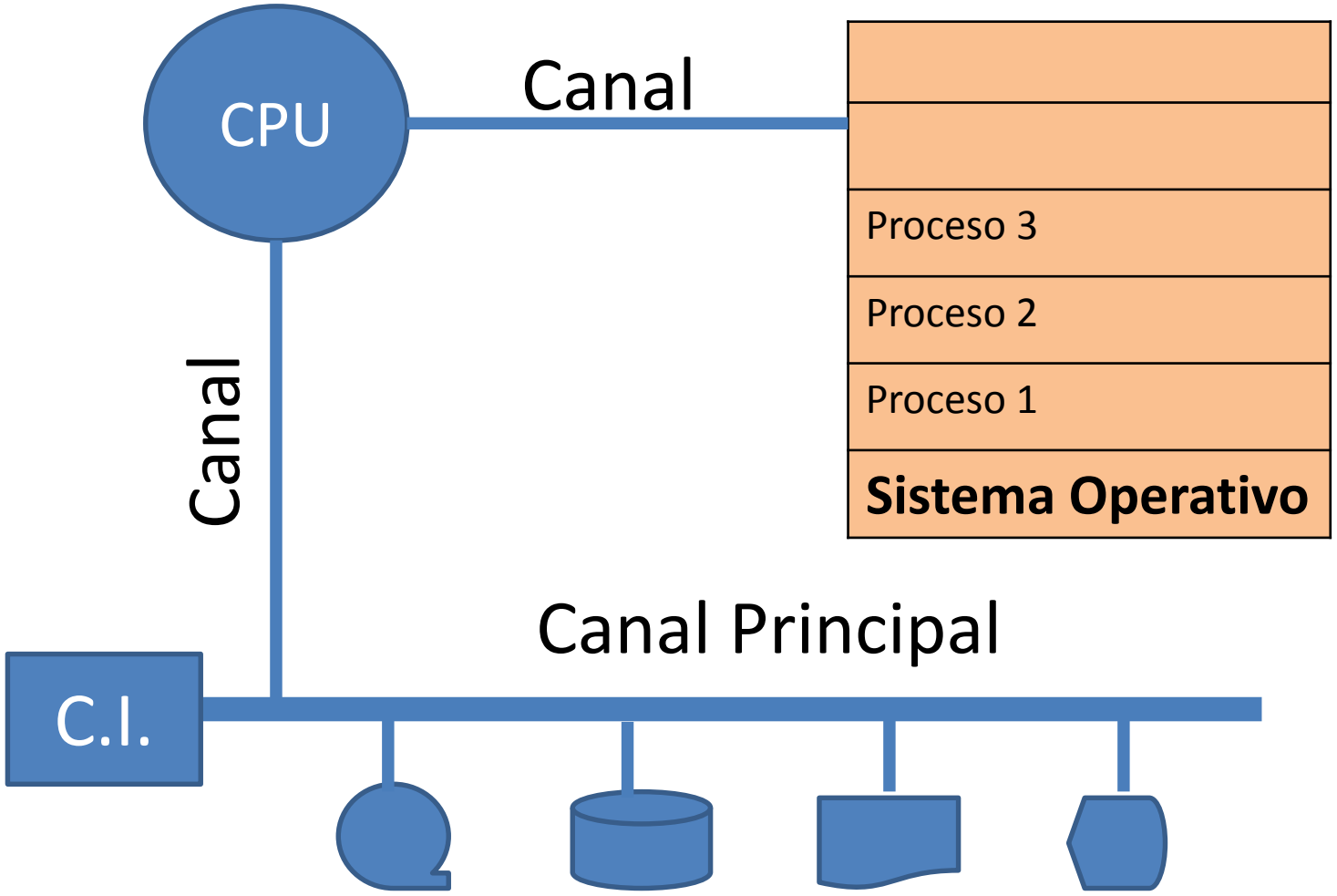
Sistemas Operativos

Cursada 2022

Comisión S21 y S22

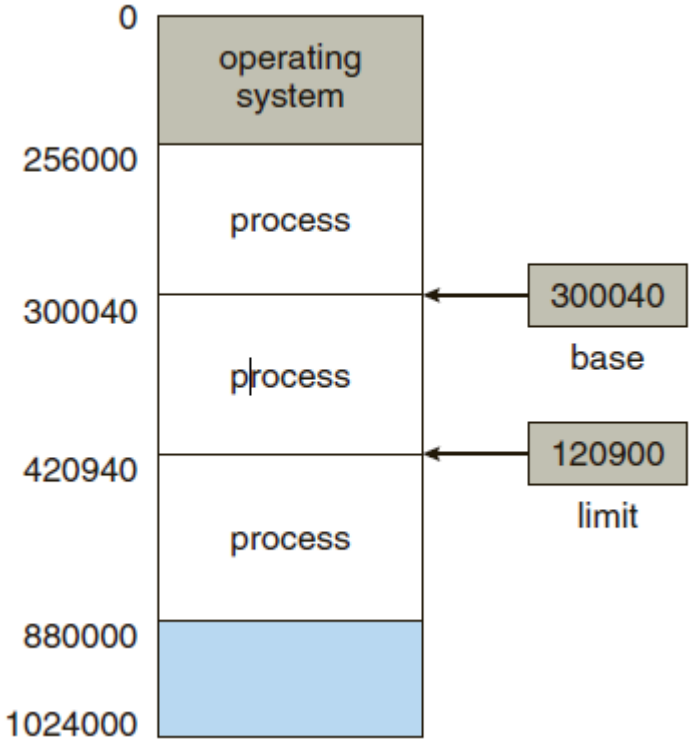

CPU
Canal
Proceso 3
Proceso 2
Proceso 1
Sistema Operativo
Canal
Canal Principal
C.I.

Memoria

Asignación Contigua de Memoria

En este tipo de asignación se producen huecos
Hay algunas técnicas para la asignación del espacio

- **Primer Ajuste:** Asigna el 1ero lo suficientemente grande
- **Mejor Ajuste**: El mas pequeño lo suficientemente grande
- **Peor Ajuste:** Asigna el hueco mas grande

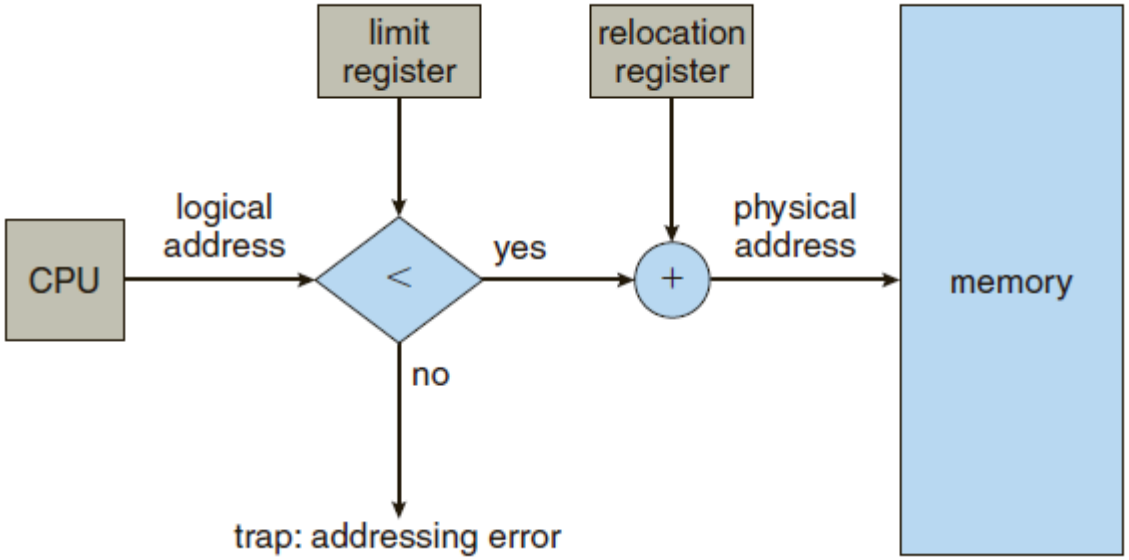


Memoria

Asignación Contigua de Memoria

La memoria se divide en lo que es el S.O. y los procesos de usuario.

