Introducción a los Sistemas Operativos / Conceptos de Sistemas Operativos

Administración de Memoria – I

Profesores:
Lía Molinari
Juan Pablo Pérez
Nicolás del Rio

Facultad de Informática
UNIVERSIDAD NACIONAL DE LA PLATA

ISO / CSO

- ☑ Versión: Septiembre 2019
- ☑ Palabras Claves: Procesos, Espacio de Direcciones, Memoria, Seguridad, Paginación, Segmentación Fragmentación

Algunas diapositivas han sido extraídas de las ofrecidas para docentes desde el libro de Stallings (Sistemas Operativos) y el de Silberschatz (Operating Systems Concepts). También se incluyen diapositivas cedidas por Microsoft S.A.

Memoria

- La organización y administración de la *“memoria principal* es uno de los factores más importantes en el diseño de los S. O.
- Los programas y datos deben estar en el almacenamiento principal para:
  - Poderlos ejecutar.
  - Referenciarlos directamente.

Memoria *(cont.)*

- El SO debe:
  - Llevar un registro de las partes de memoria que se están utilizando y de aquellas que no.
  - Asignar espacio en memoria principal a los procesos cuando estos la necesitan.
  - Libera espacio de memoria asignada a procesos que han terminado.
- Se espera de un S.O. un uso eficiente de la memoria con el fin de alojar el mayor número de procesos

Memoria *(cont.)*

- El S.O. debe:
  - Lograr que el programador se abstraiga de la alocación de los programas
  - Brindar seguridad entre los procesos para que unos no accedan a secciones privadas de otros
  - Brindar la posibilidad de acceso compartido a determinadas secciones de la memoria (librerías, código en común, etc.)
  - Garantizar la performance del sistema

Administración de Memoria

- División Lógica de la Memoria Física para alojar múltiples procesos
  - Garantizando protección
  - Depende del mecanismo provisto por el HW
- Asignación eficiente
  - Contener el mayor numero de procesos para garantizar el mayor uso de la CPU por los mismos

Requisitos

- Reubicación
  - El programador no debe ocuparse de conocer donde será colocado en la Memoria RAM
  - Mientras un proceso se ejecuta, puede ser sacado y traído a la memoria (swap) y, posiblemente, colocarse en diferentes direcciones.
  - Las referencias a la memoria se deben “traducir” según ubicación actual del proceso.

Requisitos (cont).

- Protección
  - Los procesos NO deben referenciar – acceder - a direcciones de memoria de otros procesos
    - Salvo que tengan permiso
  - El chequeo se debe realizar durante la ejecución:
    - NO es posible anticipar todas las referencias a memoria que un proceso puede realizar.

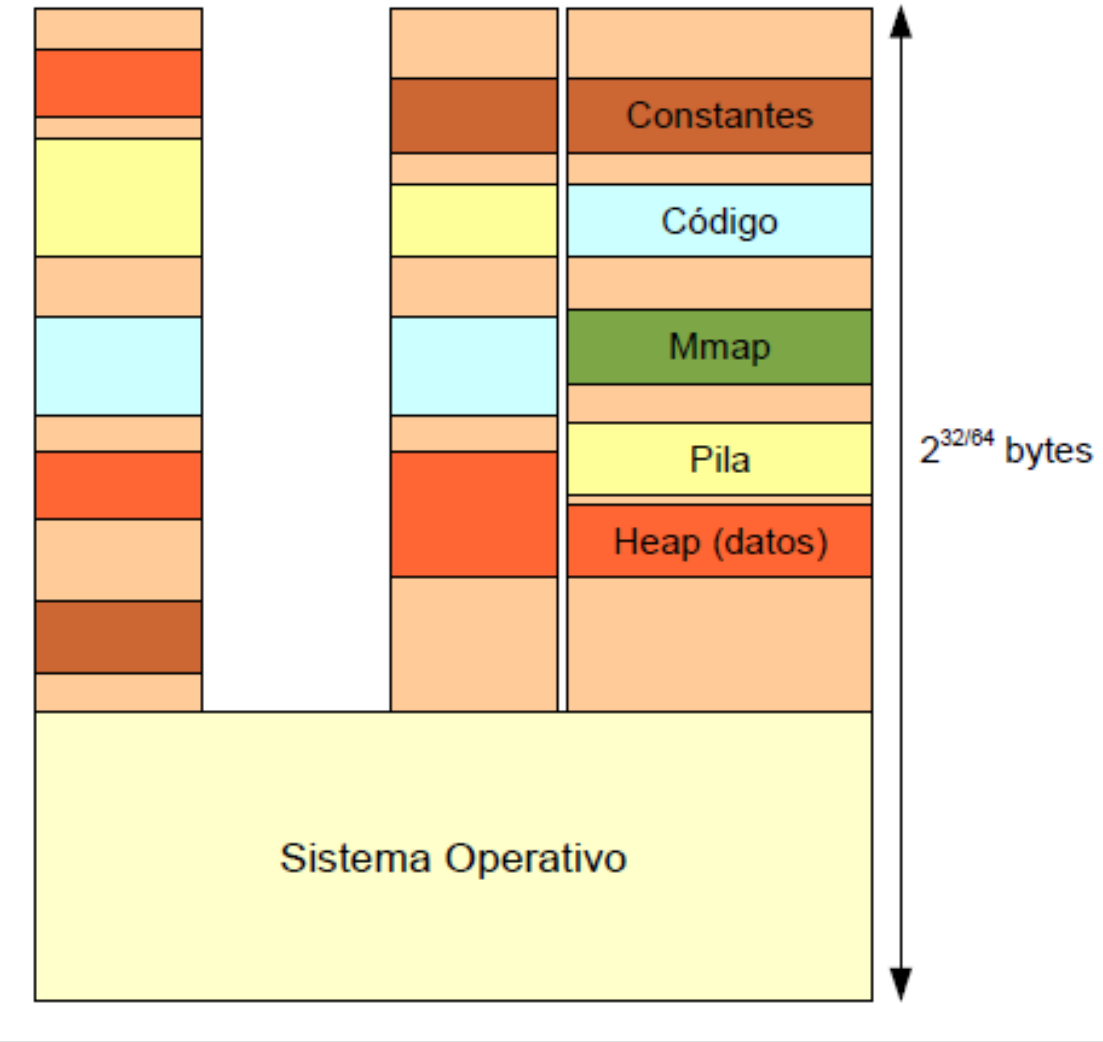
Requisitos (cont).

- Compartición
  - Permitir que varios procesos accedan a la misma porción de memoria.
    - Ej: Rutinas comunes, librerías, espacios explícitamente compartidos, etc.
  - Permite un mejor uso – aprovechamiento - de la memoria RAM, evitando copias innecesarias (repetidas) de instrucciones

Abstracción - Espacio de Direcciones

- Rango de direcciones (a memoria) posibles que un proceso puede utilizar para direccionar sus instrucciones y datos.
- El tamaño depende de la Arquitectura del Procesador
  - 32 bits: 0 .. 2^{32} - 1
  - 64 bits: 0 .. 2^{64} – 1
- Es independiente de la ubicación “real” del proceso en la Memoria RAM

Abstracción -Espacio de Direcciones (cont.)



