CMPE 110: Computer Architecture

Week 10
Final Review

Jishen Zhao (http://users.soe.ucsc.edu/~jzhao/)

Reminders

- Final exam
  - Dec. 7 (Wed) 12-3pm
  - DRC exam rooms
  - Overflow room (TBD)
  - This classroom (can fit rest of you)

Summary: covered in midterms

- Performance
  - Latency vs. throughput
  - IPC, CPI, and speedup
  - Amdahl’s law
- ISA
  - Program execution model:
    high-level language->assembly->machine language
  - Instruction execution model
    The loop of fetch, decode, read input, execution, write output, next instruction
- Pipelining
  - Pipeline data path
  - Hazards
  - Out-of-order execution

Summary: to be covered in final

- Caching
  - Given an address, how to access cache?
  - Calculate AMAT
  - Direct-mapped cache, associative cache, fully-associative cache
- Main memory
  - Main memory organization
- Virtual memory
  - How to access memory with single-level page table?
- Multicore
  - Cache coherence (VI, MSI, MESI)
  - Memory consistency (sequential consistency)
- GPU
  - GPU processor design: three major ideas, calculate GFLOPS
  - GPU memory: three major ideas

Computer Engineering 110: Computer Architecture

Final Examination

Fall 2016

Name: ____

Email: ____

1	20	
2	20	
3	25	
4	15	
5	20	
6	20	
Total	120	

This exam is closed book and closed notes. Personal calculators (four-function calculators only) *are* allowed. Show your work on the attached sheets (front and back) and insert your answer in the space(s) provided. **Please provide details on how you reach a result.** Ask for extra paper sheets if necessary.

You have 180 minutes to complete the exam. This exam is worth 120 points. This midterm exam counts for 30% of your course grade.

Example Questions

1. Calculate CPI with cache misses

- Simple pipeline with base CPI=1
- Instruction mix: 30% loads/stores
- I$: $\%_{miss}$ = 2%, t_{miss} = 10 cycles
- D$: $\%_{miss}$ = 10%, t_{miss} = 10 cycles

- What is CPI?

1. Calculate CPI with cache misses (cont.)

- Simple pipeline with base CPI=1
- Instruction mix: 30% loads/stores
- I$: $\%_{miss}$ = 2%, t_{miss} = 10 cycles
- D$: $\%_{miss}$ = 10%, t_{miss} = 10 cycles

  $t_{avg} = t_{hit} + \%_{miss} * t_{miss}$

- What is CPI?
  - $CPI = CPI_{base} + (\%_{cache\_access} \%_{miss} * CPI_{miss})$
  - CPI = 1 + (100%*2%*10 cycles/insn) + (30%*10%*10 cycles/insn)

    = 1 + 0.2 + 0.3

    = 1.5

2. Cache coherence

Given

- Dual-core processor
- write-back caches
- private L1 data caches (L1Ds) (no L2)
- shared memory
- MESI coherence policy, the initial state of all cache blocks is invalid (I)
- A sequence of loads and stores (next slide)

Questions:

1) Indicate in the cache tables what would be

- (1)the values (V) of the variables X and Y and
- (2) their coherency (C) state (either M, E, S or I) in the L1D's with the following sequence of reads and writes

2) What would be the values (V) in the shared memory?

2. Cache coherence (cont.)

- Step 0: Initially, X =5, Y = 6 in the shared memory, processor caches are empty
- Step 1: Core 1 reads X (from the shared memory)
- Step 2: Core 2 reads X
- Step 3: Core 1 writes X = 3
- Step 4: Core 1 writes Y = 4
- Step 5: Core 2 reads Y

Solution

Core 1's L1D Cache (private)												
	Step 0		Step 1		Step 2		Step 3		Step 4		Step 5	
	V	C	V	C	V	C	V	C	V	C	V	C
X	--	I	5	E	5	S	3	M	3	M	3	M
Y	--	I	--	I	--	I	--	I	4	M	4	S

Core 2's L1D Cache (private)												
	Step 0		Step 1		Step 2		Step 3		Step 4		Step 5	
	V	C	V	C	V	C	V	C	V	C	V	C
X	--	I	--	I	5	S	5	I	5	I	5	I
Y	--	I	--	I	--	I	--	I	--	I	4	S

Solution (cont.)

Shared Memory						
	Step 0	Step1	Step2	Step3	Step4	Step5
	V	V	V	V	V	V
X	5	5	5	5	5	5
Y	6	6	6	6	6	4

Hint: review the MSI protocol example in lecture notes on multicore.

*Note: In step3, X can also be changed to 3.
This is not required in final exam (not covered in lecture notes).

3. Sequential Consistency

Thread 1 (on P1)

```
A = 1;
if (B == 0)
  print “Hello”;
```

Thread 2 (on P2)

```
B = 1;
if (A == 0)
  print “World”;
```

- Assume:
  - A and B are initialized to 0, writes propagate immediately
  - Sequential consistency
- **Question: Imagine threads 1 and 2 are being run simultaneously on a two processor system. What will get printed?**

3. Sequential Consistency (cont.)

Thread 1 (on P1)	Thread 2 (on P2)

`A = 1;`
`if (B == 0)`
`  print “Hello”;`
	`B = 1;`
	`if (A == 0)`
	`  print “World”;`

- Sequential consistency =>
  - P1: the write to A has to complete before the read of B can begin
  - P2: the write to B has to complete before the read of A can begin

- Answer: Code will either print “hello” or “world” or nothing, but not both.

