Assignment #3

For the exercises below, consider the following code. The loop is called the DAXPY loop (Double-precision aX plus Y) and is the central operation in Gaussian elimination. This code implements the operation \( Y = aX + y \) for vectors of length 100. Initially, R1 and R2 contain the addresses of \( X[0] \) and \( Y[0] \) respectively, R3 contains the address of the memory location following the last element of \( X \), and F0 contains \( a \).

```
foo:  FLD    F2, 0(R1)  ; load X[i]
       FMUL.D F4, F2, F0  ; do a*X[i]
       FLD    F6, 0(R2)  ; get Y[i]
       FADD.D F6, F4, F6  ; calc a*X[i] + Y[i]
       FSD    F6, 0(R2)  ; Store Y[i]
       ADDUI R1, R1, #8  ; increment X index
       ADDUI R2, R2, #8  ; increment Y index
       BNE    R1, R3, foo ; No? Then loop again
```

1. (Static scheduling) Assuming the pipeline latencies from Figure 3.2, unroll the loop as many times as necessary to schedule it without any delays, and collapsing the loop overhead. Assume a one-cycle delayed branch. Show the schedule, and determine the number of cycles required for each iteration of the original loop.

ANS

The largest latency is 3, between one FP ALU OP and another FP ALU OP. In this code, this situation occurs between the FMUL.D and the subsequent FADD.D. There is already one cycle between these two, and we can move the ADDUI instructions up, so we need to unroll just once to hide all the latency.

```
foo:  FLD    F2, 0(R1)  ; load X[i]
       FLD    F6, 8(R1)  ; load X[i]
       FMUL.D F4, F2, F0  ; do a*X[i]
       FMUL.D F10, F8, F0  ; do a*X[i]'
       FLD    F6, 0(R2)  ; get Y[i]
       FLD    F12, 8(R2)  ; get Y[i]
       FADD.D F6, F4, F6  ; calc a*X[i] + Y[i]
       ADDUI R2, R2, #16  ; increment Y index
       FADD.D F12, F10, F12  ; calc a*X[i] + Y[i]
       FSD    F6, -16(R2)  ; Store Y[i]
       ADDUI R1, R1, #16  ; increment X index
       FSD    F12, -8(R2)  ; Store Y[i]
       BNE    R1, R3, foo ; No? Then loop again
```

ALL REQUIRED LATENCIES ARE MET

13 CYCLES TOTAL OR 6.5 CYCLES/LOOP
2. (Dynamic Scheduling) For this exercise we will add timing to the Tomasulo machine shown in Figure 3.10. Assume the following:

- The number of reservation stations are as shown in Figure 3.10
- There is no forwarding between function units; results are communicated by the CDB.
- One instruction can issue at each clock cycle. Execution of the instruction can begin in the next clock cycle following issue.
- Only 1 value can be on the CDB at a time. If two instructions complete in the same clock cycle, the later instruction is stalled until the next cycle.
- Loads take 1 cycle in execution.

Show the clock cycle when each instruction issues, starts execution, and completes (writes its result onto the CDB) for the first three iterations of the loop. Report your answer in a table similar to Figure 3.23 (Note that figure 3.23 shows a dual issue processor - we are doing a single issue example here.)

ANS.

DEPENDING ON YOUR ASSUMPTIONS, YOUR ANSWER MAY DIFFER FROM MINE IN FACT, I GET A DIFFERENT ANSWER EVERY TIME I DO THE PROBLEM!

I'LL LET YOU COME UP WITH THE CORRECT ANSWER!