Debemos proteger la memoria



Memoria

Asignación Contigua de Memoria

Cuando aparece la multiprogramación, varios procesos en memoria, por lo tanto debemos cambiar la forma de administración de memoria.

Método desarrollado por IBM (360/OS)

Hay dos formas de administración en lo que denominamos asignación contigua

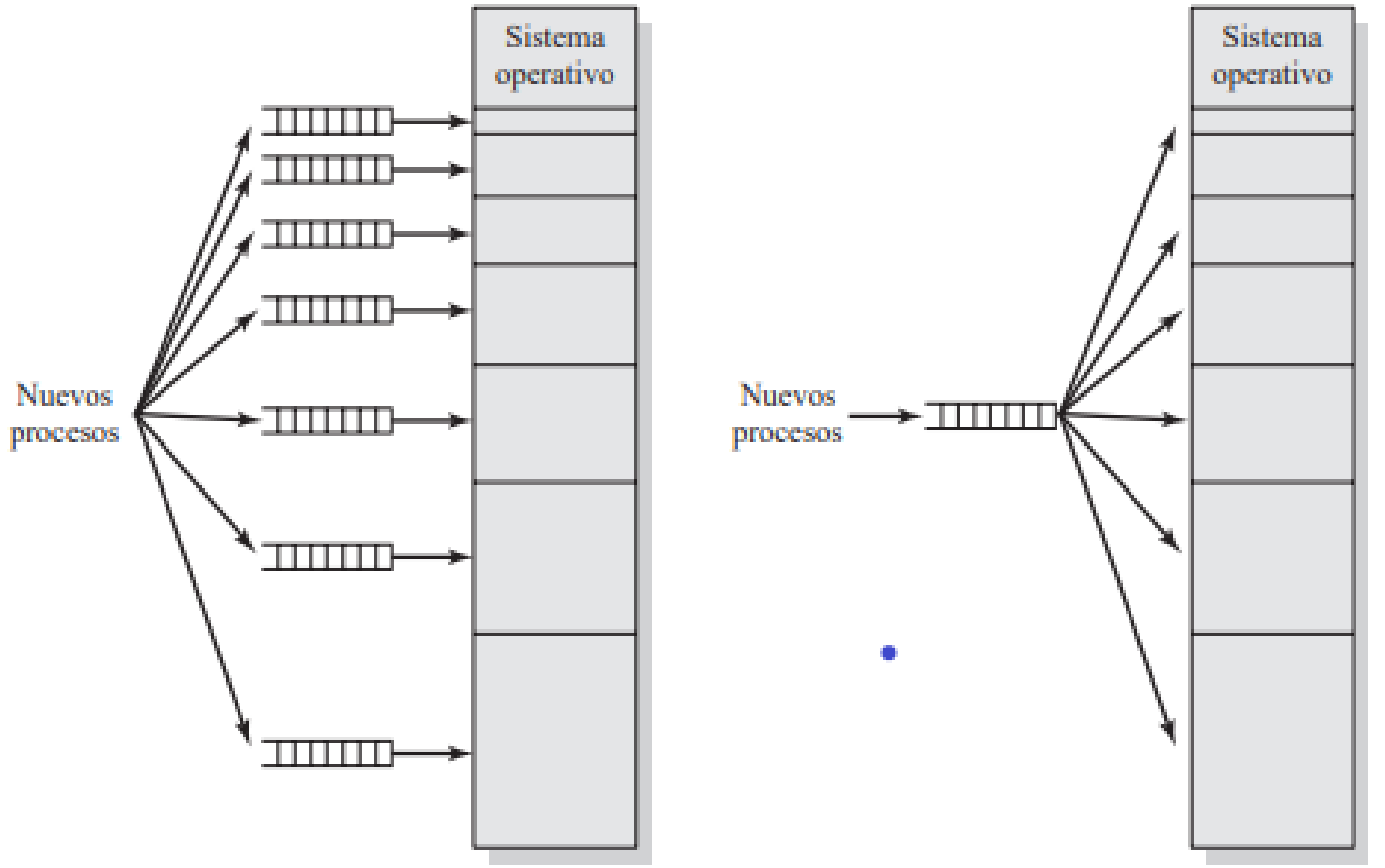
- **MFT (multiprogramación con numero fijo de tareas)**
- **MVT (multiprogramación con numero Variable de tareas)**

Memoria

- **MFT (multiprogramación con numero fijo de tareas)**

Se divide la memoria en particiones

Lo realiza el operador con un comando del S.O.