Direcciones

- Lógicas
  - Referencia a una localidad de memoria independiente de la asignación actual de los datos en la memoria.
  - Representa una dirección en el “Espacio de Direcciones del Proceso”
- Físicas
  - Referencia una localidad en la Memoria Física (RAM)
    - Dirección absoluta

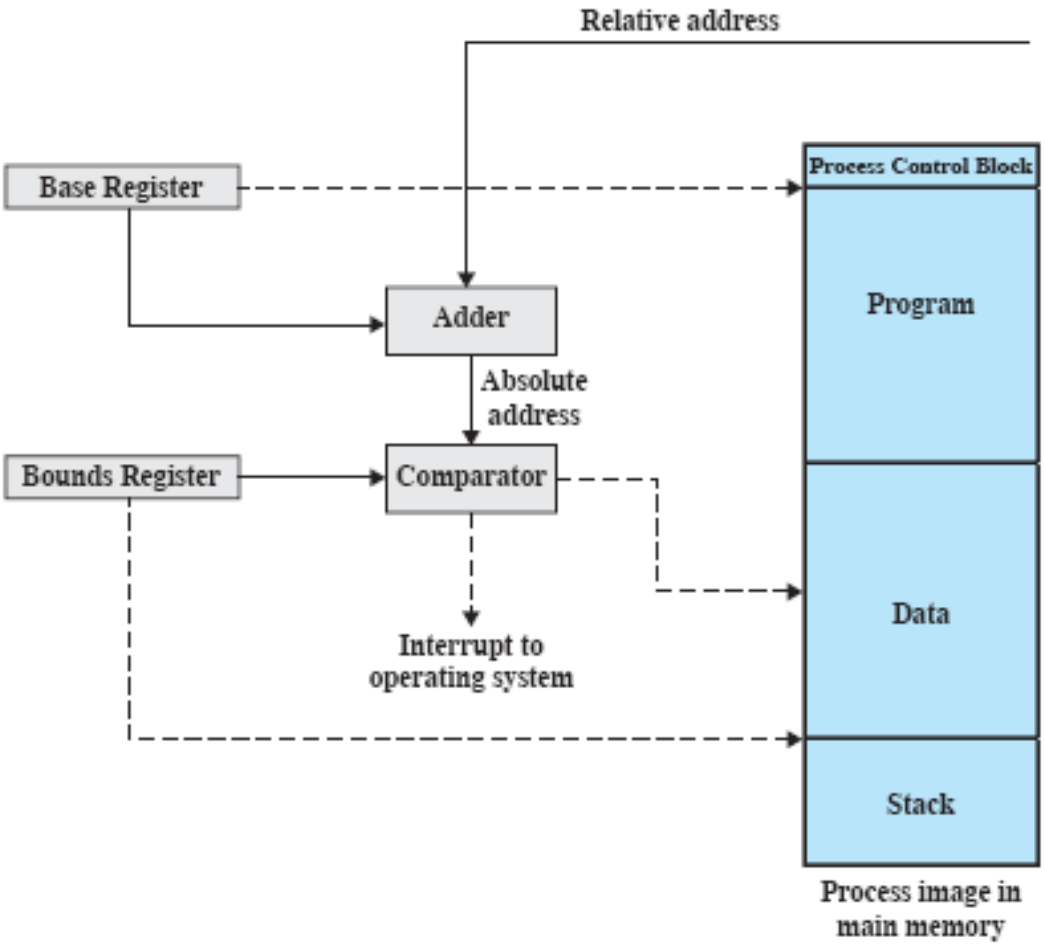
En caso de usar direcciones Lógicas, es necesaria algún tipo de conversión a direcciones Físicas.

Conversión de Direcciones

Una forma simple de hacer esto es utilizando registros auxiliares

- Registro Base
  - Dirección de comienzo del Espacio de Direcciones del proceso en la RAM
- Registro Limite
  - Dirección final del proceso o medida del proceso – Tamaño de su Espacio de Direcciones
- Ambos valores se fijan cuando el espacio de direcciones del proceso es cargado a memoria.
- Varían entre procesos (Context Switch)

Direcciones (cont.)



Dir. Lógicas vs. Físicas

- Si la CPU trabaja con direcciones lógicas, para acceder a memoria principal, se deben transformar en direcciones físicas.
  - Resolución de direcciones (address-binding): transformar la dirección lógica en la dirección física correspondiente

- Resolución en momento de compilación (Archivos .com de DOS) y en tiempo de carga
  - Direcciones Lógicas y Físicas son idénticas
  - Para reubicar un proceso es necesario recompilarlo o recargarlo.

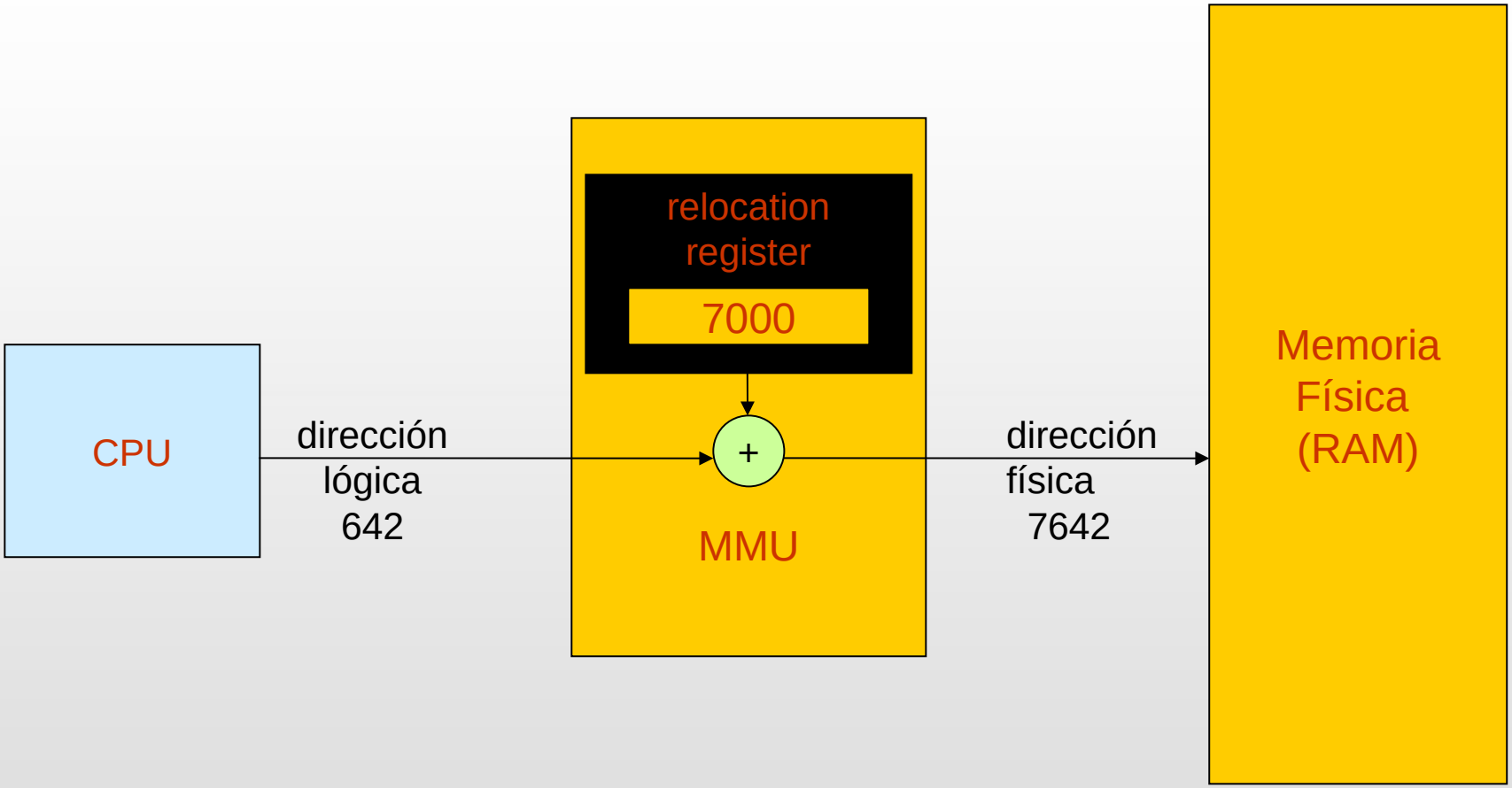
Dir. Lógicas vs. Físicas

- Resolución en tiempo de ejecución
  - Direcciones Lógicas y Físicas son diferentes
  - Direcciones Lógicas son llamadas “Direcciones Virtuales”
  - La reubicación se puede realizar fácilmente
  - El mapeo entre “Virtuales” y “Físicas” es realizado por hardware
    - Memory Management Unit (MMU)

