CSE 471 Autumn 2001

Computer Design and Organization

# Midterm

Wednesday November 7th

NAME : _____

Do all your work on these pages. Do not add any pages. Use back pages if necessary. Show your work to get partial credit.

This exam is worth 40 points. After each question, you will find the number of points it is worth. You should spend approximately x minutes on a question worth x points. That will leave you with 10 minutes to read the statement of the problem and to look over your work.

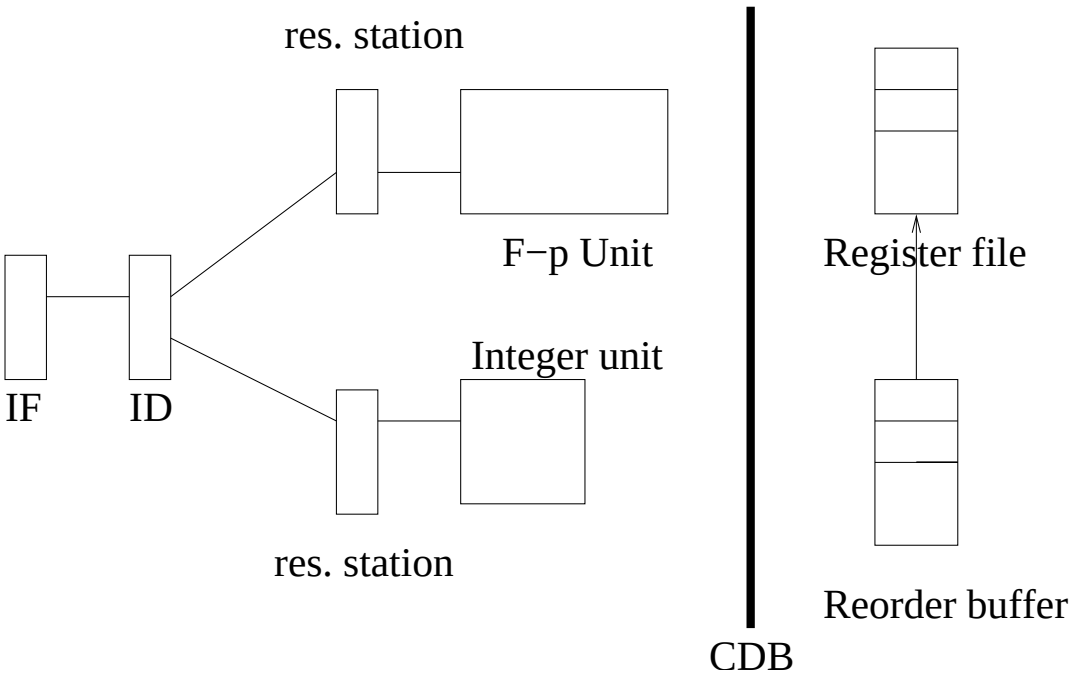
1. 20 points
   - (a) 1 point _____
   - (b) 3 points _____
   - (c) 3 points _____
   - (d) 2 points _____
   - (e) 3 points _____
   - (f) 2 points _____
   - (g) 4 points _____
   - (h) 2 points _____
2. 20 points _____
   - (a) 4 points _____
   - (b) 2 points _____
   - (c) 3 points _____
   - (d) 8 points _____
   - (e) 3 points _____

The figure below sketches the skeleton of an out-of-order processor that can dispatch, issue, and execute 32-bit instructions out-of-order but requires a commit step to complete them in order. It has two functional units: one for integer, load/store and branches, and the other for floating-point operations. Each of the two units has one reservation station. There is a reorder buffer and a register file. Communication between various units use a Common Data Bus (CDB).

Many of the features of this *single issue* processor are left undefined on purpose; you'll have to define some through your answers to the questions that follow.

The stages through which an instruction go through are:

1. Fetch (IF in the figure).
2. Decode, issue, and dispatch (ID). In this stage, structural hazards are detected (the flow of instructions stalls if there is a structural hazard), renaming takes place, reservation station(s) are filled etc.
3. Execute: The integer unit takes 2 cycles to execute; the floating-point unit takes 4. The units are pipelined.
4. Commit.



1. (20 points) (This question continues on the next 3 pages)
Associated with the processor is a Branch Prediction Unit. It consists of:

- An untagged Branch Prediction Buffer (BPB) of 1024 2-bit saturating counters.
- A direct-mapped Branch Target Buffer (BTB) of 256 entries.

(a) (1 point)
The BPB is accessed using an index formed by XORing the PC (Program Counter) bits (19+x-1,19) with PC bits (3+x-1,3). What is the value of x?

(b) (3 points)
How many bits are needed for each entry of the BTB. You can assume that the two rightmost bits of the 32-bit PC are always 0.

Consider now the flow of a branch instruction through the pipeline.

(c) (3 points)
What actions related to branch prediction occur during the IF stage?

(d) (2 points)
What actions related to branch prediction/execution occur during the ID stage?

(e) (3 points)
What actions related to branch prediction/execution occur at the end of the EX stage?

(f) (2 points)
What actions related to branch execution occur during the commit stage?

(g) (4 points)
The BPB is now replaced by a 2-level branch predictor. The global history is recorded by 2 shift registers of 10 bits each. The Pattern History Table consists of 1024 2-bit saturating counters. The branch predictor is thus an SAg(10,2).

How is the direction of a branch predicted?
When and how are the history registers being updated?

(h) (2 points)
What is needed to transform the 2-level predictor into a *gshare* predictor?

2. (20 points) (This question continues on the next 2 pages)

The processor shown on page 2 has a MIPS-like (or DLX-like) ISA and is the subject of this question.

(a) (4 points)
Give examples of sequences of instructions (instructions can be written in the form $R1 <- R2 + R3$) exhibiting respectively RAW, WAW, and WAR hazards. Which of these hazards is independent of the underlying architecture?

(b) (2 points)
The instruction stream stalls in the presence of structural hazards. Indicate two sources of structural hazards.

(c) (3 points)
Which units in the Figure should broadcast their results on the CDB and which should listen to that broadcast?

(d) (8 points)
There are several possibilities for implementing register renaming in this processor. Describe one particular implementation. Be sure to specify what happens at the various stages, how you allocate/deallocate resources etc.

(e) (3 points)
In the processor shown in the figure, throughput could be enhanced if completion out-of-order was allowed. Why is it not a good idea to allow out-of-order completion?