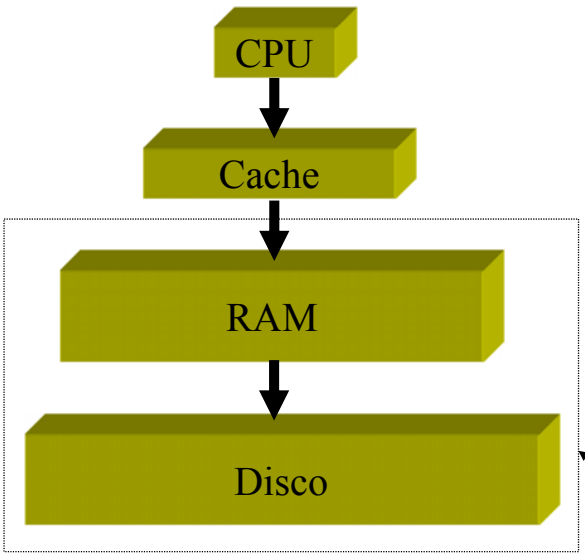
Jerarquía de Memoria
Memoria Virtual

Marcelo Tosini - Elías Todorovich

Arquitectura I - Curso 2015

Segundo nivel: memoria principal



- Rendimiento
  - latencia
  - ancho de banda
- Nivel inferior demasiado lento respecto al superior

Memoria Virtual

- La memoria virtual involucra a dos niveles de la jerarquía de memoria: MP (DRAM), y MS (discos magnéticos). Recordemos que la caché involucra a otros niveles de jerarquía de memoria: caché (SRAM) y MP (DRAM).

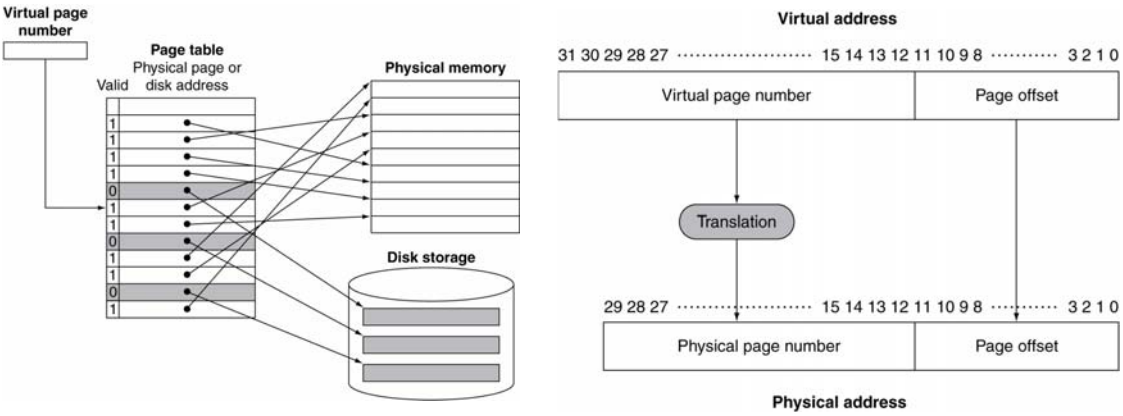
Parameter	First-level cache	Virtual memory
Block (page) size	16–128 bytes	4096–65,536 bytes
Hit time	1–2 clock cycles	40–100 clock cycles
Miss penalty	8–100 clock cycles	700,000–6,000,000 clock cycles
(Access time)	(6–60 clock cycles)	(500,000–4,000,000 clock cycles)
(Transfer time)	(2–40 clock cycles)	(200,000–2,000,000 clock cycles)
Miss rate	0.5–10%	0.00001– 0.001%
Data memory size	0.016–1MB	16–8192 MB

Memoria Virtual

- Se usa a la memoria principal como “cache” para el disco
  - Se maneja en conjunto entre HW (MMU) y sistema operativo
- Los programas comparten la memoria principal
  - Cada programa tiene un espacio de direcciones virtuales privado
  - Protegido de otros programas
- CPU y OS traducen direcciones virtuales en direcciones físicas
  - En este nivel los bloques se llaman páginas
  - Los fallos en la traducción se llaman fallos de página

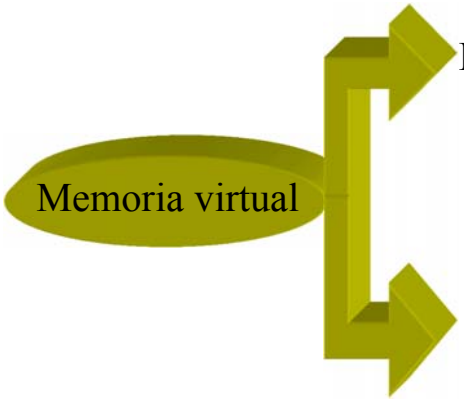
Traducción de direcciones

- Páginas de tamaño fijo (Ej., 4K)



