**CSC521 Q233 Final Exam (Part A)**

Total: /50

Name (last, first): \_\_Hassan, Sunzid\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

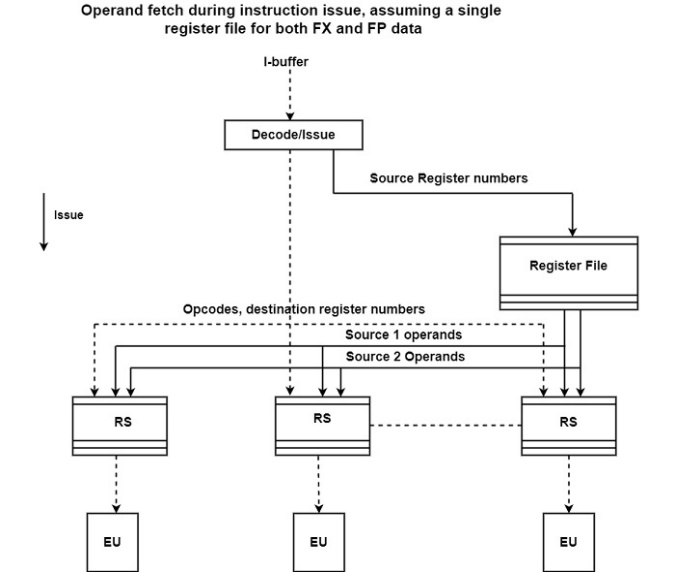
“*Things [exams] should be made as simple as possible –   
but not simpler”*

*Parallel Computation*

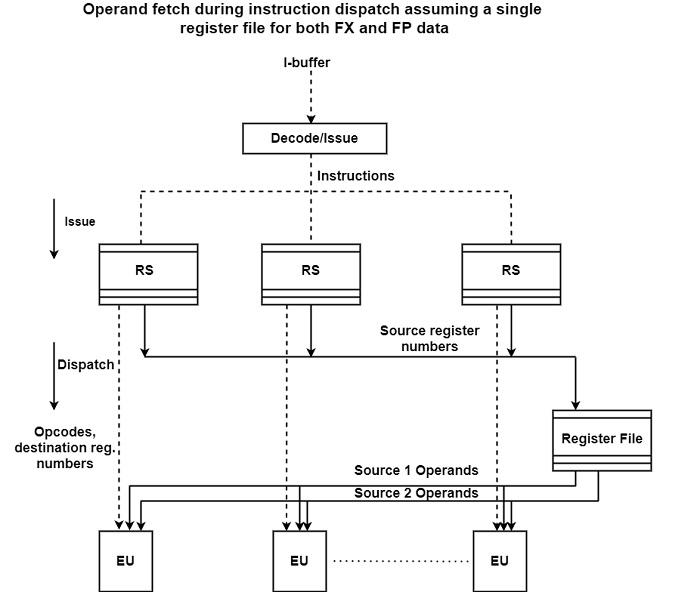
1. Provide two design choices of when to fetch operands in a Superscalar processor (fetch operands during what stages or steps). (15 points)

**Ans: Design choices to fetch operands in a superscalar processor**

**Issue-Bound fetch:** Operands are fetched during instruction issue. Then shelving buffers hold source operand values. For example, IBM 360/91 (1967), PowerPC 620(1996) use issue bound operand fetch policy.

****

**Dispatch-Bound fetch:** Operands are fetched during instruction dispatch. Then shelving buffers hold source register numbers. For example, CDC 6600 (1994), PA 8000 (1996) use dispatch bound operand fetch policy.

****

1. **Using a Re-Order Buffer (ROB) for preserving the sequential consistency, when can an instruction be allowed to retire? (15 points)**

* Re-Order Buffer (ROB) for preserving the sequential consistency:

1. Each entry is extended to include a speculative status field indicating whether the corresponding instruction has been executed speculatively.
2. speculatively executed instruction are not allowed to retire before the related condition is resolved.
3. After the related condition is resolved, if the guess turn out to be right, the instruction can retire in order.
4. if the guess is wrong, the speculative instructions are marked to be cancelled. Then, instruction execution continues with the correct instructions.

