16.482 / 16.561: Computer Architecture and Design

Fall 2013

Homework #5

Due **Monday, 11/18/13**

Notes:

- While typed submissions are preferred, handwritten submissions are acceptable.
- Any handwritten solutions that are scanned and submitted electronically must be clearly legible and combined into a single file—simply sending a picture of each scanned page is not an acceptable form of submission.
- Note that this assignment is worth a total of **150 points**, not 100.

1. Dynamic scheduling (30 points) Given the loop below:

```
          DADDI     R3, R0, #4
outer:    DADDI     R2, R1, #32
inner:    L.D       F0, 0(R1)
          MULT.D    F6, F0, F6
          S.D       F6, 8(R1)
          DADDI     R1, R1, #16
          BNE       R2, R1, inner
          DADDI     R3, R3, #-2
          BNEZ      R3, outer
```

Assume the following latencies:

- 1 cycle for DADDI, BNE, and BNEZ
- 3 cycles (1 EX, 2 MEM) for L.D and S.D
- 4 cycles for MULT.D

How long would this nested loop take without speculation? Remember, without speculation, you cannot fetch past a branch until the outcome of the branch is known.

2. Speculation (30 points) How many cycles will the sequence in Question 1 take if we do allow speculation and assume every branch prediction—including the predicted target from the BTB—is correct?

3. Speculation & branch prediction (40 points) Now, assume the processor has a 2-bit BHT to predict branch outcomes. On a mispredicted branch, the correct instructions are fetched starting with the cycle after the misprediction is recognized (EX). Assume that all BHT entries are initially equal to 00, and that the two branches in this example use separate BHT entries. Also, assume the BTB correctly predicts all targets for taken branches. How long will the loop in Question 1 now take?

4. Multithreading (50 points) Given the three threads shown below, determine how long they take to execute using (a) fine-grained multithreading, (b) coarse-grained multithreading, and (c) simultaneous multithreading.

For coarse-grained multithreading, switch threads on any stall longer than 1 cycle. (Note that you must determine the number of stall cycles based on dependences between instructions.) For simultaneous multithreading, treat thread 1 as the preferred thread, followed by thread 2 and thread 3.

Assume you are using a processor with the following characteristics:

- 6 functional units: 3 ALUs, 2 memory ports (load/store), 1 branch
- In-order execution
- The following instruction latencies:
    - L.D/S.D: 4 cycles (1 EX, 3 MEM)
    - ADD.D/SUB.D: 2 cycles
    - All other operations: 1 cycle

Thread 1:
```
L.D F0, 0(R1)
L.D F2, 8(R1)
ADD.D F4, F0, F2
SUB.D F6, F2, F0
S.D F4, 16(R1)
S.D F6, 24(R1)
DSUBUI R1, R1, #32
BNEZ R1, loop
```

Thread 2:
```
DADDUI R1, R1, #24
ADD.D F2, F0, F4
ADD.D F4, F6, F8
ADD.D F6, F0, F6
S.D F2, -24(R1)
S.D F4, -16(R1)
S.D F6, -8(R1)
BEQ R1, R7, end
```

Thread 3:
```
L.D F6, 0(R1)
ADD.D F8, F8, F6
S.D F8, 8(R1)
DADDUI R1, R1, #16
BNE R1, R2, loop
L.D F6, 0(R1)
ADD.D F8, F8, F6
S.D F8, 8(R1)
DADDUI R1, R1, #16
BNE R1, R2, loop
```