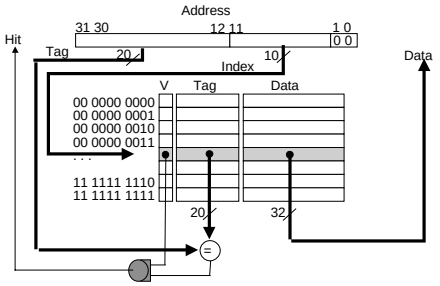
Cache Behavior

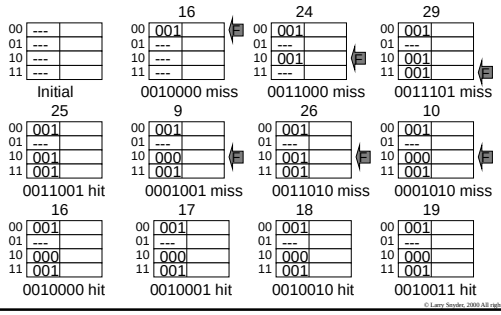
Constructing an effective cache requires the balancing of many properties. Bigger is always better, but how it is arranged is also important.

Recall Direct Mapped Cache

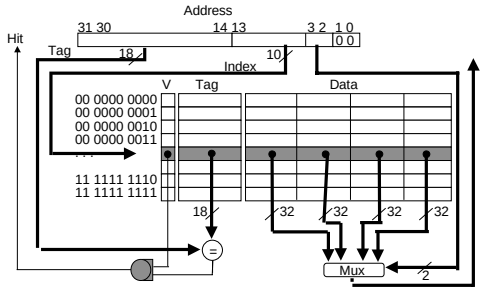


Reference Sequence

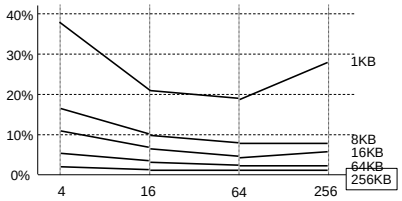
Memory Refs: 16, 24, 29, 25, 9, 26, 10, 16, 17, 18, 19



Direct Mapped Cache (4 word blks)



Miss rate vs block size



VAX traces (Agarwal,1987)

Benefits of Multiword Blocks

Increasing block size improves performance, to a point

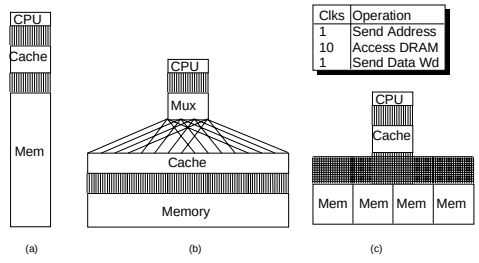
- Larger blocks increase benefits of spatial locality
- Larger blocks = fewer blocks for a given cache size = greater likelihood a useful block is flushed when another block is brought in (conflict misses)

Memory request techniques --

- Early restart
- Requested word first

Pgm	Wd	Inst Miss	Data Miss	Effective Miss Ratio
GCC	1	6.1%	2.1%	5.4%
	4	2.0%	1.7%	1.9%
Spice	1	1.2%	1.3%	1.2%
	4	0.3%	0.6%	0.4%

Memory Organizations



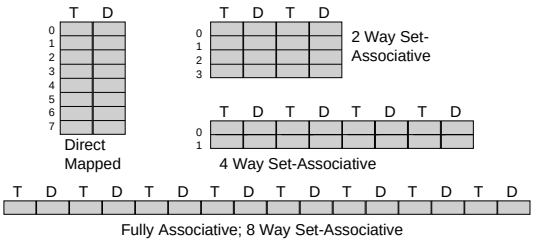
Clks	Operation
1	Send Address
10	Access DRAM
1	Send Data Wd

Miss Penalty: 1+4x10+4x1=45 1+1x10+1=12 1+1x10+4x1=15

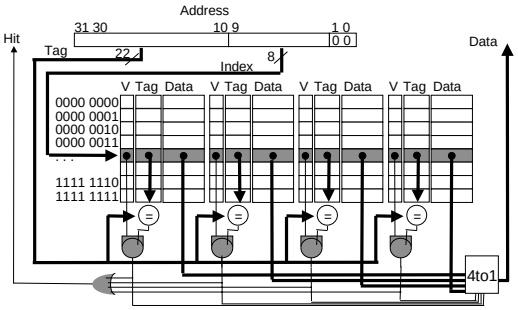
Bytes per Cycle: 4x4/45=0.35 4x4/12=1.33 4x4/15=1.0

Alternative Designs

Possible arrangements for cache elements



Set Associativity



Associativity

- 8-way is pretty much the upper limit except for TLB and memory
- Replacement policy
  - Optimal
  - Least recently used
  - Random
  - Pgm control

Writing vs Reading Cache

- Writing has two basic forms
  - Write through
  - Write back
- Since writing not on critical path, write buffers
- When an element is written, need it be kept in cache?
  - Load on write … especially of block > word