Memoria Virtual

Águstín Fernández, Josep Llosa, Fermín Sánchez

Estructura de Computadors II
Departament d'Arquitectura de Computadors
Facultat d'Informàtica de Barcelona

Índice

- Introducción
- Traducción de direcciones
  - Segmentación
  - Paginación
    - TLB
- Memoria Virtual
- Juntando memoria virtual y memoria cache

Memoria Virtual: Introducción

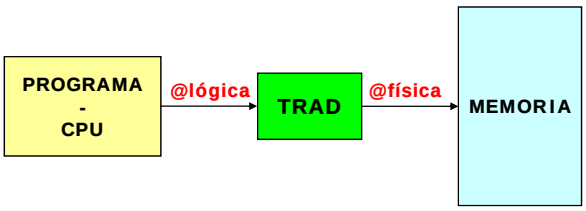
- Sistemas multiusuario o multiprogramado con varios programas ejecutándose concurrentemente
  - Tamaño memoria necesario >> memoria principal
  - Sólo una pequeña porción de la memoria se está utilizando activamente en un instante determinado.
- Los programas siempre tienen las mismas direcciones lógicas:
  - Reubicación
  - Traducción de direcciones
- Tamaño de un programa > memoria física
  - Overlays
  - Memoria Virtual

Memoria Virtual: Traducción de direcciones

- Idea Básica:
  - Diferenciar Espacio Lógico (dirección generada por el procesador) de Espacio Físico (dirección con la que se accede a memoria).
  - En general son diferentes

    → Mecanismo de Traducción de Direcciones

- Esquemas básicos de traducción:
  - Segmentación
  - Paginación



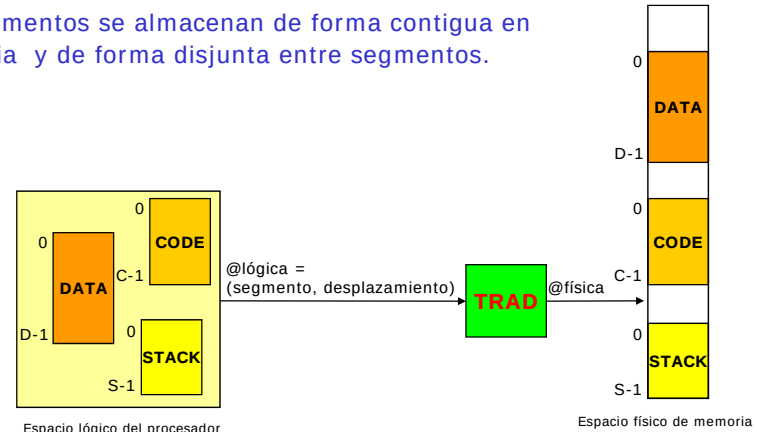
Memoria Virtual: Traducción de direcciones

- Diferencia entre Espacio Lógico y Espacio Físico.
- Ejemplos reales (¡antiguos!)

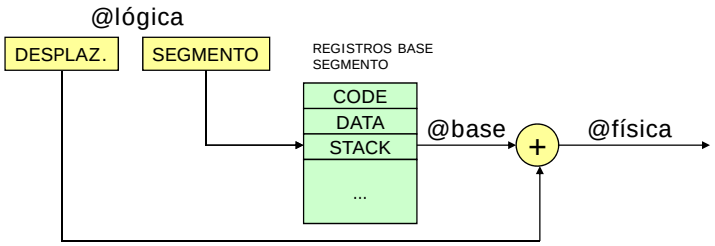
	Espacio lógico	Espacio físico
PDP 11/70	64 Kbytes	256 Kbytes
VAX-11	4 Gbytes	32 Mbytes

Segmentación

- El programa se descompone en segmentos: código, datos, pila, ...
- Cada segmento se identifica por su dirección inicial y tamaño.
- Los segmentos se almacenan de forma contigua en memoria y de forma disjunta entre segmentos.