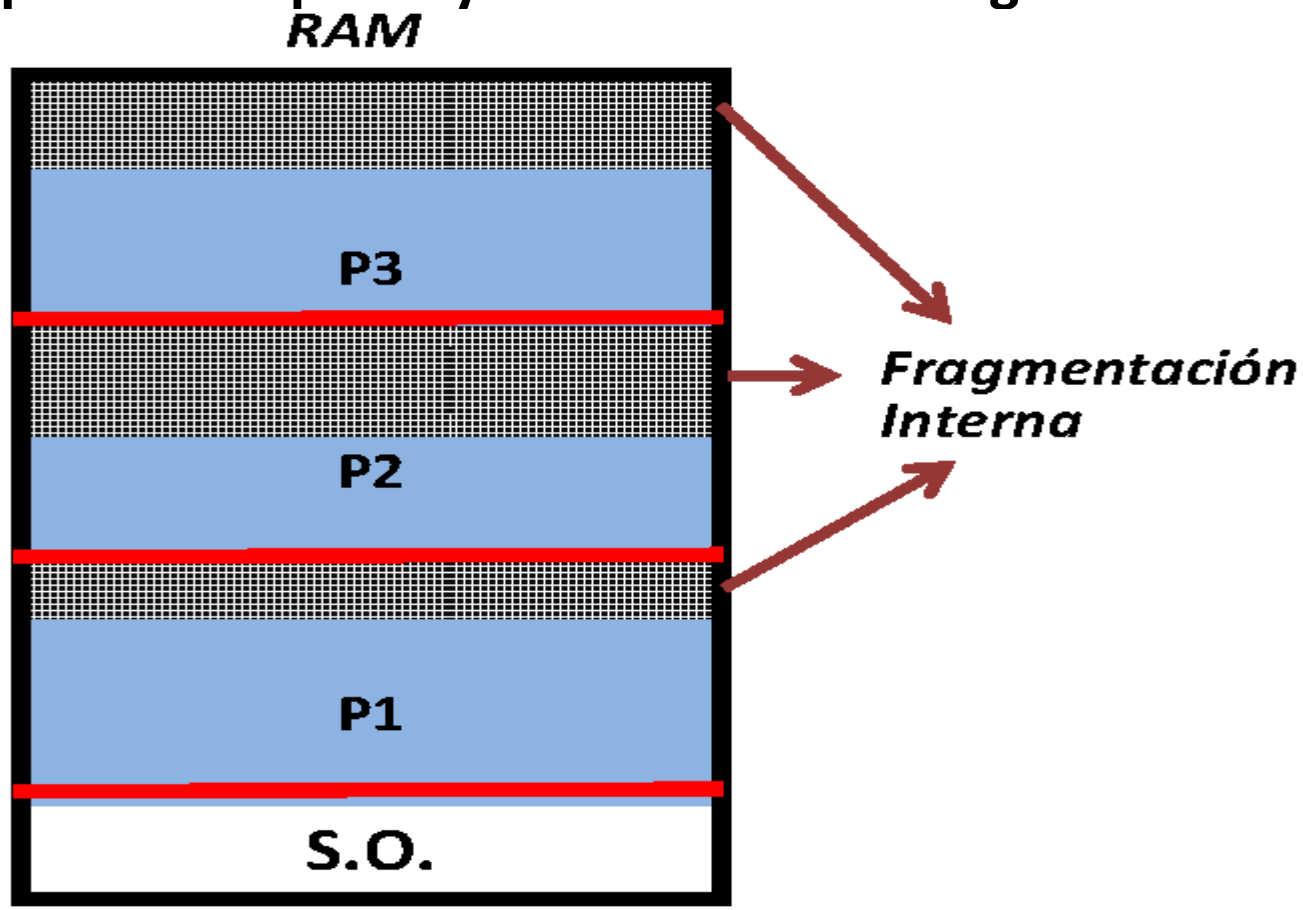
(a) Una cola de procesos por participación

(b) Un única cola

Memoria

➢ MFT (multiprogramación con numero fijo de tareas)

- El operador tiene una carga de trabajo preestablecida
- Decide de acuerdo a las prioridades que procesos carga
- Se evalúan los tipos de procesos que tiene para ejecutar
- Cual es el problema que hay con esta metodología



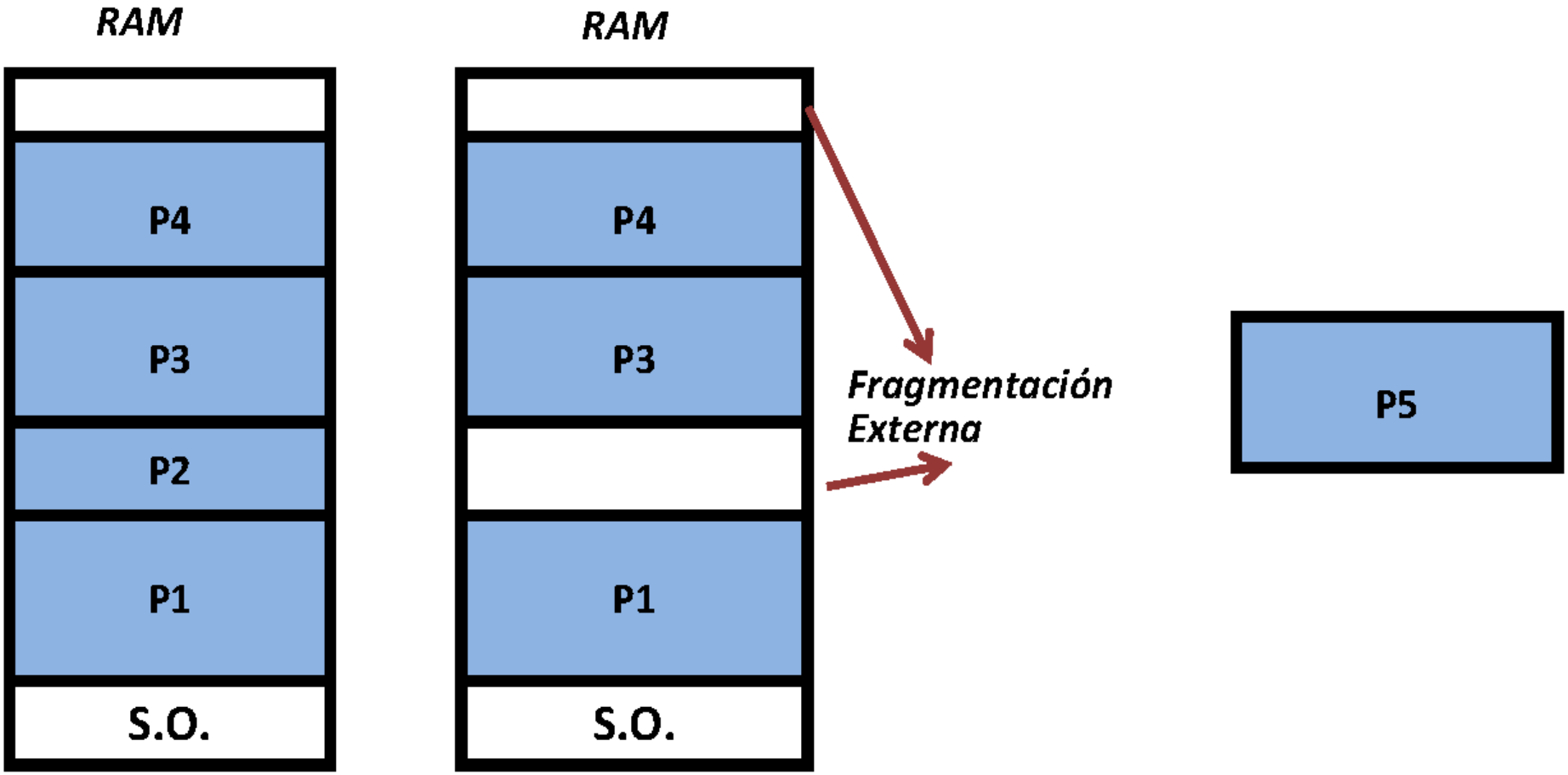
Memoria

- **MVT (multiprogramación con numero Variable de tareas)**

En este caso la RAM se divide en particiones de longitud variable

Aparece el planificador de largo plazo

Las particiones son creadas por el S.O.



Memoria

Llamamos *fragmentación interna* a la pedida de un recurso por estar asignado a un proceso que no lo utiliza.

Llamamos *fragmentación externa* a la pedida de un recurso por no estar asignado a ningún proceso, porque este recurso no alcanza a satisfacer las necesidades que requieren dicho/s proceso/s.

Memoria (Paginación Simple)

Llamamos *Paginación Simple* a la técnica por el cual el espacio de direcciones físicas de un proceso no es contiguo.

