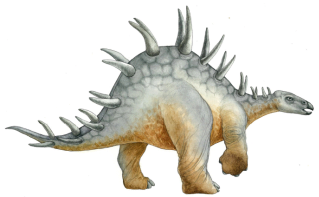
Capítulo 8: Memoria principal

Silberschatz, Galvin and Gagne ©2007

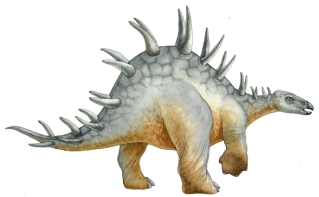
Capítulo 8: Manejo de memoria

- Antecedentes
- Swapping
- Asignación de memoria contigua
- Paginación
- Estructura de la tabla Página
- Segmentación
- Ejemplo: Intel Pentium



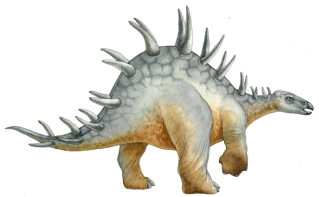
Objetivos

- Proveer descripciones detalladas de distintas formas de organizar hardware de memoria
- Discutir las distintas técnicas de manejo de memoria, incluyendo paginación y segmentación
- Proveer una descripción detallada del Intel Pentium, que soporta segmentación pura y con paginación



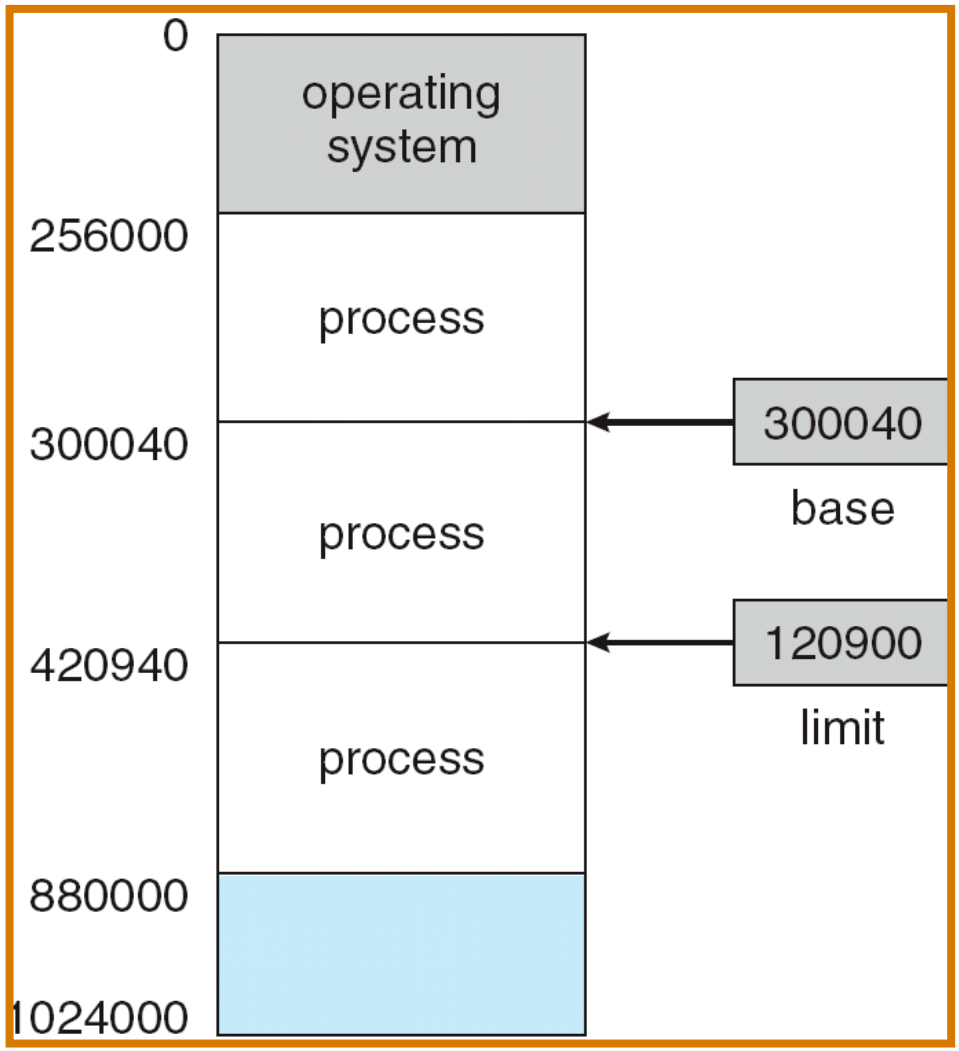
Antecedentes

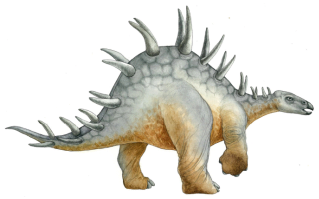
- Un programa debe traerse (del disco) a la memoria y colocarse dentro de un proceso para ejecución
- El CPU puede acceder exclusivamente a la memoria principal y los registros
- Acceso a los registros en un ciclo de reljo (o menos)
- Memoria principal varios ciclos
- **Cache** está entre la memoria principal y los registros
- Se requiere de protección de memoria para asegurar la correcta operación



Registros base y límite

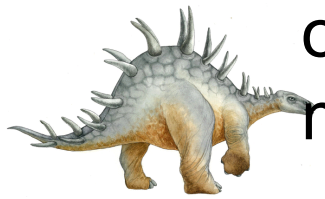
- Un par de registros **base** y **límite** definen el espacio lógico de direcciones



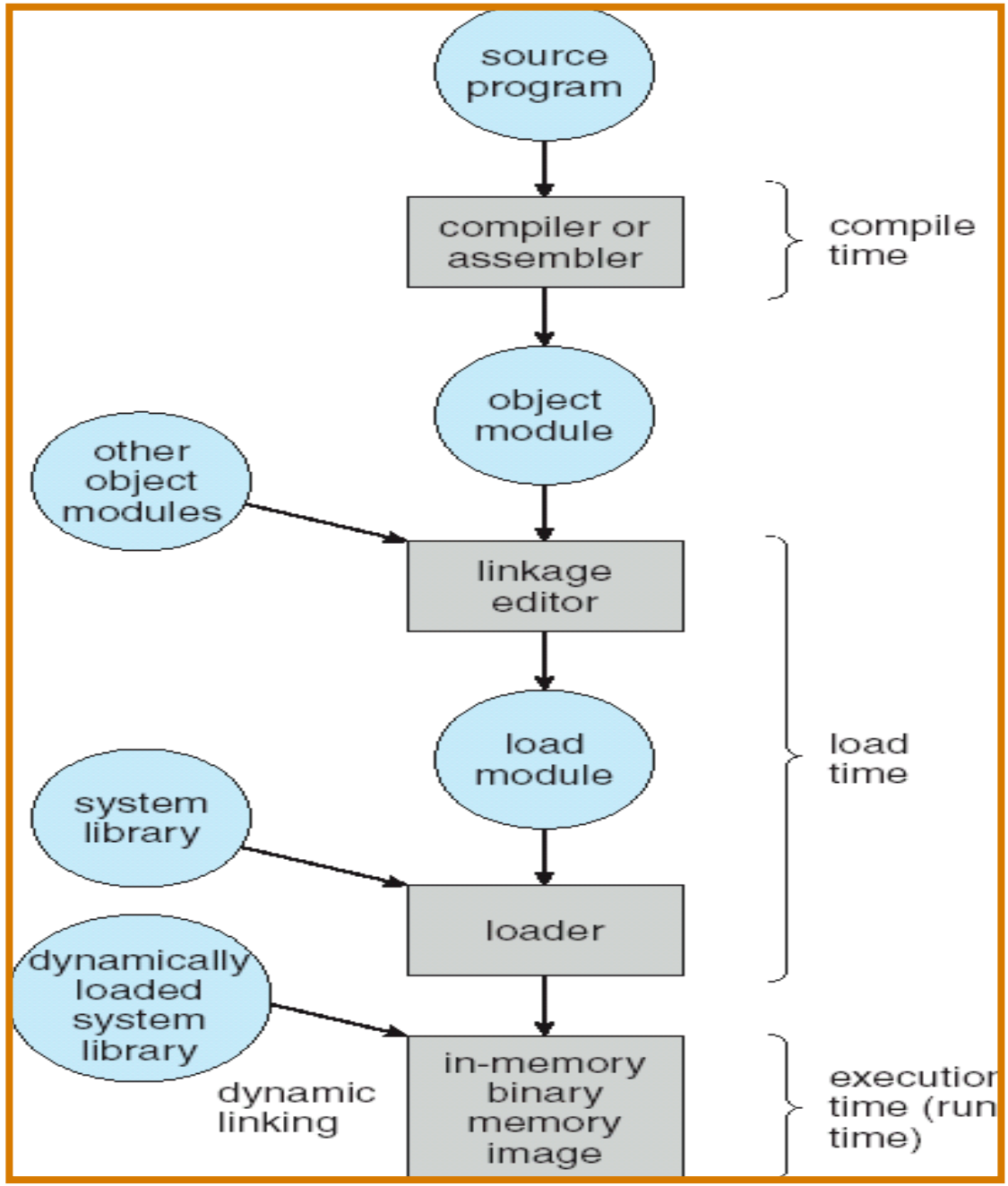


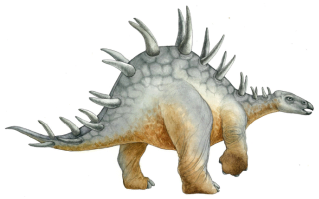
Ligado de instrucciones y datos a memoria

- El ligado de instrucciones y datos a direcciones de memoria puede suceder en tres etapas
  - **Tiempo de compilación**: Si se conoce la localidad de memoria *a priori,* se puede generar **código absoluto**. Se debe recompilar si la localidad de inicio cambia
  - **Tiempo de cargado**: Debe generar **código relocalizable** si la posición de memoria no se conoce en tiempo de compilación
  - **Tiempo de ejecución**: El ligado se retrasa hasta tiempo de ejecución si el proceso puede ser movido durante su ejecución de un segmento de memoria a otro. Requiere soporte de hardware para mapas de memoria (v.gr. registros base y límite)



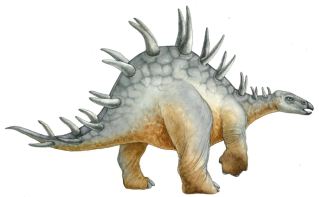
Procesamiento multi-paso de un programa de usuario





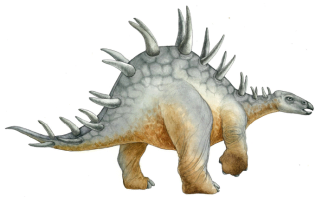
Espacio de direcciones físico vs. lógico

- El concepto de espacio de direcciones lógico asociado a distintos *espacios de direcciones físicas* es central para el manejo de memoria
  - **Dirección lógica** – generada por el CPU; también conocida como **dirección virtual**
  - **Dirección física** – dirección vista por la unidad de memoria
- Las direcciones lógicas y físicas son idénticas en tiempo de compilación y cargado; lógicas (virtuales) y físicas difieren tiempo de ejecución y en asociación de memoria

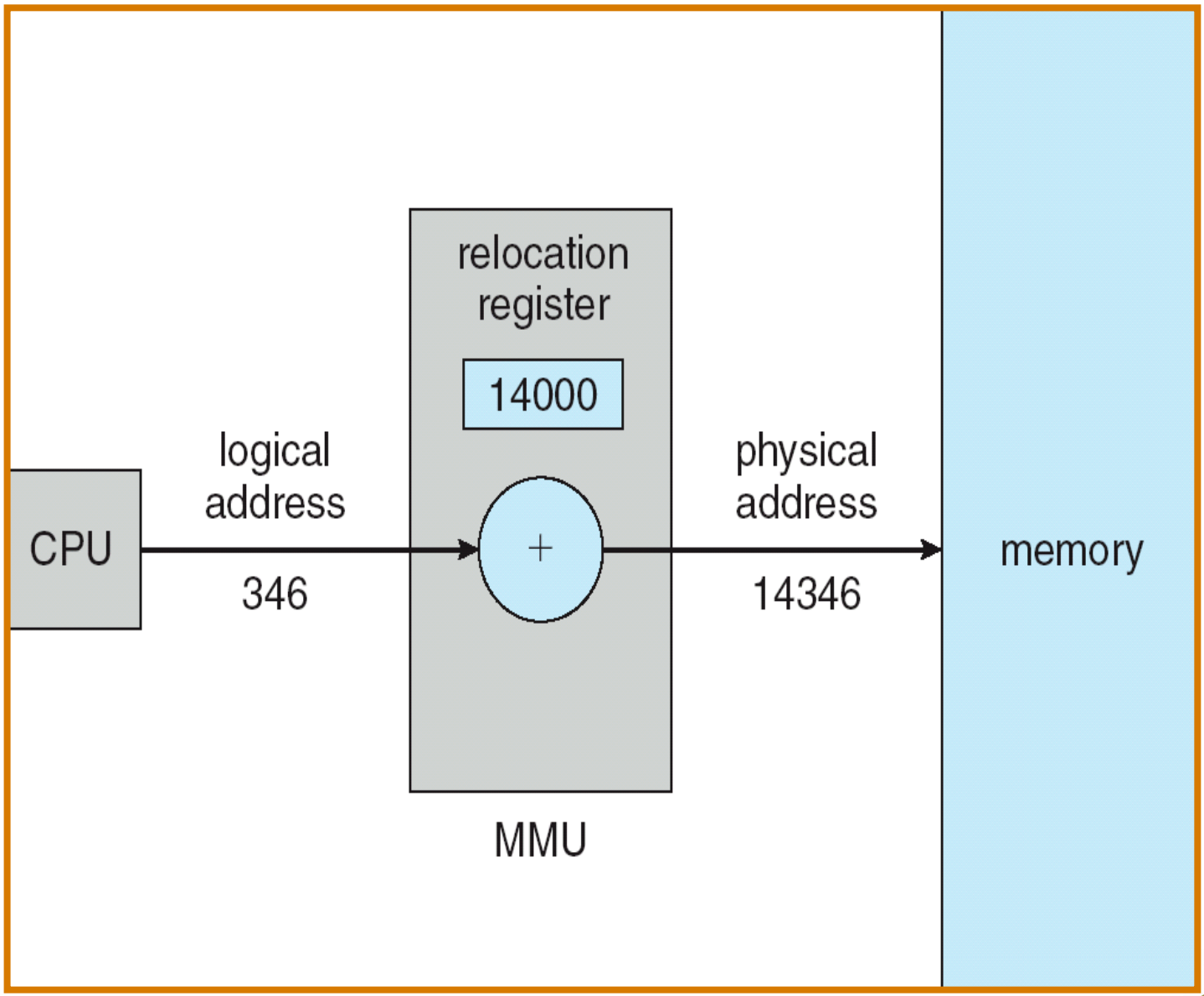


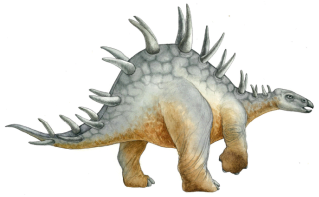
Unidad de Manejo de Memoria (MMU)