Memory Management Unit (MMU)

- Dispositivo de Hardware que mapea direcciones virtuales a físicas
  - Es parte del Procesador
  - Re-programar el MMU es una operación privilegiada
    - solo puede ser realizada en Kernel Mode
- El valor en el “registro de realocación” es sumado a cada dirección generada por el proceso de usuario al momento de acceder a la memoria.
  - Los procesos nunca usan direcciones físicas

MMU


CPU
dirección
lógica
642
relocation
register
7000
+
MMU
dirección
física
7642
Memoria
Física
(RAM)

Mecanismos de asignación de memoria

- Particiones Fijas: El primer esquema implementado
  - La memoria se divide en particiones o regiones de tamaño Fijo (pueden ser todas del mismo tamaño o no)
  - Alojan un proceso cada una
  - Cada proceso se coloca de acuerdo a algún criterio (First Fit, Best Fit, Worst Fit, Next Fit) en alguna partición
- Particiones dinámicas: La evolución del esquema anterior
  - Las particiones varían en tamaño y en número
  - Alojan un proceso cada una
  - Cada partición se genera en forma dinámica del tamaño justo que necesita el proceso

¿Qué problemas se generan en cada caso?

Fragmentación

- La fragmentación se produce cuando una localidad de memoria no puede ser utilizada por no encontrarse en forma contigua
- **Fragmentación Interna:**
  - Se produce en el esquema de particiones Fijas
  - Es la porción de la partición que queda sin utilizar
- **Fragmentación Externa:**
  - Se produce en el esquema de particiones dinámicas
  - Son huecos que van quedando en la memoria a medida que los procesos finalizan
  - Al no encontrarse en forma contigua puede darse el caso de que tengamos memoria libre para alocar un proceso, pero que no la podamos utilizar
  - Para solucionar el problema se puede acudir a la compactación, pero es muy costosa

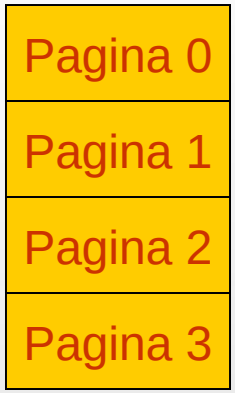
Problemas del esquema

- El esquema de Registro Base + Limite presenta problemas:
  - Necesidad de almacenar el Espacio de Direcciones de forma continua en la Memoria Física
  - Los primeros SO definían particiones fijas de memoria, luego evolucionaron a particiones dinámicas
  - Fragmentación
  - Mantener “partes” del proceso que no son necesarias
  - Los esquemas de particiones fijas y dinámicas no se usan hoy en día
- Solución:
  - Paginación
  - Segmentación

Paginación

- Memoria Física es dividida lógicamente en pequeños trozos de igual tamaño ➔ **Marcos**
- Memoria Lógica (espacio de direcciones) es dividido en trozos de igual tamaño que los marcos ➔ **Paginas**
- El SO debe mantener una tabla de paginas por cada proceso, donde cada entrada contiene (entre otras) el **Marco** en la que se coloca cada pagina.
- La dirección lógica se interpreta como:
  - un numero de pagina y un desplazamiento dentro de la misma.

Paginación – Ejemplo I



Memoria
Lógica
(Espacio de Direcciones)

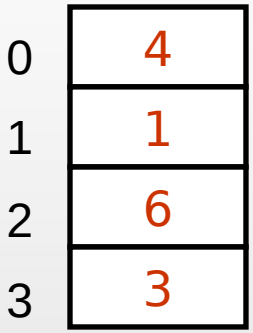
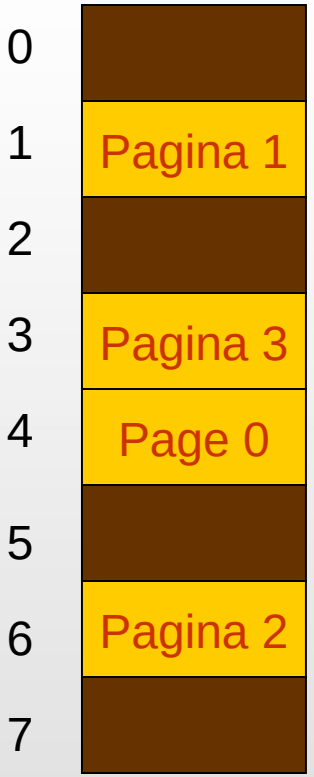


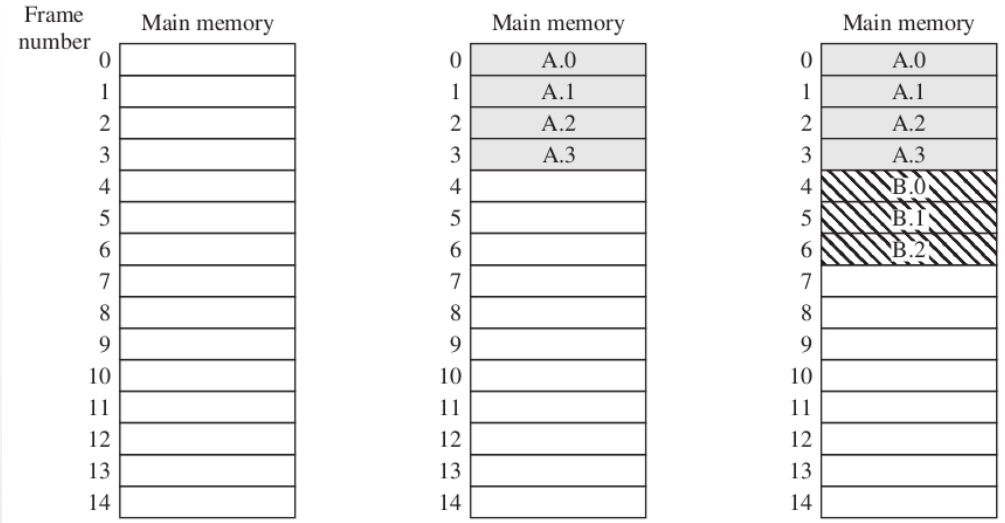
Tabla de
Paginas

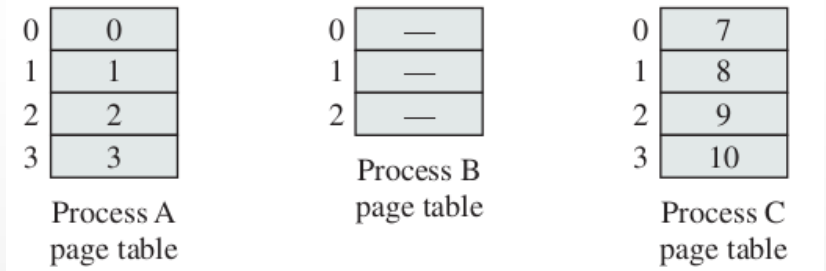


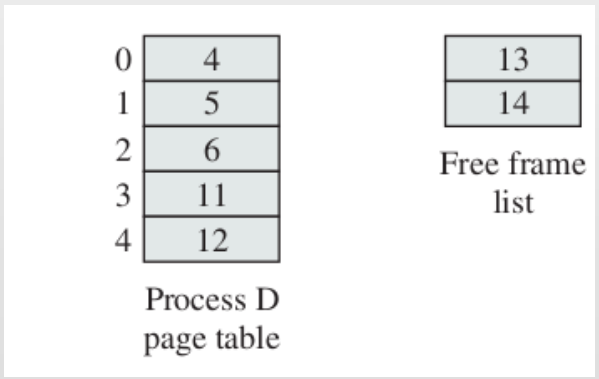
Memoria Física
(RAM)

