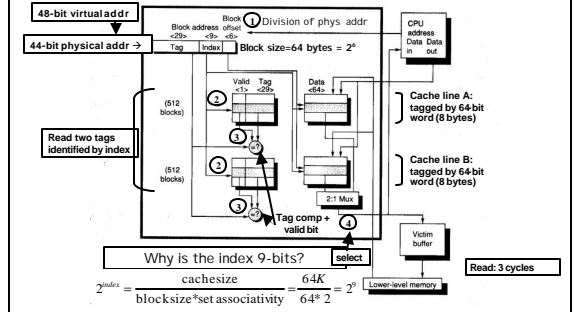


## Example

- Fully associative, write-back cache (empty at start)

	Write allocate	No Write allocate
Memwrite[100]	miss (load 100)	miss
Memwrite[100]	hit	miss
Memread[200]	miss (load 200)	miss (load 200)
Memwrite[200]	hit	hit
Memwrite[100]	hit	miss

## Alpha 21264 Data Cache Hit



## Alpha 21264 Data Cache

- On a miss...
  - Cache signal processor
  - 64 bytes from next lower level
    - ES40 BW ~140MB/s (single transfer 16 bytes in 2.25ns)
    - 9ns for 64 bytes to arrive (@667MHz)
  - FIFO selection for block replacement (1-bit)
  - Write back: use write buffer (8 entries)
  - Write allocation
- Other details
  - 64K instruction cache

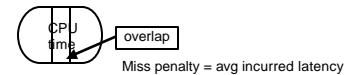
## CPU Time

CPU execution time = (CPU clock cycles + memory stall cycles) x clock cycle

Simple in-order processors:



Modern superscalar processors:



## Cache Measures

- Hit rate: fraction found in that level
  - So high that usually talk about Miss rate
  - Miss rate fallacy: as MIPS to CPU performance, miss rate to average memory access time in memory
- Average memory-access time = Hit time + Miss rate x Miss penalty
- Miss penalty: time to replace a block from lower level, including time to replace in CPU
  - access time: time between read request and when desired word arrives
  - transfer time: time to transfer block
  - cycle time: min time between requests to memory

## Cache Performance

cpu execution time = (cpu clock cycles + memstall cycles) \* clock cycle time

$$\begin{aligned}
 \text{memstall cycles} &= (\# \text{misses} * \text{miss penalty}) \\
 &= IC * \frac{\# \text{misses}}{\text{instruction } n} * \text{miss penalty} \\
 &= IC * \frac{\text{memory accesses}}{\text{instruction } n} * \text{miss rate} * \text{miss penalty}
 \end{aligned}$$

### Example 1

Size	I cache	Dcache	Ucache
16KB	.00382	.0409	.0510
32KB	.00136	.0384	.0433

Single ported unified cache

	Unified cache Data instr	All other instr
Hit time	2	1
Miss time	100	100

miss rate =  $\frac{\# \text{misses}/\# \text{instr}}{\# \text{mem accesses}/\# \text{instr}} = \frac{\# \text{misses}}{\# \text{mem accesses}}$

Split cache

1 mem accesses/1 instr:  $\text{miss rate}_{\text{data}} = \frac{.00382}{1} = .00382$

.36 mem accesses/1 instr:  $\text{miss rate}_{\text{non-data}} = \frac{.0409}{.36} = .1136$

split cache miss rate =  $.74(.00382) + .26(.1136) = .0324$

Unified cache

1+.36 data accesses/1 instr:  $\text{miss rate}_{\text{data}} = \frac{.0433}{1+.36} = .0318$

Split cache miss rate > Unified cache miss rate

Instruction stream: 74% instr refs, 26% data refs

36% data transfers

64% non-data transfers

### Example 1

Size	I cache	Dcache	Ucache
16KB	.00382	.0409	.0510
32KB	.00136	.0384	.0433

AMAT = hit time + missrate \* misspenalty

AMAT =  $\% I(AMAT_I) + \% D(AMAT_D)$

AMAT<sub>split</sub> =  $.74(1 + .00382*100) + .26(1 + .1136*100) = 4.24$

AMAT<sub>unified</sub> =  $.74(1 + .0318*100) + .26(2 + .0318*100) = 4.44$

Split cache AMAT < Unified cache AMAT

[opposite of cache miss rate result (split > unified)]

Instruction stream: 74% instr refs, 26% data refs

36% data transfers

64% non-data transfers

### Example 2

- Calculate CPI with perfect and real cache
- With cache
  - miss rate = 5%
  - dmiss rate = 10%
  - Miss penalty 40 cycles
- Performance with/without cache?

Instruction type	Frequency	Clock cycle count
ALU ops	43%	1
Loads	21%	2
Stores	12%	2
Branches	24%	2

Machine with perfect cache has given distribution behavior.

Introducing a cache implies some misses/hits to instruction and data cache.

Compare performance with and without cache.

Instruction stream: 33% data transfers, 67% non-data transfers

5% miss rate

10% dmiss rate

### Example 2

AMAT =  $\sum_{i=1}^n \%_i (AMAT_i)$

Instruction type	Frequency	Clock cycle count	Instr Accesses	Data Accesses
ALU ops	43%	1	1	0
Loads	21%	2	1	1
Stores	12%	2	1	1
Branches	24%	2	1	0

AMAT<sub>ideal</sub> =  $.43(1) + .21(2) + .12(2) + .24(2) = 1.57 = \text{AMAT}_{\text{perfect}}$

AMAT<sub>real</sub> =  $.43(1 + .05*40) + .21(2 + .05*40 + .10*40) + .12(2 + .05*40 + .10*40) + .24(2 + .05*40) = 4.9$

Type	%	Hit time	Perfect Cache Stall time	Perfect Cache Totl time	Real Cache Stall time	Real Cache Totl time
ALU ops	43%	1	0	1	.05*40=2	3
Loads	21%	2	0	2	.05*40+.10*40=6	8
Stores	12%	2	0	2	.05*40+.10*40=6	8
Branches	24%	2	0	2	.05*40=2	4
		AMAT <sub>ideal</sub> = 1.57	AMAT <sub>perfect</sub> = 1.57	AMAT <sub>real</sub> = 4.9		

## Summary

- CPU-Memory gap is major obstacle for performance, HW and SW
- Take advantage of program behavior: locality
- Time of program still only reliable performance measure
- 4Qs of memory hierarchy
- Lots of formulas (summary pg 412 H&P)