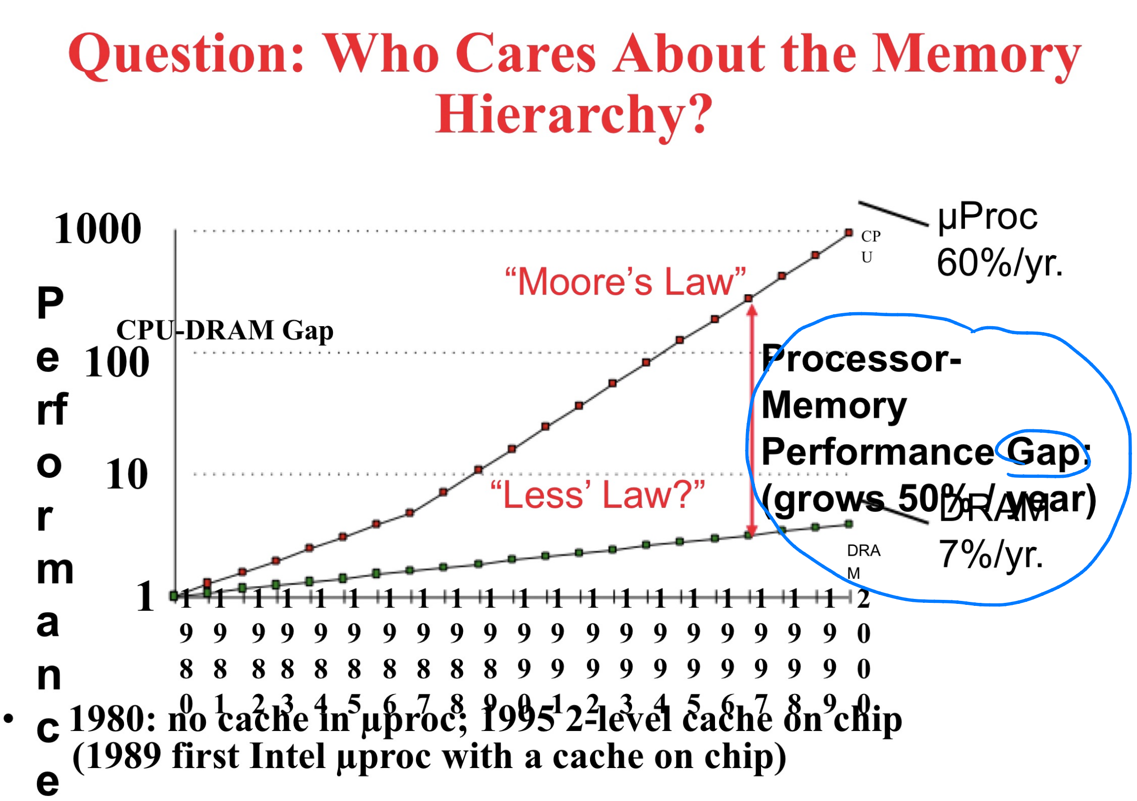
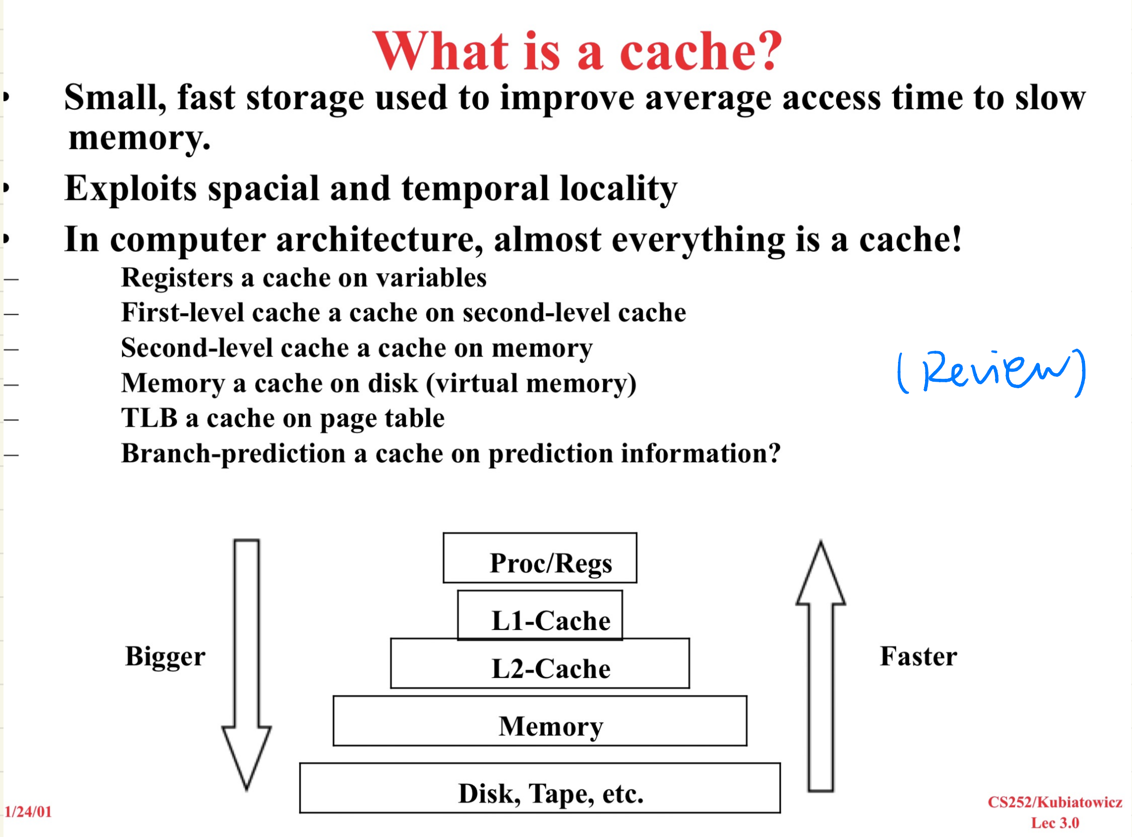
Begin from AMAT.



