154B Discussion 9

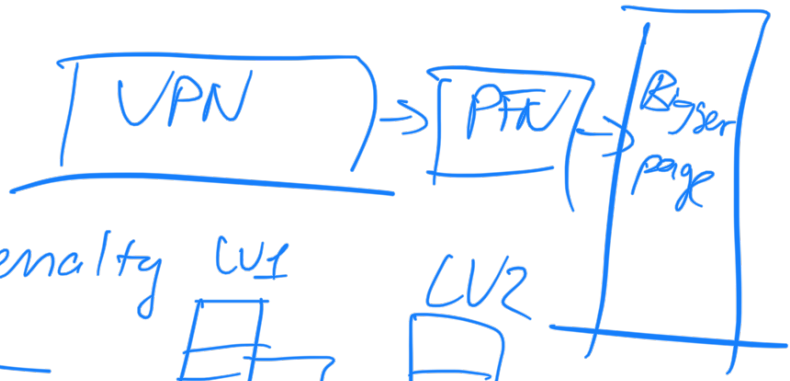
March 11th, 2022

Goals

- Going over practice final.

Q1. Large pages (2MiB) Compared to 4KiB
Huge pages (1GiB)

a. Spatial / temporal



b. No difference in TLB miss penalty if the # of level doesn't change

c. VAddr: 32 bits (suppose this)

[VPN | page-offset]

11 | 21

2^{11} VPNs ← 21 bits ← 2MiB page (2^{21} bytes)
for page offset

2^{20} VPNs ← 12 bits ← 4KiB page
for page offset

20 | 12

Assume that we have a fixed # of slots in TLB (e.g. 96)

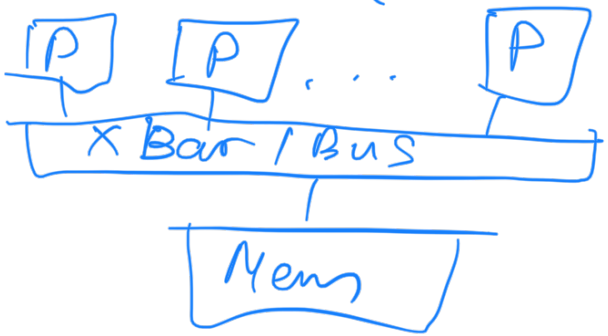
2^{11} VPNs → 2^{11} VPNs competing for 96 slots in TLB
(2 MiB page)

2^{20} VPNs → 2^{20} VPNs competing for 96 slots in TLB
(4 KiB pages)

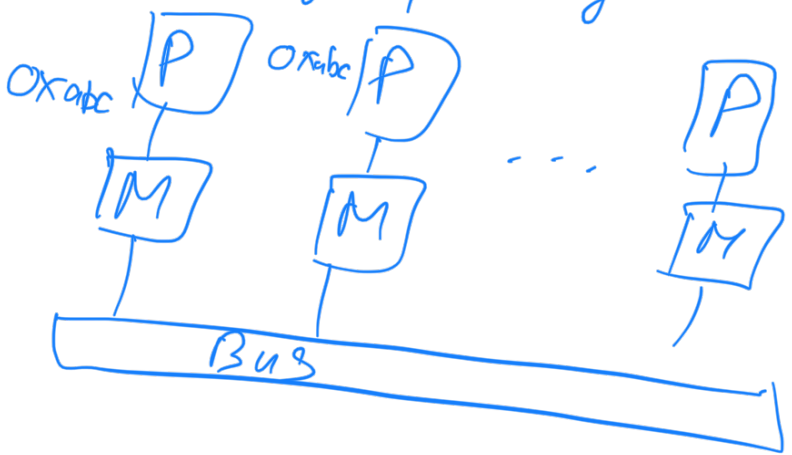
2 MiB pages have lower TLB miss ratio

Q2. Shared memory vs. message passing

a. Shared mem



b. message passing



c.