Segmentación: Implementación Hardware

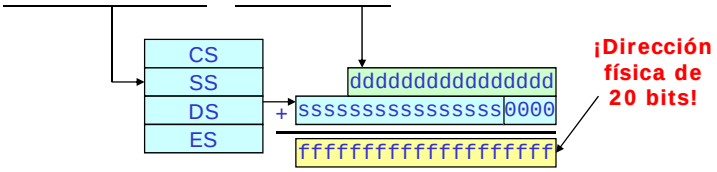


- Un cambio de contexto (usuario o programa) implica cambiar el contenido de los registros.
- Acceso lento: Acceso al banco de registros de segmentos y suma.
- Reubicación muy simple.
- Fragmentación de la memoria.
- Permite protección de los segmentos.

Segmentación: Ejemplo i8086/88

Procesador de 16 bits (bus @ de 16 bits) que generaba direcciones físicas de 20 bits.

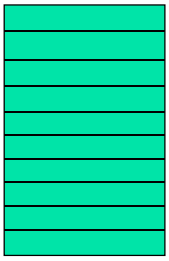
- Disponía de 4 registros de segmento:
  - CS: Segmento de código
  - SS: Segmento de pila
  - DS y ES: Segmentos de datos
- Todas las direcciones se formaban con 2 componentes:
  - Registro de Segmento y desplazamiento



- Cada segmento tenía un tamaño máximo de 64Kbytes.
- Un programa sólo podía direccionar directamente 256Kbytes, para direccionar más memoria había que cambiar el contenido de los registros de segmento.
- Los actuales procesadores de Intel siguen manteniendo los registros de segmento (CS, SS, DS, ES, FS y GS). En modo real funcionan igual que los antiguos i8086.
- ¡Lo vais a necesitar para PROSO!

Paginación

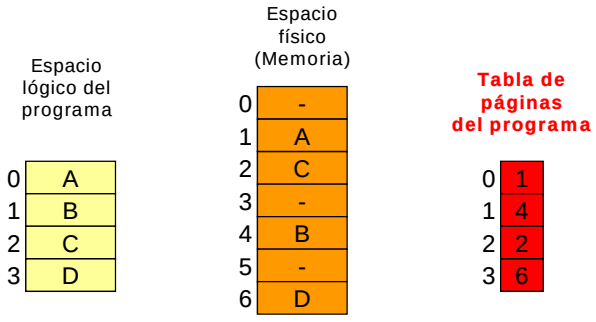
- El espacio lógico se divide en bloques de tamaño fijo → **PÁGINAS**
- Los sistemas actuales tienen páginas con tamaño entre 4 y 16 KB
- El espacio físico (MP) se divide en marcos de tamaño una página (*frames*, tramas).



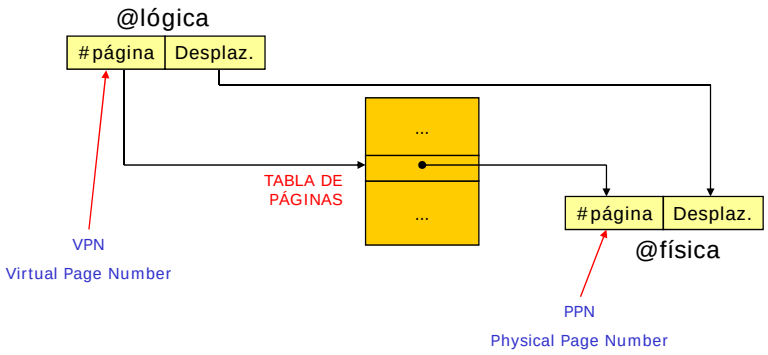
- Los programas se trocean en páginas (están en disco)
- Una página puede colocarse en CUALQUIER marco de página de MP (correspondencia completamente asociativa)
- Las páginas se copian desde disco a MP cuando son referenciadas

Paginación

- Hace falta una estructura de datos para saber qué hay en cada marco de página → **TABLA DE PÁGINAS**.



