Allocation

MO801

Dispatch

- Send instructions to specific functional units (FUs)
- Some instructions need more than one functional unit
  - Reserve all of them
  - Do a step by step execution
- Multiple functional units of the same type
  - Select which one to queue each instruction

Register Renaming

- Processors do not have enough register for all variables
- Registers are reused
- Out-of-order processors can execute instructions in different order
- Whenever a new value is assigned to a register, the old value is lost

Data and Name Dependences

```
r1 = r2 + r3
…
r4 = r1 + r5
```

Data dependence
Read after write

```
r1 = r2 + r3
…
r1 = r4 + r5
```

Name dependence
Write after write

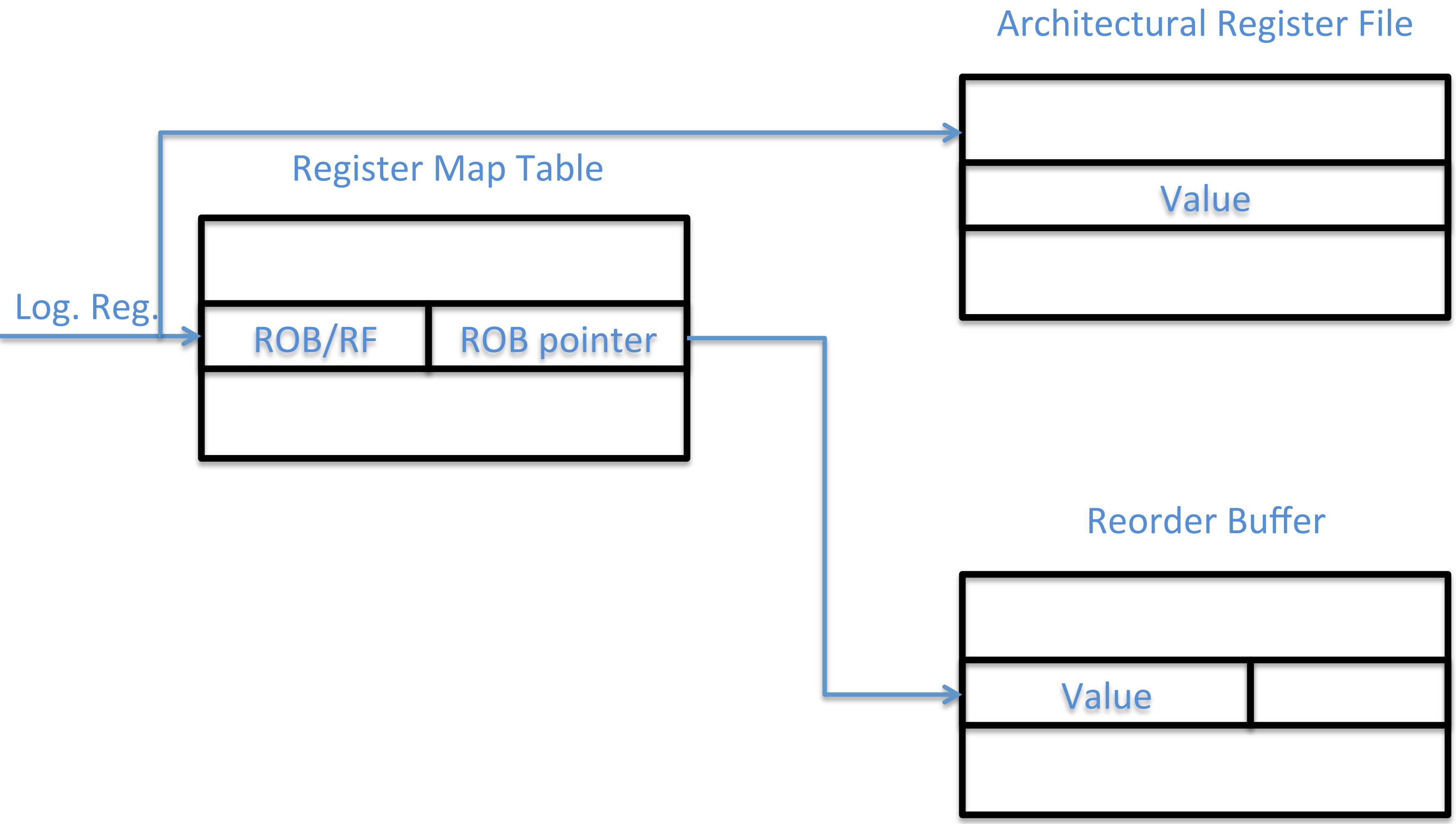
```
r1 = r2 + r3
…
r2 = r4 + r5
```

Name dependence
Write after read

Renaming

- Use extra registers to store additional values
- Processors can have about 100 instructions on the fly
  - Possibly requiring 100 registers in total
  - Usually they have 32 registers
- Three alternatives
  - Renaming through reorder buffer
  - Renaming through a rename buffer
  - Merged register file