- Segmentos de tamaño variable
- Segmentos con páginas de tamaño fijo

Tipos de memoria virtual



Paginada :

- bloques fijos
- tamaño : 512 a 8192 bytes
- relacionada con aspectos físicos del nivel inferior

Segmentada :

- bloques variables
- tamaño : 1 a 2^{32} bytes
- relacionada con aspectos de programación

Funcionamiento de la MV

¿Por qué es mejor paginación que segmentación?

- La partición en páginas es trasparente al programador.
- La dirección virtual está formada por una sola palabra, mientras que en segmentación está formada por dos, segmento y desplazamiento.
- Es fácil reemplazar un bloque ya que todos tienen el mismo tamaño.
- Elimina la fragmentación externa de memoria aunque no elimina completamente la interna.

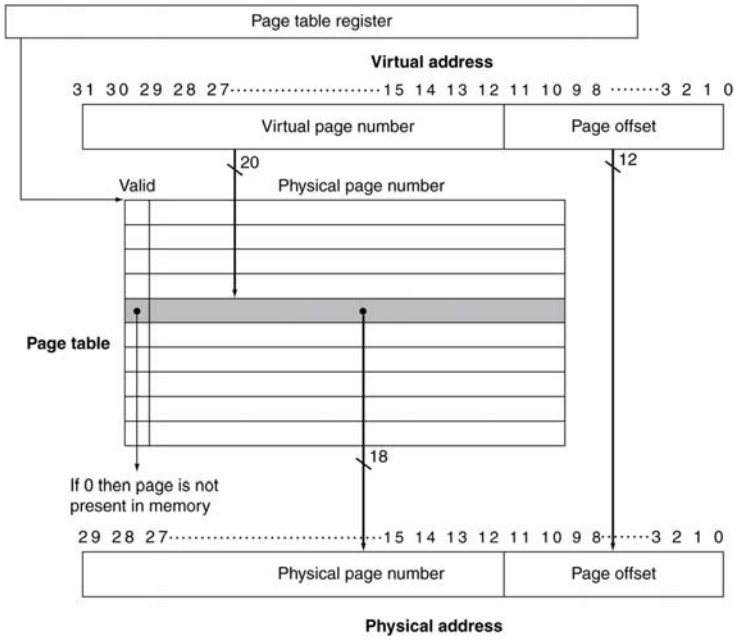
¿Por qué es mejor segmentación que paginación?

- Facilita las operaciones con la memoria, ya que programas y tablas de datos tienen diferente tamaño.
- Permite más fácilmente compartir zonas de memoria entre procesos.
- Permite la existencia de varios espacios de direcciones, por lo que facilita la separación entre código y datos y la modificación independiente de programas.

Conceptos a definir

- ¿Cuándo se debe producir una transferencia entre MS y MP?
- ¿Cuál es el tamaño de los bloques de información que deben ser transferidos entre la memoria principal (MP) y la memoria secundaria (MS)?
- ¿Cuáles son los mecanismos automáticos de traducción de la dirección virtual (DV) a la dirección física o real (DR)?
- Cuando ocurra la transferencia de un bloque desde MS, y la MP esté llena, ¿cuál es el bloque de MP a eliminar para permitir dicha transferencia?

Traducción mediante tabla de páginas



Esta tabla emplea un esquema de correspondencia directa (*direct mapping*).

Sólo se traduce el nº página al marco (el offset queda igual)

La MMU. Traducción de páginas

Ejemplo: Memoria virtual: 4 GB (2^{32}), real: 16 MB (2^{24}). Páginas: 4 kB (2^{12})

CPU → {FFAACD00} D.V.

20 b **12 b**

FFAAC | **D00**

00000

312

FFFFF

312 | **D00**

{312D00} → MEMORIA D.R

¡Necesita memoria SRAM 2^{20}x12 bits!

