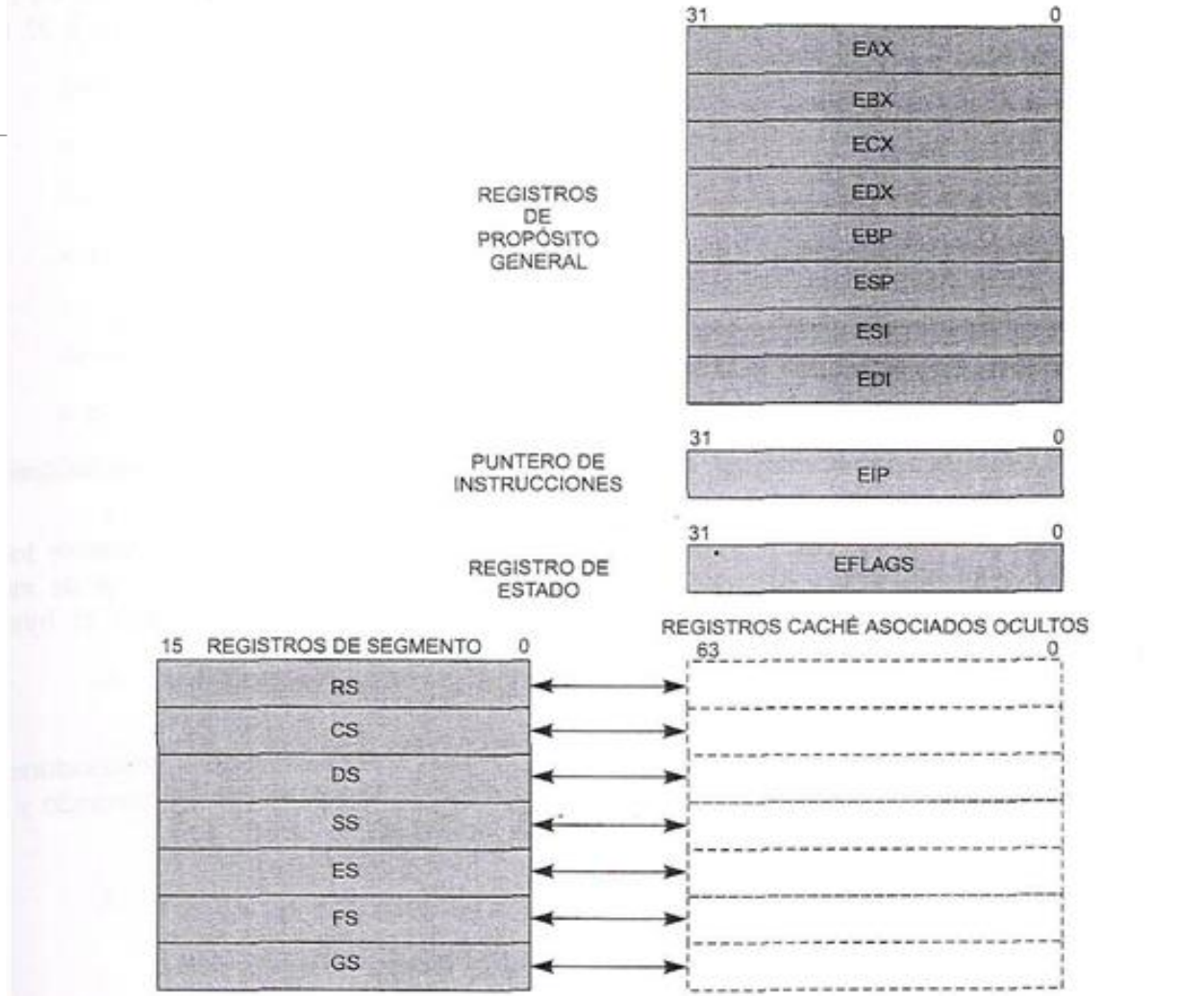

Figura 7.1. *Estructura general de los 16 registros del Pentium para el programador de aplicaciones.*

Registros internos para el programador de aplicaciones: Registros de propósito general

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

- Son los mismos registros que tenía el 8086 de 16 bits, pero ampliados a 32 bits.
- Este grupo consta de ocho registros capaces de trabajar con información de 32 bits cuando utilizan todo su tamaño aunque también pueden manejar datos de 16 bits, e incluso cuatro de ellos pueden manejar informaciones de 8 bits.
- Los registros de propósito general pueden usarse tanto para almacenar datos como direcciones, en este último caso, el contenido del registro es un desplazamiento que apunta a una dirección de memoria.
- A los registros de propósito general se les ha mantenido el mismo nombre que en el 8086 pero anteponiendo la letra E de «extendido», para hacer referencia a la extensión a 32 bits de los mismos.
  - EAX: Acumulador.
  - EBX: Base.
  - ECX: Contador.
  - EDX: Datos.
  - ESP: Puntero de pila.
  - EBP: Puntero de base.
  - ESI: índice fuente.
  - EDI: índice destino.

Registros internos para el programador de aplicaciones: Registros de propósito general

- Cuando se accede únicamente a los 16 bits de menos peso de estos registros, se designan por **AX, BX, CX, DX, SP, BP, SI, DI** respectivamente.
- También son accesibles, de forma independiente, los dos bytes de menos peso de los registros **AX, BX, CX y DX.**
- En estos cuatro registros cuando se accede al byte de menos peso se le denomina **AL, BL, CL y DL,** respectivamente. Cuando se maneja el byte de más peso se les denomina **AH, BH, CH y DH** respectivamente (Figura 7.2).