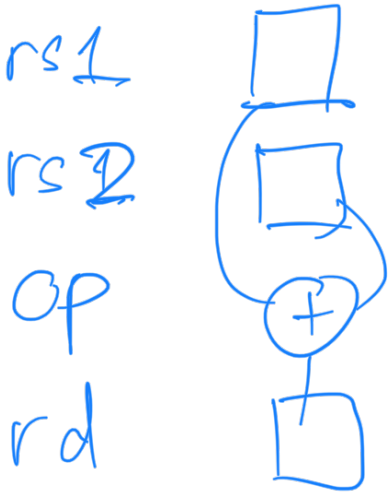
d. Shared memory: prog model: coherency is taken care of by HW

Message passing: // SW

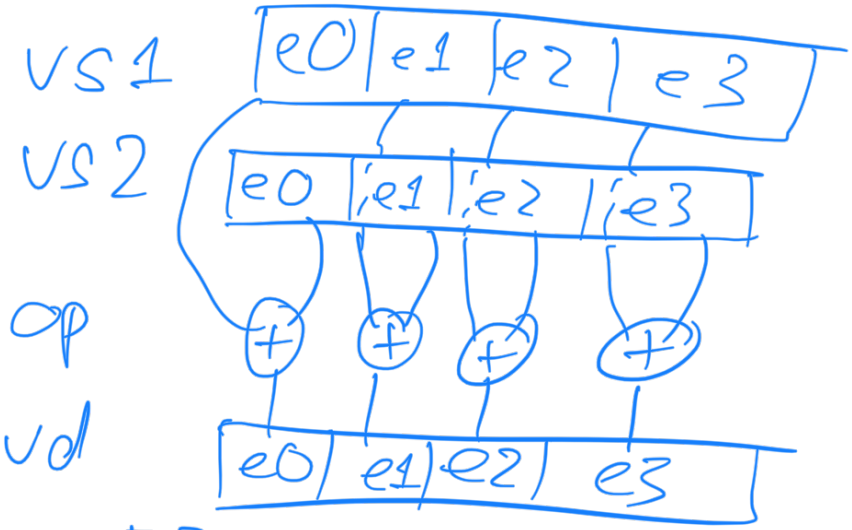
→ shared memory ~~to~~ programming is easier.

Q3. Vector architecture

Scalar register (DINOCPU)



Vector register



~~def~~ a : int[ ] b : int[ ] c : int[ ]

loop: a + b = c

Scalar

```
load  x1, a[i]
load  x2, b[i]
add   x3, x1, x2
store x3, c[i]
```

K times

→ K × 4 dynamic insts

Vector

```
loop:
load  vs1, (a[i], a[i+1], a[i+2], a[i+3])
load  vs2, (b[i], b[i+1], ...)
vadd  vs3, vs1, vs2
store vs3, ...
```

$\frac{K \times 4}{4}$ dynamic insts

Q4: | SISD | MISD |
| --- | --- |
| SIMD | MIMD |

a. SISD arch

b. SIMD arch

c. SISD / SIMD

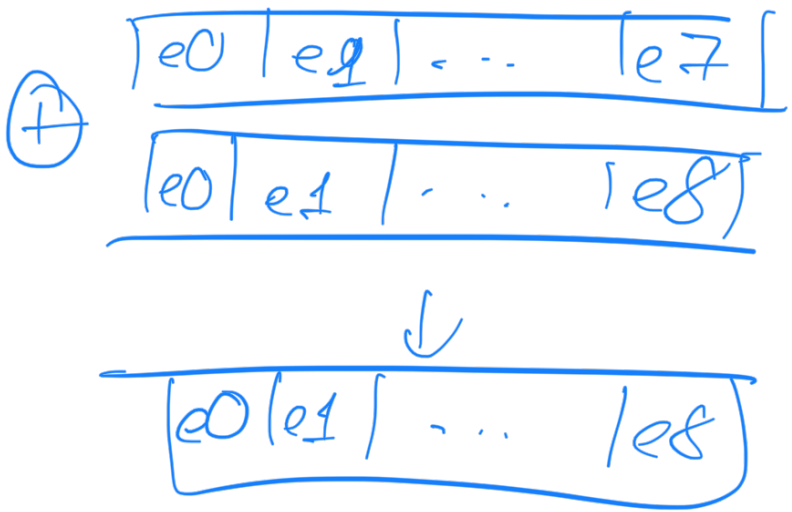
Data-level Parallel (DLP)