Example instruction sequences

Thread 1

Thread 2

1. Hello gets printed

A = 1

If (B==0) print "Hello"

B= 1

If (A==0) print "World"

2. World gets printed

B = 1

If (A==0) print "World"

A= 1

If (B==0) print "Hello"

3. Nothing gets printed

A=1

B=1

If (A==0) print "World"

If (B==0) print "Hello"

4. Nothing gets printed

B=1

A=1

If (A==0) print "World"

If (B==0) print "Hello"

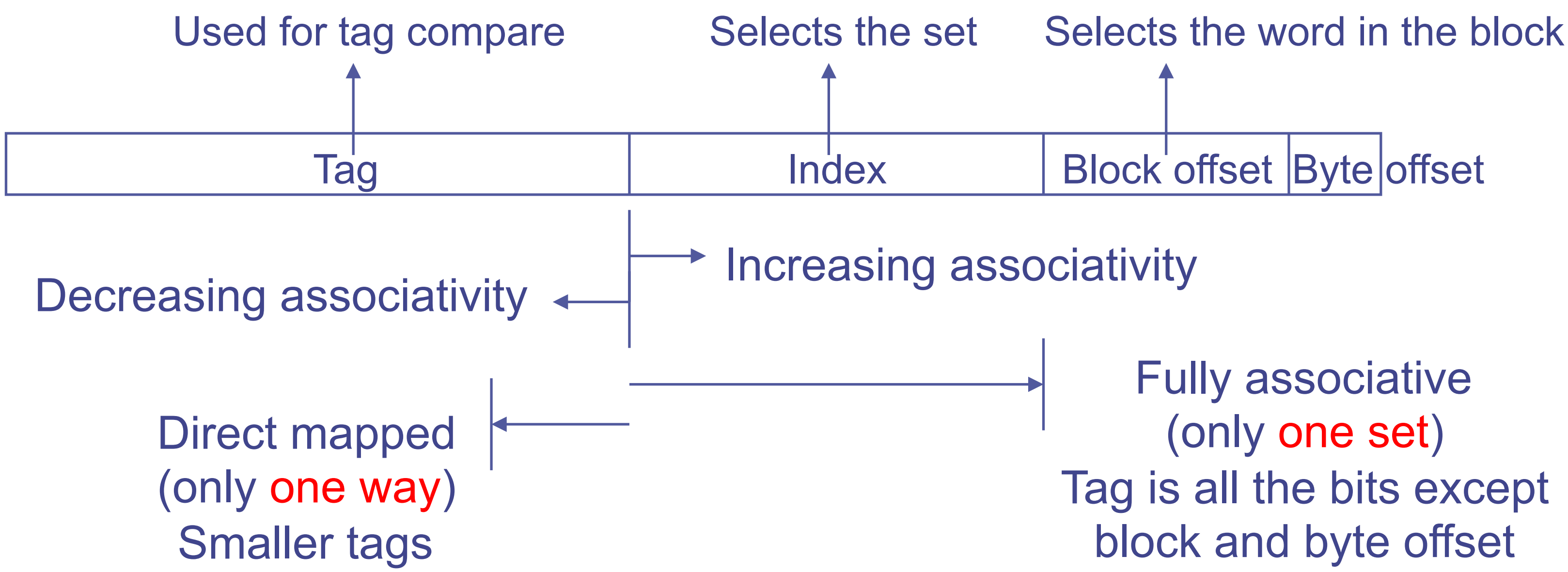
Caches

- What is write-back cache?
  - Answer: Allow cache and memory to be inconsistent, i.e., write the data only into the cache block.

- What is write-through cache?
  - Answer: Require cache and memory to be consistent, i.e., always write the data into both the cache block and the next level in the memory hierarchy.

Caches (cont.)

- For a fixed size cache, each increase by a factor of two in associativity doubles the number of blocks per set (i.e., the number or ways) and halves the number of sets – decreases the size of the index by 1 bit and increases the size of the tag by 1 bit

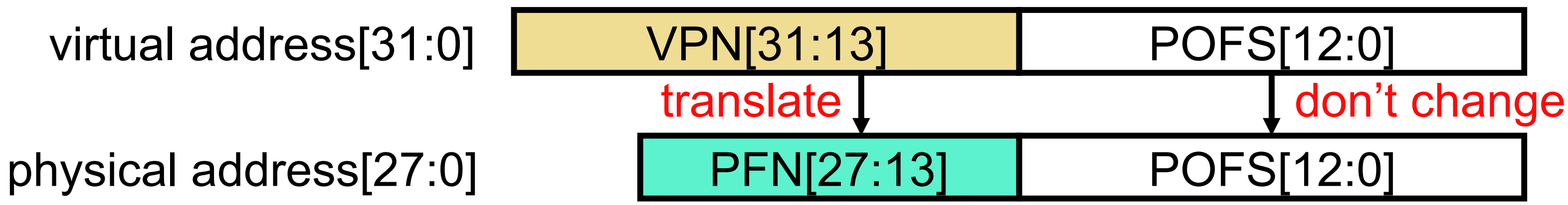


GPUs

- What are the three basic ideas of GPU processor architecture design?
  - (No answer provided.)

- Assume a hypothetical GPU with the following characteristics: Clock rate 1.5 GHz. Contains 16 SIMD processors, each containing 16 single-precision floating-point units, each instruction performs one single-precision floating-point operation. What is the peak single-precision floating-point throughput for this GPU in GFLOP/sec, assuming that all memory latencies can be hidden?
  - (No answer provided.)

Virtual Memory



- Given the above virtual to physical address mapping,
  - What is the page size?
  - How many virtual pages can we have in maximum?
  - How many physical pages can we have in maximum?

  - (No answer provided.)