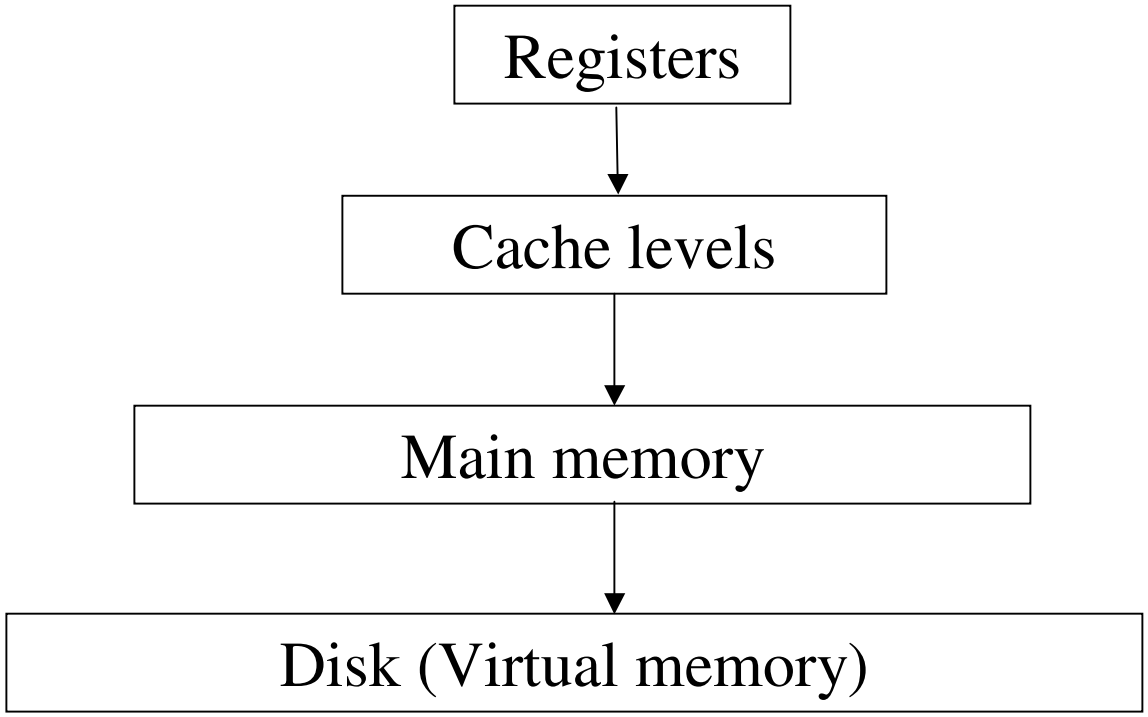
Cache Addressing Basics

CS654 – September 27, 2001

What is a Cache?

- Small memory close to CPU
- Typically SRAM
  - Hence fast
  - Hence expensive too!
- Processor – memory gap
- Principle of locality – make the common case fast!

Memory Hierarchy



Each level is a ‘cache’ for lower levels

Placement

- Units of transfer – ‘blocks’ or ‘lines’
- ‘Hit’ and ‘Miss’
- Fully associative, Direct mapped, Set associative
- Cache size = bytes/line * lines/set * sets/cache (some numerical examples)
- Why these types? – speed/hit rate tradeoff

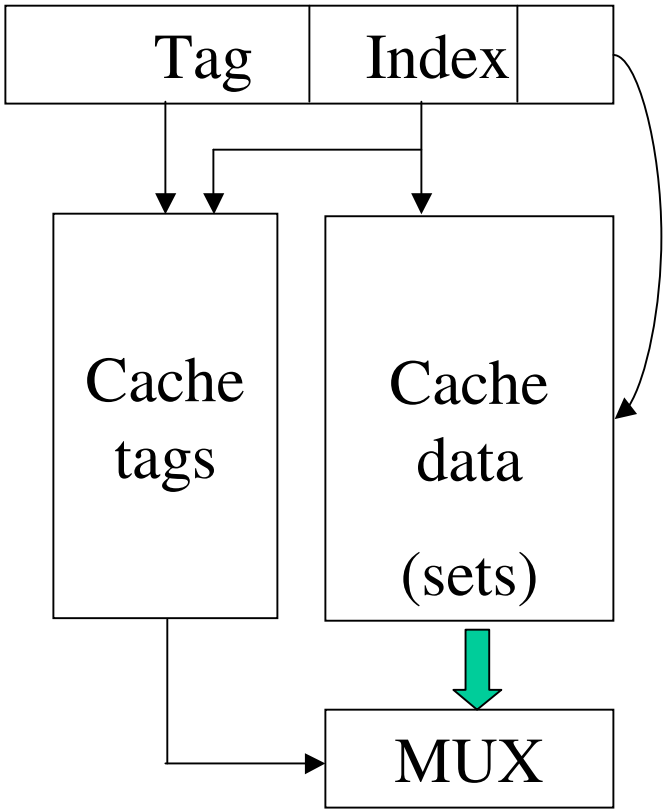
Terminology (digression!)

