PennState
College of Engineering

CMPEN431
SAMPLE QUESTIONS

CMPEN 431

Fall 2017

Huaipan Jiang

Question1

Give the performance equation (ignoring memory system effects) discussed in the class.

Execution time= Instruction count * CPI *clock cycle time

Question2

a. Consider the following loop instruction sequence:

```
Loop:   add $3, $3, $2
        lw $4, -100($3)
        beq $3, $4, Loop
```

Suppose this loop executes exactly 3 times (iterations). Further assume that we have 5 execution stages, Instruction fetch, reading resource from register file, performance and ALU computation, reading or writing memory, storing data back to the register file and that the clock times for these stages are 4ns, 1ns, 2ns, 4ns, 1ns, in that order. What is the CPI and CCT of the 3 iteration loop in a *single cycle* machine?

b. In part (a), what is the CPI and CCT in 3 iterations in a *multi cycle* machine?

a. CCT = 4+1+2+4+1 = 12ns
   CPI = 1

b. CCT = 4ns
   CPI = 4

Question3.

Consider the following code sequence:
1. add $3,$1,$2
2. lw $1,0($4)
3. and $5,$3,$4
4. and $6,$1,$2
5. or $1,$3,$6
6. sw $1,4($4)
7. lw $2,4($4)
8. sub $3,$5,$6

Assume that the pipelined datapath has NO forwarding. Find ALL the register hazards in the following code. Also, for each hazard that you find, classify the hazard into one of the following three types:

Type-1: The write register of the instruction in the EXECUTION stage is the same as the read register of the instruction in the INSTRUCTION DECODE stage.

Type-2: The write register of the instruction in the MEMORY stage is the same as the read register of the instruction in the INSTRUCTION DECODE stage.

Type-3: The write register of the instruction in the WRITE-BACK stage is the same as the read register of the instruction in the INSTRUCTION DECODE stage.

Question3.

Instruction 1 #	Instruction 2 #	Register	Hazard type
1	3	$3	2
2	4	$1	3
4	5	$6	1
5	6	$1	1

PennState
College of Engineering

Question4.

Consider the following high level language code segment:

```
int array[1000] = { /* random values */ };
int sum1 = 0, sum2 = 0, sum3 = 0, sum4 = 0;
for (i = 0; i < 1000; i ++) // LOOP BRANCH
{
  if (i % 4 == 0) // IF CONDITION 1
  sum1 += array[i]; // TAKEN PATH
  else sum2 += array[i]; // NOT-TAKEN PATH
  if (i % 2 == 0) // IF CONDITION 2
  sum3 += array[i]; // TAKEN PATH
  else sum4 += array[i]; // NOT-TAKEN PATH
}
```

Your task is to find the prediction accuracy for the LOOP BRANCH (which is taken whenever the loop repeats, and not taken when the loop exits) and both of the IF CONDITION branches inside the loop (which are taken when the if-condition is true, and not taken when the if-condition is false), for different kinds of branch predictors. Show all your work for full credit.

Question4 a.

a. (5 pts) What is the prediction accuracy for each individual branch when using a per-branch last-time predictor (i.e., a one-bit predictor), assuming that every per-branch counter starts at “not-taken”?

loop branch
998/1000 = 99.8%. The branch is mispredicted the first time & the last time it’s executed.
if condition 1
500/1000 × 100 = 50%
if condition 2
0%. The branch changes direction every time it’s executed.

Question4 b.

b. (5 pts) What is the prediction accuracy for each individual branch when a per-branch 2-bit predictor is used, assuming that every per-branch counter starts at “strongly not-taken”?

loop branch
997/1000 × 100 = 99.7%. The branch is mispredicted the first two times it’s executed and the last time (when the loop exits).
if condition 1
750/1000 × 100 = 75%. The branch repeats the pattern T N N N T N N N ... The saturating counter moves between “strongly not-taken” and “weakly not-taken” (once out of every four predictions, after the branch is actually taken), and the prediction is always not-taken.
if condition 2
500/1000 × 100 = 50%. The branch repeats the pattern T N T N ... The saturating counter moves between “strongly not-taken” and “weakly not-taken” every prediction, and every prediction is not-taken, which is correct half the time.

Question4 c.

c. (5 pts) One way of improving the accuracy of branch prediction is to exploit the correlation between different branches. Explain, using the code sequence above, how such correlation can be exploited to improve the performance of a branch predictor.

If the branch of “IF CONDITION 1” is taken, the branch of “IF CONDITION 2” must be taken.
Because if i can be divided by 4, it should be divided by 2.

Question5

1. (36 points) Consider the following three processors (X, Y, and Z) that are all of varying areas. Assume that the single-thread performance of a core increases with the square root of its area.