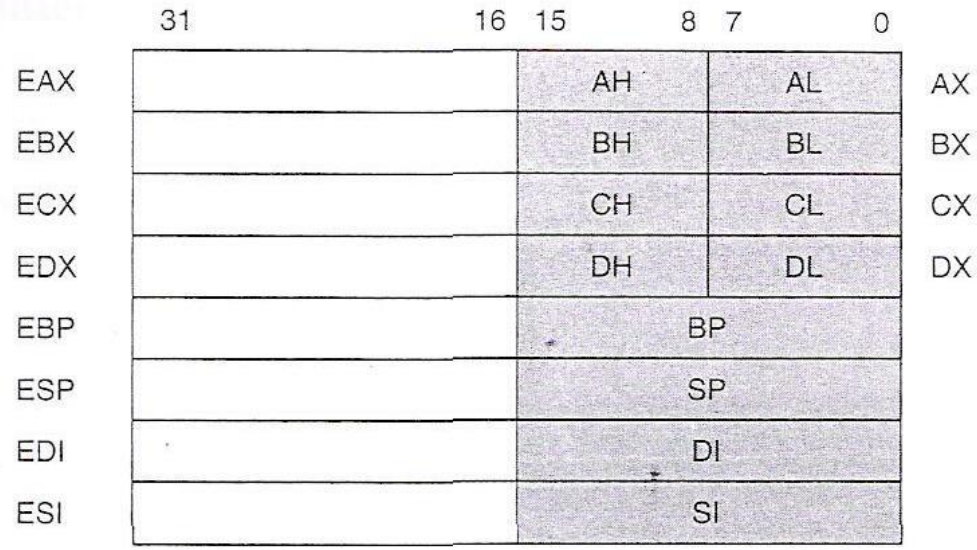

Figura 7.2. *Formatos y denominaciones de los registros de propósito general.*

Registros internos para el programador de aplicaciones: Registro acumulador

- **Acumulador (EAX, AX, AL, AH)**
  - Es un registro que se emplea en todas las operaciones lógico-aritméticas. Los procesadores de Intel hacen un uso intensivo del acumulador ya que por una parte contiene un operando y por otra se carga con el resultado de las operaciones de la ALU.

Ejemplo 7.1. **MOV AH, AL**

Solución

- El byte de menos peso del Acumulador (AX) se mueve al byte AH, que es el de más peso.

Registros internos para el programador de aplicaciones

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

- **Base (EBX, BX, BL, BH)**

Contiene una dirección que apunta a la base de un conjunto de datos. La longitud de las direcciones puede ser larga o corta. Se utiliza el registro EBX (32 bits) para el direccionamiento en la memoria que maneja el Pentium (dirección larga), mientras el registro BX (16 bits) para manejar la memoria del 8086 (dirección corta).

- **Contador (ECX, CX, CL, CH)**

Se carga con el número de veces que se ejecuta una instrucción (iteraciones).

- **Datos (EDX, DX, DL, DH)**

Este registro de propósito general se emplea para contener las direcciones de los puertos de entrada y salida en las instrucciones que manejan el mapa de E/S.

Registros internos para el programador de aplicaciones

- **Puntero de pila (ESP, SP) y Puntero de base (EBP, BP)**

Sirven para controlar el direccionamiento de la pila.

Esas operaciones en la pila son soportadas por tres registros diferentes que se exponen a continuación (Figura 7.3):

1. Registro de segmento de pila (SS). Registro de base de segmento de pila o stack que reside en memoria.
2. Registro puntero de pila (ESP). Contiene el desplazamiento de la cima de la pila en el segmento de la pila actual. Lo usan las operaciones PUSH y POP, las llamadas a subrutinas. el retorno, las excepciones y las interrupciones. Cuando se introduce un elemento a la pila, el procesador decrementa el puntero ESP, y escribe el elemento en la cima de la pila. Cuando se saca un elemento de la pila se hace la operación contraria, es decir, se incrementa el ESP.
3. Registro puntero base de la pila (EBP). Se usa normalmente para acceder a estructuras de datos pasadas a la pila. Cuando el registro EBP se usa para direccionar memoria, el segmento de pila en curso es referenciado. Este registro apunta a la base de la pila y cuando existen rutinas hace el papel de ESP para no modificar el valor de este último.

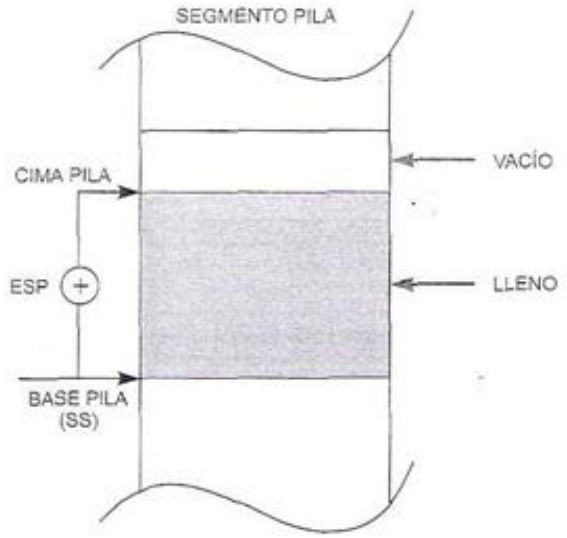

Figura 7.3. *La cima de la pila se direcciona sumando a la base del segmento el valor del registro ESP.*

Registros internos para el programador de aplicaciones

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

• **Índice fuente (ESI, SI) e índice destino (EDI, DI)**

Son dos punteros de direcciones necesarios para trabajar con cadenas de caracteres.

Las instrucciones que realizan operaciones entre los elementos de las cadenas o «strings» se aplican a una cadena fuente y a una cadena destino, cada una de ellas está direccionada por el registro ESI y el EDI, respectivamente (Figura 7.4).

Tienen la propiedad de autoincremento y autodecremento. Cada vez que se ejecuta una instrucción se incrementan o decrementan los registros índices en el número de bytes que tenga cada elemento de la cadena.