Las paginas de un proceso se divide en *paginas* de igual tamaño

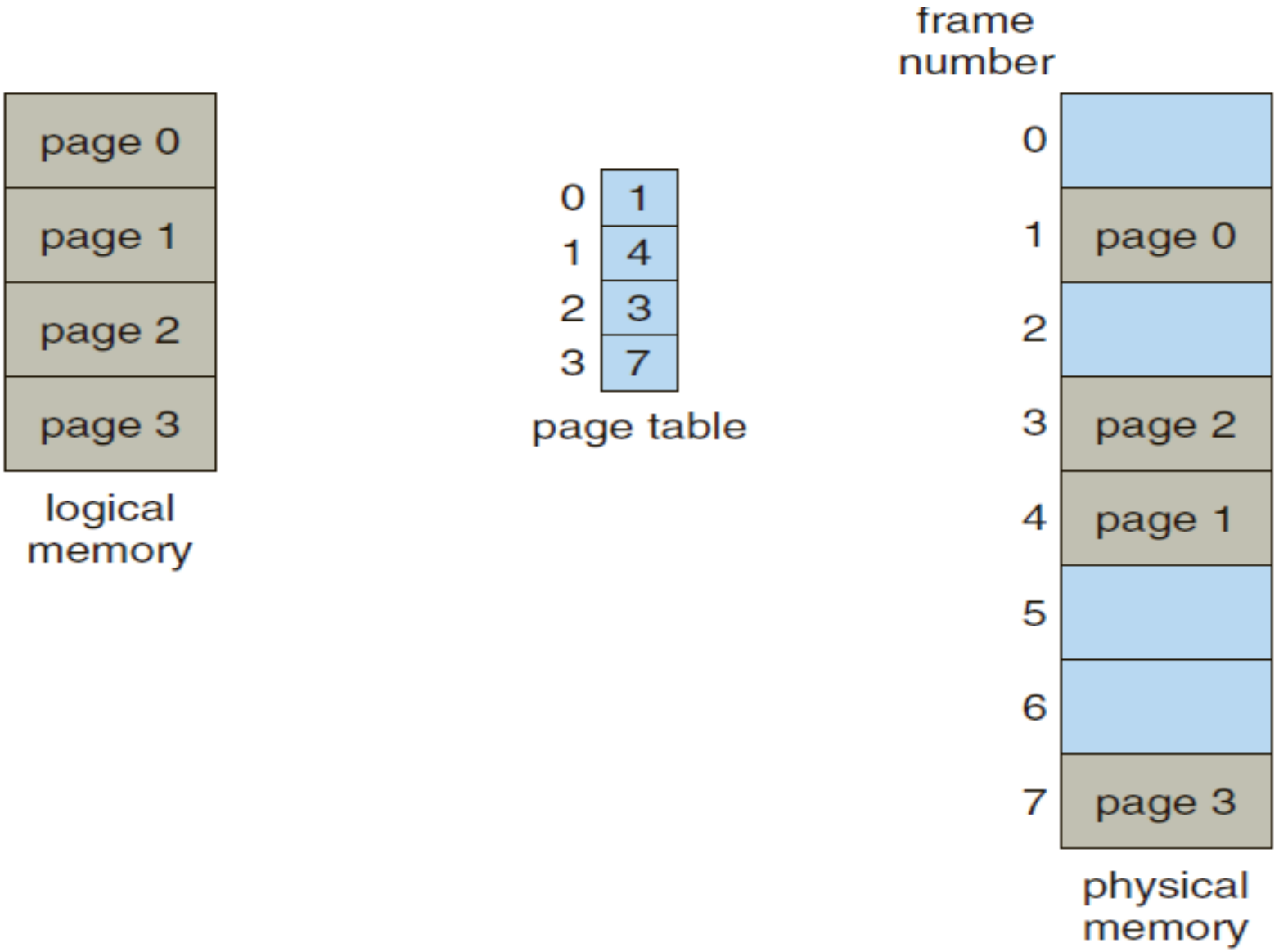
La memoria física se divide en *marcos* del mismo tamaño que las paginas.

Esta técnica evita la fragmentación externa y la interna se puede dar en la ultima pagina.

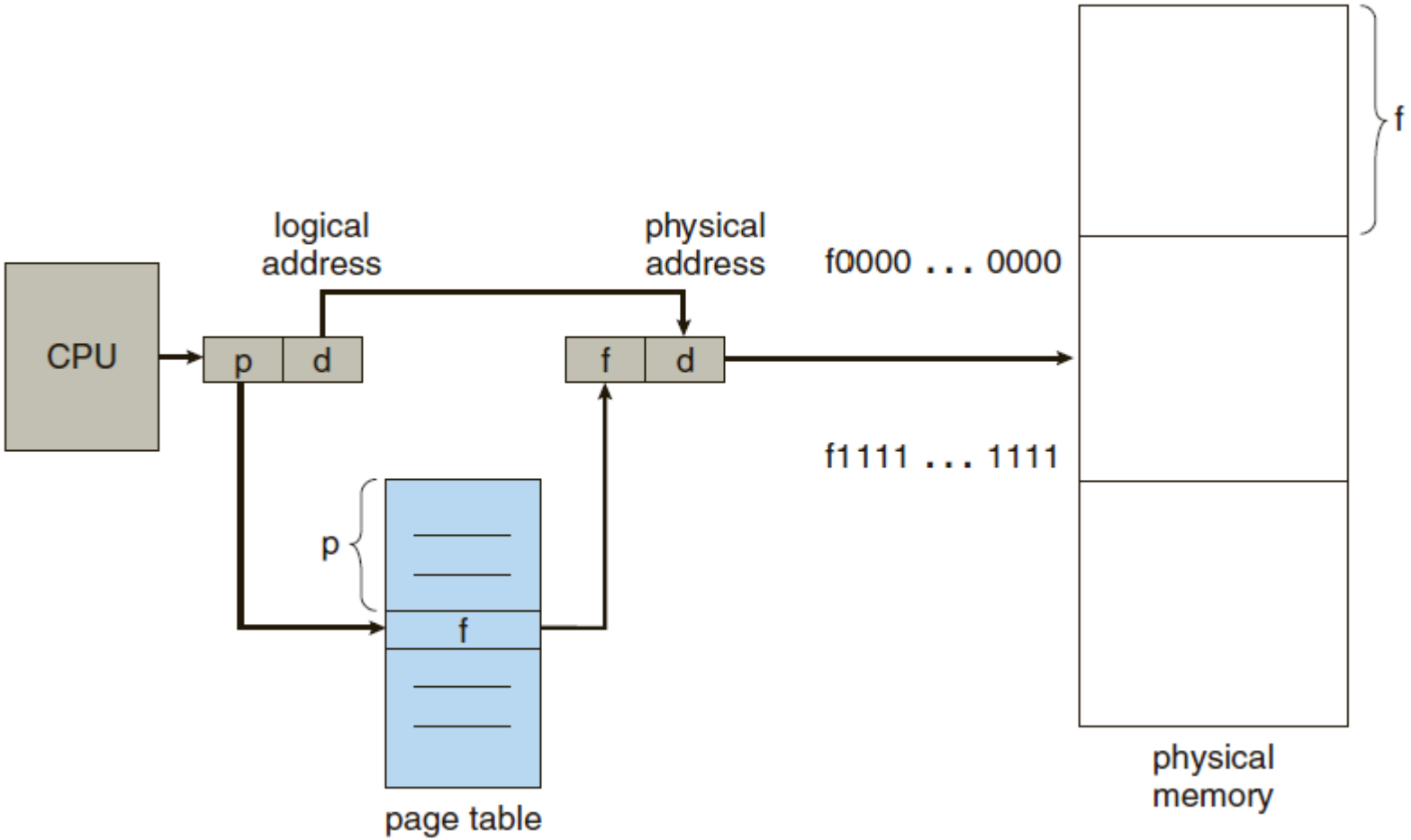
El tamaño de pagina puede ser 1k, 2k, 4K

