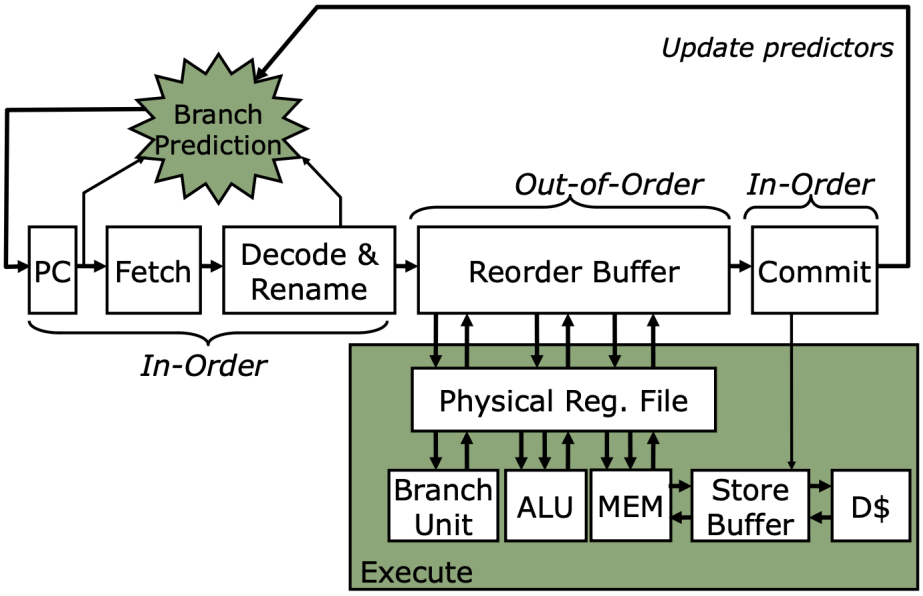
Quiz 1 Handout - Physical Register File Design

This handout defines a simple RISC-V Out-of-Order processor. All register data is stored directly in a physical register file. The ROB simply contains tags that point to physical registers.

The processor follows the "Physical Register File" design seen in lecture. A diagram representing the processor is shown in Figure 1.

Figure 1: Processor Overview



The processor follows the RISC-V ISA, though we will only focus on integer and control instructions, so you do not need to worry about memory instructions.

The processor has the following key stages:

1. **Fetch:** the instruction at PC is fetched from the instruction cache/memory
2. **Decode & Rename:** the fetched instruction is decoded. If the decoded instruction is a conditional branch, its direction is predicted by the branch predictor. The instruction's architectural source registers are renamed to the appropriate physical registers. A destination physical register is assigned.
3. **Reorder Buffer:** the instruction waits to be executed in the reorder buffer. When all hazards are cleared, it is able to execute.
4. **Commit:** the instruction is committed.

Figure 2: Current Processor State

Rename Table

Arch. Reg.	Physical Reg.
x0	
x1	P7
x2	P8
x3	
x4	P4
x5	P6
x6	P9
x7	

Physical Register File

Physical Reg.	Value	Present?
P0	37	
P1	38	
P2	823	
P3	5900	
P4	2816	
P5	0	
P6	1123	
P7	314	
P8	217	
P9	415	
P10		

Free List

Physical Reg.
P10

Reorder Buffer

	Inum	Use	Ex	Op	PR1 Present?	PR1	PR2 Present?	PR2	Rd	Last PRd	PRd
	…										
Next to Commit →	I5	1	1	addi	1	P0			x1	P0	P1
	I6	1	0	xor		P2		P4	x2	P2	P3
	I7	1	0	div		P3		P5	x5	P5	P6
	I8	1	0	bne		P1		P9			
	I9	1	0	add		P1			x1	P1	P7
	I10	1	0	xor		P3		P4	x2	P3	P8
Next Available →											

- **Rename Table:** mapping between architectural registers and physical registers
- **Physical Register File:** central physical register file that holds all register data
- **Free List:** available physical registers not currently assigned to any architectural register
- **Reorder Buffer**: holds inflight instructions and the tags of their input and output registers

Label List

A. Satisfy a dependence on _____ by stalling
B. Satisfy a dependence on _____ by bypassing a speculative value
C. Satisfy a dependence on _____ by bypassing a committed value
D. Satisfy a dependence on _____ by speculation using a static prediction
E. Satisfy a dependence on _____ by using a dynamic prediction
F. Write a speculative value using lazy data management
G. Write a speculative value using greedy data management
H. Speculatively update a prediction on _____ using lazy value management
I. Speculatively update a prediction on _____ using greedy value management
J. Non-speculatively update a prediction on _____
K. Check the correctness of a speculation on _____ and find a correct speculation
L. Check the correctness of a speculation on _____ and find an incorrect speculation
M. Abort speculative action and cleanup lazily managed values
N. Abort speculative action and cleanup greedily managed values
O. Commit correctly speculated instruction, where there was no value management
P. Commit correctly speculated instruction, and mark lazily updated values as non-speculative
Q. Commit correctly speculated instruction, and free log associated with greedily updated values
R. Illegal or broken actions

Blank Choices

i. Register Value
ii. PC value
iii. Branch direction
iv. Memory address
v. Memory value
vi. Latency of operation
vii. Functional unit
viii. Storage space