

TEMA 3: Hardware de gestión de memoria y E/S

Contenidos

1. Gestión de memoria. Introducción
2. Memoria Virtual. Paginación, Segmentación, Seg con paginación.
3. Gestión de E/S. DMA.
4. DMA. Gestión de tareas
5. DMA. Estructura Hardware
6. DMA. Secuencia de funcionamiento
7. IOMMU. Funcionamiento, Características, Traducción de estructuras, Interface con el Software

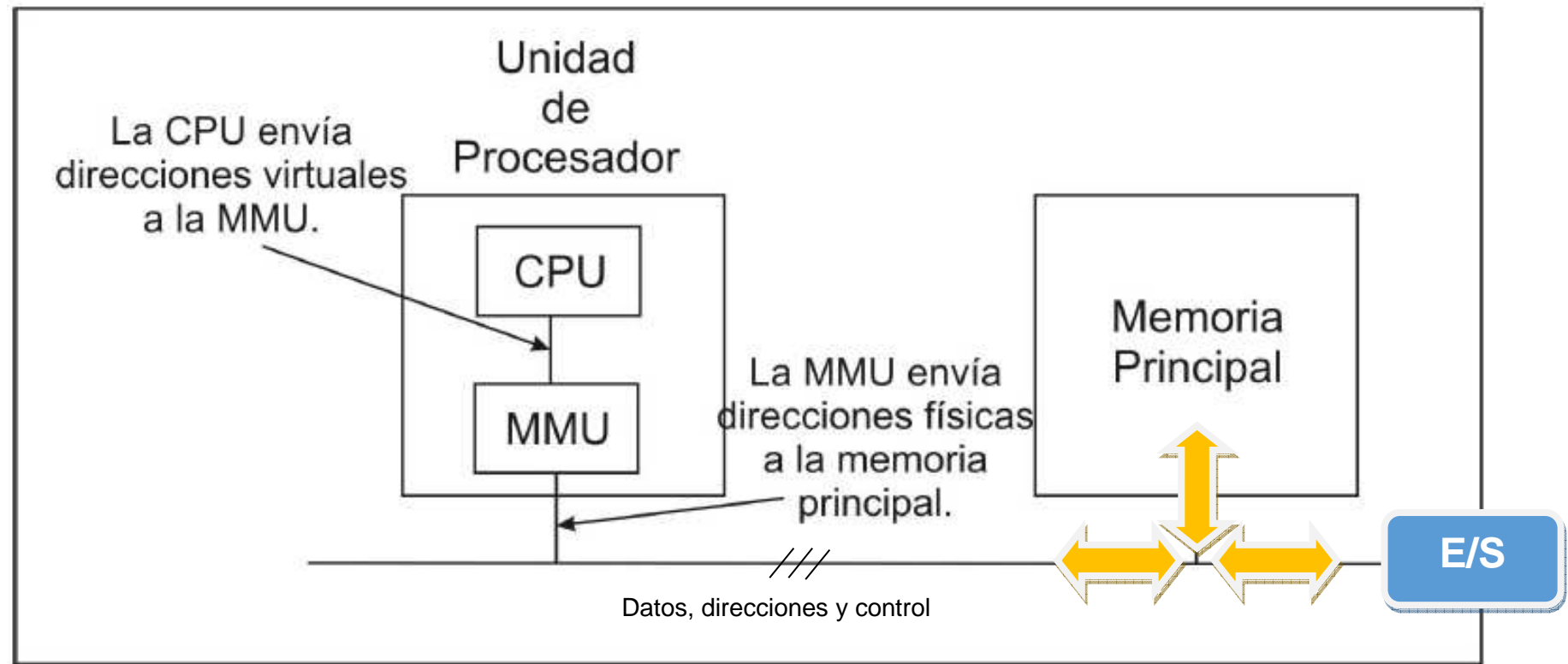
1. Gestión de Memoria. Introducción

- La necesidad de ejecutar programas que requerían más memoria de que se disponía en el sistema generó la técnica de memoria virtual propuesta por Fotheringham en 1961.
- La idea es mantener en memoria física solamente la memoria que cada **proceso** está utilizando.
- De esta forma, el programador se despreocupa de la limitación de memoria principal que imponía el sistema.
- La técnica de **memoria virtual** abstrae la memoria principal en una gran matriz uniforme de bytes.
- Si bien es una técnica muy potente, su uso descuidado puede generar una **degradación** importante del sistema.

- ¿Por qué hay que tener en cuenta estas técnicas al estudiar la arquitectura de E/S de los computadores?
 - El acceso a los dispositivos de E/S se realiza a través de buffers (Espacios dedicados a E/S) dentro de la memoria.
 - Estos buffers deben ser gestionados por la **unidad de administración de memoria (MMU)**. ¿No era E/S?
 - Existirán **frames de memoria** dedicados a E/S.
 - En el fondo, al procesador le “da igual” de dónde procedan los datos. A “quien” le interesa realmente es al Chipset

Cómo funciona todo esto (de forma sencilla)