Tabla de pagina

Memoria (Paginación Simple)



Memoria (Paginación Simple)

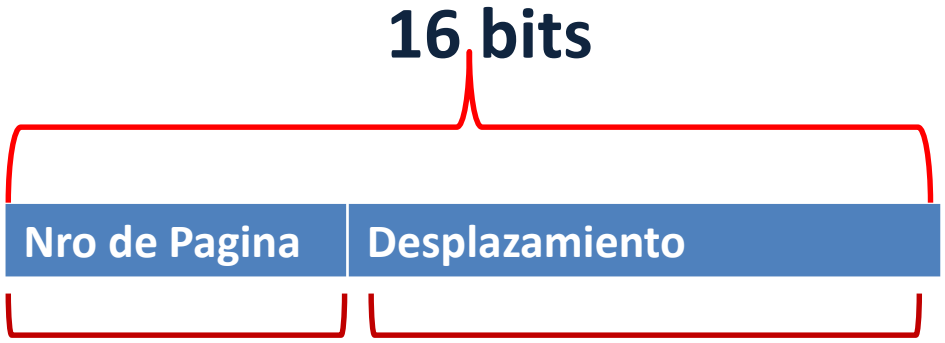


T.P. en MMU

Memoria (Paginación Simple)

La CPU emite una Dirección Lógica

Esta a través de la MMU se transforma en una Dirección Física.



Supongamos que la dirección lógica es de 16bits y que el tamaño de pagina es de 1K

Memoria (Paginación Simple)

La CPU en realidad emite un conjunto de bits, al cual se le aplica una máscara para determinar **(p,d)**

En un sistema de **16bits**, si tengo el tamaño de la página es de 4K, y la CPU emite una dirección lógica como la siguiente $30F4)_{16}$ en binario seria 0011000011110100.

Determinar la **D.F.** a la que pertenece dicha **D.L.** Determinar:

- **Cantidad de bits que usamos para el offset**
- **Cantidad para el tamaño de página.**

$4K = 2^2 \times 2^{10} = 2^{12}$ por lo tanto el numero 0011000011110100

Pagina = 0011 - desplazamiento = 000011110100

3

$F4)_{16} = 244)_{10}$

Pertenece a (p,d) (3,244)

Memoria (Paginación Simple)

Ejercicio de Practica

Dada la siguiente tabla de páginas de un proceso, donde el tamaño de las mismas es de 1Kbyte

Nº Página	Dirección de Inicio
P0	4096
P1	8192
P2	2048
P3	6144
P4	1.024

Hacer un scripts que calcule la dirección física de las siguientes direcciones lógicas:

página 0 desplazamiento 80

página 3 desplazamiento 150

página 1 desplazamiento 600

Memoria (Paginación HW)

- **Esto funciono muy bien hasta la década del ‘80, perooo**
- **Donde reside la Tabla de Pagina ?**
- **Que fue pasando con los P. con el paso del tiempo ?**
- **Los tamaños de las memorias eran de 64, 128, 256KB**

Cada vez que un proceso esta en ejecucion se debe cargar la T.P. en los registros de la MMU, originalmente la tabla reside en la RAM.

Los fabricantes de HW cada 2 años casi duplicaban la memoria.

Para 1990 mas o menos había computadoras de 4MB x lo tanto la cantidad de paginas aumento

Memoria (Paginación)

- **Esto funciono muy bien hasta la década del ‘80, perooo**
- **Donde reside la Tabla de Pagina ?**
- **Que fue pasando con los P. con el paso del tiempo ?**
- **Los tamaños de las memorias eran de 64, 128, 256KB**

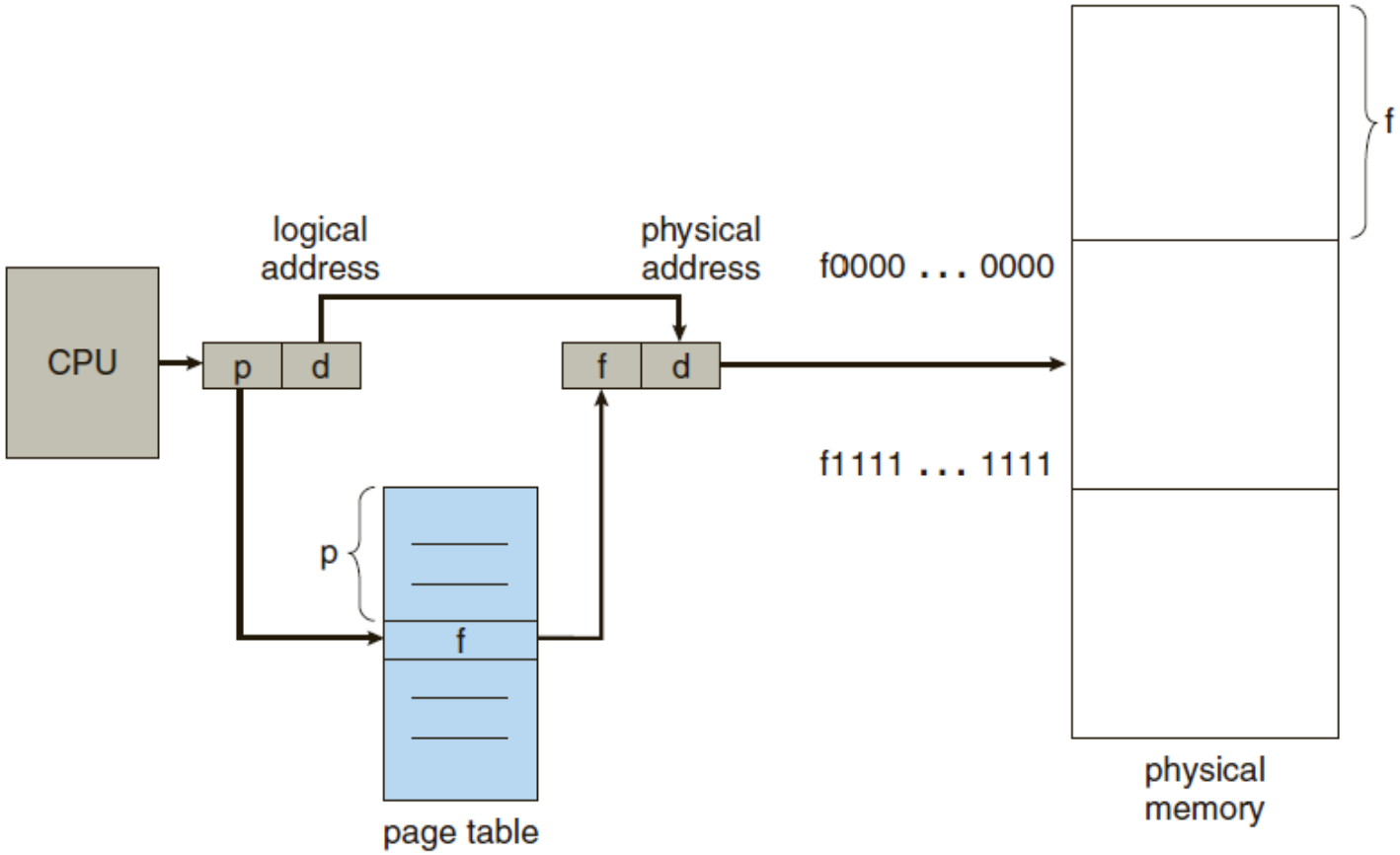
Cada vez que un proceso esta en ejecucion se debe cargar la T.P. en los registros de la MMU, originalmente la tabla reside en la RAM.

