ECE 571 – Advanced Microprocessor-Based Design Lecture 9

Vince Weaver

http://web.eece.maine.edu/~vweaver

vincent.weaver@maine.edu

20 February 2018

Announcements

- HW#4 was posted. About branch predictors
- Don't wait until last minute, is a bit more complex. Log into two machines. Any trouble logging into Jetson?

HW#3 Review

		sleep	stream	matrix	iozone
Energy	cores	0.08	94.59	370.0	3.64
	gpu	0.00	0.00	0.00	0.00
	pkg	42.97	162.70	406.59	9.31
	ram	15.84	33.40	23.09	1.28
	time	10.0	6.22	5.2	0.74

RAM a lot higher, IOzone a lot faster than last year. Why? More RAM: 16GB vs 4GB, and SSD/PCIe drive rather than SATA

		sleep	stream	matrix	iozone
Power	cores	0.01	15.2	71.2	4.9
	gpu	0.00	0.00	0.00	0.00
	pkg	4.30	26.2	78.2	12.6
	ram	1.58	5.4	4.4	1.7

Energy-delay

	1	2	3	4	5	8	16
E	2625	2096	3092	3416	3508	3514	3542
time	140	95	83	80	85	79	84
ED	368k	199k	257k	273k	298k	277k	298k
ED2	51M	19M	21M	22M	25M	22M	25M
P	19W	22W	37W	43W	41W	44W	42W

- Does have a GPU. Does seem suspect at 0, but in headless mode. On a similar haswell machine have gotten the GPU to show things but running OpenCL as well as KSP.
- Energy-delay-squared is E*t*t.
  Not E*E*t, or (E*t)*(E*t)
- Could we show weak scaling? No, problem size is staying same.
- Why did it stop scaling at 4?
  Poorly written benchmark (possible) Not enough

memory (not really, this is a 2001 benchmark) Because even though it has 8 threads, only has 4 cores (likely) How could you test that? (find machine with more cores and run it)

Do this on Haswell-EP?

haswell-EP – note 2 sockets (perf adds them together)

16 cores (32 threads) 80 GB of RAM

How is NUMA set up? Does OS group cores together, threads together?

	1	2	4	8	16	32	64
E	10922	8708	5689	4709	5139	5162	7609
time	199	131	62	41	39	37	78
ED	2.2M	1.1M	352k	193k	200k	191k	593k
ED2	432M	149M	21M	7.9M	7.8M	7.1M	46M
P	55W	66W	91W	114W	131W	143W	97W

Virtual vs Physical Addressing

Programs operate on Virtual addresses.

- PIPT, PIVT (Physical Index, Physical/Virt Tagged) – easiest but requires TLB lookup to translate in critical path

- VIPT, VIVT (Virtual Index, Physical/Virt Tagged) – No need for TLB lookup, but can have aliasing between processes. Can use page coloring, OS support, or ASID (address space id) to keep things separate

Oh No, More Caches!

Cache Miss Types

- Compulsory (Cold) — miss because first time seen
- Capacity — wouldn't have been a miss with larger cache
- Conflict — miss caused by conflict with another address (would not have been miss with fully assoc cache)
- Coherence — miss caused by other processor

Fixing Compulsory Misses

Prefetching

- Hardware Prefetchers – very good on modern machines. Automatically bring in nearby cachelines.
- Software – loading values before needed
  also special instructions available
- Large-blocksize of caches. A load brings in all nearby values in the rest of the block.

Fixing Capacity Misses

- Build Bigger Caches

Fixing Conflict Misses

- More Ways in Cache
- Victim Cache
- Code/Variable Alignment, Cache Conscious Data Placement

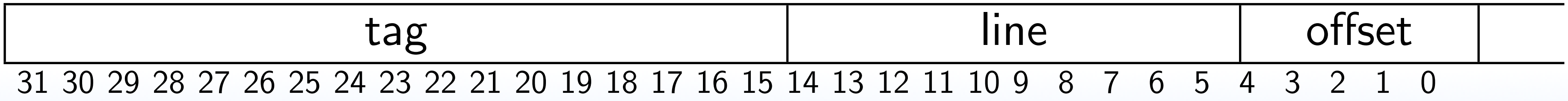
Fixing Coherence Misses

- False Sharing – independent values in a cache line being accessed by multiple cores

Cache Parameters Example 1

32kB cache (2^{15}), direct mapped (2^{0})
32 Byte linesize (2^{5}), 32-bit address size (2^{32})

offset $= log_2(linesize) =$ 5 bits
lines $= log_2((cachesize/\#ways)/linesize) =$ 1024 lines (10 bits)
tag $=$ addresssize - (offset bits $+$ line bits) $=$ 17 bits



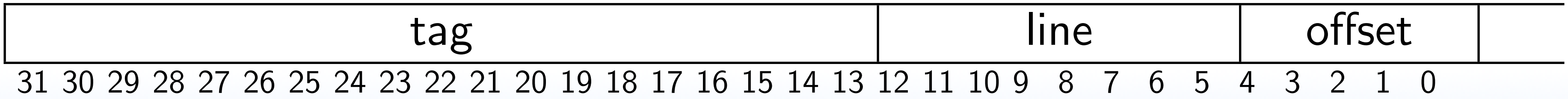
tag | line | offset

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

Cache Parameters Example 2

32kB cache (2^{15}), 4-way (2^2)
32 Byte linesize (2^5), 32-bit address size (2^{32})