- Dispositivo de hardware que mapea direcciones virtuales a físicas
- En el esquema MMU, el valor en el registro de relocalización **se añade** a cada dirección generada por un proceso de usuario al momento de enviarla a la memoria
- El programa de usuario trabaja con direcciones *lógicas;* nunca ve las direcciones físicas *reales*



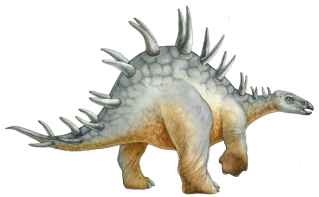
Relocalización dinámica utilizando un registro de relocalización





Cargado dinámico

- La rutina es cargada hasta que es llamada
- Mejor utilización de espacio de memoria; una rutina que no se utiliza, no se carga
- Útil cuando grandes cantidades de código se requieren para manejar casos poco frecuentes
- No se requiere soporte del sistema operativo, se implementa a través del diseño del programa



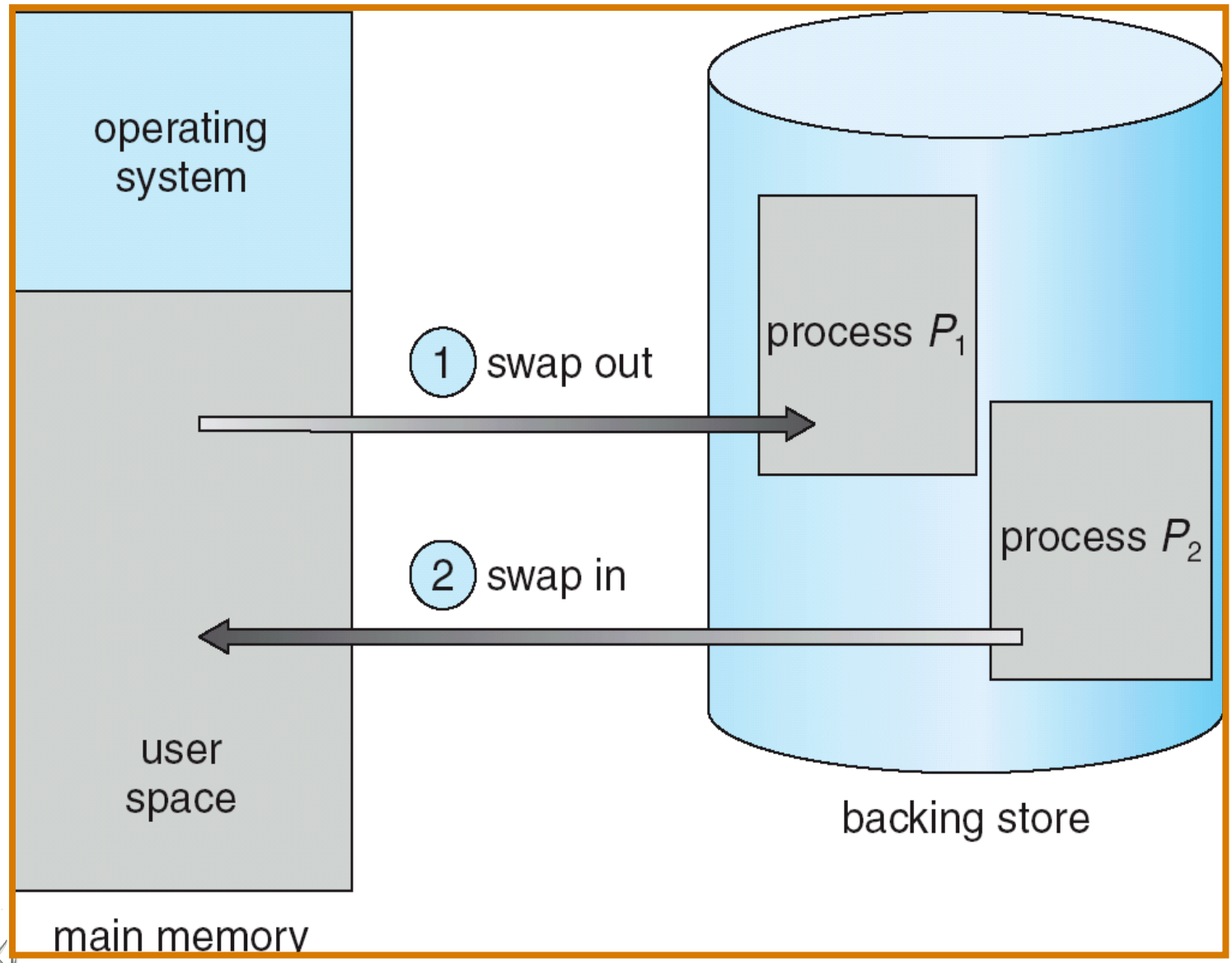
Ligado dinámico

- Se pospone el ligado hasta tiempo de ejecución
- Un pedacito de código, *stub*, se utiliza para localizar la rutina apropiada residente en memoria
- Stub se sustituye a sí misma con la dirección de la rutina y ejecuta la rutina
- El sistema operativo verifica si la rutina está en una dirección de memoria de procesos
- Ligado dinámico es útil para bibliotecas
- Sistema de conoce como **bibliotecas compartidas**

Swapping

- Un proceso puede ser removido (swapped) temporalmente fuera de la memoria a un almacenamiento de respaldo y después de regreso a memoria para continuar su ejecución
- **Almacenamiento de respaldo** – disco rápido y lo suficientemente grande para acomodar copias de toda las imágenes de memoria de los usuarios; debe permitir acceso directo a estas imágenes
- **Roll out, roll in** – variante a swapping utilizada para algoritmos de calendarización basada en prioridades; procesos de baja prioridad salen para que los de prioridad alta puedan cargarse y ejecutarse
- Mayor parte del tiempo de swap es tiempo de transferencia; tiempo total de transferencia es directamente proporcional a la cantidad de memoria movida
- Versiones modificadas de swapping se encuentran en varios sistemas (i.e., UNIX, Linux, and Windows)
- El sistema mantiene una **cola de listos** de procesos listos para ejecución que tienen imágenes de memoria en disco

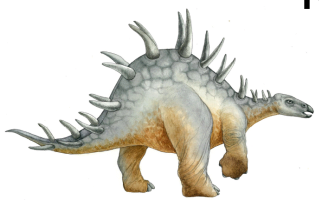
Vista esquemática de swapping



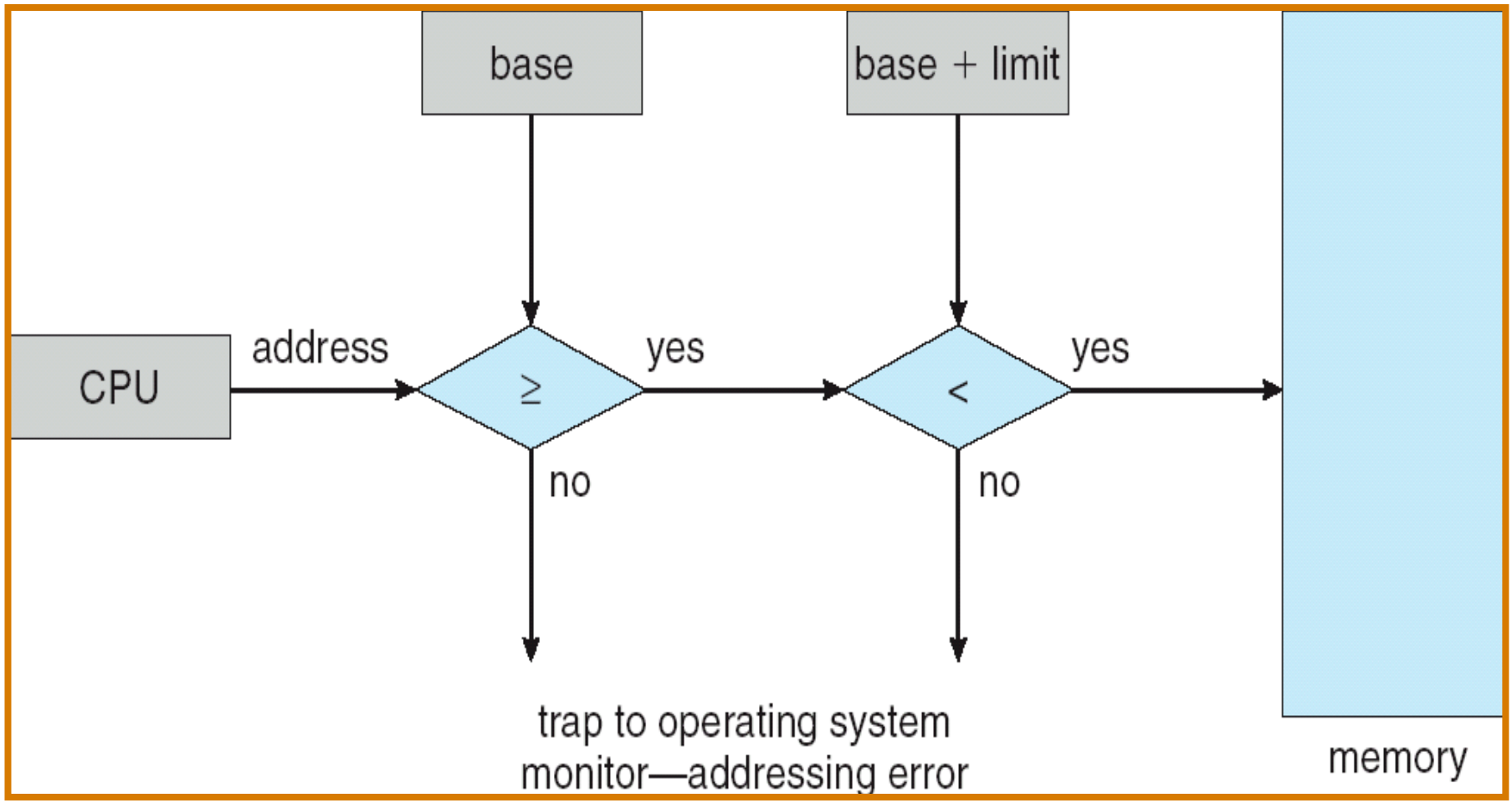
Asignación contigua

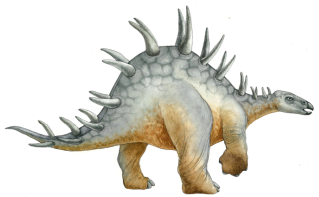
- Memoria principal usualmente en dos particiones:
  - Sistema operativo residente, usualmente se mantiene en memoria baja con vector de interrupción
  - Proesos de usuario se mantienen en memoria alta

- Registros de relocalización utilizados para proteger los procesos de usuario entre sí y también de datos y código cambiante del sistema operativo
  - El registro base contiene el valor de la dirección física más pequeña
  - El registro límite contiene el rango de direcciones lógicas – toda dirección lógica debe ser menor que el registro limite
  - MMU mapea direcciones lógicas *dinámicamente*



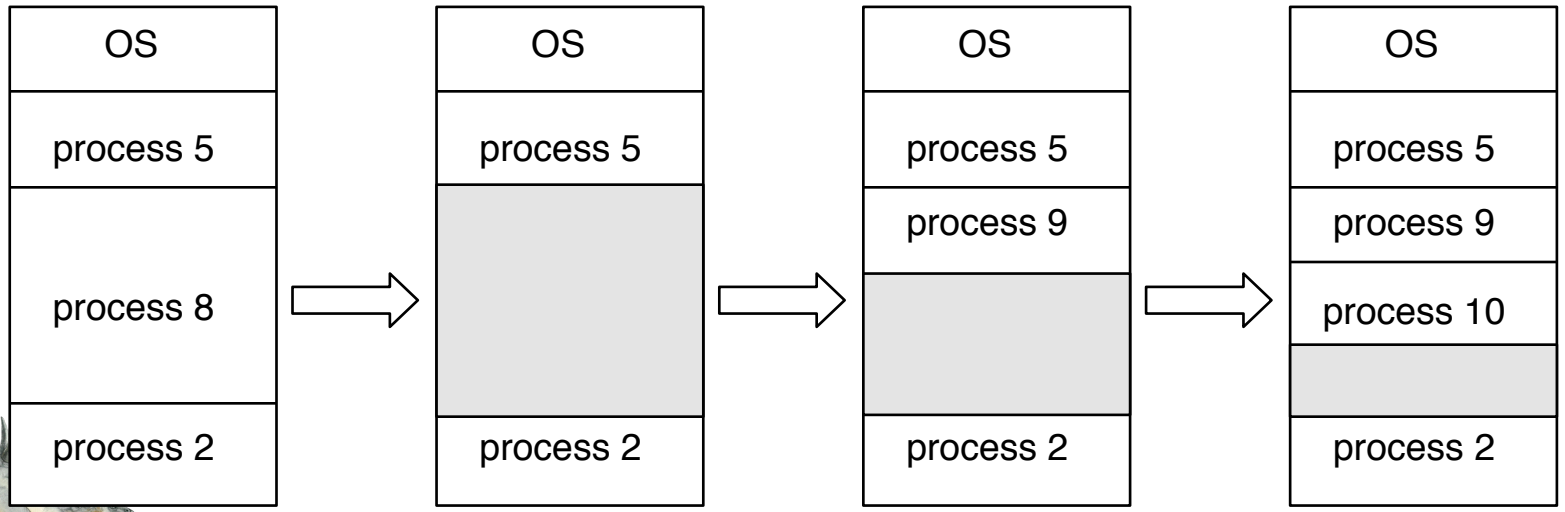
Protección de direcciones de HW con registros base y límite

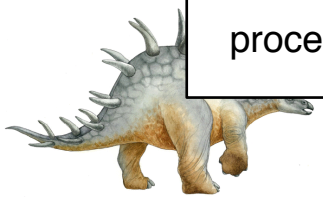




Asignación contigua (Cont.)