Los fabricantes de HW cada 2 años casi duplicaban la memoria.

Para 1990 mas o menos había computadoras de 4MB x lo tanto la cantidad de paginas aumento

Memoria (Paginación HW)

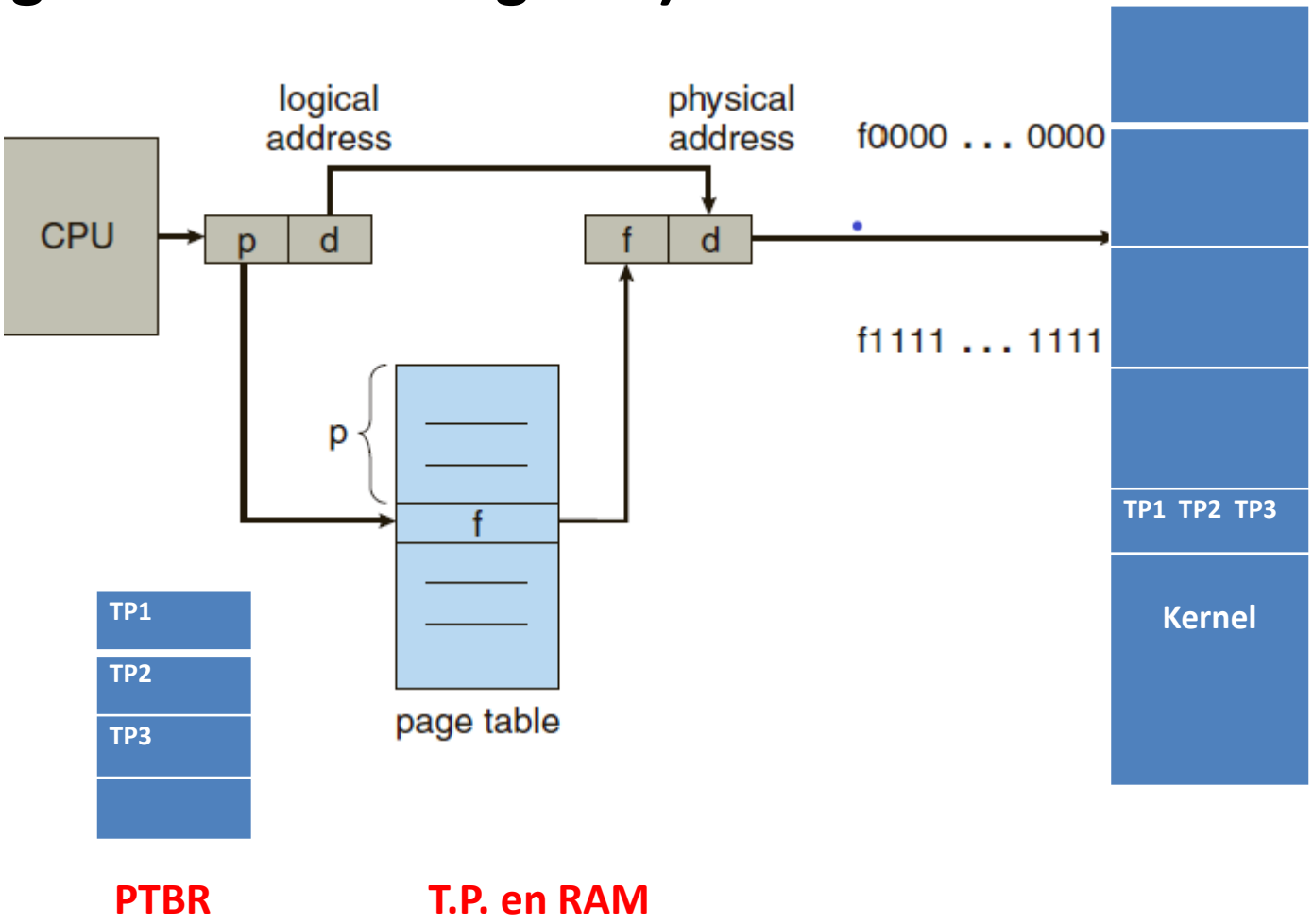
Ahora sobra RAM por lo tanto se decide colocar la T.P. en RAM



T.P. en RAM

Memoria (Paginación HW)

Ahora los registros de la MMU para la T.P. los voy a usar para otra cosa (Page Table Base Register)



Memoria (Paginación HW)

El compilador deja la T.P. en RAM, luego el cargador la coloca en la MMU

Algunas MMU llegaron a colocar registros para mas de un proceso.

