CS 3853: Computer Architecture (Spring 2017)
Homework 3
Assigned: April 4
Due in class April/4 (No delayed submission is allowed)
Total points: 92

Solution

Instructions:
Please write your name and banner ID on your homework submissions for posting grades.

Problem 1: Tomasulo's algorithm (38 points)
This exercise examines Tomasulo's algorithm on a simple loop operation. Consider the following code fragment:

```
LOOP:   L.D     F2, 0(R1)
        L.D     F4, 8(R1)
        DIV.D   F6, F2, F4
        MUL.D   F8, F6, F6
        ADD.D   F6, F2, F4
        MUL.D   F10, F6, F6
        S.D     F8, 0(R1)
        S.D     F10, 8(R1)
        DADDI   R1, R1, 16
        BNEZ    R1, LOOP
```

1. The pipeline functional units are described by the following table:

FU type	Cycles in EX	#of FU's	# of Reservation Stations
Integer	1	1	5
FP add/subtract	4	1	4
FP multiply/divide	15	2	4

2. Functional units are NOT pipelined (i.e.,· if one instruction is using the functional unit, another instruction cannot enter it).

3. All stages except EX take one cycle to complete.

4. There is no forwarding between functional units. Both integer and floating point results are communicated through the CDB.

5. Memory accesses use the integer functional unit to perform effective address calculation..

6. There is an infinite instruction queue. Loads and stores use integer reservation stations.

7. Loads and stores take one cycle to execute. Loads and stores share a memory access unit.

8. If an instruction finishes execution in cycle *x*, then another instruction that is waiting on the same functional unit (due to a structural hazard) can begin execution in cycle *x+1*.

9. There is only 1 CDB. So, only one instruction can write to the CDB in a clock cycle.

10. Branches and stores do not need the CDB since they don't have WR stage.

11. Whenever there is a conflict for a functional unit or the CDB, assume program order.

12. When an instruction is done executing in its functional unit and is waiting for the CDB, it is still occupying its reservation station. (meaning no other instruction may use the same reservation station). If the instruction broadcasts in CDB in cycle x, then its reservation station will be available in cycle x+1. For a store, its reservation station will be available in cycle x+1 if the store writes to memory in cycle x.

13. Treat the BNEZ instruction as an Integer instruction. Assume L.D instruction after the BNEZ can be issued the cycle after BNEZ instruction is issued due to branch prediction.

14. Initially, R1 < -16.

Fill in the execution profile for the first two iterations of the above code fragment in Table 2, including

- The reservation station used by each instruction. This should include both the functional unit type and the number of the reservation station. If multiple reservation stations of a particular type are available, associate early program order with lower cardinality.

0.5pt 0.5pt 0.5pt 0.5 pt

Instruction	Reservation Station	ISSUE	EXE Begin-End	Mem Access	CDB Write
L.D F2, 0(R1)	Integer 1	1	2	3	4
L.D F4, 8(R1)	Int 2	2	3	4	5
DIV.D F6,F2,F4	Mult 1 (1)	3	6-20		21
MUL.D F8,F6,F6	Mult 2 (1)	4	22-36		37
ADD.D F6,F2,F4	Add 1	5	6-8		10
MUL.D F10,F6,F6	Mult 3 (2)	6	11-25		26
S.D F8, 0(R1)	Int 1	7	8	38	
S.D F10, 8(R1)	Int 2	8	9	27	
DADDI R1,R1,16	Int 3	9	10		11
BNEZ R1, LOOP	Int 4	10	12		
L.D F2, 0(R1)	Int 5	11	13	14	15
L.D F4, 8(R1)	Int 3	12	14	15	16
DIV.D F6,F2,F4	Mult 4 (2)	13	27-41		42
MUL.D F8,F6,F6	Mult 1 (2)	22	43-57		58
ADD.D F6,F2,F4	Add 2	23	24-27		28
MUL.D F10,F6,F6	Mult 3 (1)	27	38-52		53
S.D F8, 0(R1)	Int 2	28	29	59	
S.D F10, 8(R1)	Int 3	29	30	54	
DADDI R1,R1,16	Int 4	30	31		32
BNEZ R1, LOOP	Int 5	31	33		

Table 1. Execution profile using Tomasulo's algorithm.