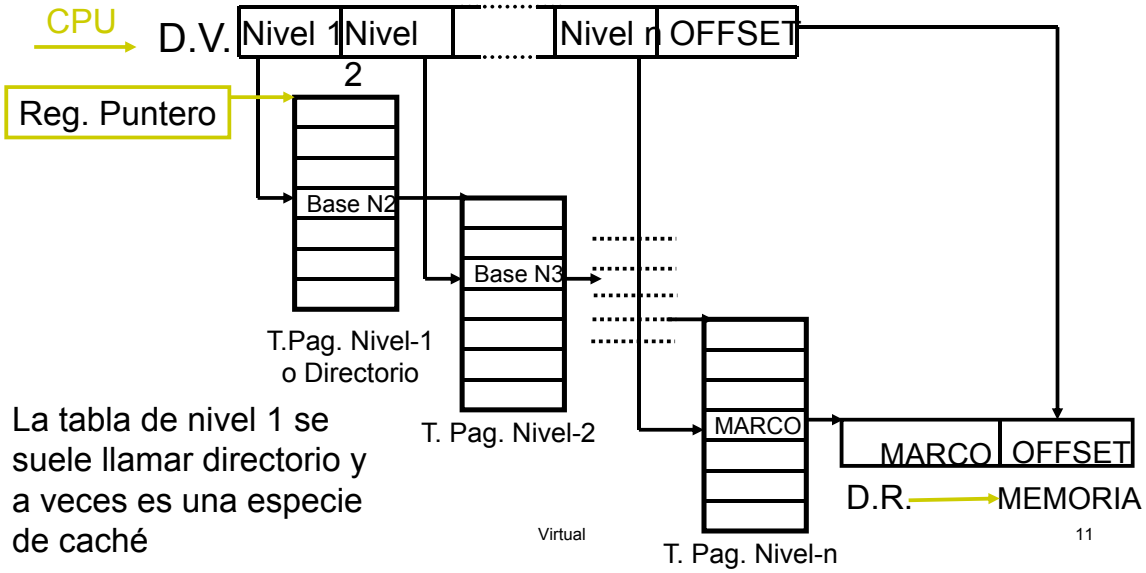
Impensable un esquema tipo completamente asociativo (2^{20} comparadores)

Para reducir el tamaño de la tabla de páginas, muchas veces no es completa, sino que se va creando según las necesidades del proceso (en una o en dos direcciones para pila y *heap*)

La MMU. Traducción de páginas

❑ **Tabla de páginas multipaginada o multinivel**

Se usa también para reducir el tamaño de tabla de páginas (no todas las subtablas estarán presentes)

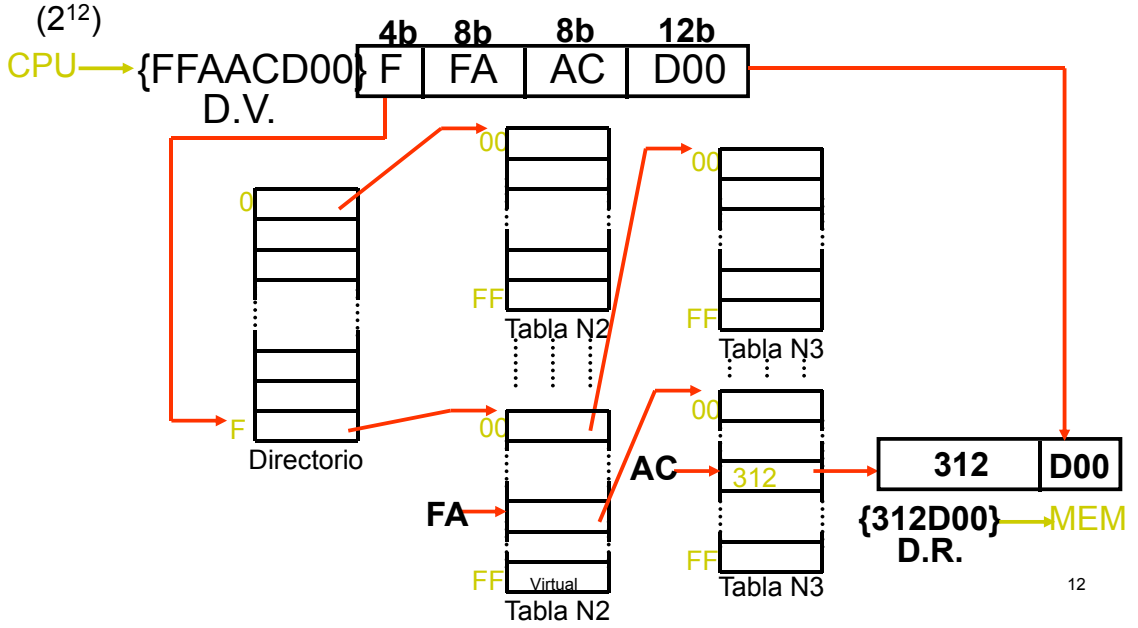


La tabla de nivel 1 se suele llamar directorio y a veces es una especie de caché

La MMU. Traducción de páginas

Tabla de páginas en 3 niveles

Ejemplo: Memoria: Virtual: 4GB (2^{32}), Real: 16MB (2^{24}). Páginas:4kB (2^{12})



