Introducción a los Sistemas Operativos

Repaso
Administración de Memoria

Administración de Memoria

- División Lógica de la Memoria para alojar múltiples procesos
- La Memoria debe ser asignada eficientemente para contener el mayor numero de procesos como sea posible.
- Cuanto más procesos estén en memoria, más procesos competirán por la CPU. Mas probabilidad que la CPU no este ociosa.

Requisitos

- Reubicación
  - El programador no debe ocuparse de conocer donde será colocado el programa para ser ejecutado.
  - Mientras un proceso se ejecuta, puede ser sacado y traído a la memoria (swap) y colocarse en diferentes lugares.
  - Las referencias a la memoria se deben traducir según dirección actual del proceso.

Requisitos (cont).

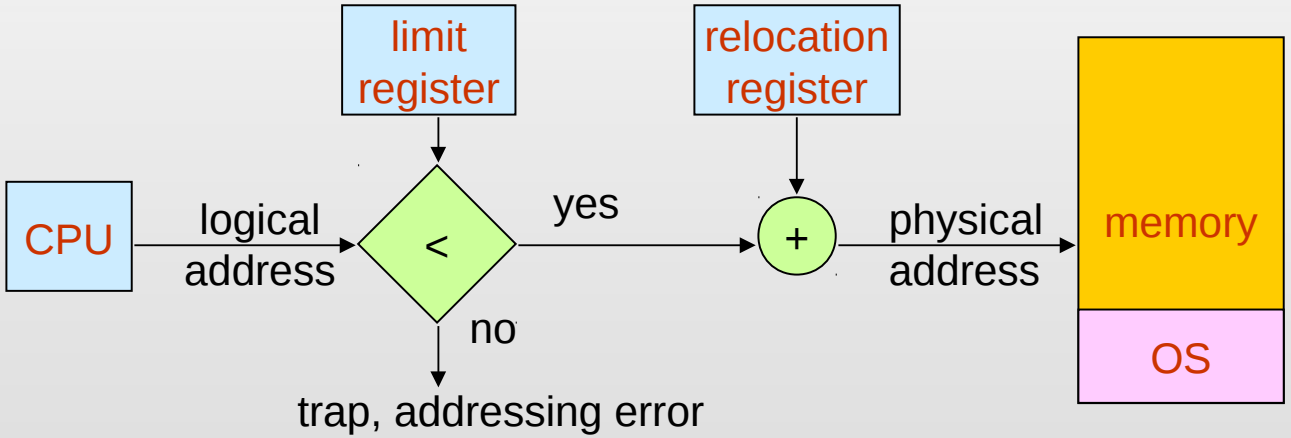
- Protección
  - Los procesos no deben ser capaces de hacer referencias a direcciones de memoria de otros procesos (salvo que tengan permiso)
  - El chequeo se debe realizar durante la ejecución:
    - El SO no puede anticipar todas las referencias a memoria que un proceso puede realizar.

Requisitos (cont).

- Compartición
  - Permitir que varios procesos accedan a la misma porción de memoria.
    - Ej: Rutinas comunes, librerías, espacios explícitamente compartidos, etc.
  - Lleva a un mejor uso de la memoria, evitando copias innecesarias de instrucciones

Única Partición

- Única Partición:
  - Los procesos ocupan una única partición de memoria
  - La protección se implementa por un “limite” y un registro de “reubicación”



Múltiples Particiones

- La memoria es dividida en varias regiones (particiones).
- Los procesos (su espacio de direcciones) se colocan en las particiones según su tamaño.
- Técnicas:
  - Particiones Fijas del mismo tamaño
  - Particiones Fijas de distinto tamaño
  - Particiones Dinámicas

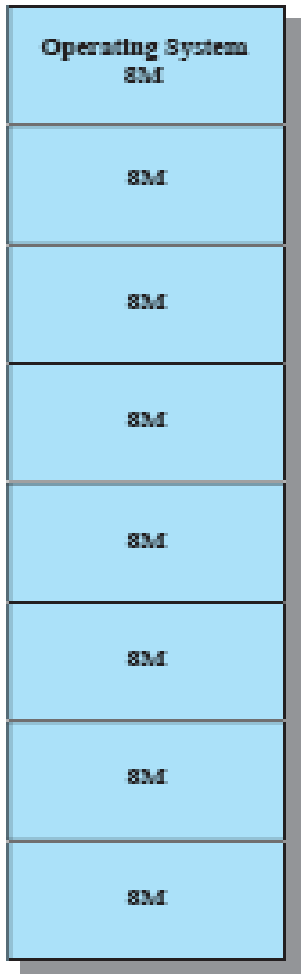
Particiones Fijas del mismo tamaño

- ☑ Hay fragmentación interna
- ☑ No Hay fragmentación externa
- ☑ Si el proceso es mas grande que el tamaño de la partición, no se puede ejecutar
- ☑ Procesos pequeños causarán MUCHA fragmentación interna