Problem 2: Tomasulo's algorithm with speculation (38 points)

Repeat the same problem as in Problem 1. But this time assume that the machine has support for speculation. Assume that there are infinite ROBs. Stores, once finished address calculation, does not sit in reservation station anymore. Rather it sits in ROB until commits. The other assumptions remain the same as before.

0.5 0.5 0.5 0.5

Instruction	Res. Station	ISSUE	EXE Begin-End	Mem Rd	CDB Write	Commit/ Mem Write
L.D F2, 0(R1)	Integer 1	1	2	3	4	5
L.D F4, 8(R1)	Int 2	2	3	4	5	6
DIV.D F6,F2,F4	Mult1(1)	3	6-20		21	22
MUL.D F8,F6,F6	Mult2(1)	4	22-36		37	38
ADD.D F6,F2,F4	Add1	5	6-9		10	39
MUL.D F10,F6,F6	Mult3(2)	6	11-25		26	40
S.D F8, 0(R1)	Int 3	7	8			41
S.D F10, 8(R1)	Int 2	8	9			42
DADDI R1,R1,16	Int 3	9	10		11	43
BNEZ R1, LOOP	Int 1	10	12			44
L.D F2, 0(R1)	Int 2	11	13	14	15	45
L.D F4, 8(R1)	Int 3	12	14	15	16	46
DIV.D F6,F2,F4	Mult4(2)	13	27-41		42	47
MUL.D F8,F6,F6	Mult1(2)	22	43-57		58	59
ADD.D F6,F2,F4	Add1	23	24-27		28	60
MUL.D F10,F6,F6	Mult3(1)	27	38-52		53	61
S.D F8, 0(R1)	Int 1	28	29			62
S.D F10, 8(R1)	Int 2	29	30			63
DADDI R1,R1,16	Int 1	30	31		32	64
BNEZ R1, LOOP	Int 2	31	33			65

Table 2. Execution profile using Tomasulo's algorithm with speculation

Problem 3: Dynamic Branch Prediction (16 points)

Consider the following MIPS code. The register R0 is always 0.

```
      DADDI   R1, R0, 2

L1:   DADDI   R12, R0, 4

L2:   DSUBI   R12, R12, 1
      BNEZ    R12, L2          -- Branch 1
      DSUBI   R1, R1, 1
      BNEZ    R1, L1           -- Branch 2
```

Each table below refers to only one branch. For instance, branch 1 will be executed 8 times. Those 8 times should be recorded in the table for branch 1. Similarly branch 2 is executed only 2 times.

Now assume that 2 level correlating predictors of the form (2,1) are used. Also assume that the branch predictor table has only 1 row. When the processor starts to execute the above code, the outcome of the previous two branches is not taken (N). Also assume that the initial state of predictors of all branches is not taken (N). What is the number of correct predictions? Use the following table to record your steps. Record the "New State" of predictors in the form W/X/Y/Z where,

- W - state corresponds to the case where the last branch and the branch before the last are both TAKEN
- X - state corresponds to the case where the last branch is TAKEN and the branch before the last is NOT TAKEN
- Y - state corresponds to the case where the last branch is NOT TAKEN and the branch before the last is TAKEN
- Z - state corresponds to the case where the last branch and the branch before the last are both NOT TAKEN

The first entry is filled in for you.

Branch 1:

Step	Branch 1 Prediction	Actual Branch 1 Action	New State
1	N	T	N/ N/ N/ T
2	N	T	N/T/ N/T
3	N	T	T/T/ N/T
4	T	N	N/T/N/T
5	T	T	N/T/N/T
6	N	T	T/T/N/T
7	T	T	T/T/N/T
8	T	N	N/T/N/T

B1(T) B1(T) B1(T) B1(N)
B2(T) B1(T) B1(T) B1(T)
B1(NT) B2(NT)

(*) 2 pt for each row

Branch 2:

Step	Branch 2 Prediction	Actual Branch 2 Action	New State
1	N	T	N/ N/ T/ N
2	T	N	N/N/N/N

n begins? Show your work.