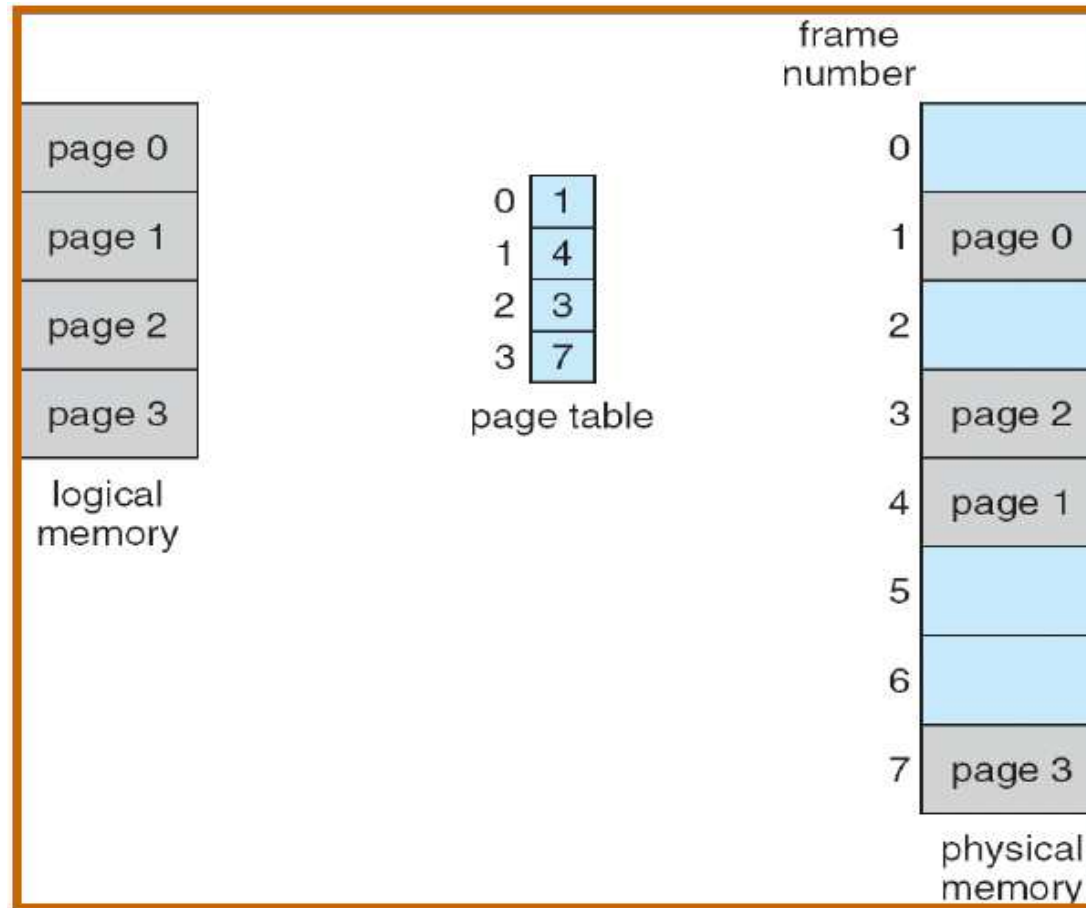
- Cada “proceso” tiene un espacio de direccionamiento virtual. (Jerarquía de memoria: cachés, memoria virtual,...)
- ... pero el procesador necesita trabajar con datos que estarán colocados en **direcciones de memoria que se localizan de forma absoluta** dentro el espacio de direccionamiento.
- Ej.: en un sistema de 32 bits un proceso y realizando un mapeo directo de direcciones, un proceso tiene un espacio de direccionamiento virtual de $32^2 = 4\text{GiB}$.
- La unidad de administración de memoria (MMU – Memory Management Unit) es la encargada de realizar la traducción de direcciones virtuales a físicas.



La clave es: gestionar datos de E/S dentro de una estructura de memoria. Lo importante (en este momento) es que los datos **no** se “pierdan” dentro del aparente “caos” existente dentro de la jerarquía de memoria. ¿Cómo hacerlo? **Tablas**, tablas ...

