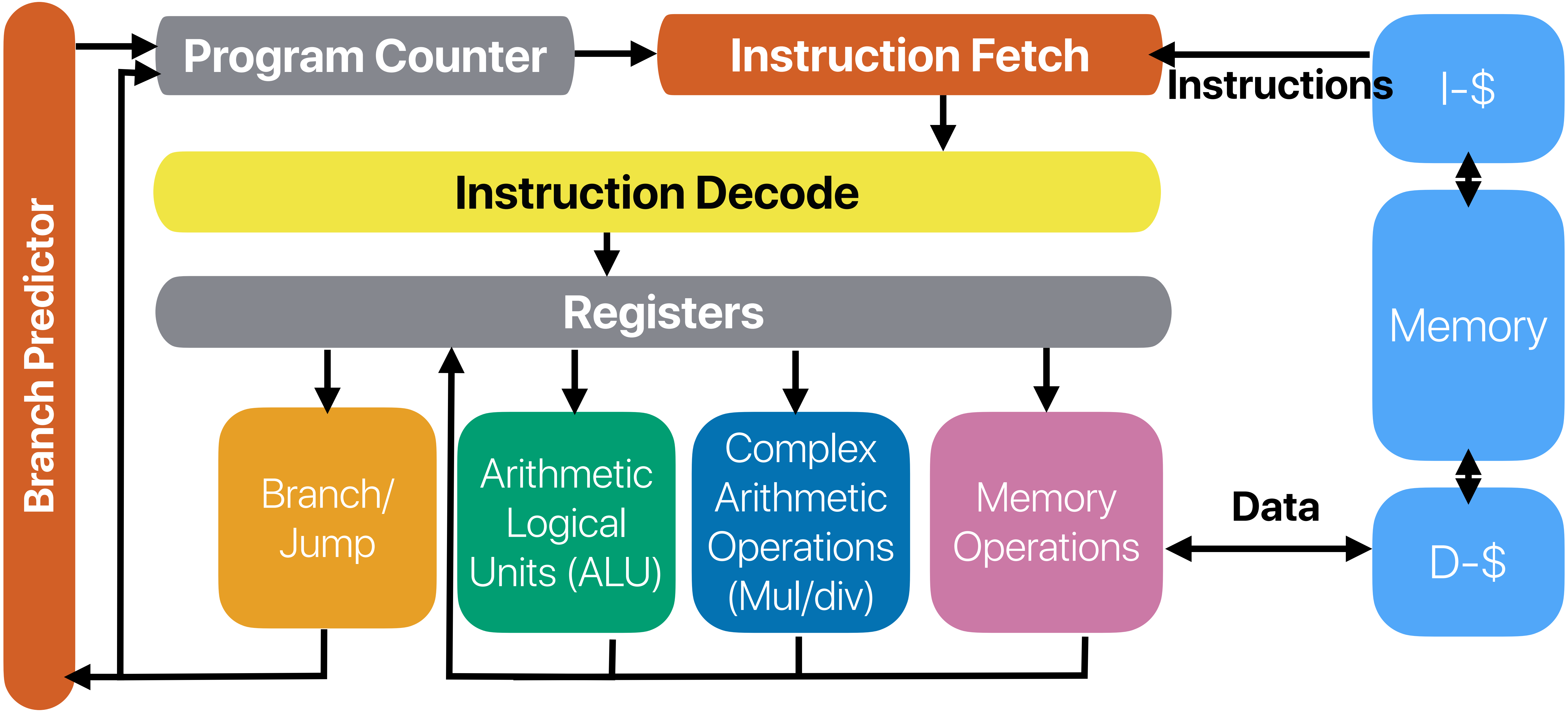
# Modern Processor Design (IV): Don’t wait for me

Hung-Wei Tseng

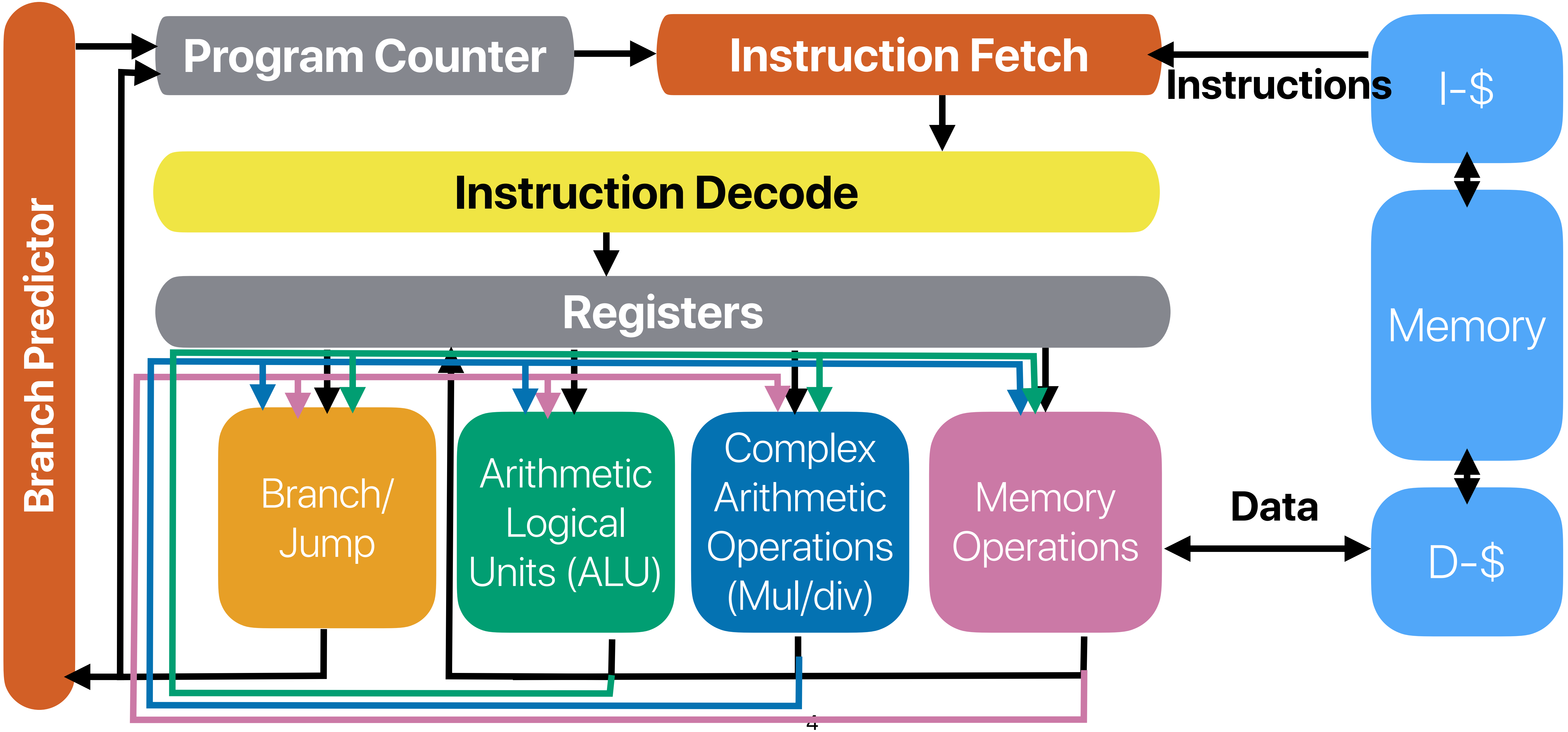
# Recap: addressing hazards

- Structural hazards
  - Stall
  - Modify hardware design
- Control hazards
  - Stall
  - Static prediction
  - Dynamic prediction — all "high-performance" processors nowadays have pretty decent branch predictors
    - Local bimodal
    - Global 2-level
    - Perceptron
    - TAGE
- Data hazards
  - Stall
  - Data forwarding

# The "current" pipeline



# Data "forwarding"



# Let's extend the example a bit...

```
for(i = 0; i < count; i++) {
    int64_t temp = a[i];
    a[i] = b[i];
    b[i] = temp;
} .L9:
  ①  movq    (%rdi,%rax), %rsi
  ②  movq    (%rcx,%rax), %r8
  ③  movq    %r8, (%rdi,%rax)
  ④  movq    %rsi, (%rcx,%rax)
  ⑤  addq    $8, %rax
  ⑥  cmpq    %r9, %rax
  ⑦  jne     .L9
  ⑧  movq    (%rdi,%rax), %rsi
  ⑨  movq    (%rcx,%rax), %r8
  ⑩  movq    %r8, (%rdi,%rax)
  ⑪  movq    %rsi, (%rcx,%rax)
  ⑫  addq    $8, %rax
  ⑬  cmpq    %r9, %rax
  ⑭  jne     .L9
```

| | IF | ID | ALU/BR/M1 | M2 | M3 | M4/XORL | WB |
|---|---|---|---|---|---|---|---|
| 1 | (1) | | | | | | |
| 2 | (2) | (1) | | | | | |
| 3 | (3) | (2) | (1) | | | | |
| 4 | (4) | (3) | (2) | (1) | | | |
| 5 | (4) | (3) | | (2) | (1) | | |
| 6 | (4) | (3) | | | (2) | (1) | |
| 7 | (4) | (3) | | | | (2) | (1) |
| 8 | (5) | (4) | (3) | | | | (2) |
| 9 | (6) | (5) | (4) | (3) | | | |
| 10 | (7) | (6) | (5) | (4) | (3) | | |
| 11 | (8) | (7) | (6) | (5) | (4) | (3) | |
| 12 | (9) | (8) | (7) | (6) | (5) | (4) | (3) |
| 13 | (10) | (9) | (8) | (7) | (6) | (5) | (4) |
| 14 | (11) | (10) | (9) | (8) | (7) | (6) | (5) |
| 15 | (11) | (10) | | (9) | (8) | (7) | (6) |
| 16 | (11) | (10) | | | (9) | (8) | (7) |
| 17 | (11) | (10) | | | | (9) | (8) |
| 18 | (12) | (11) | (10) | | | | (9) |
| 19 | (13) | (12) | (11) | (10) | | | |
| 20 | (14) | (13) | (12) | (11) | (10) | | |
| 21 | | (14) | (13) | (12) | (11) | (10) | |
| 22 | | | (14) | (13) | (12) | (11) | (10) |

10 cycles for 7 instructions

CPI = 1.43

# Outline

- Dynamic Instruction Scheduling
- SuperScalar
- Programming modern processors

# The mechanism of OoO: Register renaming + speculative execution

- K. C. Yeager, “The MIPS R10000 superscalar microprocessor,” in IEEE Micro, vol. 16, no. 2, pp. 28-41, April 1996.

# Register renaming + OoO

- Redirecting the output of an instruction instance to a **physical register**
- Redirecting inputs of an instruction instance from **architectural registers** to correct **physical registers**
  - You need a mapping table between architectural and physical registers
  - You may also need reference counters to reclaim physical registers
- OoO: Executing an instruction all operands are ready (the values of depending physical registers are generated)
  - You will need an **issue logic** to **issue** an instruction to the target functional unit

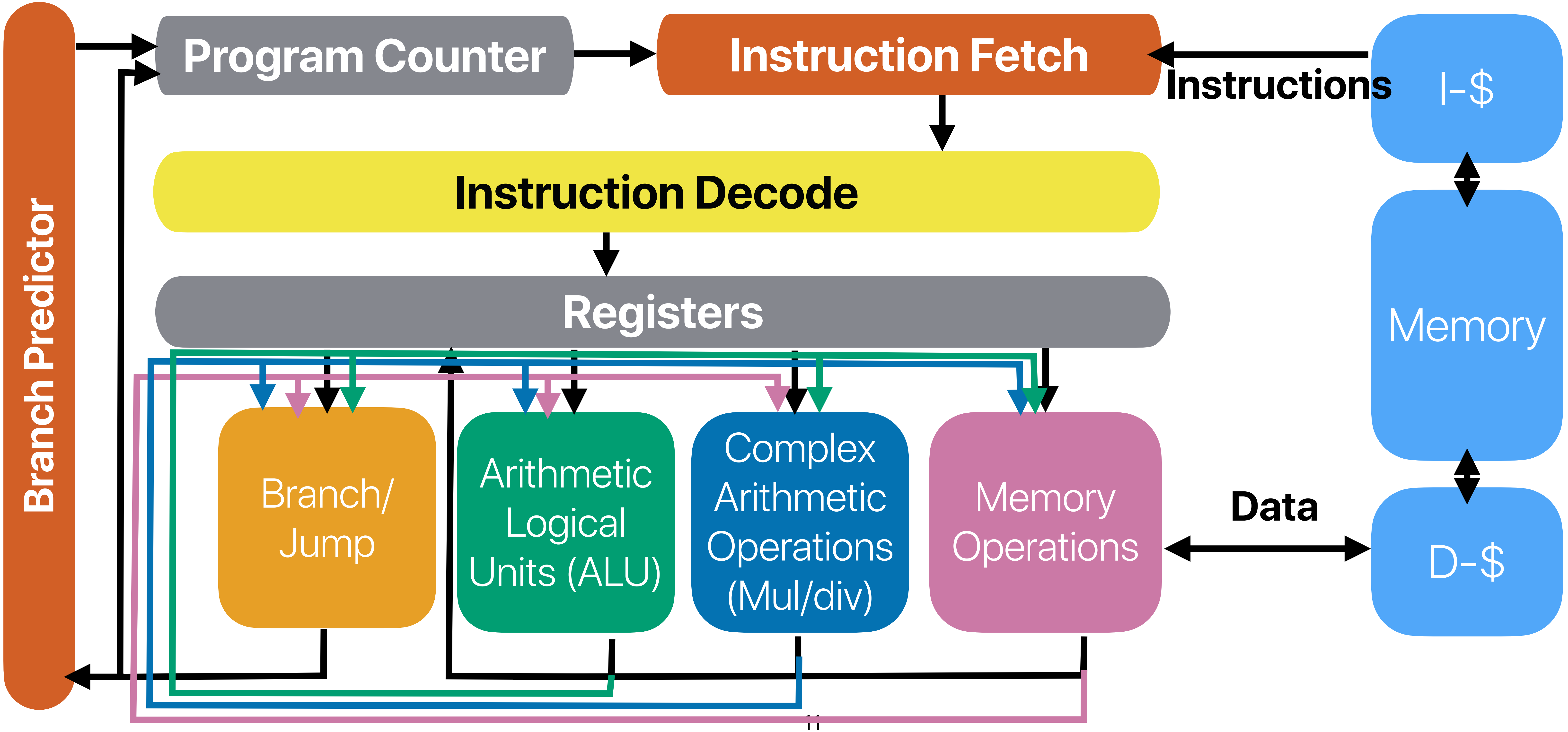
# Can we really execute instructions OoO?

- Exceptions may occur anytime — divided by 0, page fault
  - A later instruction cannot write back its own result otherwise the architectural states won't be correct
  - Instructions after the one causes the exception should not be executed
- Hardware can schedule instruction across branch instructions with the help of branch prediction
  - Fetch instructions according to the branch prediction
  - However, branch predictor can never be perfect

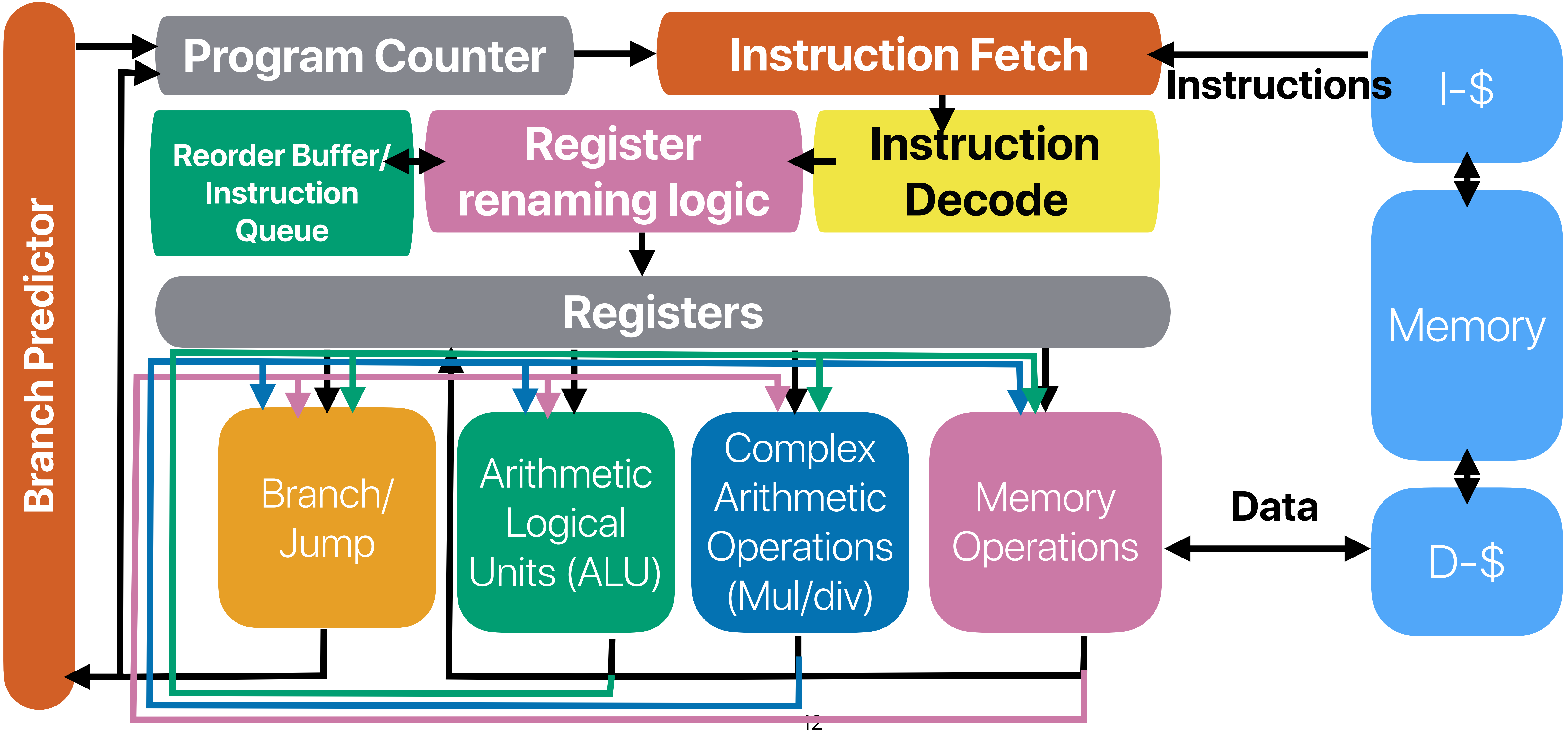
# Speculative Execution

- **Speculative** execution mode: an executing instruction is considered as **speculative** before the processor hasn't determined if the instruction should be executed or not
- Reorder buffer (ROB)
  - The processor allocates an entry for each instruction in a reorder buffer
  - Store results in **reorder buffer and physical registers** when the instruction is still speculative
  - If an earlier instruction failed to commit due to an exception or mis-prediction, the physical registers and all ROB entries after the failed-to-commit instruction are flushed
- Commit/Retire
  - Present the execution result to the running program and in architectural registers when all prior instructions are non-speculative
  - Release the ROB entry

# Data "forwarding"



# Register renaming + OoO + RoB



# Register renaming

```
① movq (%rdi,%rax), %rsi
② movq (%rcx,%rax), %r8
③ movq %r8, (%rdi,%rax)
④ movq %rsi, (%rcx,%rax)
⑤ addq $8, %rax
⑥ cmpq %r9, %rax
⑦ jne  .L9
⑧ movq (%rdi,%rax), %rsi
⑨ movq (%rcx,%rax), %r8
⑩ movq %r8, (%rdi,%rax)
⑪ movq %rsi, (%rcx,%rax)
⑫ addq $8, %rax
⑬ cmpq %r9, %rax
⑭ jne  .L9
```