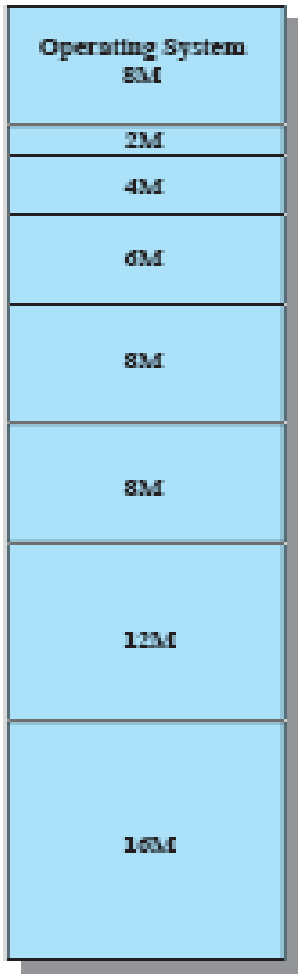
Particiones Fijas de distinto tamaño

- ☑ Particiones de DIFERENTE tamaño
- ☑ Procesos pequeños pueden usar particiones pequeñas
- ☑ Procesos grandes tiene la posibilidad de ejecutarse
- ☑ Hay Fragmentación interna.
- ☑ No hay Fragmentación externa
- ☑ Se necesita de un algoritmo de selección de partición para determinar donde alojar procesos nuevos.