2.1 Memoria virtual: paginación (repaso)

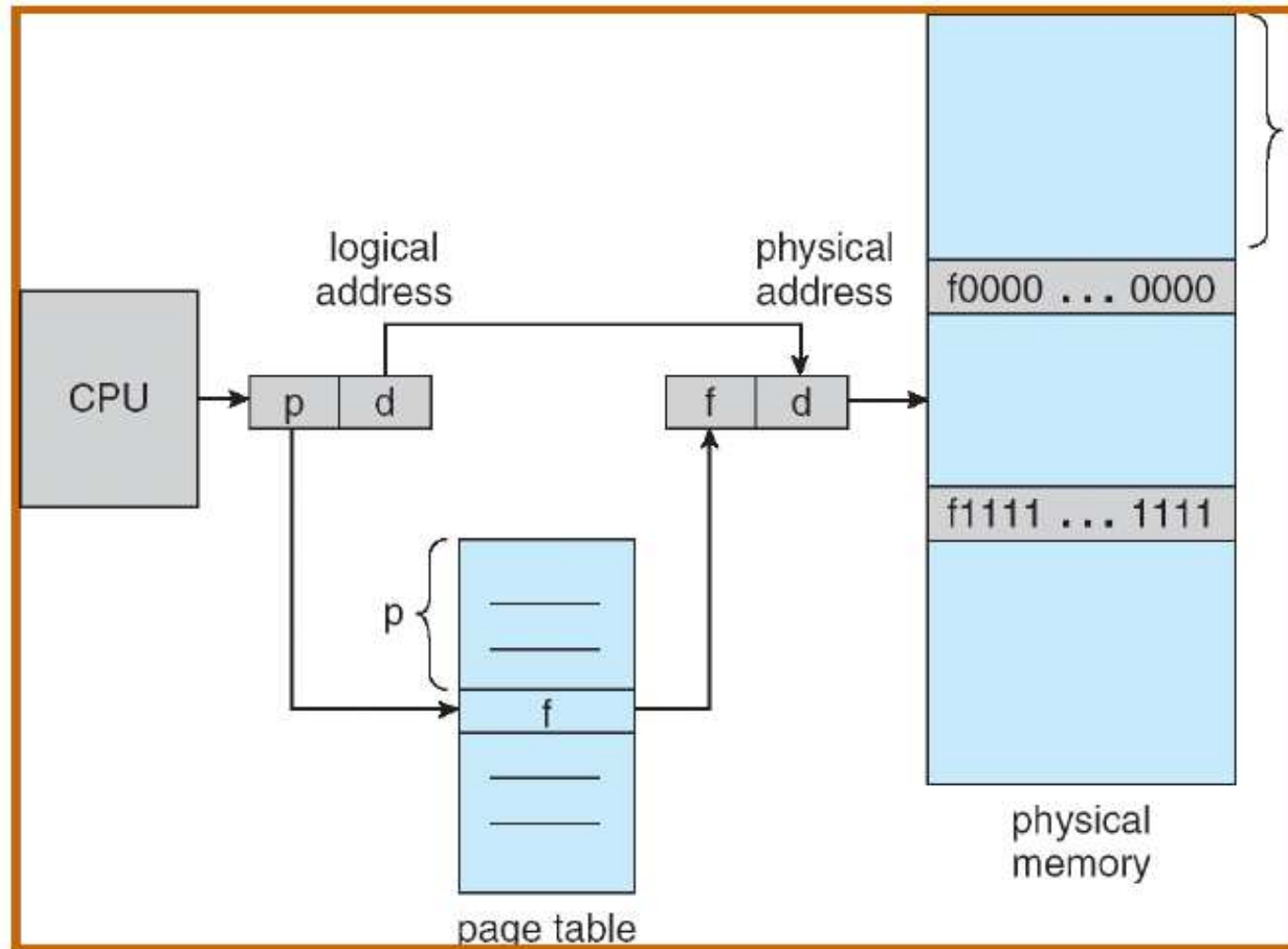
- La paginación es una técnica que divide a la memoria física en particiones de tamaño fijo llamados frames.
- A su vez, el espacio de direccionamiento virtual es dividido en unidades fijas del mismo tamaño que los frames (page size) denominadas páginas (**pages**).
- La transferencia entre la memoria principal y el disco se realiza en unidades de página.
- En necesario “traducir” las direcciones para localizar los datos. (**page table**)



- Las direcciones virtuales en paginación se componen de un número de página (page number) y un desplazamiento (offset).
- El número de página es un **índice** sobre una tabla de páginas (page table) y el **desplazamiento** es la referencia dentro del frame.