Only 1 of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1) | | | | | | | | | | |
| 2 | (2) | (1) | | | | | | | | | |
| 3 | (3) | (2) | (1) | | | | | | | | |
| 4 | | | | | | | | | | | |
| 5 | | | | | | | | | | | |
| 6 | | | | | | | | | | | |
| 7 | | | | | | | | | | | |
| 8 | | | | | | | | | | | |
| 9 | | | | | | | | | | | |
| 10 | | | | | | | | | | | |
| 11 | | | | | | | | | | | |
| 12 | | | | | | | | | | | |
| 13 | | | | | | | | | | | |
| 14 | | | | | | | | | | | |
| 15 | | | | | | | | | | | |
| 16 | | | | | | | | | | | |

| | Physical Register |
|---|---|
| rax | |
| rcx | |
| rdi | |
| rsi | |
| r8 | |

| | Valid | Value | In use | | Valid | Value | In use |
|---|---|---|---|---|---|---|---|
| P1 | | | | P6 | | | |
| P2 | | | | P7 | | | |
| P3 | | | | P8 | | | |
| P4 | | | | P9 | | | |
| P5 | | | | P10 | | | |

# Register renaming

```
①  movq (%rdi,%rax), %rsi → P1
②  movq (%rcx,%rax), %r8
③  movq %r8, (%rdi,%rax)
④  movq %rsi, (%rcx,%rax)
⑤  addq $8, %rax
⑥  cmpq %r9, %rax
⑦  jne  .L9
⑧  movq (%rdi,%rax), %rsi
⑨  movq (%rcx,%rax), %r8
⑩  movq %r8, (%rdi,%rax)
⑪  movq %rsi, (%rcx,%rax)
⑫  addq $8, %rax
⑬  cmpq %r9, %rax
⑭  jne  .L9
```

Only 1 of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1) | | | | | | | | | | |
| 2 | (2) | (1) | | | | | | | | | |
| 3 | (3) | (2) | (1) | | | | | | | | |
| 4 | (4) | (3) | (2) | (1) | | | | | | | |
| 5 | | | | | | | | | | | |
| 6 | | | | | | | | | | | |
| 7 | | | | | | | | | | | |
| 8 | | | | | | | | | | | |
| 9 | | | | | | | | | | | |
| 10 | | | | | | | | | | | |
| 11 | | | | | | | | | | | |
| 12 | | | | | | | | | | | |
| 13 | | | | | | | | | | | |
| 14 | | | | | | | | | | | |
| 15 | | | | | | | | | | | |
| 16 | | | | | | | | | | | |

| | Physical Register |
|---|---|
| rax | |
| rcx | |
| rdi | |
| rsi | P1 |
| r8 | |

| | Valid | Value | In use | | Valid | Value | In use |
|---|---|---|---|---|---|---|---|
| P1 | 0 | | 1 | P6 | | | |
| P2 | | | | P7 | | | |
| P3 | | | | P8 | | | |
| P4 | | | | P9 | | | |
| P5 | | | | P10 | | | |

# Register renaming

① `movq (%rdi,%rax), %rsi` → **P1**
② `movq (%rcx,%rax), %r8` → **P2**
③ `movq %r8, (%rdi,%rax)`
④ `movq %rsi, (%rcx,%rax)`
⑤ `addq $8, %rax`
⑥ `cmpq %r9, %rax`
⑦ `jne .L9`
⑧ `movq (%rdi,%rax), %rsi`
⑨ `movq (%rcx,%rax), %r8`
⑩ `movq %r8, (%rdi,%rax)`
⑪ `movq %rsi, (%rcx,%rax)`
⑫ `addq $8, %rax`
⑬ `cmpq %r9, %rax`
⑭ `jne .L9`

Only 1 of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1) | | | | | | | | | | |
| 2 | (2) | (1) | | | | | | | | | |
| 3 | (3) | (2) | (1) | | | | | | | | |
| 4 | (4) | (3) | (2) | (1) | | | | | | | |
| 5 | (5) | (4) | (3) | (2) | (1) | | | | | | |
| 6 | | | | | | | | | | | |
| 7 | | | | | | | | | | | |
| 8 | | | | | | | | | | | |
| 9 | | | | | | | | | | | |
| 10 | | | | | | | | | | | |
| 11 | | | | | | | | | | | |
| 12 | | | | | | | | | | | |
| 13 | | | | | | | | | | | |
| 14 | | | | | | | | | | | |
| 15 | | | | | | | | | | | |
| 16 | | | | | | | | | | | |

| | Physical Register |
|---|---|
| rax | |
| rcx | |
| rdi | |
| rsi | P1 |
| r8 | P2 |

| | Valid | Value | In use | | Valid | Value | In use |
|---|---|---|---|---|---|---|---|
| P1 | 0 | | 1 | P6 | | | |
| P2 | 0 | | 1 | P7 | | | |
| P3 | | | | P8 | | | |
| P4 | | | | P9 | | | |
| P5 | | | | P10 | | | |

# Register renaming

```
① movq (%rdi,%rax), %rsi → P1
② movq (%rcx,%rax), %r8  → P2
③ movq %r8, (%rdi,%rax)
④ movq %rsi, (%rcx,%rax)
⑤ addq $8, %rax
⑥ cmpq %r9, %rax
⑦ jne  .L9
⑧ movq (%rdi,%rax), %rsi
⑨ movq (%rcx,%rax), %r8
⑩ movq %r8, (%rdi,%rax)
⑪ movq %rsi, (%rcx,%rax)
⑫ addq $8, %rax
⑬ cmpq %r9, %rax
⑭ jne  .L9
```

Only 1 of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1) | | | | | | | | | | |
| 2 | (2) | (1) | | | | | | | | | |
| 3 | (3) | (2) | (1) | | | | | | | | |
| 4 | (4) | (3) | (2) | (1) | | | | | | | |
| 5 | (5) | (4) | (3) | (2) | (1) | | | | | | |
| 6 | (6) | (5) | (3)(4) | | (2) | (1) | | | | | |
| 7 | | | | | | | | | | | |
| 8 | | | | | | | | | | | |
| 9 | | | | | | | | | | | |
| 10 | | | | | | | | | | | |
| 11 | | | | | | | | | | | |
| 12 | | | | | | | | | | | |
| 13 | | | | | | | | | | | |
| 14 | | | | | | | | | | | |
| 15 | | | | | | | | | | | |
| 16 | | | | | | | | | | | |

| | Physical Register |
|---|---|
| rax | |
| rcx | |
| rdi | |
| rsi | P1 |
| r8 | P2 |

| | Valid | Value | In use | | Valid | Value | In use |
|---|---|---|---|---|---|---|---|
| P1 | 0 | | 1 | P6 | | | |
| P2 | 0 | | 1 | P7 | | | |
| P3 | | | | P8 | | | |
| P4 | | | | P9 | | | |
| P5 | | | | P10 | | | |

# Register renaming

```
①  movq (%rdi,%rax), %rsi  → P1
②  movq (%rcx,%rax), %r8   → P2
③  movq %r8, (%rdi,%rax)
④  movq %rsi, (%rcx,%rax)
⑤  addq $8, %rax           → P3
⑥  cmpq %r9, %rax
⑦  jne  .L9
⑧  movq (%rdi,%rax), %rsi
⑨  movq (%rcx,%rax), %r8
⑩  movq %r8, (%rdi,%rax)
⑪  movq %rsi, (%rcx,%rax)
⑫  addq $8, %rax
⑬  cmpq %r9, %rax
⑭  jne  .L9
```

Only 1 of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1) | | | | | | | | | | |
| 2 | (2) | (1) | | | | | | | | | |
| 3 | (3) | (2) | (1) | | | | | | | | |
| 4 | (4) | (3) | (2) | (1) | | | | | | | |
| 5 | (5) | (4) | (3) | (2) | (1) | | | | | | |
| 6 | (6) | (5) | (3)(4) | | (2) | (1) | | | | | |
| 7 | (7) | (6) | (3)(4)(5) | | | (2) | (1) | | | | |
| 8 | | | | | | | | | | | |
| 9 | | | | | | | | | | | |
| 10 | | | | | | | | | | | |
| 11 | | | | | | | | | | | |
| 12 | | | | | | | | | | | |
| 13 | | | | | | | | | | | |
| 14 | | | | | | | | | | | |
| 15 | | | | | | | | | | | |
| 16 | | | | | | | | | | | |

| | Physical Register |
|---|---|
| rax | P3 |
| rcx | |
| rdi | |
| rsi | P1 |
| r8 | P2 |

| | Valid | Value | In use | | Valid | Value | In use |
|---|---|---|---|---|---|---|---|
| P1 | 0 | | 1 | P6 | | | |
| P2 | 0 | | 1 | P7 | | | |
| P3 | 0 | | 1 | P8 | | | |
| P4 | | | | P9 | | | |
| P5 | | | | P10 | | | |

# Register renaming

```
① movq (%rdi,%rax), %rsi → P1
② movq (%rcx,%rax), %r8  → P2
③ movq %r8, (%rdi,%rax)
④ movq %rsi, (%rcx,%rax)
⑤ addq $8, %rax          → P3
⑥ cmpq %r9, %rax
⑦ jne  .L9
⑧ movq (%rdi,%rax), %rsi
⑨ movq (%rcx,%rax), %r8
⑩ movq %r8, (%rdi,%rax)
⑪ movq %rsi, (%rcx,%rax)
⑫ addq $8, %rax
⑬ cmpq %r9, %rax
⑭ jne  .L9
```

Only 1 of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1) | | | | | | | | | | |
| 2 | (2) | (1) | | | | | | | | | |
| 3 | (3) | (2) | (1) | | | | | | | | |
| 4 | (4) | (3) | (2) | (1) | | | | | | | |
| 5 | (5) | (4) | (3) | (2) | (1) | | | | | | |
| 6 | (6) | (5) | (3)(4) | | (2) | (1) | | | | | |
| 7 | (7) | (6) | (3)(4)(5) | | | (2) | (1) | | | | |
| 8 | (8) | (7) | (3)(5)(6) | (4) | | | (2) | | | | (1) |
| 9 | | | | | | | | | | | |
| 10 | | | | | | | | | | | |
| 11 | | | | | | | | | | | |
| 12 | | | | | | | | | | | |
| 13 | | | | | | | | | | | |
| 14 | | | | | | | | | | | |
| 15 | | | | | | | | | | | |
| 16 | | | | | | | | | | | |

Instruction (4) is running ahead of (3)

| | Physical Register |
|---|---|
| rax | P3 |
| rcx | |
| rdi | |
| rsi | P1 |
| r8 | P2 |

| | Valid | Value | In use | | Valid | Value | In use |
|---|---|---|---|---|---|---|---|
| P1 | 1 | | 1 | P6 | | | |
| P2 | 0 | | 1 | P7 | | | |
| P3 | 0 | | 1 | P8 | | | |
| P4 | | | | P9 | | | |
| P5 | | | | P10 | | | |

# Register renaming

```
①  movq (%rdi,%rax), %rsi  → P1
②  movq (%rcx,%rax), %r8   → P2
③  movq %r8, (%rdi,%rax)
④  movq %rsi, (%rcx,%rax)
⑤  addq $8, %rax           → P3
⑥  cmpq %r9, %rax
⑦  jne  .L9
⑧  movq (%rdi,%rax), %rsi
⑨  movq (%rcx,%rax), %r8
⑩  movq %r8, (%rdi,%rax)
⑪  movq %rsi, (%rcx,%rax)
⑫  addq $8, %rax
⑬  cmpq %r9, %rax
⑭  jne  .L9
```

Only 1 of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1) | | | | | | | | | | |
| 2 | (2) | (1) | | | | | | | | | |
| 3 | (3) | (2) | (1) | | | | | | | | |
| 4 | (4) | (3) | (2) | (1) | | | | | | | |
| 5 | (5) | (4) | (3) | (2) | (1) | | | | | | |
| 6 | (6) | (5) | (3)(4) | | (2) | (1) | | | | | |
| 7 | (7) | (6) | (3)(4)(5) | | | (2) | (1) | | | | |
| 8 | (8) | (7) | (3)(5)(6) | (4) | | | (2) | | | | ~~(1)~~ |
| 9 | (9) | (8) | (5)(6)(7) | (3) | (4) | | | | | | (2) |
| 10 | | | | | | | | | | | |
| 11 | | | | | | | | | | | |
| 12 | | | | | | | | | | | |
| 13 | | | | | | | | | | | |
| 14 | | | | | | | | | | | |
| 15 | | | | | | | | | | | |
| 16 | | | | | | | | | | | |

| | Physical Register |
|---|---|
| rax | P3 |
| rcx | |
| rdi | |
| rsi | P1 |
| r8 | P2 |

| | Valid | Value | In use | | Valid | Value | In use |
|---|---|---|---|---|---|---|---|
| P1 | 1 | | 1 | P6 | | | |
| P2 | 1 | | 1 | P7 | | | |
| P3 | 0 | | 1 | P8 | | | |
| P4 | | | | P9 | | | |
| P5 | | | | P10 | | | |

# Register renaming

```
① movq (%rdi,%rax), %rsi   → P1
② movq (%rcx,%rax), %r8    → P2
③ movq %r8, (%rdi,%rax)
④ movq %rsi, (%rcx,%rax)
⑤ addq $8, %rax            → P3
⑥ cmpq %r9, %rax
⑦ jne  .L9
⑧ movq (%rdi,%rax), %rsi
⑨ movq (%rcx,%rax), %r8
⑩ movq %r8, (%rdi,%rax)
⑪ movq %rsi, (%rcx,%rax)
⑫ addq $8, %rax
⑬ cmpq %r9, %rax
⑭ jne  .L9
```

Only 1 of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1) | | | | | | | | | | |
| 2 | (2) | (1) | | | | | | | | | |
| 3 | (3) | (2) | (1) | | | | | | | | |
| 4 | (4) | (3) | (2) | (1) | | | | | | | |
| 5 | (5) | (4) | (3) | (2) | (1) | | | | | | |
| 6 | (6) | (5) | (3)(4) | | (2) | (1) | | | | | |
| 7 | (7) | (6) | (3)(4)(5) | | | (2) | (1) | | | | |
| 8 | (8) | (7) | (3)(5)(6) | (4) | | | (2) | | | | ~~(1)~~ |
| 9 | (9) | (8) | (5)(6)(7) | (3) | (4) | | | | | | ~~(2)~~ |
| 10 | | | | | | | | | | | |
| 11 | | | | | | | | | | | |
| 12 | | | | | | | | | | | |
| 13 | | | | | | | | | | | |
| 14 | | | | | | | | | | | |
| 15 | | | | | | | | | | | |
| 16 | | | | | | | | | | | |

| | Physical Register |
|---|---|
| rax | P3 |
| rcx | |
| rdi | |
| rsi | P1 |
| r8 | P2 |

| | Valid | Value | In use | | Valid | Value | In use |
|---|---|---|---|---|---|---|---|
| P1 | 1 | | 1 | P6 | | | |
| P2 | 1 | | 1 | P7 | | | |
| P3 | 0 | | 1 | P8 | | | |
| P4 | | | | P9 | | | |
| P5 | | | | P10 | | | |

# Register renaming

```
①  movq (%rdi,%rax), %rsi → P1
②  movq (%rcx,%rax), %r8  → P2
③  movq %r8, (%rdi,%rax)
④  movq %rsi, (%rcx,%rax)
⑤  addq $8, %rax          → P3
⑥  cmpq %r9, %rax
⑦  jne  .L9
⑧  movq (%rdi,%rax), %rsi → P4
⑨  movq (%rcx,%rax), %r8
⑩  movq %r8, (%rdi,%rax)
⑪  movq %rsi, (%rcx,%rax)
⑫  addq $8, %rax
⑬  cmpq %r9, %rax
⑭  jne  .L9
```

Only 1 of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1) | | | | | | | | | | |
| 2 | (2) | (1) | | | | | | | | | |
| 3 | (3) | (2) | (1) | | | | | | | | |
| 4 | (4) | (3) | (2) | (1) | | | | | | | |
| 5 | (5) | (4) | (3) | (2) | (1) | | | | | | |
| 6 | (6) | (5) | (3)(4) | | (2) | (1) | | | | | |
| 7 | (7) | (6) | (3)(4)(5) | | | (2) | (1) | | | | |
| 8 | (8) | (7) | (3)(5)(6) | (4) | | | (2) | | | | ~~(1)~~ |
| 9 | (9) | (8) | (5)(6)(7) | (3) | (4) | | | | | | ~~(2)~~ |
| 10 | (10) | (9) | (6)(7)(8) | | (3) | (4) | | (5) | | | |
| 11 | | | | | | | | | | | |
| 12 | | | | | | | | | | | |
| 13 | | | | | | | | | | | |
| 14 | | | | | | | | | | | |
| 15 | | | | | | | | | | | |
| 16 | | | | | | | | | | | |

| | Physical Register |
|---|---|
| rax | P3 |
| rcx | |
| rdi | |
| rsi | P4 |
| r8 | P2 |

| | Valid | Value | In use | | Valid | Value | In use |
|---|---|---|---|---|---|---|---|
| P1 | 1 | | 1 | P6 | | | |
| P2 | 1 | | 1 | P7 | | | |
| P3 | 0 | | 1 | P8 | | | |
| P4 | 0 | | 1 | P9 | | | |
| P5 | | | | P10 | | | |

# Register renaming

```
①  movq (%rdi,%rax), %rsi → P1
②  movq (%rcx,%rax), %r8  → P2
③  movq %r8, (%rdi,%rax)
④  movq %rsi, (%rcx,%rax)
⑤  addq $8, %rax          → P3
⑥  cmpq %r9, %rax
⑦  jne  .L9
⑧  movq (%rdi,%rax), %rsi → P4
⑨  movq (%rcx,%rax), %r8  → P5
⑩  movq %r8, (%rdi,%rax)
⑪  movq %rsi, (%rcx,%rax)
⑫  addq $8, %rax
⑬  cmpq %r9, %rax
⑭  jne  .L9
```

Only 1 of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1) | | | | | | | | | | |
| 2 | (2) | (1) | | | | | | | | | |
| 3 | (3) | (2) | (1) | | | | | | | | |
| 4 | (4) | (3) | (2) | (1) | | | | | | | |
| 5 | (5) | (4) | (3) | (2) | (1) | | | | | | |
| 6 | (6) | (5) | (3)(4) | | (2) | (1) | | | | | |
| 7 | (7) | (6) | (3)(4)(5) | | | (2) | (1) | | | | |
| 8 | (8) | (7) | (3)(5)(6) | (4) | | | (2) | | | | ~~(1)~~ |
| 9 | (9) | (8) | (5)(6)(7) | (3) | (4) | | | | | | ~~(2)~~ |
| 10 | (10) | (9) | (6)(7)(8) | | (3) | (4) | | (5) | | | |
| 11 | (11) | (10) | (7)(8)(9) | | | (3) | (4) | (6) | | | (5) |
| 12 | | | | | | | | | | | |
| 13 | | | | | | | | | | | |
| 14 | | | | | | | | | | | |
| 15 | | | | | | | | | | | |
| 16 | | | | | | | | | | | |

| | Physical Register |
|---|---|
| rax | P3 |
| rcx | |
| rdi | |
| rsi | P4 |
| r8 | P5 |

| | Valid | Value | In use | | Valid | Value | In use |
|---|---|---|---|---|---|---|---|
| P1 | 1 | | 1 | P6 | | | |
| P2 | 1 | | 1 | P7 | | | |
| P3 | 1 | | 1 | P8 | | | |
| P4 | 0 | | 1 | P9 | | | |
| P5 | 0 | | 1 | P10 | | | |

# Register renaming

```
① movq (%rdi,%rax), %rsi → P1
② movq (%rcx,%rax), %r8  → P2
③ movq %r8, (%rdi,%rax)
④ movq %rsi, (%rcx,%rax)
⑤ addq $8, %rax          → P3
⑥ cmpq %r9, %rax
⑦ jne  .L9
⑧ movq (%rdi,%rax), %rsi → P4
⑨ movq (%rcx,%rax), %r8  → P5
⑩ movq %r8, (%rdi,%rax)
⑪ movq %rsi, (%rcx,%rax)
⑫ addq $8, %rax
⑬ cmpq %r9, %rax
⑭ jne  .L9
```

