CS/ECE 752 ADVANCED COMPUTER ARCHITECTURE I
HOMEWORK # 2
(Due by 11:59 AM on Monday, Feb 22, via upload of PDF to Canvas)

Contact Haocheng Xiao (hxiao55@wisc.edu) for questions

1. (6 points)
Assume that in a particular program, 15% of the instructions are branches. The remaining 85% of the instructions has a CPI of 1.2. Consider a microarchitecture design A that does not have a branch predictor and has to expend 2 cycles for every branch instruction. Determine the CPI of design A.

Consider another design B that has a branch predictor with an accuracy of 90%. While a correct prediction (in B) requires only 1 cycle, a misprediction requires 3 cycles for the branch to execute. Determine the CPI of design B.

Consider yet another design C that has a branch predictor with an accuracy of 60%, and a branch execution time of 1 cycle for correct prediction and 5 cycles for misprediction. Determine the CPI of design C.

Which of these designs are the best and the worst?

2. (12 points, 4+4+4)
Answer the following questions based on the code sequence given in Figure 1.

```
Loop: LD     F2,0(Rx)
I0:   MULTD  F2,F0,F2
I1:   DIVD   F8,F2,F0
I2:   LD     F4,0(Ry)
I3:   ADDD   F4,F0,F4
I4:   ADDD   F10,F8,F2
I5:   SD     F4,0(Ry)
I6:   ADDI   Rx,Rx,#8
I7:   ADDI   Ry,Ry,#8
I8:   SUB    R20,R4,Rx
I9:   BNZ    R20,Loop
```

Latencies beyond single cycle	
Memory LD	+2
Memory SD	+1
Integer ADD, SUB	+1
Branches	+1
ADDD	+1
MULTD	+3
DIVD	+11

Figure 1. Code and latencies for Problem 2

(a) What would be the baseline performance (in cycles, per loop iteration) of the code sequence if no new instruction execution could be initiated until the previous instruction execution had completed? Ignore front-end fetch and decode. Assume for now that execution does not stall for lack of the next

instruction, but only one instruction/cycle can be issued. Assume the branch is taken.

(b) Consider a multiple-issue design. Suppose you have two execution pipelines, each capable of beginning execution of one instruction per cycle, and enough fetch/decode bandwidth in the front end so that it will not stall your execution. Assume results can be immediately forwarded from one execution unit to another, or to itself. Further assume that the only reason an execution pipeline would stall is to observe a true data dependence. Now how many cycles does the loop require?

(c) In the multiple-issue design of (b), you may have recognized some subtle issues. Even though the two pipelines have the exact same instruction repertoire, they are not identical nor interchangeable, because there is an implicit ordering between them that must reflect the ordering of the instructions in the original program. If instruction N + 1 begins execution in Execution Pipe 1 at the same time that instruction N begins in Pipe 0, and N + 1 happens to require a shorter execution latency than N, then N + 1 will complete before N(even though program ordering would have implied otherwise). Recite at least two reasons why that could be hazardous and will require special considerations in the microarchitecture. Give an example of two instructions from the code in Figure 1 that demonstrate this hazard.

3. (12 points) Introduction to ***gem5***
Learning gem5 book **-** ***http://learning.gem5.org/book/***

Step 1: Compile gem5

Go through the *Introduction* and *Building gem5* pages of the Learning gem5 book. Make sure to get your gem5 install working before moving onto the next step. It is advised to use Linux for this assignment.

NOTE: There is one small issue with the compilation command in the Learning gem5 book: it will not compile the *MinorCPU* model by default. Use the following command instead:

scons build/X86/gem5.opt -jX
CPU_MODELS=AtomicSimpleCPU,TimingSimpleCPU,O3CPU,MinorCPU

Step 2: gem5 Book, Part I

For this assignment, the most important parts of the Learning gem5 book are:

- downloading and building gem5,
- creating a simple configuration script,
- how to run gem5,
- adding some complexity to your first script by adding a two-level cache hierarchy,
- how to parse the gem5 output and understand the statistics, and

- using the default configuration scripts (se.py).

Step 3: Exercise

In this part, we will run applications from the GAP (Graph Algorithms Platform) benchmark suite on gem5 using ***se.py*** config file.

- Change directories to your gem5 directory.
- Then, clone the gapbs repository from *https://github.com/sbeamer/gapbs* using the *git clone* command.
- Go to *gapbs/* directory and run *make* command. This should generate the binaries for all applications.
- You can run *make test* to verify if the tests pass on your build.

You will perform a simple characterization of one of
(a) tc, with arguments "-g 15 -n 1" [if your last name begins with A-I]
(b) pr, with arguments "-g 20 -n 1" [if your last name begins with J-Q]
(c) bfs, with arguments "-g 17 -n 1" [if your last name begins with R-Z]

Submit the following.
Report the IPC and the cache hierarchy miss rates for:

Cache hierarchy for parts (a) and (b):
L1 instruction cache: 64KB 4-way set associative with 64B lines
L1 data cache: 64KB 4-way set associative with 64B lines
L2 cache: 256KB 8-way set associative L2 cache with 64B lines

(a) Simulate using the in-order CPU (MinorCPU) for the cache hierarchy
(b) Simulate using the out-of-order CPU (DerivO3CPU) for the cache hierarchy
(c) Simulate using the out-of-order CPU (DerivO3CPU), and changing the cache hierarchy to the following setting:

Two Level Cache:
L1 instruction cache: 32KB 2-way set associative with 64B lines
L1 data cache: 32KB 2-way set associative with 64B lines
L2 cache: 512KB 8-way set associative L2 cache with 64B lines

For all of the above tests, fast-forward 200M instructions, and simulate for next 80M instructions. The relevant config parameters can be found in configs/common/Options.py