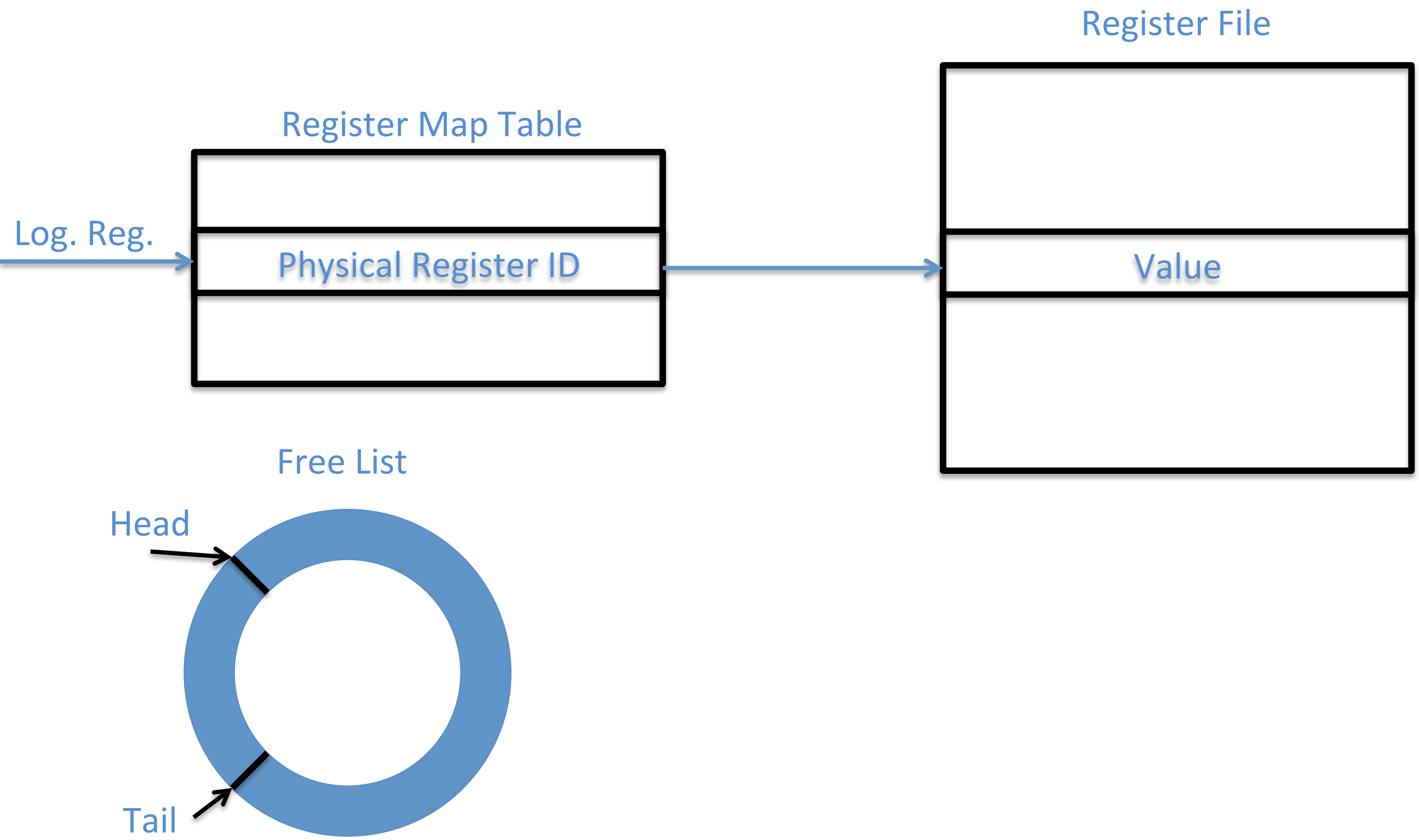
Reorder Buffer



Rename Buffer

- Variation of the previous method
- One third of instructions do not produce values!
- Use a separate structure to store the results, instead of ROB
- Rename buffer can be smaller than ROB
  - If a instruction needs a rename buffer and there is no empty space, the instruction is stalled

Merged Register File



Register File

Register Map Table

Log. Reg.

Physical Register ID

Value

Free List

Head

Tail

Merged Register File

- Single and Bigger Register File
  - Holds speculative and committed values
  - Registers can be free or allocated
- Uses a free list to keep track of free registers
- Uses a register map table to map logical registers to physical registers
- If there is no free register, instructions are stalled
- Only frees a register when another instruction writes to it
  - Need to wait until commit because instruction could be squashed

Register File Read

- Read before issue
  - Not all operands are available
  - Process non-available later through bypass network
  - Fewer register ports, more temporary values
  - More energy
- Read after issue
  - Store id in the issue queue
  - Process operands through register file and bypass network
  - Larger number of register port
  - Does not require intermediate storage

Recovery from Misspeculation

- Whenever in flight instructions need to be squashed
- Release the reserved resources
- Reverses allocated registers
  - Rename tables
  - Reorder buffer

Comparison

Reorder Buffer and Rename Buffer

- Do not require the free list
- May require two writes
- May require extra bypasses
- Better to read before issue

Merged Register File

- Register values are written only once
- All source operands come from a single location
- Can be used with the two read approaches