Only 1 of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1) | | | | | | | | | | |
| 2 | (2) | (1) | | | | | | | | | |
| 3 | (3) | (2) | (1) | | | | | | | | |
| 4 | (4) | (3) | (2) | (1) | | | | | | | |
| 5 | (5) | (4) | (3) | (2) | (1) | | | | | | |
| 6 | (6) | (5) | (3)(4) | | (2) | (1) | | | | | |
| 7 | (7) | (6) | (3)(4)(5) | | | (2) | (1) | | | | |
| 8 | (8) | (7) | (3)(5)(6) | (4) | | | (2) | | | | ~~(1)~~ |
| 9 | (9) | (8) | (5)(6)(7) | (3) | (4) | | | | | | ~~(2)~~ |
| 10 | (10) | (9) | (6)(7)(8) | | (3) | (4) | | (5) | | | |
| 11 | (11) | (10) | (7)(8)(9) | | | (3) | (4) | (6) | | | (5) |
| 12 | (12) | (11) | (9)(10) | | | | (3) | | | (7) | (4)(5)(6) |
| 13 | | | | | | | | | | | |
| 14 | | | | | | | | | | | |
| 15 | | | | | | | | | | | |
| 16 | | | | | | | | | | | |

| | Physical Register |
|---|---|
| rax | P3 |
| rcx | |
| rdi | |
| rsi | P4 |
| r8 | P5 |

| | Valid | Value | In use | | Valid | Value | In use |
|---|---|---|---|---|---|---|---|
| P1 | 1 | | 1 | P6 | | | |
| P2 | 1 | | 1 | P7 | | | |
| P3 | 1 | | 1 | P8 | | | |
| P4 | 0 | | 1 | P9 | | | |
| P5 | 0 | | 1 | P10 | | | |

# Register renaming

```
①  movq (%rdi,%rax), %rsi → P1
②  movq (%rcx,%rax), %r8  → P2
③  movq %r8, (%rdi,%rax)
④  movq %rsi, (%rcx,%rax)
⑤  addq $8, %rax          → P3
⑥  cmpq %r9, %rax
⑦  jne  .L9
⑧  movq (%rdi,%rax), %rsi → P4
⑨  movq (%rcx,%rax), %r8  → P5
⑩  movq %r8, (%rdi,%rax)
⑪  movq %rsi, (%rcx,%rax)
⑫  addq $8, %rax
⑬  cmpq %r9, %rax
⑭  jne  .L9
```

Only 1 of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1) | | | | | | | | | | |
| 2 | (2) | (1) | | | | | | | | | |
| 3 | (3) | (2) | (1) | | | | | | | | |
| 4 | (4) | (3) | (2) | (1) | | | | | | | |
| 5 | (5) | (4) | (3) | (2) | (1) | | | | | | |
| 6 | (6) | (5) | (3)(4) | | (2) | (1) | | | | | |
| 7 | (7) | (6) | (3)(4)(5) | | | (2) | (1) | | | | |
| 8 | (8) | (7) | (3)(5)(6) | (4) | | | (2) | | | | ~~(1)~~ |
| 9 | (9) | (8) | (5)(6)(7) | (3) | (4) | | | | | | ~~(2)~~ |
| 10 | (10) | (9) | (6)(7)(8) | | (3) | (4) | | (5) | | | |
| 11 | (11) | (10) | (7)(8)(9) | | | (3) | (4) | (6) | | | (5) |
| 12 | (12) | (11) | (9)(10) | | | | (3) | | | (7) | (4)(5)(6) |
| 13 | (13) | (12) | (10)(11) | (8) | | | | | | | (3)(4)(5)(6)(7) |
| 14 | | | | | | | | | | | |
| 15 | | | | | | | | | | | |
| 16 | | | | | | | | | | | |

| | Physical Register |
|---|---|
| rax | P3 |
| rcx | |
| rdi | |
| rsi | P4 |
| r8 | P5 |

| | Valid | Value | In use | | Valid | Value | In use |
|---|---|---|---|---|---|---|---|
| P1 | 1 | | 1 | P6 | | | |
| P2 | 1 | | 1 | P7 | | | |
| P3 | 1 | | 1 | P8 | | | |
| P4 | 0 | | 1 | P9 | | | |
| P5 | 0 | | 1 | P10 | | | |

# Register renaming

```
① movq (%rdi,%rax), %rsi → P1
② movq (%rcx,%rax), %r8  → P2
③ movq %r8, (%rdi,%rax)
④ movq %rsi, (%rcx,%rax)
⑤ addq $8, %rax          → P3
⑥ cmpq %r9, %rax
⑦ jne  .L9
⑧ movq (%rdi,%rax), %rsi → P4
⑨ movq (%rcx,%rax), %r8  → P5
⑩ movq %r8, (%rdi,%rax)
⑪ movq %rsi, (%rcx,%rax)
⑫ addq $8, %rax
⑬ cmpq %r9, %rax
⑭ jne  .L9
```

Only 1 of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1) | | | | | | | | | | |
| 2 | (2) | (1) | | | | | | | | | |
| 3 | (3) | (2) | (1) | | | | | | | | |
| 4 | (4) | (3) | (2) | (1) | | | | | | | |
| 5 | (5) | (4) | (3) | (2) | (1) | | | | | | |
| 6 | (6) | (5) | (3)(4) | | (2) | (1) | | | | | |
| 7 | (7) | (6) | (3)(4)(5) | | | (2) | (1) | | | | |
| 8 | (8) | (7) | (3)(5)(6) | (4) | | | (2) | | | | ~~(1)~~ |
| 9 | (9) | (8) | (5)(6)(7) | (3) | (4) | | | | | | ~~(2)~~ |
| 10 | (10) | (9) | (6)(7)(8) | | (3) | (4) | | (5) | | | |
| 11 | (11) | (10) | (7)(8)(9) | | | (3) | (4) | (6) | | | (5) |
| 12 | (12) | (11) | (9)(10) | | | | (3) | | | (7) | (4)(5)(6) |
| 13 | (13) | (12) | (10)(11) | (8) | | | | | | | ~~(3)(4)(5)(6)(7)~~ |
| 14 | | | | (9) | (8) | | | | | | |
| 15 | | | | | | | | | | | |
| 16 | | | | | | | | | | | |

| | Physical Register |
|---|---|
| rax | P3 |
| rcx | |
| rdi | |
| rsi | P4 |
| r8 | P5 |

| | Valid | Value | In use | | Valid | Value | In use |
|---|---|---|---|---|---|---|---|
| P1 | 1 | | 1 | P6 | | | |
| P2 | 1 | | 1 | P7 | | | |
| P3 | 1 | | 1 | P8 | | | |
| P4 | 0 | | 1 | P9 | | | |
| P5 | 0 | | 1 | P10 | | | |

# Register renaming

```
① movq (%rdi,%rax), %rsi  → P1
② movq (%rcx,%rax), %r8   → P2
③ movq %r8, (%rdi,%rax)
④ movq %rsi, (%rcx,%rax)
⑤ addq $8, %rax           → P3
⑥ cmpq %r9, %rax
⑦ jne  .L9
⑧ movq (%rdi,%rax), %rsi  → P4
⑨ movq (%rcx,%rax), %r8   → P5
⑩ movq %r8, (%rdi,%rax)
⑪ movq %rsi, (%rcx,%rax)
⑫ addq $8, %rax           → P6
⑬ cmpq %r9, %rax
⑭ jne  .L9
```

Only 1 of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1) | | | | | | | | | | |
| 2 | (2) | (1) | | | | | | | | | |
| 3 | (3) | (2) | (1) | | | | | | | | |
| 4 | (4) | (3) | (2) | (1) | | | | | | | |
| 5 | (5) | (4) | (3) | (2) | (1) | | | | | | |
| 6 | (6) | (5) | (3)(4) | | (2) | (1) | | | | | |
| 7 | (7) | (6) | (3)(4)(5) | | | (2) | (1) | | | | |
| 8 | (8) | (7) | (3)(5)(6) | (4) | | | (2) | | | | ~~(1)~~ |
| 9 | (9) | (8) | (5)(6)(7) | (3) | (4) | | | | | | ~~(2)~~ |
| 10 | (10) | (9) | (6)(7)(8) | | (3) | (4) | | (5) | | | |
| 11 | (11) | (10) | (7)(8)(9) | | | (3) | (4) | (6) | | | (5) |
| 12 | (12) | (11) | (9)(10) | | | | (3) | | | (7) | (4)(5)(6) |
| 13 | (13) | (12) | (10)(11) | (8) | | | | | | | ~~(3)(4)(5)(6)(7)~~ |
| 14 | (14) | (13) | (10)(11)(12) | (9) | (8) | | | | | | |
| 15 | | (14) | (10)(11)(12)(13) | | (9) | (8) | | | | | |
| 16 | | | | | | | | | | | |

| | Physical Register |
|---|---|
| rax | P6 |
| rcx | |
| rdi | |
| rsi | P4 |
| r8 | P5 |

| | Valid | Value | In use | | Valid | Value | In use |
|---|---|---|---|---|---|---|---|
| P1 | 1 | | 1 | P6 | 0 | | 1 |
| P2 | 1 | | 1 | P7 | | | |
| P3 | 1 | | 1 | P8 | | | |
| P4 | 0 | | 1 | P9 | | | |
| P5 | 0 | | 1 | P10 | | | |

# Register renaming

```
① movq (%rdi,%rax), %rsi → P1
② movq (%rcx,%rax), %r8  → P2
③ movq %r8, (%rdi,%rax)
④ movq %rsi, (%rcx,%rax)
⑤ addq $8, %rax          → P3
⑥ cmpq %r9, %rax
⑦ jne  .L9
⑧ movq (%rdi,%rax), %rsi → P4
⑨ movq (%rcx,%rax), %r8  → P5
⑩ movq %r8, (%rdi,%rax)
⑪ movq %rsi, (%rcx,%rax)
⑫ addq $8, %rax          → P6
⑬ cmpq %r9, %rax
⑭ jne  .L9
```

Only 1 of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1) | | | | | | | | | | |
| 2 | (2) | (1) | | | | | | | | | |
| 3 | (3) | (2) | (1) | | | | | | | | |
| 4 | (4) | (3) | (2) | (1) | | | | | | | |
| 5 | (5) | (4) | (3) | (2) | (1) | | | | | | |
| 6 | (6) | (5) | (3)(4) | | (2) | (1) | | | | | |
| 7 | (7) | (6) | (3)(4)(5) | | | (2) | (1) | | | | |
| 8 | (8) | (7) | (3)(5)(6) | (4) | | | (2) | | | | ~~(1)~~ |
| 9 | (9) | (8) | (5)(6)(7) | (3) | (4) | | | | | | ~~(2)~~ |
| 10 | (10) | (9) | (6)(7)(8) | | (3) | (4) | | (5) | | | |
| 11 | (11) | (10) | (7)(8)(9) | | | (3) | (4) | (6) | | | (5) |
| 12 | (12) | (11) | (9)(10) | | | | (3) | | | (7) | (4)(5)(6) |
| 13 | (13) | (12) | (10)(11) | (8) | | | | | | | ~~(3)(4)(5)(6)(7)~~ |
| 14 | (14) | (13) | (10)(11)(12) | (9) | (8) | | | | | | |
| 15 | | (14) | (10)(11)(12)(13) | | (9) | (8) | | | | | |
| 16 | | | | | | (9) | (8) | | | | |

| | Physical Register |
|---|---|
| rax | P6 |
| rcx | |
| rdi | |
| rsi | P4 |
| r8 | P5 |

| | Valid | Value | In use | | Valid | Value | In use |
|---|---|---|---|---|---|---|---|
| P1 | 1 | | 1 | P6 | 0 | | 1 |
| P2 | 1 | | 1 | P7 | | | |
| P3 | 1 | | 1 | P8 | | | |
| P4 | 0 | | 1 | P9 | | | |
| P5 | 0 | | 1 | P10 | | | |

# Register renaming

```
①  movq (%rdi,%rax), %rsi  → P1
②  movq (%rcx,%rax), %r8   → P2
③  movq %r8, (%rdi,%rax)
④  movq %rsi, (%rcx,%rax)
⑤  addq $8, %rax           → P3
⑥  cmpq %r9, %rax
⑦  jne  .L9
⑧  movq (%rdi,%rax), %rsi  → P4
⑨  movq (%rcx,%rax), %r8   → P5
⑩  movq %r8, (%rdi,%rax)
⑪  movq %rsi, (%rcx,%rax)
⑫  addq $8, %rax           → P6
⑬  cmpq %r9, %rax
⑭  jne  .L9
⑮  movq (%rdi,%rax), %rsi
⑯  movq (%rcx,%rax), %r8
⑰  movq %r8, (%rdi,%rax)
⑱  movq %rsi, (%rcx,%rax)
⑲  addq $8, %rax
⑳  cmpq %r9, %rax
㉑  jne  .L9
```

Only 1 of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1) | | | | | | | | | | |
| 2 | (2) | (1) | | | | | | | | | |
| 3 | (3) | (2) | (1) | | | | | | | | |
| 4 | (4) | (3) | (2) | (1) | | | | | | | |
| 5 | (5) | (4) | (3) | (2) | (1) | | | | | | |
| 6 | (6) | (5) | (3)(4) | | (2) | (1) | | | | | |
| 7 | (7) | (6) | (3)(4)(5) | | | (2) | (1) | | | | |
| 8 | (8) | (7) | (3)(5)(6) | (4) | | | (2) | | | | ~~(1)~~ |
| 9 | (9) | (8) | (5)(6)(7) | (3) | (4) | | | | | | ~~(2)~~ |
| 10 | (10) | (9) | (6)(7)(8) | | (3) | (4) | | (5) | | | |
| 11 | (11) | (10) | (7)(8)(9) | | | (3) | (4) | (6) | | | (5) |
| 12 | (12) | (11) | (9)(10) | | | | (3) | | | (7) | (4)(5)(6) |
| 13 | (13) | (12) | (10)(11) | (8) | | | | | | | ~~(3)(4)(5)(6)(7)~~ |
| 14 | (14) | (13) | (10)(11)(12) | (9) | (8) | | | | | | |
| 15 | (15) | (14) | (10)(11)(13) | | (9) | (8) | | (12) | | | |
| 16 | (16) | (15) | (10)(11)(14) | | | (9) | (8) | (13) | | | (12) |
| 17 | (17) | (16) | (10)(14)(15) | (11) | | | (9) | | | | ~~(8)~~(12)(13) |

# Register renaming

```
1  movq (%rdi,%rax), %rsi   → P1
2  movq (%rcx,%rax), %r8    → P2
3  movq %r8, (%rdi,%rax)
4  movq %rsi, (%rcx,%rax)
5  addq $8, %rax            → P3
6  cmpq %r9, %rax
7  jne  .L9
8  movq (%rdi,%rax), %rsi   → P4
9  movq (%rcx,%rax), %r8    → P5
10 movq %r8, (%rdi,%rax)
11 movq %rsi, (%rcx,%rax)
12 addq $8, %rax            → P6
13 cmpq %r9, %rax
14 jne  .L9
15 movq (%rdi,%rax), %rsi
16 movq (%rcx,%rax), %r8
17 movq %r8, (%rdi,%rax)
18 movq %rsi, (%rcx,%rax)
19 addq $8, %rax
20 cmpq %r9, %rax
21 jne  .L9
```

Only 1 of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1) | | | | | | | | | | |
| 2 | (2) | (1) | | | | | | | | | |
| 3 | (3) | (2) | (1) | | | | | | | | |
| 4 | (4) | (3) | (2) | (1) | | | | | | | |
| 5 | (5) | (4) | (3) | (2) | (1) | | | | | | |
| 6 | (6) | (5) | (3)(4) | | (2) | (1) | | | | | |
| 7 | (7) | (6) | (3)(4)(5) | | | (2) | (1) | | | | |
| 8 | (8) | (7) | (3)(5)(6) | (4) | | | (2) | | | | ~~(1)~~ |
| 9 | (9) | (8) | (5)(6)(7) | (3) | (4) | | | | | | ~~(2)~~ |
| 10 | (10) | (9) | (6)(7)(8) | | (3) | (4) | | (5) | | | |
| 11 | (11) | (10) | (7)(8)(9) | | | (3) | (4) | (6) | | | (5) |
| 12 | (12) | (11) | (9)(10) | | | | (3) | | | (7) | (4)(5)(6) |
| 13 | (13) | (12) | (10)(11) | (8) | | | | | | | ~~(3)(4)(5)(6)(7)~~ |
| 14 | (14) | (13) | (10)(11)(12) | (9) | (8) | | | | | | |
| 15 | (15) | (14) | (10)(11)(13) | | (9) | (8) | | (12) | | | |
| 16 | (16) | (15) | (10)(11)(14) | | | (9) | (8) | (13) | | | (12) |
| 17 | (17) | (16) | (10)(14)(15) | (11) | | | (9) | | | | ~~(8)~~(12)(13) |
| 18 | (18) | (17) | (14)(15)(16) | (10) | (11) | | | | | | ~~(9)~~(12)(13) |

# Register renaming

```
① movq (%rdi,%rax), %rsi → P1
② movq (%rcx,%rax), %r8  → P2
③ movq %r8, (%rdi,%rax)
④ movq %rsi, (%rcx,%rax)
⑤ addq $8, %rax          → P3
⑥ cmpq %r9, %rax
⑦ jne  .L9
⑧ movq (%rdi,%rax), %rsi → P4
⑨ movq (%rcx,%rax), %r8  → P5
⑩ movq %r8, (%rdi,%rax)
⑪ movq %rsi, (%rcx,%rax)
⑫ addq $8, %rax          → P6
⑬ cmpq %r9, %rax
⑭ jne  .L9
⑮ movq (%rdi,%rax), %rsi
⑯ movq (%rcx,%rax), %r8
⑰ movq %r8, (%rdi,%rax)
⑱ movq %rsi, (%rcx,%rax)
⑲ addq $8, %rax
⑳ cmpq %r9, %rax
㉑ jne  .L9
```

Only 1 of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1) | | | | | | | | | | |
| 2 | (2) | (1) | | | | | | | | | |
| 3 | (3) | (2) | (1) | | | | | | | | |
| 4 | (4) | (3) | (2) | (1) | | | | | | | |
| 5 | (5) | (4) | (3) | (2) | (1) | | | | | | |
| 6 | (6) | (5) | (3)(4) | | (2) | (1) | | | | | |
| 7 | (7) | (6) | (3)(4)(5) | | | (2) | (1) | | | | |
| 8 | (8) | (7) | (3)(5)(6) | (4) | | | (2) | | | | ~~(1)~~ |
| 9 | (9) | (8) | (5)(6)(7) | (3) | (4) | | | | | | ~~(2)~~ |
| 10 | (10) | (9) | (6)(7)(8) | | (3) | (4) | | (5) | | | |
| 11 | (11) | (10) | (7)(8)(9) | | | (3) | (4) | (6) | | | (5) |
| 12 | (12) | (11) | (9)(10) | | | | (3) | | | (7) | (4)(5)(6) |
| 13 | (13) | (12) | (10)(11) | (8) | | | | | | | ~~(3)(4)(5)(6)(7)~~ |
| 14 | (14) | (13) | (10)(11)(12) | (9) | (8) | | | | | | |
| 15 | (15) | (14) | (10)(11)(13) | | (9) | (8) | | (12) | | | |
| 16 | (16) | (15) | (10)(11)(14) | | | (9) | (8) | (13) | | | (12) |
| 17 | (17) | (16) | (10)(14)(15) | (11) | | | (9) | | | | ~~(8)~~(12)(13) |
| 18 | (18) | (17) | (14)(15)(16) | (10) | (11) | | | | | | ~~(9)~~(12)(13) |
| 19 | (19) | (18) | (15)(16)(17) | | (10) | (11) | | | | (14) | (12)(13) |