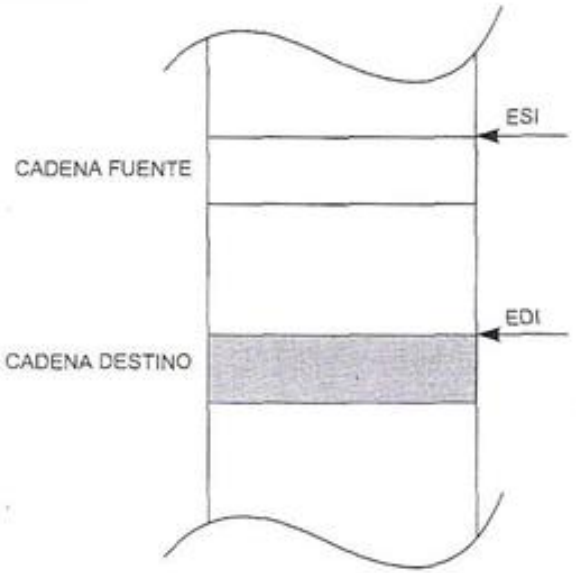

Figura 7.4. *Los registros ESI y EDI apuntan al elemento en curso de la cadena fuente y la cadena destino, respectivamente.*

Registros internos para el programador de aplicaciones: Registro puntero de instrucciones

EIP: Registro puntero de instrucciones

El EIP es el puntero de las instrucciones o contador de programa y no está disponible para el programados lo gobierna implícitamente el flujo de control de las instrucciones, las interrupciones y las excepciones.

Este registro puede trabajar en dos modos:

- En modo Nativo o Protegido tiene una longitud de 32 bits y recibe el nombre de EIP.
- Almacena el desplazamiento que hay que añadir a la base del segmento de código para obtener la dirección donde está la siguiente instrucción a ejecutar.
- La base del segmento de código se obtiene a partir del valor del registro de segmento CS.
- El valor máximo del desplazamiento en el segmento de código será de 4 GB.

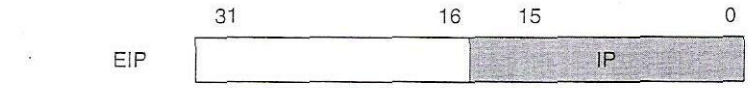

Figura 7.5. *El Puntero de Instrucciones tiene un tamaño de 32 bits (EIP) en modo Protegido y de 16 bits (IP), cuando soporta el direccionamiento «reducido» del 8086.*

Registros internos para el programador de aplicaciones: Registro puntero de instrucciones

EIP: Registro puntero de instrucciones

- En modo Real se emplea un tipo de direccionamiento reducido, compatible con los procesadores 8086 y 80286. que sólo precisa de 16 bits para especificar el desplazamiento en el segmento de código. Son los dos bytes de menos peso de EIP que se denominan IP. En este caso, el valor máximo del desplazamiento será de 64 KB.

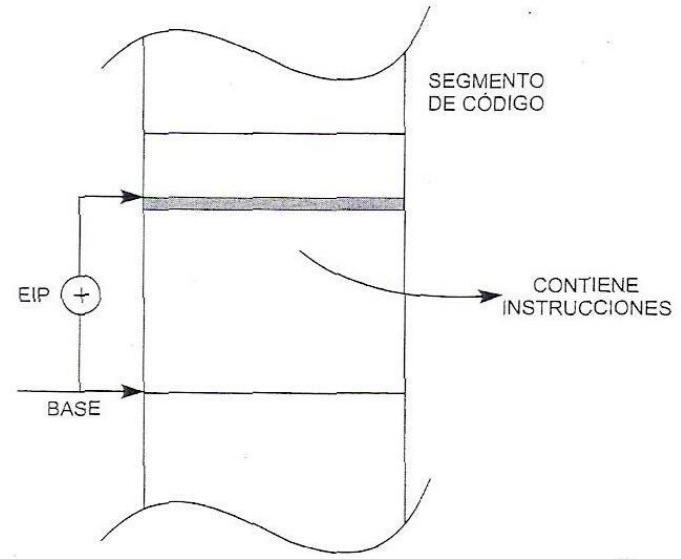

Figura 7.6. *En el segmento de código para direccionar la instrucción a ejecutar en cada momento, se suma a la base del segmento el contenido del registro puntero de instrucciones (EIP o IP).*

Registros internos para el programador de aplicaciones: Registro EFLAGS

Registro de estado o de señalizadores (EFLAGS)

Este registro, también conocido como registro EFLAGS, consta de 32 bits, de los cuales la mayoría son señalizadores de estado, controlados por la ALU (acarreo, paridad, acarreo auxiliar, cero, signo y overflow), actuando los restantes como señalizadores del sistema, ligados al mecanismo de protección y a otros recursos que dispone el sistema.

