McGill University
ECSE 425: COMPUTER ORGANIZATION AND ARCHITECTURE
Winter 2015
Midterm Examination

8:35 – 9:55 AM, March 10th, 2015

Duration: 80 minutes

- **Write your name and student number in the space below. Do the same on the top of each page of this exam.**
- **The exam is 12 pages long. Please check that you have all 12 pages.**
- **There are five questions for a total of 80 points. Not all parts of all questions are worth the same number of points; read the whole exam first and spend your time wisely!**
- **This is a closed-book exam. You may use one double-sided sheet of notes; please turn this sheet in with your exam.**
- **Faculty standard calculators are permitted; no cell phones or laptops.**
- **Clearly state any assumptions you make.**
- **Write your answers in the space provided. Show your work to receive full credit, and clearly indicate your final answer.**

Name: ________________________________

Student Number: ________________________________

Name: ID:

Q1: ___________/15 **Q2: ___________/20** **Q3: ___________/15**

Q4: ___________/15 **Q5: ___________/15**

Total:

Name: ID:

Question 1: Short and Not-so-short Answer (15 pts total)

Respond to each of the following; two to four sentences should be adequate in each case.

a. **(2 pts)** Even fully associative caches miss once. Why?

b. **(2 pts)** Assume a 40-bit virtual address space, with 8 KB pages. Main memory has a 4 GB capacity. L1 is virtually indexed, physically tagged, and direct-mapped, with 64B blocks. How many bits are used for block tag, index, and offset?

c. **(2 pts)** List four reasons CPI deviates from the ideal in a pipelined processor.

Name: ID:

d. (2 pts) Why do virtual memory systems require precise exceptions?

e. (2 pts) How does the re-order buffer ensure precise exceptions when instructions execute out of order?

f. (5 pts) What is the behavior of a (2, 1) correlating branch predictor for the following sequence of decisions for two branches, B1 and B2? Assume all predictor state is initialized to "taken" (T), and complete the following table.

Branch	Predictor State	BHR	Pred.	Outcome
B1				N
B2				T
B1				N
B2				T
B1				T
B2				T
B1				T
B2				T
B1				N
B2				T

Question 2: My Memory is Hierarchical (20 pts)

Consider a pipelined processor that runs at 2 GHz and has a base CPI of 1.1 when all cache accesses hit. The only instructions that read data from or write data to memory are loads (12% of all instructions) and stores (8% of all instructions).

The memory system for this computer is composed of a split L1 cache. Both the I-cache and D-cache are direct-mapped and hold 64 KB each. The I-cache has a 1% miss rate; the D-cache has a miss rate of 10%.

The 1 MB inclusive, unified L2 cache has an access time of 15 ns. Of all memory references sent to the L2 cache in this system, 70% are satisfied without going to main memory. Main memory has an access latency of 100 ns. Assume that L2 and main memory transfer times are accounted for in the access times above.

Assume that all caches use a write-back policy, and 40% of evicted data is dirty.

a. (4 pts) What is the average memory access time (in cycles) for instruction accesses?

b. (4 pts) What is the average memory access time (in cycles) for data accesses?

c. (4 pts) What is the overall CPI of the system?

d. (4 pts) If any one of the above access times could be reduced to a single cycle (i.e., so there are no stalls, as in the ideal case), what is the maximum speedup that can be achieved?

e. (4 pts) Rather than improve the memory hierarchy as in part (d), would it be more advantageous to instead increase the clock frequency to 3 GHz? Assume that L1 accesses still require a single cycle in the case of a hit, and that all other access latencies remain the same.