# Register renaming

```
①  movq (%rdi,%rax), %rsi  → P1
②  movq (%rcx,%rax), %r8   → P2
③  movq %r8, (%rdi,%rax)
④  movq %rsi, (%rcx,%rax)
⑤  addq $8, %rax           → P3
⑥  cmpq %r9, %rax
⑦  jne  .L9
⑧  movq (%rdi,%rax), %rsi  → P4
⑨  movq (%rcx,%rax), %r8   → P5
⑩  movq %r8, (%rdi,%rax)
⑪  movq %rsi, (%rcx,%rax)
⑫  addq $8, %rax           → P6
⑬  cmpq %r9, %rax
⑭  jne  .L9
⑮  movq (%rdi,%rax), %rsi
⑯  movq (%rcx,%rax), %r8
⑰  movq %r8, (%rdi,%rax)
⑱  movq %rsi, (%rcx,%rax)
⑲  addq $8, %rax
⑳  cmpq %r9, %rax
㉑  jne  .L9
```

Only 1 of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1) | | | | | | | | | | |
| 2 | (2) | (1) | | | | | | | | | |
| 3 | (3) | (2) | (1) | | | | | | | | |
| 4 | (4) | (3) | (2) | (1) | | | | | | | |
| 5 | (5) | (4) | (3) | (2) | (1) | | | | | | |
| 6 | (6) | (5) | (3)(4) | | (2) | (1) | | | | | |
| 7 | (7) | (6) | (3)(4)(5) | | | (2) | (1) | | | | |
| 8 | (8) | (7) | (3)(5)(6) | (4) | | | (2) | | | | ~~(1)~~ |
| 9 | (9) | (8) | (5)(6)(7) | (3) | (4) | | | | | | ~~(2)~~ |
| 10 | (10) | (9) | (6)(7)(8) | | (3) | (4) | | (5) | | | |
| 11 | (11) | (10) | (7)(8)(9) | | | (3) | (4) | (6) | | | (5) |
| 12 | (12) | (11) | (9)(10) | | | | (3) | | | (7) | (4)(5)(6) |
| 13 | (13) | (12) | (10)(11) | (8) | | | | | | | ~~(3)(4)(5)(6)(7)~~ |
| 14 | (14) | (13) | (10)(11)(12) | (9) | (8) | | | | | | |
| 15 | (15) | (14) | (10)(11)(13) | | (9) | (8) | | (12) | | | |
| 16 | (16) | (15) | (10)(11)(14) | | | (9) | (8) | (13) | | | (12) |
| 17 | (17) | (16) | (10)(14)(15) | (11) | | | (9) | | | | ~~(8)~~(12)(13) |
| 18 | (18) | (17) | (14)(15)(16) | (10) | (11) | | | | | | ~~(9)~~(12)(13) |
| 19 | (19) | (18) | (15)(16)(17) | | (10) | (11) | | | | (14) | (12)(13) |
| 20 | (20) | (19) | (16)(17)(18) | (15) | | (10) | (11) | | | | (12)(13)(14) |

# Register renaming

```
①  movq (%rdi,%rax), %rsi  → P1
②  movq (%rcx,%rax), %r8   → P2
③  movq %r8, (%rdi,%rax)
④  movq %rsi, (%rcx,%rax)
⑤  addq $8, %rax           → P3
⑥  cmpq %r9, %rax
⑦  jne  .L9
⑧  movq (%rdi,%rax), %rsi  → P4
⑨  movq (%rcx,%rax), %r8   → P5
⑩  movq %r8, (%rdi,%rax)
⑪  movq %rsi, (%rcx,%rax)
⑫  addq $8, %rax           → P6
⑬  cmpq %r9, %rax
⑭  jne  .L9
⑮  movq (%rdi,%rax), %rsi
⑯  movq (%rcx,%rax), %r8
⑰  movq %r8, (%rdi,%rax)
⑱  movq %rsi, (%rcx,%rax)
⑲  addq $8, %rax
⑳  cmpq %r9, %rax
㉑  jne  .L9
```

Only 1 of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1) | | | | | | | | | | |
| 2 | (2) | (1) | | | | | | | | | |
| 3 | (3) | (2) | (1) | | | | | | | | |
| 4 | (4) | (3) | (2) | (1) | | | | | | | |
| 5 | (5) | (4) | (3) | (2) | (1) | | | | | | |
| 6 | (6) | (5) | (3)(4) | | (2) | (1) | | | | | |
| 7 | (7) | (6) | (3)(4)(5) | | | (2) | (1) | | | | |
| 8 | (8) | (7) | (3)(5)(6) | (4) | | | (2) | | | | ~~(1)~~ |
| 9 | (9) | (8) | (5)(6)(7) | (3) | (4) | | | | | | ~~(2)~~ |
| 10 | (10) | (9) | (6)(7)(8) | | (3) | (4) | | (5) | | | |
| 11 | (11) | (10) | (7)(8)(9) | | | (3) | (4) | (6) | | | (5) |
| 12 | (12) | (11) | (9)(10) | | | | (3) | | | (7) | (4)(5)(6) |
| 13 | (13) | (12) | (10)(11) | (8) | | | | | | | ~~(3)(4)(5)(6)(7)~~ |
| 14 | (14) | (13) | (10)(11)(12) | (9) | (8) | | | | | | |
| 15 | (15) | (14) | (10)(11)(13) | | (9) | (8) | | (12) | | | |
| 16 | (16) | (15) | (10)(11)(14) | | | (9) | (8) | (13) | | | (12) |
| 17 | (17) | (16) | (10)(14)(15) | (11) | | | (9) | | | | ~~(8)~~(12)(13) |
| 18 | (18) | (17) | (14)(15)(16) | (10) | (11) | | | | | | ~~(9)~~(12)(13) |
| 19 | (19) | (18) | (15)(16)(17) | | (10) | (11) | | | | (14) | (12)(13) |
| 20 | (20) | (19) | (16)(17)(18) | (15) | | (10) | (11) | | | | (12)(13)(14) |
| 21 | (21) | (20) | (17)(18)(19) | (16) | (15) | | (10) | | | | (11)(12)(13)(14) |

# Register renaming

① `movq (%rdi,%rax), %rsi` → P1
② `movq (%rcx,%rax), %r8` → P2
③ `movq %r8, (%rdi,%rax)`
④ `movq %rsi, (%rcx,%rax)`
⑤ `addq $8, %rax` → P3
⑥ `cmpq %r9, %rax`
⑦ `jne .L9`
⑧ `movq (%rdi,%rax), %rsi` → P4
⑨ `movq (%rcx,%rax), %r8` → P5
⑩ `movq %r8, (%rdi,%rax)`
⑪ `movq %rsi, (%rcx,%rax)`
⑫ `addq $8, %rax` → P6
⑬ `cmpq %r9, %rax`
⑭ `jne .L9`
⑮ `movq (%rdi,%rax), %rsi`
⑯ `movq (%rcx,%rax), %r8`
⑰ `movq %r8, (%rdi,%rax)`
⑱ `movq %rsi, (%rcx,%rax)`
⑲ `addq $8, %rax`
⑳ `cmpq %r9, %rax`
㉑ `jne .L9`

Only 1 of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1) | | | | | | | | | | |
| 2 | (2) | (1) | | | | | | | | | |
| 3 | (3) | (2) | (1) | | | | | | | | |
| 4 | (4) | (3) | (2) | (1) | | | | | | | |
| 5 | (5) | (4) | (3) | (2) | (1) | | | | | | |
| 6 | (6) | (5) | (3)(4) | | (2) | (1) | | | | | |
| 7 | (7) | (6) | (3)(4)(5) | | | (2) | (1) | | | | |
| 8 | (8) | (7) | (3)(5)(6) | (4) | | | (2) | | | | ~~(1)~~ |
| 9 | (9) | (8) | (5)(6)(7) | (3) | (4) | | | | | | ~~(2)~~ |
| 10 | (10) | (9) | (6)(7)(8) | | (3) | (4) | | (5) | | | |
| 11 | (11) | (10) | (7)(8)(9) | | | (3) | (4) | (6) | | | (5) |
| 12 | (12) | (11) | (9)(10) | | | | (3) | | | (7) | (4)(5)(6) |
| 13 | (13) | (12) | (10)(11) | (8) | | | | | | | ~~(3)(4)(5)(6)(7)~~ |
| 14 | (14) | (13) | (10)(11)(12) | (9) | (8) | | | | | | |
| 15 | (15) | (14) | (10)(11)(13) | | (9) | (8) | | (12) | | | |
| 16 | (16) | (15) | (10)(11)(14) | | | (9) | (8) | (13) | | | (12) |
| 17 | (17) | (16) | (10)(14)(15) | (11) | | | (9) | | | | ~~(8)~~(12)(13) |
| 18 | (18) | (17) | (14)(15)(16) | (10) | (11) | | | | | | ~~(9)~~(12)(13) |
| 19 | (19) | (18) | (15)(16)(17) | | (10) | (11) | | | | (14) | (12)(13) |
| 20 | (20) | (19) | (16)(17)(18) | (15) | | (10) | (11) | | | | (12)(13)(14) |
| 21 | (21) | (20) | (17)(18)(19) | (16) | (15) | | (10) | | | | (11)(12)(13)(14) |
| 22 | | (21) | (17)(18)(20) | | (16) | (15) | | (19) | | | (10)(11)(12)(13)(14) |

# Register renaming

```
① movq (%rdi,%rax), %rsi → P1
② movq (%rcx,%rax), %r8  → P2
③ movq %r8, (%rdi,%rax)
④ movq %rsi, (%rcx,%rax)
⑤ addq $8, %rax          → P3
⑥ cmpq %r9, %rax
⑦ jne  .L9
⑧ movq (%rdi,%rax), %rsi → P4
⑨ movq (%rcx,%rax), %r8  → P5
⑩ movq %r8, (%rdi,%rax)
⑪ movq %rsi, (%rcx,%rax)
⑫ addq $8, %rax          → P6
⑬ cmpq %r9, %rax
⑭ jne  .L9
⑮ movq (%rdi,%rax), %rsi
⑯ movq (%rcx,%rax), %r8
⑰ movq %r8, (%rdi,%rax)
⑱ movq %rsi, (%rcx,%rax)
⑲ addq $8, %rax
⑳ cmpq %r9, %rax
㉑ jne  .L9
```

Only 1 of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1) | | | | | | | | | | |
| 2 | (2) | (1) | | | | | | | | | |
| 3 | (3) | (2) | (1) | | | | | | | | |
| 4 | (4) | (3) | (2) | (1) | | | | | | | |
| 5 | (5) | (4) | (3) | (2) | (1) | | | | | | |
| 6 | (6) | (5) | (3)(4) | | (2) | (1) | | | | | |
| 7 | (7) | (6) | (3)(4)(5) | | | (2) | (1) | | | | |
| 8 | (8) | (7) | (3)(5)(6) | (4) | | | (2) | | | | ~~(1)~~ |
| 9 | (9) | (8) | (5)(6)(7) | (3) | (4) | | | | | | ~~(2)~~ |
| 10 | (10) | (9) | (6)(7)(8) | | (3) | (4) | | (5) | | | |
| 11 | (11) | (10) | (7)(8)(9) | | | (3) | (4) | (6) | | | (5) |
| 12 | (12) | (11) | (9)(10) | | | | (3) | | | (7) | (4)(5)(6) |
| 13 | (13) | (12) | (10)(11) | (8) | | | | | | | ~~(3)(4)(5)(6)(7)~~ |
| 14 | (14) | (13) | (10)(11)(12) | (9) | (8) | | | | | | |
| 15 | (15) | (14) | (10)(11)(13) | | (9) | (8) | | (12) | | | |
| 16 | (16) | (15) | (10)(11)(14) | | | (9) | (8) | (13) | | | (12) |
| 17 | (17) | (16) | (10)(14)(15) | (11) | | | (9) | | | | ~~(8)~~(12)(13) |
| 18 | (18) | (17) | (14)(15)(16) | (10) | (11) | | | | | | ~~(9)~~(12)(13) |
| 19 | (19) | (18) | (15)(16)(17) | | (10) | (11) | | | | (14) | (12)(13) |
| 20 | (20) | (19) | (16)(17)(18) | (15) | | (10) | (11) | | | | (12)(13)(14) |
| 21 | (21) | (20) | (17)(18)(19) | (16) | (15) | | (10) | | | | (11)(12)(13)(14) |
| 22 | | (21) | (17)(18)(20) | | (16) | (15) | | (19) | | | ~~(10)(11)(12)(13)(14)~~ |
| 23 | | | (17)(18)(21) | | | (16) | (15) | (20) | | | (19) |

# Register renaming

```
1  movq (%rdi,%rax), %rsi → P1
2  movq (%rcx,%rax), %r8  → P2
3  movq %r8, (%rdi,%rax)
4  movq %rsi, (%rcx,%rax)
5  addq $8, %rax          → P3
6  cmpq %r9, %rax
7  jne  .L9
8  movq (%rdi,%rax), %rsi → P4
9  movq (%rcx,%rax), %r8  → P5
10 movq %r8, (%rdi,%rax)
11 movq %rsi, (%rcx,%rax)
12 addq $8, %rax          → P6
13 cmpq %r9, %rax
14 jne  .L9
15 movq (%rdi,%rax), %rsi
16 movq (%rcx,%rax), %r8
17 movq %r8, (%rdi,%rax)
18 movq %rsi, (%rcx,%rax)
19 addq $8, %rax
20 cmpq %r9, %rax
21 jne  .L9
```

Only 1 of them can have a instruction at the same cycle

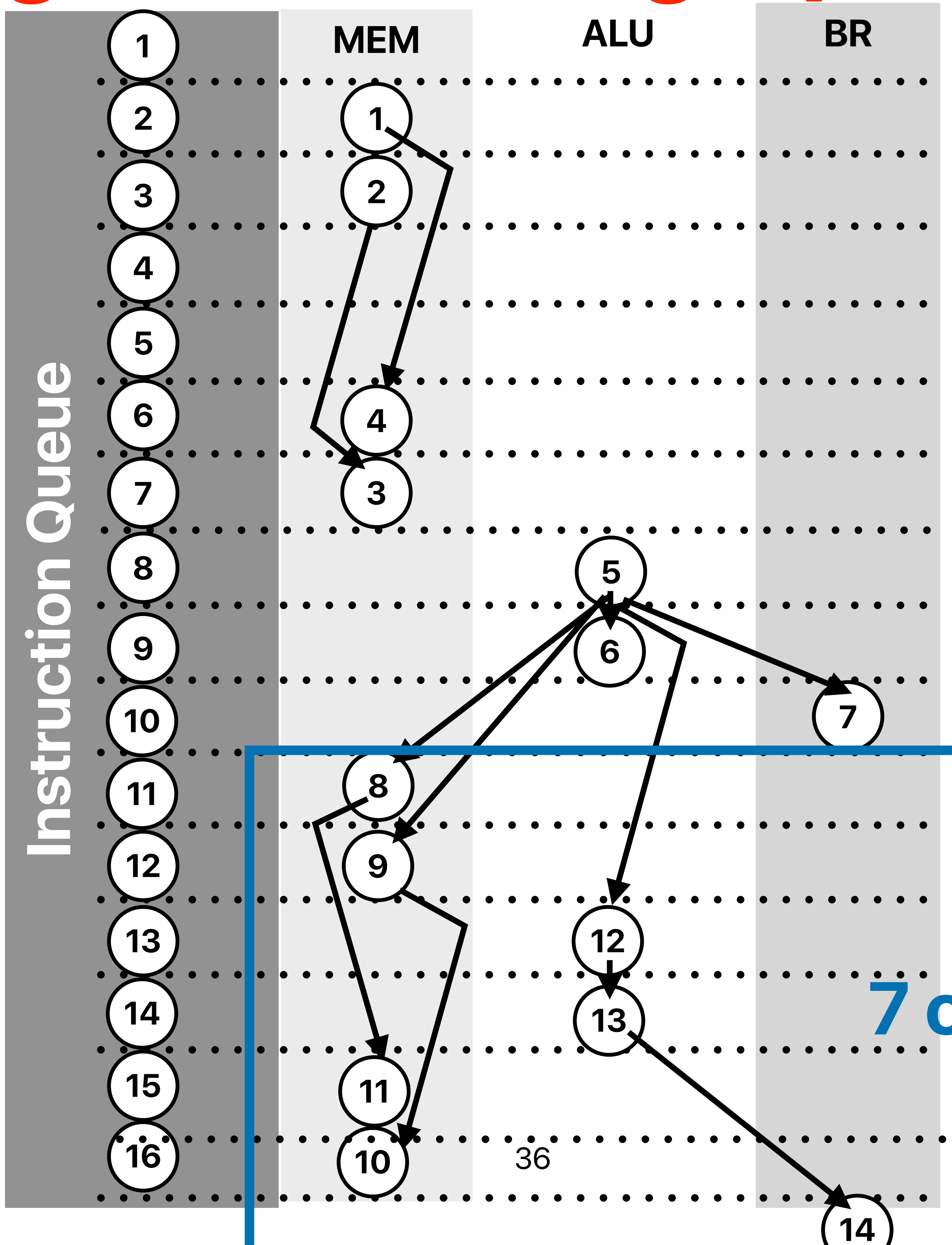
| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1) | | | | | | | | | | |
| 2 | (2) | (1) | | | | | | | | | |
| 3 | (3) | (2) | (1) | | | | | | | | |
| 4 | (4) | (3) | (2) | (1) | | | | | | | |
| 5 | (5) | (4) | (3) | (2) | (1) | | | | | | |
| 6 | (6) | (5) | (3)(4) | | (2) | (1) | | | | | |
| 7 | (7) | (6) | (3)(4)(5) | | | (2) | (1) | | | | |
| 8 | (8) | (7) | (3)(5)(6) | (4) | | | (2) | | | | ~~(1)~~ |
| 9 | (9) | (8) | (5)(6)(7) | (3) | (4) | | | | | | ~~(2)~~ |
| 10 | (10) | (9) | (6)(7)(8) | | (3) | (4) | | (5) | | | |
| 11 | (11) | (10) | (7)(8)(9) | | | (3) | (4) | (6) | | | (5) |
| 12 | (12) | (11) | (9)(10) | | | | (3) | | | (7) | (4)(5)(6) |
| 13 | (13) | (12) | (10)(11) | (8) | | | | | | | ~~(3)(4)(5)(6)(7)~~ |
| 14 | (14) | (13) | (10)(11)(12) | (9) | (8) | | | | | | |
| 15 | (15) | (14) | (10)(11)(13) | | (9) | (8) | | (12) | | | |
| 16 | (16) | (15) | (10)(11)(14) | | | (9) | (8) | (13) | | | (12) |
| 17 | (17) | (16) | (10)(14)(15) | (11) | | | (9) | | | | ~~(8)~~(12)(13) |
| 18 | (18) | (17) | (14)(15)(16) | (10) | (11) | | | | | | ~~(9)~~(12)(13) |
| 19 | (19) | (18) | (15)(16)(17) | | (10) | (11) | | | | (14) | (12)(13) |
| 20 | (20) | (19) | (16)(17)(18) | (15) | | (10) | (11) | | | | (12)(13)(14) |
| 21 | (21) | (20) | (17)(18)(19) | (16) | (15) | | (10) | | | | (11)(12)(13)(14) |
| 22 | | (21) | (17)(18)(20) | | (16) | (15) | | (19) | | | ~~(10)(11)(12)(13)(14)~~ |
| 23 | | | (17)(18)(21) | | | (16) | (15) | (20) | | | (19) |
| 24 | | | (17)(21) | (18) | | | (16) | | | | ~~(15)~~(19)(20) |

7 cycles for 7 instructions

CPI = 1

# Through data flow graph analysis

```
①  movq (%rdi,%rax), %rsi
②  movq (%rcx,%rax), %r8
③  movq %r8, (%rdi,%rax)
④  movq %rsi, (%rcx,%rax)
⑤  addq $8, %rax
⑥  cmpq %r9, %rax
⑦  jne  .L9
⑧  movq (%rdi,%rax), %rsi
⑨  movq (%rcx,%rax), %r8
⑩  movq %r8, (%rdi,%rax)
⑪  movq %rsi, (%rcx,%rax)
⑫  addq $8, %rax
⑬  cmpq %r9, %rax
⑭  jne  .L9
⑮  movq (%rdi,%rax), %rsi
⑯  movq (%rcx,%rax), %r8
⑰  movq %r8, (%rdi,%rax)
⑱  movq %rsi, (%rcx,%rax)
⑲  addq $8, %rax
⑳  cmpq %r9, %rax
㉑  jne  .L9
```