Ejemplo de Particiones Fijas



(a) Equal-size partitions

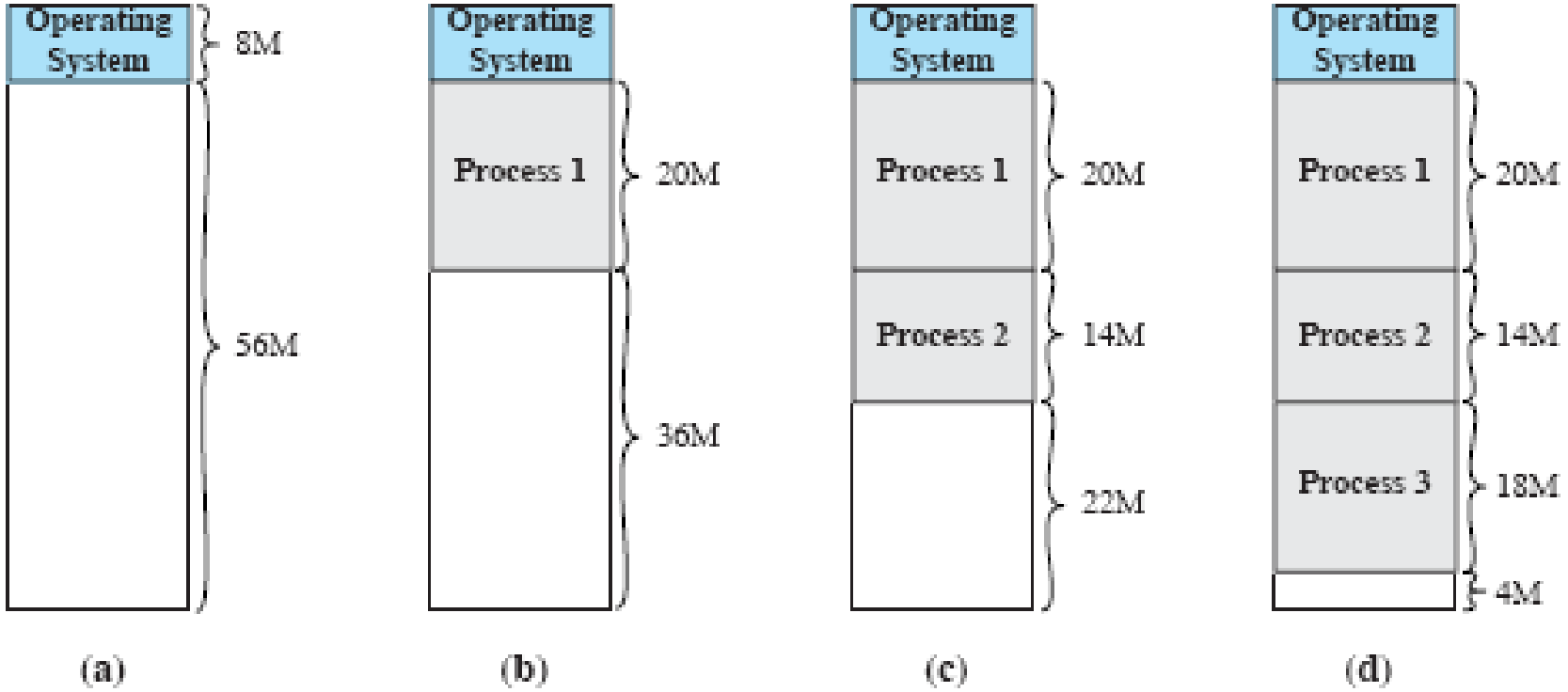


(b) Unequal-size partitions

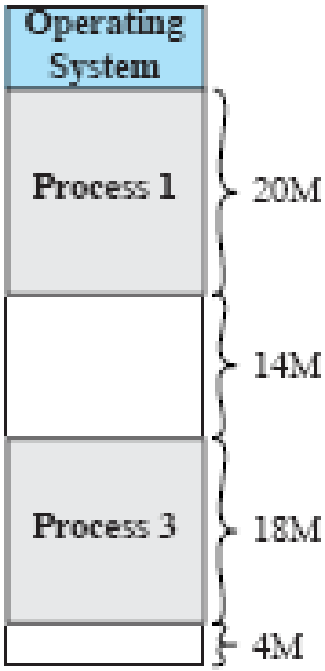
Particiones Dinámicas

- Los procesos son colocados exactamente en particiones de igual a su tamaño (generadas dinámicamente)
- No hay fragmentación interna
- Hay fragmentación externa
- Es necesario administrar la memoria para saber en cualquier momento, cuales porciones están siendo usadas y cuales libres.

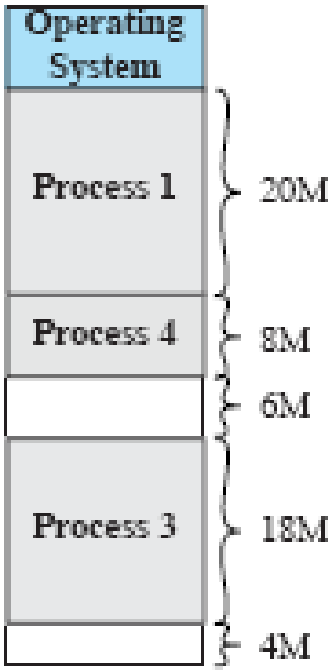
Particiones Dinámicas (cont.)



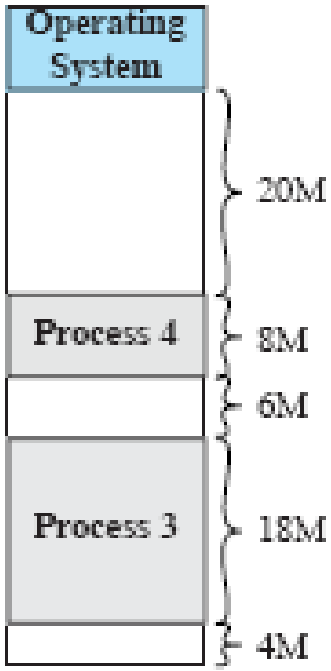
Particiones Dinámicas (cont.)



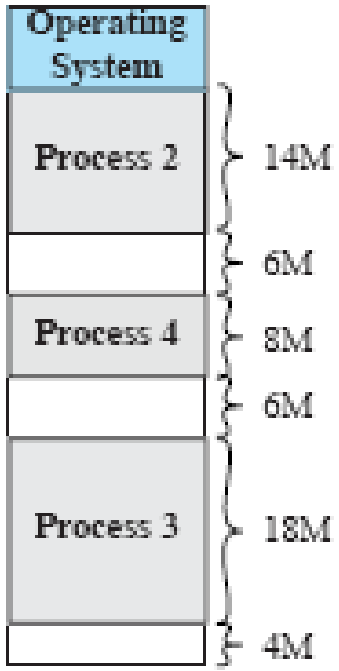
(e)



(f)



