UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Departament d'Arquitectura de Computadors

Estructura de Computadores
Tema 7: Memoria Virtual

Agustín Fernández

Departament d'Arquitectura de Computadors

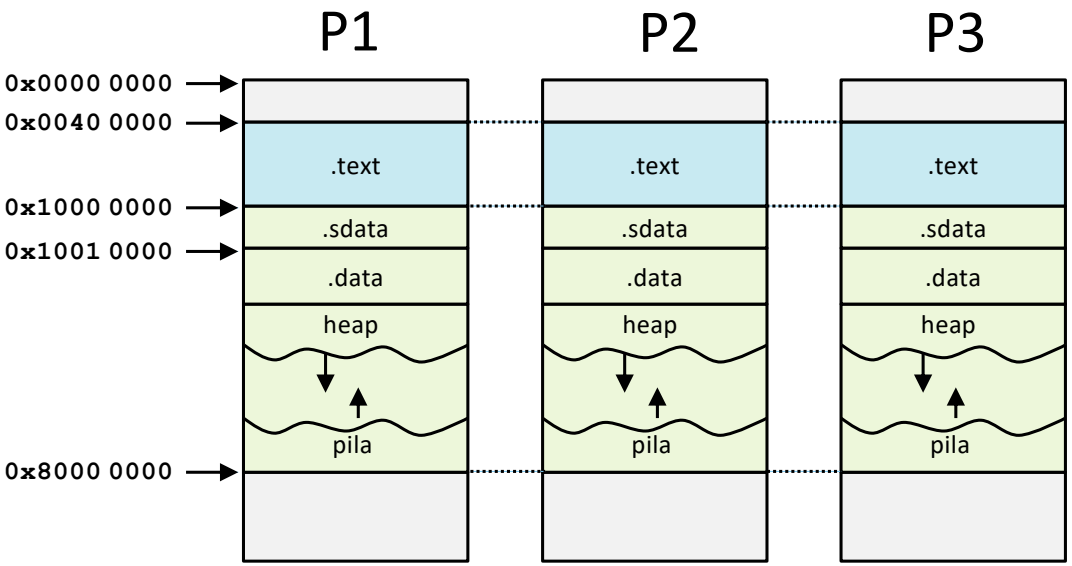
Facultat d'Informàtica de Barcelona

Universitat Politècnica de Catalunya

Introducción

- Sistemas multiusuario/multiproceso con varios programas ejecutándose concurrentemente
  - Tamaño memoria necesario >> memoria principal
  - Sólo una pequeña porción de la memoria se está utilizando activamente en un instante determinado.
  - Necesitamos mecanismos de protección
- Los programas siempre tienen las mismas direcciones lógicas:
  - Reubicación
  - Traducción de direcciones
- Hace unos años, tamaño de un programa > memoria física
  - Overlays
  - Memoria Virtual

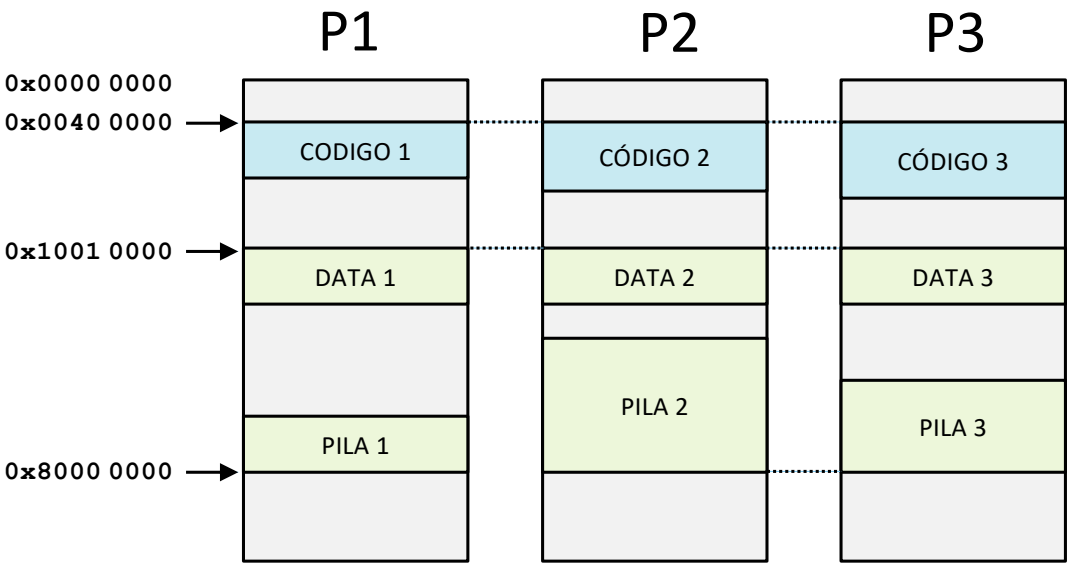
Necesidad de Reubicación



En el proceso de Compilación y enlazado se asignan direcciones absolutas a instrucciones y datos.

Se asignan las MISMAS DIRECCIONES a TODOS LOS PROGRAMAS.

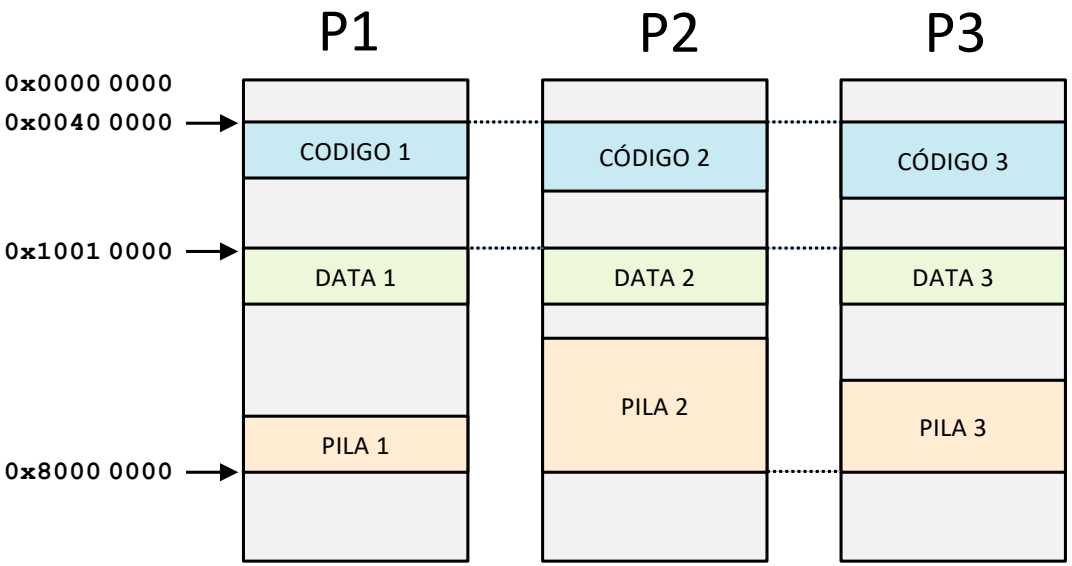
Necesidad de Reubicación



En el proceso de Compilación y enlazado se asignan direcciones absolutas a instrucciones y datos.