La MMU. Traducción de páginas

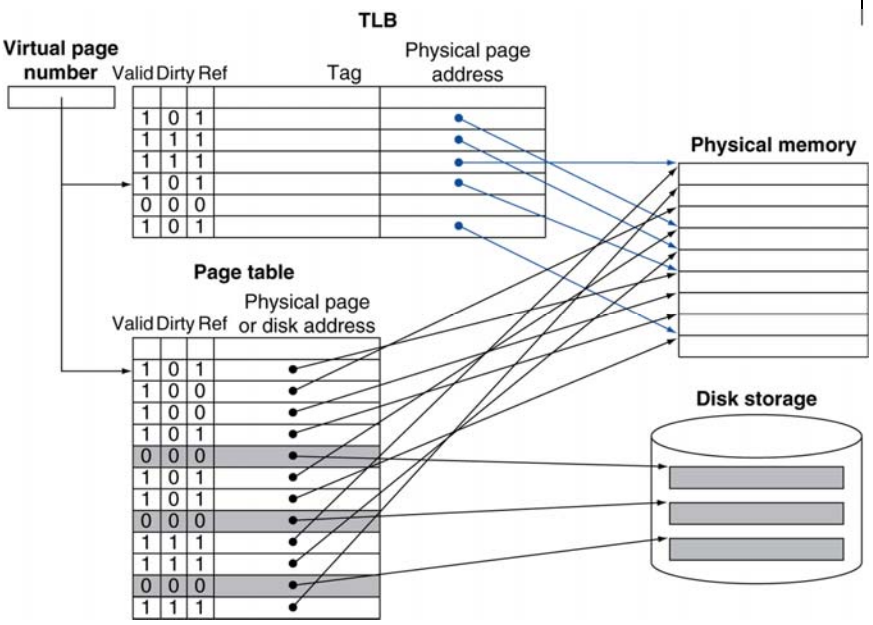
¿Cómo reducir la penalización del sistema de traducción automática de direcciones?

- Optimizando el tamaño de página
  - Pág. pequeñas.- Reducen la fragmentación, pero aumentan el trasiego (*thrashing*)
  - Pág. grandes.- Aumenta el rendimiento en el cambio MP⇔MS

¡Cada acceso a memoria se convierte en 2! Uno para traducir la dirección (o varios con tabla multinivel) y luego ya el acceso al dato buscado

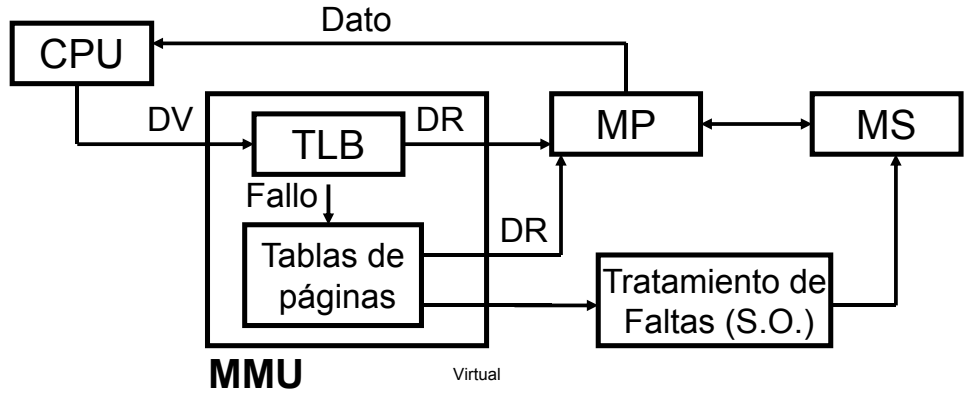
- Se utilizan traductores de menor tamaño y rápido acceso (tipo caché), denominados TLB's (*Translation Look-Aside Buffers*)
  - Completamente asociativos (*Fully Associative Mapping*)
  - Asociativos por vías (*Set Associative Mapping*)