La longitud de los registros era del tamaño de la palabra de datos de la CPU

Reg 16 bit

Despl 512

Reg 16 bit

Despl 1K

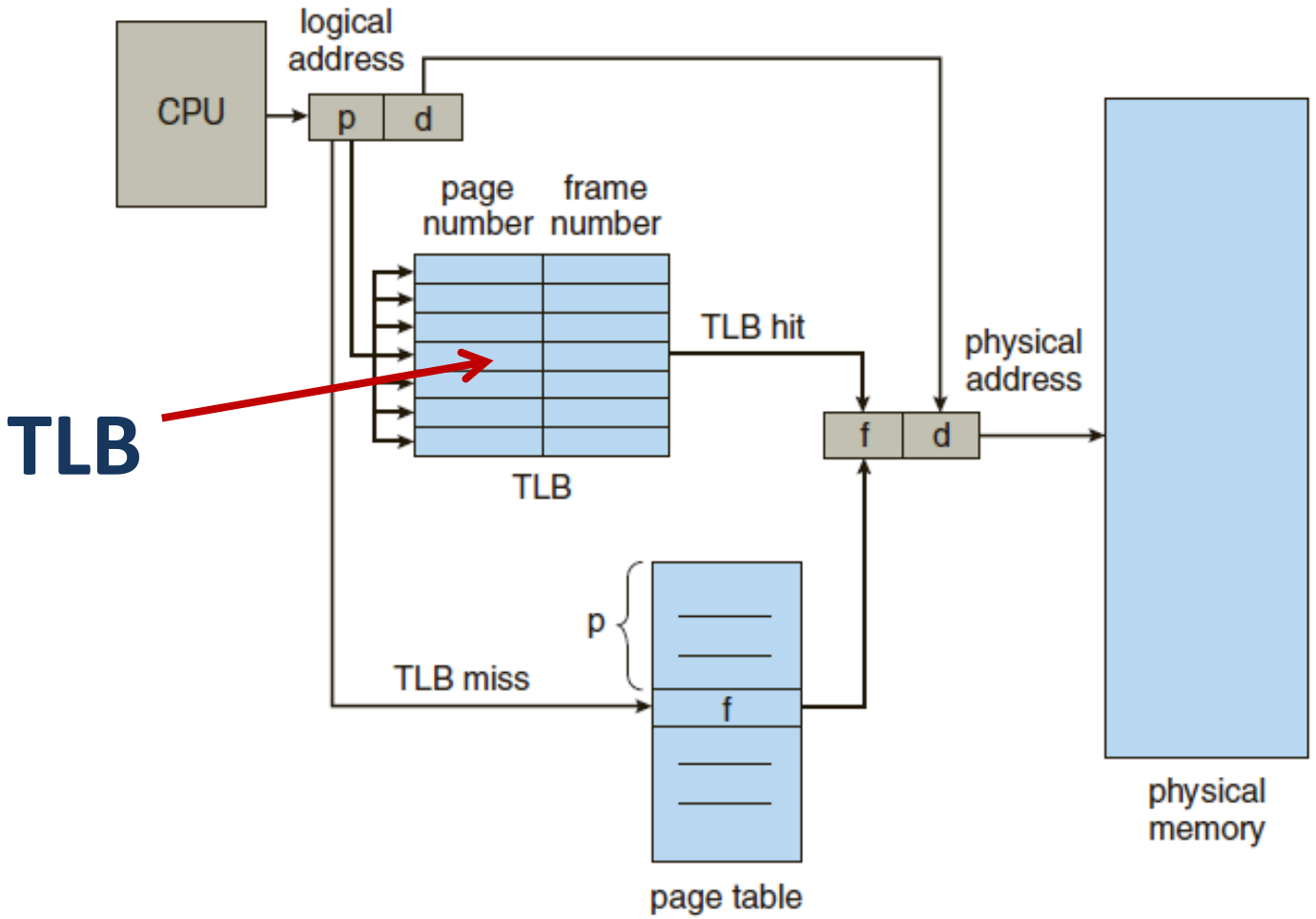
Reg 32 bit

Despl 4K

Muchos Registros en la **MMU**

Memoria (Paginación HW)

La solución fue usar una cache de HW especial, se la conoce como registros Asociativos o Buffers de Traducción de vista lateral.



Memoria (Paginación HW)

Fines del siglo anterior y principio de este siglo era común tener equipos con 64/128/256/512 Mb de RAM.

Los procesos empezaron a crecer en tamaño.

La longitud de los registros era del tamaño de la palabra de datos de la CPU

Reg 32 bit 

Despl 4K

El espacio en la RAM para las T.P. de los procesos era fijo.

Por lo tanto al ir terminando procesos quedaban huecos.

Los procesos pueden tener 250000 paginas o mas

Como resolvemos la fragmentación externa??

Se resuelve haciendo paginación, aquí nace lo que se llama

Paginación Jerárquica

Memoria (Paginación Jerárquica)

Ahora la dirección de la pagina a buscar en la RAM va a estar constituida por:

- **Nro pagina Externa**
- **Nro pagina Interna**
- **Desplazamiento**

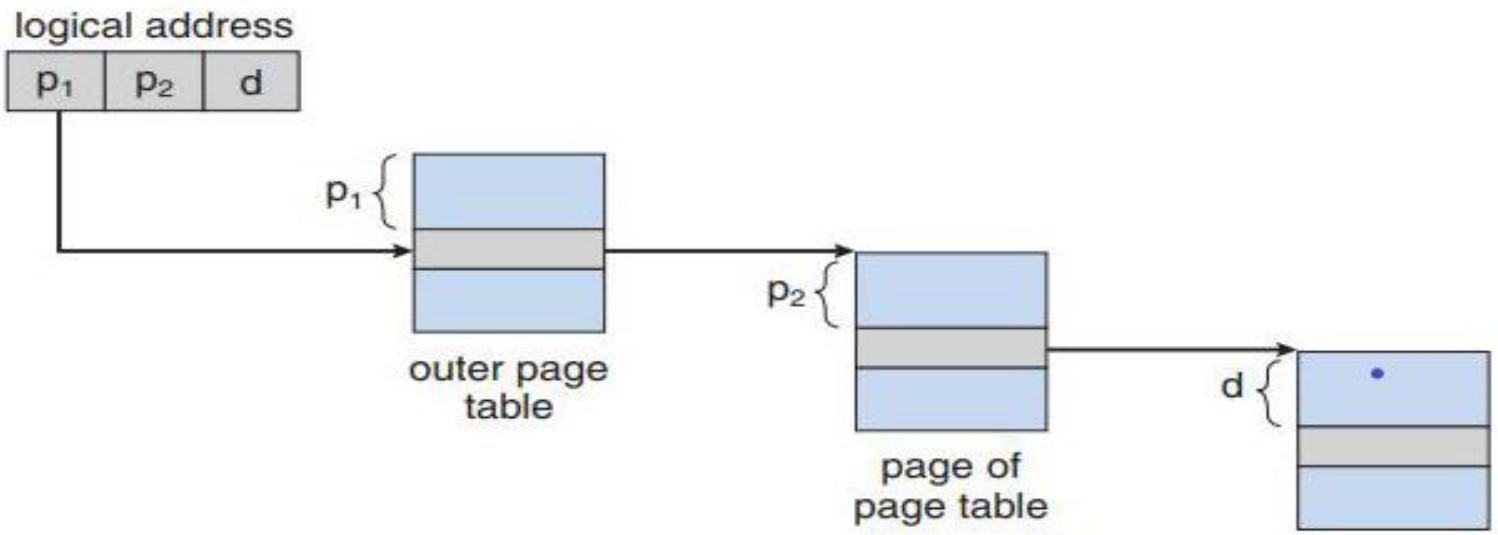
Los procesos empezaron a crecer en tamaño.

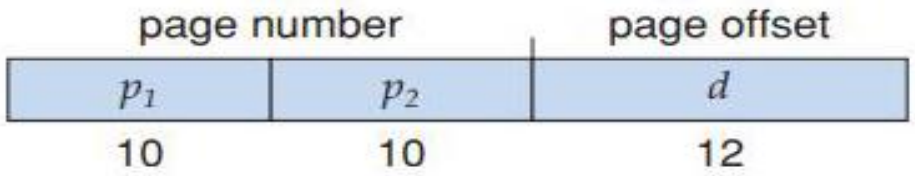
Para evitar la fragmentación dentro del espacio para las T.P.

Se paginaron las tablas de pagina.

Memoria (Paginación Jerárquica)