Paginación: implementación hardware



Paginación

- Características
  - Cálculo rápido de la dirección (no hay operaciones aritméticas).
  - Fragmentación (ficheros pequeños ocupan 1 página completa).
  - Reubicación muy simple.
  - Permite protección de páginas.
  - Página físicas y virtuales suelen tener el mismo tamaño.
  - VPN y PPN pueden tener longitud diferente.
  - La mayoría de sistemas tienen VPN > PPN.

Problema

- En un sistema con direcciones virtuales de 64 bits y direcciones físicas de 43 bits (similar a los primeros procesadores de 64 bits), ¿Cuántos bits se necesitan para el VPN y el PPN si el tamaño de página es de 8 KB?

Implementación de la Tabla de Páginas

- Tabla de páginas de un nivel
  - Cada programa tiene sus propias @ lógicas y físicas.
  - Cada programa tiene su propia Tabla de Páginas.
  - P: bit de presencia (indica si la página está almacenada en MP).
  - M: bit de modificación (indica si la página ha sido modificada en MP).

# de página física	P	M
PPN página Virtual 0		
PPN página Virtual 1		
PPN página Virtual 2		
PPN página Virtual 3		
PPN página Virtual 4		
PPN página Virtual 5		
PPN página Virtual 6		
PPN página Virtual 7		

Tabla de páginas de un nivel

Implementación de la Tabla de Páginas

Tabla de páginas

- La TP puede necesitar mucha memoria
- Una TP, para un espacio de direcciones físicas y lógicas de 32 bits y páginas de 4 KB, necesitaría 2^{20} elementos (más de 10^6).
- Si cada elemento ocupa 3 bytes, la TP ocupa 3 MB.
- Si la MP es de 64 MB, la TP de un programa ocupa casi el 5% de toda la MP
- Con menos de 3 MB de MP no se podría implementar la Memoria Virtual (¡pero los primeros PCs tenían 128 KB!)
- 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

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, se almacena en Memoria Principal
  - 1 acceso a MP necesita 1 acceso a la Tabla de Páginas 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

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
  - → Contiene más entradas de página que las páginas que caben en la cache L1 (contiene traducciones de datos residentes en L2 y en MP).
  - → Procesadores de mediados de los 90 tenían TLB de 128 entradas, 32-64 KB de cache L1 y páginas de 4-16KB

Características principales:

- Integrado en el mismo chip que en el procesador
- Pocas entradas (64-128) (1 entrada por página)
- Completamente asociativo
- Tasa de fallos muy baja
- Muy rápido (debido a que tiene pocas entradas de pocos bits)
- Algoritmo de reemplazo (LRU, PsudoLRU, FIFO, Random, ..)

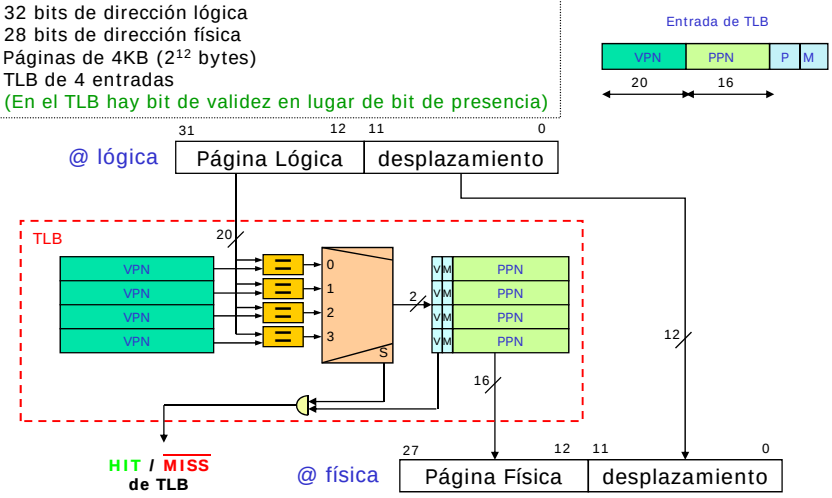
Problema

- Translation Lookaside Buffer (TLB)
  - → Procesadores de mediados de los 90 tenían TLB de 128 entradas, 32-64 KB de cache L1 y páginas de 4-16KB

¿Cuánta memoria se puede traducir directamente con el TLB?