MMU. Traducción usando TLB



La MMU. TLB

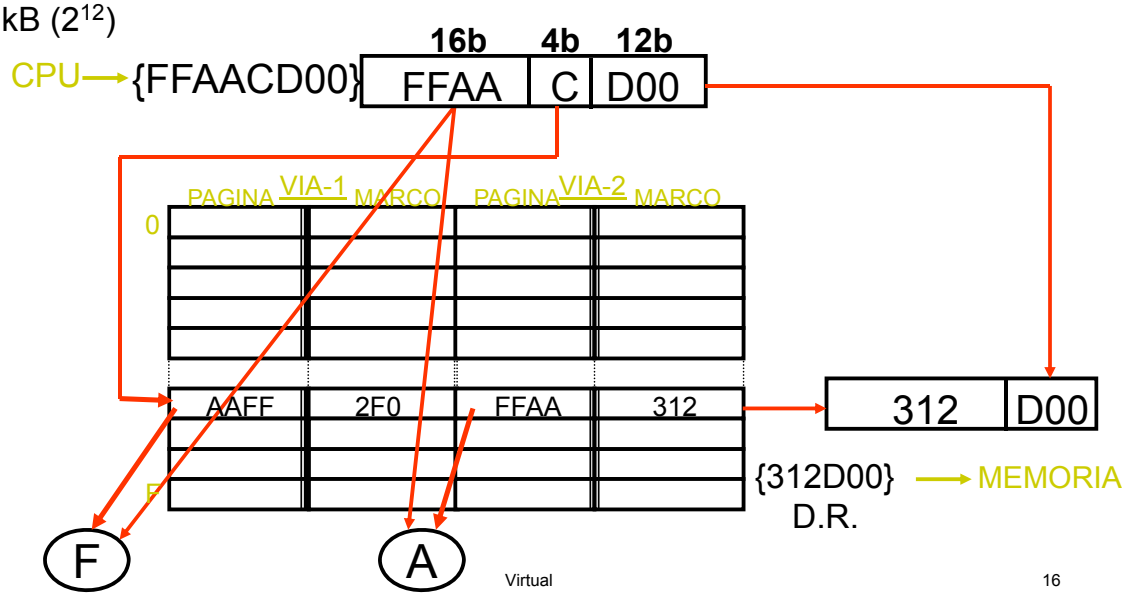
- El TLB (Translation Lookaside Buffer) o ATC (Address Translation Cache) es para la tabla de páginas como la caché para la memoria
- Por su reducido tamaño (menor que una caché) se suele emplear un esquema completamente asociativo o asociativo por vías para minimizar los fallos, que penalizan tanto o más que en caché



La MMU. TLB

TLB Asociativo por vías, 2 vías, 16 entradas/vía

Ejemplo: Memoria: Virtual: 4 GB (2^{32}), Real: 16 MB (2^{24}). Páginas: 4 kB (2^{12})



TLB Misses

- Si la página está en memoria
  - Se carga el PTE desde la memoria
  - Se puede implementar en HW
    - Puede volverse complejo
  - O en software
    - Se produce una excepción especial, con un handler optimizado
- Si la página no está en memoria (page fault)
  - El OS trae la página y actualiza la tabla de páginas
  - Reinicia la instrucción que falló.

TLB Miss Handler

- Un fallo a nivel del TLB indica
  - Página presente, pero la PTE no está en el TLB
  - La página no está presente
- Se debe detectar el fallo de TLB antes de escribir el registro de destino
  - Se produce una excepción
- El handler copia la PTE de memoria a TLB
  - Entonces reinicia la instrucción
  - Si la página no está presente, se produce un fallo de página

Page Fault Handler

- Se una la dirección virtual que falló para encontrar la PTE
- Se busca la página en el disco
- Se elige qué página sustituir
  - Si el bit dirty está activo, escribir primero en disco
- Cargar la página y actualizar la tabla de páginas
- Indicar que el proceso que falló se puede reactivar
  - Reiniciar a partir de la instrucción que falló

Información en la Tabla de Páginas

Se llama descriptor:

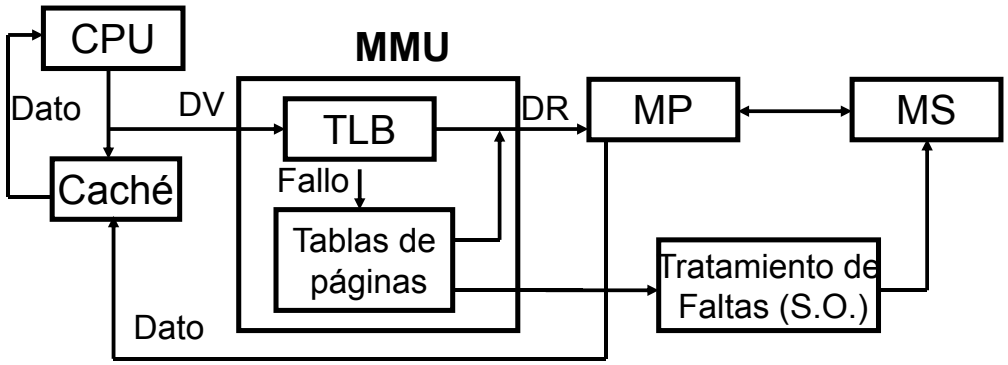
- Page frame: Sirve para obtener la dirección real (= FRAME & OFFSET)
- Bits para control:
  - Present bit: ’1’ indica que la página está en memoria
  - Use bit: ’1’ indica que algún elemento de la página fue referenciado. Se usa para elegir qué página reemplazar.
  - Dirty bit:’1’ indica que algón elemento de la página fue escrito.
  - Protection bits: supervisor, only-readable, non-cacheable, usada por el OS.
  - Replacement bits: Para uso del algoritmo de reemplazo (LRU, etc).

Estrategia de escritura

- Para reducir la tasa de fallos se usa *least-recently* (LRU) para reemplazos
  - El bit de referencia (aka use bit) se hace 1 cuando se accede a la página
  - El OS periodicamente resetea esos bits
- Las escrituras en disco llevan millones de ciclos
  - Se transfiere todo el bloque, no palabras individuales
  - Write through es impractico. Se usa write-back
  - Se usa *dirty bit*

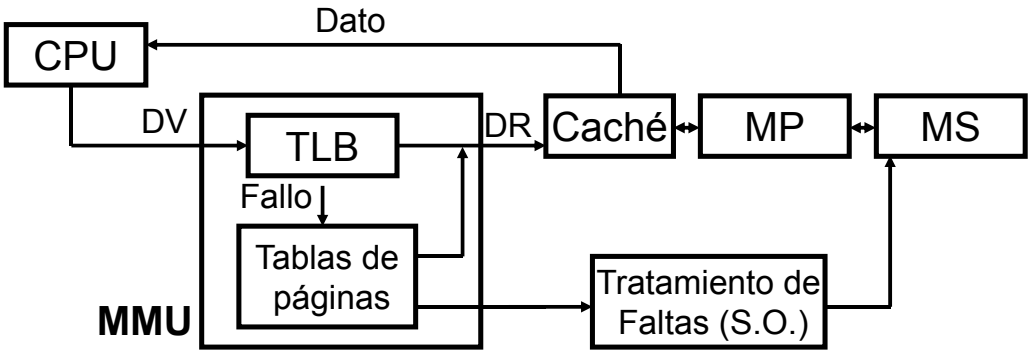
Integración de TLB y caché

- Caché virtual (se accede con dirección virtual)

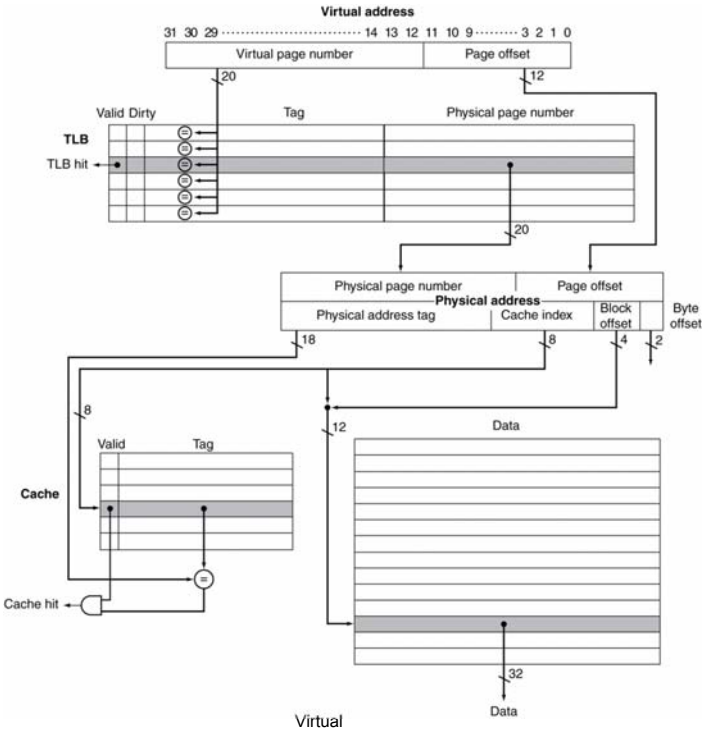


Integración de TLB y caché

- Caché real (se accede con dirección real)