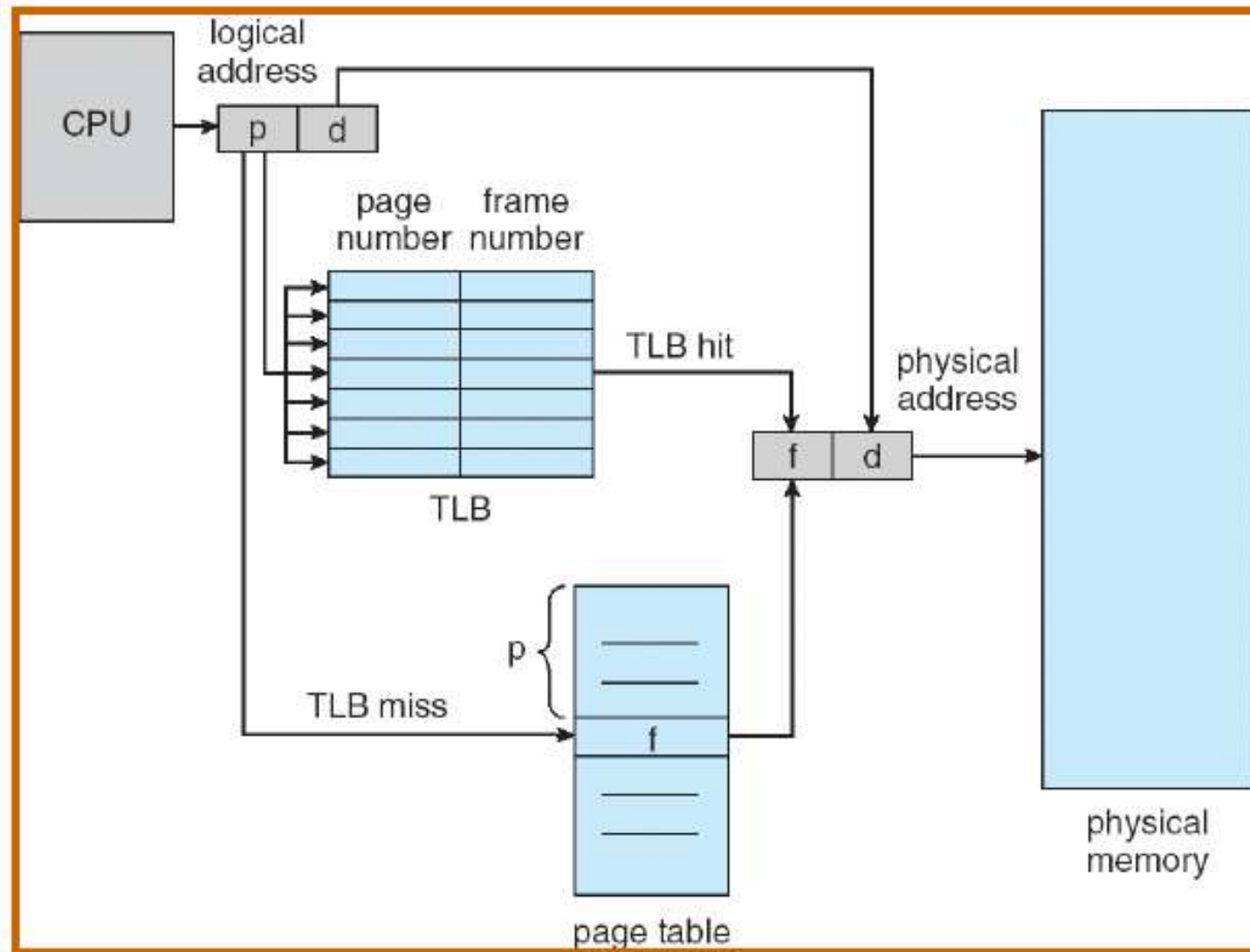
page number	page offset
p	d
$m - n$	n

- La tabla de páginas contiene el mapeo del frame correspondiente.
- El tamaño de la página es 2^n .



- En los sistemas actuales no es posible guardar todas las entradas en registros de rápido acceso. Para ello se usa espacios de memoria.
- La tabla se mantiene en memoria principal y se asigna un registro que apunta a la dirección base de la tabla (**PTBR** – **P**age **T**able **B**ase **R**egister). (Sist. Operativo)
- La opción del PTBR mejora el cambio de contexto entre procesos ya que solo es necesario cambiar un registro para acceder a la tabla de página.
- Sin embargo, el acceso a memoria se duplica debido a que es necesario primero acceder a la tabla para obtener el número de frame y, posteriormente, al lugar de memoria solicitado.

- La solución es utilizar una cache de la tabla de página: TLB (**T**ranslation **L**ook-aside **B**uffer).
- El cache TLB es asociativa y de rápido acceso.
- Cada entrada consiste de dos partes: una clave (tag) y un valor (el número de frame).
- La búsqueda de un clave en la cache TLB es simultánea entre todas las tags.
- Si la clave es encontrada (TLB hit), inmediatamente se genera la dirección buscada a partir del valor asociado. En caso contrario (TLB miss), es necesario realizar el acceso a memoria para obtener la entrada. Posteriormente, se guarda el valor obtenido en la cache TLB para posteriores accesos (principio de localidad).

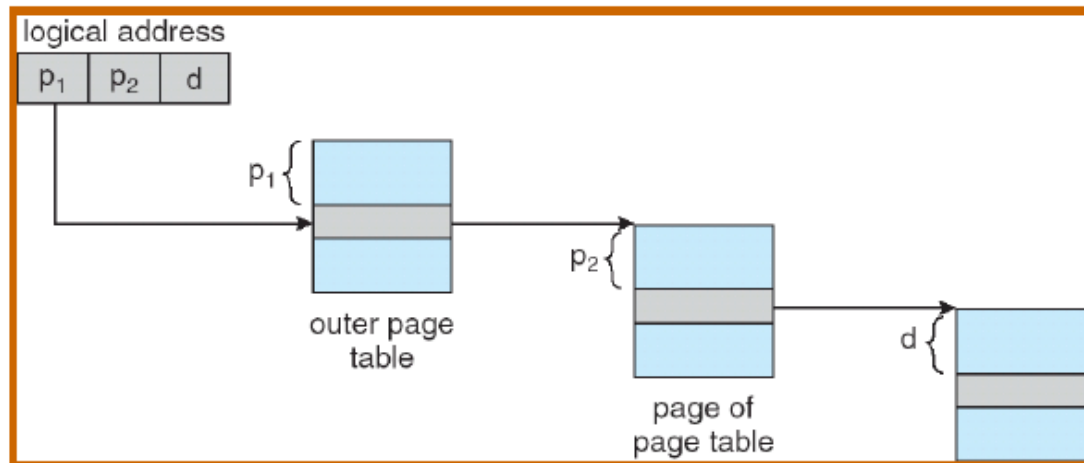


Las TLB por lo general tienen pocas entradas (64 a 1024).

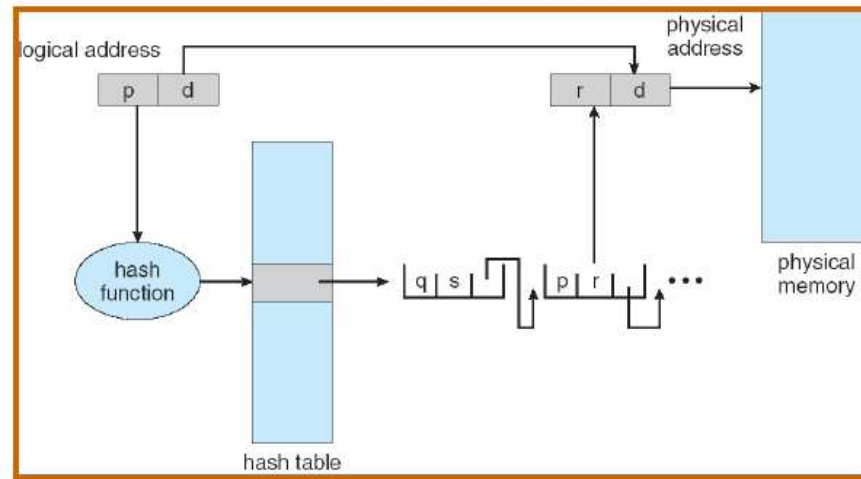
- Algunas caches TLB agregan a cada entrada un **identificador de espacio de direccionamiento** (ASID – Address Space Identifier).
- En la búsqueda de una clave solo serán tenidas en cuenta las entradas cuyo ASID coincida con el del proceso que está ejecutando en el procesador.
- El uso del identificador permite que en la cache TLB contengan entradas para varios procesos de forma simultánea.
- Si no se utiliza el ASID, en cada cambio de contexto es necesario “limpiar” las entradas de la TLB, sino se realizarían accesos equivocados a memoria.