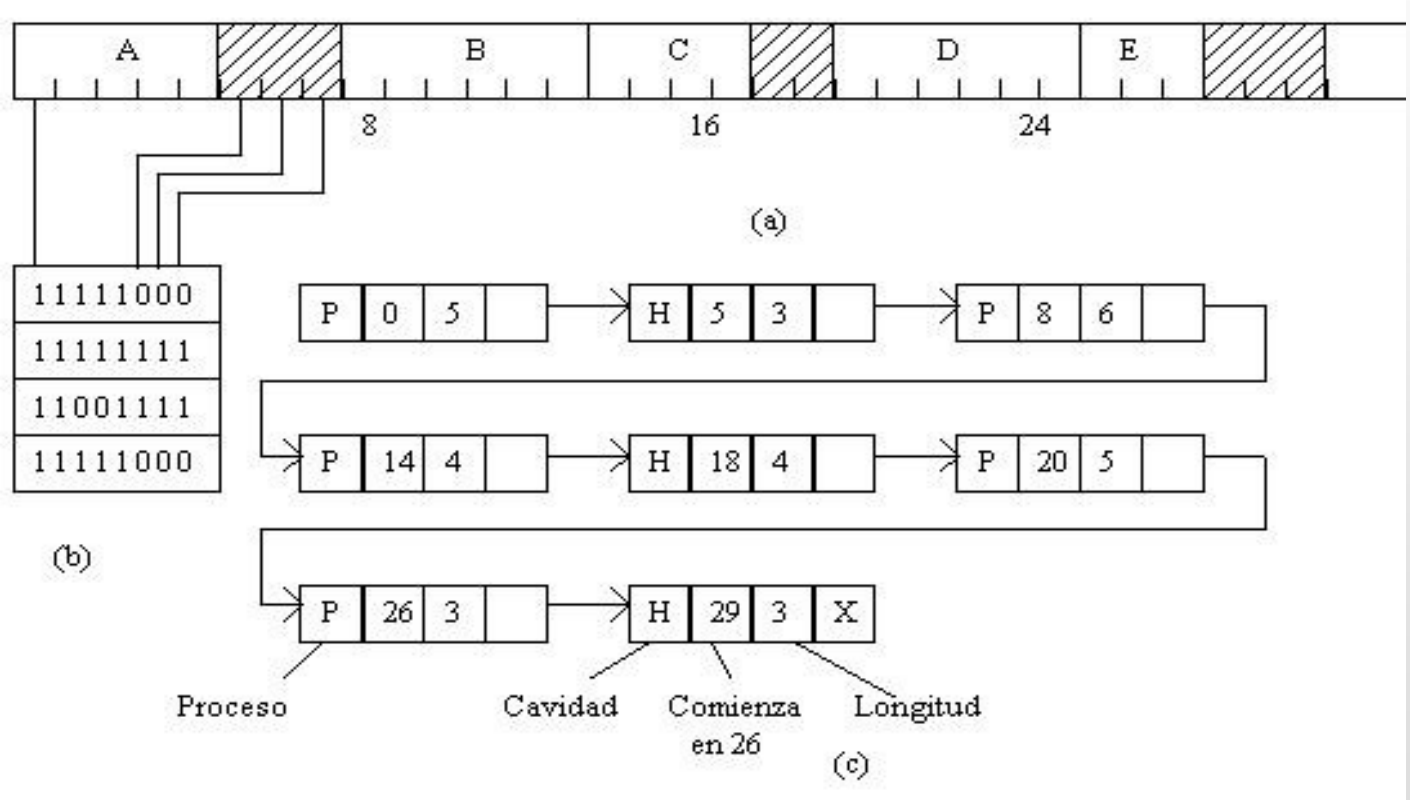
(g)



(h)

Particiones dinámicas - Manejo de los “huecos”

Dos formas de llevar el registro del uso de la memoria: mapas de bits y listas.



Fragmentación

- Espacio libre de la memoria que no puede ser utilizado

- Interna ➔ Particiones Fijas
  - Espacio dentro de la partición sin utilizar

- Externa ➔ Particiones Dinámicas
  - Cada vez que entra y sale un proceso se genera huecos en la memoria, en los que eventualmente un proceso no podría entrar, pero si entraría si unimos todos los huecos ➔ COMPACTACION

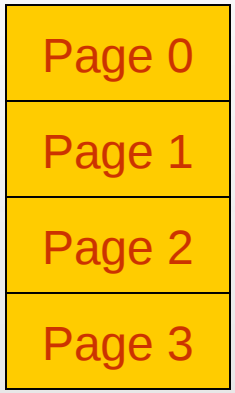
Particiones - Algoritmos de Ubicación

- No es necesario para Particiones Fijas de igual tamaño:
- Para particiones Fijas de diferente tamaño y Particiones Dinámicas si:
  - First Fit
  - Best Fit
  - Worst Fit
  - Next Fit