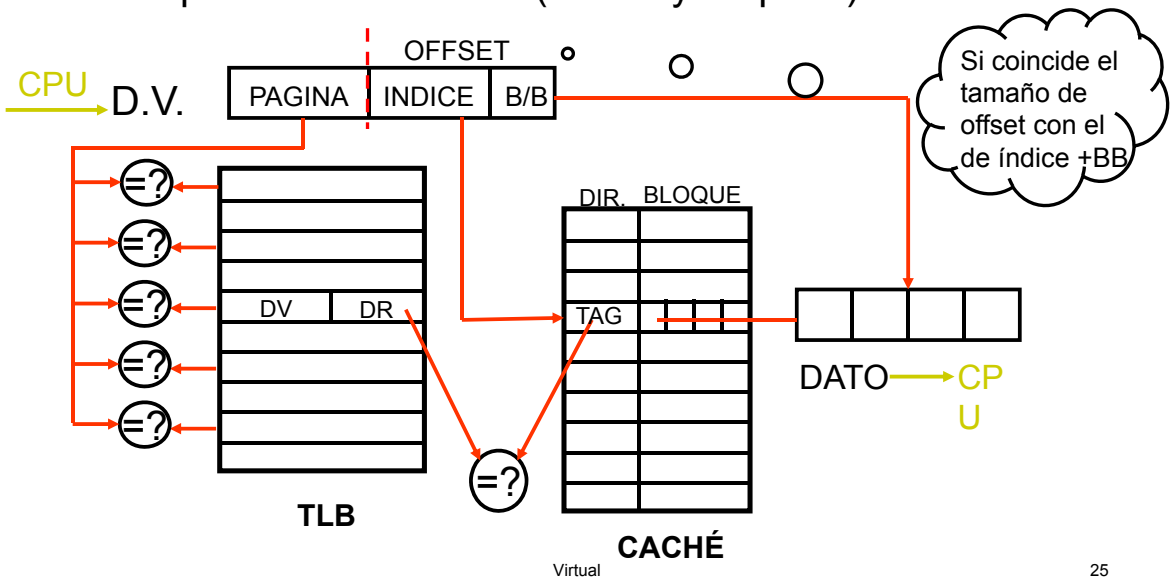


Integración TLB / Cache



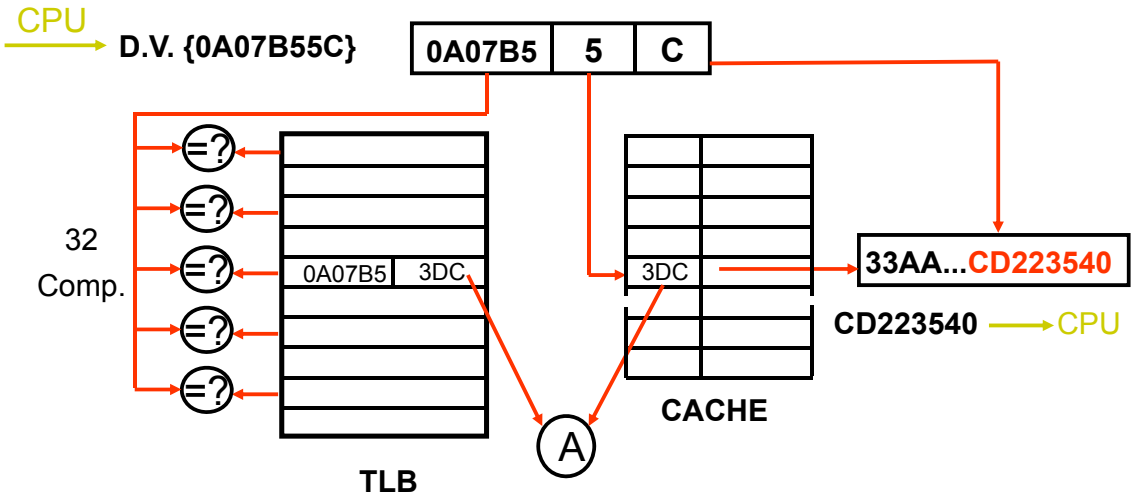
Integración de TLB y caché

- Caché real con acceso en paralelo al marco del TLB y la etiqueta de la caché. Después sólo queda la comparación de ambos (marco y etiqueta)



Integración de TLB y caché

Ejemplo.- DV: 32b; DR: 20b; T. Pág: 256 Bytes
TLB CA de 32 Entradas; Caché CD 256 Bytes, con 16 B/B.