- EJ: En un sistema de 32bits que utilice páginas de 4KB se necesitarán cerca de 1 millón de entradas en la tabla de página.
- Es necesario buscar alguna estructura más eficiente en cuanto al tamaño ocupado por la tabla de página.
- Respuesta: Se utilizan las siguientes **estructuras** (->)

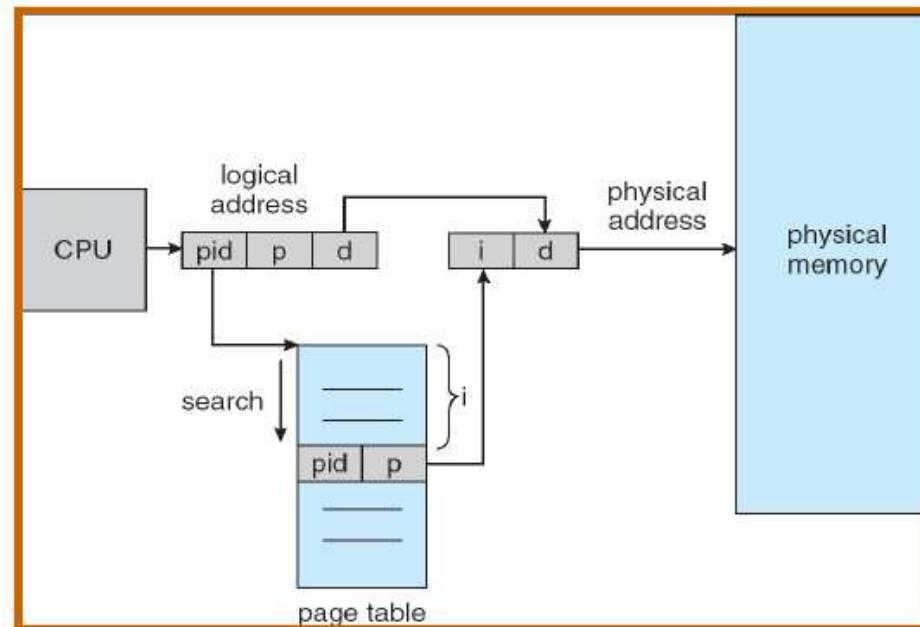
○ Jerárquica



○ Diccionarios



○ Invertida



Contestar las siguientes preguntas:

GA: ¿Qué es un proceso? (dentro de la ejecución de programas)

GB: ¿Qué es memoria virtual? Características

GC: ¿Cómo definirías “degradación de memoria” virtual?

GA: ¿Para qué se emplea una MMU? Por un ejemplo.

GB: ¿Cómo se define un frame de memoria? ¿Cómo se localiza?

GC: ¿Qué es direccionamiento absoluto de memoria? Por un ejemplo

GA: ¿Cómo podemos crear una tabla en una memoria?

GB: ¿Cómo se define una página de memoria?

GC: ¿Cómo se gestiona una tabla de páginas de memoria (Page Table)?

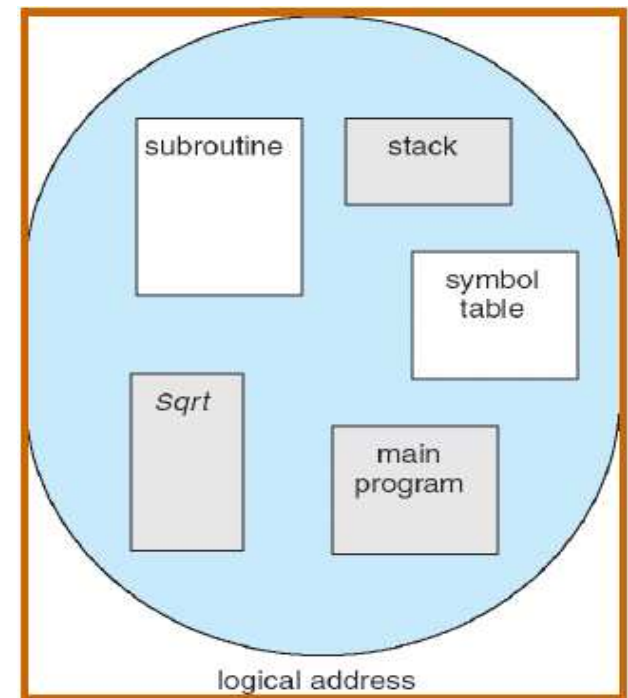
GA: Diferencia entre índice (index) y desplazamiento (offset)

