CMSC 611 – Project Questions

I am posting some queries that were asked to me and Dr.Younis regarding the project. They might help you to clear some doubts you may have about your own implementation.

Question :
If there are three instructions like this:
BNE R1, R2, GG
ADDI R3, R3, 1
SUBI R4, R4, 1

After ADDI get issued, can SUBI be fetched? If not, we just need to flush ADDI and If yes, do we need to flush both ADDI and SUBI given branch taken?
Answer : Both ADDI and SUBI need to be flushed (SUBI will be fetched speculatively).

Question :
When we need to write memory, if D-cache miss happens, we first need to allocate a block(may be two, but here we assume it is one), if is dirty, we need to write previous data to memory and read data from memory and then write the new data to cache. How long does this take? 1(calculation) + 12(write prev data to mem) + 12(read data from mem) + 1(write new data to cache and this may be 2 if two words) = 26(or 27). Is this right?

And if the allocated block is not dirty, we just need to read data from memory and then write the new data to cache. This will take 1(calculation) + 12(read data from mem) + 1(write new data to cache and this may be 2 if two words) = 14(or 15). Is this right?

And if we need to read memory, D-cache miss happens and the allocated block is dirty, we need to write previous data to memory, read data from memory to cache. This is will take 1(calculation) + 12(write prev data to mem) + 12(read data from mem) + 0(or 1 if two words) = 25 (or 26) . Is this right?
Answer : All the D-cache scenarios described above are correct.

Question : If a load miss requires writing the old block to memory, will that be another 12 cycles in addition to the 12 cycles used to fill the cache block?
Answer : Yes, it will take 12 more cycles to write the dirty block.

Question : Would an instruction cache miss still be able to occur between the write-back and load?
Answer : No, the handling of cache miss involving dirty block is atomic and cannot be preempted with an instruction cache miss.

Question : When the project statement says: "All caches are blocking and do not support “hit under misses”", does this mean that if there's a request out to obtain data from memory, that no further requests can be made to memory until that request has been fulfilled?
Answer : Yes. Hits under misses means allowing only 1 outstanding miss.

Question : It is ok for multiple functional units to complete the write stage in the same cycle correct?
Answer : Yes, we can allow multiple write in the same cycle.

Question : The structure of the data cache is 4 4-word blocks. Does this mean that there are 16 total data words, and then only 2 indexes?
Answer : Yes, total of 16 words. For the data cache 1 bit will be used for selecting the set.

Question : When a BEQ or BNE is in the issue stage, does the instruction after the branch stall at the fetch stage or continue until the branch is resolved in the read stage?
Answer : Continue until the branch condition is resolved.

Question : On the updated project slides (Example with memory hierarchy, slide 8), should there be a structural hazard on the instruction "ADD.D F6, F1, F7" in the second iteration.
Answer : It depends on the sequence of checks that are being applied. If you check for data hazards first (which is what we did in the example), you will not report structural hazard.

Question : At the end of the project Instruction Format and Semantics table, some of the instructions has a "." in between, such as ADD.D or L.D Is these as same as the ADDD or LD, which does not have that '.' Will the input file we are reading have the '.' such as ADD.D or L.D or just ADDD or LD

Answer : There will be "." The project statement and slides have been corrected.

Question : If a branching instruction is stalled in the read queue for a RAW hazard (as an example), then the pipeline stalls. Is this a special case only for branching? In the example without memory, the first ADD.D stalls in the read but the following instruction (SUB.D) is able to be read while the ADD.D is stalled in the read stage. This contradicts the stall from the branch instruction unless its a special case. Should a stall in the read stage cause the following instructions to be stalled?

Answer : The instruction that is stalled for RAW hazard will not prevent the subsequent instructions from proceeding (which is what we assume in the example). However, if the instruction is conditional branching, the RAW hazard will block subsequent instructions from processing (i.e., jam the Read stage).

Question : Do we need to handle the case where the label is not on the same line as the next instruction or can we assume the label will always appear at the beginning of the next instruction that should be executed if the label is targeted by a branch statement?

Answer : A label will be on the same line with the instruction.

Question : Even though the BNE and BEQ are decided in the Read stage, can these instructions stall in the Issue stage because of structural hazards waiting for the Integer Unit?

Answer : A branch does not stall of structural hazards waiting for the integer unit.