- Asignación de particiones múltiples
  - Hoyo – bloque de memoria disponible; hoyos de distintos tamaños distintos distribuidos en la memoria
  - Cuando un proceso llega, es asignado en un hoyo de memoria lo suficientemente grande para acomodarlo
  - Sistema operativo mantiene información acerca:
    a) particiones asignadas b) particiones libres (hoyo)



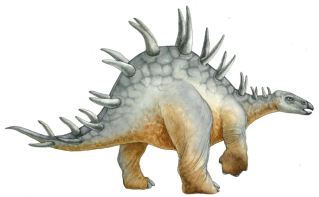


Problema asignación almacenamiento

¿Cómo satisfacer una solicitud de tamaño *n* de una lista de tres hoyos?

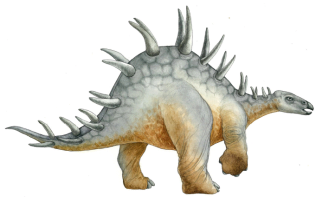
- **First-fit**: Asigna el *primer* hoyo lo suficientemente grande
- **Best-fit**: Asigna el hoyo *más pequeño* que sea lo suficientemente grande; debe buscar toda la lista, a menos que esté ordenada
  - Produce el sobrante (hoyo) más pequeño
- **Worst-fit**: Asigna el hoy *más grande*; también busca en toda la lista
  - Produce el sobrante (hoyo) más grande

First-fit y best-fit mejores que worst-fit en términos de velocidad y utilización de almacenamiento



Fragmentación

- **Fragmentación externa** – existe espacio total de memoria para satisfacer la solicitud, pero no está contigua
- **Fragmentación interna** – memoria asignada puede ser mayor que la solicitada; esta diferencia de tamaño es memoria interna a una partición, pero no utilizada
- Reducir fragmentación externa a través de **compactar**
  - Mover contenidos de memoria para juntar todos la memoria libre en un bloque grande
  - Compactar es posible ***sólo si*** la relocalización es dinámica y se realiza en tiempo de ejecución
  - Problema de E/S
    - Fijar trabajo en memoria mientras hace E/S
    - Hacer E/S sólo en buffers del SO

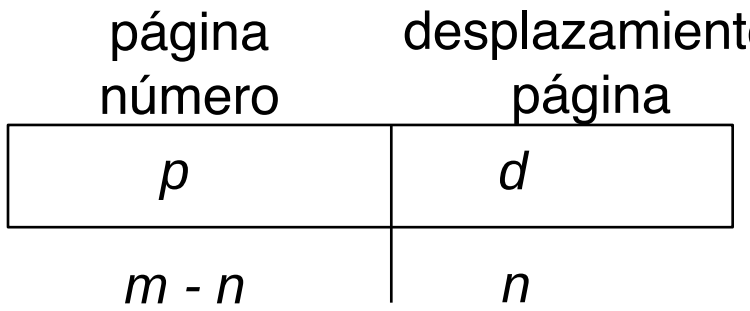


Paginación

- El espacio de direcciones lógicas de un proceso pueden no ser contiguas; se asigna memoria física al proceso, conforme está disponible
- Dividir memoria física en bloques de tamaño fijo llamados **frames** (tamaño es potencia de 2, entre 512 y 8,192 bytes)
- Dividir memoria lógica en bloques del mismo tamaño llamados **páginas**
- Mantener registro de todos los frames
- Para ejecutar un programa de tamaño ***n*** páginas, necesita encontrar *n* frames disponibles y cargarlo
- Crear una tabla de páginas para traducir direcciones lógicas a física
- Fragmentación interna

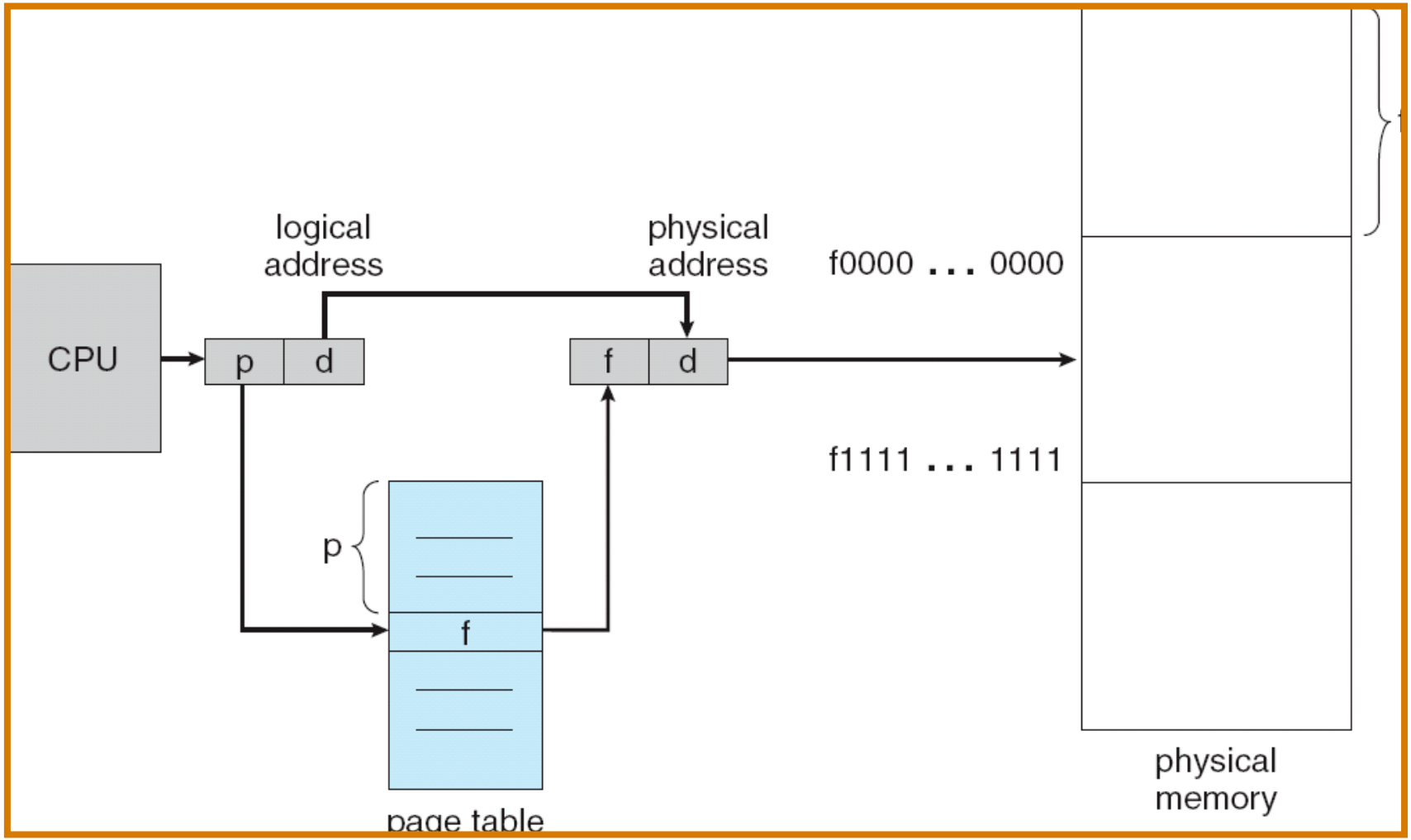
Esquema de traducción de direcciones

- Las direcciones generadas por CPU divididas en:
  - **Página número (*p*)** – utilizado como un índice en una *tabla de páginas* que contiene las direcciones base de cada página en memoria física
  - **Desplazamiento de página (d)** – combinado con direcciones base para definir la dirección de memoria física que es enviada a la unidad de memoria

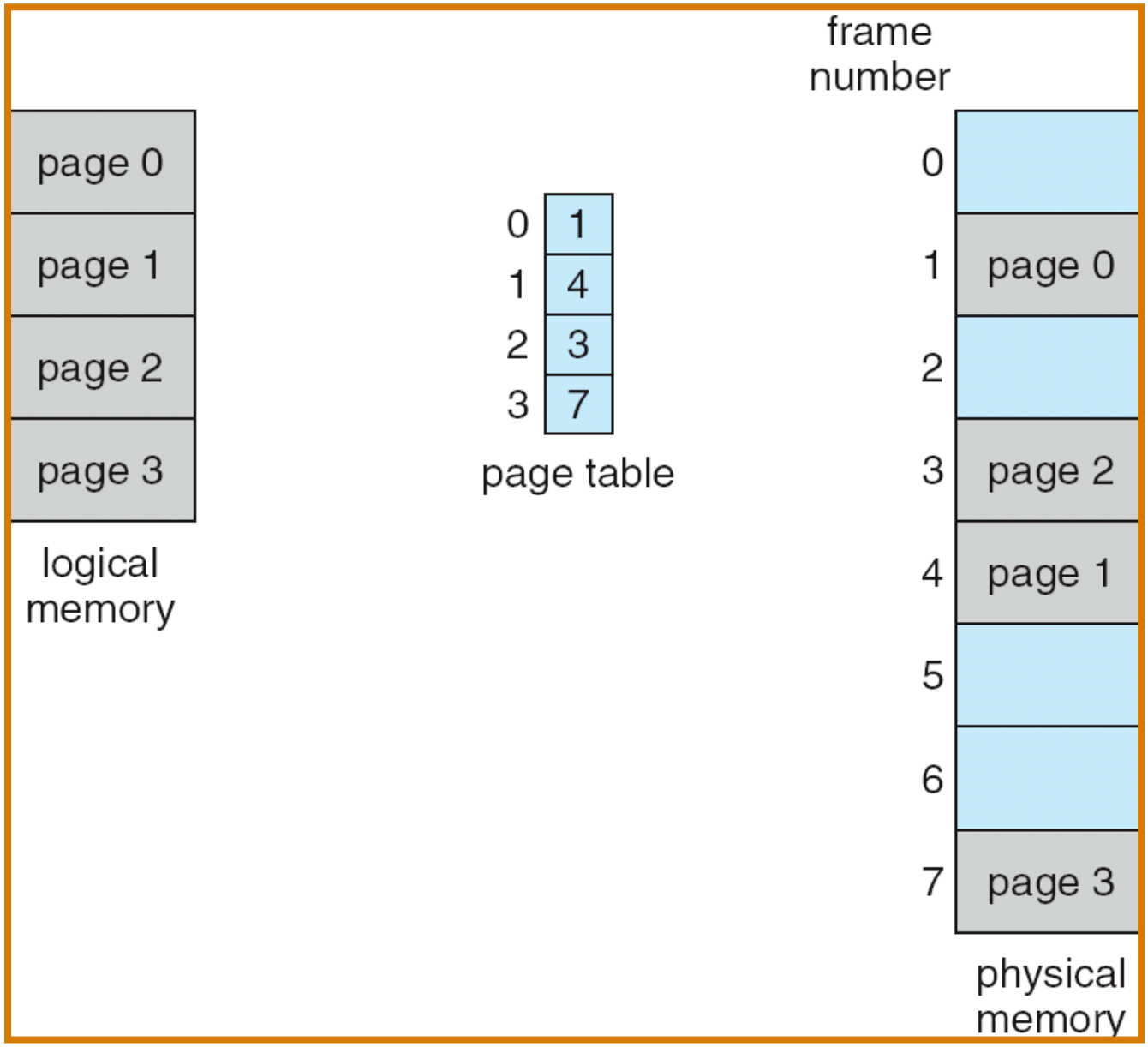


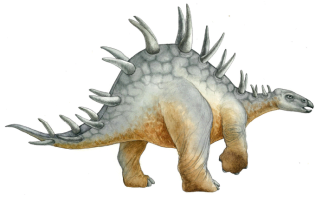
  - Para un espacio de direcciones lógicas 2^m *y tamaño de página* 2^n

