COE 501: Computer Architecture

Problem Set 5: Complex Pipelining

1) Consider the following so-called DAXPY loop used in Gaussian elimination. The loop implements the vector operation **Y = a * X + Y** for vectors **X** and **Y** of length 100. Initially, R1 is the address of X[0], R2 = address of Y[0], and R3 = address of X[100]. Here is the MIPS code:

```
loop:  L.D    F2, 0(R1)    ; load F2 = X[i]
       MUL.D  F3, F1, F2   ; F3 = a * X[i]
       L.D    F4, 0(R2)    ; load F4 = Y[i]
       ADD.D  F4, F3, F4   ; F4 = a * X[i] + Y[i]
       S.D    F4, 0(R2)    ; store Y[i] = F4
       ADDIU  R1, R1, 8    ; R1 = address of next X[i]
       ADDIU  R2, R2, 8    ; R2 = address of next Y[i]
       BNE    R1, R3, loop ; loop if not done
```

Consider the execution of the above loop on an in-order execution pipeline that has an instruction fetch (IF) stage, an instruction decode (ID) stage, three independent function units: ALU unit (1 pipeline EX stage), FP unit for floating-point addition, subtraction, and multiplication (4 pipeline stages: FP1, FP2, FP3, and FP4), and a memory unit for load and store instructions (2 pipeline stages: A for address calculation and M for memory access). Finally, there is a write-back (WB) stage that writes-back the result of an instruction into the register file. Each cycle, at most one instruction is fetched and at most one is completed. All hazards are detected in the ID stage. Assume a simple scoreboard is used for in-order execution. If an instruction in the ID stage cannot issue for execution then it will stall the pipeline.

a) Show the execution of one loop iteration. Assume a full-forwarding network that forwards results between function units. In addition, assume that the BNE instruction is predicted to be always taken (with zero delay) in the IF stage. Draw a timing diagram, showing stall cycles and forwarding. What is the total number of cycles to execute the 100 iterations of the above loop and the average CPI?

b) Show the content of the scoreboard (busy bit and destination register) for each cycle of a loop iteration. If an instruction in the ID stage cannot issue for execution, then indicate the type of hazard caused by this instruction during the given clock cycle.

c) Reorder the instructions in the above loop to reduce the total number of clock cycles. Redraw the timing diagram and compute the average CPI.

2) This problem explores hardware register renaming. To eliminate name dependences, WAR, and WAW hazards, the hardware renames registers. Assume the hardware has a pool of temporary registers (called T1 through T63), that are allocated in sequence. Unroll the first two iterations of the above DAXPY loop and rename all the destination registers, starting at T1. Make sure to rename the source registers as well to preserve data dependences.

3) Consider the execution of the above DAXPY loop on an out-of-order execution pipeline. A new Issue Stage (IS) is added to the pipeline with sufficient number of reservation stations. The reservation stations are distributed among the function units. As in problem 1, there are three independent functions units: ALU unit (1 pipeline EX stage), FP unit (4 pipeline stages: FP1, FP2, FP3, and FP4), and a memory unit for load and store instructions (2 pipeline stages: A-Unit for address calculation and M-Unit for Data Cache access). In addition, there is a Store Buffer (SB) for store instructions only. Store instructions wait in the store buffer until their data is present. A load instruction is allowed to bypass a previous store waiting in the store buffer if the load address is different from the store address. Instructions are allowed to complete out of program order. Draw a timing diagram showing the execution of the first two iterations of the DAXPY loop on the OOO execution pipeline. Assume that the loop branch is predicted to be always taken.

4) A new sufficiently large reorder buffer (ROB) and a Commit Stage (C) are now added to the pipeline of problem 3. Instructions are issued for execution out of program order. However, they commit their results in program order by updating the register file (or memory in the case of a store instruction). Show the timing diagram of the first two iterations of the DAXPY loop, given that the loop branch is predicted to be always taken.

5) Consider a superscalar pipeline that fetches two instructions, decodes two instructions, dispatches two instructions, writes two results into the ROB, and commits the results of at most two instructions each cycle. Because there are three independent function units, up to three instructions can be issued for execution in a given clock cycle. Redraw the timing diagram of the first two iterations of the DAXPY loop on the superscalar processor, given that the loop branch is predicted to be always taken.