- Replacement – LRU, random
- Write back / Write through
- Write allocate, no write allocate

Address make up

- Tag, index and offset

Typical organization



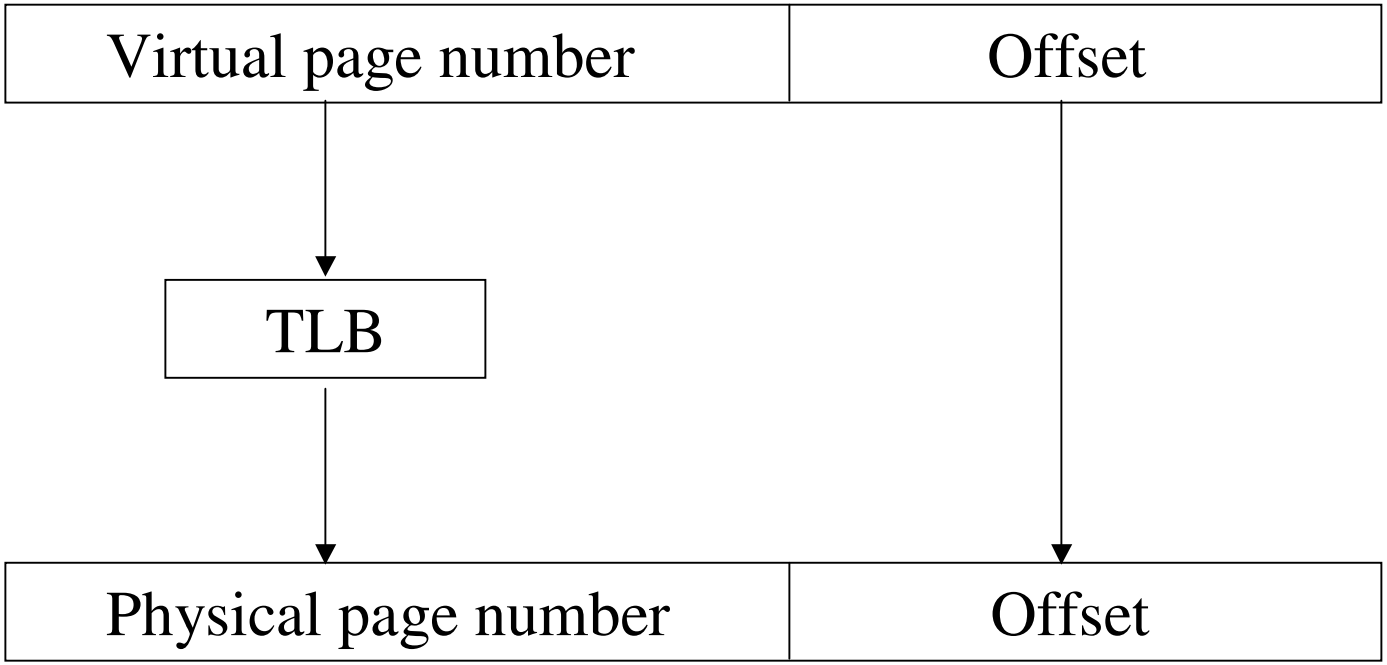
Some examples

- 128M physical memory
  - 64K direct mapped with 64 byte lines
  - 16K 4-way with 32 byte lines

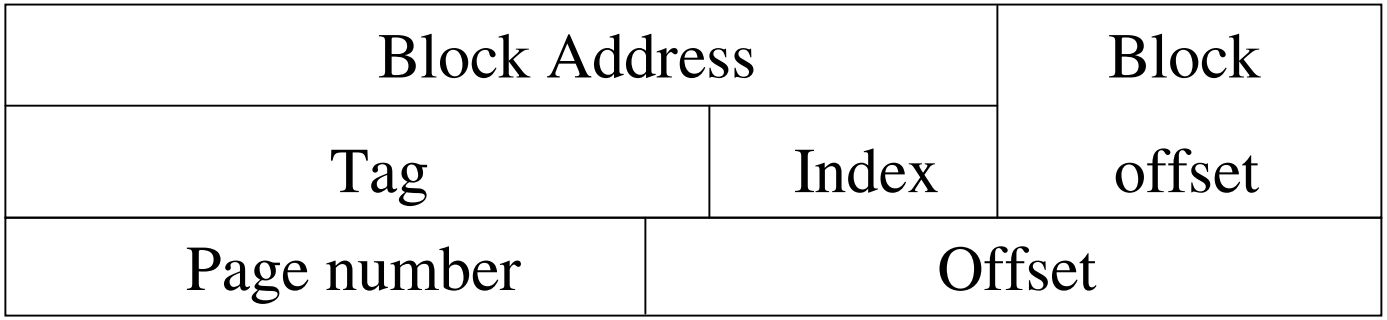
Virtual Memory

- Protection, address spaces, virtual memory
- Paging and segmentation
- Similarity with caches
- Address translation
- TLB – cache of the translation (why?)

Organization



Physical Address with TLB



- Virtual vs physical indexing
- Virtual index physical tag!

Some examples

- 32 bit virtual address, 128 MB main memory, 128 entry fully associative TLB, 64K cache, 64 byte lines