Hardware de paginación

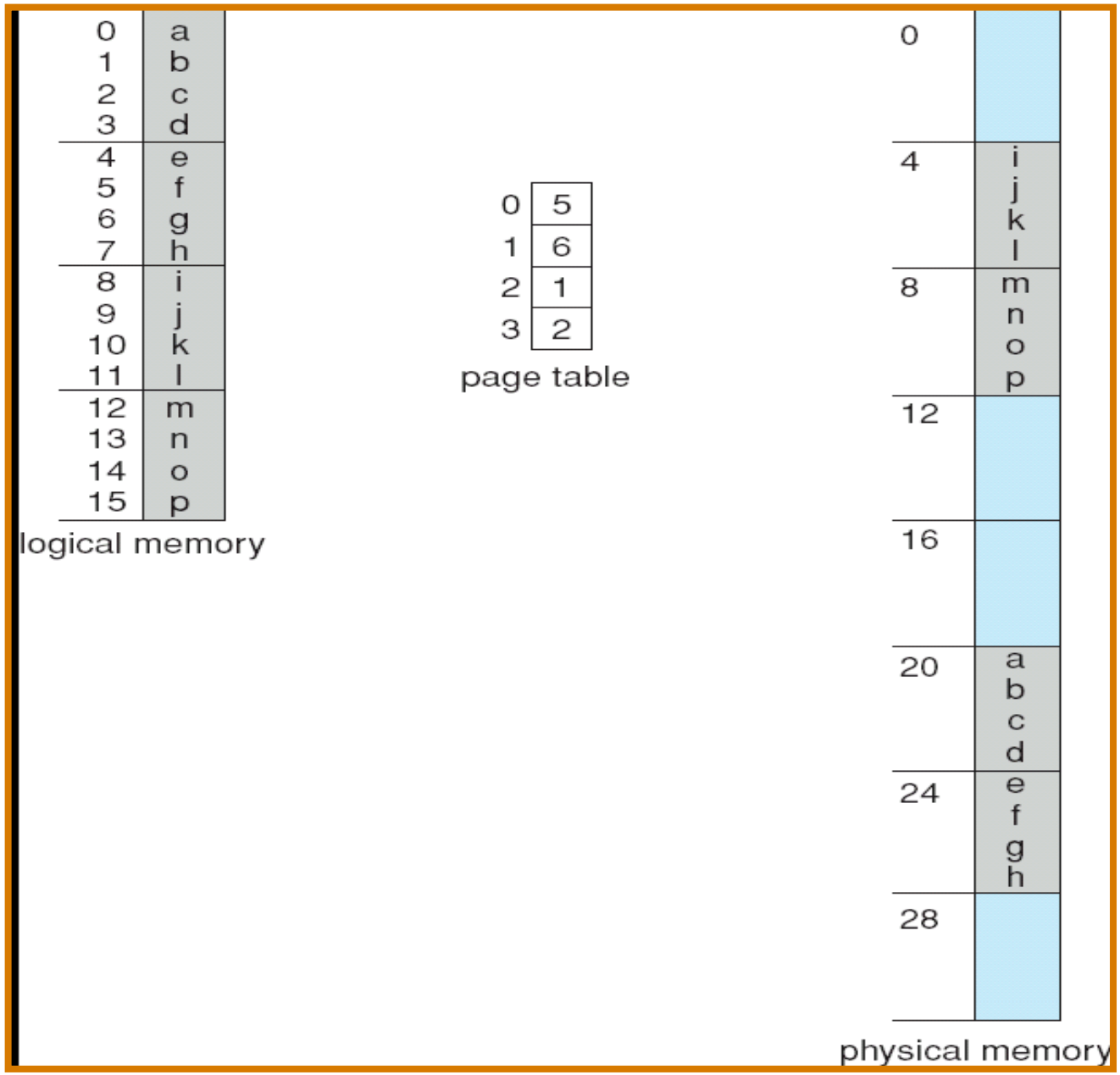


Modelo de paginación de memoria lógica y física

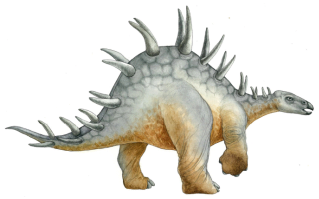




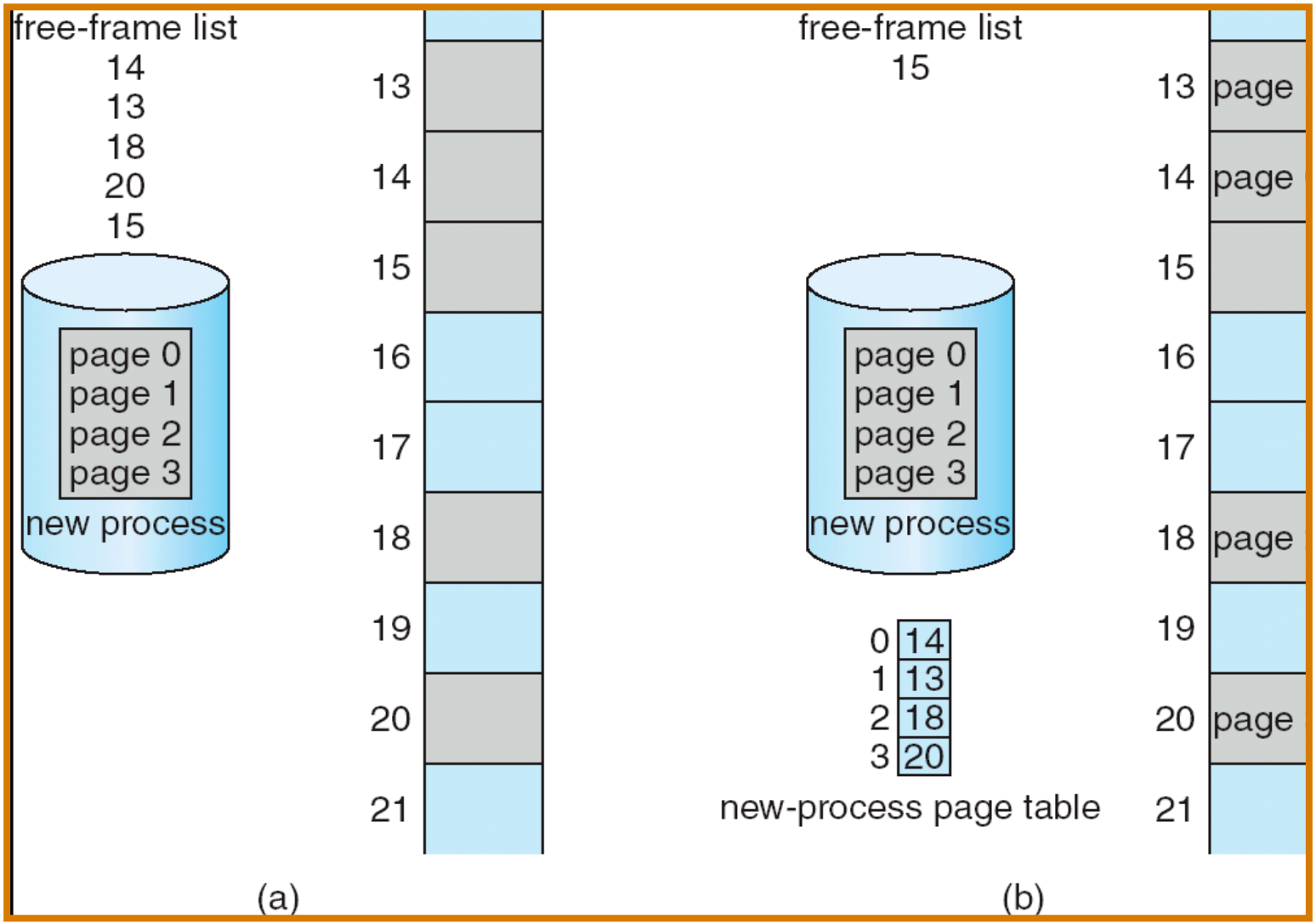
Ejemplo de paginación



memoria 32-byte y páginas 4-byte

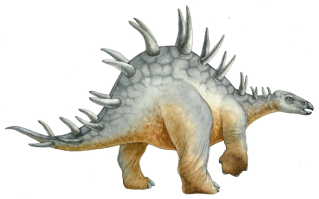


Frames disponibles



antes asignación

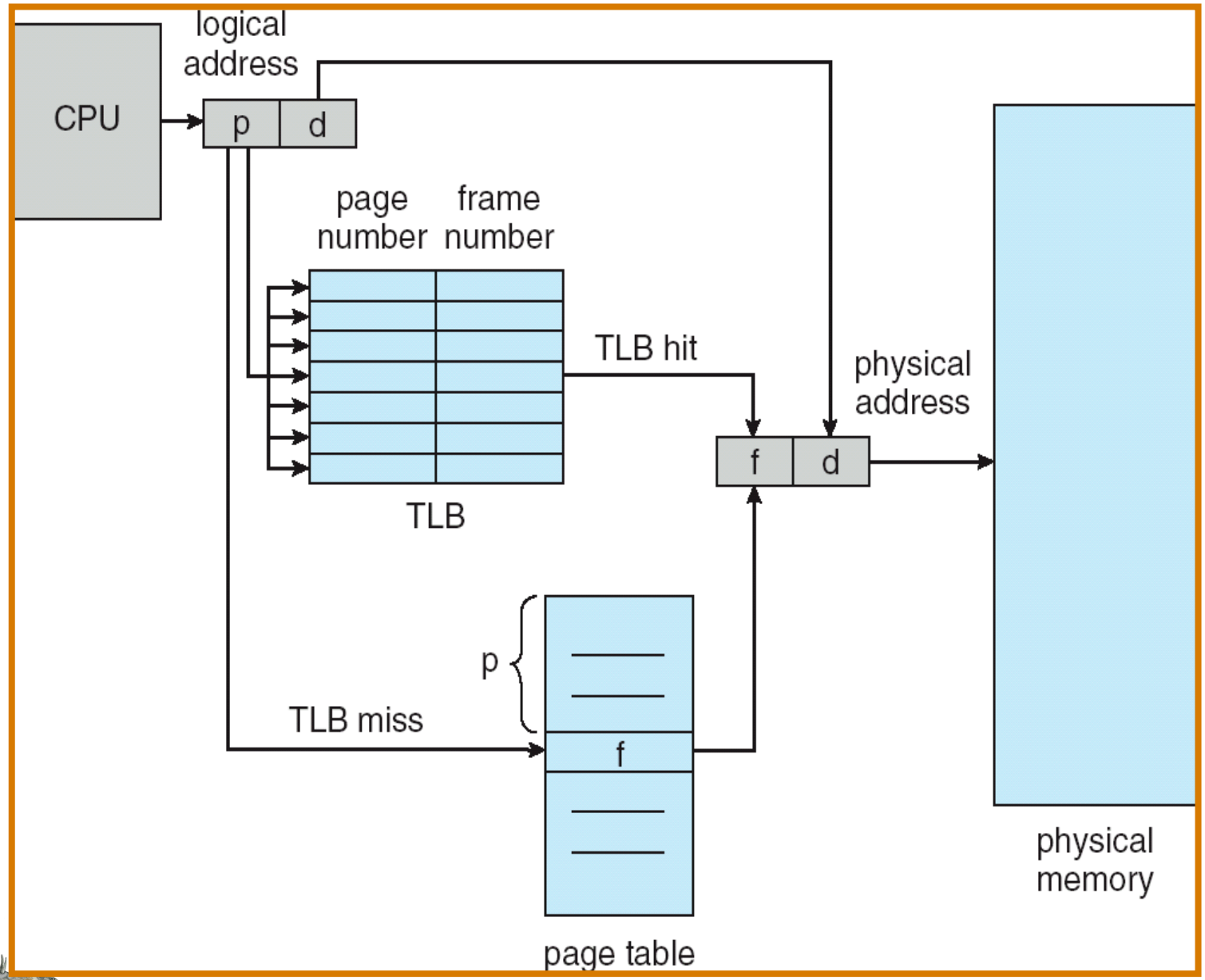
después asignación



Implementación de tabla de páginas

- Tabla de páginas se mantiene en memoria
- **Registro base de tabla de páginas (PTBR)** apunta a la tabla de páginas
- **Registro longitud de tabla de páginas (PRLR)** indica el tamaño de la tabla
- En este esquema cada acceso datos/instrucción requiere dos accesos a memoria. Uno para la tabla de páginas y otro para los datos/instrucción.
- El problema de accesos dobles a memoria puede resolverse con hardware especializado llamado **associative memory** o **translation look-aside buffers (TLBs)**
- Algunos TLBs almacenan **address-space identifiers (ASIDs)** en cada entrada TLB – identifican de manera unívoca cada proceso para proveer protección espacio-dirección para ese proceso

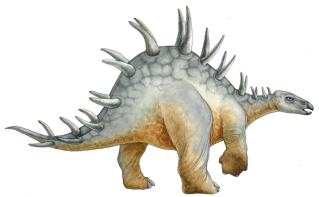
Hardware de paginación con TLB



Tiempo efectivo de acceso (EAT)

- Búsqueda asociativa = ε unidades de tiempo
- Asumimos ciclo de memoria es 1 microsegundo
- Radio de encuentro – porcentaje de veces que un número de página se encuentra en los registros asociativos; radio relativo al número de registros asociativos
- Radio de encuentro = α
- **Tiempo Efectivo de Acceso** (EAT)

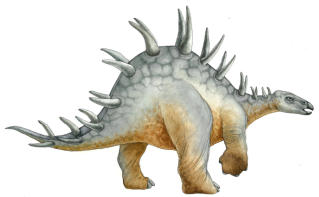
$$EAT = (1 + \varepsilon)\,\alpha + (2 + \varepsilon)(1 - \alpha)$$
$$= 2 + \varepsilon - \alpha$$



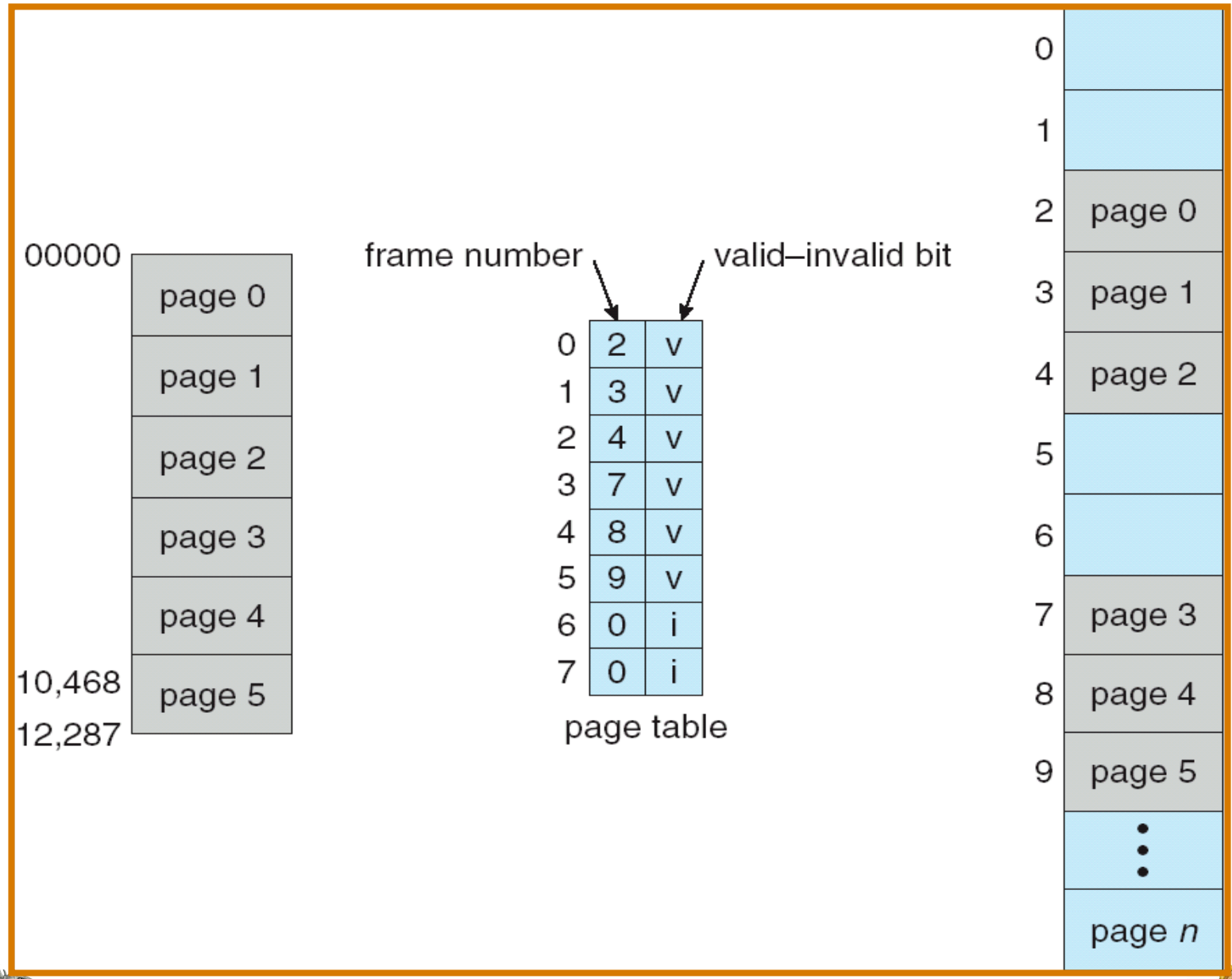
Protección de memoria

- Protección de memoria implementada asociando un *bit de protección* con cada frame