Se asignan las MISMAS DIRECCIONES a TODOS LOS PROGRAMAS.

Necesidad de Reubicación



Memoria Principal

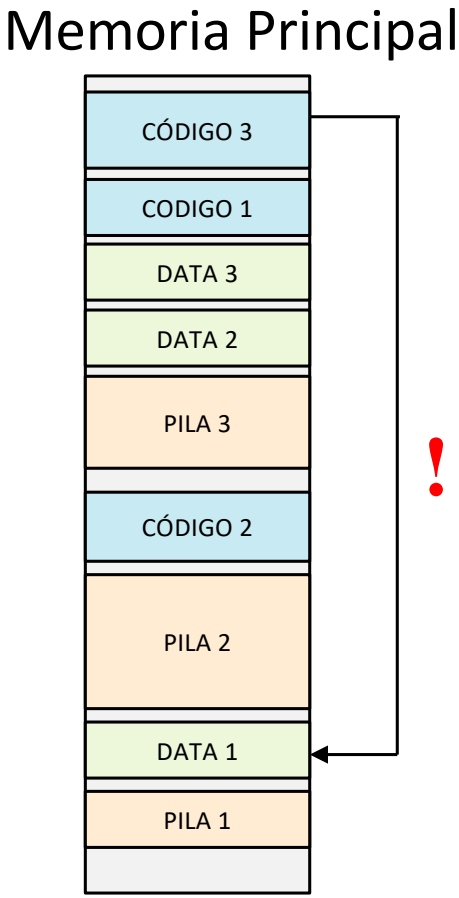
Para ejecutarlos, es necesario REUBICARLOS.
Hay que modificar todas las direcciones.

Necesidad de Protección

Un programa podría acceder a los datos de otro:

- Podría examinar toda la memoria ...
- ... y buscar passwords!
- ... o modificar datos!

Necesitamos mecanismos de PROTECCIÓN



Necesidad de utilizar más memoria de la disponible

- La capacidad de la Memoria Principal, depende de lo que instalemos:
  - 4 GB, 8GB, 16GB, ..., 128GB
- ¿Qué pasa si el tamaño de nuestro programa excede la capacidad de la memoria instalada?
  - Por suerte, para ejecutar un programa NO es necesario que esté completamente cargado en memoria. En un periodo de tiempo determinado, sólo utilizaremos una parte del programa (tanto datos, como código)
- En la prehistoria de la informática se utilizaban **overlays**
  - El programa se dividía en módulos
  - En ejecución, sólo se cargaban en memoria los módulos indispensables.
  - La gestión de la carga de los módulos en memoria la hacía el propio programa
    - ✓ ⇒ Solución muy farragosa y compleja

Traducción de Direcciones

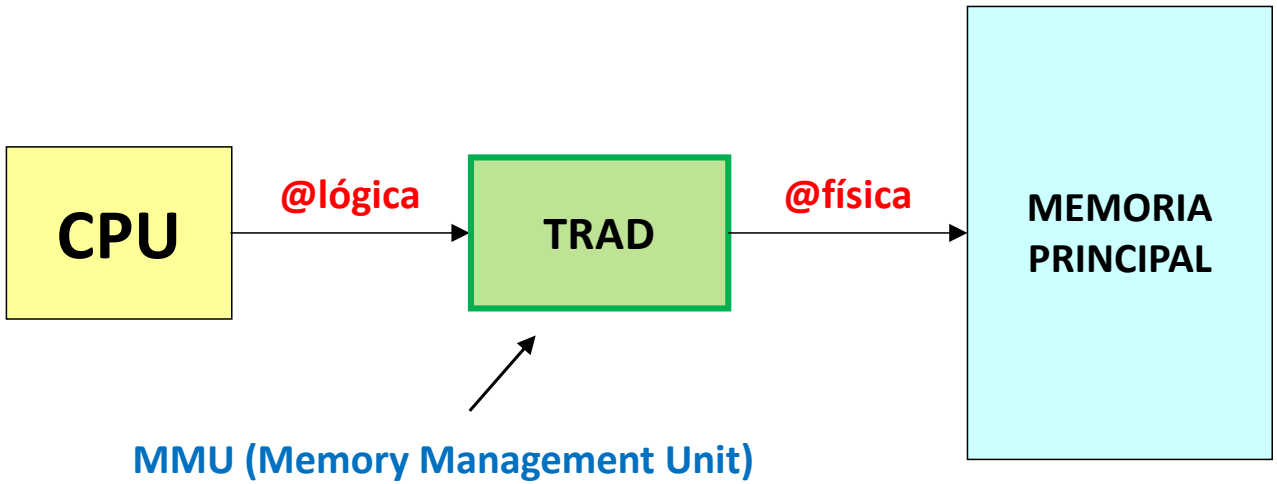
Idea Básica

- Diferenciar **ESPACIO LÓGICO** (dirección generada por el procesador) de **ESPACIO FÍSICO** (dirección con la que se accede a Memoria Principal).
- En general son diferentes