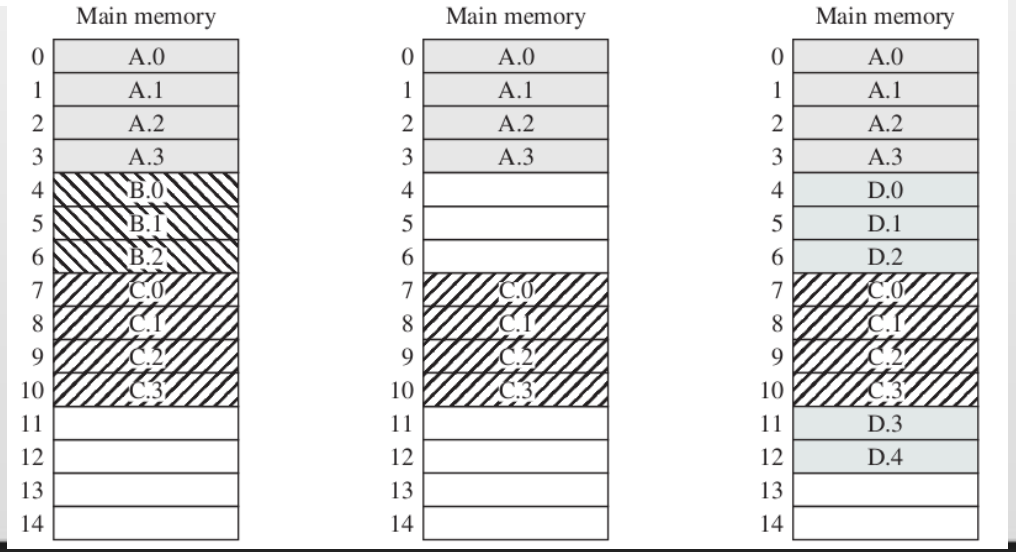
Pensar: ¿Esta técnica puede producir Fragmentación?