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)



Problema

- Un procesador tiene un TLB con una tasa de aciertos h = 0,95. Cuando hay acierto de TLB, la traducción requiere 1 ciclo (TsaTLB = 1). La penalización por fallo de TLB es de 160 ciclos.

  a) ¿Cuál es el tiempo medio necesario para realizar la traducción de una dirección?

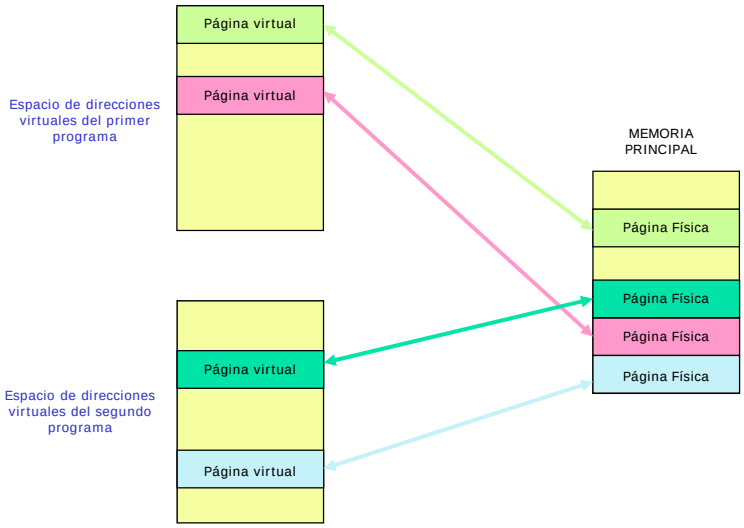
- Suponiendo que, sin TLB, traducir una dirección cuesta 135 ciclos

  b) ¿Qué ganancia en el tiempo de traducción supone tener un TLB frente a no tenerlo?

Paginación: protección

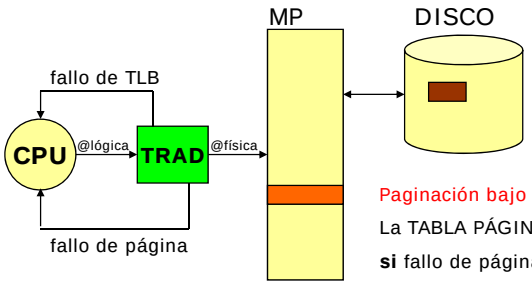
- Cada programa tiene su propia Tabla de Páginas
- Ventajas
  - Los programas comparten espacio físico de direcciones, pero tienen espacios virtuales distintos
  - El sistema de traducción de direcciones asegura que las páginas virtuales de cada programa se mapean en páginas físicas distintas (en MP y en disco)
  - Si dos programas quieren compartir sus datos, algunos SO permiten realizar una petición específica para que algunas de sus direcciones virtuales se asignen a las mismas direcciones físicas
- Inconvenientes
  - El mapeo de direcciones virtuales a físicas es parte del estado del programa
  - Cuando el SO realiza un cambio de contexto, hay que invalidar el TLB
  - Cuando se comienza a ejecutar un programa hay muchos fallos de TLB
  - Para solventar este problema, algunos sistemas actuales incorporan un identificador de proceso en el TLB (coexisten entradas de procesos distintos)

Paginación: protección



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 PÁGINAS dispone de un bit de presencia