  ⇒ **Mecanismo de TRADUCCIÓN de DIRECCIONES**

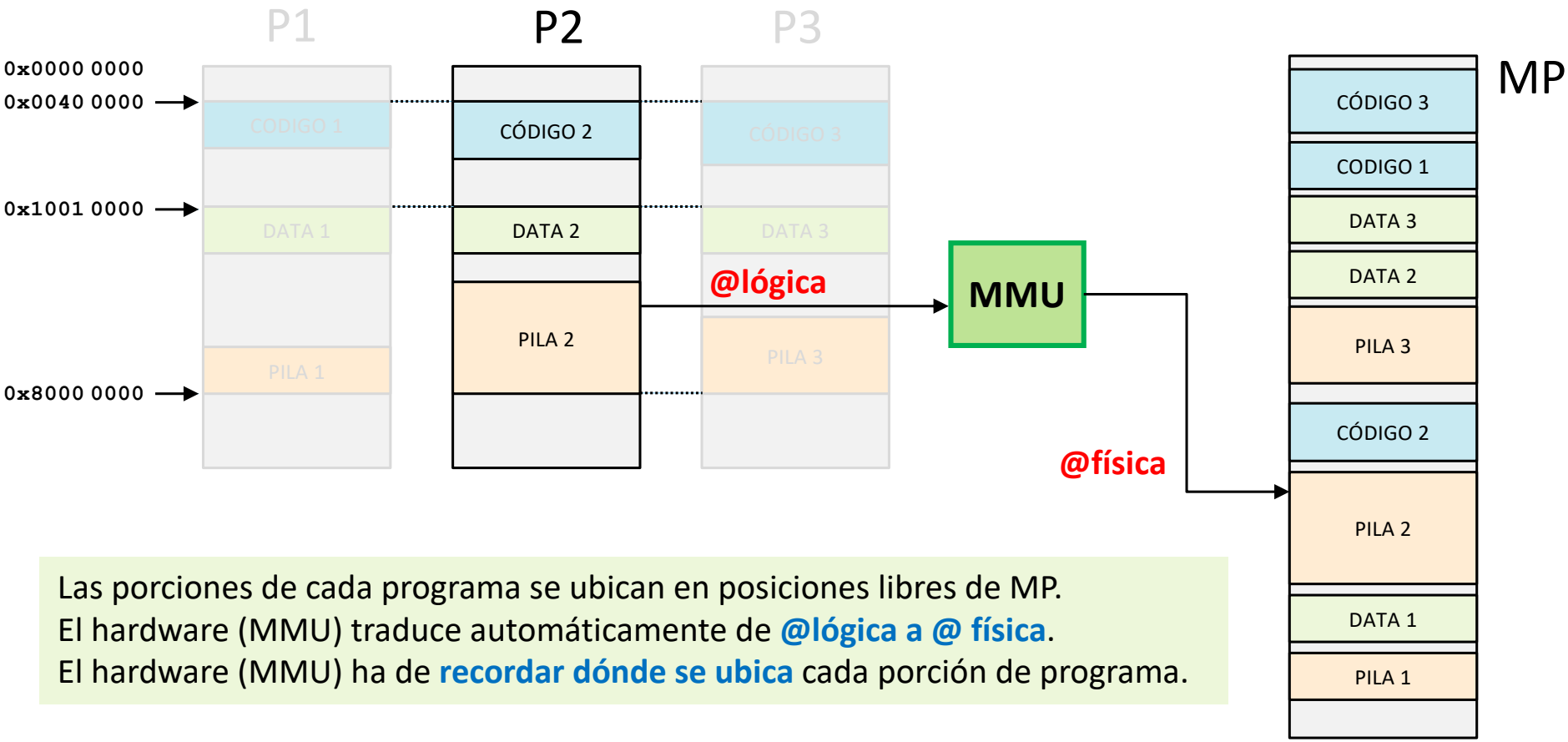
	Espacio lógico	Espacio físico
PDP 11/70	64 KB	256 KB
VAX-11	4 GB	32 MB

Ejemplo real (¡muy antiguo!)



Traducción de Direcciones

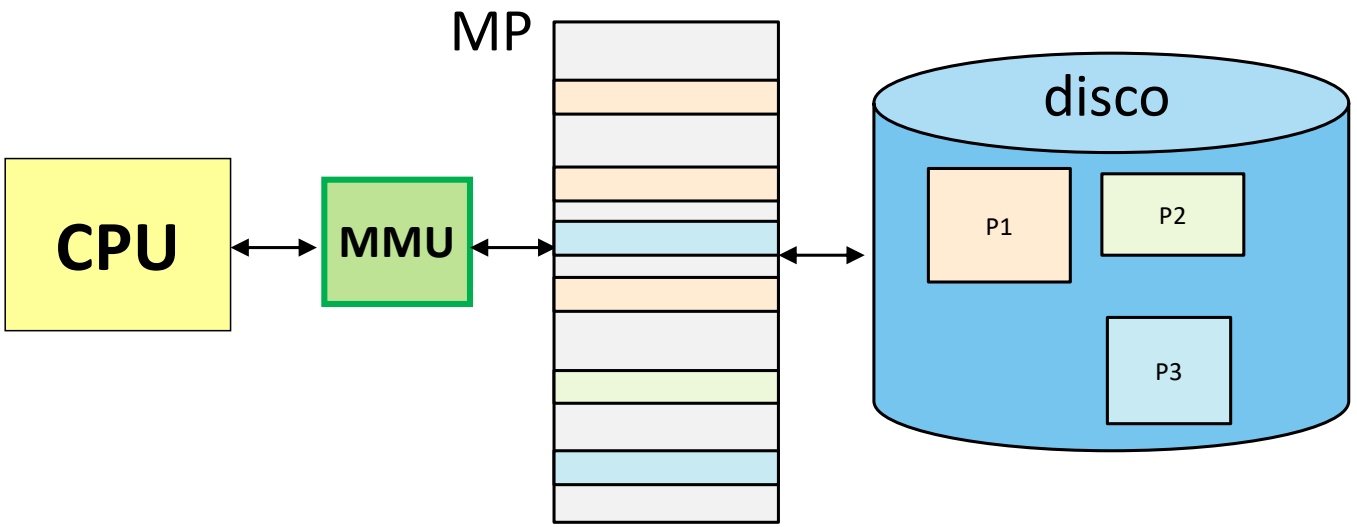
Espacio de Direccionamiento Lógico vs Físico.