Paginación – Ejemplo II

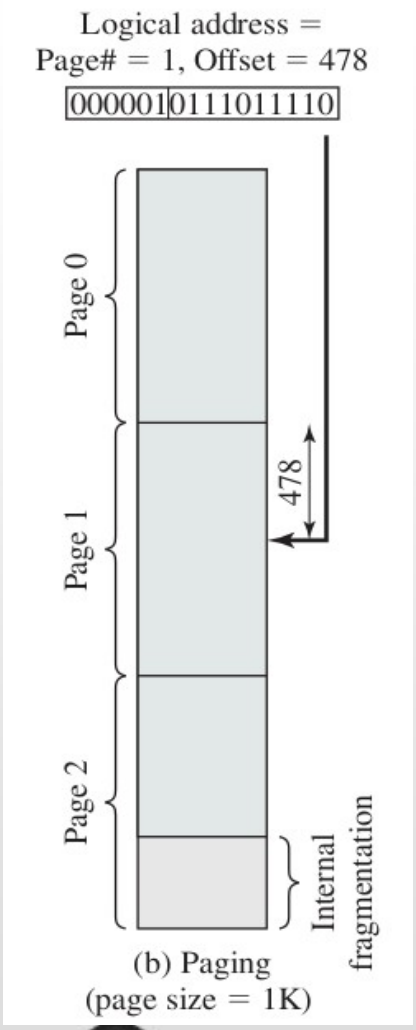


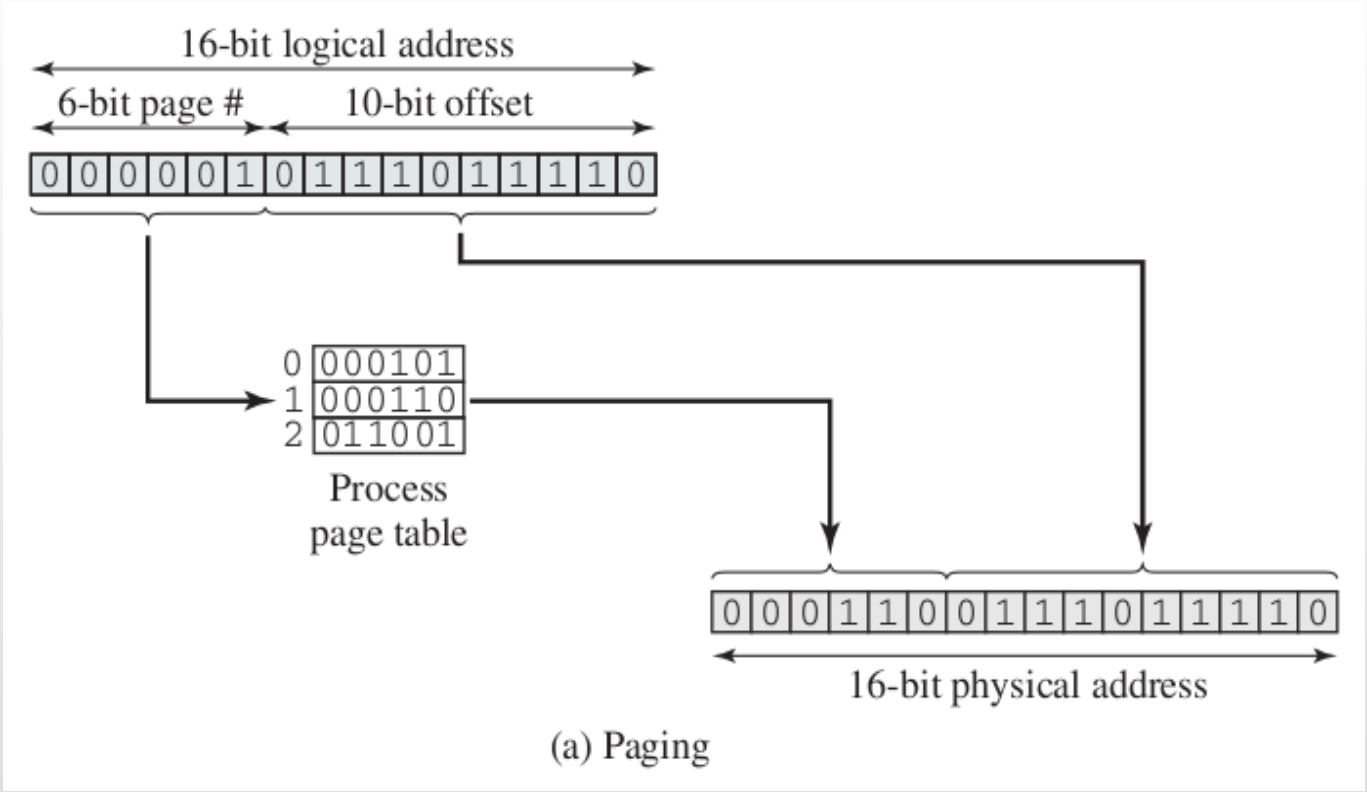






Paginación – Direcciones Lógicas





Traducción de direcciones

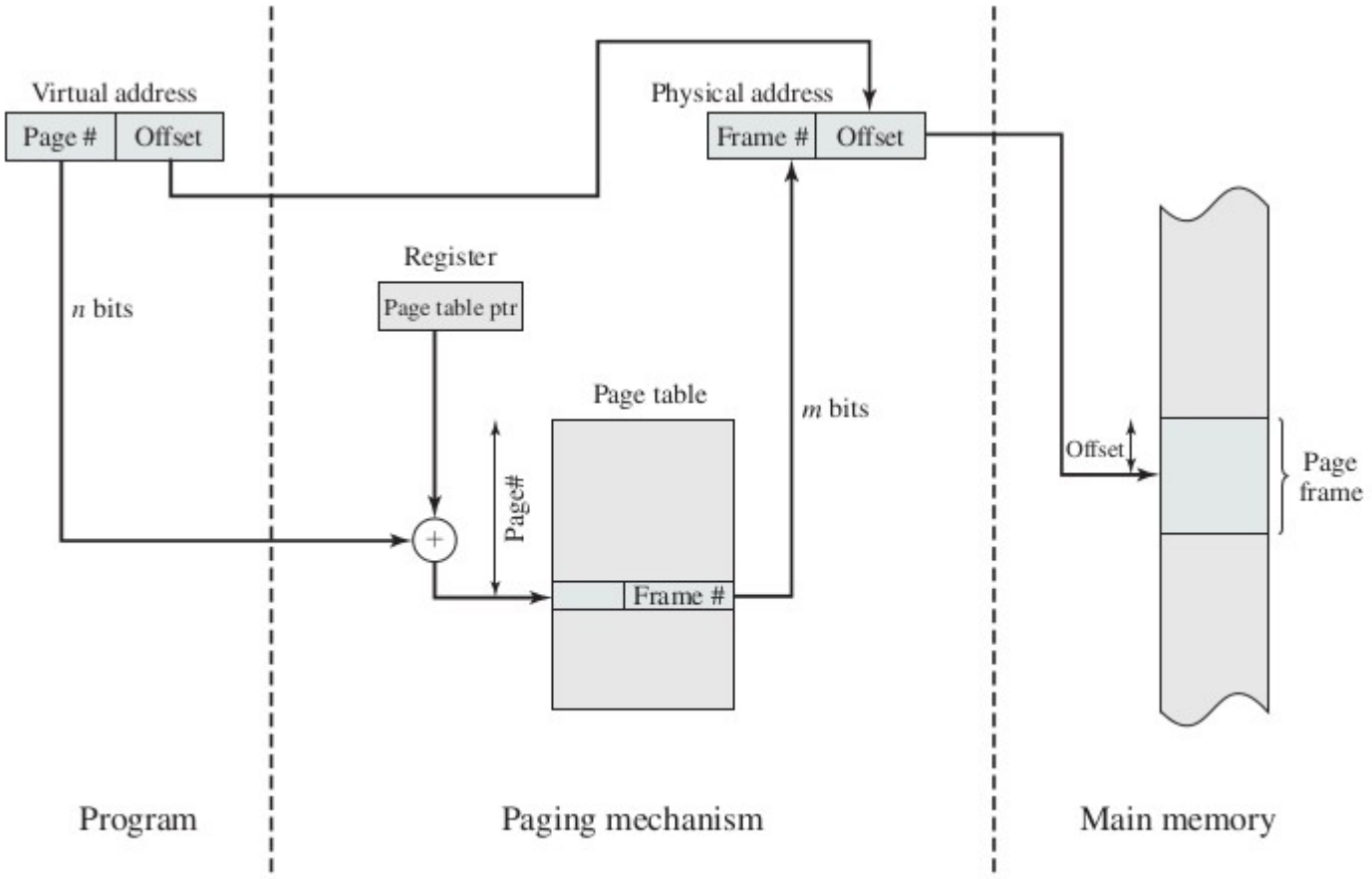
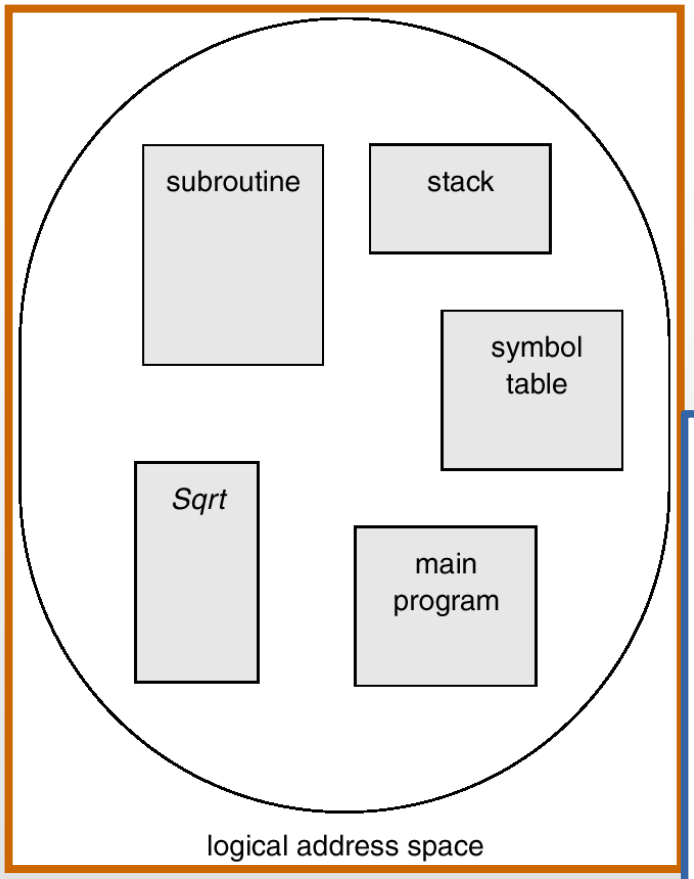


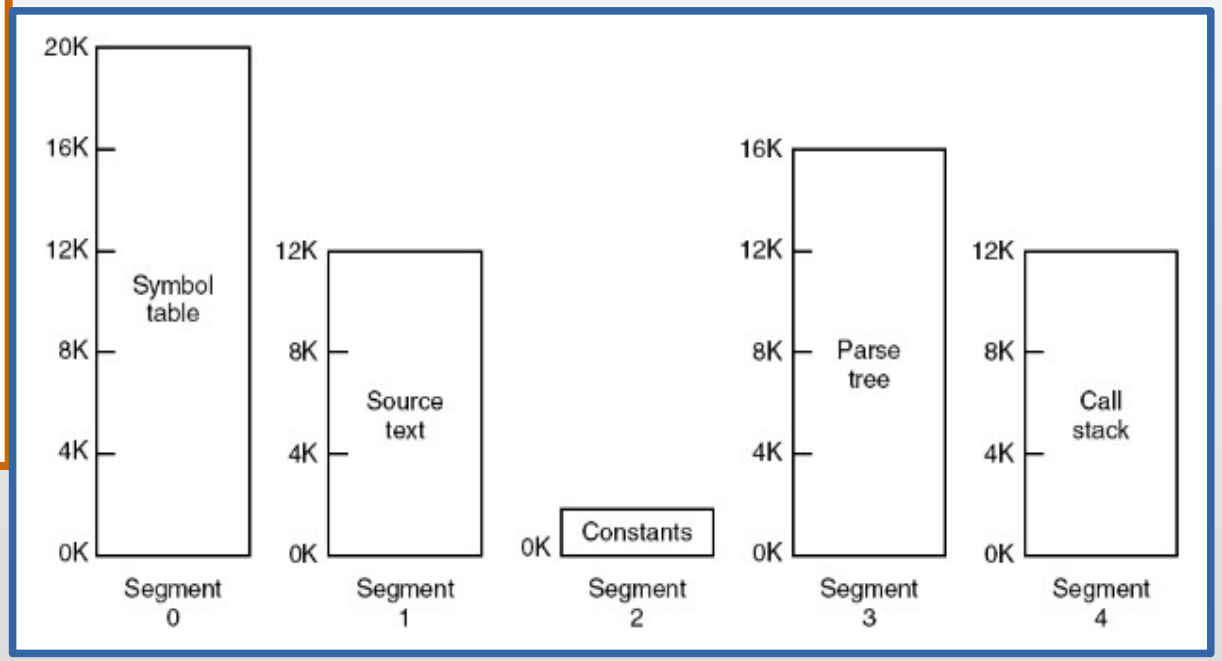
Figure 8.3 **Address Translation in a Paging System**