**7 cycles every iteration**

**CPI** $= \frac{7}{7} = 1!$

# If CPI==1 the limitation?

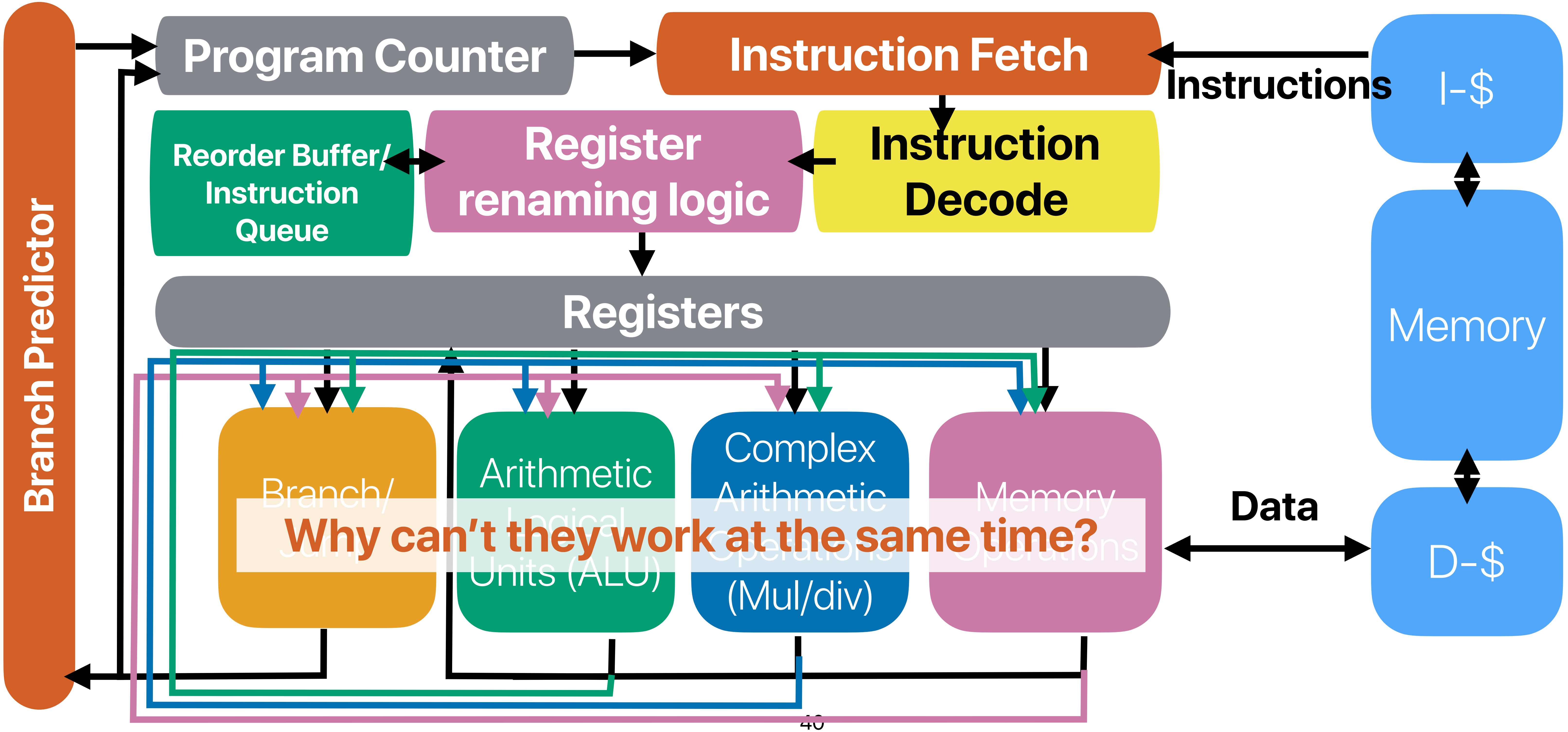
# Super Scalar

# Superscalar

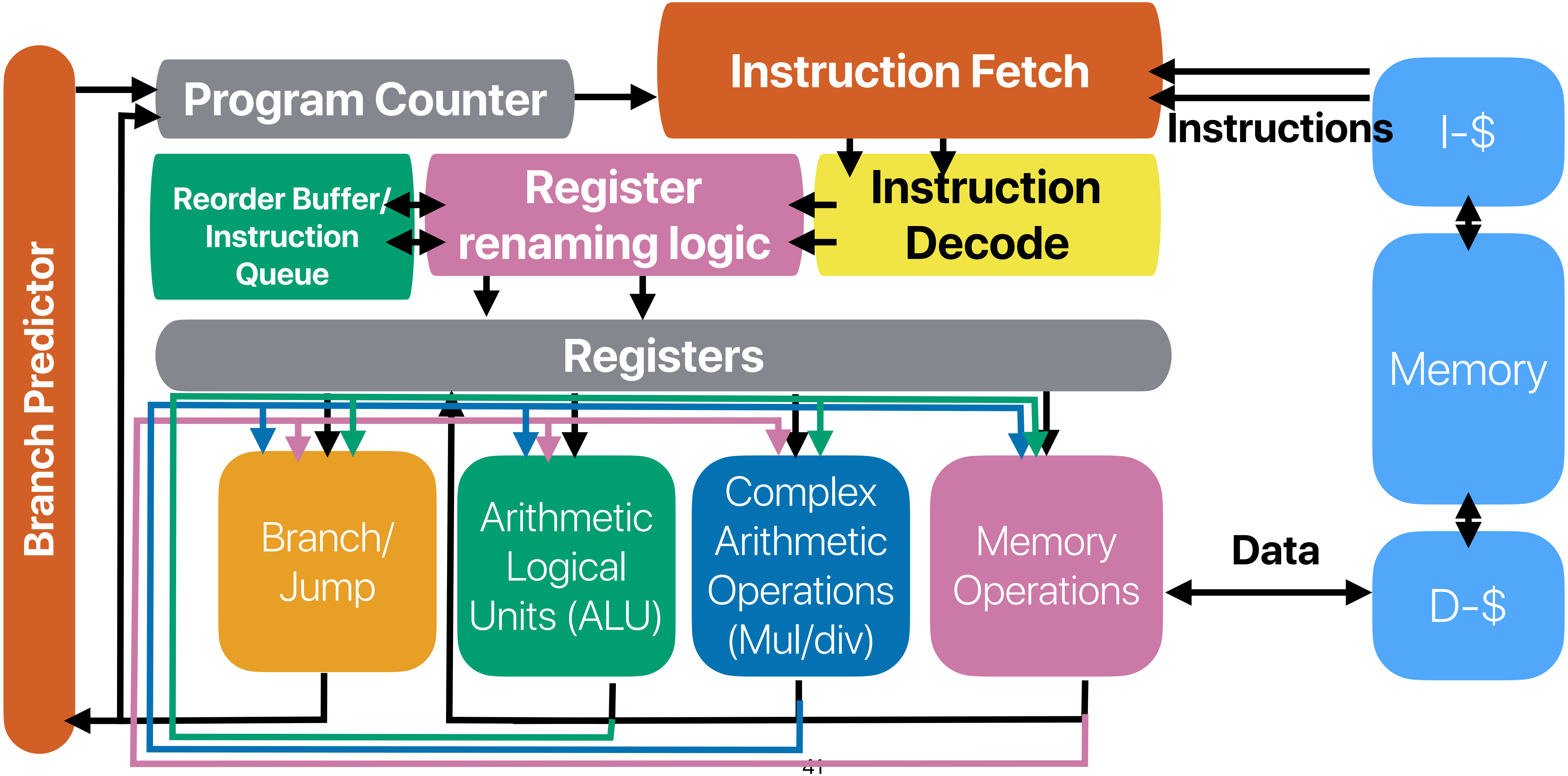
- Since we have many functional units now, we should fetch/decode more instructions each cycle so that we can have more instructions to issue!
- Super-scalar: fetch/decode/issue more than one instruction each cycle
  - **Fetch width:** how many instructions can the processor fetch/decode each cycle
  - **Issue width**: how many instructions can the processor issue each cycle
- The theoretical CPI should now be

$$\frac{1}{min(issue\ width, fetch\ width, decode\ width)}$$

# Register renaming + OoO + RoB



# Register renaming + SuperScalar



# 2-issue SS + Register renaming +OoO

```
① movq (%rdi,%rax), %rsi  → P1
② movq (%rcx,%rax), %r8   → P2
③ movq %r8, (%rdi,%rax)
④ movq %rsi, (%rcx,%rax)
⑤ addq $8, %rax           → P3
⑥ cmpq %r9, %rax
⑦ jne  .L9
⑧ movq (%rdi,%rax), %rsi  → P4
⑨ movq (%rcx,%rax), %r8   → P5
⑩ movq %r8, (%rdi,%rax)
⑪ movq %rsi, (%rcx,%rax)
⑫ addq $8, %rax           → P6
⑬ cmpq %r9, %rax
⑭ jne  .L9
⑮ movq (%rdi,%rax), %rsi
⑯ movq (%rcx,%rax), %r8
⑰ movq %r8, (%rdi,%rax)
⑱ movq %rsi, (%rcx,%rax)
⑲ addq $8, %rax
⑳ cmpq %r9, %rax
㉑ jne  .L9
```

Only "2" of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1)(2) | | | | | | | | | | |
| 2 | (3)(4) | (1)(2) | | | | | | | | | |
| 3 | (5)(6) | (3)(4) | (1)(2) | | | | | | | | |
| 4 | | | | | | | | | | | |
| 5 | | | | | | | | | | | |
| 6 | | | | | | | | | | | |
| 7 | | | | | | | | | | | |
| 8 | | | | | | | | | | | |
| 9 | | | | | | | | | | | |
| 10 | | | | | | | | | | | |
| 11 | | | | | | | | | | | |
| 12 | | | | | | | | | | | |
| 13 | | | | | | | | | | | |
| 14 | | | | | | | | | | | |
| 15 | | | | | | | | | | | |
| 16 | | | | | | | | | | | |

# 2-issue SS + Register renaming +OoO

Only "2" of them can have a instruction at the same cycle

```
① movq (%rdi,%rax), %rsi → P1
② movq (%rcx,%rax), %r8  → P2
③ movq %r8, (%rdi,%rax)
④ movq %rsi, (%rcx,%rax)
⑤ addq $8, %rax          → P3
⑥ cmpq %r9, %rax
⑦ jne  .L9
⑧ movq (%rdi,%rax), %rsi → P4
⑨ movq (%rcx,%rax), %r8  → P5
⑩ movq %r8, (%rdi,%rax)
⑪ movq %rsi, (%rcx,%rax)
⑫ addq $8, %rax          → P6
⑬ cmpq %r9, %rax
⑭ jne  .L9
⑮ movq (%rdi,%rax), %rsi
⑯ movq (%rcx,%rax), %r8
⑰ movq %r8, (%rdi,%rax)
⑱ movq %rsi, (%rcx,%rax)
⑲ addq $8, %rax
⑳ cmpq %r9, %rax
㉑ jne  .L9
```

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1)(2) | | | | | | | | | | |
| 2 | (3)(4) | (1)(2) | | | | | | | | | |
| 3 | (5)(6) | (3)(4) | (1)(2) | | | | | | | | |
| 4 | (7)(8) | (5)(6) | (2)(3)(4) | (1) | | | | | | | |
| 5 | | | | | | | | | | | |
| 6 | | | | | | | | | | | |
| 7 | | | | | | | | | | | |
| 8 | | | | | | | | | | | |
| 9 | | | | | | | | | | | |
| 10 | | | | | | | | | | | |
| 11 | | | | | | | | | | | |
| 12 | | | | | | | | | | | |
| 13 | | | | | | | | | | | |
| 14 | | | | | | | | | | | |
| 15 | | | | | | | | | | | |
| 16 | | | | | | | | | | | |

# 2-issue SS + Register renaming +OoO

```
① movq (%rdi,%rax), %rsi → P1
② movq (%rcx,%rax), %r8  → P2
③ movq %r8, (%rdi,%rax)
④ movq %rsi, (%rcx,%rax)
⑤ addq $8, %rax          → P3
⑥ cmpq %r9, %rax
⑦ jne  .L9
⑧ movq (%rdi,%rax), %rsi → P4
⑨ movq (%rcx,%rax), %r8  → P5
⑩ movq %r8, (%rdi,%rax)
⑪ movq %rsi, (%rcx,%rax)
⑫ addq $8, %rax          → P6
⑬ cmpq %r9, %rax
⑭ jne  .L9
⑮ movq (%rdi,%rax), %rsi
⑯ movq (%rcx,%rax), %r8
⑰ movq %r8, (%rdi,%rax)
⑱ movq %rsi, (%rcx,%rax)
⑲ addq $8, %rax
⑳ cmpq %r9, %rax
㉑ jne  .L9
```

Only "2" of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1)(2) | | | | | | | | | | |
| 2 | (3)(4) | (1)(2) | | | | | | | | | |
| 3 | (5)(6) | (3)(4) | (1)(2) | | | | | | | | |
| 4 | (7)(8) | (5)(6) | (2)(3)(4) | (1) | | | | | | | |
| 5 | (9)(10) | (7)(8) | (3)(4)(5)(6) | (2) | (1) | | | | | | |

# 2-issue SS + Register renaming +OoO

Only "2" of them can have a instruction at the same cycle

```
①  movq (%rdi,%rax), %rsi   → P1
②  movq (%rcx,%rax), %r8    → P2
③  movq %r8, (%rdi,%rax)
④  movq %rsi, (%rcx,%rax)
⑤  addq $8, %rax            → P3
⑥  cmpq %r9, %rax
⑦  jne  .L9
⑧  movq (%rdi,%rax), %rsi   → P4
⑨  movq (%rcx,%rax), %r8    → P5
⑩  movq %r8, (%rdi,%rax)
⑪  movq %rsi, (%rcx,%rax)
⑫  addq $8, %rax            → P6
⑬  cmpq %r9, %rax
⑭  jne  .L9
⑮  movq (%rdi,%rax), %rsi
⑯  movq (%rcx,%rax), %r8
⑰  movq %r8, (%rdi,%rax)
⑱  movq %rsi, (%rcx,%rax)
⑲  addq $8, %rax
⑳  cmpq %r9, %rax
㉑  jne  .L9
```

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1)(2) | | | | | | | | | | |
| 2 | (3)(4) | (1)(2) | | | | | | | | | |
| 3 | (5)(6) | (3)(4) | (1)(2) | | | | | | | | |
| 4 | (7)(8) | (5)(6) | (2)(3)(4) | (1) | | | | | | | |
| 5 | (9)(10) | (7)(8) | (3)(4)(5)(6) | (2) | (1) | | | | | | |
| 6 | (11)(12) | (9)(10) | (3)(4)(6)(7)(8) | | (2) | (1) | | (5) | | | |

# 2-issue SS + Register renaming +OoO

```
① movq (%rdi,%rax), %rsi → P1
② movq (%rcx,%rax), %r8 → P2
③ movq %r8, (%rdi,%rax)
④ movq %rsi, (%rcx,%rax)
⑤ addq $8, %rax → P3
⑥ cmpq %r9, %rax
⑦ jne .L9
⑧ movq (%rdi,%rax), %rsi → P4
⑨ movq (%rcx,%rax), %r8 → P5
⑩ movq %r8, (%rdi,%rax)
⑪ movq %rsi, (%rcx,%rax)
⑫ addq $8, %rax → P6
⑬ cmpq %r9, %rax
⑭ jne .L9
⑮ movq (%rdi,%rax), %rsi
⑯ movq (%rcx,%rax), %r8
⑰ movq %r8, (%rdi,%rax)
⑱ movq %rsi, (%rcx,%rax)
⑲ addq $8, %rax
⑳ cmpq %r9, %rax
㉑ jne .L9
```

Only "2" of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1)(2) | | | | | | | | | | |
| 2 | (3)(4) | (1)(2) | | | | | | | | | |
| 3 | (5)(6) | (3)(4) | (1)(2) | | | | | | | | |
| 4 | (7)(8) | (5)(6) | (2)(3)(4) | (1) | | | | | | | |
| 5 | (9)(10) | (7)(8) | (3)(4)(5)(6) | (2) | (1) | | | | | | |
| 6 | (11)(12) | (9)(10) | (3)(4)(6)(7)(8) | | (2) | (1) | | (5) | | | |
| 7 | (13)(14) | (11)(12) | (3)(4)(7)(9)(10) | (8) | | (2) | (1) | (6) | | | (5) |

# 2-issue SS + Register renaming +OoO

```
①  movq (%rdi,%rax), %rsi  → P1
②  movq (%rcx,%rax), %r8   → P2
③  movq %r8, (%rdi,%rax)
④  movq %rsi, (%rcx,%rax)
⑤  addq $8, %rax           → P3
⑥  cmpq %r9, %rax
⑦  jne  .L9
⑧  movq (%rdi,%rax), %rsi  → P4
⑨  movq (%rcx,%rax), %r8   → P5
⑩  movq %r8, (%rdi,%rax)
⑪  movq %rsi, (%rcx,%rax)
⑫  addq $8, %rax           → P6
⑬  cmpq %r9, %rax
⑭  jne  .L9
⑮  movq (%rdi,%rax), %rsi
⑯  movq (%rcx,%rax), %r8
⑰  movq %r8, (%rdi,%rax)
⑱  movq %rsi, (%rcx,%rax)
⑲  addq $8, %rax
⑳  cmpq %r9, %rax
㉑  jne  .L9
```

Only "2" of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1)(2) | | | | | | | | | | |
| 2 | (3)(4) | (1)(2) | | | | | | | | | |
| 3 | (5)(6) | (3)(4) | (1)(2) | | | | | | | | |
| 4 | (7)(8) | (5)(6) | (2)(3)(4) | (1) | | | | | | | |
| 5 | (9)(10) | (7)(8) | (3)(4)(5)(6) | (2) | (1) | | | | | | |
| 6 | (11)(12) | (9)(10) | (3)(4)(6)(7)(8) | | (2) | (1) | | (5) | | | |
| 7 | (13)(14) | (11)(12) | (3)(4)(7)(9)(10) | (8) | | (2) | (1) | (6) | | | (5) |
| 8 | (15)(16) | (13)(14) | (3)(9)(10)(11)(12) | (4) | (8) | | (2) | | | (7) | (1)(5)(6) |

# 2-issue SS + Register renaming +OoO

```
①  movq (%rdi,%rax), %rsi  → P1
②  movq (%rcx,%rax), %r8   → P2
③  movq %r8, (%rdi,%rax)
④  movq %rsi, (%rcx,%rax)
⑤  addq $8, %rax           → P3
⑥  cmpq %r9, %rax
⑦  jne  .L9
⑧  movq (%rdi,%rax), %rsi  → P4
⑨  movq (%rcx,%rax), %r8   → P5
⑩  movq %r8, (%rdi,%rax)
⑪  movq %rsi, (%rcx,%rax)
⑫  addq $8, %rax           → P6
⑬  cmpq %r9, %rax
⑭  jne  .L9
⑮  movq (%rdi,%rax), %rsi
⑯  movq (%rcx,%rax), %r8
⑰  movq %r8, (%rdi,%rax)
⑱  movq %rsi, (%rcx,%rax)
⑲  addq $8, %rax
⑳  cmpq %r9, %rax
㉑  jne  .L9
```

Only "2" of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1)(2) | | | | | | | | | | |
| 2 | (3)(4) | (1)(2) | | | | | | | | | |
| 3 | (5)(6) | (3)(4) | (1)(2) | | | | | | | | |
| 4 | (7)(8) | (5)(6) | (2)(3)(4) | (1) | | | | | | | |
| 5 | (9)(10) | (7)(8) | (3)(4)(5)(6) | (2) | (1) | | | | | | |
| 6 | (11)(12) | (9)(10) | (3)(4)(6)(7)(8) | | (2) | (1) | | (5) | | | |
| 7 | (13)(14) | (11)(12) | (3)(4)(7)(9)(10) | (8) | | (2) | (1) | (6) | | | (5) |
| 8 | (15)(16) | (13)(14) | (3)(9)(10)(11)(12) | (4) | (8) | | (2) | | | (7) | ~~(1)~~(5)(6) |
| 9 | (17)(18) | (15)(16) | (9)(10)(11)(12)(13)(14) | (3) | (4) | (8) | | (12) | | | (2)(5)(6)(7) |