Processor X: Core Area = A
Processor Y: Core Area = 4A
Processor Z: Core Area = 16A

(a) You are given a workload where S fraction of the work is serial and (1 – S) fraction of the work is infinitely parallelizable. If executed on a die composed of 16 Processor X's, what value of S would give a speedup of 4 over the performance of the workload on just Processor X?

$$S + \frac{1-S}{16} = \frac{1}{4}$$

$$1 + 15S = 4$$

$$15S = 3$$

$$S = 20\%$$

(b) Given a homogeneous die of area 16A, which of the three processors (X, Y, or Z) would you use on your die to achieve maximal speedup? What is that speedup over just a single Processor X? Assume the same

$1+15S = 4$

$15S = 3$

$S = 20\%$

Question5

(b) Given a homogeneous die of area 16A, which of the three processors (X, Y, or Z) would you use on your die to achieve maximal speedup? What is that speedup over just a single Processor X? Assume the same workload as in part (a).

X: $S + \frac{1-S}{16} = \frac{1}{16} + \frac{15}{16}S$ if $S = 20\%$ speedup $= 4$

Y: $\frac{S}{2} + \frac{1-S}{8} = \frac{1}{8} + \frac{3}{8}S$ if $S = 20\%$ speedup $= 5$ ✓

Z: $\frac{S}{4} + \frac{1-S}{4} = \frac{1}{4}$ Speedup $= 4$

(c) Now you are given a heterogeneous processor of area 16A to run the above workload. The die consists of 1 Processor Y and 12 Processor X's. When running the workload, all sequential parts of the program will be run on the larger core while all parallel parts of the program run exclusively on the smaller cores. What is the overall speedup achieved over a single Processor X?

$\frac{S}{2} + \frac{1-S}{12} = \frac{1}{12} + \frac{5S}{12}$

PennState
College of Engineering

$\frac{S}{2} + \frac{1-S}{8} = \frac{1}{8} + \frac{S}{8}$ if $S=20\%$, speedup = 5

Question5

$\frac{S}{4} + \frac{1-S}{4} = \frac{1}{4}$ Speedup = 4

(c) Now you are given a heterogeneous processor of area 16A to run the above workload. The die consists of 1 Processor Y and 12 Processor X's. When running the workload, all sequential parts of the program will be run on the larger core while all parallel parts of the program run exclusively on the smaller cores. What is the overall speedup achieved over a single Processor X?

$$\frac{S}{2} + \frac{1-S}{12} = \frac{1}{12} + \frac{5S}{12}$$

if $S=20\%$

$$\text{Speedup} = \frac{1}{\frac{1}{12} + \frac{5\times 0.2}{12}} = \frac{6}{1} = 6.$$

PennState
College of Engineering

Question5

(d) One programmer optimizes the given workload so that it has 4% of its work in serial sections and 96% of its work in parallel sections. Which configuration would you use to run the workload if given the choices between the processors from part (a), part (b), and part (c)?

(a) $\frac{4\%}{1} + \frac{96\%}{16} = 4\% + 6\% = 10\%$ speedup = 10. ✓

(b) Y: $\frac{4\%}{2} + \frac{96\%}{8} = 2\% + 12\% = 14\%$ speedup = 7.14

Z: $\frac{4\%}{4} + \frac{96\%}{4} = 25\%$ speedup = 4.

(c) $\frac{4\%}{2} + \frac{96\%}{12} = 2\% + 8\% = 10\%$ speedup = 10. ✓

(e) What value of S would warrant the use of Processor Z over the configuration in part (c)?

$Z: S + \frac{1-S}{4} = \frac{1}{4}$ speedup = 4

Z: $\frac{4\%}{4} + \frac{96\%}{4} = 25\%$

(C) $\frac{4\%}{2} + \frac{96\%}{12} = 2\% + 8\% = 10\%$ speedup = 10 ✓

Question5

(e) What value of S would warrant the use of Processor Z over the configuration in part (c)?

Z: $S + \frac{1-S}{4} = \frac{1}{4}$ speedup = 4

C: $\frac{1}{12} + \frac{5}{12}S > \frac{1}{4}$

$\frac{5}{12}S > \frac{1}{6}$

$5S > 2$

$S > 40\%$

(f) Typically, for a realistic workload, the parallel fraction is not infinitely parallelizable. What are the three fundamental reasons why?

i) Sychronization,

ii) number of codes are not infinite.

iii) limit of hardware.

PennState
College of Engineering

PennState
College of Engineering

2. (36 points)

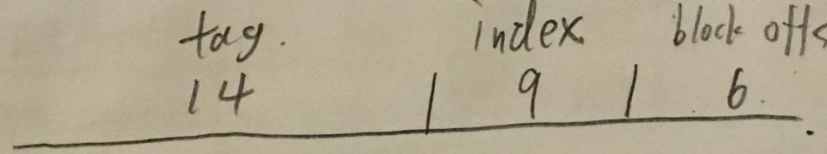
Question6

Consider the complete memory hierarchy. You have a paged memory system. The processor has 512 Mbytes of memory and an 8 GB virtual address space with 4 Kbyte pages. The L1 cache is a 64 KB, 2-way set associative with 64 byte lines. The L2 cache is a 1 MB, 8-way set associative with 256 byte lines. Address translation is supported by a 4 entry TLB. Each page table entry is 32 bits.