Segmentación

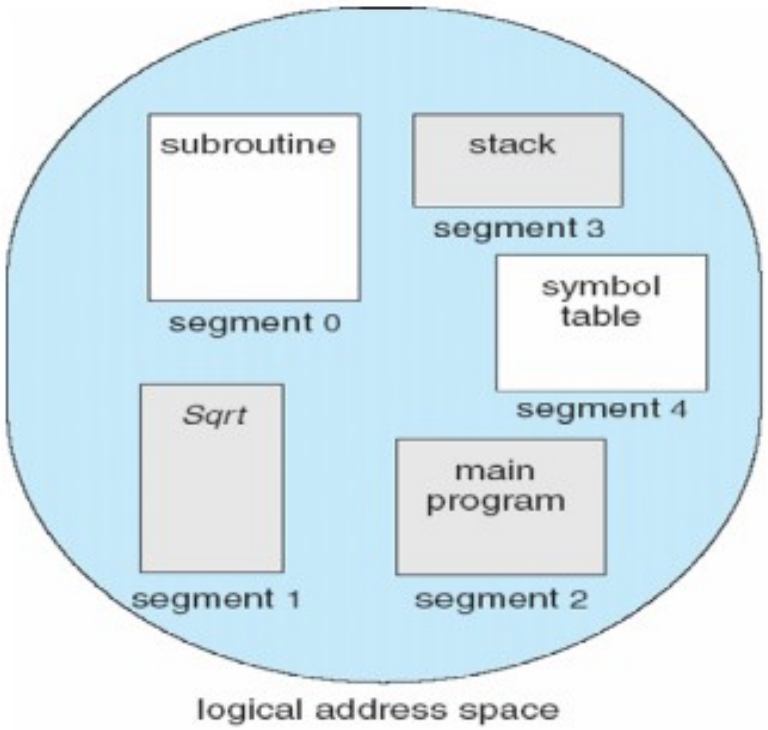
- Esquema que se asemeja a la “visión del usuario”. El programa se divide en partes/secciones
- Un programa es una colección de segmentos. Un segmento es una unidad lógica como:
  - Programa Principal, Procedimientos y Funciones, variables locales y globales, stack, etc.
- Puede causar Fragmentación

Programa desde la visión del usuario



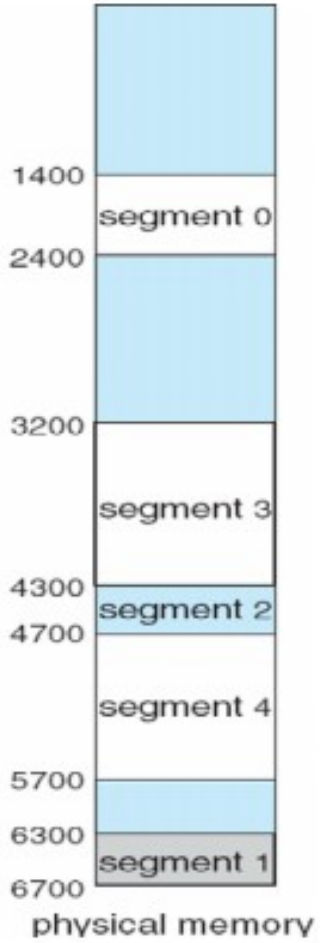


Ejemplo de Segmentación



	limit	base
0	1000	1400
1	400	6300
2	400	4300
3	1100	3200
4	1000	4700

segment table



Pensar: ¿Esta técnica puede producir Fragmentación?

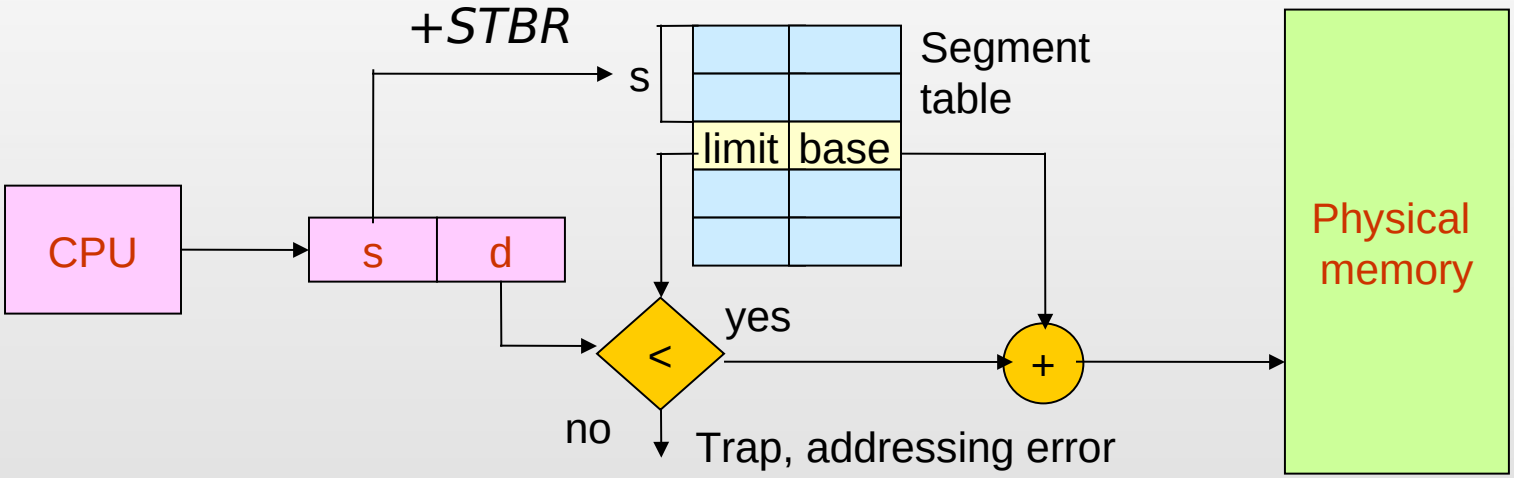
Segmentación (cont.)

- Todos los segmentos de un programa pueden no tener el mismo tamaño (código, datos, rutinas).
- Las direcciones Lógicas consisten en 2 partes:
  - Selector de Segmento
  - Desplazamiento dentro del segmento