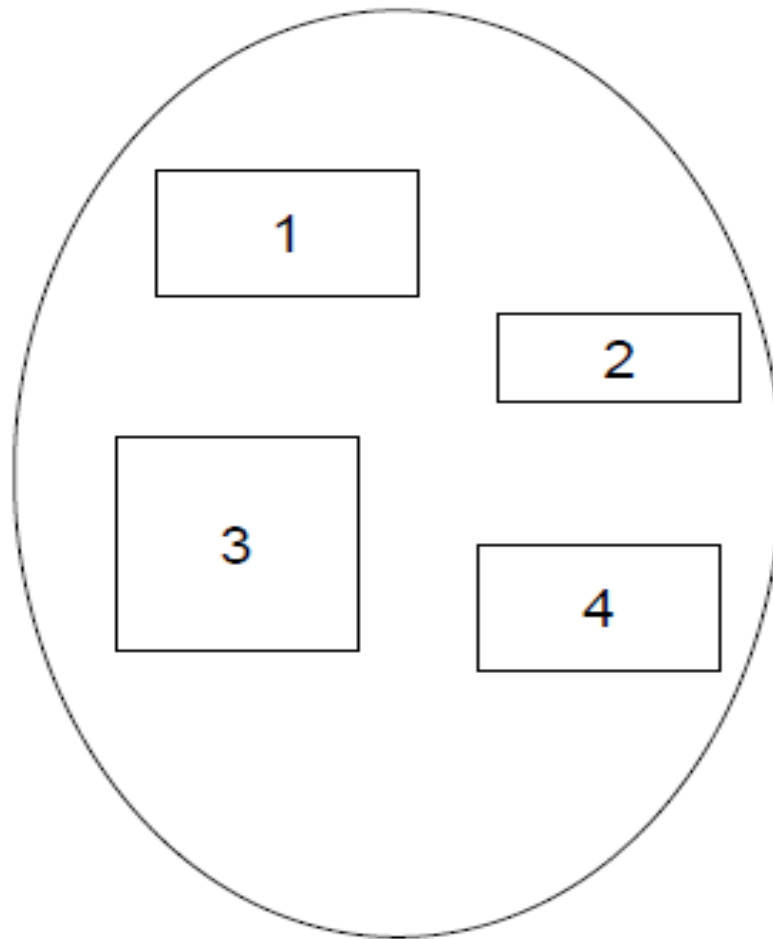
GB: Para poder gestionar 16 PTBRs... ¿Cuánta memoria tendríamos que utilizar en un sistema MIPS (tamaño de TLB)?

GC: ¿Cómo podríamos incorporar ASIDs a la tabla TLB?

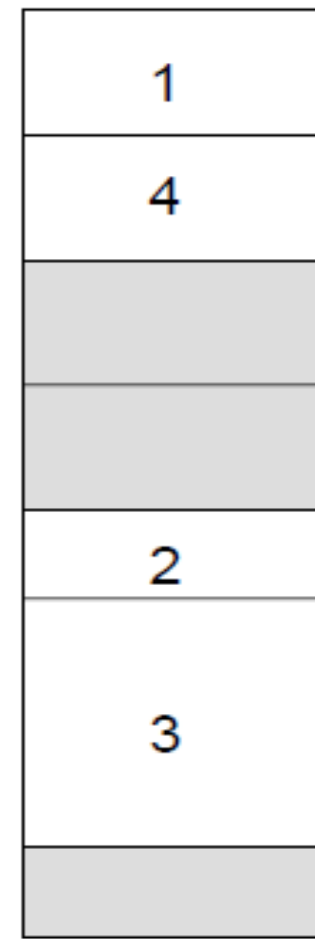
2.2 Memoria virtual: Segmentación (repaso)

- La segmentación es una técnica que asigna segmentos memoria para espacios funcionales de un proceso.
- Es más sencillo de entender desde el punto de vista del usuario-diseñador.
- Un proceso puede utilizar varios segmentos Ej: un segmento para código, una pila (stack), un espacio para la memoria auxiliar (heap), la tabla de símbolos, Espacio de E/S, etc.



