CPSC 313, 05w Term 1— Midterm 2

Date: November 9, 2005; Instructor: Mike Feeley

This is a closed book exam; no notes; you may use calculators to perform simple arithmetic calculations. NOTE: the last page shows the PIPE architecture diagram. Answer in the space provided; use the backs of pages if needed. There are **??** questions on **??** pages, totaling **??** marks. You have **50 minutes** to complete the exam.

NAME: ____ **SCORE:** _____ / **??**

STUDENT NUMBER: ____________________

1. **(10 marks)** Short answers.

1a. Briefly explain how an n-channel MOSFET transistor closes the circuit between its source and drain when its gate voltage is high.

1b. Can an general-purpose processor (just the processor itself) be design using only combinational logic? Why or why not?

1c. Give one advantage the y86 (and/or a RISC processor) gets from confining memory access to special instructions that do nothing else but load from and store to memory (i.e., `mrmovl` and `rmmovl`.

1d. Explain the difference between a pipeline stall and a pipeline bubble?

1e. Briefly explain what a control hazard is, what stage of the y86 pipeline it affects and how.

_____ / 10

2. (6 marks) Pipelines.

2a. Carefully explain how pipelines improve performance.

2b. Give one reason why adding pipeline stages may **reduce** processor performance (i.e., make it worse).

2c. Give a formula that computes the maximum clock frequency (units: millions of cycles / second) of a five-stage pipeline with pipeline-stage delays (including memory delays) of 50 ps, 75 ps, 100 ps, 50 ps and 25 ps (ps = 1 trillionth of a second).

3. (7 marks) Each of the following pieces of code may contain a dependency. Indicate whether it does. If there is a dependency, give the **name of the dependency** (i.e., causal, anti or output) and **the register(s) involved**. Then indicate whether the code can be re-written to eliminate the dependency (without changing its meaning) and if so, modify the code to do so. You can assume that the remainder of the program will also be modified appropriately.

3a.
```
addl    %eax, %ebx
addl    %ebx, %ecx
```

Dependency (”NO” or name and register(s)):

Re-write possible (”YES” or ”NO”; if YES, modify the code):

3b.
```
addl    %eax, %ebx
irmovl  $1, %ebx
```

Dependency (”NO” or name and register(s)):

Re-write possible (”YES” or ”NO”; if YES, modify the code):

3c.
```
addl    %eax, %ecx
addl    %ebx, %eax
```

Dependency (”NO” or name and register(s)):

Re-write possible (”YES” or ”NO”; if YES, modify the code):

4. **(7 marks)** Consider the y86 PIPE discussed in class. For each of the following indicate whether a hazard exists, how it is handled (very briefly) and under what circumstances y86 introduces a pipeline bubble.

4a.
```
addl    %eax, %ebx
addl    %ebx, %ecx
```

Hazard (”YES” or ”NO”) and how it is handled:

Under what circumstances, if any, does y-86 introduce pipeline bubbles and how many:

4b.
```
jle     target
```

Hazard (”YES” or ”NO”) and how it is handled:

Under what circumstances, if any, does y-86 introduce pipeline bubbles and how many:

4c.
```
ret
```

Hazard (”YES” or ”NO”) and how it is handled:

Under what circumstances, if any, does y-86 introduce pipeline bubbles and how many:

_____ / 7

5. **(7 marks)** Consider the y86 PIPE implementation discussed in class (diagram attached to exam). The load-use hazard occurs when an instruction that reads a value from memory is immediately followed by an instruction uses that value.

5a. Give an example of the load-use hazard.

5b. Carefully explain how y86-PIPE resolves this dependency and why this solution is different from that of causal dependencies that don't involve memory.

5c. Notice that `pushl` (*IPUSHL*) and `rmmovl` (*IRMMOVL*) do not use the register value they read in the execute phase; their first use of this value is in the memory phase. It should thus be possible to handle load-use hazards involving these instructions without stalling. Explain why and very briefly outline a solution (the next question asks for more details)

.

6. **(7 marks)** This question continues the previous one. One solution that avoids stalling IPUSHL and IRMMOVL is to introduce a forwarding circuit in the execute stage that picks the correct `valA` to send to the memory stage for these two instructions. Give the HCL description of that circuit (refer to the diagram on last page of exam).

```
        int e_valA = [
```

7. **(6 marks)** Memory.

7a. By referencing properties of SRAM and DRAM describe why a memory hierarchy is necessary.

7b. Carefully explain the relationship between locality and caching.

7c. Carefully explain the difference between temporal and spatial locality.