easier → SISD: programers (users) don't have to find data parallelism

SIMD: // have to do extra work to find and exploit DLP

Q5. 75% of an application has data parallelism
-> 25% " has to be done serially

a. 8 lanes -> a vector



Amdahl's law: speed up of parallel machines over a serial one

$$\text{Time}_{\text{serial}} = 1 = (0.75 + 0.25)$$

$$\text{Time}_{\text{parallel}} \left(\frac{0.75}{\#\text{lanes or \# cores}} + 0.25 \right)$$

speed up of A compared to B $= \frac{t_B}{t_A}$

$$\frac{t_{serial}}{t_{parallel}} = \frac{1}{\frac{0.75}{8} + 0.25}$$

b. 16 lanes

c. Speed up application on 16 lanes compared to 4 lanes.

$$\frac{t_{4\,lanes}}{t_{16\,lanes}} = \frac{\frac{0.75}{4} + 0.25}{\frac{0.75}{16} + 0.25}$$

d. Maximum speed up.

$$\frac{1}{0.25 + \frac{0.75}{N}} \to \frac{1}{0.25} = 4x$$

N: very big

3.9x ⇒ 117 lanes

3.99x ⇒ 1197 lanes

3.999x ⇒ 11997 lanes

We cannot get 4x b/c cost / or no N could get $\frac{0.75}{N} = 0$

Q6. Effecient

-> Performance : SIMD is faster when there's a lot of DLP

-> energy : less decoding / fetching

Q7. True.

Q8. scalar vs. multi-threaded

HW thread ≠ SW thread

↓ different execution context

↓ ~~can~~ share mem space

a. SISD application. → inst stream → SMT (2+ HW threads) → ALU (1) → 2+

The performance is the same

b. MIMD. Multi-threaded is better

c. Many different applications → ~~Through~~ Multi-threaded produces better thruput overall (individual app can have higher latency)

Q9. VAddr 40 bits 4KiB page size TLB: VPN → PFN

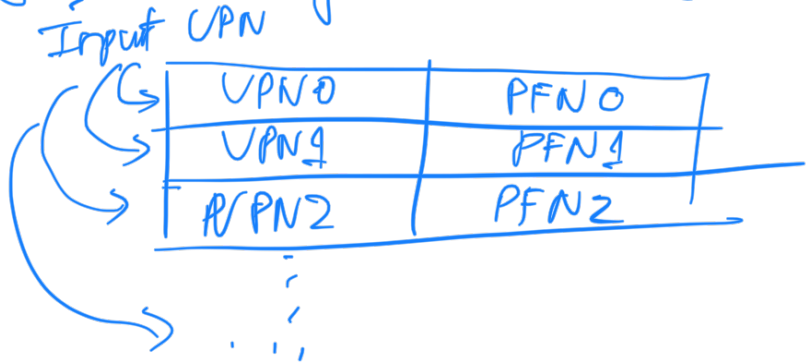
↳ 2^{12} bytes → 12 bits for page offset

a. VAddr: [VPN | page offset]

39 12 11 0

28 bits 12 bits

b. Fully associative TLB (cache)



~~VPN~~

Tag: bits (39, 12)

d. 16 entries, 8-way set associative

16 entries
8-way

[TAG | INDEX | offset]
39 13 12 12 11 0

[VPN | page-offset]
39 12 11 0

Index → which row ?
Index bits are (12, 12)

2^1 rows → 1 bit for index