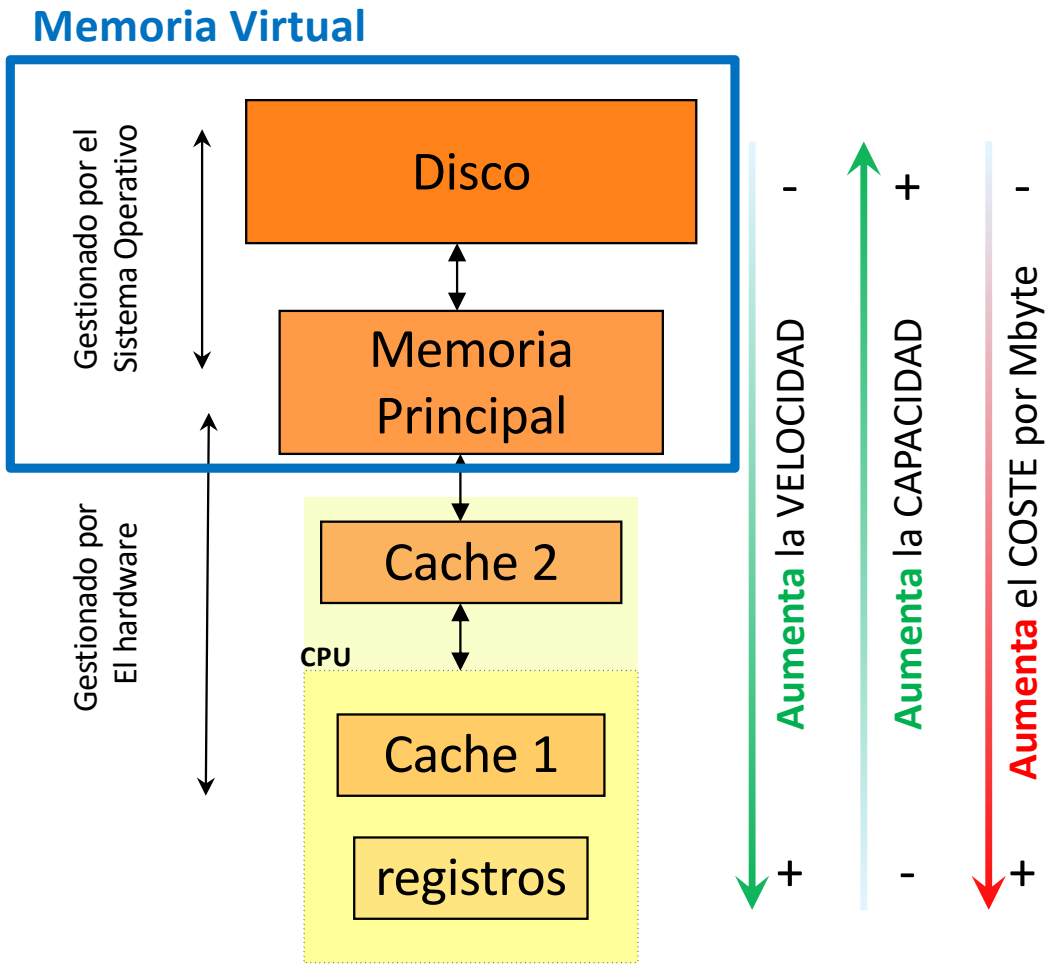
Las porciones de cada programa se ubican en posiciones libres de MP.
El hardware (MMU) traduce automáticamente de **@lógica a @ física**.
El hardware (MMU) ha de **recordar dónde se ubica** cada porción de programa.

Programas parcialmente cargados en Memoria Principal

- Los programas residen integramente en el disco.
- Las porciones de programa accedidas recientemente se guardan en MP.
- La Memoria Virtual se encarga de mover datos entre disco y MP, explotando la localidad.
- Similar a MC -- MP, pero ahora la gestión la realiza el Sistema Operativo.



Jerarquía de Memorias: Memoria Virtual



Objetivo: que cuando el procesador acceda a un dato, éste se encuentre en los niveles de la jerarquía más cercanos al procesador.

Si se consigue, obtendríamos una memoria con la velocidad de los niveles rápidos, el tamaño de los niveles más grandes y todo ello con un coste razonable.

¡La jerarquía funciona por las propiedades (localidad) de los programas!

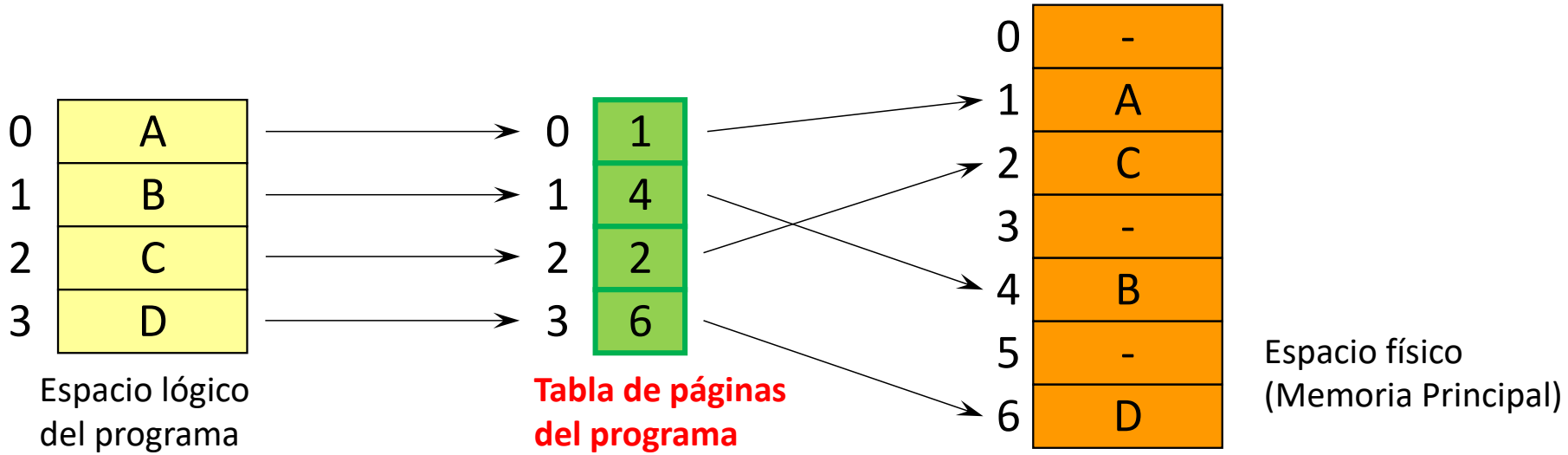
Memoria Virtual Paginada

- El espacio lógico se divide en bloques de tamaño fijo ⇒ **PÁGINAS**
- Un tamaño razonable de página sería entre 4 y 32 KB
- El espacio físico (MP) se divide en **marcos** de tamaño una página (**frames**, tramas).
- Los programas se trocean en páginas (están en disco)
- Una página puede colocarse en **CUALQUIER** marco de página de MP (correspondencia completamente asociativa)
- Las páginas se copian desde disco a MP **cuando son referenciadas**
- Hace falta una estructura de datos para saber qué hay en cada marco de página
  ⇒ **TABLA DE PÁGINAS**