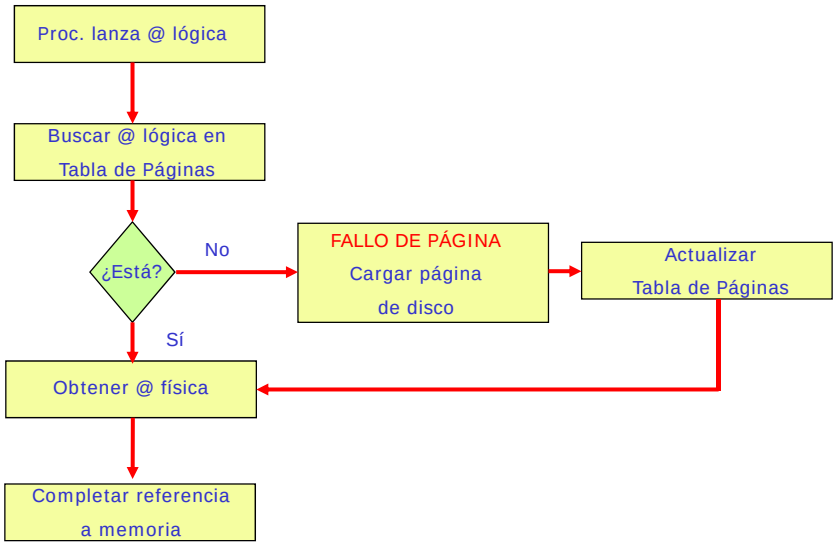
si fallo de página **entonces**

Reemplazar página: MP → DISCO

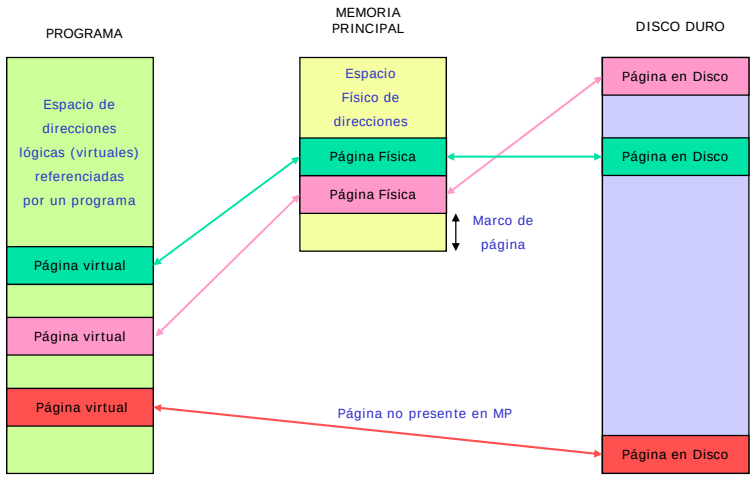
Cargar la página solicitada: DISCO → MP

fsi

Paginación bajo demanda (sin TLB)