Question: Who Cares About the Memory Hierarchy?

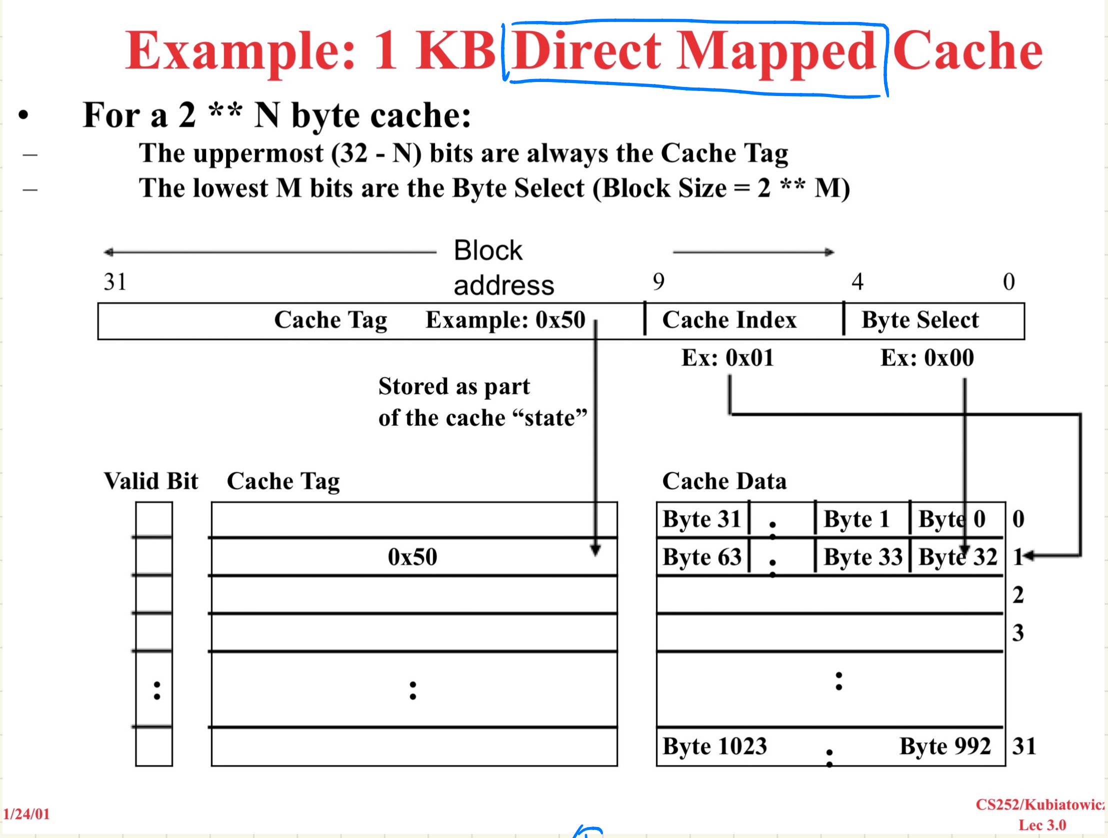
- 1980: no cache in µproc; 1995 2-level cache on chip (1989 first Intel µproc with a cache on chip)