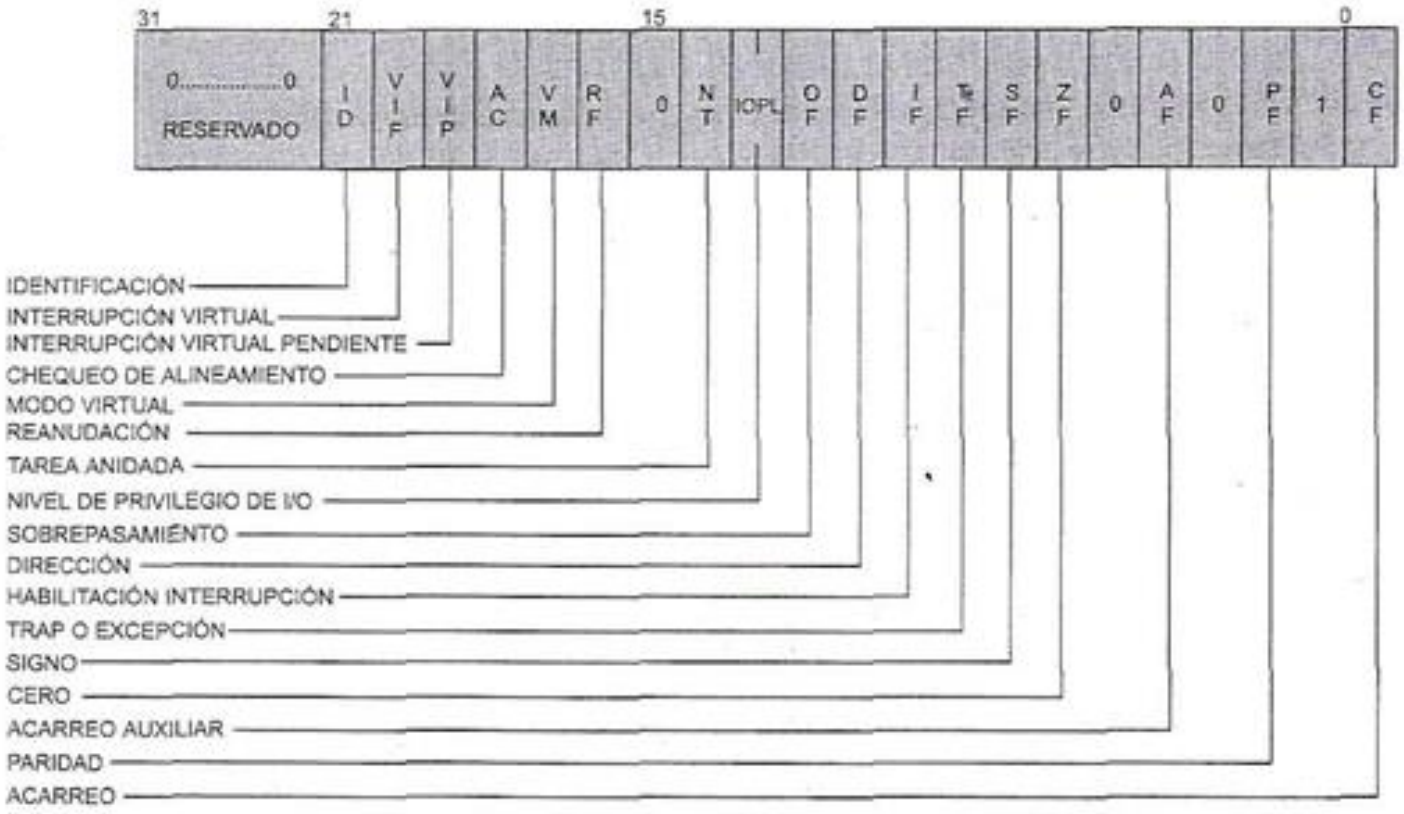

Figura 7.7. *Registro EFLAGS.*

Registros internos para el programador de aplicaciones: Registro EFLAGS

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Registro de estado o de señalizadores (EFLAGS)

A continuación, se describe cada uno de los bits que forman parte del registro de estado.

• CF: Es el señalizador de acarreo en el bit de más peso al realizarse una operación de suma aritmética.

- 1: Indica que ha existido acarreo en el bit de más peso en una operación de suma aritmética.
- 0: Cuando el señalizador CF toma este valor significa que no ha habido acarreo en el bit de más peso de una suma, o bien, que ha existido «llevada» en el bit de más peso de una resta aritmética.

• PF: Bit de paridad impar.

- 1: Toma este valor para generar la paridad impar con los bits que conforman el resultado de una operación.
- 0: Valor para generar paridad impar.

• AF: Señalizador de acarreo auxiliar (o intermedio).

- 1: Toma este valor cuando ha habido acarreo en el bit 3 del resultado. Se utiliza en las operaciones con números BCD.
- 0: En caso de no haber acarreo en el bit 3 del resultado.

Registros internos para el programador de aplicaciones: Registros EFLAGS

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Registro de estado o de señalizadores (EFLAGS)

• ZF: Señalizador de cero.
- 1: Cuando todos los bits del resultados son ceros.
- 0: En caso de que el resultado no sea 0.
• SF: Señalizador de signo.
- 1: Si el bit de más peso del resultado de la operación es 1.
- 0: Si el bit de más peso del resultado de la operación es 0.
• TF: Excepción al terminar la ejecución de la instrucción.
- 1: Provoca una excepción al completarse la ejecución de la instrucción en curso. Posibilita la depuración de los programas al permitir la ejecución paso a paso, es decir, instrucción por instrucción.
- 0: No hay excepción de depuración al terminar cada instrucción.
• IF: Flag de habilitación de interrupciones mascarables.
-- 1: Permite el reconocimiento de las peticiones de interrupción mascarables provocadas por la activación de la patita ÍNTR del Pentium.
- 0: Prohibe el reconocimiento de la interrupción externa e ignora las peticiones de interrupción mascarables.

Registros internos para el programador de aplicaciones: Registros EFLAGS

Registro de estado o de señalizadores (EFLAGS)

• DF: Flag de dirección de exploración de las cadenas de caracteres o strings.

- 1: Postdecremento automático de los registros ESI y EDI, qne direccionan la cadena.
- 0: Postincremento automático de ESI. EDI.

• OF: Flag de desbordamiento (u o vertió wt.

- 1: En operaciones con números enteros con signo se activa y vale 1 si el resultado es muy grande (es positivo) o muy pequeño (si es negativo). Indica resultados erróneos.
- 0: Si no existe desbordamiento.

• IOPL: Nivel de privilegio de las entradas y salidas. Es un campo de 2 bits que se emplean en modo Protegido y determinan el nivel de privilegio que debe igualar o superar el segmento de código en el que se desean ejecutar instrucciones de *E/S* que manejan datos en el espacio de E S del procesador que está reservado a guardar la información de los periféricos del sistema.

Los valores que se pueden tomar son:

- 11: nivel 3 (pueden acceder todos).
- 10: nivel 2.
- 01: nivel 1.
- 00: nivel O (únicamente puede acceder el SO).

• NT: Tarea anidada. Este señalizador actúa automáticamente al producirse una conmutación de tareas.

- 1: La tarea en curso está anidada con la anterior. Hay que retornar obligatoriamente a la tarea previa.
- 0: La conmutación de tareas es libre.

Registros internos para el programador de aplicaciones: Registros EFLAGS

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Registro de estado o de señalizadores (EFLAGS)

RF: Flag de reanudación. Su activación provoca la ejecución de la siguiente instrucción cuando se produce un excepción de depuración en una instrucción, es decir, se ignora.

- 1: Se ignoran los plintos de depuración o parada,
- 0: No se ignoran los puntos de parada.