- Bit **valido-invalido** asociado a cada entrada en la tabla de páginas:
  - "valido" indica que la página asociada está en el espacio de direcciones lógico del proceso y es, por tanto, una página valida
  - "invalido" indica que la página no está en el espacio de direcciones lógico del proceso



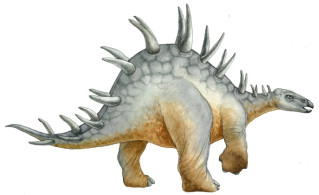
Bit Valido (v) o Invalido (i) en una tabla



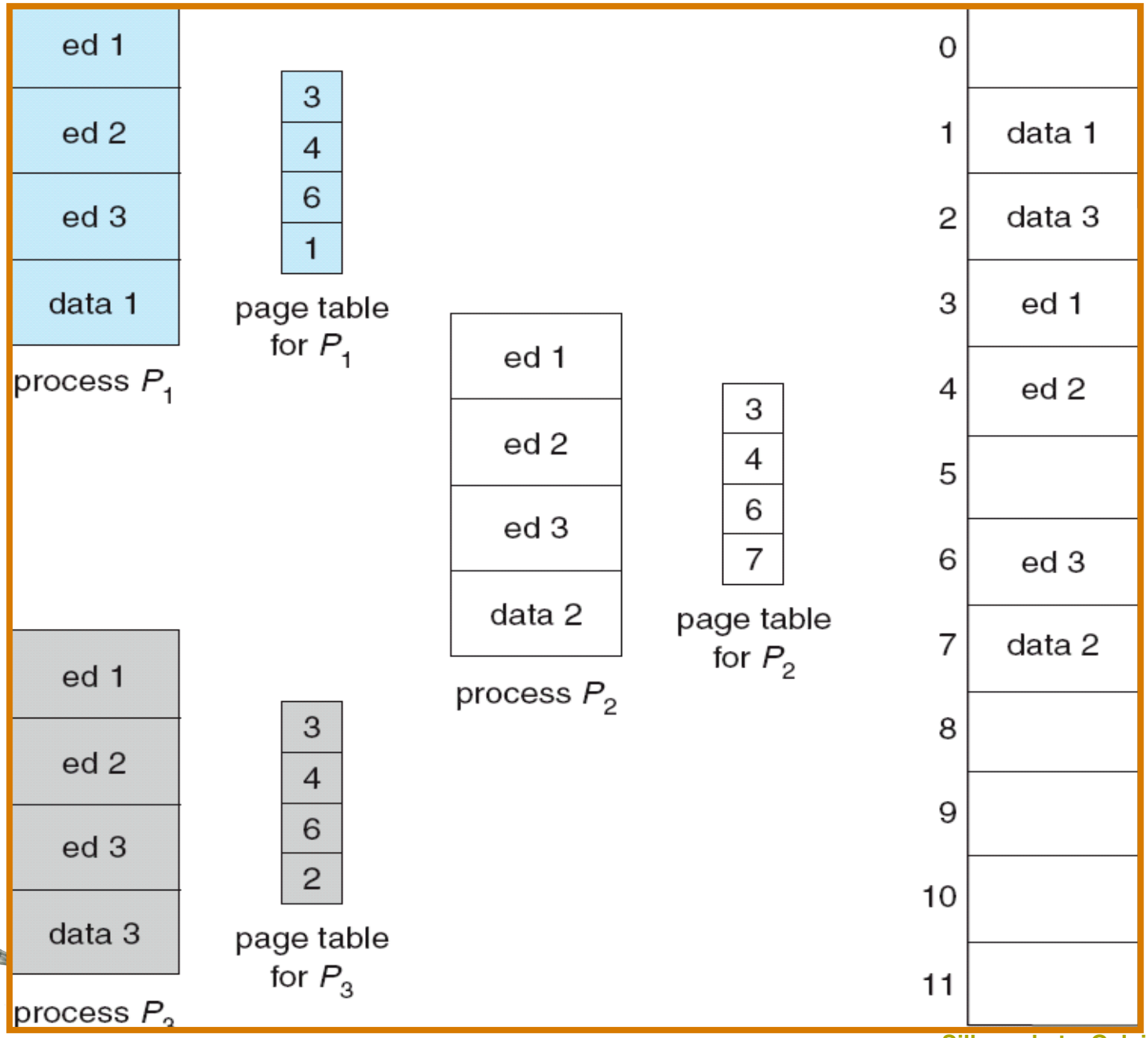
Páginas compartidas

- **Código compartido**
  - Una copia de código de sólo lectura compartido entre procesos (i.e., editores de texto, compiladores, ambientes de ventanas).
  - Código compartido debe aparecer en la misma posición en el espacio de direcciones lógico de todos los procesos

- **Datos y código privados**
  - Cada proceso mantiene una copia separada de datos y código
  - Las páginas para el código privado y datos puede aparecer en cualquier lugar en el espacio de direcciones lógico

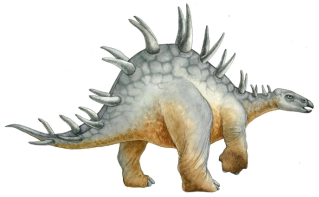


Ejemplo de páginas compartidas



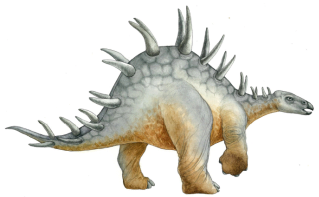
Estructura de la tabla de páginas

- Paginación jerárquica
- Tablas de páginas con dispersión (hash)
- Tablas de páginas invertidas

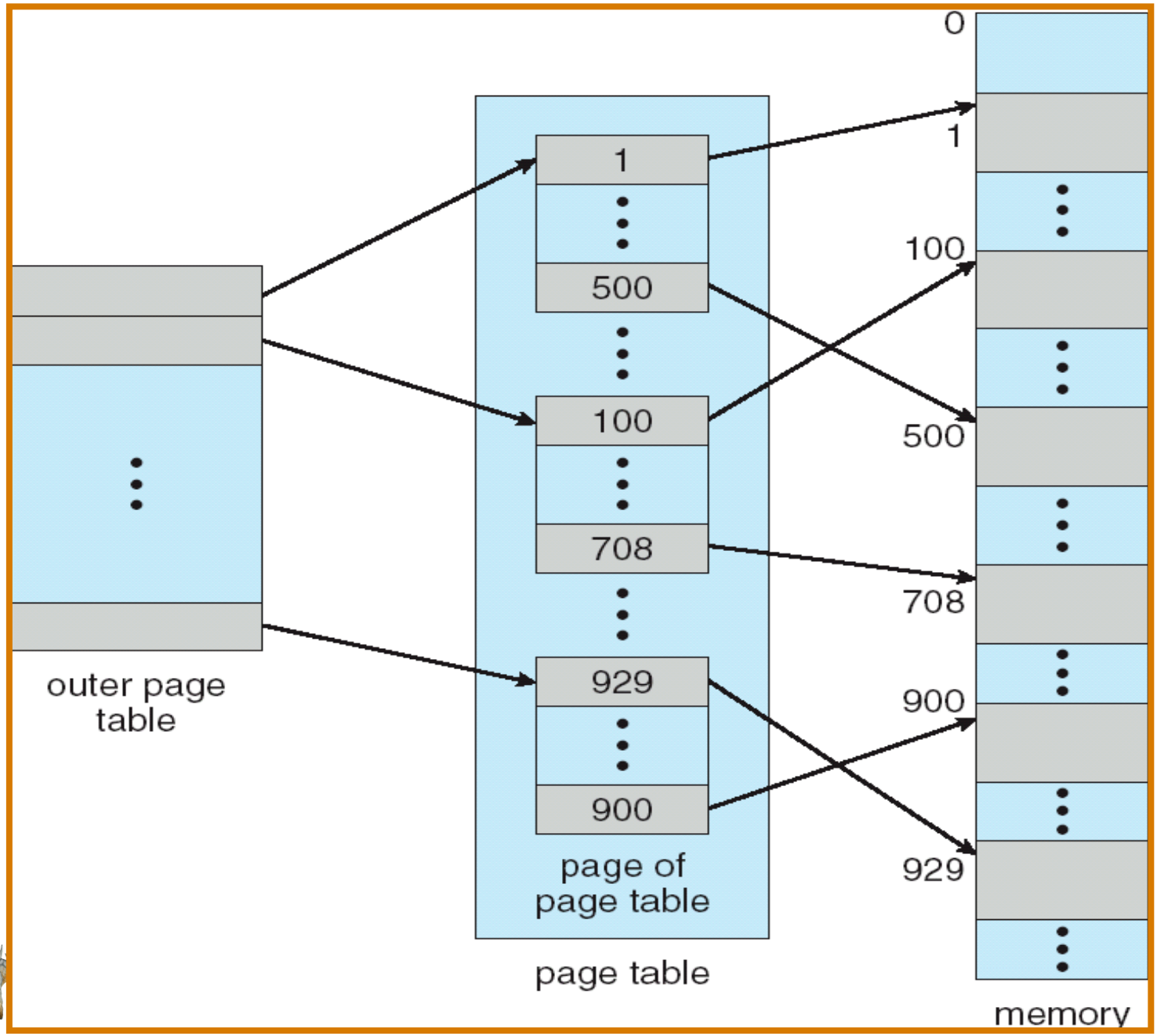


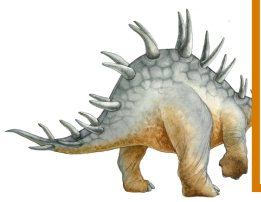
Tablas de páginas jerárquicas

- Dividimos el espacio de direcciones lógico en varias tablas de páginas
- Un técnica sencilla es una tabla de páginas de dos niveles



Esquema de tabla-páginas dos niveles



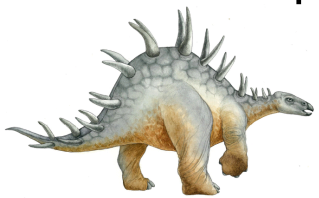


Ejemplo de paginación de dos niveles

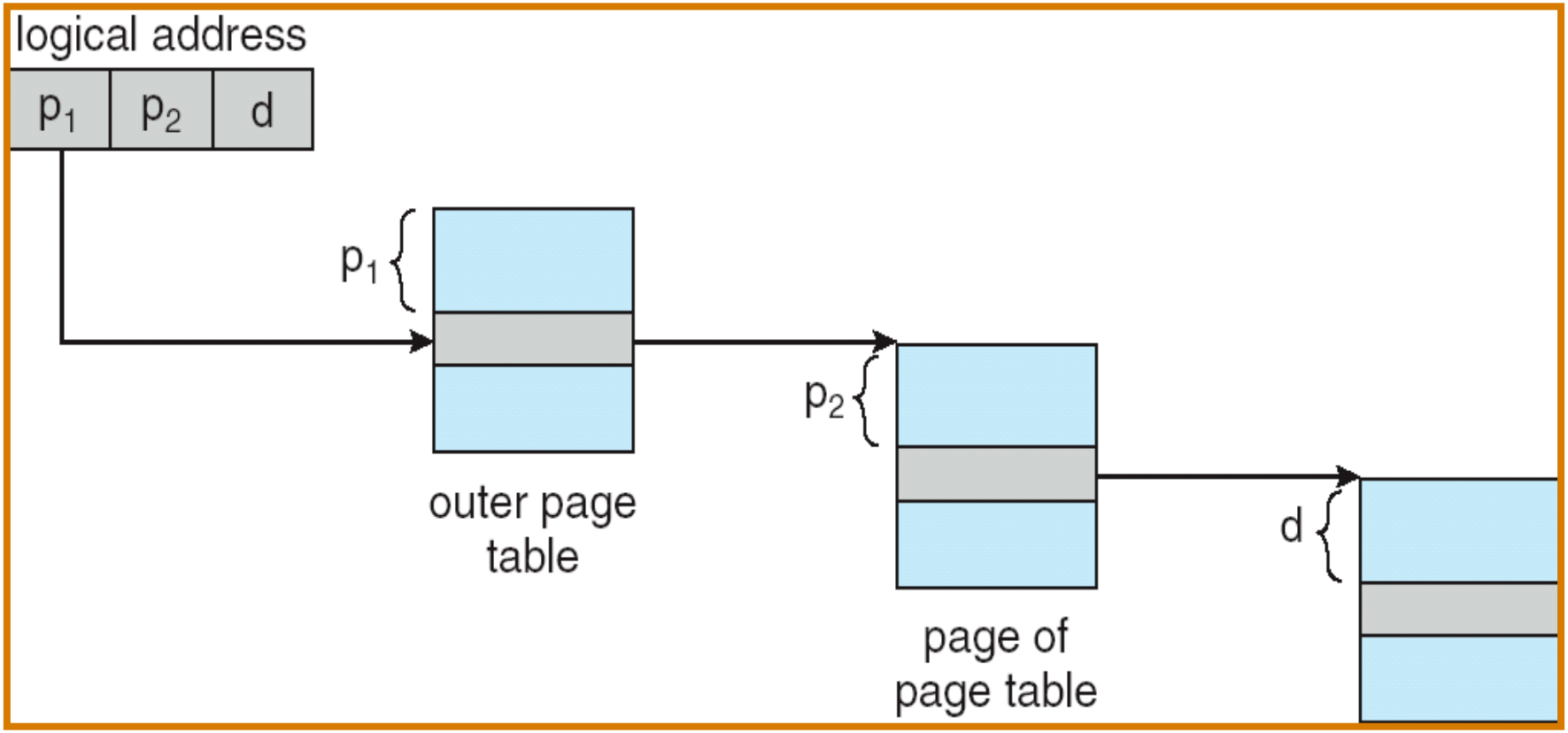
- Dirección lógica (máquina 32-bit / tamaño de página 1K) está dividido en:
  - un número de página consiste de 22 bits
  - un desplazamiento de página de 10 bits
- Dado que la tabla de páginas está paginada, el número de página se divide nuevamente en:
  - un número de página de 12-bit
  - un desplazamiento de 10-bit
- Entonces, una dirección lógica es así:

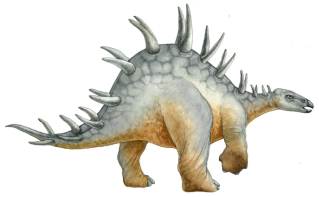
número página		desplazamiento
p_1	p_2	d
12	10	10

donde p_i es un índice en tabla de páginas exterior, y p_2 es el desplazamiento dentro de la página de la tabla interior



Esquema dirección-traducción

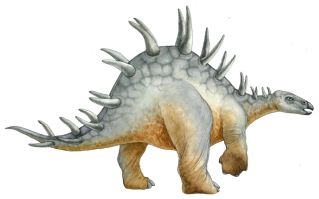




Esquema de paginación de tres niveles

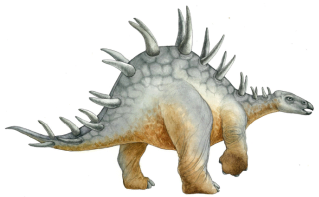
outer page	inner page	offset
p_1	p_2	d
42	10	12

2nd outer page	outer page	inner page	offset
p_1	p_2	p_3	d
32	10	10	12

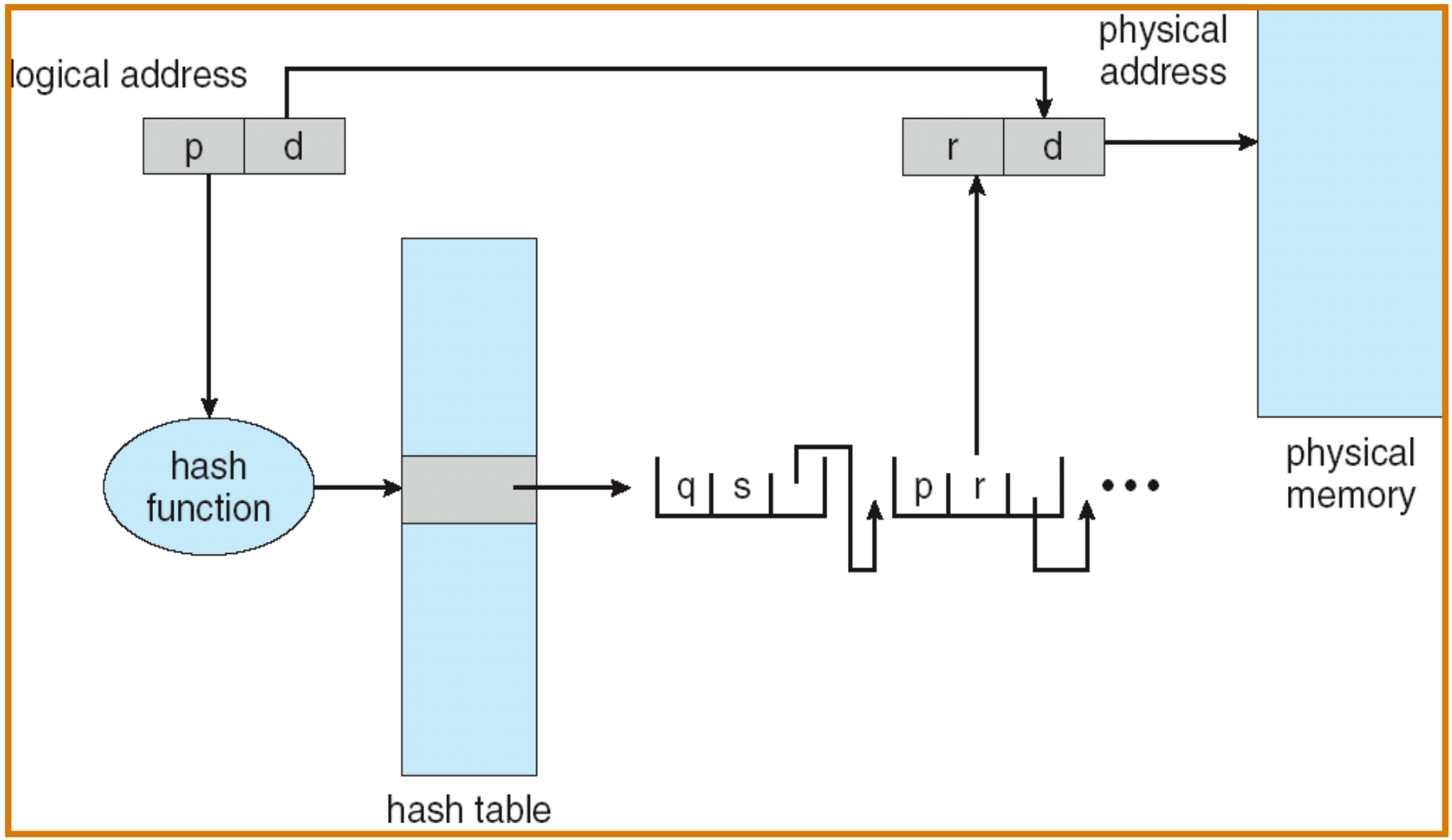


Tablas de páginas con dispersión (hash)

- Comunes en espacios de direcciones > 32 bits
- El número de página virtual se mete con un hash en una tabla de páginas. Esta tabla contiene una cadena de elementos con el mismo hash.
- Número de página virtual se compara en esta cadena hasta encontrarlo. Si se encuentra, se extrae el frame físico correspondiente.



Tablas de páginas con dispersión



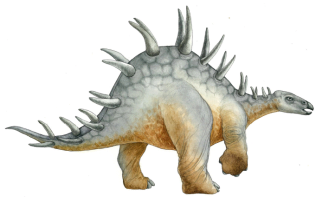
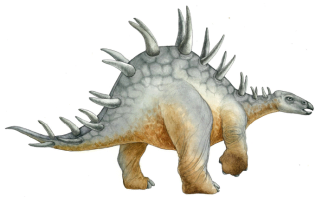
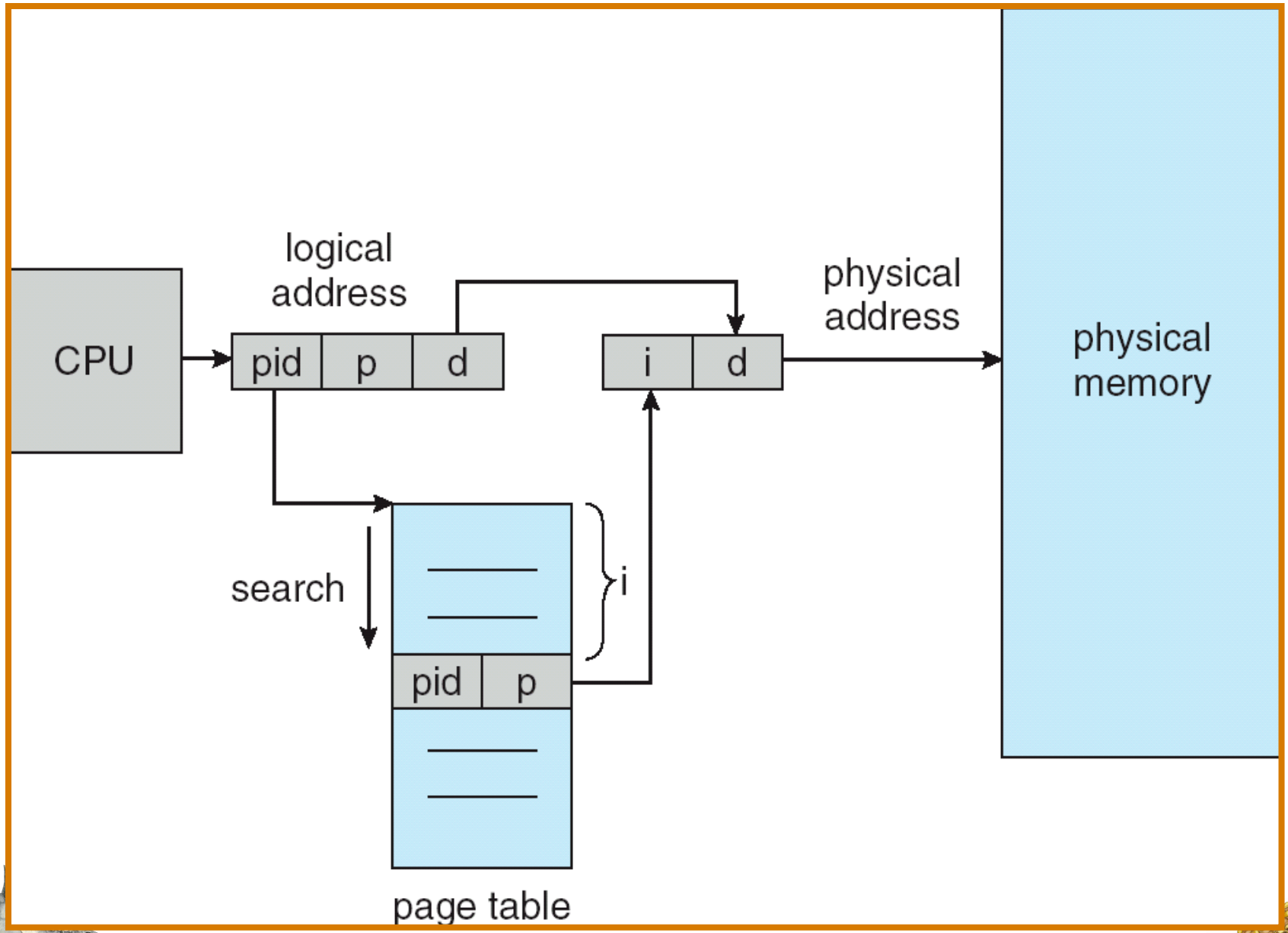


Tabla de páginas invertidas

- Una entrada por cada página real de memoria
- Cada entrada consiste de la dirección virtual de la página almacenada en la localidad de memoria real, con información acerca del proceso dueño de ella
- Reduce la cantidad de memoria requerida para almacenar cada tabla de páginas, pero aumenta el tiempo requerido para buscar en la tabla cuando hay una referencia a una página



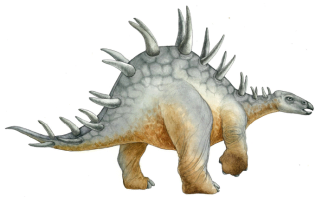
Arquitectura de tabla-páginas invertida



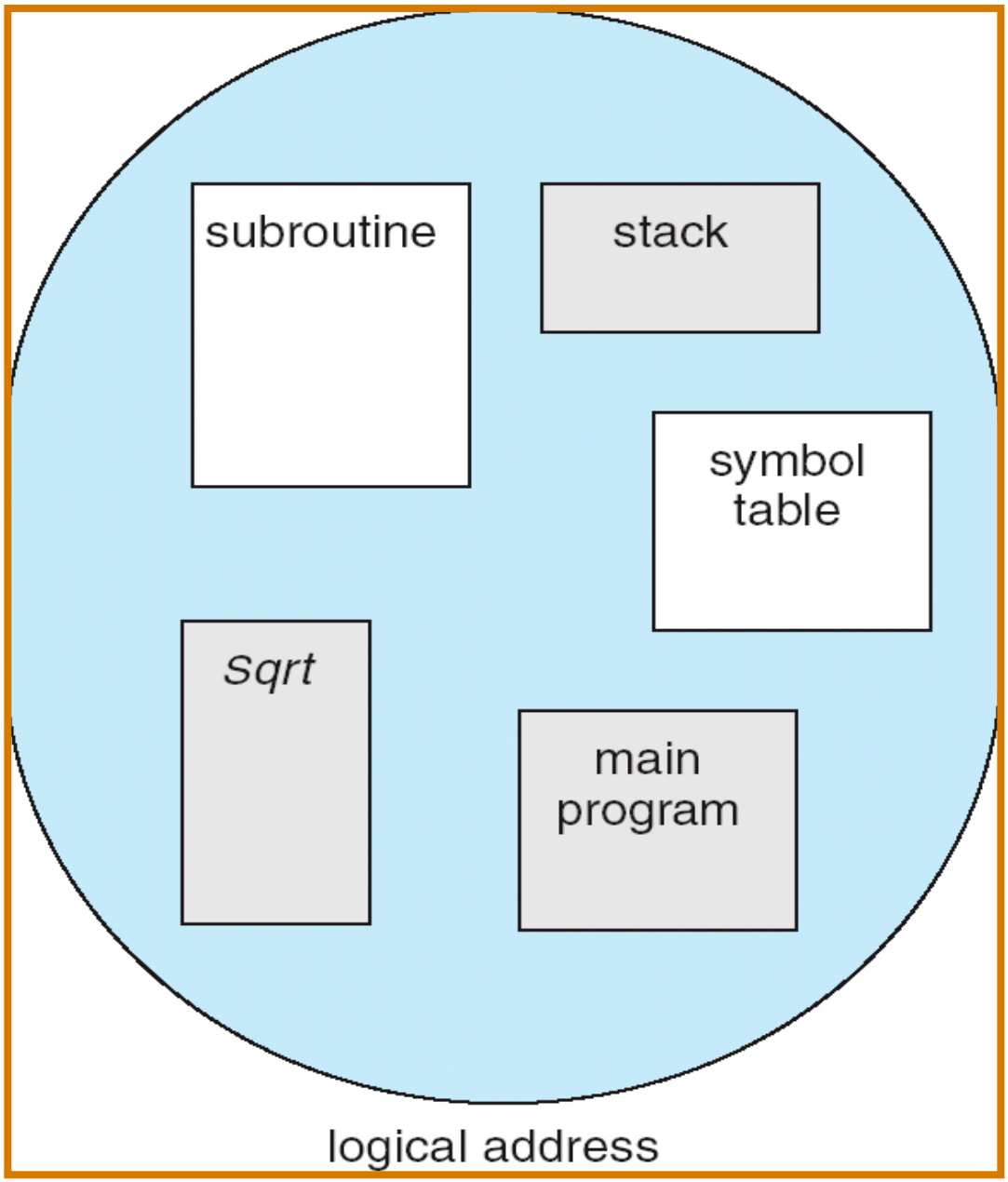
Segmentación

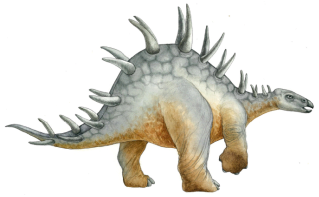
- Esquema para manejo de memoria que soporta vista de usuario de la memoria