Memoria virtual



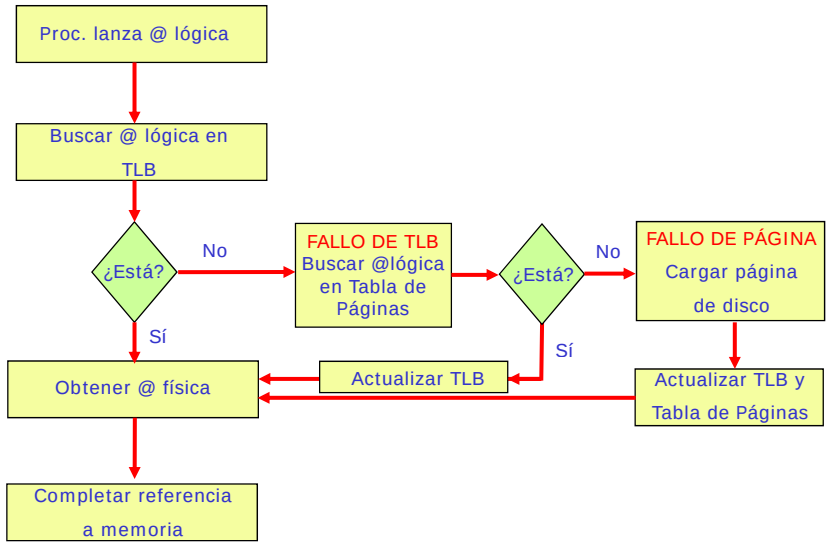
Memoria virtual

- ¿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, 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

Memoria virtual con TLB

- Fallo de TLB
  - requiere un tiempo relativamente corto para ser resuelto si la página está en la Tabla de Páginas
  - típicamente, se resuelve en unos centenares de ciclos
- Fallo de página
  - necesita acceder al disco (varios milisegundos)
  - puede tardar en resolverse varios millones de ciclos
  - Los SO suelen aprovechar un fallo de página para cambiar de contexto

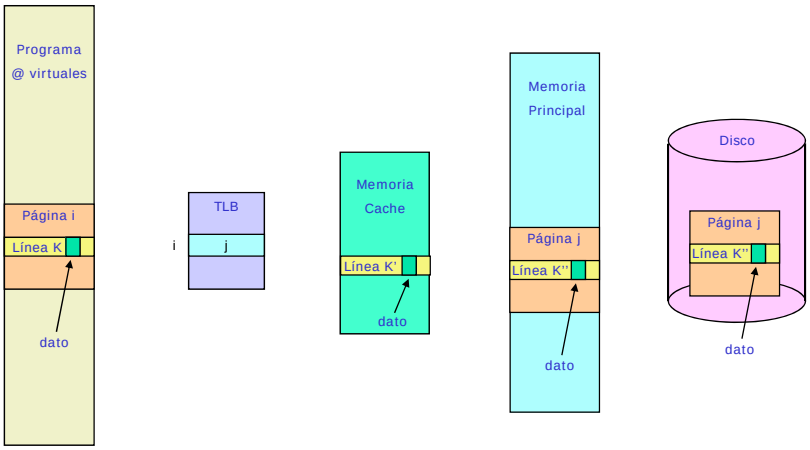
Paginación con TLB (bajo demanda)



Juntando Memoria Virtual y Memoria Cache

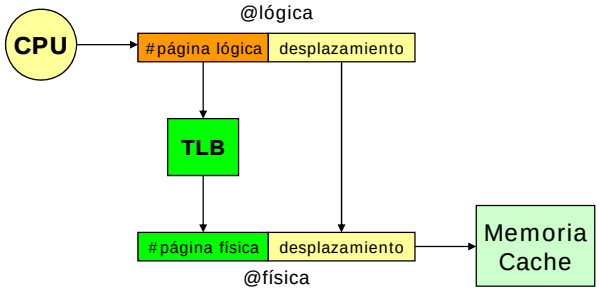
- La traducción de direcciones y la memoria cache son conceptos ortogonales:
  - La memoria cache permite acelerar los accesos a memoria
  - La traducción de direcciones permite soportar memoria virtual
    - El TLB es sólo un mecanismo de aceleración del proceso de traducción
- Un sistema puede tener 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



Juntando Memoria Virtual y Memoria Cache

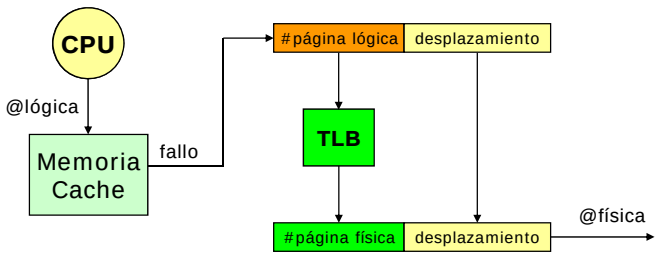
- Traducción antes de acceder a Memoria Cache