• VM: Modo Virtual. Mediante este bit se permite el paso desde el modo Protegido al modo Virtual 86.

- 1: Estando el procesador en modo Protegido se pasa a modo Virtual.
- 0: No hay paso al modo Virtual 86.

• AC: Bit de chequeo de alineamiento.

- 1: Se produce una excepción cuando se encuentra una palabra o dato desalineado en la memoria.
- 0: No hay excepción por desalineamiento.

Para que el bus de datos de 64 líneas pueda recibir en un solo ciclo el contenido de 8 bytes es necesario que los mismos comiencen en una dirección múltiplo de 8. Lo mismo puede suceder con las palabras de 4 bytes que deben ocupar posiciones a partir de una dirección múltiplo de 4. Si el programador no deja alineados los daros en la memoria, obligará a realizar más de un acceso para recoger toda la información.

Registros internos para el programador de aplicaciones: Registros EFLAGS

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Registro de estado o de señalizadores (EFLAGS)

• VIF: Se trata de un bit que realiza una misión similar al IF, pero en el modo Virtual 86. Permite o prohíbe la atención a las peticiones de interrupción mascarables. Se utiliza con el flag VIP. El procesador sólo reconoce VIF cuando el flag VME o el PVI del registro de control CR4 están activados y el IOPL es menor que 3.

• VIP: Interrupción (mascarable) pendiente en modo Virtual. Trabaja en combinación con el flag VIF para que cada tarea en modo Virtual disponga de un flag equivalente al IF. Se activa por software para indicar que una interrupción está pendiente. El procesador lee este flag pero nunca lo modifica. El procesador sólo reconoce el flag VIP cuando el flag VME o el PVI en el registro de control CR4 están activados y el IOPL es menor que 3.

- 1: Interrupción mascarable pendiente.
- 0: No hay interrupción pendiente.

• ID: Bit de identificación. ID informa si el Pentium soporta la instrucción CPU1D que sirve para su identificación. Mediante la ejecución de CPUID se muestran las características más importantes del procesador.

- 1: El Pentium soporta la instrucción CPUID.
- 0: En caso contrario.

Registros internos para el programador de aplicaciones: Registros de segmento

Registros de segmento

Intel ha incorporado en la arquitectura IA-32 a la segmentación como sistema principal en la organización de la memoria.

Los segmentos son trozos de la memoria de tamaño variable que contienen el mismo tipo de información. Así, hay tres tipos de segmentos en los programas de aplicación.

- **De pila.**
- **De código.**
- **De datos.**

El manejo de la memoria con segmentos favorece la programación estructurada y la modularidad, siendo una técnica apropiada para permitir la reubicación y soportar una estructura de protección muy segura y flexible.

El Pentium controla en cada instante seis segmentos a los que referencia a través de los Registros de Segmento (RS). Esto no significa que sólo pueda manejar seis segmentos, sino que. si desea acceder a otro que no está referenciado por dichos registros, deberá cargarse previamente en uno de ellos, el valor correspondiente al nuevo segmento.

Para controlar los segmentos de trabajo el Pentium dispone de seis registros de 16 bits, que se muestran en la Figura 7.8

Figura 7.8. *Registros de segmento en el Pentium.*

Registros internos para el programador de aplicaciones: Registros de segmento

Desde el punto de vista del programador de aplicaciones, los seis registros de segmento determinan en cada momento los segmentos que es capaz de identificar y manipular la CPU.

Cuando el Pentium trabaja en modo Nativo o Protegido la dirección lógica o virtual de todo elemento accesible en la memoria está formada por una información que consta de los siguientes campos:

- **Selector:** Son los 14 bits de más peso del registro de segmento que referencia al segmento al que se desea acceder y con ellos se encuentran la base donde comienza el segmento, el límite o tamaño que tiene y sus atributos. Los dos bits de menos peso del registro de segmento conforman el campo RPL que indica el nivel de privilegio del segmento peticionario (Figura 7.9).
- **Desplazamiento:** Es un valor que se añade a la base del segmento para localizar la dirección que hay que acceder en él. El tamaño del desplazamiento determina su longitud máxima que en el caso del Pentium, es de 4 GB en el modo Protegido y 64 KB en modo Real.

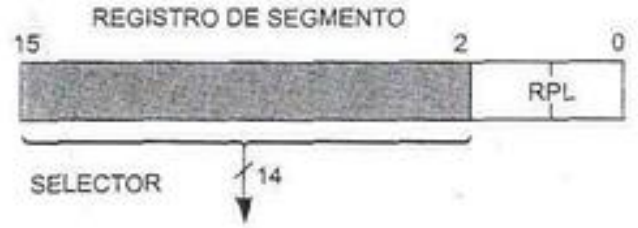

Figura 7.9. *Contenido de los registros de segmento.*

Registros internos para el programador de aplicaciones: Registros de segmento

Cuando el Pentium trabaja en modo Real el valor de la base del segmento se calcula añadiendo cuatro ceros a los 16 bits del registro de segmento correspondiente (Figura 7.10)

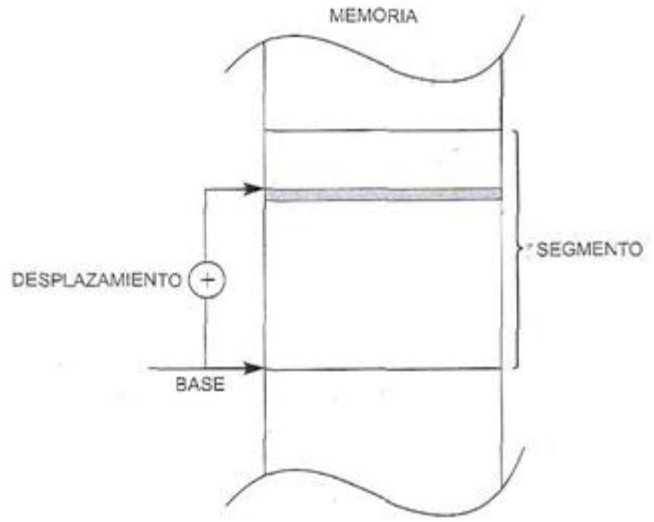

Figura 7.10. *Dirección lógica de un elemento en un segmento.*

Registros internos para el programador de aplicaciones: Registros de segmento

Cada registro de segmento hace referencia a un tipo de segmento determinado.

