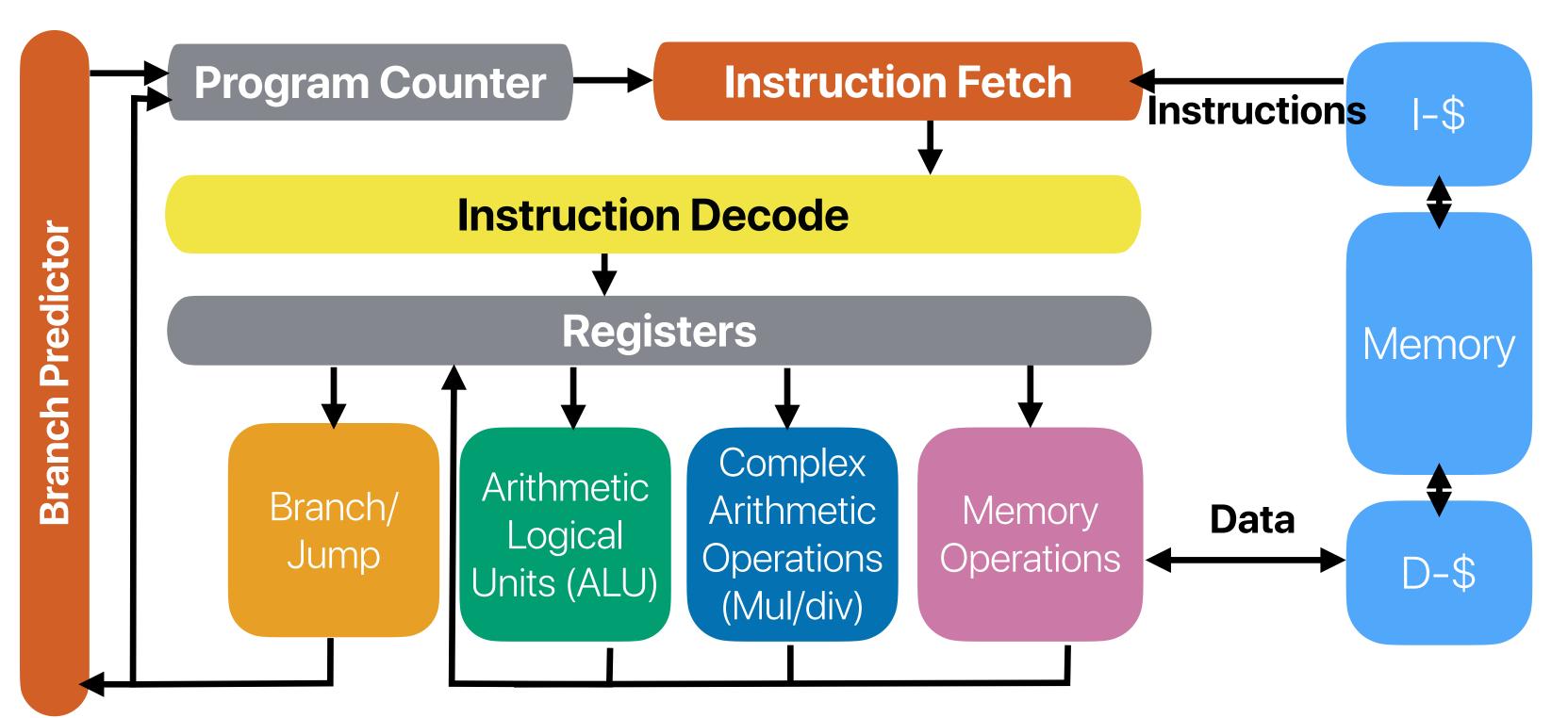
Modern Processor Design (IV): Don't wait for me

