16.482 / 16.561: Computer Architecture and Design
Spring 2014

Homework #4
Due **Thursday, 3/6/14**

Notes:
* While typed submissions are preferred, handwritten submissions are acceptable.
* Any handwritten solutions that are scanned and submitted electronically <u>must</u> be clearly legible and combined into a single file—<u>simply sending a picture of each scanned page is not an acceptable form of submission</u>.
* This assignment is worth a total of 100 points.

All problems deal with the code 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
```

The assembly code above represents a nested loop. The C code below gives a simple example of such a programming structure:

```
for (i = 0; i < 3; i++) {          // Outer loop
   for (j = 0; j < 4; j++) {       // Inner loop
      // Loop body
   }
}
```

In each outer loop iteration of a nested loop, you execute the entire inner loop. Therefore, in this example, you're executing the loop body four times for every outer loop iteration. Since there are three outer loop iterations, that means you're executing the loop body a total of 3 x 4 = 12 times.

The loop with which you are dealing doesn't have that many iterations, but you do have to account for all iterations of the outer and inner loops. You can determine exactly how many iterations there are using the code above.

Also, one other note: the DADDI instruction is a 64-bit integer add instruction, which used in 64-bit MIPS processors.

1. Dependences (40 points) Identify all true dependences, anti-dependences, and name
   dependences in the code above. Remember, you have a pair of nested loops, so you
   will likely find loop-carried dependences in the code.

2. Register renaming (20 points) Write out the instructions used in one outer loop
   iteration (which will contain multiple inner loop iterations) and show how the
   registers in that loop iteration are renamed.

   Assume that DADDI instructions can use reservation stations named Add1-Add9,
   load instructions can use reservation stations named Load1-Load9, and multiply
   instructions use reservation stations named Mult1-Mult9. (You should not need all of
   these reservation stations.) Note that, although store and branch instructions are
   assigned reservation stations, they do not write to registers and therefore do not need
   to rename their destination operands.

3. Dynamic scheduling (40 points) 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

Write a pipeline diagram to show how long all instructions in this nested loop will take in
a processor using Tomasulo's Algorithm. Assume that you do not have speculation and
that you cannot fetch the target of a branch until the branch outcome is known. Assume
the earliest you can determine the branch outcome is the IS stage, but note that you may
have to wait for all source operands to become available before you can compare them
and determine if the branch is taken.