High Performance Computer Architecture (CS60003)

Tutorial 1

Indian Institute of Technology Kharagpur

Exercises

1. Assembly code for a C program is given as follows:

```
Loop:   LD      F0   0(R1)
        MULTD   F4   F0   F2
        SD      F4   0(R1)
        SUBI    R1   R1   #8
        BNEZ    R1   Loop
```

If we predict that the branches are always taken, Tomasulo's Algorithm will allow multiple executions of this loop to proceed at once. Unroll the loop twice and update the following Instruction status table for the unrolled instructions. Consider the following assumptions while answering the question.

- The MULTD instruction takes 4 clock cycles. The 1st LD instruction takes 8 clock cycles (*L1 cache miss*) and the 2nd LD instruction takes 1 clock cycle (*cache hit*). The SD instruction takes 1 clock cycle.
- Since, we are only considering floating point operations, the integer ALU operation is ignored and it is assumed that the branch is predicted as taken. Therefore, we ignore the execution of decrement and branch instruction. The issue is always done in-order and there can be only one issue at a particular clock cycle. The instructions SUBI and BNEZ consume 1 cycle each.
- The initial content of register R1 is 80 and value of the scalar s is 10 (stored in F2). The contents in address locations #80 and #72 are 20 and 30 respectively.
- There are total 3 execution units - one for load and store, one for multiplication, and one for addition. Also assume that each execution unit has 4 reservation stations.
- Same cycle ISSUE -> DISPATCH and CAPTURE -> DISPATCH are not allowed.

Instruction	Issue	Execute	Write
LD			
MULTD			
SD			
LD			
MULTD			
SD			

High Performance Computer Architecture (CS60003)

Tutorial 1

Indian Institute of Technology Kharagpur

2. Remember that Tomasulo's algorithm requires tag broadcast and comparison to enable wake-up of dependent instructions. In this question, we will calculate the number of tag comparators and size of tag storage required to implement Tomasulo's algorithm in a machine that has the following properties:

   - 8 functional units where each functional unit has a dedicated separate tag and data broadcast bus
   - 32 64-bit architectural registers
   - 16 reservation stations per functional unit
   - Each reservation station entry can have two source registers

   Answer the following questions.

   (a) What is the total number of tag comparators per reservation station entry?

   (b) What is the total number of tag comparators in the entire machine?

   (c) What is the (minimum possible) size of the tag?

   (d) What is the (minimum possible) size of the Register Alias Table (RAT) in bits? Assume there is a 'valid' bit for each entry in RAT.

   (e) What is the total (minimum possible) size of the tag storage in the entire machine in bits?