# 2-issue SS + Register renaming +OoO

```
①  movq (%rdi,%rax), %rsi  → P1
②  movq (%rcx,%rax), %r8   → P2
③  movq %r8, (%rdi,%rax)
④  movq %rsi, (%rcx,%rax)
⑤  addq $8, %rax           → P3
⑥  cmpq %r9, %rax
⑦  jne  .L9
⑧  movq (%rdi,%rax), %rsi  → P4
⑨  movq (%rcx,%rax), %r8   → P5
⑩  movq %r8, (%rdi,%rax)
⑪  movq %rsi, (%rcx,%rax)
⑫  addq $8, %rax           → P6
⑬  cmpq %r9, %rax
⑭  jne  .L9
⑮  movq (%rdi,%rax), %rsi
⑯  movq (%rcx,%rax), %r8
⑰  movq %r8, (%rdi,%rax)
⑱  movq %rsi, (%rcx,%rax)
⑲  addq $8, %rax
⑳  cmpq %r9, %rax
㉑  jne  .L9
```

Only "2" of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1)(2) | | | | | | | | | | |
| 2 | (3)(4) | (1)(2) | | | | | | | | | |
| 3 | (5)(6) | (3)(4) | (1)(2) | | | | | | | | |
| 4 | (7)(8) | (5)(6) | (2)(3)(4) | (1) | | | | | | | |
| 5 | (9)(10) | (7)(8) | (3)(4)(5)(6) | (2) | (1) | | | | | | |
| 6 | (11)(12) | (9)(10) | (3)(4)(6)(7)(8) | | (2) | (1) | | (5) | | | |
| 7 | (13)(14) | (11)(12) | (3)(4)(7)(9)(10) | (8) | | (2) | (1) | (6) | | | (5) |
| 8 | (15)(16) | (13)(14) | (3)(9)(10)(11)(12) | (4) | (8) | | (2) | | | (7) | ~~(1)~~(5)(6) |
| 9 | (17)(18) | (15)(16) | (9)(10)(11)(12)(13)(14) | (3) | (4) | (8) | | (12) | | | ~~(2)~~(5)(6)(7) |
| 10 | (19)(20) | (17)(18) | (10)(11)(12)(13)(14)(15)(16) | (9) | (3) | (4) | (8) | (13) | | | (5)(6)(7)(12) |

# 2-issue SS + Register renaming +OoO

```
①  movq (%rdi,%rax), %rsi  → P1
②  movq (%rcx,%rax), %r8   → P2
③  movq %r8, (%rdi,%rax)
④  movq %rsi, (%rcx,%rax)
⑤  addq $8, %rax           → P3
⑥  cmpq %r9, %rax
⑦  jne  .L9
⑧  movq (%rdi,%rax), %rsi  → P4
⑨  movq (%rcx,%rax), %r8   → P5
⑩  movq %r8, (%rdi,%rax)
⑪  movq %rsi, (%rcx,%rax)
⑫  addq $8, %rax           → P6
⑬  cmpq %r9, %rax
⑭  jne  .L9
⑮  movq (%rdi,%rax), %rsi
⑯  movq (%rcx,%rax), %r8
⑰  movq %r8, (%rdi,%rax)
⑱  movq %rsi, (%rcx,%rax)
⑲  addq $8, %rax
⑳  cmpq %r9, %rax
㉑  jne  .L9
```

Only "2" of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1)(2) | | | | | | | | | | |
| 2 | (3)(4) | (1)(2) | | | | | | | | | |
| 3 | (5)(6) | (3)(4) | (1)(2) | | | | | | | | |
| 4 | (7)(8) | (5)(6) | (2)(3)(4) | (1) | | | | | | | |
| 5 | (9)(10) | (7)(8) | (3)(4)(5)(6) | (2) | (1) | | | | | | |
| 6 | (11)(12) | (9)(10) | (3)(4)(6)(7)(8) | | (2) | (1) | | (5) | | | |
| 7 | (13)(14) | (11)(12) | (3)(4)(7)(9)(10) | (8) | | (2) | (1) | (6) | | | (5) |
| 8 | (15)(16) | (13)(14) | (3)(9)(10)(11)(12) | (4) | (8) | | (2) | | | (7) | ~~(1)~~(5)(6) |
| 9 | (17)(18) | (15)(16) | (9)(10)(11)(12)(13)(14) | (3) | (4) | (8) | | (12) | | | ~~(2)~~(5)(6)(7) |
| 10 | (19)(20) | (17)(18) | (10)(11)(12)(13)(14)(15)(16) | (9) | (3) | (4) | (8) | (13) | | | (5)(6)(7)(12) |
| 11 | | (19)(20) | (10)(12)(13)(15)(16)(17)(18) | (11) | (9) | (3) | (4) | | | (14) | (5)(6)(7)(8)(12)(13) |

# 2-issue SS + Register renaming +OoO

```
①  movq (%rdi,%rax), %rsi  → P1
②  movq (%rcx,%rax), %r8   → P2
③  movq %r8, (%rdi,%rax)
④  movq %rsi, (%rcx,%rax)
⑤  addq $8, %rax           → P3
⑥  cmpq %r9, %rax
⑦  jne  .L9
⑧  movq (%rdi,%rax), %rsi  → P4
⑨  movq (%rcx,%rax), %r8   → P5
⑩  movq %r8, (%rdi,%rax)
⑪  movq %rsi, (%rcx,%rax)
⑫  addq $8, %rax           → P6
⑬  cmpq %r9, %rax
⑭  jne  .L9
⑮  movq (%rdi,%rax), %rsi
⑯  movq (%rcx,%rax), %r8
⑰  movq %r8, (%rdi,%rax)
⑱  movq %rsi, (%rcx,%rax)
⑲  addq $8, %rax
⑳  cmpq %r9, %rax
㉑  jne  .L9
```

Only "2" of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1)(2) | | | | | | | | | | |
| 2 | (3)(4) | (1)(2) | | | | | | | | | |
| 3 | (5)(6) | (3)(4) | (1)(2) | | | | | | | | |
| 4 | (7)(8) | (5)(6) | (2)(3)(4) | (1) | | | | | | | |
| 5 | (9)(10) | (7)(8) | (3)(4)(5)(6) | (2) | (1) | | | | | | |
| 6 | (11)(12) | (9)(10) | (3)(4)(6)(7)(8) | | (2) | (1) | | (5) | | | |
| 7 | (13)(14) | (11)(12) | (3)(4)(7)(9)(10) | (8) | | (2) | (1) | (6) | | | (5) |
| 8 | (15)(16) | (13)(14) | (3)(9)(10)(11)(12) | (4) | (8) | | (2) | | | (7) | ~~(1)~~(5)(6) |
| 9 | (17)(18) | (15)(16) | (9)(10)(11)(12)(13)(14) | (3) | (4) | (8) | | (12) | | | ~~(2)~~(5)(6)(7) |
| 10 | (19)(20) | (17)(18) | (10)(11)(12)(13)(14)(15)(16) | (9) | (3) | (4) | (8) | (13) | | | (5)(6)(7)(12) |
| 11 | | (19)(20) | (10)(12)(13)(15)(16)(17)(18) | (11) | (9) | (3) | (4) | | | (14) | (5)(6)(7)(8)(12)(13) |
| 12 | | | (12)(13)(15)(16)(17)(18)(19)(20) | (10) | (11) | (9) | (3) | | | | (4)(5)(6)(7)(8)(12)(13)(14) |

# 2-issue SS + Register renaming +OoO

```
①  movq (%rdi,%rax), %rsi   → P1
②  movq (%rcx,%rax), %r8    → P2
③  movq %r8, (%rdi,%rax)
④  movq %rsi, (%rcx,%rax)
⑤  addq $8, %rax            → P3
⑥  cmpq %r9, %rax
⑦  jne  .L9
⑧  movq (%rdi,%rax), %rsi   → P4
⑨  movq (%rcx,%rax), %r8    → P5
⑩  movq %r8, (%rdi,%rax)
⑪  movq %rsi, (%rcx,%rax)
⑫  addq $8, %rax            → P6
⑬  cmpq %r9, %rax
⑭  jne  .L9
⑮  movq (%rdi,%rax), %rsi
⑯  movq (%rcx,%rax), %r8
⑰  movq %r8, (%rdi,%rax)
⑱  movq %rsi, (%rcx,%rax)
⑲  addq $8, %rax
⑳  cmpq %r9, %rax
㉑  jne  .L9
```

Only "2" of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1)(2) | | | | | | | | | | |
| 2 | (3)(4) | (1)(2) | | | | | | | | | |
| 3 | (5)(6) | (3)(4) | (1)(2) | | | | | | | | |
| 4 | (7)(8) | (5)(6) | (2)(3)(4) | (1) | | | | | | | |
| 5 | (9)(10) | (7)(8) | (3)(4)(5)(6) | (2) | (1) | | | | | | |
| 6 | (11)(12) | (9)(10) | (3)(4)(6)(7)(8) | | (2) | (1) | | (5) | | | |
| 7 | (13)(14) | (11)(12) | (3)(4)(7)(9)(10) | (8) | | (2) | (1) | (6) | | | (5) |
| 8 | (15)(16) | (13)(14) | (3)(9)(10)(11)(12) | (4) | (8) | | (2) | | | (7) | ~~(1)~~(5)(6) |
| 9 | (17)(18) | (15)(16) | (9)(10)(11)(12)(13)(14) | (3) | (4) | (8) | | (12) | | | ~~(2)~~(5)(6)(7) |
| 10 | (19)(20) | (17)(18) | (10)(11)(12)(13)(14)(15)(16) | (9) | (3) | (4) | (8) | (13) | | | (5)(6)(7)(12) |
| 11 | | (19)(20) | (10)(12)(13)(15)(16)(17)(18) | (11) | (9) | (3) | (4) | | | (14) | (5)(6)(7)(8)(12)(13) |
| 12 | | | (12)(13)(15)(16)(17)(18)(19)(20) | (10) | (11) | (9) | (3) | | | | (4)(5)(6)(7)(8)(12)(13)(14) |
| 13 | | | (12)(13)(16)(17)(18)(20) | (15) | (10) | (11) | (9) | (19) | | | ~~(3)(4)(5)(6)(7)(8)~~(12)(13)(14) |

# 2-issue SS + Register renaming +OoO

```
① movq (%rdi,%rax), %rsi  → P1
② movq (%rcx,%rax), %r8   → P2
③ movq %r8, (%rdi,%rax)
④ movq %rsi, (%rcx,%rax)
⑤ addq $8, %rax           → P3
⑥ cmpq %r9, %rax
⑦ jne  .L9
⑧ movq (%rdi,%rax), %rsi  → P4
⑨ movq (%rcx,%rax), %r8   → P5
⑩ movq %r8, (%rdi,%rax)
⑪ movq %rsi, (%rcx,%rax)
⑫ addq $8, %rax           → P6
⑬ cmpq %r9, %rax
⑭ jne  .L9
⑮ movq (%rdi,%rax), %rsi
⑯ movq (%rcx,%rax), %r8
⑰ movq %r8, (%rdi,%rax)
⑱ movq %rsi, (%rcx,%rax)
⑲ addq $8, %rax
⑳ cmpq %r9, %rax
㉑ jne  .L9
```

Only "2" of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1)(2) | | | | | | | | | | |
| 2 | (3)(4) | (1)(2) | | | | | | | | | |
| 3 | (5)(6) | (3)(4) | (1)(2) | | | | | | | | |
| 4 | (7)(8) | (5)(6) | (2)(3)(4) | (1) | | | | | | | |
| 5 | (9)(10) | (7)(8) | (3)(4)(5)(6) | (2) | (1) | | | | | | |
| 6 | (11)(12) | (9)(10) | (3)(4)(6)(7)(8) | | (2) | (1) | | (5) | | | |
| 7 | (13)(14) | (11)(12) | (3)(4)(7)(9)(10) | (8) | | (2) | (1) | (6) | | | (5) |
| 8 | (15)(16) | (13)(14) | (3)(9)(10)(11)(12) | (4) | (8) | | (2) | | | (7) | ~~(1)~~(5)(6) |
| 9 | (17)(18) | (15)(16) | (9)(10)(11)(12)(13)(14) | (3) | (4) | (8) | | (12) | | | ~~(2)~~(5)(6)(7) |
| 10 | (19)(20) | (17)(18) | (10)(11)(12)(13)(14)(15)(16) | (9) | (3) | (4) | (8) | (13) | | | (5)(6)(7)(12) |
| 11 | | (19)(20) | (10)(12)(13)(15)(16)(17)(18) | (11) | (9) | (3) | (4) | | | (14) | (5)(6)(7)(8)(12)(13) |
| 12 | | | (12)(13)(15)(16)(17)(18)(19)(20) | (10) | (11) | (9) | (3) | | | | (4)(5)(6)(7)(8)(12)(13)(14) |
| 13 | | | (12)(13)(16)(17)(18)(20) | (15) | (10) | (11) | (9) | (19) | | | ~~(3)(4)(5)(6)(7)(8)~~(12)(13)(14) |
| 14 | | | (12)(13)(17)(18) | (16) | (15) | (10) | (11) | (20) | | | (9)(12)(13)(14)(19) |

# 2-issue SS + Register renaming +OoO

```
① movq (%rdi,%rax), %rsi   → P1
② movq (%rcx,%rax), %r8    → P2
③ movq %r8, (%rdi,%rax)
④ movq %rsi, (%rcx,%rax)
⑤ addq $8, %rax            → P3
⑥ cmpq %r9, %rax
⑦ jne  .L9
⑧ movq (%rdi,%rax), %rsi   → P4
⑨ movq (%rcx,%rax), %r8    → P5
⑩ movq %r8, (%rdi,%rax)
⑪ movq %rsi, (%rcx,%rax)
⑫ addq $8, %rax            → P6
⑬ cmpq %r9, %rax
⑭ jne  .L9
⑮ movq (%rdi,%rax), %rsi
⑯ movq (%rcx,%rax), %r8
⑰ movq %r8, (%rdi,%rax)
⑱ movq %rsi, (%rcx,%rax)
⑲ addq $8, %rax
⑳ cmpq %r9, %rax
㉑ jne  .L9
```

Only "2" of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1)(2) | | | | | | | | | | |
| 2 | (3)(4) | (1)(2) | | | | | | | | | |
| 3 | (5)(6) | (3)(4) | (1)(2) | | | | | | | | |
| 4 | (7)(8) | (5)(6) | (2)(3)(4) | (1) | | | | | | | |
| 5 | (9)(10) | (7)(8) | (3)(4)(5)(6) | (2) | (1) | | | | | | |
| 6 | (11)(12) | (9)(10) | (3)(4)(6)(7)(8) | | (2) | (1) | | (5) | | | |
| 7 | (13)(14) | (11)(12) | (3)(4)(7)(9)(10) | (8) | | (2) | (1) | (6) | | | (5) |
| 8 | (15)(16) | (13)(14) | (3)(9)(10)(11)(12) | (4) | (8) | | (2) | | | (7) | ~~(1)~~(5)(6) |
| 9 | (17)(18) | (15)(16) | (9)(10)(11)(12)(13)(14) | (3) | (4) | (8) | | (12) | | | ~~(2)~~(5)(6)(7) |
| 10 | (19)(20) | (17)(18) | (10)(11)(12)(13)(14)(15)(16) | (9) | (3) | (4) | (8) | (13) | | | (5)(6)(7)(12) |
| 11 | | (19)(20) | (10)(12)(13)(15)(16)(17)(18) | (11) | (9) | (3) | (4) | | | (14) | (5)(6)(7)(8)(12)(13) |
| 12 | | | (12)(13)(15)(16)(17)(18)(19)(20) | (10) | (11) | (9) | (3) | | | | (4)(5)(6)(7)(8)(12)(13)(14) |
| 13 | | | (12)(13)(16)(17)(18)(20) | (15) | (10) | (11) | (9) | (19) | | | ~~(3)(4)(5)(6)(7)(8)~~(12)(13)(14) |
| 14 | | | (12)(13)(17)(18) | (16) | (15) | (10) | (11) | (20) | | | ~~(9)~~(12)(13)(14)(19) |
| 15 | | | | | (16) | (15) | (10) | | | (21) | (11)(12)(13)(14)(19)(20) |

# 2-issue SS + Register renaming +OoO

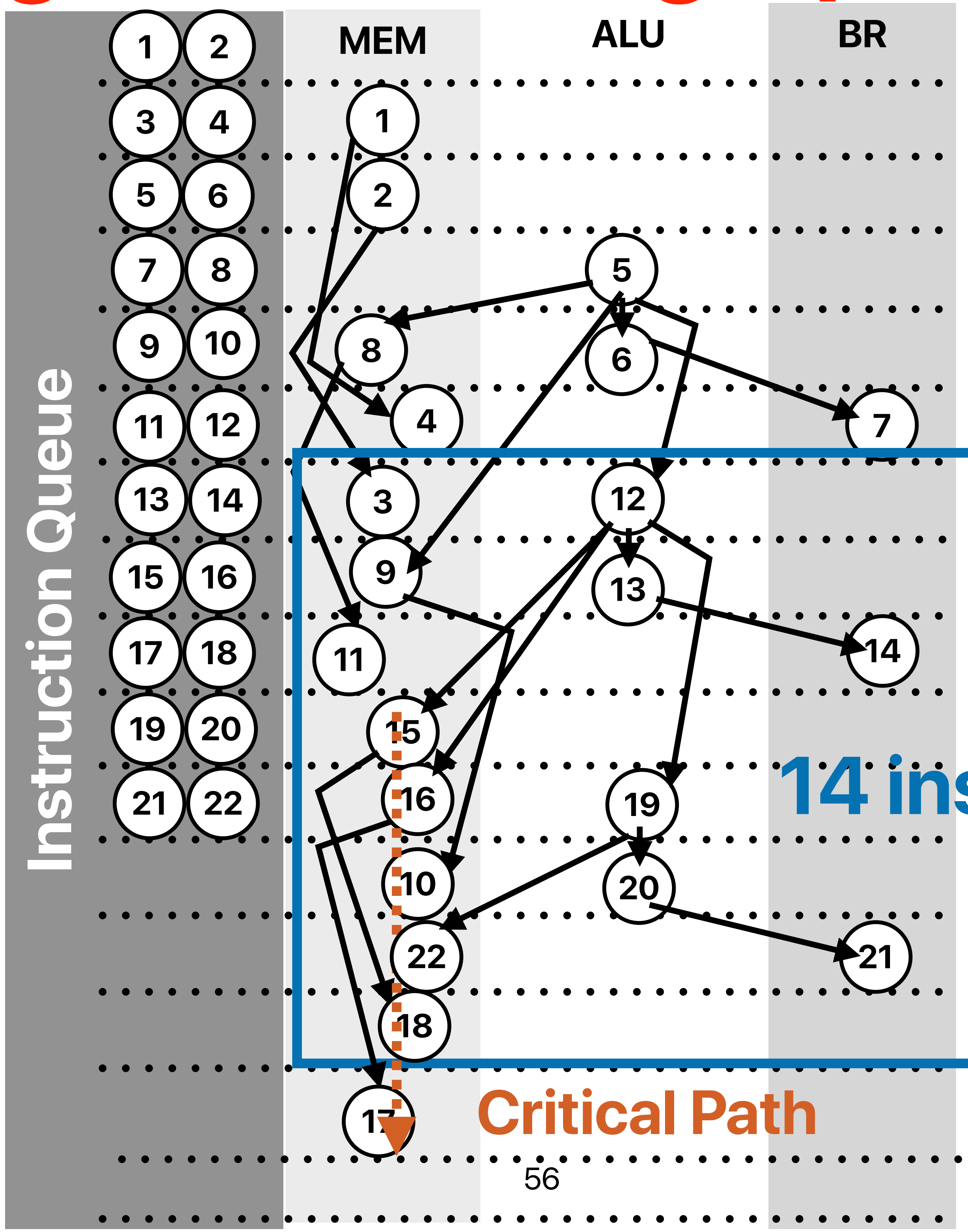
```
① movq (%rdi,%rax), %rsi  → P1
② movq (%rcx,%rax), %r8   → P2
③ movq %r8, (%rdi,%rax)
④ movq %rsi, (%rcx,%rax)  → P3
⑤ addq $8, %rax
⑥ cmpq %r9, %rax
⑦ jne  .L9
⑧ movq (%rdi,%rax), %rsi  → P4
⑨ movq (%rcx,%rax), %r8   → P5
⑩ movq %r8, (%rdi,%rax)
⑪ movq %rsi, (%rcx,%rax)
⑫ addq $8, %rax           → P6
⑬ cmpq %r9, %rax
⑭ jne  .L9
⑮ movq (%rdi,%rax), %rsi
⑯ movq (%rcx,%rax), %r8
⑰ movq %r8, (%rdi,%rax)
⑱ movq %rsi, (%rcx,%rax)
⑲ addq $8, %rax
⑳ cmpq %r9, %rax
㉑ jne  .L9
```

