332 Advanced Computer Architecture

Exercise 5.1. Software pipelining

In this question, assume a static-pipeline machine with the following characteristics:

- Integer arithmetic always takes just one clock cycle.
- There are no memory system delays.
- Note that the example exploits a delayed conditional branch. You should find that no stalls are caused the branch.
- FP operations are identified at the ID stage and are dispatched to separate non-pipelined functional units. Results are routed direct to the WB stage where they are handled in issue order.
- There are two FP functional units, one for add/subtract, the other for multiply/divide. FP addition takes 2 cycles, FP multiplication takes 5 cycles, and FP division takes 19 cycles (although you will not need division).
- There is full bypassing, i.e. wherever a hazard can be resolved using bypassing, the bypass circuitry is present.

Consider the following loop:

```
LOOP: SD   0(R1),F4     ; stores into M[i]
      ADDD F4,F0,F2     ; adds to M[i-1]
      LD   F0,-16(R1)   ; loads M[i-2]
      BNEZ R1,LOOP      ; delayed branch
      SUB  R1,R1,#8     ; executed whether branch taken or not
```

1. Using your paper sideways, draw a diagram showing the timing for the execution of the loop. You will need to look at more than one iteration.
   - **Indicate clearly where forwarding occurs**
   - **If you find a stall is necessary, explain clearly why**
2. What is the average CPI when the loop is executed many times?
3. Explain what this loop does, and describe how it is likely to be used to perform a *straightforward* useful function. Show any initialisation needed. What is going on? How does the performance of this loop compare with a straightforward implementation?

Paul Kelly, Imperial College, 2001