• **CS:** El registro CS (Segmento de Código), contiene en cada momento la información necesaria del segmento de instrucciones que está ejecutando la CPU, es decir, el segmento en curso. El desplazamiento que hay que añadir a la base del segmento de código reside en el Registro Puntero de Instrucciones EIP.
• **SS:** El registro SS (Segmento de Pila), guarda el valor del selector del segmento de pila en curso. El registro ESP contiene el desplazamiento que debe añadirse a la base del segmento de pila para determinar la cima por donde se cargan y descargan los datos.
• DS: El registro DS (Segmentos de Datos), soporta el valor del selector del segmento de datos y el desplazamiento viene especificado en el modo de direccionamiento usado en la instrucción para expresar operandos y el resultado.
En procesadores x86 de 32 bits disponen de los registros de segmento ES, FS y GS para poder tener activos otros tres segmentos de datos, además del proporcionado por DS.
Ejemplo 7.2. MOV EAX, (5555)
Solución: Se carga EAX con los 32 bits obtenidos en las posiciones de memoria del segmento DS que comienza en la dirección formada por su base más el desplazamiento 5555. Cuando no se explícita el segmento de datos se sobreentiende que es DS. Si fuese el segmento de datos GS. la instrucción sería MOV EAX, GS: 5555.

Registros internos para el programador de aplicaciones: Registros de segmento

Con los registros de segmento la CPU mantiene activos 6 registros en cada instante

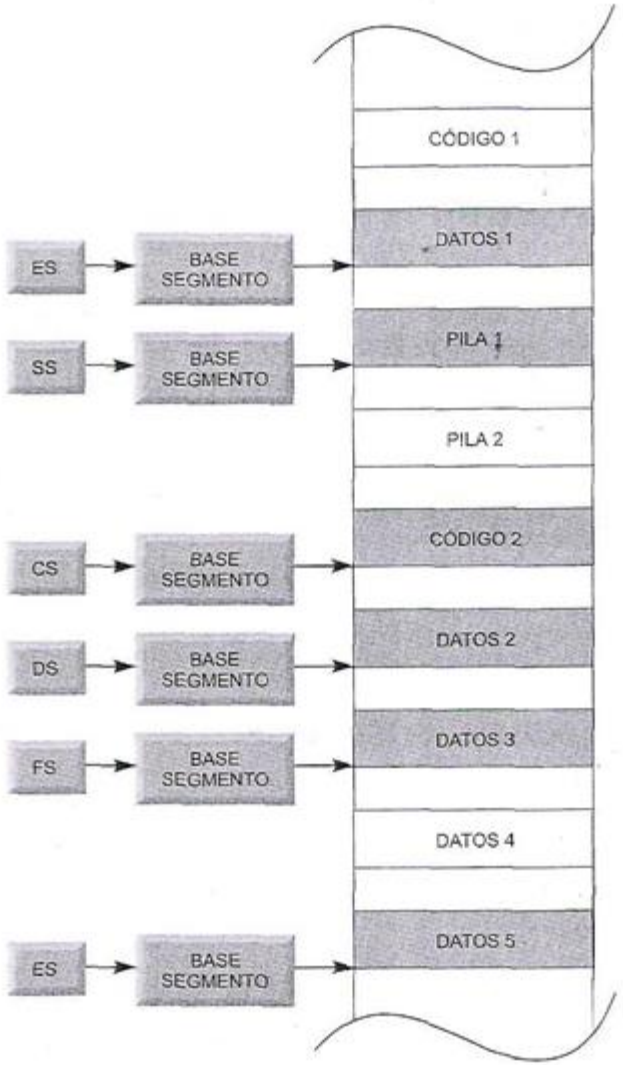

Figura 7.11. *A través del valor de los registros de segmento, la CPU tiene activos a seis segmentos en cada instante.*

Modo real vs Modo Protegido

Modo real:

- ✓ 20 bits de espacio de direcciones segmentado (2^{20} = 1 MB de memoria máximo)
- ✓ Acceso directo del software a las rutinas del BIOS y el hardware periférico
- ✓ No tiene conceptos de protección de memoria o multitarea a nivel de hardware.
- ✓ Palabra de 16 bits.
- ✓ La memoria es segmentada. (Tamaño máximo de segmento 2^{16} = 64 KB).
- ✓ Rango de direcciones de un segmento de 0000 a FFFF en Hexadecimal.
- ✓ Cada dirección identifica 8 bits de almacenamiento.

Modo Protegido: (Se vera mas adelante con mas detalle)

- ✓ En este modo existe la Memoria Virtual.
- ✓ Trabaja con Memoria Principal de hasta 4GB y Memoria Virtual de hasta 64TB (a partir del 80386).
- ✓ La memoria es segmentada, con o sin paginación.
- ✓ Protección de programas.
- ✓ Capacidad de Multitareas.
- ✓ Posible Conexión a Memoria Cache.

Segmentación en modo real

Cuando el Pentium funciona en **modo real (monotarea),** compatible con el 8086, y sin tipo alguno de protección ni posibilidad de manejo de memoria virtual, un segmento queda definido básicamente por los siguientes elementos:

- Base o dirección de comienzo de 20 bits.

- Desplazamiento o tamaño de 16 bits. El tamaño máximo que se admite en este modo para mantener la compatibilidad con el 8086, es de 64 KBytes y la capacidad máxima de la memoria principal sólo es de 1 MByte.

