Department of Computer Science
National Tsing Hua University
CS510000 Advanced Computer Architecture
Spring, 2017
Homework 3
Due: May 10, 2017

1. (40%) Consider the following code segment:

```
Loop:   LW      R3, 0(R0)
        SUB     R4, R1, R3
        LW      R1, 0(R3)
        LW      R2, 0(R1)
        ADDI    R1, R1(#1)
        SUB     R4, R3, R2
        SW      R1, 0(R3)
        SW      R2, 0(R0)
        BNZ     R4, Loop
```

Assume a five-stage single-pipeline microarchitecture (fetch, decode, execute, memory, write-back). All ops are one cycle except LW and SW, which are 1 + 3 cycles, and branches, which are 1 + 2 cycles. There is no forwarding. Show the phases of each instruction per clock cycle for one iteration of the loop.

Hint: Addition cycles are added after MEM in SW, LW and after EXE in BNZ.

(1) How many clock cycles per loop iteration are lost to branch overhead?

(2) Assuming a static branch predictor capable of recognizing a backward branch in the Decode stage, how many clock cycles are wasted on branch overhead?

(3) Assuming a dynamic branch predictor, how many cycles are lost on a correct prediction?

2. (40%) Consider the following code segment.

```
LD      F3, 0(R0)
LD      F1, 0(R3)
ADDD    F1, F1, F3
ADDD    F2, F1,F3
MULTD   F4, F1, F2
LD      F3, 0(R0)
SUBD    F4, F3, F2
DIVD    F1, F3, F4
```

Show the number of stall cycles for each instruction and the clock cycle that each instruction begins execution (i.e., enters its first fetch cycle) for one iterations of the loop. How many cycles does it take? Report your answer using a table with the following column headers:

- Instruction
- Issues
- Executes/Memory access

- Write result
- Comment (wait for something)

Hint: Assume SD/LD takes 8 cycles, ADDD/SUBD takes 4 cycles, and MULTD/DIVD takes 10 cycles.

Ex:

Instruction	Issue	Execute/memory	Write result	Comment
LW F3, 0(R0)	1	2	3	Wait for F3

3. (20%) 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.** Furthermore, assume that the only reason an execution pipeline would stall is to observe a true data dependency. Now how many cycles does the loop require? Presume there is a superscalar architecture.

```
Loop:   LD      F2,0(RX)
I0:     DIVD    F8,F2,F0
I1:     MULTD   F2,F6,F2
I2:     SD      F2,0(Rx)
I3:     LD      F4,0(Ry)
I4:     DIVD    F8,F4,F2
I5:     ADDD    F4,F0,F4
I6:     ADDD    F10,F8,F2
I7:     ADDI    Rx,Rx,#8
I8:     ADDI    Ry,Ry,#8
I9:     SD      F4,0(Rx)
I10:    SUB     R20,R4,Rx
I11:    BNZ     R20,Loop
```

Latencies beyond single cycle:

LD +6
SD +2
ADDI, SUB +0
Branch +3
ADDD +1
MULTD +4
DIVD +8