EE 4720 **Homework 4 & 5** **Due: 23 April 1999**

In all problems below assume there are no cache misses and that all register values are available at the beginning of execution.

Problem 1: Show a pipeline execution diagram for the first 41 cycles of the code below on a dynamically scheduled implementation of DLX in which:

- There is one floating point multiply unit with a latency of 5 and an initiation interval of **2**.
- There is a load/store functional unit with a latency of 1. The segments are labeled `L1` and `L2`.
- The FP add functional unit has a latency of 3 and an initiation interval of 1.
- The integer functional unit has a latency of 0 and an initiation interval of 1.
- The functional units have reservation stations with the following numbers: integer, 6-9; fp add, 0-1; fp multiply, 2-3; load/store, 4-5.
- There is no reorder buffer.
- The branch delay is one. (There are no branch delay slots.)
- Ignore load/store ordering.

Initially all reservation stations are available.

```
LOOP:
 addi  r1, r1, #8
 sub   r2, r1, r3
 lf    f0, 0(r1)
 multf f1, f0, f0
 multf f2, f0, f1
 sf    4(r1), f1
 bneq  r2, LOOP   ! Assume always taken.
 xor   r4, r5, r6
```

Problem 2: Determine the CPI for a large number of iterations of the loop above (or give a good reason why it would be very difficult to determine the CPI).

Problem 3: What are the minimum number of reservation stations of each type needed so that the code above executes at maximum speed? What is the CPI at maximum speed? *(This part was not in the problem as originally assigned:)* The CDB can handle any number of writebacks per cycle and there are an unlimited number of functional units.

The problem as originally assigned was more tedious than intended. To solve it one would need to find a repeating pattern of iterations. Because of contention for the CDB, the repeating pattern does not occur in the first few iterations and so one would have to tediously construct the diagram for many iterations.

Problem 4: The code below executes on a machine similar to the type described in the first problem except that it uses a reorder buffer. Draw a pipeline execution diagram for the code below, be sure to show when each instruction commits. Remember that instructions stall in the functional unit if they are not granted access to the CDB.

```
LOOP:
 lf    f0, 0(r1)
 multf f1, f0, f0
 multf f2, f0, f1
 addf  f3, f3, f0
 lf    f4, 8(r1)
 sf    4(r1), f1
 multf f1, f4, f4
 multf f2, f4, f1
 addi  r1, r1, #16
 sub   r3, r4, r5
 xor   r6, r7, r8
 or    r9, r10, r11
```

Problem 5: Consider the code execution from the problem above. Suppose there is an exception in the `L2` segment executing the second `lf`. At what cycle would the trap instruction be inserted? What might go wrong if a reorder buffer had not been used?

Problem 6: Show the execution of the code below on a dynamically scheduled 4-way superscalar machine using a reorder buffer. Instruction fetch is aligned. There is one of each floating-point functional unit, with latencies and initiation intervals given in the first problem. There are four integer execution units. The reservation station numbers are as given in the first problem.

```
LOOP: = 0x1008
 lf    f0, 0(r1)
 multf f1, f0, f0
 multf f2, f0, f1
 addf  f3, f3, f0
 lf    f4, 8(r1)
 sf    4(r1), f1
 multf f1, f4, f4
 multf f2, f4, f1
 addi  r1, r1, #16
 sub   r3, r4, r5
 xor   r6, r7, r8
 or    r9, r10, r1
```

Problem 7: (Modified 12 November 1999) Rewrite the code below for the VLIW DLX ISA presented in class. Instructions can be rearranged and register numbers changed. In order of priority, try to minimize the number of bundles, minimize the use of the serial bit, and maximize the value of the lookahead field. When determining the lookahead assume that any register can be used following the last bundle in your code.

```
LOOP:
 lf    f0, 0(r1)
 multf f1, f0, f0
 multf f2, f0, f1
 addf  f3, f3, f0
 lf    f4, 8(r1)
 sf    4(r1), f1
 multf f1, f4, f4
 multf f2, f4, f1
 addi  r1, r1, #16
 sub   r3, r4, r5
 xor   r6, r7, r8
 or    r9, r10, r11
```