Memoria Virtual Paginada

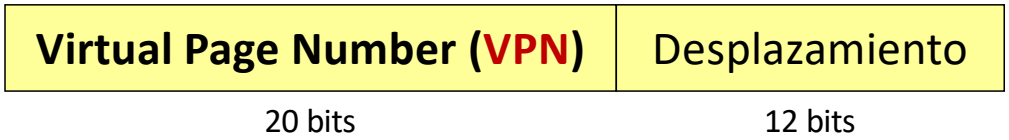
- Hace falta una estructura de datos para saber qué hay en cada marco de página

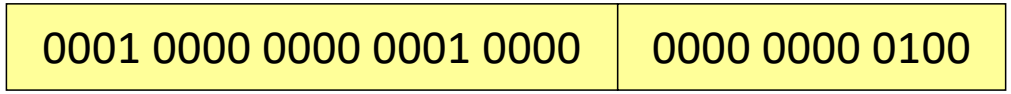
  ⇒ **TABLA DE PÁGINAS**



Subdivisión de la dirección

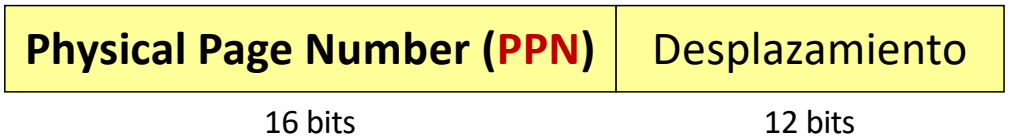
- Dada una dirección lógica A, ¿a que página (**VPN**, Virtual Page Number) pertenece?
  - Por ejemplo, con direcciones de 32 bits, A=0x10010004 y el tamaño de página es 4 KB (2^{12} bytes)

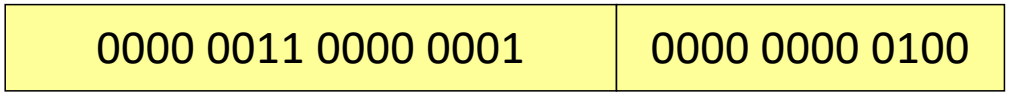


  - A=0x10010004 = 

, VPN = 0x10010, off = 0x004

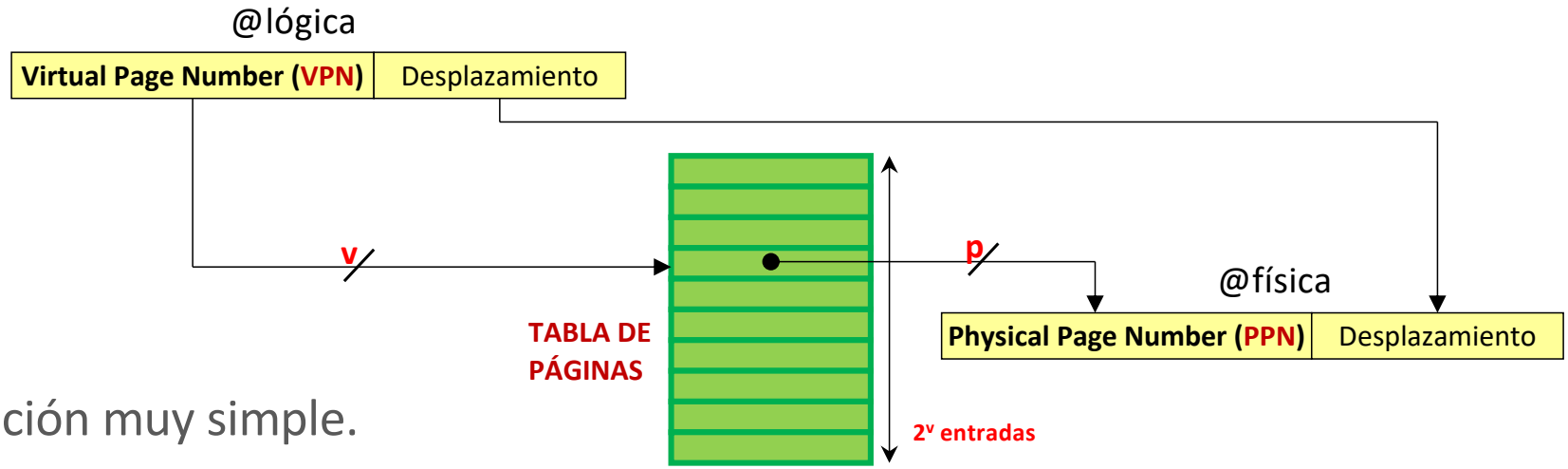
- Dada una dirección física AF, ¿a que frame (**PPN**, Physical Page Number) pertenece?
  - Por ejemplo, con direcciones de 28 bits, AF=0x0301004 y el tamaño de página es 4 KB (2^{12} bytes)



  - AF=0x0301004 = 

, PPN = 0x0301, off = 0x004

Paginación: implementación hardware de la traducción



- Reubicación muy simple.
- Permite protección de páginas.
- Cálculo rápido de la dirección (no hay operaciones aritméticas).
- Fragmentación.
- Página físicas y virtuales tienen el mismo tamaño.
- VPN y PPN pueden tener longitud diferente.
- En la mayoría de sistemas se cumple: $2^v > 2^p$.

Implementación de la Tabla de Páginas

- Cada proceso tiene su propia Tabla de Páginas.
- El contenido de la TP lo gestiona el SO.
- La tabla de páginas se indexa con el VPN
- Cada entrada de la tabla de páginas (PTE) contiene:
  - **Bit de Presencia P**. Vale 1 si la entrada es válida y la página está en MP.
  - **PPN**. Si P=1, indica el frame dónde está la página en MP
  - **Bit de Modificación D**. Si D=1, significa que la página ha sido modificada (por un store).
  - Otros bits con permisos diversos: READ_ONLY, READ/WRITE, EXEC, bits para gestionar LRU
- Si la página no está en memoria, la tabla de páginas contiene su ubicación en disco