29 bits

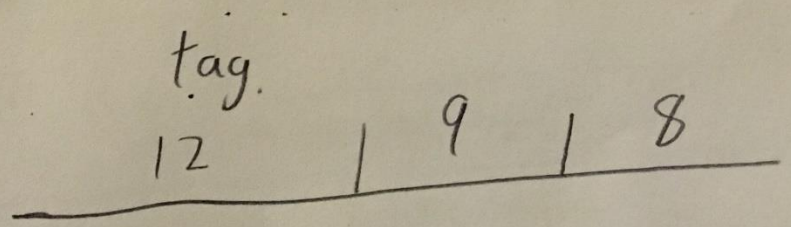
(a) Show the breakdown of the fields of virtual address, physical address, the physical address as interpreted by the L1 cache, and the physical address as interpreted by the L2 cache. Clearly mark the number of bits in each address and the number of bits in each field.

L1:



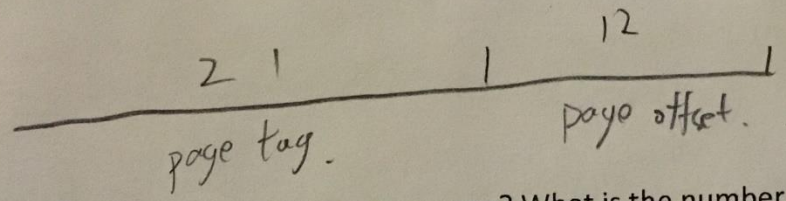
64KB / 2 / 64byte = 512 sets

L2:



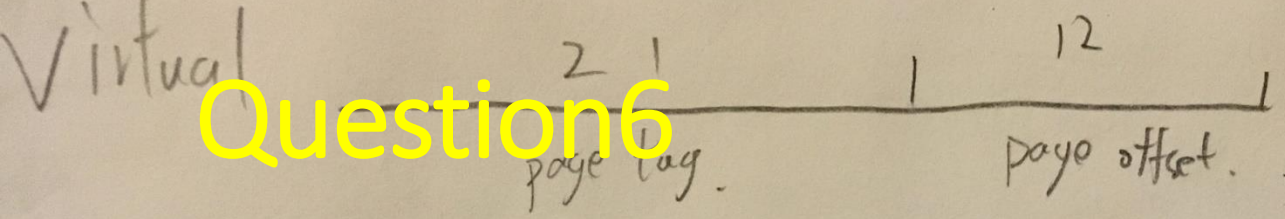
1MB / 8 / 256byte = 512 sets.

Virtual



(b) What is the number of L2 cache lines per page? What is the number of entries in the page table? What is the page table size in pages?

Question6



(b) What is the number of L2 cache lines per page? What is the number of entries in the page table? What is the page table size in pages?

① 4 kB / 256 byte. = 16.

② 512 MB / 4 kB = 128 × 1024 entry.

③ (1024 × 128.) × 32 bits. = 1024 × 128 × 4 Bytes
= 512 KB.

512 KB / 4 KB = 128 pages.

Question6

(c) If a program has a block access pattern (in decimal) :
0, 512, 4096, 1024, 1536, 0, 512, 2048, 2560, 3072, 3584, 4096, 1024, 3072, 4096, 3584, 2048,
indicate, for each access, whether it is an L1 hit/miss or L2 hit/miss (if it is an L1 miss).

	L1	L2
0	m, 0-63 → set 0.	m, 0-255 → set 0.
512	m, 512-575 → set 8	m, 512-767 → set 2
4096	m, 4096-4159 → set 64	m, 4096-4351 → set 16
1536	m, 1536-1599 → set 24	m, 1536-1791 → set 6.
0	hit.	
512.	hit.	
2048.	m, 2048-2111 → set 32	m, 2048-2303 → set 8.
2560	m, 2560-2623 → set 40	m, 2560-2815 → set 10.
3072	m, 3072-3135 → set 48	m, 3072-3327 → set 12.
3584	m, 3584-3647 → set 56	m, 3584-3839 → set 14
4096.	hit.	

	L1	L2
1024	m, 1024-1087 → set 16	m, 1024-1279 → set 4
3072	hit	
4096	hit	
3584	hit.	
2048	hit.	

(d) Is it possible for this system to address the L1 cache concurrently with the address translation provided by the TLB? Justify your answer using the address breakdowns shown in part (a). What would be the benefit of doing so?

64 bytes block

2560	m, 2560-2623 → set40	m, 2560-2815 → set10.
3072	m, 3072-3135 → set48	m, 3072-3327 → set12.
3584	[illegible]	m, 3584-3839 → set14
4096.	hit.	

Question6

(d) Is it possible for this system to address the L1 cache concurrently with the address translation provided by the TLB? Justify your answer using the address breakdowns shown in part (a). What would be the benefit of doing so?

64 bytes block

No, because block offset need 6 bits, index needs 9 bits (512 sets). totally 15 bits.

So, We need last 15 bits for addressing L1 cache. but page size is 4KB(12 bits), we can only get 12 bits of pysical address (PA).

If we can do so, we can access L1 cache without computing the PA. (if it is a L1 hit).

PennState
College of Engineering