Only "2" of them can have a instruction at the same cycle

| | IF | ID | REN | M1 | M2 | M3 | M4 | ALU | MUL | BR | ROB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | (1)(2) | | | | | | | | | | |
| 2 | (3)(4) | (1)(2) | | | | | | | | | |
| 3 | (5)(6) | (3)(4) | (1)(2) | | | | | | | | |
| 4 | (7)(8) | (5)(6) | (2)(3)(4) | (1) | | | | | | | |
| 5 | (9)(10) | (7)(8) | (3)(4)(5)(6) | (2) | (1) | | | | | | |
| 6 | (11)(12) | (9)(10) | (3)(4)(6)(7)(8) | | (2) | (1) | | (5) | | | |
| 7 | (13)(14) | (11)(12) | (3)(4)(7)(9)(10) | (8) | | (2) | (1) | (6) | | | (5) |
| 8 | (15)(16) | (13)(14) | (3)(9)(10)(11)(12) | (4) | (8) | | (2) | | | (7) | ~~(1)~~(5)(6) |
| 9 | (17)(18) | (15)(16) | (9)(10)(11)(12)(13)(14) | (3) | (4) | (8) | | (12) | | | ~~(2)~~(5)(6)(7) |
| 10 | (19)(20) | (17)(18) | (10)(11)(12)(13)(14)(15)(16) | (9) | (3) | (4) | (8) | (13) | | | (5)(6)(7)(12) |
| 11 | | (19)(20) | (10)(12)(13)(15)(16)(17)(18) | (11) | (9) | (3) | (4) | | | (14) | (5)(6)(7)(8)(12)(13) |
| 12 | | | (12)(13)(15)(16)(17)(18)(19)(20) | (10) | (11) | (9) | (3) | | | | (4)(5)(6)(7)(8)(12)(13)(14) |
| 13 | | | (12)(13)(16)(17)(18)(20) | (15) | (10) | (11) | (9) | (19) | | | ~~(3)(4)(5)(6)(7)(8)~~(12)(13)(14) |
| 14 | | | (12)(13)(17)(18) | (16) | (15) | (10) | (11) | (20) | | | ~~(9)~~(12)(13)(14)(19) |
| 15 | | | | | (16) | (15) | (10) | | | (21) | (11)(12)(13)(14)(19)(20) |
| 16 | | | | | | (16) | (15) | | | | ~~(10)(11)(12)(13)(14)~~(19)(20)(21) |

# Through data flow graph analysis

```
①   movq (%rdi,%rax), %rsi
②   movq (%rcx,%rax), %r8
③   movq %r8, (%rdi,%rax)
④   movq %rsi, (%rcx,%rax)
⑤   addq $8, %rax
⑥   cmpq %r9, %rax
⑦   jne  .L9
⑧   movq (%rdi,%rax), %rsi
⑨   movq (%rcx,%rax), %r8
⑩   movq %r8, (%rdi,%rax)
⑪   movq %rsi, (%rcx,%rax)
⑫   addq $8, %rax
⑬   cmpq %r9, %rax
⑭   jne  .L9
⑮   movq (%rdi,%rax), %rsi
⑯   movq (%rcx,%rax), %r8
⑰   movq %r8, (%rdi,%rax)
⑱   movq %rsi, (%rcx,%rax)
⑲   addq $8, %rax
⑳   cmpq %r9, %rax
㉑   jne  .L9
```




**14 instructions in 8 cycles**

$$\text{CPI} = \frac{8}{14} = 0.57!$$

# What about "linked list"

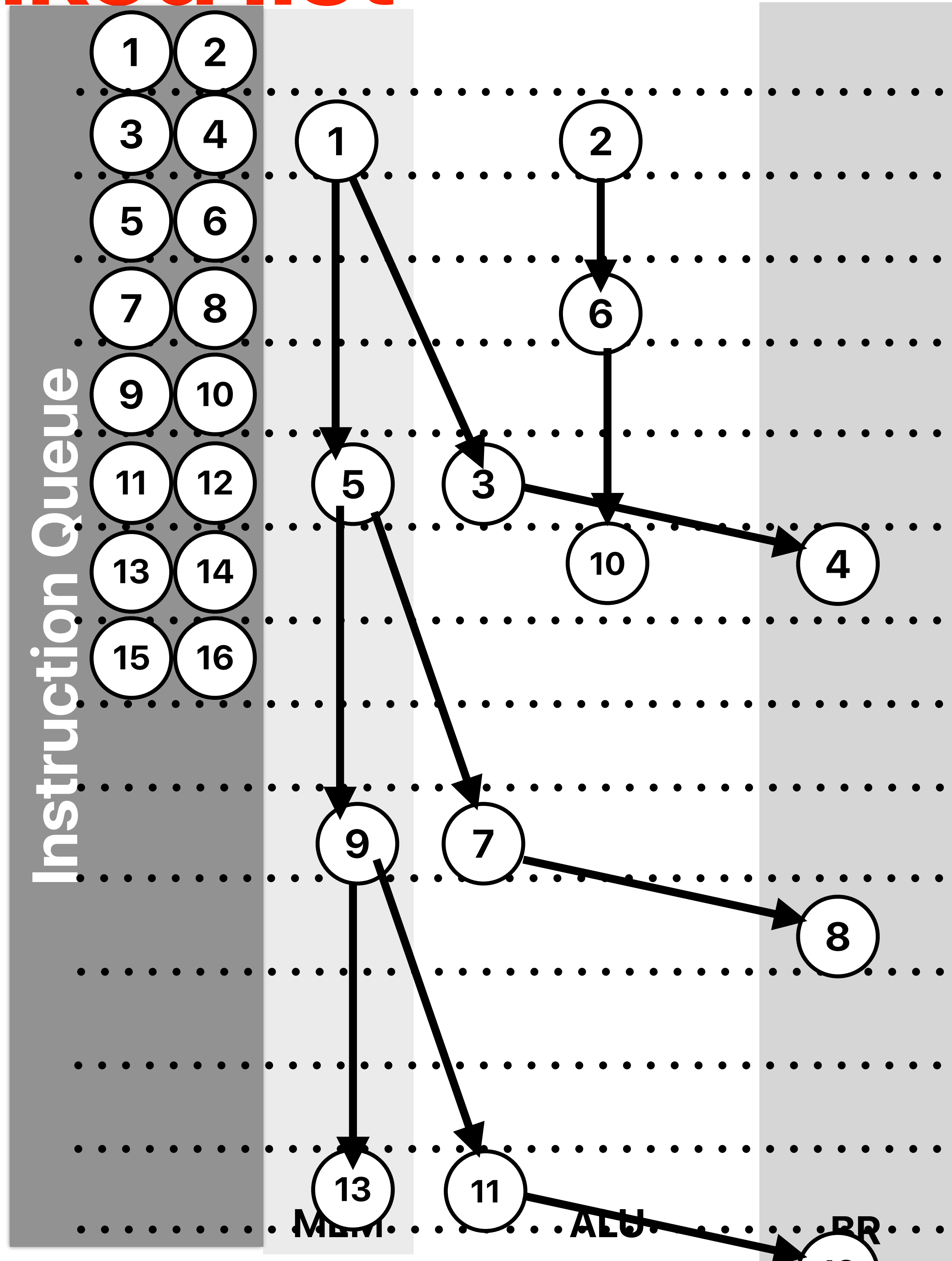
- Assume the current PC is already at instruction (1) and this linked list has only three nodes. This processor can fetch and issue 2 instructions per cycle, with exactly the same register renaming hardware and pipeline as we showed previously.
  Which of the following C state of the code snippet determines the performance?

```
A.do {
B.     number_of_nodes++;
C.     current = current->next;
D.} while ( current != NULL );
```

# What about "linked list"

## Dynamic instructions

```
①  .L3:    movq    8(%rdi), %rdi
②          addl    $1, %eax
③          testq   %rdi, %rdi
④          jne     .L3
⑤  .L3:    movq    8(%rdi), %rdi
⑥          addl    $1, %eax
⑦          testq   %rdi, %rdi
⑧          jne     .L3
⑨  .L3:    movq    8(%rdi), %rdi
⑩          addl    $1, %eax
⑪          testq   %rdi, %rdi
⑫          jne     .L3
⑬  .L3:    movq    8(%rdi), %rdi
⑭          addl    $1, %eax
⑮          testq   %rdi, %rdi
⑯          jne     .L3
```



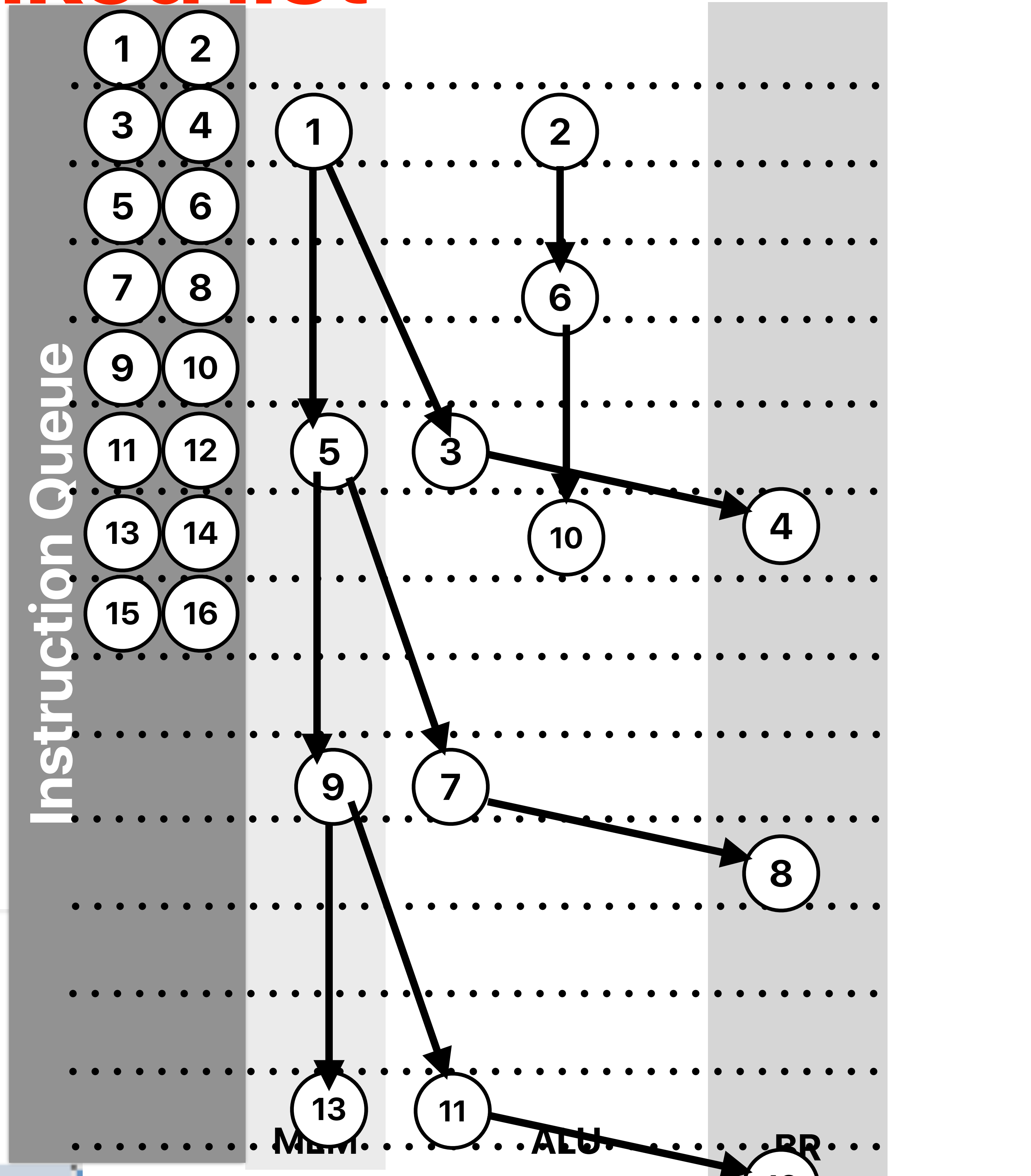
https://www.pollev.com/hungweitseng close in 1:30

# What about "linked list"

- For the following C code and it's translation in x86, **what's average CPI?** Assume the current PC is already at instruction (1) and this linked list has thousands of nodes. This processor can fetch and issue **2** instructions per cycle, with exactly the same register renaming hardware and pipeline as we showed previously.

```
do {
     number_of_nodes++;
     current = current->next;
} while ( current != NULL )
```

A. 0.5
B. 0.8
C. 1.0
D. 1.2
E. 1.5



# What about "linked list"

**Performance determined by the critical path**

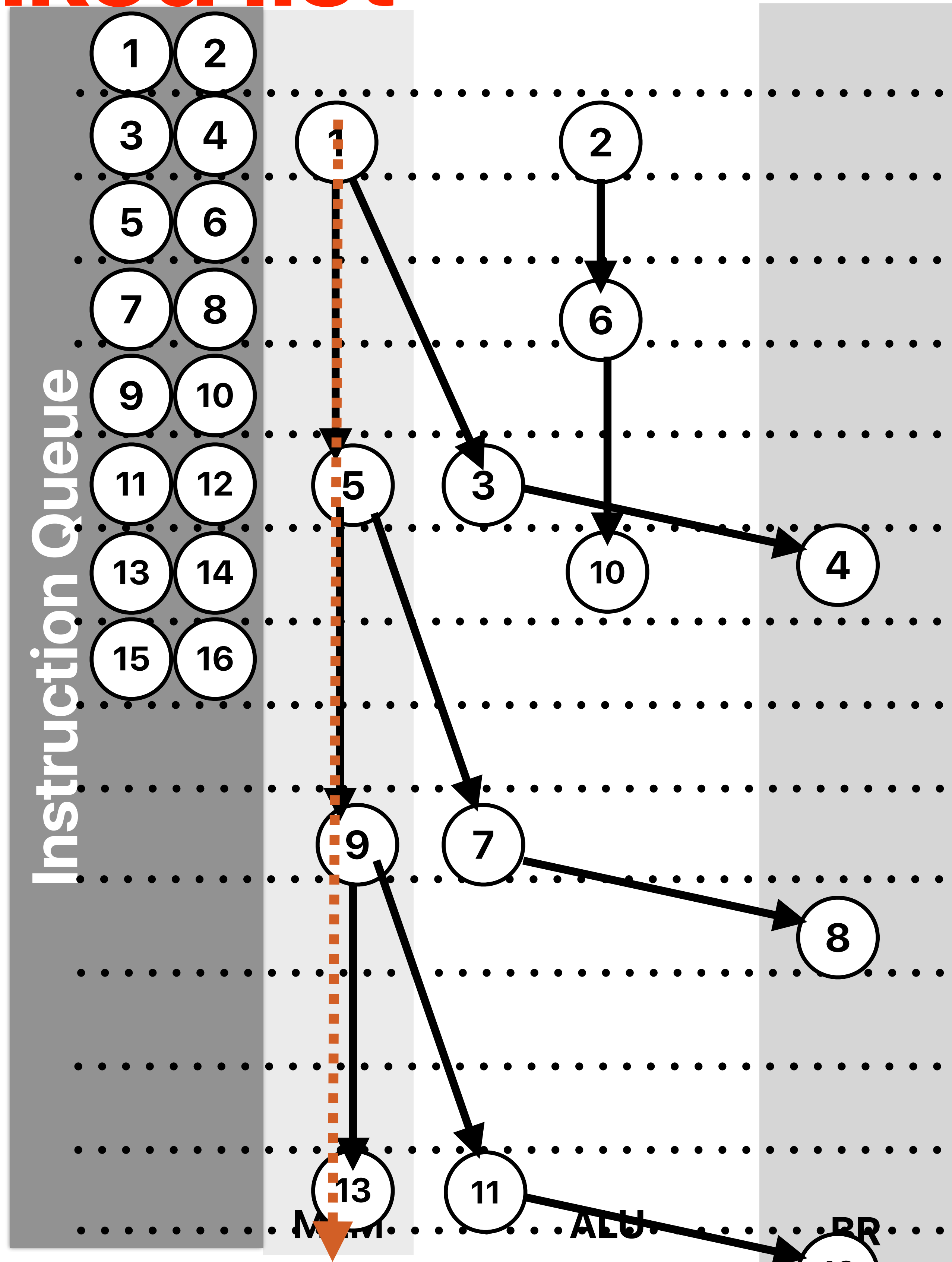
**4 cycles each iteration**

**4 instructions per iteration**

$$CPI = \frac{4}{4} = 1$$

```
do {
     number_of_nodes++;
     current = current->next;
} while ( current != NULL );
```

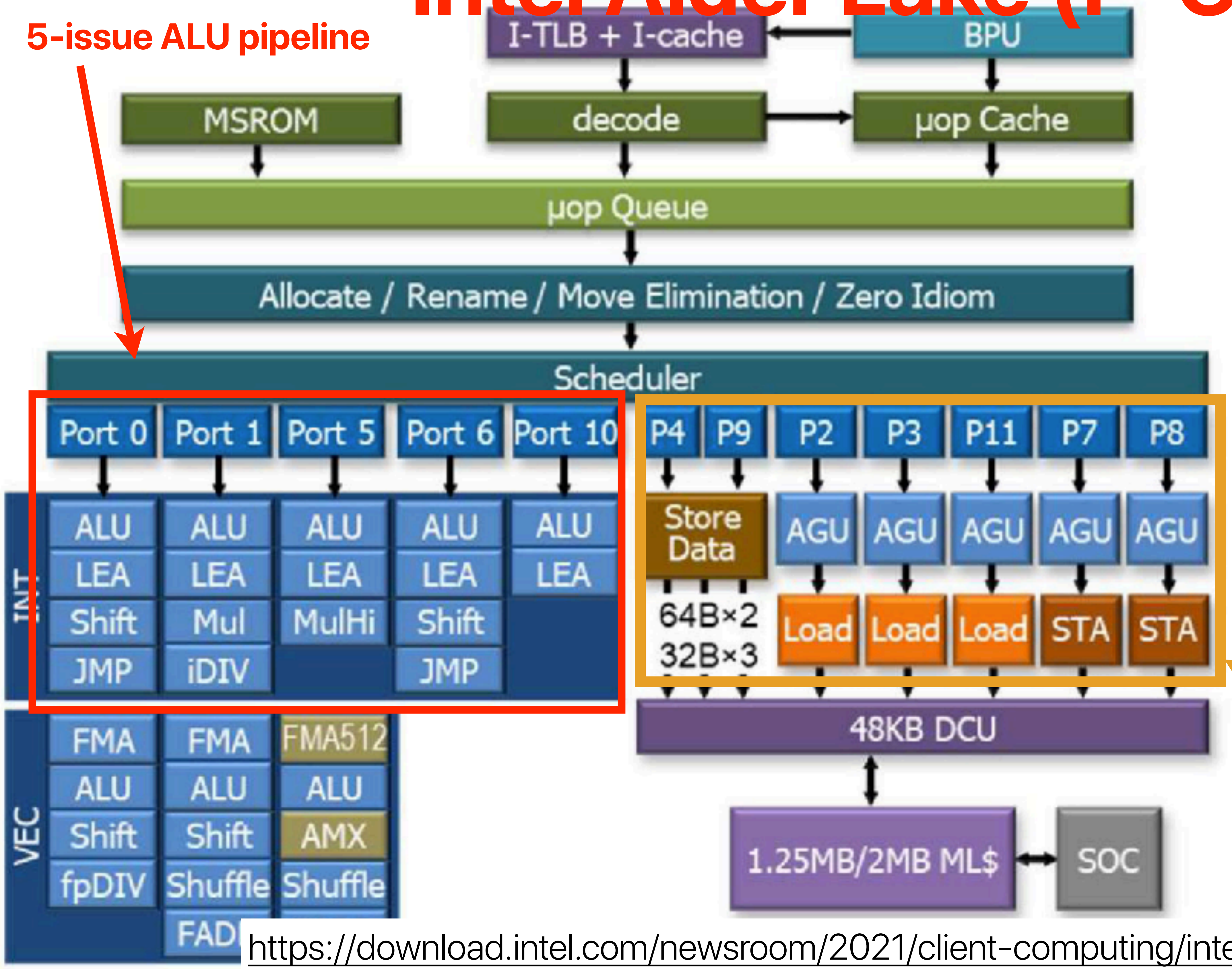
```
① .L3:    movq    8(%rdi), %rdi
②         addl    $1, %eax
③         testq   %rdi, %rdi
④         jne     .L3
```