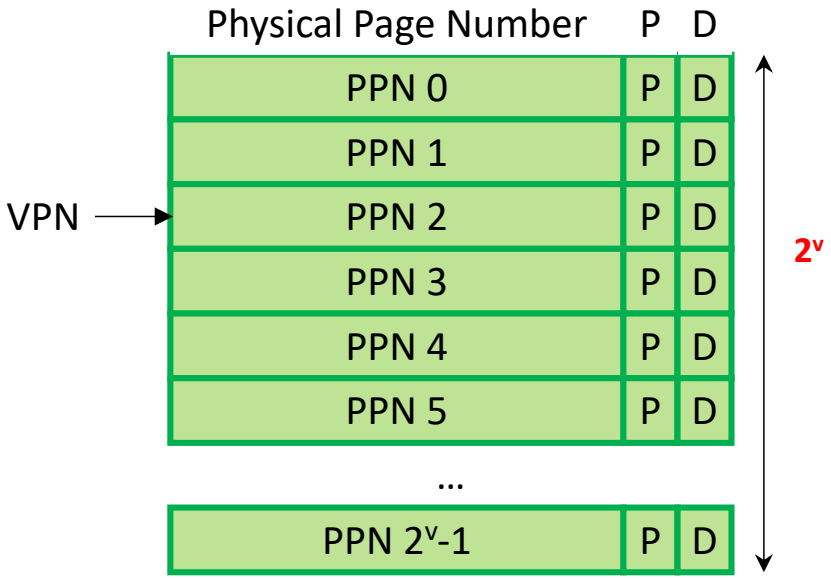


Tabla de páginas de un nivel

Paginación: ejemplo práctico

- En un sistema con direcciones virtuales de 64 bits y direcciones físicas de 43 bits, ¿Cuántos bits se necesitan para el VPN y el PPN si el tamaño de página es de 8 KB?
  - log2 (8·1024) = 13, se necesitan 13 bits para codificar el desplazamiento

@lógica 64 bits

VPN: 51 bits	Despl.: 13 bits

@física: 43 bits

PPN: 30 bits	Despl.: 13 bits

- ¿Qué tamaño tiene la tabla de páginas?
  - Necesitamos una entrada para cada VPN diferente y en cada entrada necesitamos 30 bits para codificar la PPN, y dos bits para P y D (mínimo imprescindible)

⇒Tamaño mínimo 2^{51}·(30+2) bits = 8 · 2^{50} bytes = 8 PB (¡Peta bytes!)

¡Un poco GRANDE!, ¿NO?

Implementación de la Tabla de Páginas

- En un procesador actual, la TP sería mucho más grande que la Memoria Principal.
- **Solución.** Tablas de Páginas de múltiples niveles (no las estudiaremos)
  - Sólo una parte de la tabla de páginas está en MP
  - Se requieren varios accesos a la tabla de páginas para conocer la @física de la página
- En este curso utilizaremos como modelo una Tabla de Páginas de un solo nivel almacenada siempre en MP.

- La Tabla de Páginas es accedida en cada referencia a memoria
- Si la Tabla de Páginas es de un nivel y se almacena en Memoria Principal
  - 1 acceso a MP necesita 1 acceso a la Tabla de Páginas (en MP) y 1 acceso al dato

    ⇒ **MUY LENTO**

Implementación de la Tabla de Páginas

- La Tabla de Páginas es accedida en cada referencia a memoria
- Si la Tabla de Páginas es de un nivel y se almacena en Memoria Principal
- 1 acceso a MP necesita 1 acceso a la Tabla de Páginas (en MP) y 1 acceso al dato

  ⇒ **MUY LENTO**

- **Solución**. Tener una memoria cache "especial" para la tabla de páginas
  - **Translation Lookaside Buffer (TLB)** – Buffer de traducción anticipada

¡ATENCIÓN!

En el TLB no hay datos (o programas), sólo información para acelerar la traducción de direcciones.

Idem en la tabla de páginas.

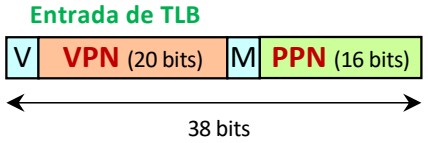
Traducción de direcciones con TLB

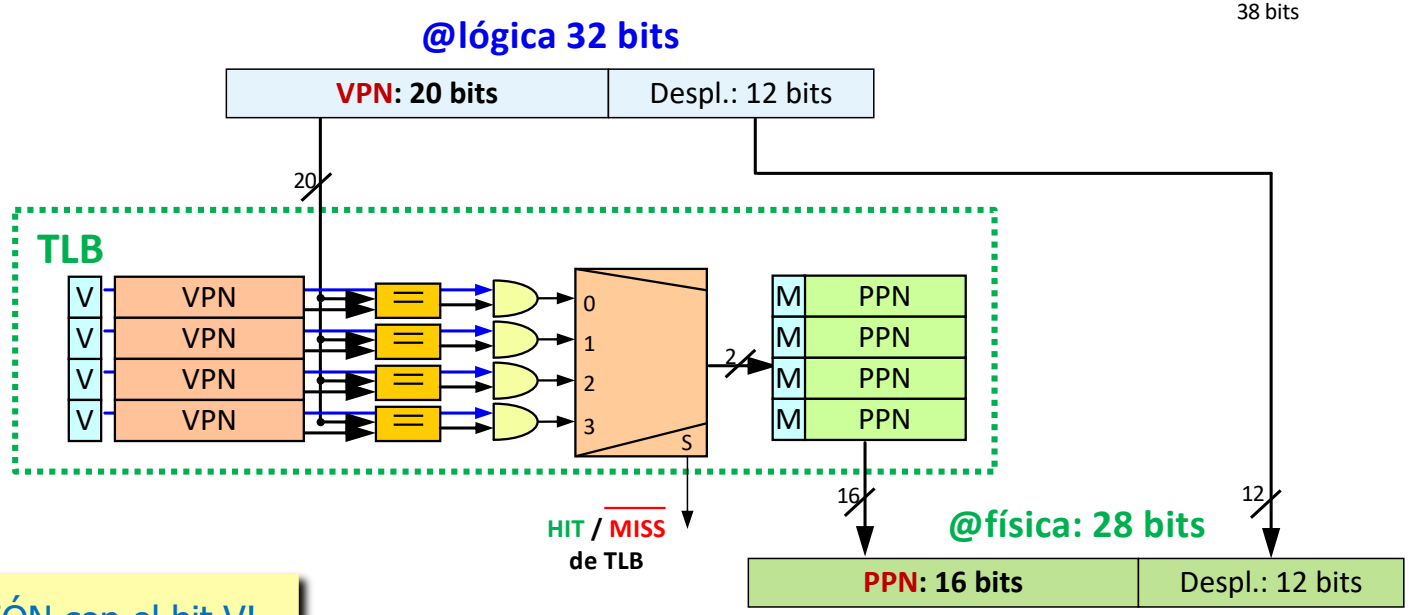
Translation Lookaside Buffer (TLB)

- Sirve para acelerar el proceso de traducción de direcciones
- Tiene una estructura similar (campos) a la Tabla de Páginas
- Sólo guarda algunas de las entradas de la TP
- Características principales:
  - Integrado en el mismo chip que el procesador, muy cercano a la MC
  - Pocas entradas (1 entrada por página): 32, 64, ...
  - Completamente asociativo, o asociatividad muy alta
  - Tasa de fallos muy baja: 0,01%-1%
  - Muy rápido, 0,5 – 1 ciclo en caso de acierto
  - Algoritmo de reemplazo (LRU, PseudoLRU)
  - AMD Opteron, tenía un TLB-I y otro TLB-D, completamente asociativos, 40 entradas.