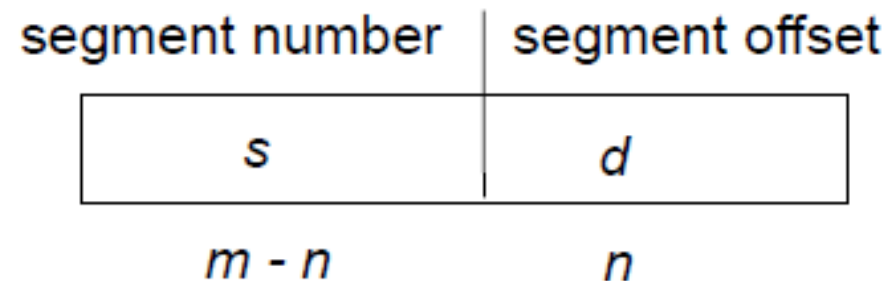


Visión del usuario



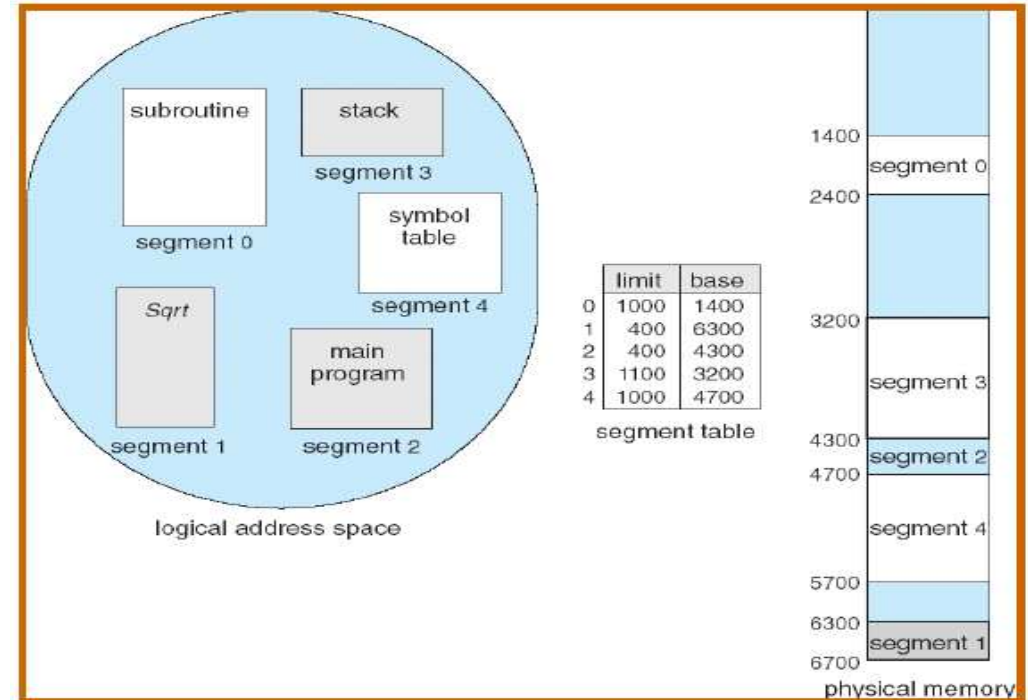
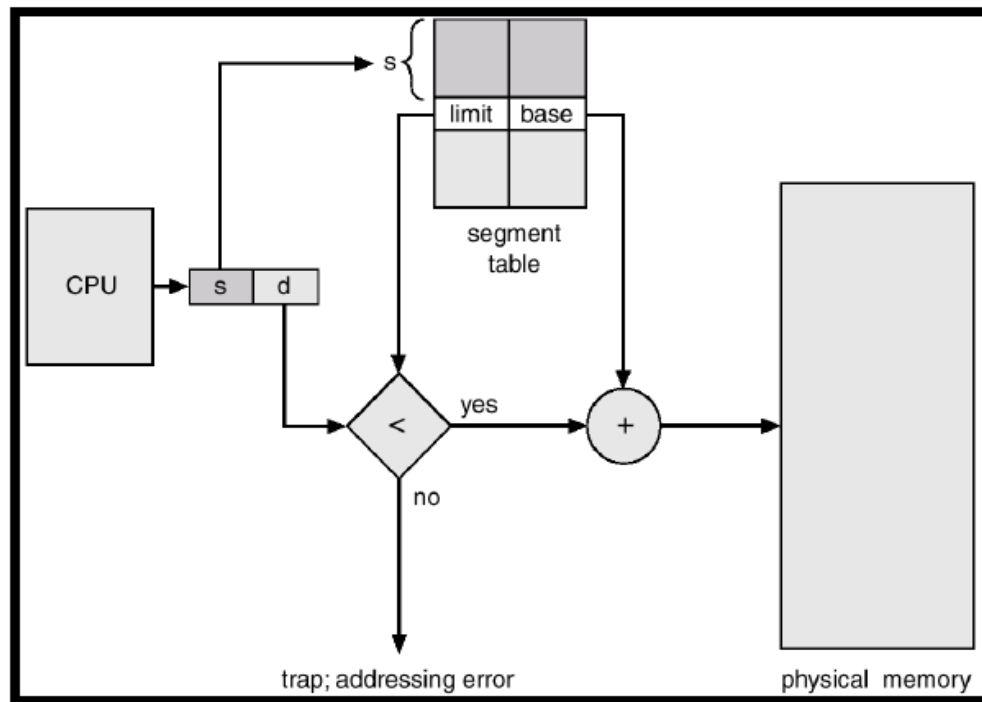
Memoria principal

- Cada segmento tendrá un “nombre” asociado que lo identificará. (código, número) y un tamaño máximo.
- Las direcciones virtuales se componen de un número de segmento y el desplazamiento dentro del segmento.



- El desplazamiento debe ser menor que el tamaño asociado al segmento


- La tabla de segmentos tendrá una entrada por cada segmento, en donde estará la dirección física base del segmento (base register) y el largo del mismo (limit register).
- En la traducción de dirección virtual a física se controlan el número de segmento con el máximo que tenga el proceso (el registro STLR – Segment Table length Register define el número máximo de segmento utilizado por el proceso) y el desplazamiento contra el registro límite.
- La tabla de segmentos es mantenida en memoria principal y se asigna un registro que apunta a la dirección base de la misma (STBR – Segment Table Base Register).