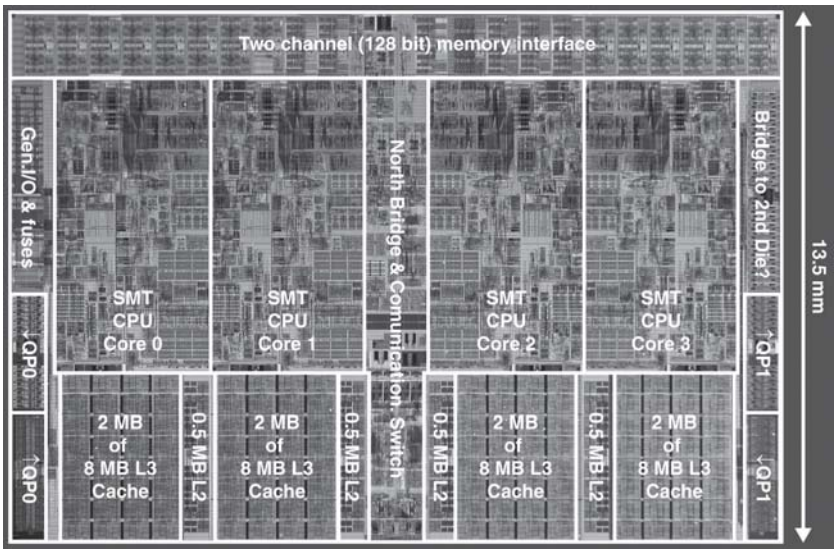
"El tamaño de la unidad caché es igual que el tamaño de la página"

Protección de Memoria

- Diferentes tareas pueden compartir sus espacios de direcciones virtuales
  - Pero necesitan protección contra accesos indevidos
  - Eso requiere asistencia del OS
- Soporte HW para protección del OS
  - Modo supervisor/kernel/privilegiado
  - Instrucciones privilegiadas
  - La tabla de páginas e información de estado se accede solamente en modo privilegiado
  - Se llaman excepciones de sistema (Ej., syscall en MIPS)

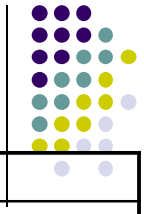
Multilevel On-Chip Caches

Intel Nehalem 4-core processor



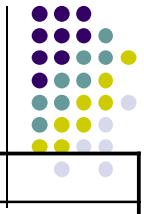
Per core: 32KB L1 I-cache, 32KB L1 D-cache, 512KB L2 cache

2-Level TLB Organization



	Intel Nehalem	AMD Opteron X4
Virtual addr	48 bits	48 bits
Physical addr	44 bits	48 bits
Page size	4KB, 2/4MB	4KB, 2/4MB
L1 TLB (per core)	L1 I-TLB: 128 entries for small pages, 7 per thread (2×) for large pages L1 D-TLB: 64 entries for small pages, 32 for large pages Both 4-way, LRU replacement	L1 I-TLB: 48 entries L1 D-TLB: 48 entries Both fully associative, LRU replacement
L2 TLB (per core)	Single L2 TLB: 512 entries 4-way, LRU replacement	L2 I-TLB: 512 entries L2 D-TLB: 512 entries Both 4-way, round-robin LRU
TLB misses	Handled in hardware	Handled in hardware

3-Level Cache Organization



	Intel Nehalem	AMD Opteron X4
L1 caches (per core)	L1 I-cache: 32KB, 64-byte blocks, 4-way, approx LRU replacement, hit time n/a L1 D-cache: 32KB, 64-byte blocks, 8-way, approx LRU replacement, write-back/allocate, hit time n/a	L1 I-cache: 32KB, 64-byte blocks, 2-way, LRU replacement, hit time 3 cycles L1 D-cache: 32KB, 64-byte blocks, 2-way, LRU replacement, write-back/allocate, hit time 9 cycles
L2 unified cache (per core)	256KB, 64-byte blocks, 8-way, approx LRU replacement, write-back/allocate, hit time n/a	512KB, 64-byte blocks, 16-way, approx LRU replacement, write-back/allocate, hit time n/a
L3 unified cache (shared)	8MB, 64-byte blocks, 16-way, replacement n/a, write-back/allocate, hit time n/a	2MB, 64-byte blocks, 32-way, replace block shared by fewest cores, write-back/allocate, hit time 32 cycles

Jerarquía de Memoria
Memoria Virtual

Marcelo Tosini - Elías Todorovich

Arquitectura I - Curso 2015