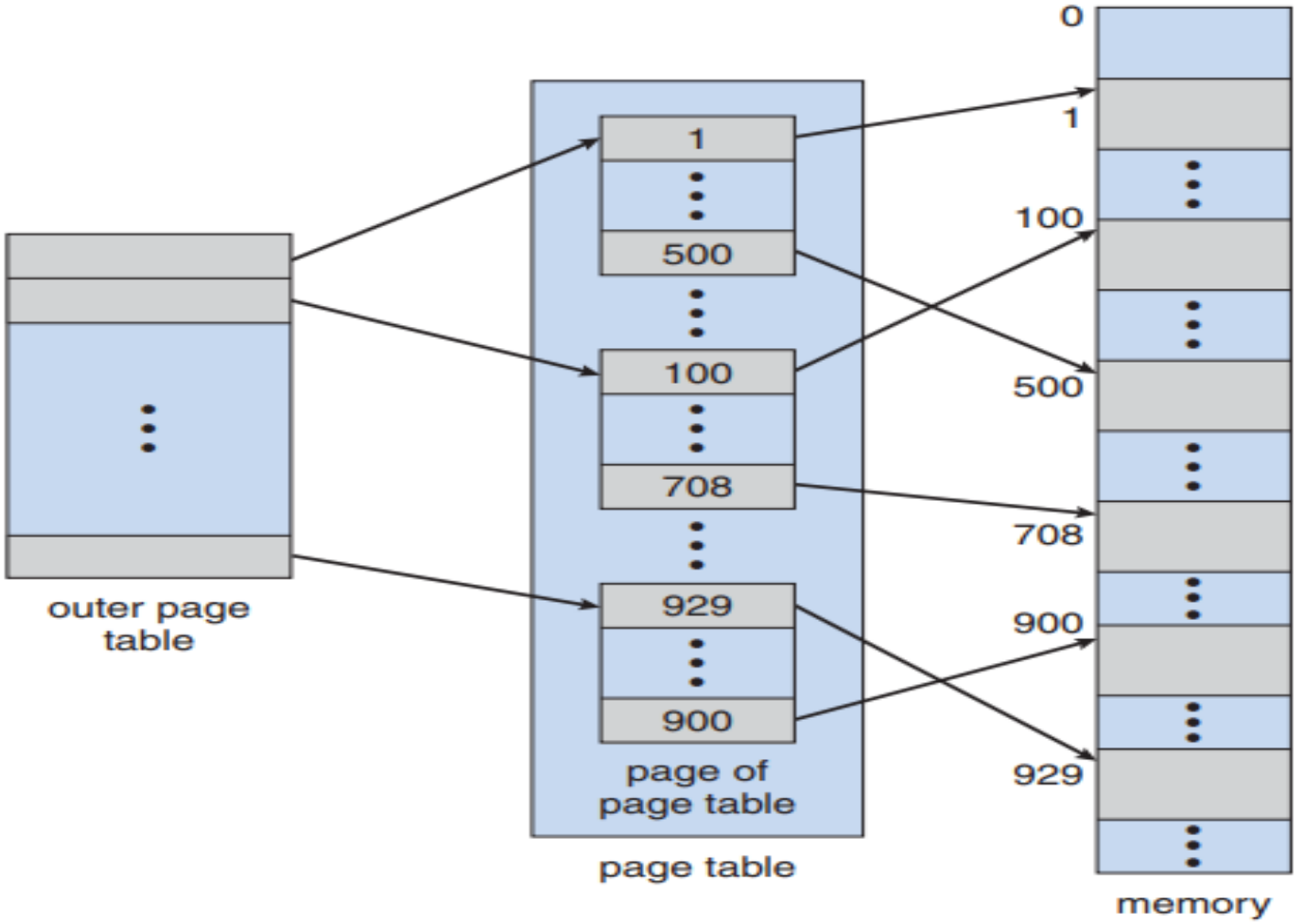
Ahora la dirección será de la siguiente manera:





Memoria (Paginación Jerárquica)

Se vería de la siguiente manera:



Memoria (Paginación Jerárquica)

- **Como sabemos, mas o menos a partir del año 2005 en adelante, nos empiezan a dar Gigas de memoria RAM**
- **En los 4GB se nos termino el direcc. de 32bits**
- **Los procesadores de 64bits cambian drásticamente la cantidad de memoria a direccionar 2^{64}**

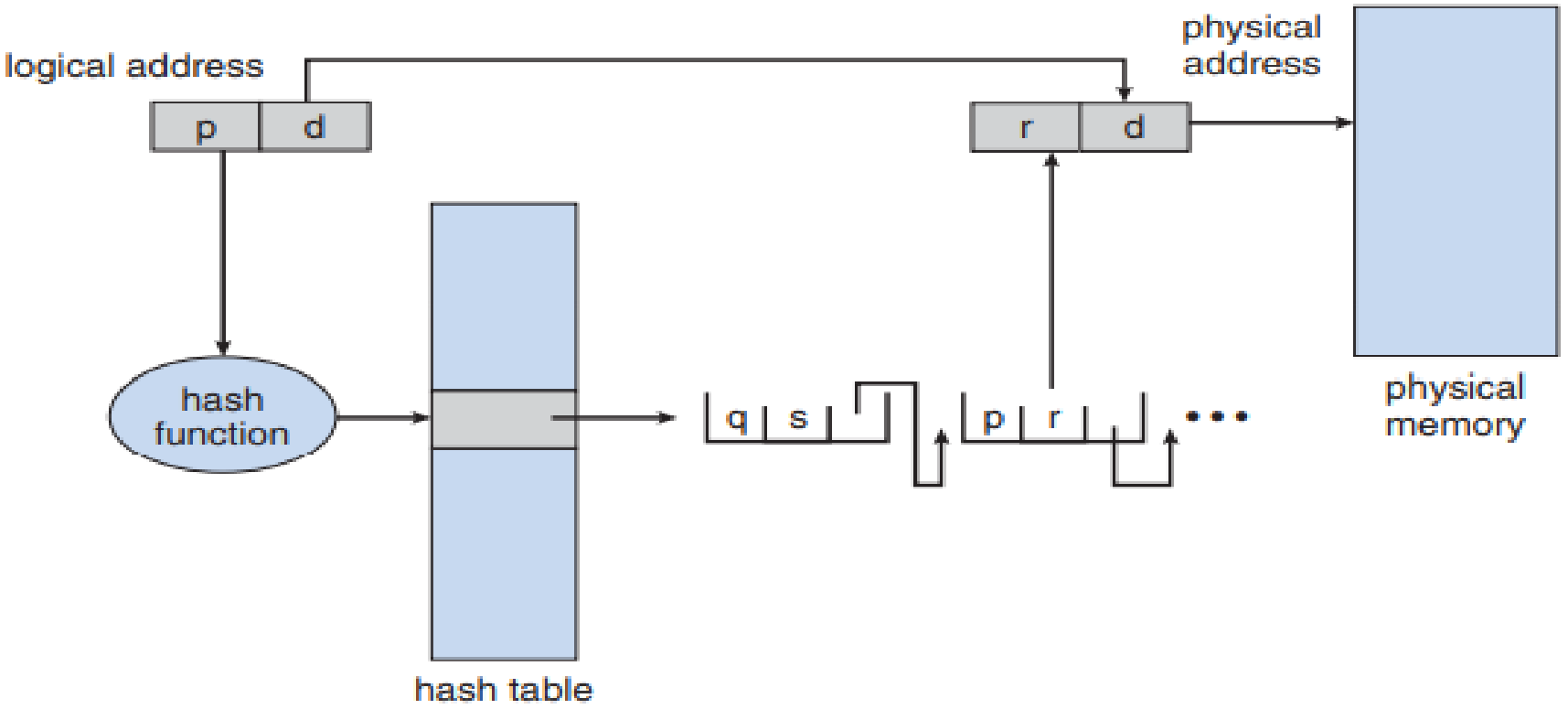
Una solución seria seguir dividiendo

64 bits

Esto nos llevaría a tener varios acceso a la RAM

Para resolver esto se uso la Tecnica de Hash

Memoria (Técnica Hash)



Para cada elemento de la tabla hash, hay tres campos:

- Número de página virtual (que es el valor hash).
- Valor del marco de página asignado.
- Un puntero al siguiente elemento de la lista vinculada.

Fin del Tema