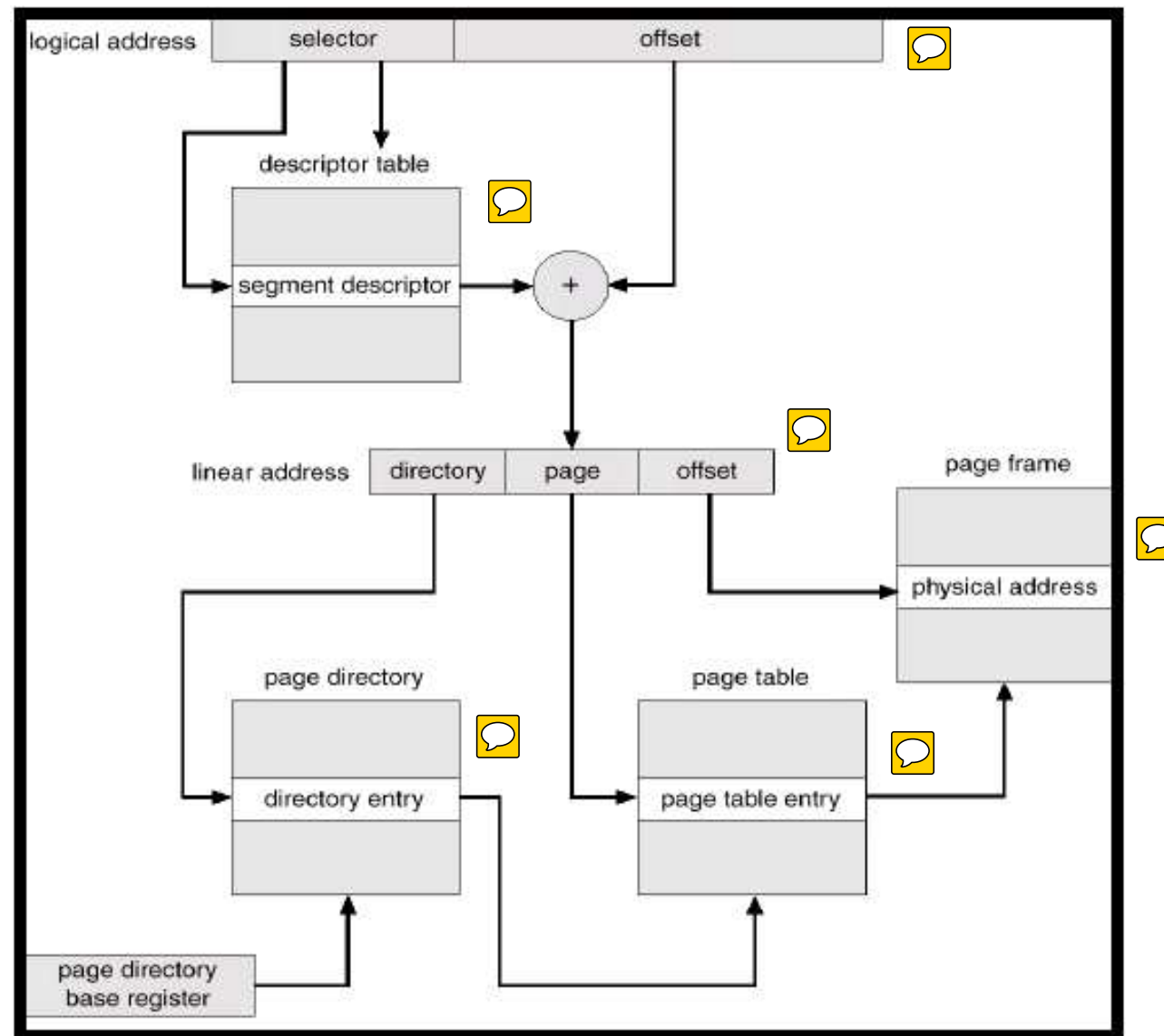
- Dividir la memoria en segmentos permite que cada segmento tenga asociado un conjunto de permisos de lectura y escritura.
 - Escritura: Bits de protección.
 - Lectura: Bits de privilegio.

2.3 Realidad: Segmentación con paginación

La paginación y la segmentación se combinan para potenciar las ventajas de cada técnica.

La memoria es segmentada, y los segmentos se conforman de páginas. 

Las direcciones virtuales contienen un identificador de segmento y un desplazamiento. A partir de ellos se genera una dirección lineal de 32 bits (en caso IA32). Luego, la dirección es traducida a una dirección física.



Esquema de paginación + segmentación.

Desarrollar las siguientes cuestiones:

Pensemos en una **estructura** de memoria paginada-segmentada según el esquema anterior. Vamos a diseñar poco a poco una MMU (iisencilla!!) en torno a una memoria, para posteriormente implementarla en las prácticas. Primeros pasos

1. Definir qué elementos nos hacen falta para poder crear un sistema de gestión de memoria paginada-segmentada a nivel de hardware (chipset). Utilizar el esquema previo como guía.
2. Realizar un esquema funcional (bloques) básico que sirva para desarrollar e implementar lo realizado en las prácticas.
3. Proponer cómo modificar la estructura de paginación previa para introducir un sistema de "permisos" de acceso
 - ¿Dónde los almacenaré?
 - ¿Qué bits me harán falta?
 - ¿Cómo se gestionarán? – Elementos de control necesarios
4. Guardar las conclusiones en el Campus virtual. Se utilizará el mismo proceso de desarrollo que hemos realizado hasta ahora.
Propuesta -> mejora -> Conceptos asociados

1) Trabajo en On-Line (1). Cada alumno responde a los posts

- a. Grupo A revisa y mejora las propuestas del grupo B
- b. Grupo B revisa y mejora las propuestas del grupo C
- c. Grupo C revisa y mejora las propuestas del grupo A

2) Trabajo en On-Line (2). Cada alumno pública su lista de conceptos que se estimen necesarios para realizar el proceso de diseño. Se realiza, como antes, dentro de los hilos creados para ello.

- a. Grupo A determina los conceptos fundamentales del grupo C
- b. Grupo C determina los conceptos fundamentales del grupo B
- c. Grupo B determina los conceptos fundamentales del grupo A