- Un programa es una colección de segmentos. Un segmento es una unidad lógica tal como:
  programa principal,
  procedimiento,
  función,
  método,
  objeto,
  variables locales, globales,
  bloque común,
  stack,
  tabla de símbolos, arreglos

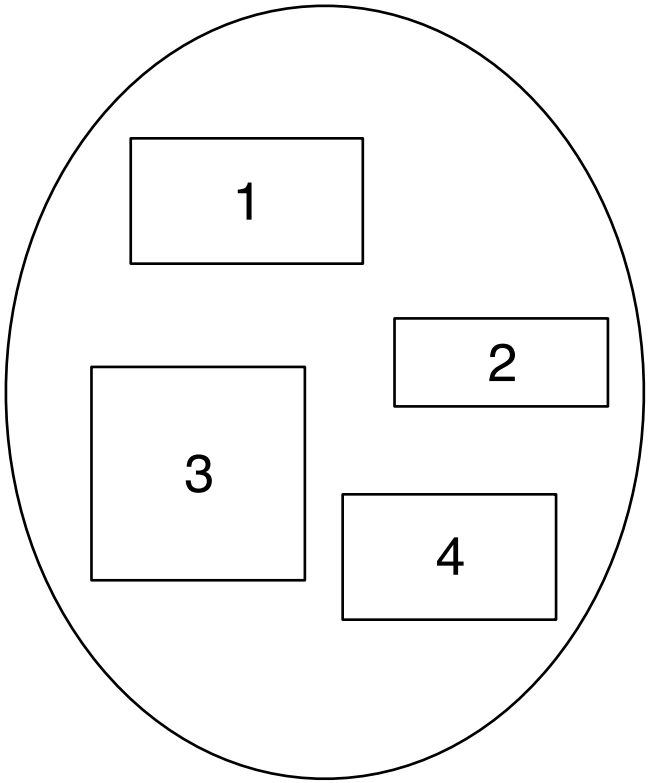


Vista del usuario de un programa

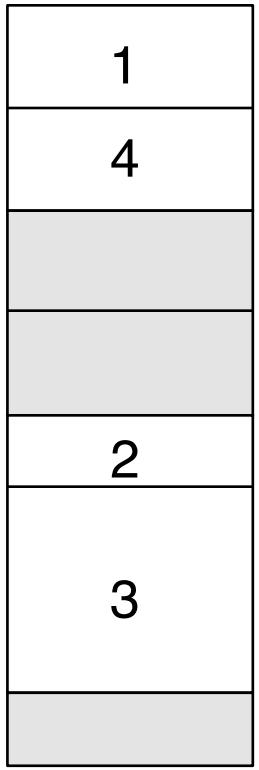




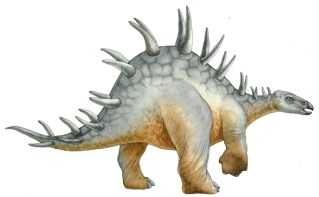
Vista lógica de la segmentación



espacio usuario

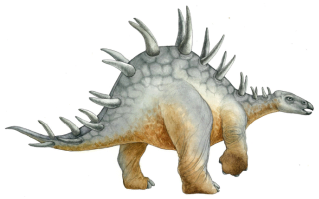


espacio de memoria física



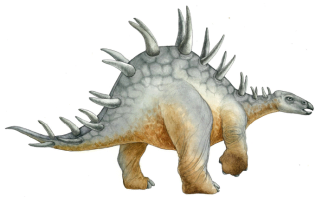
Arquitectura de segmentación

- Dirección lógica consiste de la tupla:
  <número segmento, desplazamiento>
- **Tabla de segmento** – mapea direcciones físicas de dos dimensiones; cada entrada de la tabla tiene:
  - **base** – contiene la dirección física inicial donde residen los segmentos en memoria
  - **límite** – especifica la longitud del segmento
- **Registro base tabla de segmentos (STBR)** apunta a la posición en memoria de la tabla de segmentos
- **Registro longitud tabla de segmentos (STLR)** indica el número de segmentos utilizados por un programa;
  número de segmento ***s*** es legal si ***s*** < **STLR**

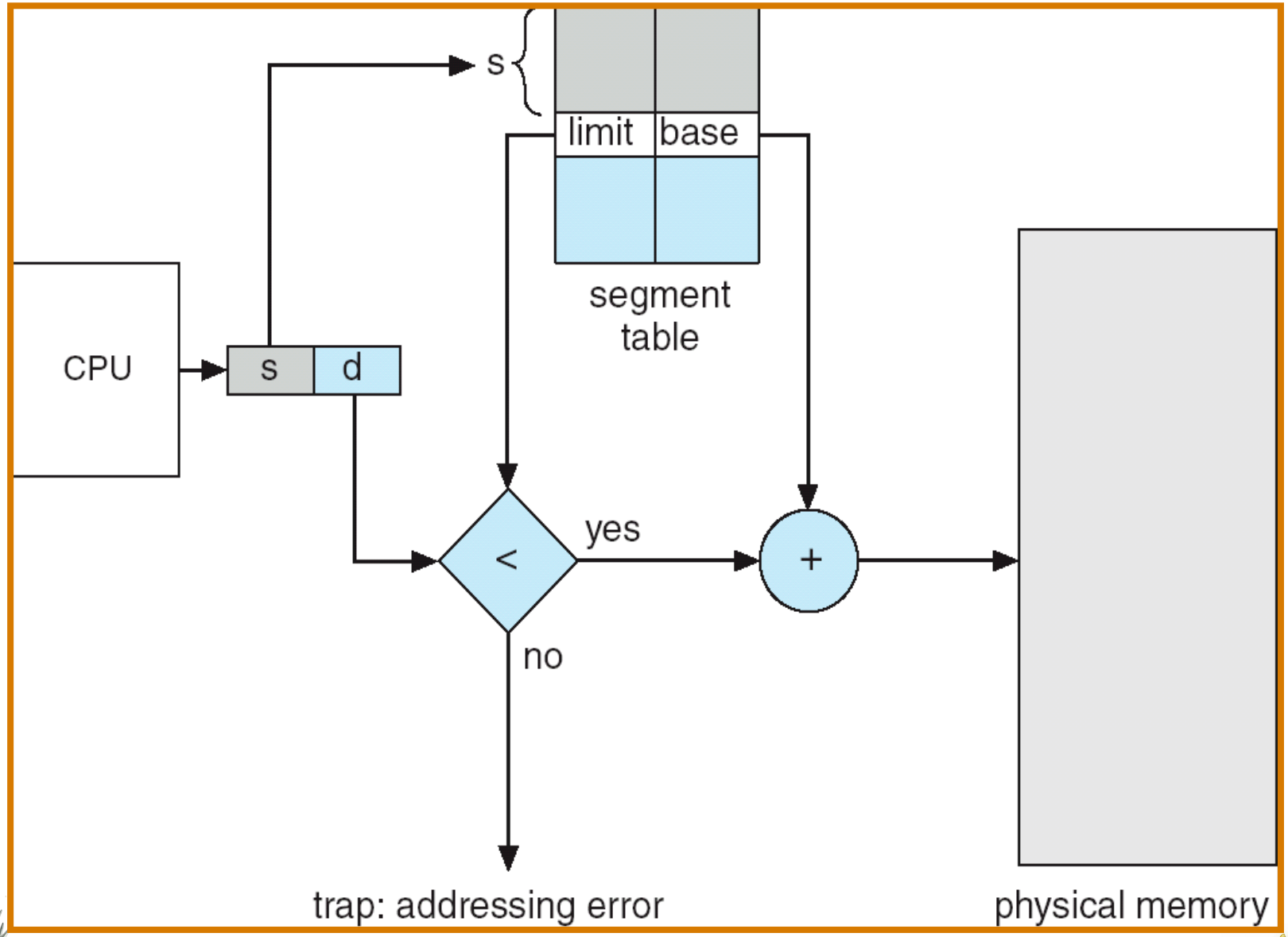


Arquitectura de segmentación (cont.)

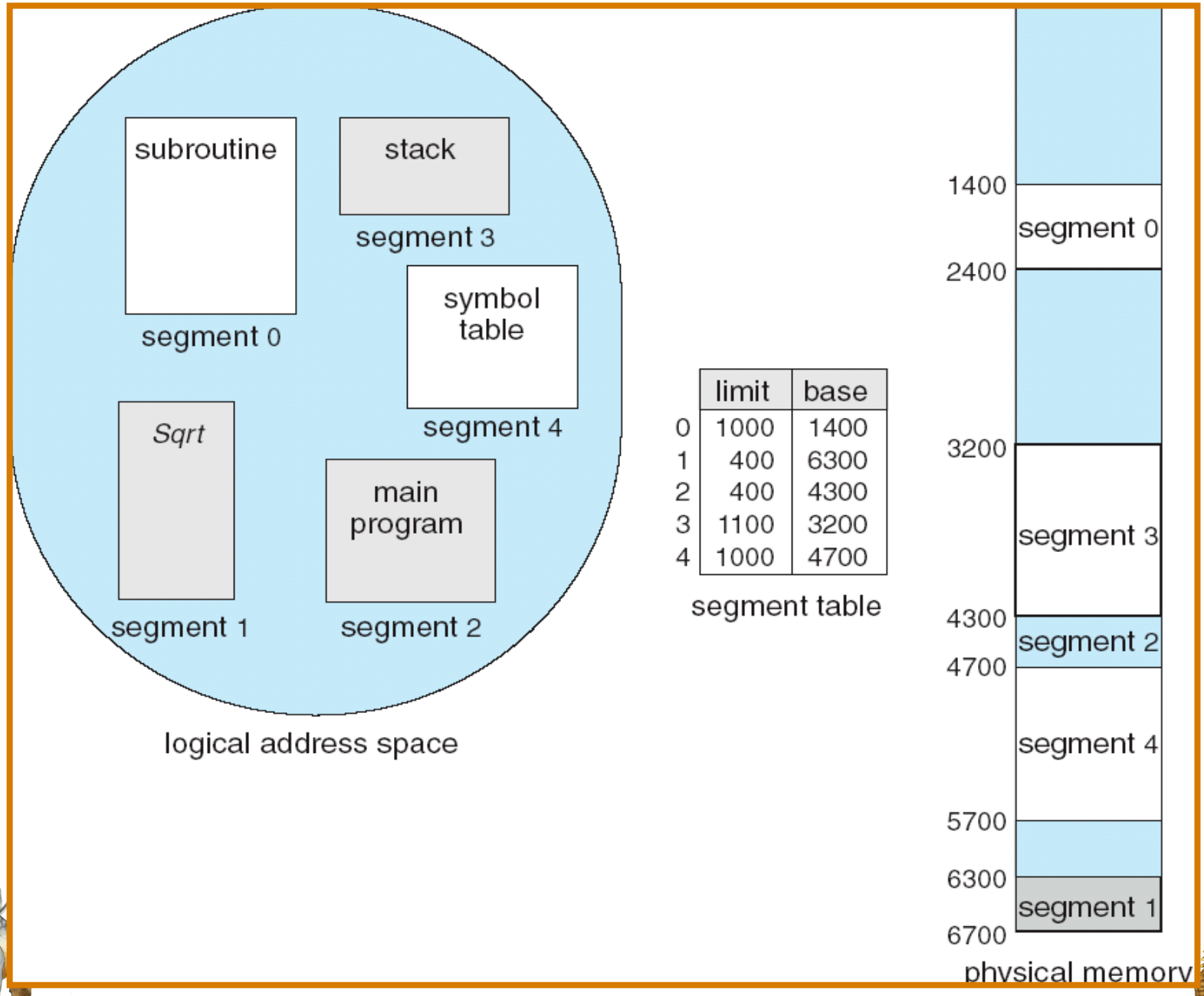
- Protección
  - Asociamos con cada entrada en tabla segmentos:
    - bit de validación = 0 ⇒ segmento ilegal
    - privilegios read/write/execute
- Bits de protección asociados con segmentos; compartir código ocurre al nivel de segmentos
- Dado que los segmentos varían en longitud, la asignación de memoria es un problema de asignación dinámica de espacio
- Un ejemplo de segmentación se muestra en el siguiente diagrama