What is a cache?

- Small, fast storage used to improve average access time to slow memory.
- Exploits spacial and temporal locality
- In computer architecture, almost everything is a cache!
  - Registers a cache on variables
  - First-level cache a cache on second-level cache
  - Second-level cache a cache on memory
  - Memory a cache on disk (virtual memory)
  - TLB a cache on page table
  - Branch-prediction a cache on prediction information?

(Review)



Example: 1 KB Direct Mapped Cache

- For a 2 ** N byte cache:
  - The uppermost (32 - N) bits are always the Cache Tag
  - The lowest M bits are the Byte Select (Block Size = 2 ** M)

Block address

31 9 4 0

Cache Tag Example: 0x50 | Cache Index | Byte Select

Ex: 0x01 Ex: 0x00

Stored as part of the cache "state"

Valid Bit Cache Tag Cache Data

0x50

Byte 31 : Byte 1 Byte 0 0

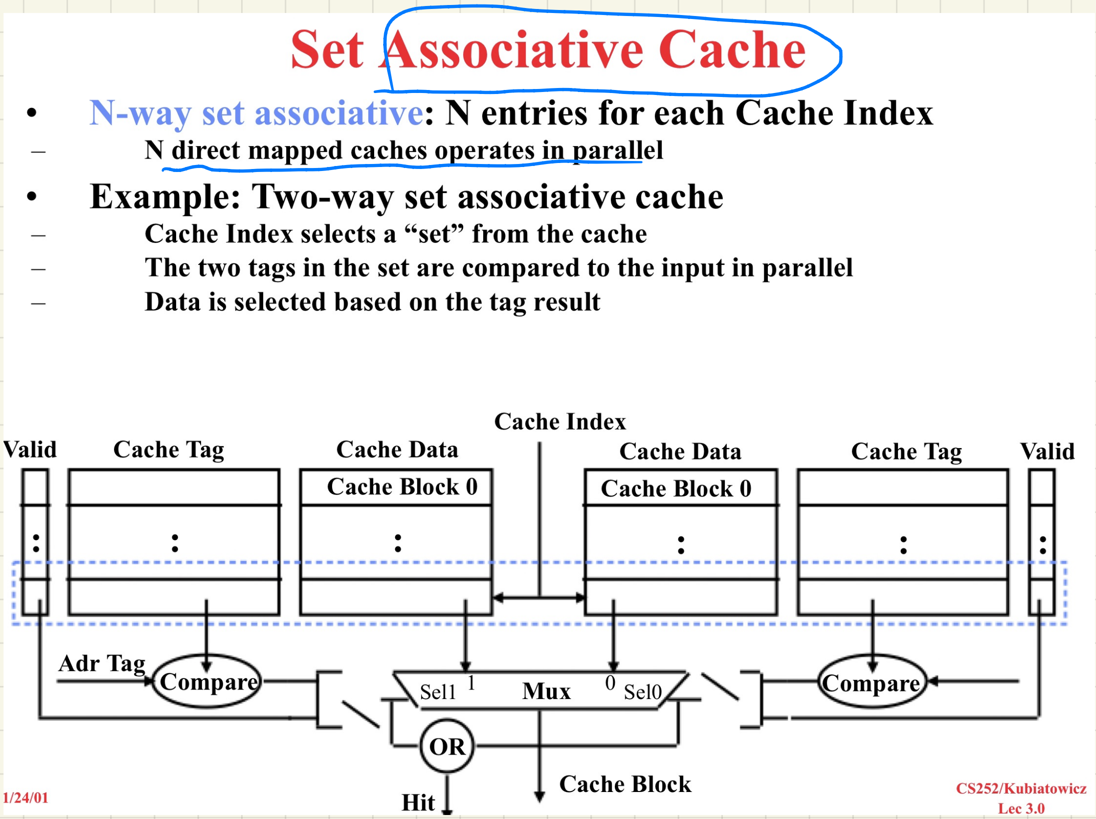
Byte 63 : Byte 33 Byte 32 1

2

3

:

Byte 1023 : Byte 992 31



Set Associative Cache

- N-way set associative: N entries for each Cache Index
  - N direct mapped caches operates in parallel
- Example: Two-way set associative cache
  - Cache Index selects a "set" from the cache
  - The two tags in the set are compared to the input in parallel
  - Data is selected based on the tag result

Cache Index

Valid Cache Tag Cache Data Cache Data Cache Tag Valid

Cache Block 0 Cache Block 0

Adr Tag

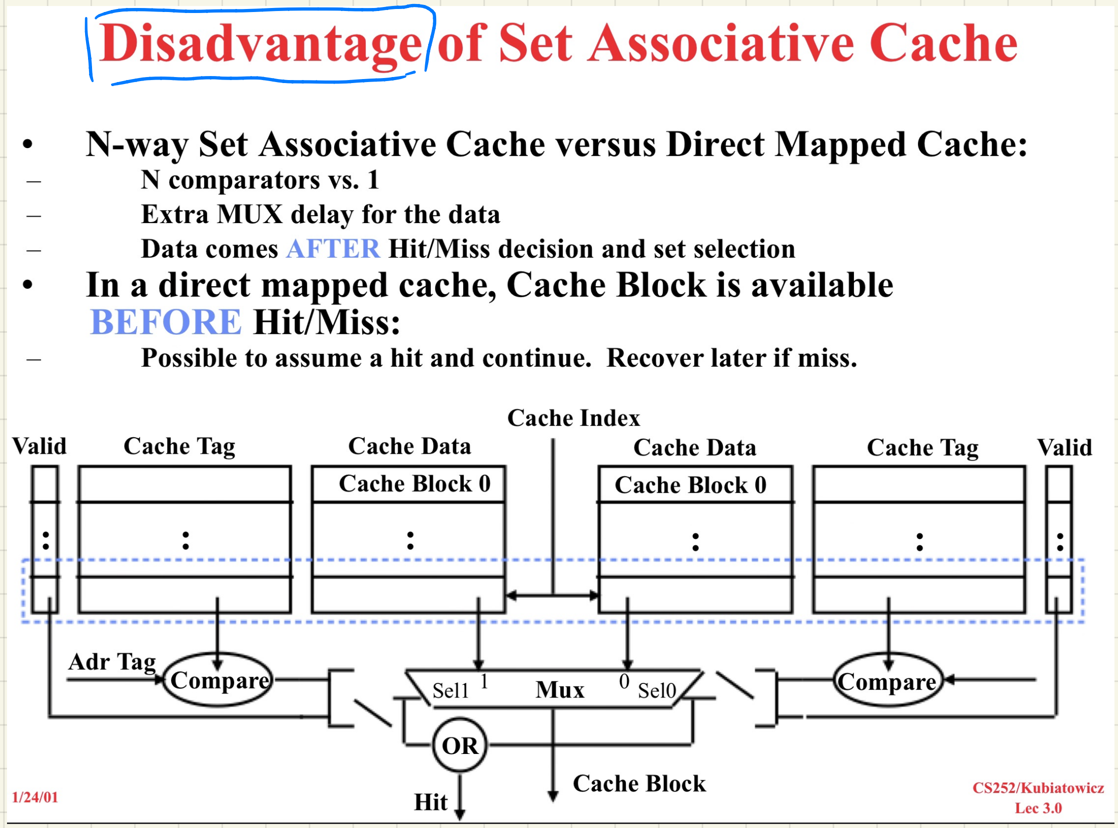
Compare Compare

Sel1 1 Mux 0 Sel0

OR

Hit

Cache Block

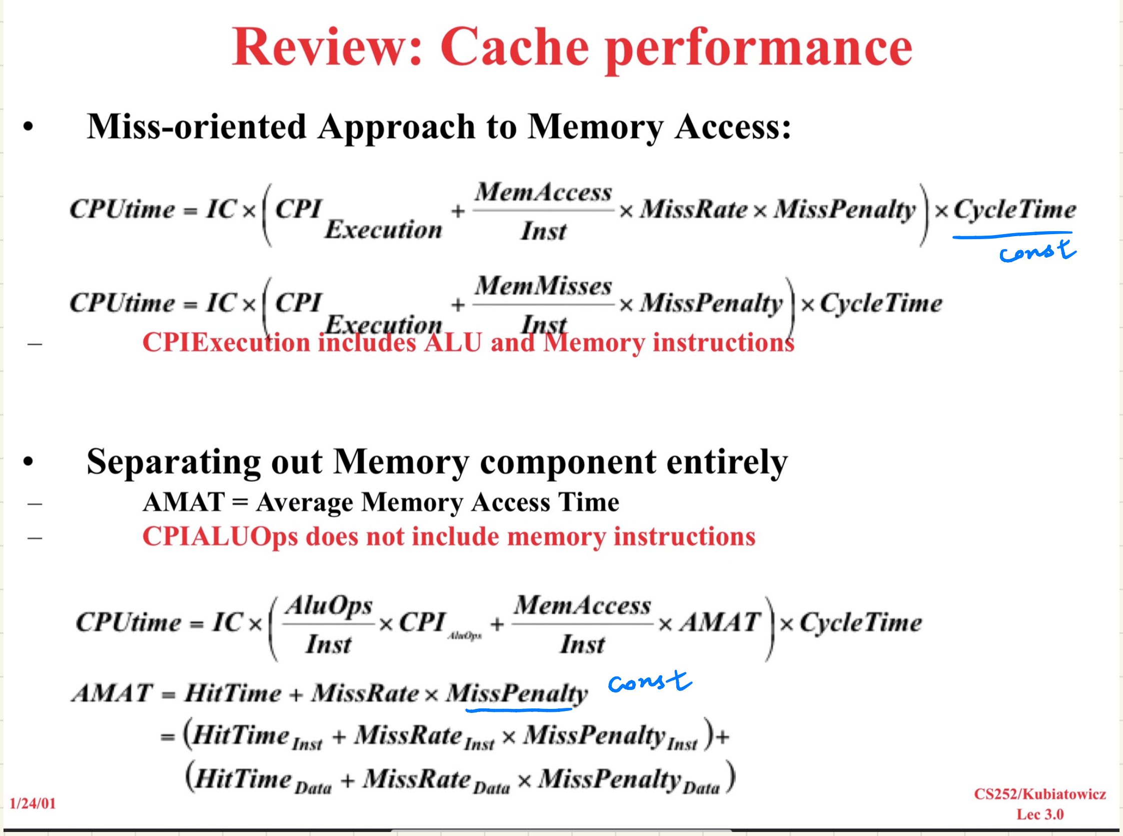


Disadvantage of Set Associative Cache

- N-way Set Associative Cache versus Direct Mapped Cache:
  - N comparators vs. 1
  - Extra MUX delay for the data
  - Data comes AFTER Hit/Miss decision and set selection
- In a direct mapped cache, Cache Block is available BEFORE Hit/Miss:
  - Possible to assume a hit and continue. Recover later if miss.

1/24/01

CS252/Kubiatowicz
Lec 3.0



Review: Cache performance

- Miss-oriented Approach to Memory Access:

$$CPUtime = IC \times \left(CPI_{Execution} + \frac{MemAccess}{Inst} \times MissRate \times MissPenalty \right) \times \underline{CycleTime}$$

const

$$CPUtime = IC \times \left(CPI_{Execution} + \frac{MemMisses}{Inst} \times MissPenalty \right) \times CycleTime$$

  - CPIExecution includes ALU and Memory instructions

- Separating out Memory component entirely
  - AMAT = Average Memory Access Time
  - CPIALUOps does not include memory instructions

$$CPUtime = IC \times \left(\frac{AluOps}{Inst} \times CPI_{AluOps} + \frac{MemAccess}{Inst} \times AMAT \right) \times CycleTime$$

$$AMAT = HitTime + MissRate \times \underline{MissPenalty}$$

const

$$= \left(HitTime_{Inst} + MissRate_{Inst} \times MissPenalty_{Inst} \right) + \left(HitTime_{Data} + MissRate_{Data} \times MissPenalty_{Data} \right)$$

1/24/01

CS252/Kubiatowicz
Lec 3.0