Traducción de direcciones con TLB

- 32 bits de dirección lógica
- 28 bits de dirección física
- Páginas de 4KB (2^{12} bytes)
- TLB de 4 entradas
- En el TLB hay bit de validez en lugar de bit de presencia



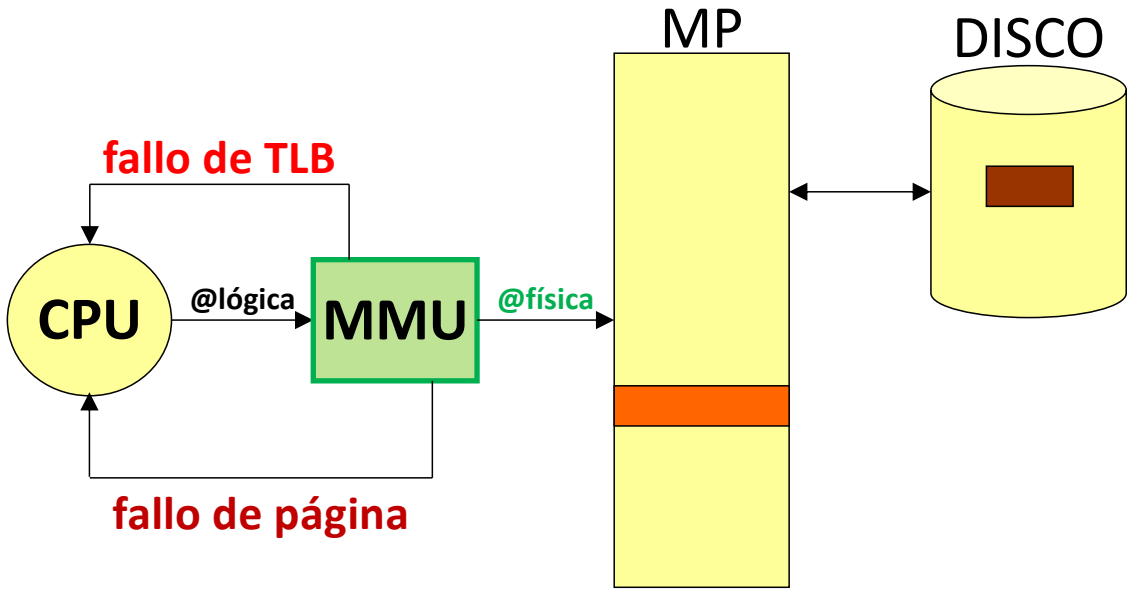


¡ATENCIÓN con el bit V!

Memoria virtual

La Memoria Virtual permite:

- Ejecutar un programa con espacio lógico > espacio físico
- Ejecutar un programa parcialmente cargado en Memoria
- Proteger el espacio de direcciones de los programas de ser accedido por otros programas



Paginación bajo demanda

La **TABLA de PÁGINAS** tiene un bit de presencia

if (**fallo de página**) {

Reemplazar página$^{(D)}$: **MP** → **DISCO**

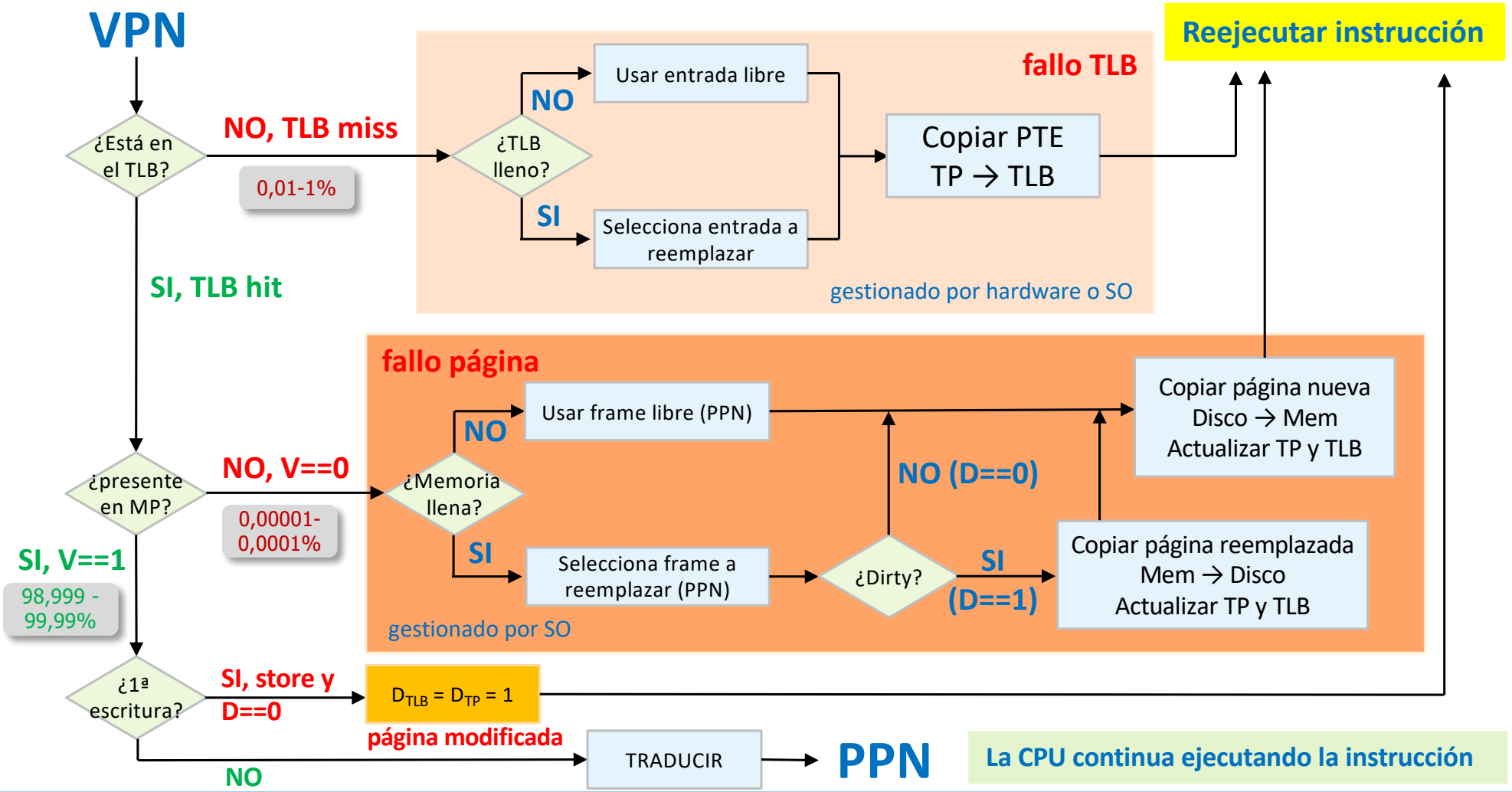
Cargar la página solicitada: **DISCO** → **MP**