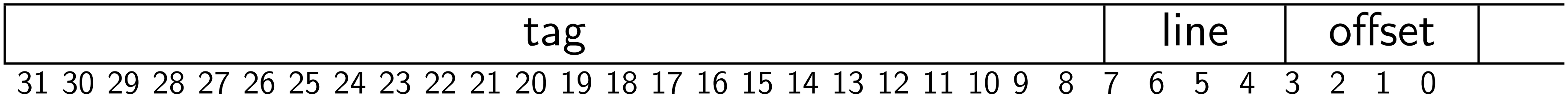
offset $= log_2(linesize) =$ 5 bits
lines $= log_2((cachesize/\#ways)/linesize) =$ 256 lines (8 bits)
tag $=$ addresssize - (offset bits $+$ line bits) $=$ 19 bits



Cache Example

512 Byte cache, 2-Way Set Associative, with 16 byte lines, LRU replacement.

24-bit tag, 16 lines (4 bits), 4-bit offset.



tag | line | offset

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

Cache Example 1

Cache Example – Instruction 1

```
ldb r1, 0x00000000
```

	Way 0				Way 1			
line	V	D	LRU	Tag	V	D	LRU	Tag
0	1	0	0	0000 00	0			
1	0				0			
2	0				0			
3	0				0			
4	0				0			
5	0				0			
. . .								
b	0				0			
c	0				0			
d	0				0			
e	0				0			
f	0				0			

Miss, Cold

Cache Example – Instruction 2

```
ldb r1, 0x00000001
```

	Way 0				Way 1			
line	V	D	LRU	Tag	V	D	LRU	Tag
0	1	0	0	0000 00	0			
1	0				0			
2	0				0			
3	0				0			
4	0				0			
5	0				0			
. . .								
b	0				0			
c	0				0			
d	0				0			
e	0				0			
f	0				0			

Hit

Cache Example – Instruction 3

```
ldb r1, 0x00000010
```

	Way 0				Way 1			
line	V	D	LRU	Tag	V	D	LRU	Tag
0	1	0	0	0000 00	0			
1	1	0	0	0000 00	0			
2	0				0			
3	0				0			
4	0				0			
5	0				0			
. . .								
b	0				0			
c	0				0			
d	0				0			
e	0				0			
f	0				0			

Miss, Cold

Cache Example – Instruction 4

```
ldb r1, 0x80000010
```

	Way 0				Way 1			
line	V	D	LRU	Tag	V	D	LRU	Tag
0	1	0	0	0000 00	0			
1	1	0	1	0000 00	1	0	0	8000 00
2	0				0			
3	0				0			
4	0				0			
5	0				0			
. . .								
b	0				0			
c	0				0			
d	0				0			
e	0				0			
f	0				0			

Miss, Cold

Cache Example – Instruction 5

```
ldb r1, 0xC0000010
```

	Way 0				Way 1			
line	V	D	LRU	Tag	V	D	LRU	Tag
0	1	0	0	0000 00	0			
1	1	0	0	C000 00	1	0	1	8000 00
2	0				0			
3	0				0			
4	0				0			
5	0				0			
. . .								
b	0				0			
c	0				0			
d	0				0			
e	0				0			
f	0				0			

Miss, Cold (never in cache previously)

Cache Example – Instruction 6

```
ldb r1, 0xC0000002
```

	Way 0				Way 1			
line	V	D	LRU	Tag	V	D	LRU	Tag
0	1	0	1	0000 00	1	0	0	c000 00
1	1	0	0	C000 00	1	0	1	8000 00
2	0				0			
3	0				0			
4	0				0			
5	0				0			
. . .								
b	0				0			
c	0				0			
d	0				0			
e	0				0			
f	0				0			

Miss, Cold

Cache Example – Instruction 7

```
ldb r1, 0x00000010
```

	Way 0				Way 1			
line	V	D	LRU	Tag	V	D	LRU	Tag
0	1	0	1	0000 00	1	0	0	c000 00
1	1	0	1	C000 00	1	0	0	0000 00
2	0				0			
3	0				0			
4	0				0			
5	0				0			
. . .								
b	0				0			
c	0				0			
d	0				0			
e	0				0			
f	0				0			

Miss, Conflict

Cache Example – Instruction 8

```
stb r1, 0x00000005
```

	Way 0				Way 1			
line	V	D	LRU	Tag	V	D	LRU	Tag
0	1	1	0	0000 00	1	0	1	c000 00
1	1	0	1	C000 00	1	0	0	0000 00
2	0				0			
3	0				0			
4	0				0			
5	0				0			
. . .								
b	0				0			
c	0				0			
d	0				0			
e	0				0			
f	0				0			

Hit

Capacity vs Conflict Miss

- It's hard to tell on the fly what kind of miss
- For example: to know if cold, need to keep list of every address that's ever been in cache
- To know if it's capacity, need to know if it would have missed even in a fully associative cache
- Otherwise, it's a conflict miss