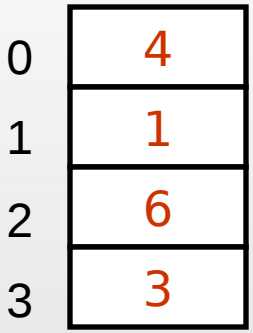
Paginación

- La memoria es dividida lógicamente en pequeños trozos de igual tamaño ➔ Marcos
- El espacio de direcciones de cada proceso es dividido en trozos de igual tamaño que los marcos ➔ Paginas
- El SO mantiene una tabla de paginas por cada proceso.
  - Contiene el marco en la que esta situada cada pagina.
  - La dirección lógica consiste en un numero de pagina y un desplazamiento dentro de la misma.

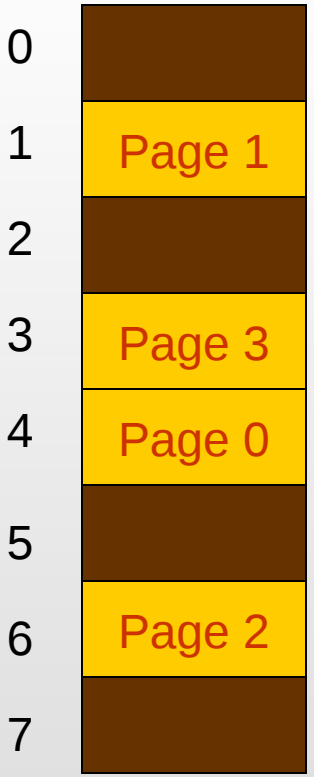
Paginación - Ejemplo



logical memory

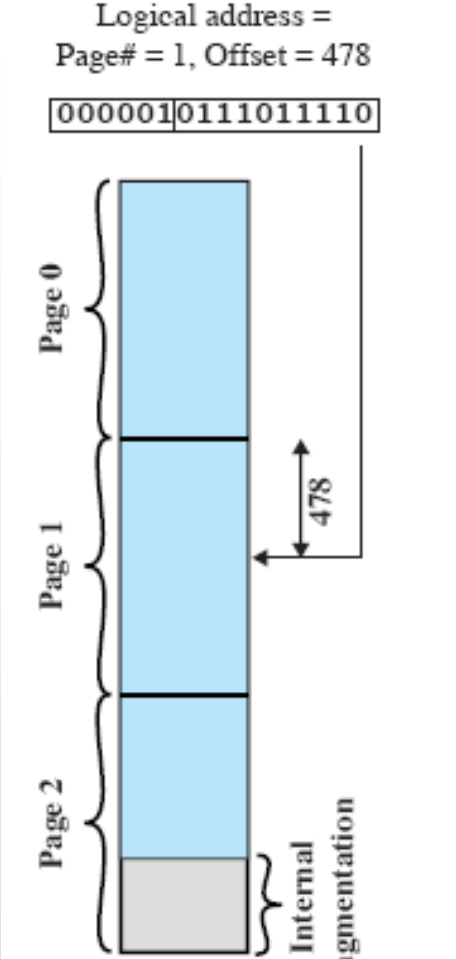


page table



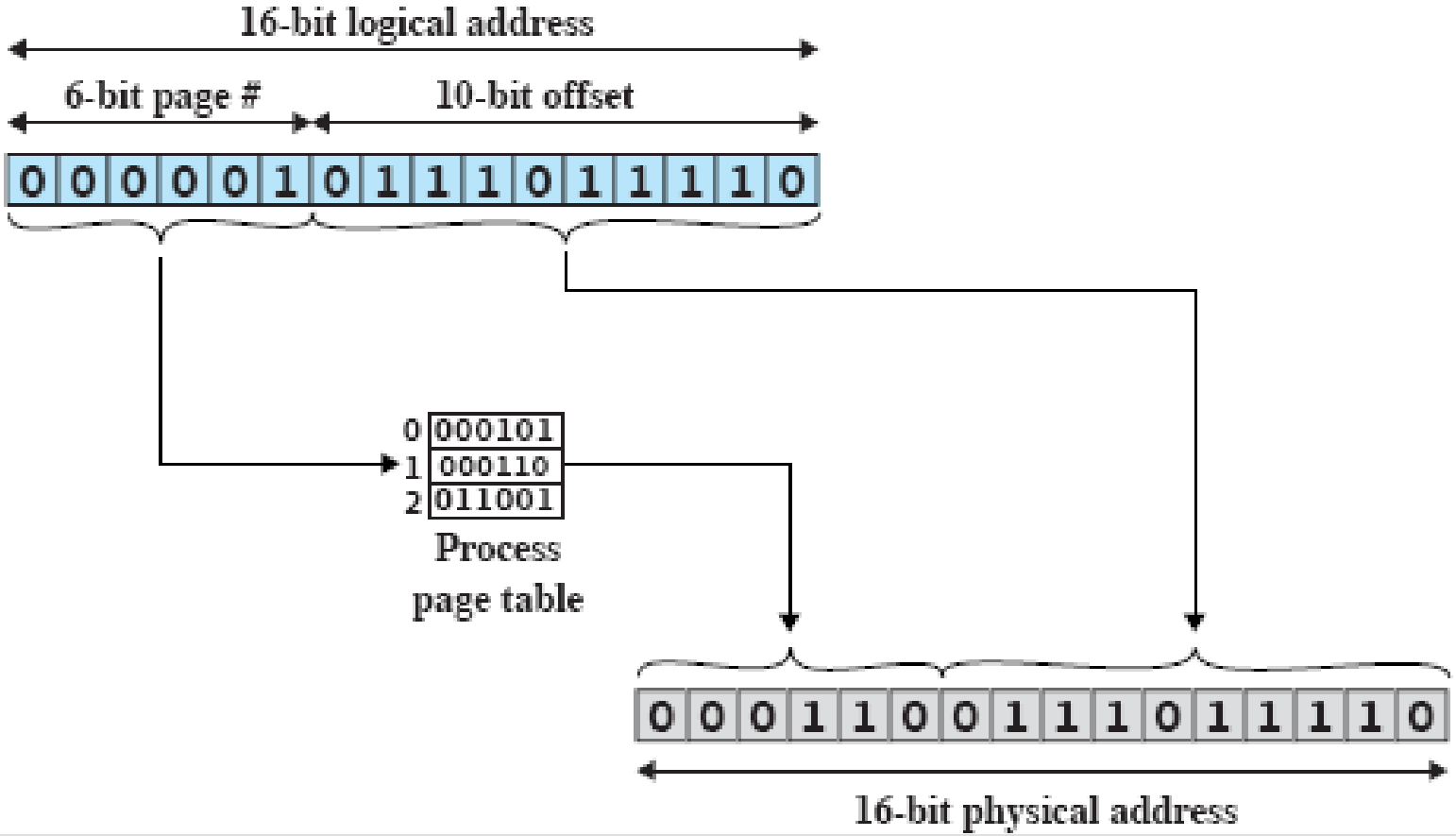
physical memory

Paginación – Direcciones Lógicas



(b) Paging
(page size = 1K)