Hardware de segmentación

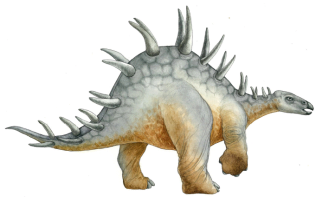


Ejemplo de segmentación

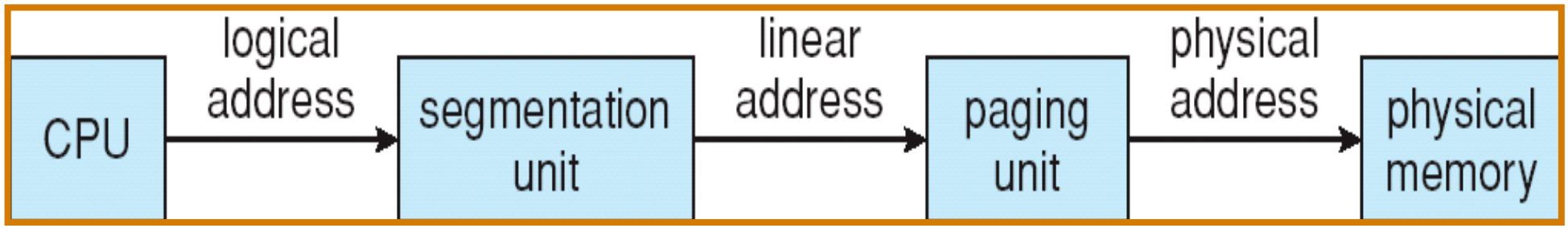


Ejemplo: Pentium Intel

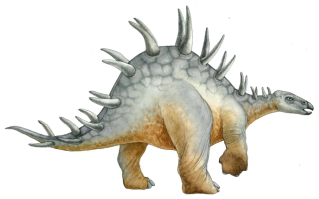
- Soporta tanto segmentación como segmentación con paginación
- CPU genera direcciones lógicas
  - Enviados a unidad de segmentación
    - Que produce direcciones lineales
  - Dirección lineal enviada a unidad de paginación
    - Que genera direcciones físicas en memoria principal
    - Unidades de paginación son equivalentes a MMU



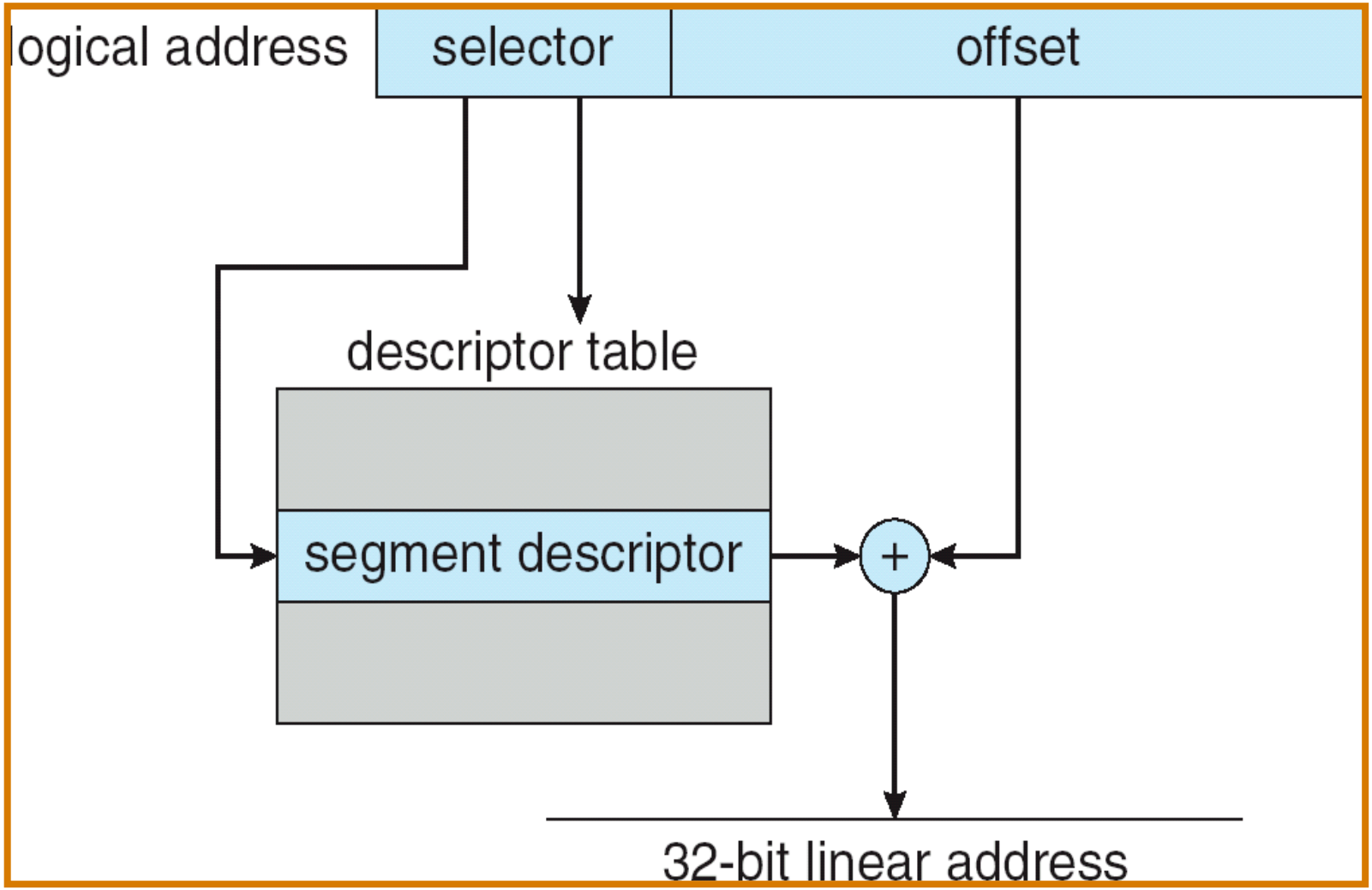
Traducción de direcciones Lógicas a Físicas en Pentium

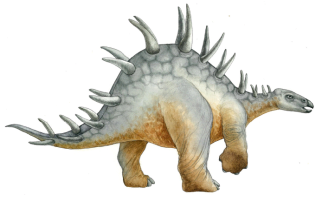


page number		page offset
p_1	p_2	d
10	10	12

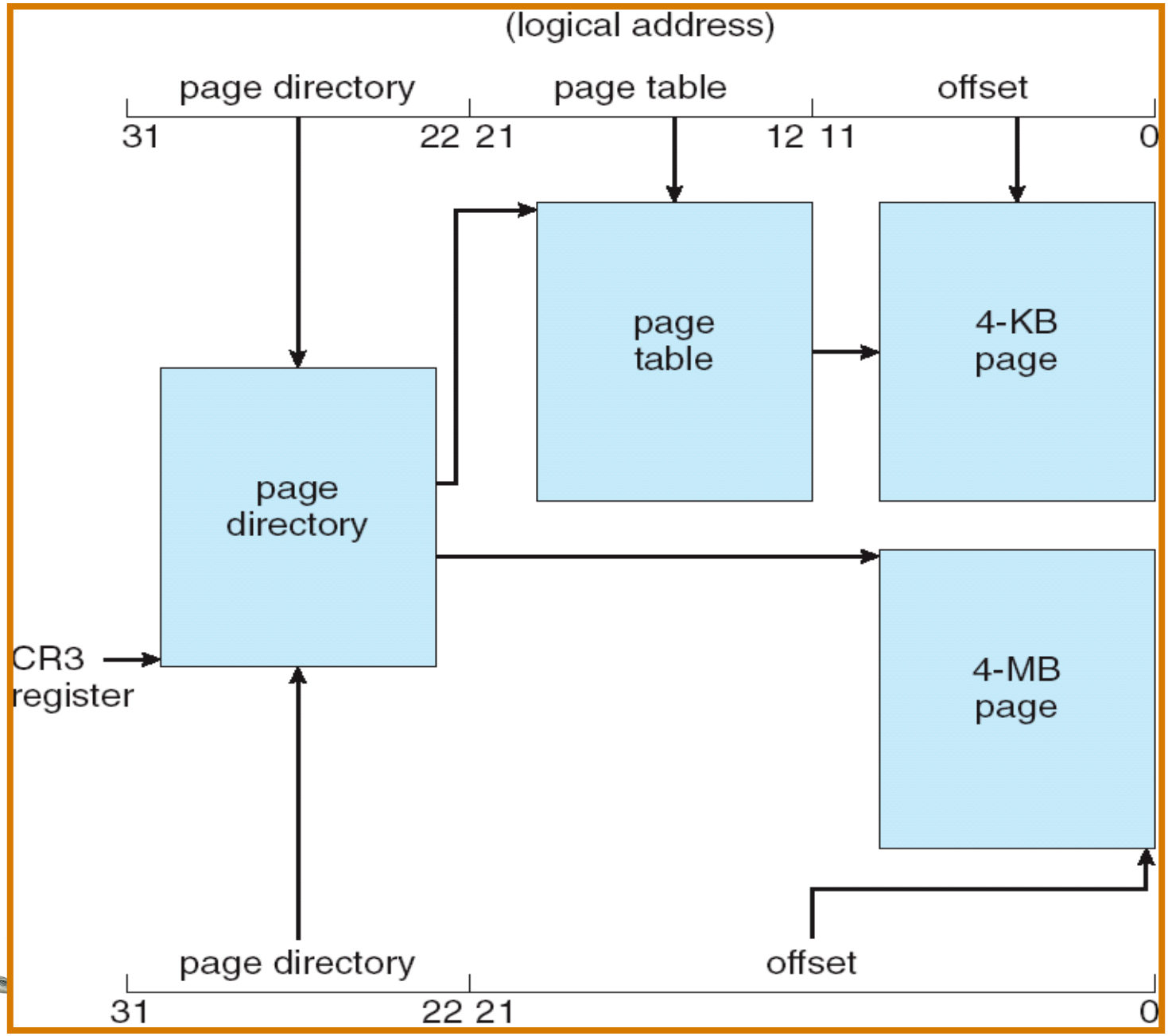


Segmentación en Pentium Intel



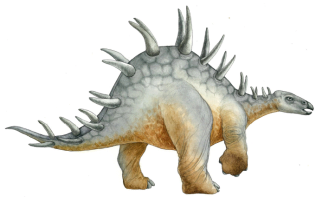


Arquitectura de paginación en Pentium

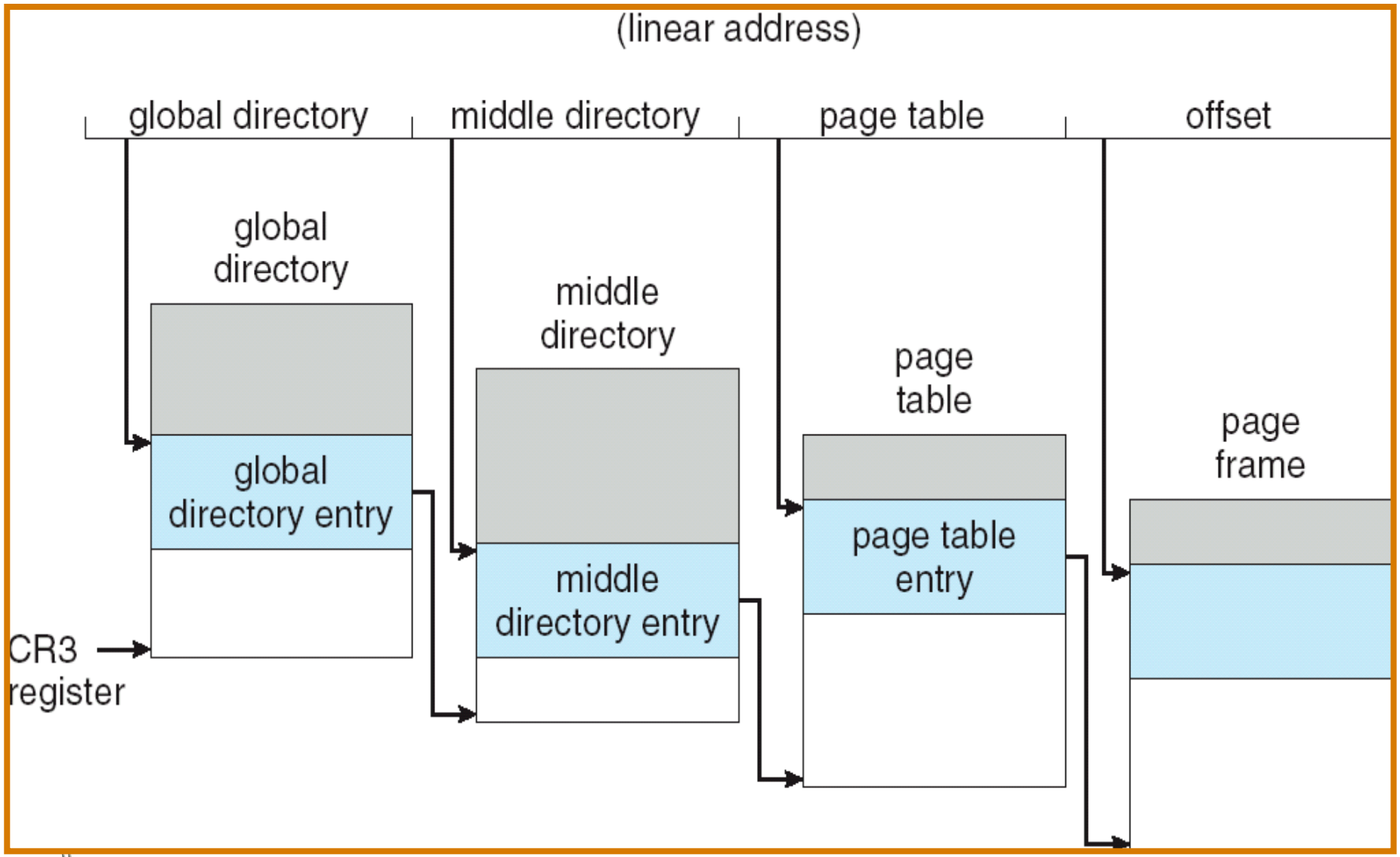


Dirección lineal en Linux

Se rompe en cuatro partes:



Paginación de tres-niveles en Linux



Final del Capítulo 8

Silberschatz, Galvin and Gagne ©2007