Please replace the following specification section in Project 2:

LOADs, STOREs and Memory Accesses with the ROB: Register values needed by LOADs and STOREs are forwarded to the respective ROB entries and when the ROB entry for a LOAD or STORE moves up to the head of the ROB and has all register operands available (that is, ready), a memory operation is initiated on a dedicated pipelined memory unit that has latency of 2 cycles (one cycle for computing the memory address and one cycle for accessing the D-cache. Assume that all D-cache accesses result in a hit for this project.

with the following specifications and suggestions that will simplify your implementation:

1. Memory Operations for Loads and Stores:

The LOAD and STORE instructions perform ALL operations (memory address computation, memory read or write) associated with these instructions when the instruction is at the head of the ROB. A memory operation is initiated on a dedicated pipelined memory unit that has latency of 2 cycles (one cycle for computing the memory address (using Stage M1) and one cycle for accessing the D-cache (using Stage **M2**). Source register operands are read out from the register file into Stage M1 using the connections shown in green in the diagram. Assume that all D-cache accesses result in a hit for this project.

To start the operation, all source register operands for the instruction need to be available. The availability of these operands are noted in the IQ entry for the load or store instruction. It is also the responsibility of the IQ to inform the ROB entry for a load or store on whether the source operands are available. Given these requirements, you may want to implement them as follows:

Add a ***mready*** bit in the ROB entry for a load or store – this *mready* bit ("ready to start memory operations") indicates if the memory operation can be initiated (via the stages M1 and M2) when the load or store is at the head of the ROB. The ROB entry for a load or store also has fields for the various *physical* source registers addresses.

2. IQ Entry for Loads and Stores:

- The IQ logic sets the *mready* bit when all source register operands for a load or store are available and the IQ entry is removed immediately in the same cycle when the IQ logic discovers that all source operands are available. In effect, the IQ is issuing the load or store instruction to the ROB entry by setting the *mready* bit in the ROB entry for the load or store.
- If at least one of the source register operands for a load or store are not available at the time of dispatch, an IQ entry needs to be created for the load or store.
- If all of the source register operands for a load or store are available at the time of dispatching, simply set up the ROB entry for the load or store and set the *mready* bit. **DO NOT** create an IQ entry for the load or store in this case.

3. Timing for Starting Memory Operations via stages M1 and M2:

- If the ROB entry for the load or store **is already at the head of the ROB** when its *mready* bit is set by the IQ logic, you need to simulate that the load or store moves into the M1 stage to begin operations as soon as the last source register value it was waiting for is produced. This is equivalent to broadcasting register tags at least one cycle prior to the availability of the operands and can be simulated in a manner identical to what you have used for the other function units.

- If the ROB entry for a load or store has its *mready* bit set before the ROB entry is at the head, the memory operation for the entry is started in the same cycle when it is at the head.

The head pointer of the ROB entry is updated as soon as the memory operation for a load or store is started.