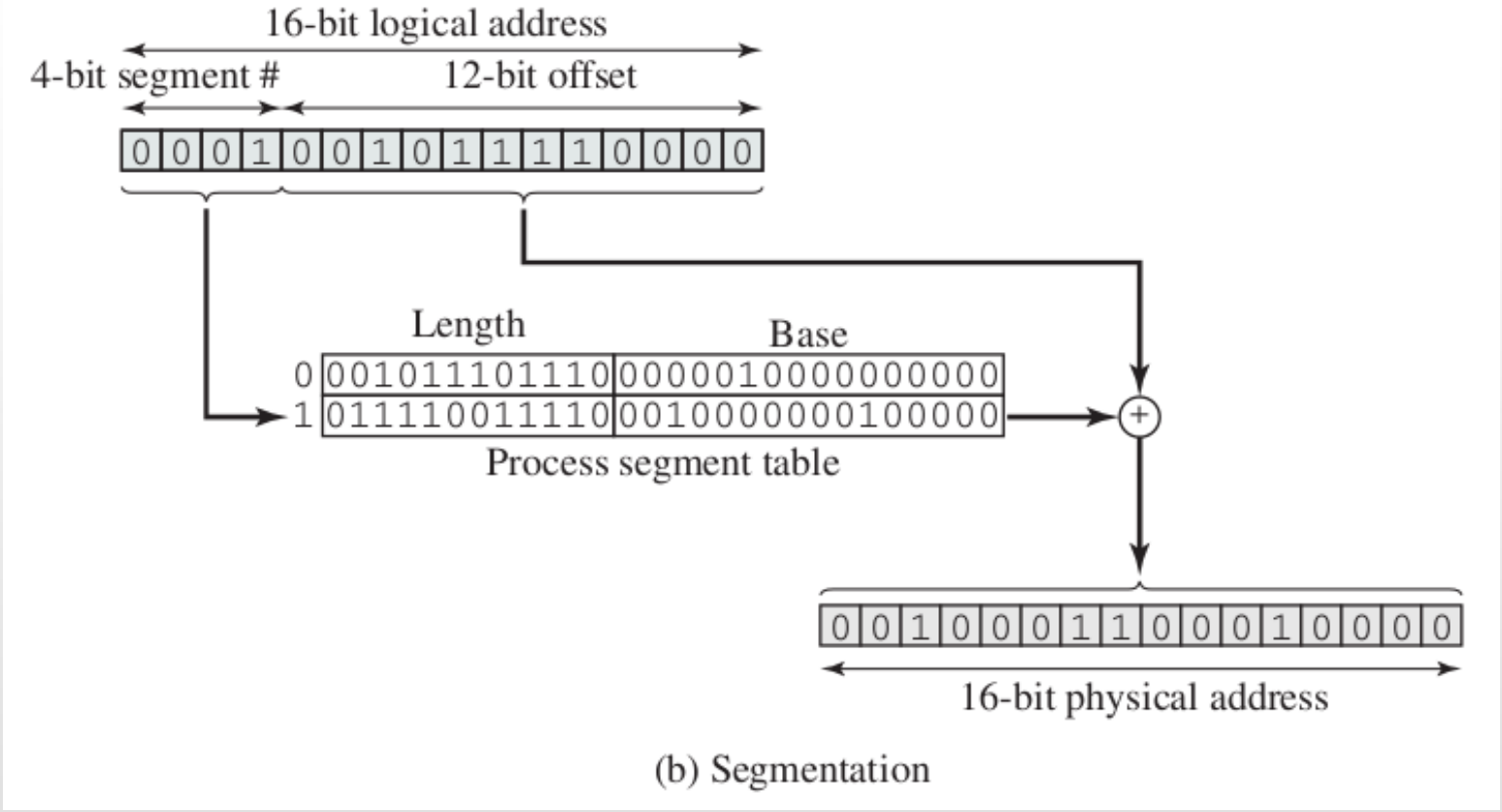
Segmentación (cont.) - Arquitectura

- *Tabla de Segmentos*
  - *Permite mapear la dirección lógica en física. Cada entrada contiene:*
    - *Base: Dirección física de comienzo del segmento*
    - *Limit: Longitud del Segmento*
- *Segment-table base register (STBR)*: apunta a la ubicación de la tabla de segmentos.
- *Segment-table length register (STLR)* : cantidad de segmentos de un programa

Segmentación (cont.)



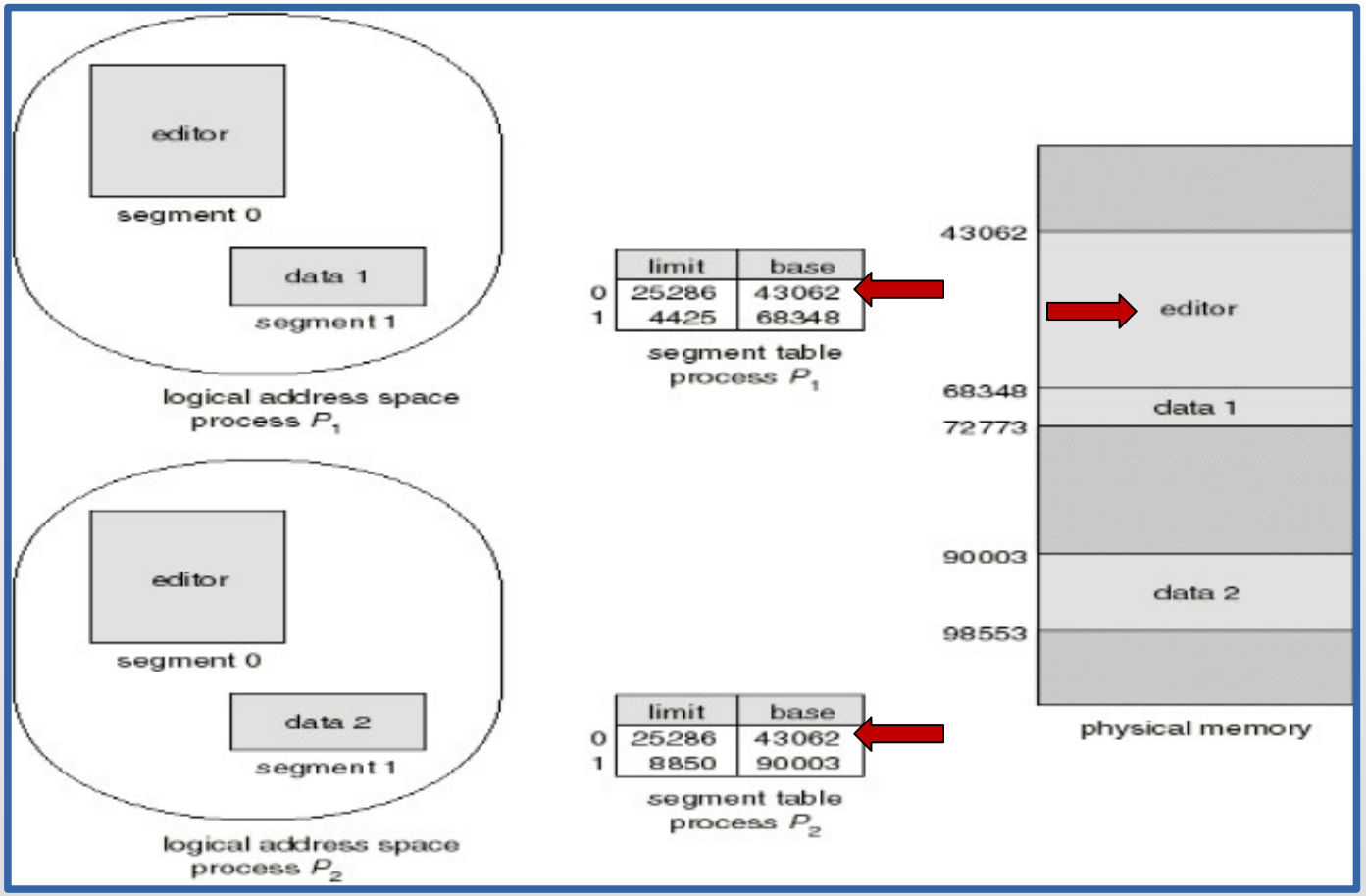
Segmentación - Direcciones (cont.)



(b) Segmentation

Ventajas sobre Segmentación

- ☑ *Compartir*
- ☑ *Proteger*



Segmentación Paginada

- La paginación
  - Transparente al programador
  - Elimina Fragmentación externa.
- Segmentación
  - Es visible al programador
  - Facilita modularidad, estructuras de datos grandes y da mejor soporte a la compartición y protección
- Segmentación Paginada: Cada segmento es dividido en paginas de tamaño fijo.

Segmentación Paginada (cont.)