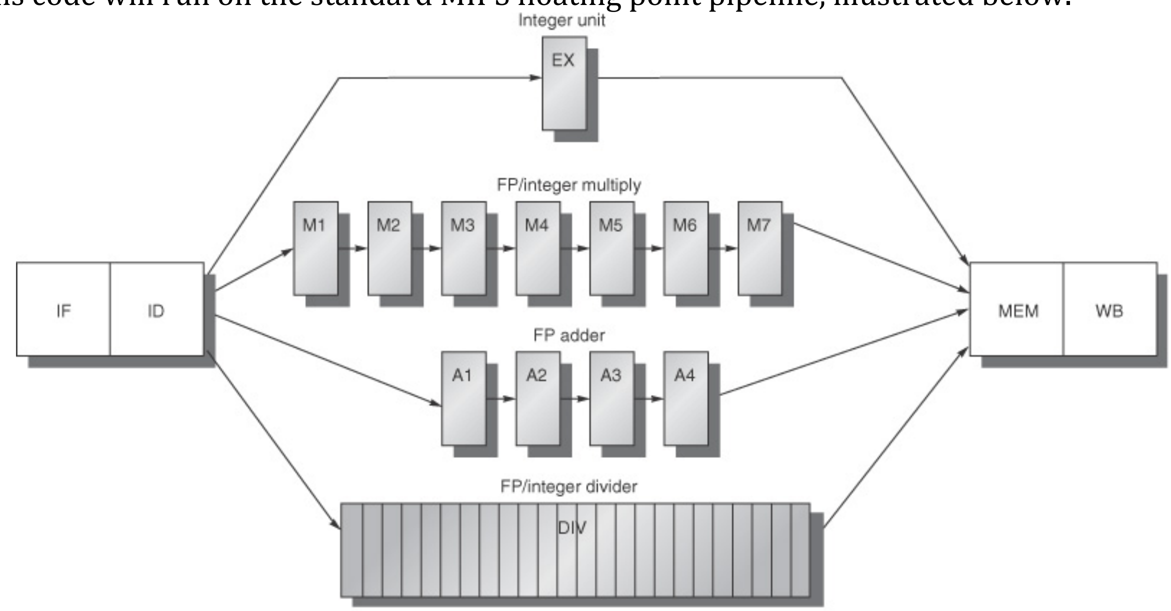
Name: ID:

Question 3: Does Obama Approve of this Pipeline? (15 pts)

Consider the following loop that computes performs a multiply-accumulate in F8.

```
Loop:   L.D       F2, 0(R1)
        L.D       F4, 8(R1)
        MULT.D    F6, F2, F4
        ADD.D     F8, F8, F6
        DADDUI    R1, R1, #16
        DSUBUI    R2, R2, #1
        BNEZ      R2, Loop
        S.D       F8, 0(R3)
```

This code will run on the standard MIPS floating point pipeline, illustrated below.



Assume:

- Full hazard detection and forwarding logic;
- The register file supports one write and two reads per clock cycle;
- Branches are resolved in ID; branches are handled by predicting not taken;
- Split L1 instruction and data caches service all requests in one clock cycle;
- Two or more instructions may simultaneously pass through MEM (WB) as long as only one makes use of the memory (register file);
- Structural hazards are resolved by giving priority to older instructions;

Complete the chart on the next page to show the execution of the loop if R2 = 2 initially. Indicate pipeline stages with {**F**, **D**, **X**, **M**, **W**}, and stalls with **S**.

Name: ID:

Instruction	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40

Name: ID:

Question 4: Architects, Roll Out! (15 pts)

Consider again the code from Q3.

```
Loop:   L.D       F2, 0(R1)
        L.D       F4, 8(R1)
        MULT.D    F6, F2, F4
        ADD.D     F8, F8, F6
        DADDUI    R1, R1, #16
        DSUBUI    R2, R2, #1
        BNEZ      R2, Loop
        S.D       F8, 0(R3)
```

Unroll the loop once and reschedule the instructions to minimize delay within the loop body (assuming the MIPS FP pipeline from Q3). Assume that control hazards are managed using a branch delay slot. In the space below, write out the re-scheduled instructions, including their operands. Insert placeholders for any stalls not eliminated by rescheduling.

Name: ID:

Question 5: This One's Out of Order (15 pts)

Consider once more the code from Q3.

```
Loop:  L.D      F2, 0(R1)
       L.D      F4, 8(R1)
       MULT.D   F6, F2, F4
       ADD.D    F8, F8, F6
       DADDUI   R1, R1, #16
       DSUBUI   R2, R2, #1
       BNEZ     R2, Loop
       S.D      F8, 0(R3)
```

This time, execute the program assuming an out-of-order processor with support for hardware speculation, and the following functional units:

Functional Unit	Cycles to Execute	No. of Reservation Stations
Integer ALU	1	4
FP Adder	4	2
FP Multiplier	7	1
Load/Store	*2	4

* The first cycle of Load/Store execution is used to calculate the effective address, and can proceed even if other operands are not yet ready.

Assume the processor predicts branches perfectly at the time of issue, and that R2 = 2 initially. Complete the following table.

Inst.	Issue	Begin EX	Finish EX	Write CDB	Commit

Name: ID:

Additional Page

Name: ID:

Additional Page