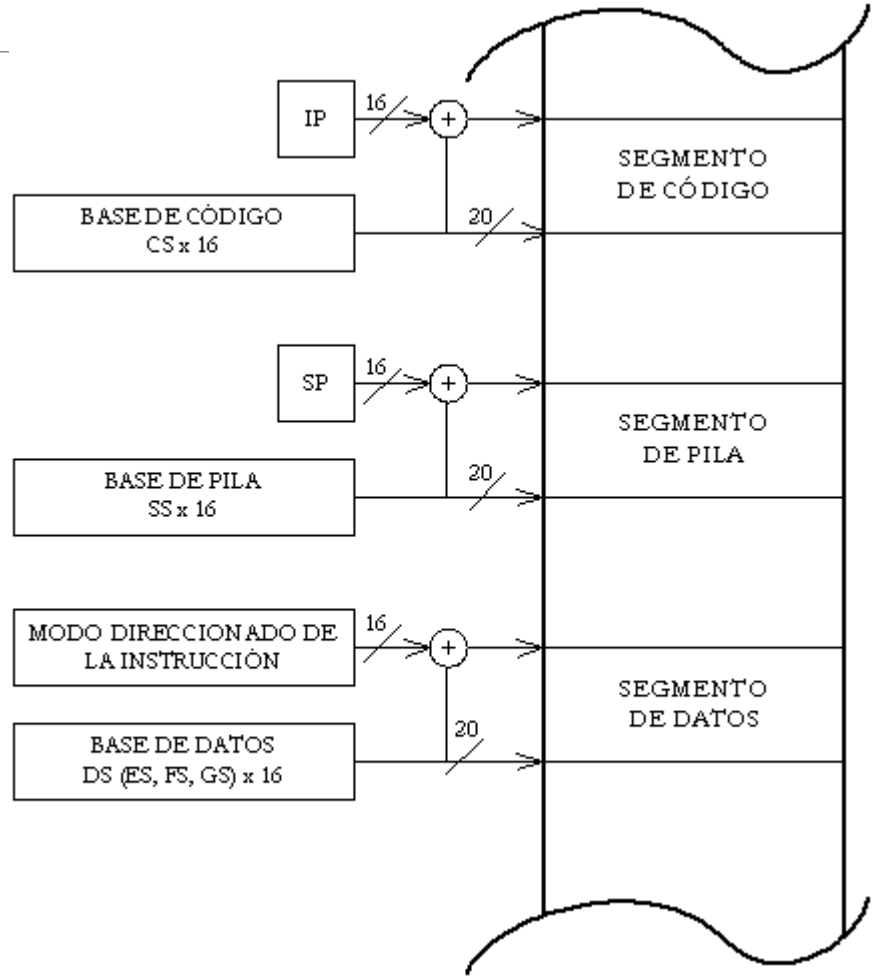

Figura 7.13 – Segmentación en modo real. En este modo se accede a elementos de memoria multiplicando por 16 el valor del registro de segmento y añadiendo un desplazamiento de 16 bits al resultado.

Segmentación en modo real

En modo real todo segmento de código, datos o pila está especificado por una dirección lógica, compuesta por dos campos de 16 bits cada uno.

- **Selector:** Referencia la base del segmento, la cual se deduce a partir del valor contenido en el registro de segmento apropiado. Como el 8086 sólo maneja una memoria de 1 MByte (2^{20}) de capacidad máxima, para obtener la base del segmento se añaden cuatro ceros a los 16 bits del registro de segmento o selector, es decir, se multiplica dicho valor binario por 16.

- **Desplazamiento:** El tamaño máximo del segmento en el 8086 es de 64 KBytes , por lo tanto bastan 16 bits para expresar el desplazamiento que hay que añadir a la base. En el caso del segmento de código, el desplazamiento lo almacena IP, que está formado por los 16 bits de menos peso de EIP. El desplazamiento correspondiente al segmento de pila está guardado en SP, y el del segmento de datos lo expresa el modo de direccionamiento de los operandos o del resultado en la instrucción en curso, por ejemplo (MOV AX, ES : 555 hex).

Segmentación en modo real

Por tanto una dirección quedaría:

Dirección efectiva/física = RS x 16 + Desplazamiento.
Esto queda especificado en la siguiente tabla. Dependiendo de si es un segmento de código, de pila o de datos, se conseguirá la base utilizando CS, SS, DS... tal y como se muestra a continuación.

	BASE	DESPLAZAMIENTO
CÓDIGO	CS X 16	IP
PILA	SS X 16	SP
DATOS	DS X 16 ES X 16 FS X 16 GS X 16	DESPLAZAMIENTO

Tabla 7.1. – Tabla para la determinación de la base y el desplazamiento de los segmentos en modo real

Segmentación en modo protegido

Cuando el Pentium trabaja en modo protegido (multitarea), un segmento queda caracterizado por tres parámetros fundamentales que son comprobados automáticamente por el sistema de protección cada vez que se utiliza. Dichos parámetros son:

Base, es la dirección lineal donde comienza el segmento. Esta formada por 32 bits, que es la longitud de la dirección de la memoria física que puede alcanzar un tamaño máximo de 2^{32} = 4 GBytes.

Límite, consta de 20 bits que determinan con exactitud el tamaño del segmento usado por el programador y en el que residen informaciones válidas. Si está expresado en bytes, el límite máximo sería de 2^{20} = 1 MByte y si está expresado en páginas de 4 KByte, un segmento puede ser tan grande como la memoria principal, es decir 4 GBytes.

Atributos o derechos de acceso, se trata de un campo de 12 bits, que proporciona las características relevantes del segmento como:

- Tipo de segmento, admitiendo las variantes de legible, escribible, ejecutable o una combinación de estos.
- Nivel de privilegio, que oscila entre 0 y 3. Es el grado de seguridad que tiene el contenido del segmento en el sistema.
- Indicadores sobre aspectos relacionados con la gestión de la memoria virtual, como el que indica si el segmento se halla cargado o no en la memoria física.

Segmentación en modo protegido

Descriptor de segmento:
Se denomina **descriptor de segmento** al conjunto de los 2 parámetros base (32 bits), límite (20 bits) y atributos (12 bits).