5. Instruction are written into the ROB in strict program order:

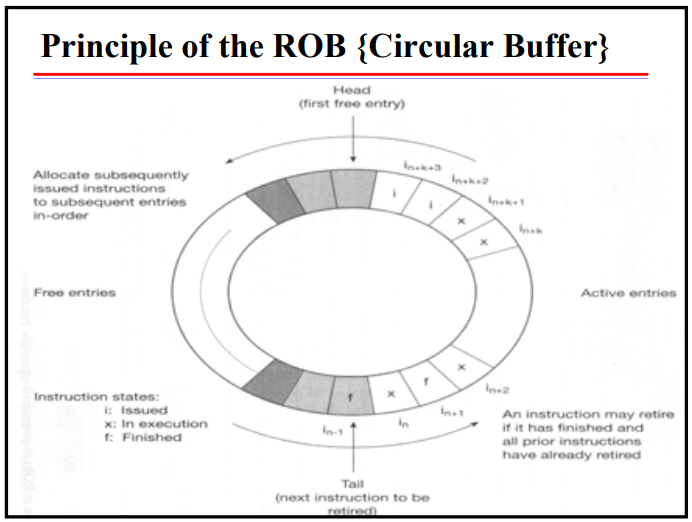
* One new entry is allocated for each active instruction.

6. Each entry indicates the status of the corresponding instruction

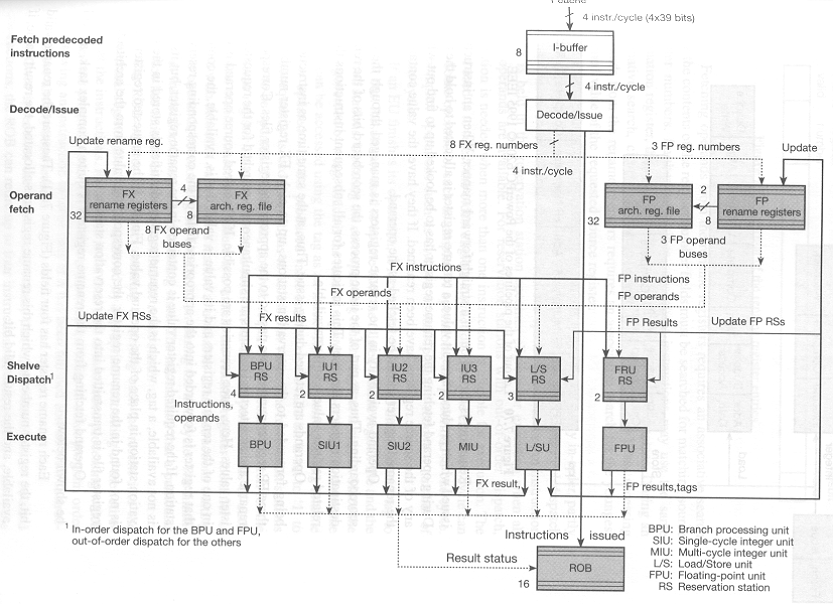
* issued (i),
* in execution (x),
* already finished (f)
* An instruction being allowed to be retired, when

An instruction is allowed to retire only if it has finished, and all previous instruction are already retired.

* retiring in strict program order
* only retiring instructions are permitted to complete, that is, to update the program state:
  + by writing their result into the referenced architectural register or memory.



# See the following figure for the case Studies of PowerPC 620 processor. (a) Describe (outline) the functions of each (and every one) of the boxes in the diagram. (b) Describe (outline) how the processor can process multiple instructions in parallel. [20 points]



1. **Description of diagram**

Instruction buffer- takes the instruction and store them.

Decode issue- decodes instruction from the instruction buffer.

FX architectural register file- store the registers during fixed point operations.

FX rename register- renames registers for fixed-point operations to remove false dependencies.

FP architecture register file- store the registers for floating point operations.

FP rename register- rename register for floating point operations.

RS- reservation stations are instruction buffers/shelves.

BPU-branch processing unit, L/SU, MIU, **SIU1(**Single cycle integer unit to get the single integer stations), **SIU2(**Single cycle integer), **FPU (**Floating point unit to get the floating-point stations) are execution units, execute unit performs the instruction.

ROB: To reorder buffer to get results.

B) **Reservation station can execute the instructions as soon as the data are ready, that is dataflow model and data flow model is parallel.**

* Inputs into I-buffer,
* Decode inputs and if it is decoded put the result directly into ROB,
* If not put the results into FX/FP rename registers into assign new register for them such as R0, R1
* Put the R0, R1 registers into FX/FP.
* Arch register file to store the state of the data,
* After the above state, get assigned data and we put data into BPU RS, IUXrs, L/S Rs.
* Then BPU RS, IUXrs, L/S Rs execute inputs and put the results into BPU, SIU, L/SU & FPU.
* The executed data is allocated into ROB.