This lab investigates Tomasulo’s algorithm and the use of a scoreboard in scheduling instructions in a loop.

1. Suppose you have one functional unit for all integer operations (add, subtract, multiply, and divide) and four separate functional units for floating point add, subtract, multiply, and divide. Each functional unit has three reservation stations available. Assume that loads and stores can be done in order by monitoring effective address calculation. Assume that \( t_1 \) points to one vector \( V \) with 5 elements, \( t_2 \) points to a second vector \( W \) with 5 elements, and \( f_0 \) contains a scalar \( \lambda \). Assume that the two vectors are in the L1 data cache. Register \( t_3 \) will be used to keep count. We will compute

\[
W = \lambda \cdot V + W
\]

```
addi $t3, $zero, 5
loop:
  l.d $f2, 0($t1)
  mul.d $f4, $f2, $f0
  l.d $f6, 0($t2)
  add.d $f8, $f6, $f4
  s.d $f8, 0($t2)
  addui $t1, $t1, 8
  addui $t2, $t2, 8
  addi $t3, $t3, -1
  bne $t3, $zero, loop
```

**Assignment** Analyze the performance of the above loop with regard to Tomasulo’s algorithm by making a table and running the loop manually for 5 iterations. Assume that it takes one clock cycle for each of the actions: fetch instruction (IF), enter decode queue (DQ), issue instruction, if possible, to reservation station (RS), and write register result (WR). I am assuming that a completed result can be handed off at the end of execution (EX) to a dependent instruction which will start executing at the beginning of the next cycle and I am assuming no local bus contention for all transfers. Assume that all data is in the L1 data cache and that this can be read/written (LD and ST) in one clock cycle. Assume that integer addition and subtraction can be done in one clock cycle (1 EX), that floating point addition takes 4 clock cycles (4 EX), and that floating point multiplication takes 7 clock cycles (7 EX). What could you do in order to speed up this loop? Options might be: additional functional units (integer or floating point), more reservation stations per functional unit, etc. Does it matter in this example whether or not the floating point multiply unit is pipelined?