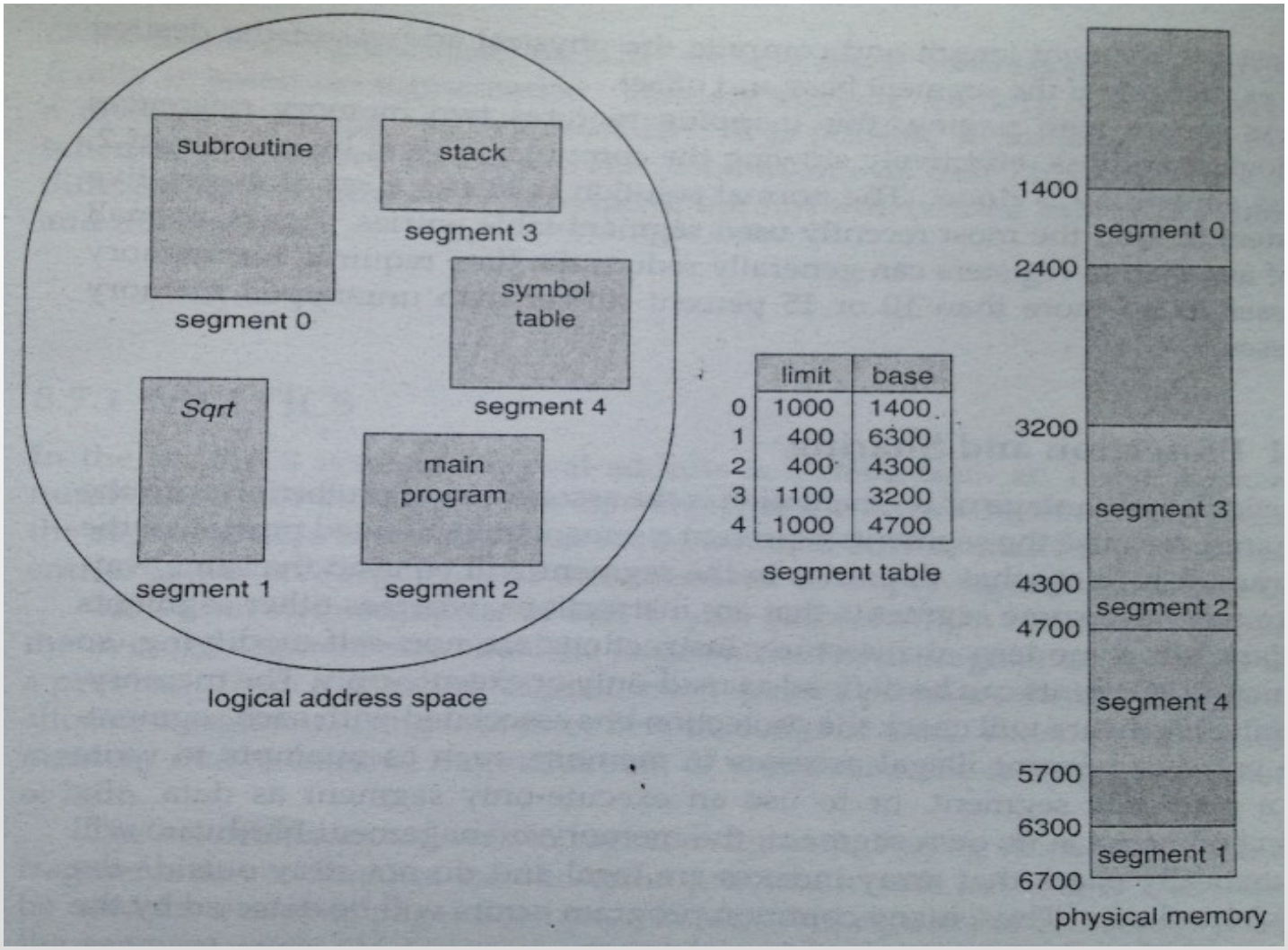
Traducción de direcciones



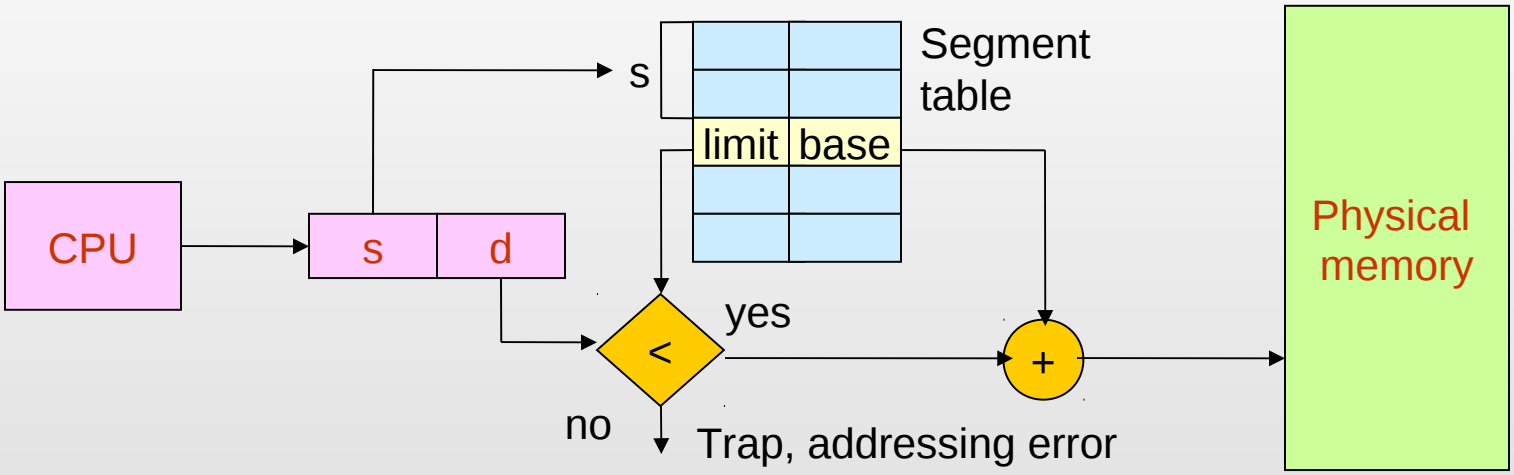
Segmentación

- Un programa es una colección de segmentos.
- Un segmento es una unidad lógica, con su espacion de direcciones propio
- Similar a particiones dinámicas, solo que el proceso tiene varias particiones, no solo una.
- En lugar de tener una tabla de páginas, se tiene una tabla de segmentos

