2721

A word-addressable computer has a cache which has 4 lines. Each line holds two words. An executing program reads data in three passes from a set of address sequences as shown here in decimal format:

Pass 1: 0, 1, 2, 3, 4, 5, 6, 7, 8
2: 8, 7, 6, 5, 4, 3, 2, 1, 0
3: 2, 3, 4, 5, 1, 2, 3, 4, 5

Draw the contents (in a table) of the cache at the end of each pass:
(a) for a direct-mapped cache
(b) for a 2-way S-A cache
(c) for a 4-way S-A cache.
LRU algorithm

Calculate the hit rate. Assume the cache is initially empty.

(a) Direct-mapped cache

2 words/line

block0 ✓ block 1 ✓ ✓ ✓

Pass 1: $\boxed{0, 1,}$ $\boxed{2, 3}$ 4, 5, 6, 7, 8

2: ✓8, ✓7, ✓6, ✓5, ✓4, ✓3, ✓2, 1, 0 ✓

3: 2, 3, 4, 5, 1, 2, 3, 4, 5

✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓

hit rate

21/27

	Pass ~~0~~1	2	3
line 0	~~0,1~~ 8,9	~~8,9~~ 0,1	0,1
1	2,3	2,3	2,3
2	4,5	4,5	4,5
3	6,7	6,7	6,7

(b) 2-way S-A

Exam: put in the "top" empty line of its set

Pass 1: 0, 1, 2, 3, 4, 5, 6, 7, 8
✓ ✓ ✓ ✓ ✓

Pass 2: 8, 7, 6, 5, 4, 3, 2, 1, 0
✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓

Pass 3: 2, 3, 4, 5, 1, 2, 3, 4, 5
✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓

LRU

	Pass 1	2	3
Set 0	~~$\boxed{0,1}$ A~~ $\boxed{8,9}$ C	~~$\boxed{8,9}$ B C~~ $\boxed{0,1}$ G [illegible]	$\boxed{0,1}$ ~~B~~ E
	$\boxed{4,5}$ B	$\boxed{4,5}$ ~~A D~~ E	$\boxed{4,5}$ ~~A C D F~~ G
Set 1	$\boxed{2,3}$ A	$\boxed{2,3}$ ~~A E~~ F	$\boxed{2,3}$ ~~B C~~ D ~~E~~ G
	$\boxed{6,7}$ B	$\boxed{6,7}$ ~~B C~~ D	$\boxed{6,7}$ A

$\frac{21}{27}$

(c) 4-way S-A

Pass 1: 0, 1, 2, 3, 4, 5, 6, 7, 8

Pass 2: 8, 7, 6, 5, 4, 3, 2, 1, 0

Pass 3: 2, 3, 4, 5, 1, 2, 3, 4, 5

~~[illegible]~~

21/27

Set	Pass 1	2	3
0	~~0,1~~ ~~A~~B 8,9 I	~~8,9~~ ~~B~~E 0,1 ~~A~~ M	0,1 ~~B~~ I
	2,3 ~~B~~ ~~C~~ D	2,3 ~~A~~ ~~C~~, K	2,3 ~~A~~ G, ~~F~~ ~~J~~ K
	4,5 ~~E~~ ~~G~~, F	4,5 ~~B~~ ~~H~~, I	4,5 B ~~C~~ ~~H~~ ~~L~~, M ~~0,1~~
	6,7 ~~D~~ ~~G~~, H	6,7 ~~C~~ ~~D~~ ~~F~~, G	6,7 A

-5-
-5-

SSD

superscalar arch (to ↓reduce bottlenecks)

cache: use principles of locality

more registers
more buses
S-A cache
avoid real time
more cache
fewer levels
async buses
faster memory
faster buses

simpler instructions
shorter distances
deeper pipeline
HDD access
more HW implementation
↑ BW
↓ latency
~~ECC~~

more cores
more parallelism
more ALUs
faster cpu

Spin HDD faster

Add R1 (R2) R3
R1 R2 R3