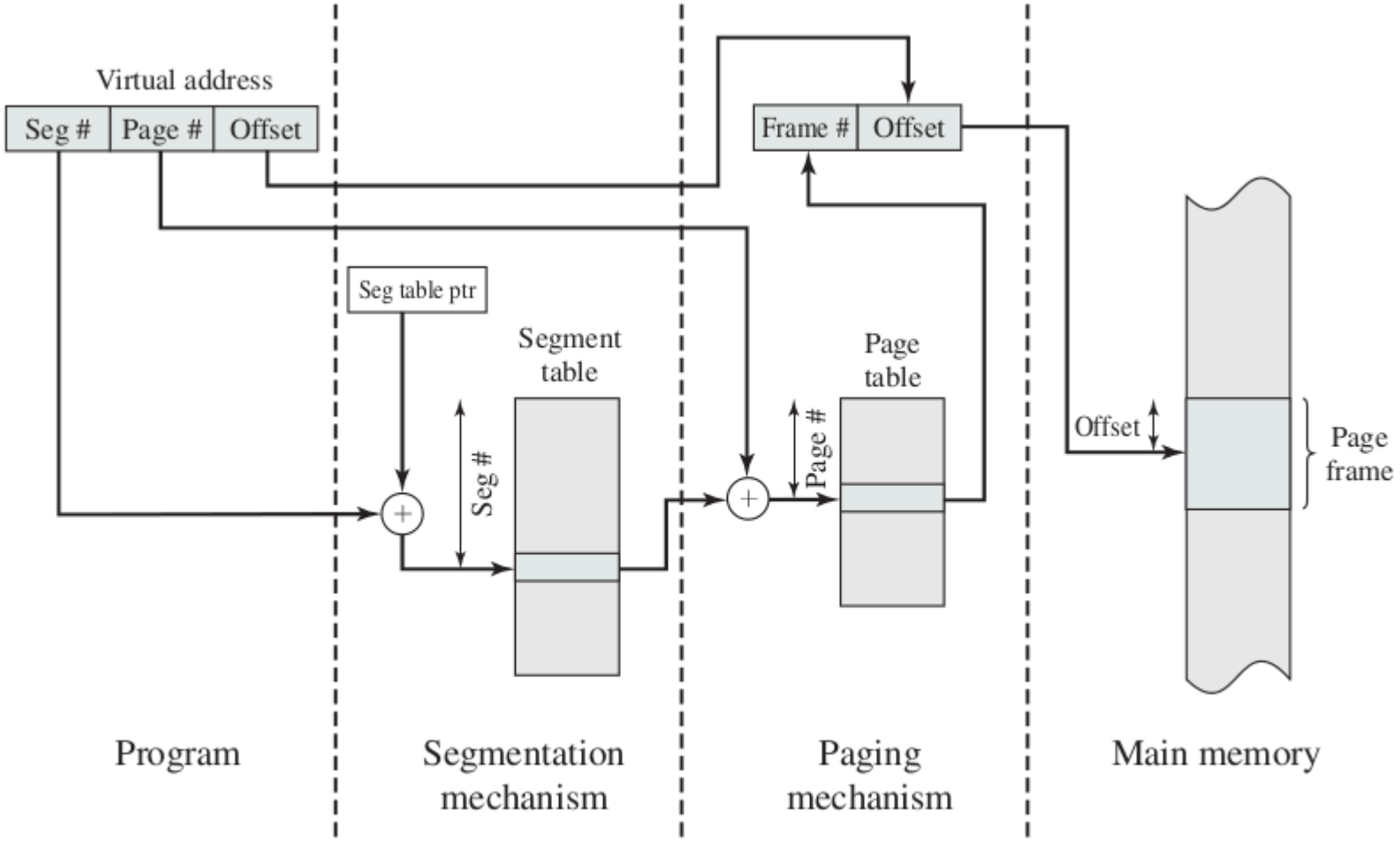
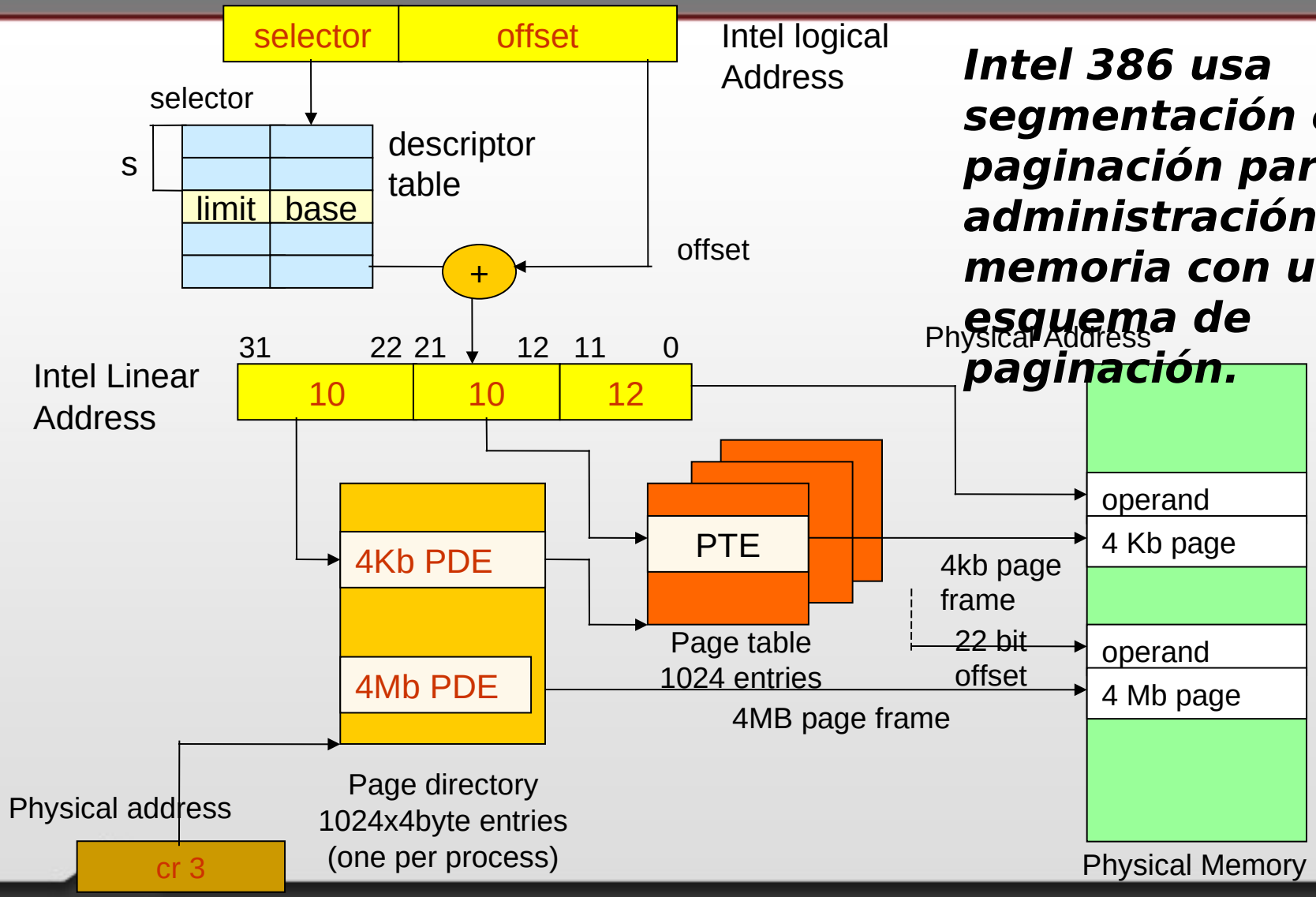


Figure 8.13 **Address Translation in a Segmentation/Paging System**

Intel x386



Intel 386 usa segmentación con paginación para la administración de la memoria con un doble esquema de paginación.