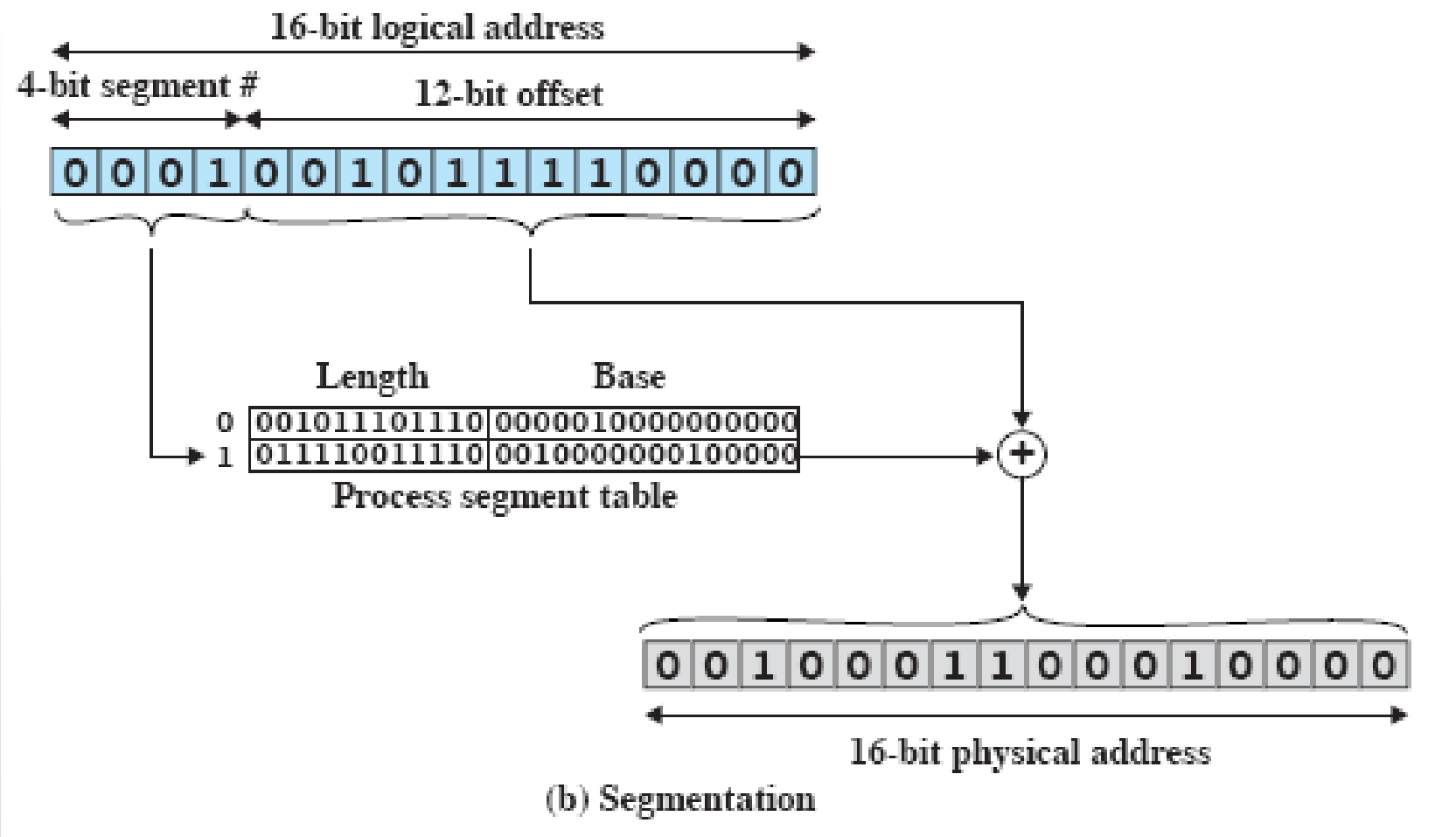
Proceso desde el punto de vista del usuario



Segmentación (cont.)



Segmentación - Direcciones (cont.)



(b) Segmentation

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
- Cada segmento es dividido en paginas de tamaño fijo.

Segmentación Paginada (cont.)