}

Memoria virtual

- **¿Quién gestiona la memoria virtual?**
  - El Sistema Operativo (software). ¿Porqué no el hardware?
- **¿Cuándo se trae una página de Disco a MP?**
  - Bajo demanda en caso de fallo (hay otros modelos)
- **¿Dónde se ubica una página en MP?**
  - En cualquier marco (frame), política totalmente asociativa
- **¿Qué página de la MP se substituye en caso de fallo?**
  - Algoritmos de reemplazo muy sofisticados. La **tasa de fallos** es **MUY IMPORTANTE**.
    Un fallo puede costar millones de ciclos porque hay que acceder a disco.
    La decisión es software y hay mucho tiempo para tomarla.
  - Las páginas modificadas hay que escribirlas en disco.
  - Tasa de fallos: 0,00001% - 0,001%
- **¿Qué se hace con las escrituras?**
  - COPY BACK + WRITE ALLOCATE

Fallos de TLB y Fallos de Página



Protección y Memoria virtual

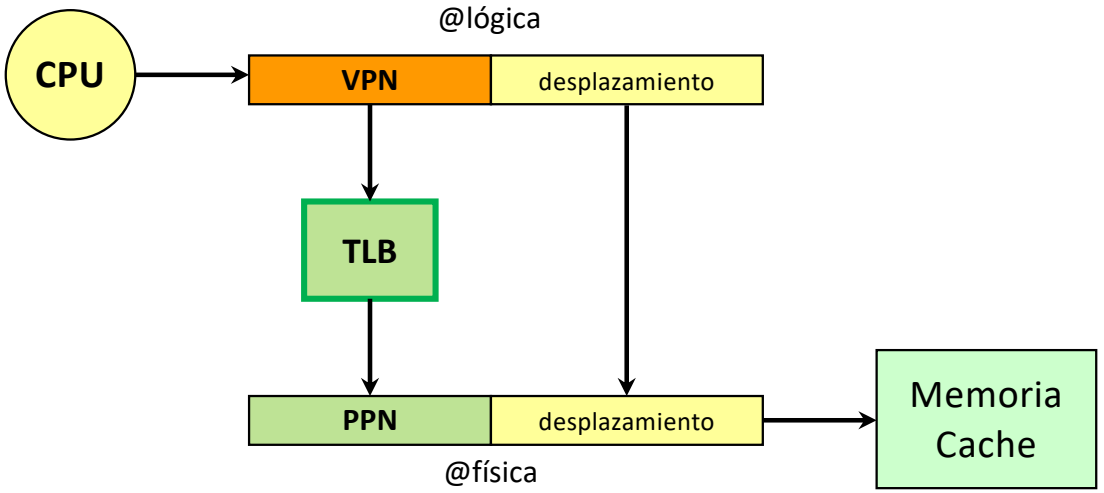
- Cada página física está asignada a un único proceso y no aparece en las tablas de páginas de otros procesos.
  - El mecanismo de traducción hace imposible que un proceso acceda a páginas de otro.
- Pero, ¿qué pasaría si un proceso intenta modificar su tabla de páginas?
  - La tabla de páginas se guardan en el espacio de direcciones reservado al SO.
  - En MIPS, son las direcciones con el bit 31 igual a 1.
- ¿Cómo accede el SO a ese espacio de direcciones?
  - El procesador tienen 2 modos de funcionamiento: usuario y sistema
  - Sólo cuando está funcionando en modo sistema tiene acceso al TLB y tabla de páginas.
- Protección contra escritura. Resulta conveniente prohibir la escritura en algunas páginas.
  - Por ejemplo, páginas de código, no se permite código automodificable
  - Se añade un bit de permiso de escritura (bit E) en la TP y TLB
  - Si un proceso intenta escribir en una página con el bit E=0, se produce una excepción y el SO aborta el programa.

Juntando Memoria Virtual y Memoria Cache

- La traducción de direcciones y la memoria cache son conceptos ortogonales:
  - La memoria cache permite acelerar los accesos a memoria
  - La traducción de direcciones permite soportar memoria virtual
    - ✓ **El TLB es sólo un mecanismo de aceleración del proceso de traducción**
- Un sistema puede tener sólo memoria cache, sólo traducción de direcciones, ambos mecanismos o ninguno de ellos
- Los actuales procesadores de propósito general cuentan con una jerarquía de uno o más niveles de cache y mecanismos de traducción de direcciones con el correspondiente TLB
- En este último caso, ¿cuándo se efectúa la traducción de direcciones lógicas a físicas, antes o después de acceder a la Memoria Cache?
- Tres posibilidades:
  - Traducción **antes** de acceder a Memoria Cache
  - Traducción **después** de acceder a Memoria Cache
  - Traducción y acceso a Memoria Cache **simultáneos**

Juntando Memoria Virtual y Memoria Cache

- Traducción **antes** de acceder a Memoria Cache



- Memoria Cache de direcciones físicas
- Lento: un acceso a memoria necesita un acceso TLB + acceso MC

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Departament d'Arquitectura de Computadors

Estructura de Computadores
Tema 7: Memoria Virtual

Agustín Fernández

Departament d'Arquitectura de Computadors

Facultat d'Informàtica de Barcelona

Universitat Politècnica de Catalunya