# The pipelines of Modern Processors

# Intel Alder Lake (P-Core)

5-issue ALU pipeline




7-issue memory pipeline

$$MinCPI = \frac{1}{12}$$

$$MinINTInst.CPI = \frac{1}{5}$$

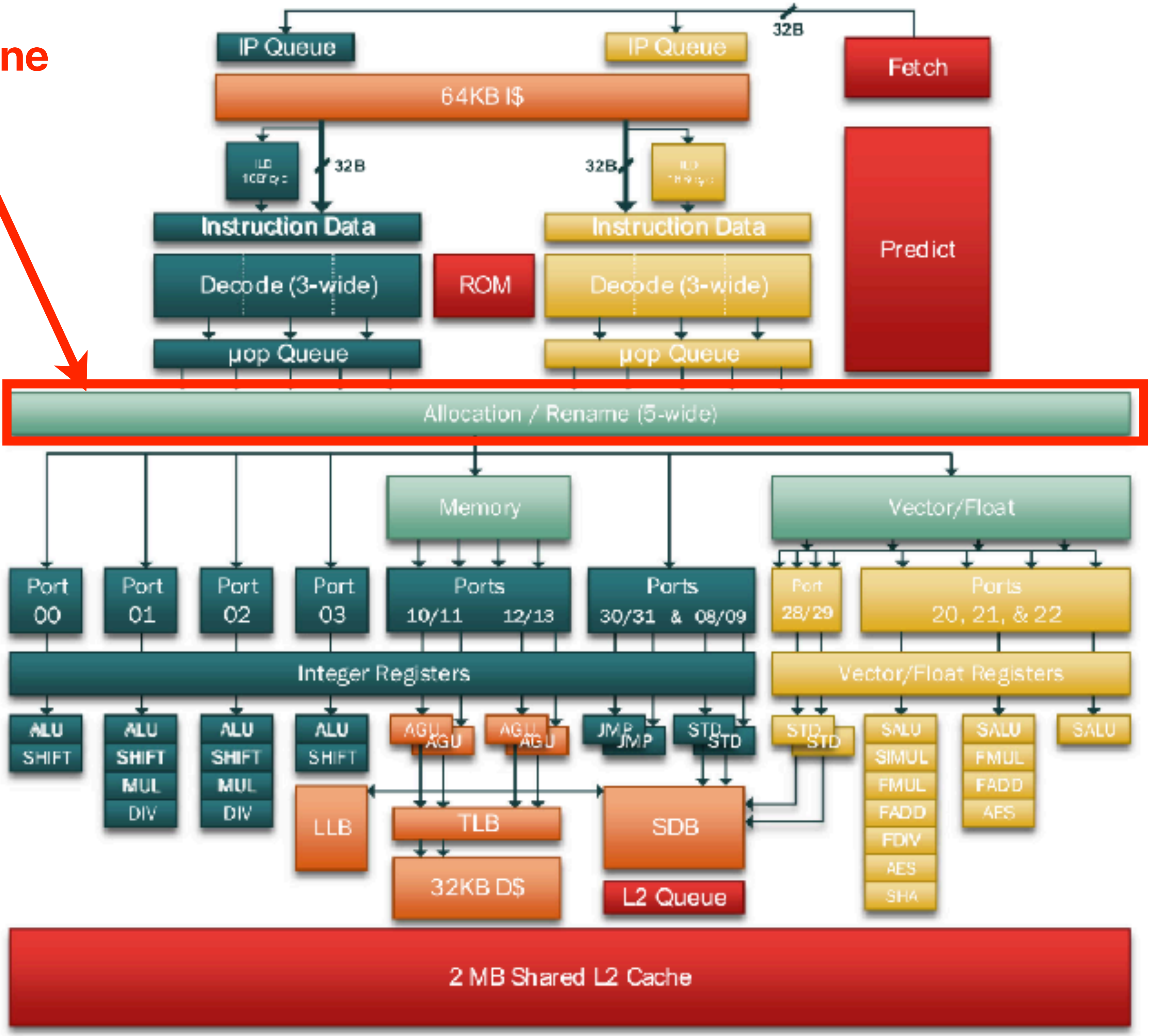
$$MinMEMInst.CPI = \frac{1}{7}$$

$$MinBRInst.CPI = \frac{1}{2}$$

https://download.intel.com/newsroom/2021/client-computing/intel-architecture-day-2021-presentation.pdf

# Intel Alder Lake (E-Core)

5-issue pipeline



# AMD Zen 3 (RyZen 5000 Series)

3-issue memory pipeline

4-issue integer pipeline + 1 additional branch

$$MinCPI = \frac{1}{8}$$

$$MinINTInst.CPI = \frac{1}{4}$$

$$MinMEMInst.CPI = \frac{1}{3}$$

$$MinBRInst.CPI = \frac{1}{2}$$

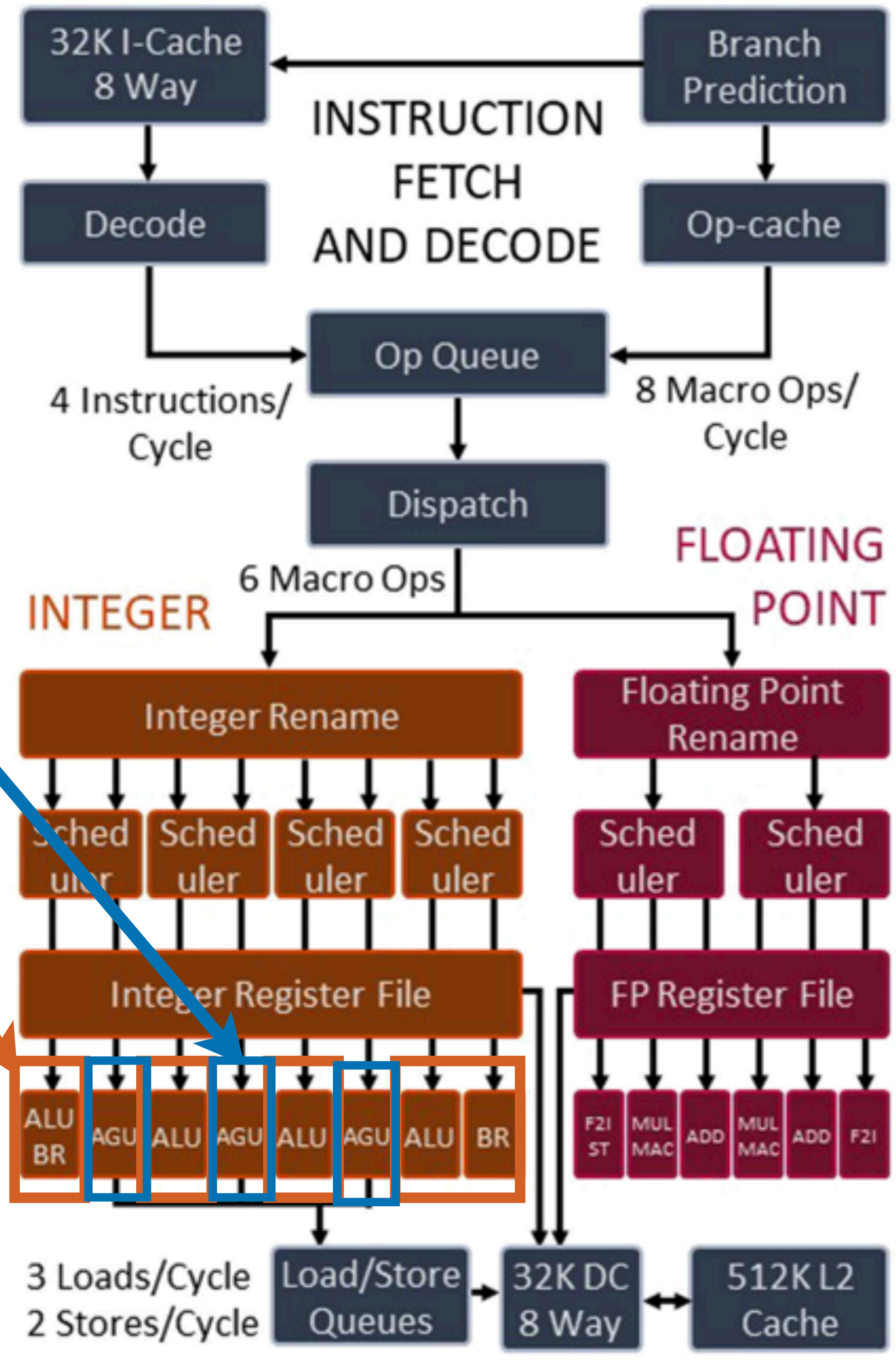


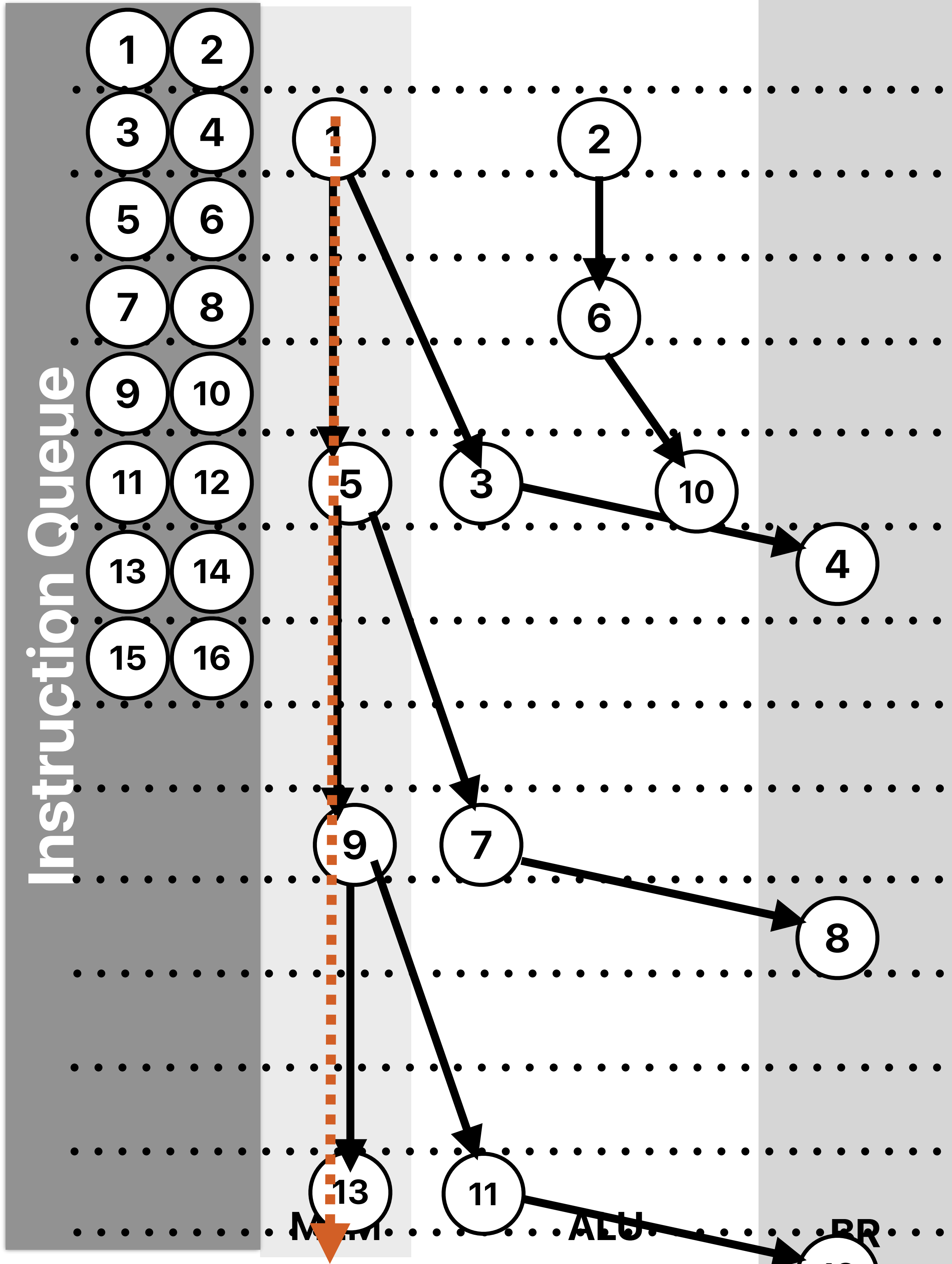

**FIGURE 1.** "Zen 3" block diagram.

# What if we have "unlimited" ILP — "linked list"

**Doesn't help that much!**
**— It's important that the programmer should write code that can exploit "ILP"**

```
do {
     number_of_nodes++;
     current = current->next;
} while ( current != NULL );
```

```
① .L3:    movq    8(%rdi), %rdi
②         addl    $1, %eax
③         testq   %rdi, %rdi
④         jne     .L3
```



# Summary: Characteristics of modern processor architectures

- Multiple-issue pipelines with multiple functional units available
  - Multiple ALUs
  - Multiple Load/store units
  - Dynamic OoO scheduling to reorder instructions whenever possible
- Cache — very high hit rate **if your code has good locality**
  - Very matured data/instruction prefetcher
- Branch predictors — very high accuracy **if your code is predictable**
  - Perceptron
  - TAGE

# Demo: Popcount

- The population count (or popcount) of a specific value is the number of set bits (i.e., bits in 1s) in that value.
- Applications
  - Parity bits in error correction/detection code
  - Cryptography
  - Sparse matrix
  - Molecular Fingerprinting
  - Implementation of some succinct data structures like bit vectors and wavelet trees.

# Demo: Popcount

- Given a 64-bit integer number, find the number of 1s in its binary representation.
- Example 1:
  Input: 59487
  Output: 10
  Explanation: 59487's binary representation is 0b1110100001011111

```
int main(int argc, char *argv[]) {

	uint64_t key = 0xdeadbeef;

	int count = 1000000000;
	uint64_t sum = 0;

	for (int i=0; i < count; i++)
	{
		sum += popcount(RandLFSR(key));
	}
	printf("Result: %lu\n", sum);
	return sum;
}
```

https://www.pollev.com/hungweitseng close in 1:30

# Five implementations

- Which of the following implementations will perform the best on modern pipeline processors?

A

```c
inline int popcount(uint64_t x){
  int c=0;
  while(x)  {
        c += x & 1;
        x = x >> 1;
    }
    return c;
}
```

B

```c
inline int popcount(uint64_t x) {
   int c = 0;
   while(x)     {
     c += x & 1;
     x = x >> 1;
     c += x & 1;
     x = x >> 1;
     c += x & 1;
     x = x >> 1;
     c += x & 1;
     x = x >> 1;
   }
   return c;
}
```

C

```c
inline int popcount(uint64_t x) {
     int c = 0;
     int table[16] = {0, 1, 1, 2, 1,
2, 2, 3, 1, 2, 2, 3, 2, 3, 3, 4};
     while(x)      {
          c += table[(x & 0xF)];
          x = x >> 4;
     }
     return c;
}
```

D

```c
inline int popcount(uint64_t x) {
     int c = 0;
     int table[16] = {0, 1, 1, 2, 1,
2, 2, 3, 1, 2, 2, 3, 2, 3, 3, 4};
     for (uint64_t i = 0; i < 16; i++)
     {
          c += table[(x & 0xF)];
          x = x >> 4;
     }
     return c;
  }
```

E

```c
inline int popcount(uint64_t x) {
     int c = 0;
     for (uint64_t i = 0; i < 16; i++)
     {
          switch((x & 0xF))
          {
               case 1: c+=1; break;
               case 2: c+=1; break;
               case 3: c+=2; break;
               case 4: c+=1; break;
               case 5: c+=2; break;
               case 6: c+=2; break;
               case 7: c+=3; break;
               case 8: c+=1; break;
               case 9: c+=2; break;
               case 10: c+=2; break;
               case 11: c+=3; break;
               case 12: c+=2; break;
               case 13: c+=3; break;
               case 14: c+=3; break;
               case 15: c+=4; break;
               default: break;
          }
          x = x >> 4;
     }
     return c;
}
```

# Five implementations

- Which of the following implementations will perform the best on modern pipeline processors?

**A**

```c
inline int popcount(uint64_t x){
  int c=0;
  while(x)  {
        c += x & 1;
        x = x >> 1;
    }
    return c;
}
```

**B**

```c
inline int popcount(uint64_t x) {
   int c = 0;
   while(x)     {
     c += x & 1;
     x = x >> 1;
     c += x & 1;
     x = x >> 1;
     c += x & 1;
     x = x >> 1;
     c += x & 1;
     x = x >> 1;
   }
   return c;
}
```

**C**

```c
inline int popcount(uint64_t x) {
     int c = 0;
     int table[16] = {0, 1, 1, 2, 1,
2, 2, 3, 1, 2, 2, 3, 2, 3, 3, 4};
     while(x)      {
         c += table[(x & 0xF)];
         x = x >> 4;
     }
     return c;
}
```

**D**

```c
inline int popcount(uint64_t x) {
     int c = 0;
     int table[16] = {0, 1, 1, 2, 1,
2, 2, 3, 1, 2, 2, 3, 2, 3, 3, 4};
     for (uint64_t i = 0; i < 16; i++)
     {
         c += table[(x & 0xF)];
         x = x >> 4;
     }
     return c;
 }
```

**E**

```c
inline int popcount(uint64_t x) {
     int c = 0;
     for (uint64_t i = 0; i < 16; i++)
     {
         switch((x & 0xF))
         {
             case 1: c+=1; break;
             case 2: c+=1; break;
             case 3: c+=2; break;
             case 4: c+=1; break;
             case 5: c+=2; break;
             case 6: c+=2; break;
             case 7: c+=3; break;
             case 8: c+=1; break;
             case 9: c+=2; break;
             case 10: c+=2; break;
             case 11: c+=3; break;
             case 12: c+=2; break;
             case 13: c+=3; break;
             case 14: c+=3; break;
             case 15: c+=4; break;
             default: break;
         }
         x = x >> 4;
     }
     return c;
}
```

# Announcements

- **Assignment 4** due next **Tuesday**
- **Reading Quiz 7** due **next Tuesday** before the lecture
- Hung-Wei's Office Hour changes to Tuesdays 10:30a—12:00p for the following two weeks (always check the calendar for the up-to-date information)
- Final Exam
  - 12/7 (in class) — 80 minutes paper-based. Same rules as the midterm, including CSMS comprehensive examine.
  - 12/11 6pm — 12/14 6pm (any 3-hour you pick) — open-ended questions, multiple choices, and programming assessments (TBD)

Computer
Science &
Engineering



つづく