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

Juntando Memoria Virtual y Memoria Cache

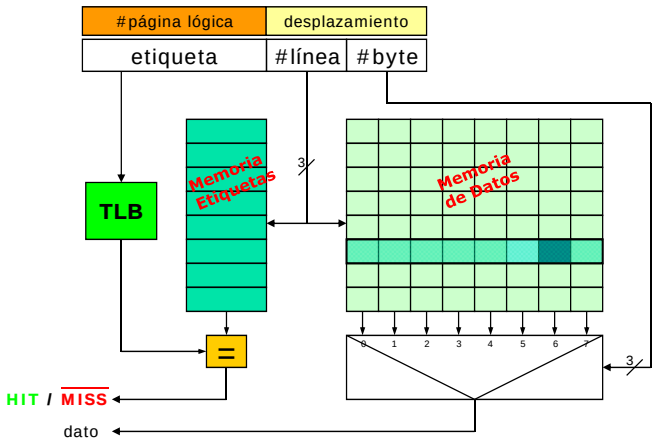
- Traducción después de acceder a Memoria Cache



- Memoria Cache de direcciones lógicas
- Se realiza traducción SÓLO en caso de fallo en MC
- Aumenta el coste de un fallo de MC

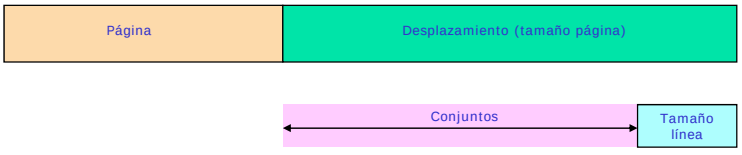
Juntando Memoria Virtual y Memoria Cache

- Traducción en TLB y acceso a Memoria Cache simultáneos

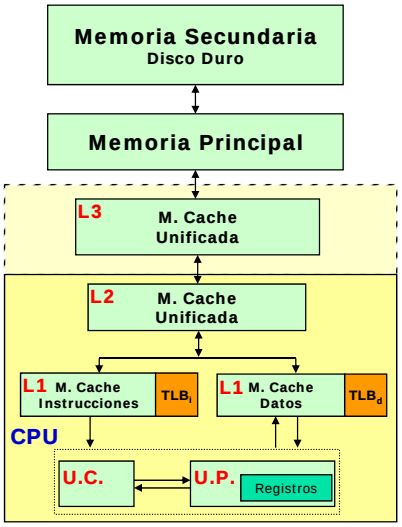


Juntando Memoria Virtual y Memoria Cache

- Traducción en TLB y acceso a Memoria Cache simultáneos
  - Se busca en la MC con la parte de la dirección que corresponde al desplazamiento (línea y byte de la línea)
  - La memoria de etiquetas contiene etiquetas FÍSICAS
  - Se traduce únicamente la página LÓGICA que corresponde a la etiqueta y se comprueba si la línea de la MC es la línea física buscada.
  - Restringe el tamaño de la Memoria Cache:
    - #conjuntos · tamaño línea ≤ tamaño página



Estructura de la Jerarquía de Memoria



Ejecución de instrucción `incl (%ebx)`

En el **PEOR CASO** puede provocar:

- Al leer instrucción
  - fallo TLB_i
  - fallo página (acceso a disco)
  - fallo L1 instrucciones
  - fallo L2
  - fallo L3
- Al leer dato
  - fallo TLB_d
  - fallo página (acceso a disco)
  - fallo L1 datos
  - fallo L2
  - fallo L3

- Al escribir dato
  - acierto TLB_d
  - acierto L1 datos

Traducción de direcciones con TLB

- ¿Dónde está el TLB?: ejemplo Pentium III 0,13 micras