CPSC 313, Winter 2016 Term 1 — Midterm

Date: October 24, 2016; Instructor: Mike Feeley

Q1	/ 8
Q2	/ 7
Q3	/ 8
Q4	/ 9
Q5	/ 8
Q6	/ 4
Q7	/ 6

This is a closed book exam. No notes or electronic calculators. The last page contains the ISA description; for convenience you may remove this page.

Answer in the space provided. Show your work; use the backs of pages if needed. There are **7** questions on **4** pages, totaling **50** marks. You have **50 minutes** to complete the exam.

STUDENT NUMBER: _____

NAME: _____

SIGNATURE: _____

1 (8 marks) RISC Pipeline and Instruction Design. For the first three questions, compare and contrast the two pipeline stages listed by identifying a similarity between them and then examining this similarity by clearly describing (a) each stage's unique role and (b) a way that one stage depends on the other. For example, in the student test-taking pipeline, if I listed the stages *Study* and *Take Test*, you might say: "The similarity is the ability to answer a question. *Study*'s role is to develop that ability and *Take Test*'s role is to measure the ability. *Take Test* depends on *Study* in that the better you study the higher the measured ability." Good answers will be two short sentences, one for (a) and one for (b).

1a Fetch and Decode

1b Execute and Memory

1c Decode and Write Back

Now, answer the next question about the difference between CISC and RISC instruction sets.

1d Recall that Intel's x86 ISA (i.e., IA32) includes instructions like "`addl 100(%eax), %ebx`", which adds a value from memory to a register, and Y86 and other RISC ISAs don't. Carefully explain how the classic 5-stage RISC pipeline we studied restricts RISC ISAs in this way.

2 (7 marks) Pipeline Performance.

2a Give a function $t(i, r, c)$ that computes the total running time of a program given:

i the total number of instructions executed by the program
r the clock rate
c CPI

$t(i, r, c) =$

2b Under what circumstances, if any, would adding a pipeline stage increase clock rate?

2c Under what circumstances, if any, would adding a pipeline stage increase CPI?

3 (8 marks) Parallelism, Dependencies and Hazards.

Consider the following Y86 assembly code.

```
[1]  pop %eax              # pop stack top into %eax
[2]  irmovl $1, %ebx       # %ebx = 1
[3]  addl %eax, %ebx       # %ebx = %ebx + %eax
[4]  mrmovl (%ebx), %eax   # %eax = M[%ebx]
[5]  push %eax             # push %eax onto the stack
```

List all of the *causal* dependencies that exist in this code by listing (a) the two line numbers involved, (b) the affected operand (e.g., register name), and (c) the number of stalls that would be required to resolve the hazard (i.e., stalls *added* for this hazard) in *Pipe-Minus* (i.e., without data forwarding).

4 (9 marks) Data Forwarding.

4a For each of the hazards listed in the previous question, describe how it is resolved in *Pipe* (i.e., with data forwarding) by listing (a) what is forwarded from where and (b) how many stalls are now required to resolve the hazard.

4b The load-use hazard requires one stall to resolve. Could this stall be avoided by forwarding to E instead of D? Carefully critique this idea.

4c Consider the following specific example of the load-use hazard.

```
mrmovl (%ebx), %eax     # %eax = M[%ebx]
push %eax               # push %eax onto the stack
```

Could the pipeline's data-forwarding mechanism be modified to avoid stalling in this case? If so, carefully explain the changes you would have to make to do so (in plain English; pseudo-code not required).

5 (8 marks) Jump Prediction.

5a Which of the following two techniques for jump prediction is better: (a) predicting that jumps are not taken or (b) predicting that a jump will be taken if and only if it was taken the last time it executed? Carefully justify your answer.

5b Carefully describe the potential benefit and drawback of moving the computation of `cnd` (aka `bch`) from E to F.

6 (4 marks) Architecting for Parallelism.

6a Is instruction-level parallelism important for good pipeline performance? Why or why not?

6b Is thread-level parallelism important for good pipeline performance? Why or why not?

7 (6 marks) Memory Hierarchy Characteristics and Tradeoffs.

7a Carefully explain why locality is important for good cache performance by indicating how caches exploit both temporal and spatial locality.

7b Give one benefit of increasing cache block size (and keeping all other cache parameters constant)?

7c Give one drawback of increasing cache size (and keeping all other cache parameters constant)?

Instructions Encoding

Byte	0		1		2	3	4	5
halt	0	0						
nop	1	0						
rrmovl **rA**, **rB**	2	0	**rA**	**rB**				
cmovXX **rA**, **rB**	2	fn	**rA**	**rB**				
irmovl **V**, **rB**	3	0	F	**rB**	**V**			
rmmovl **rA**, **D**(**rB**)	4	0	**rA**	**rB**	**D**			
mrmovl **D**(**rB**), **rA**	5	0	**rA**	**rB**	**D**			
OPl **rA**, **rB**	6	fn	**rA**	**rB**				
jXX **Dest**	7	fn	**Dest**					
call **Dest**	8	0	**Dest**					
ret	9	0						
pushl **rA**	A	0	**rA**	F				
popl **rA**	B	0	**rA**	F				

Y86 ISA Reference

Instruction	Semantics	Example
rrmovl %rs, %rd	r[rd] ← r[rs]	rrmovl %eax, %ebx
irmovl $i, %rd	r[rd] ← i	irmovl $100, %eax
rmmovl %rs, D(%rd)	m[D + r[rd]] ← r[rs]	rmmovl %eax, 100(%ebx)
mrmovl D(%rs), %rd	r[rd] ← m[D + r[rs]]	mrmovl 100(%ebx), %eax
addl %rs, %rd	r[rd] ← r[rd] + r[rs]	addl %eax, %ebx
subl %rs, %rd	r[rd] ← r[rd] - r[rs]	subl %eax, %ebx
andl %rs, %rd	r[rd] ← r[rd] & r[rs]	andl %eax, %ebx
xorl %rs, %rd	r[rd] ← r[rd] ⊕ r[rs]	xorl %eax, %ebx
jmp D	goto D	jmp foo
jle D	goto D if last alu result ≤ 0	jle foo
jl D	goto D if last alu result < 0	jl foo
je D	goto D if last alu result $= 0$	je foo
jne D	goto D if last alu result $\neq 0$	jne foo
jge D	goto D if last alu result ≥ 0	jge foo
jg D	goto D if last alu result > 0	jg foo
call D	pushl %esp; jmp D	call foo
ret	popl pc	ret
pushl %rs	m[r[esp] - 4] ← r[rs]; r[esp] = r[esp] - 4	pushl %eax
popl %rd	r[rd] ← m[r[esp]]; r[esp] = r[esp] + 4	popl %eax