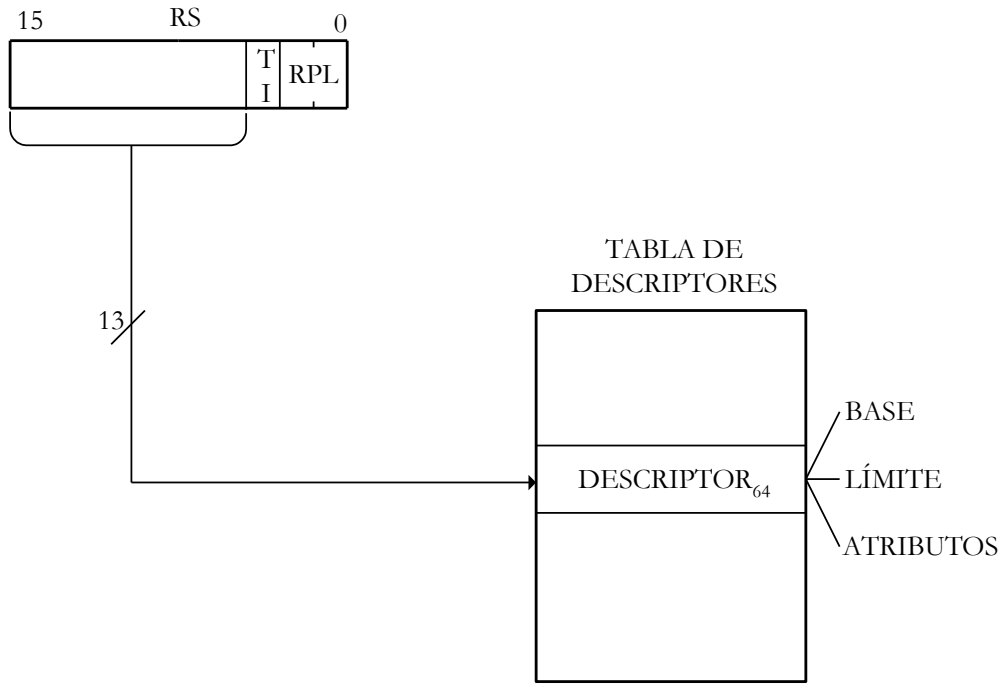

Figura 7.14 – Se accede a la tabla de descriptores consultado el selector y comprobando el índice de tabla.

Segmentación en modo protegido

Descriptor de segmento:

Los descriptores de segmento se guardan en una “tabla de descriptores” que residen en la MP y se crean y manejan desde el SO.

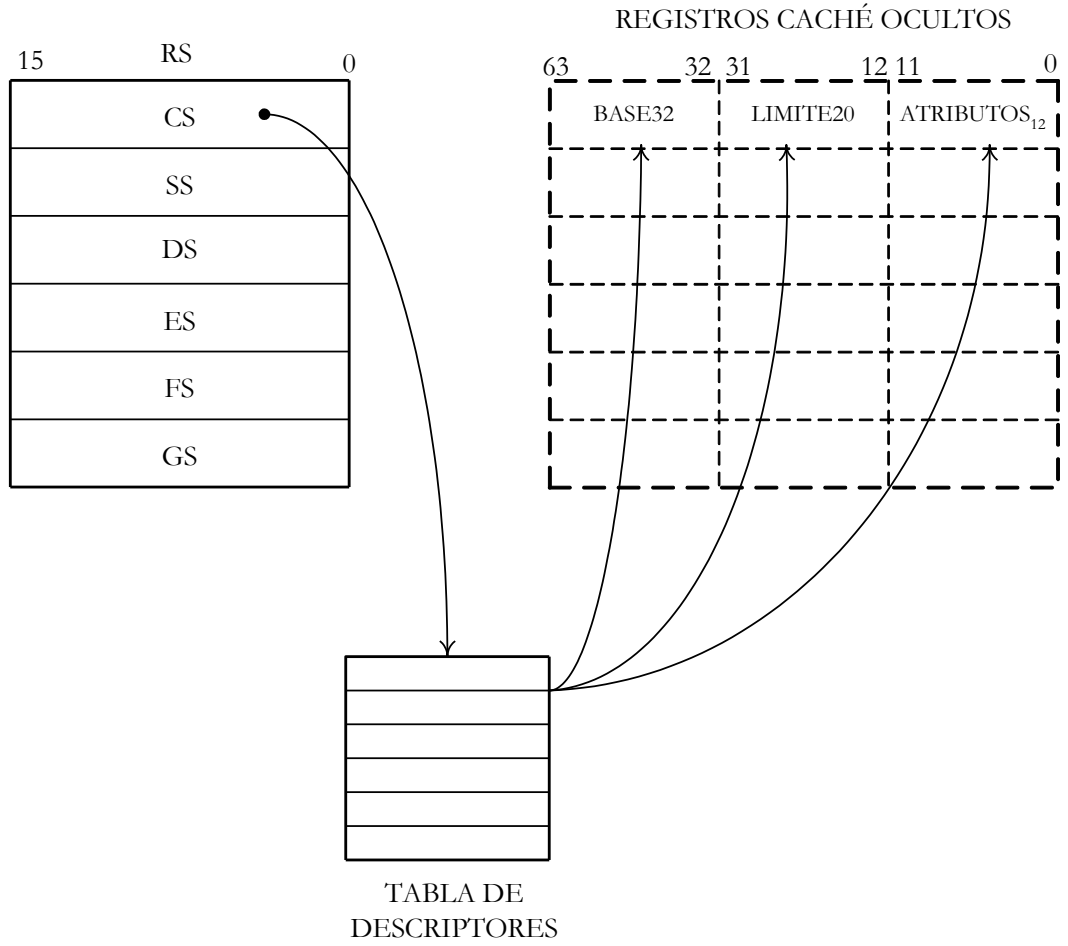

Figura 7.15 – En modo protegido, los registros de segmento actúan como selectores “visibles”, con los que se accede a tablas en las que se hallan los parámetros que definen al segmento, es decir, la base, el límite y los atributos. Estos parámetros los carga automáticamente la CPU en los registros caché asociados, que son invisibles, es decir, inaccesibles al programador de sistemas y al de aplicaciones.

Gracias