Hung-Wei Tseng

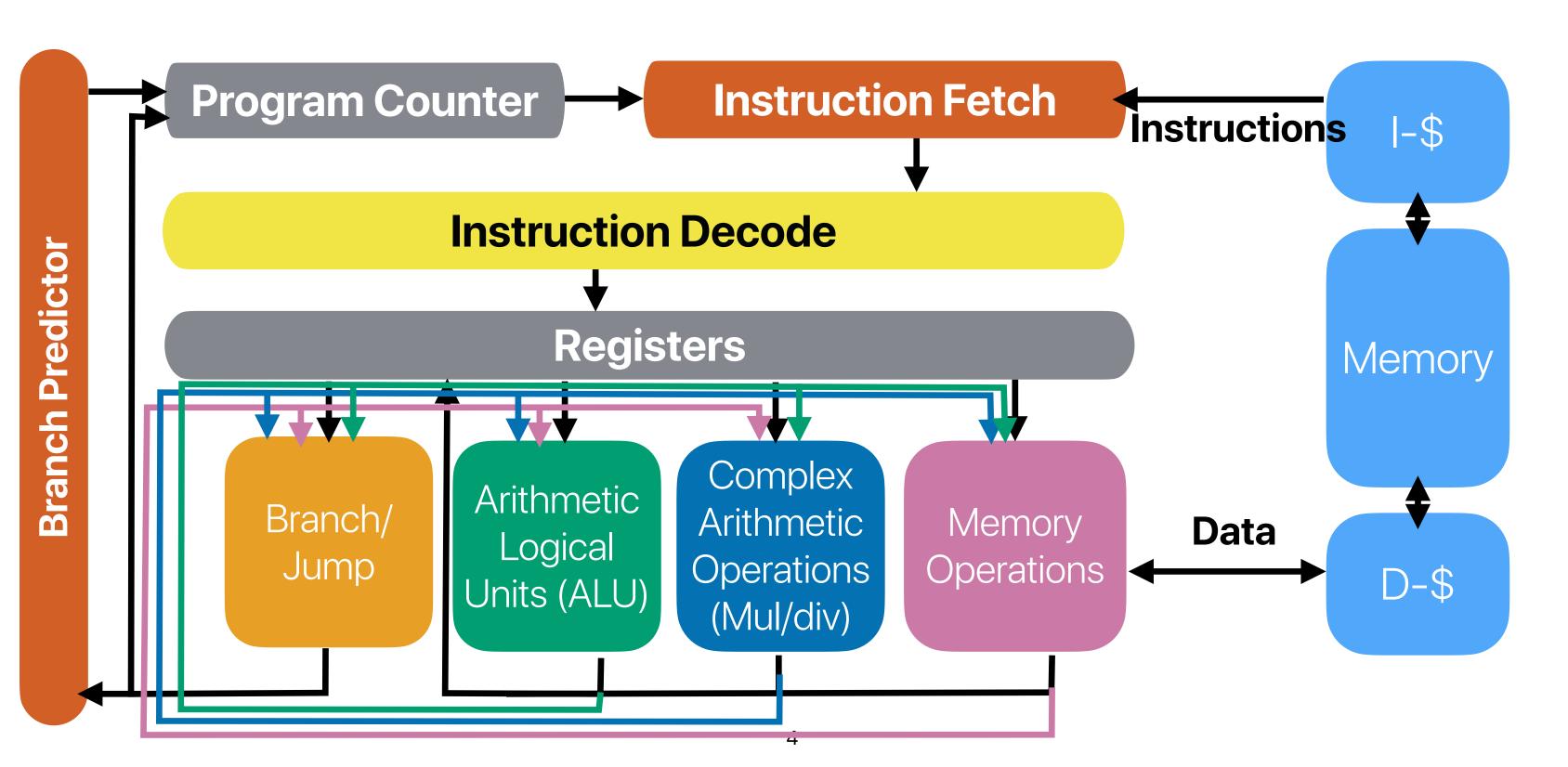
Recap: addressing hazards

- 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...

5

```
for(i = 0; i < count; i++) {
     int64_t temp = a[i];
     a[i] = b[i];
     b[i] = temp;
  .L9:
             (%rdi,%rax), %rsi
     movq
             (%rcx,%rax), %r8
     movq
             %r8, (%rdi,%rax)
     movq
             %rsi, (%rcx,%rax)
     movq
             $8, %rax
     addq
             %r9, %rax
     cmpq
     jne
             .L9
             (%rdi,%rax), %rsi
     movq
             (%rcx,%rax), %r8
     movq
             %r8, (%rdi,%rax)
     movq
             %rsi, (%rcx,%rax)
     movq
     addq
             $8, %rax
             %r9, %rax
     cmpq
             .L9
     jne
```

	IF	ID	ALU/BR/M1	M2	М3	M4/XORL	WB
1	(1)						
2	(2)	(1)				10 cycl	es for
3	(3)	(2)	(1)			instru	
4	(4)	(3)	(2)	(1)			
5	(4)	(3)		(2)	(1)	CPI =	1.43
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)

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

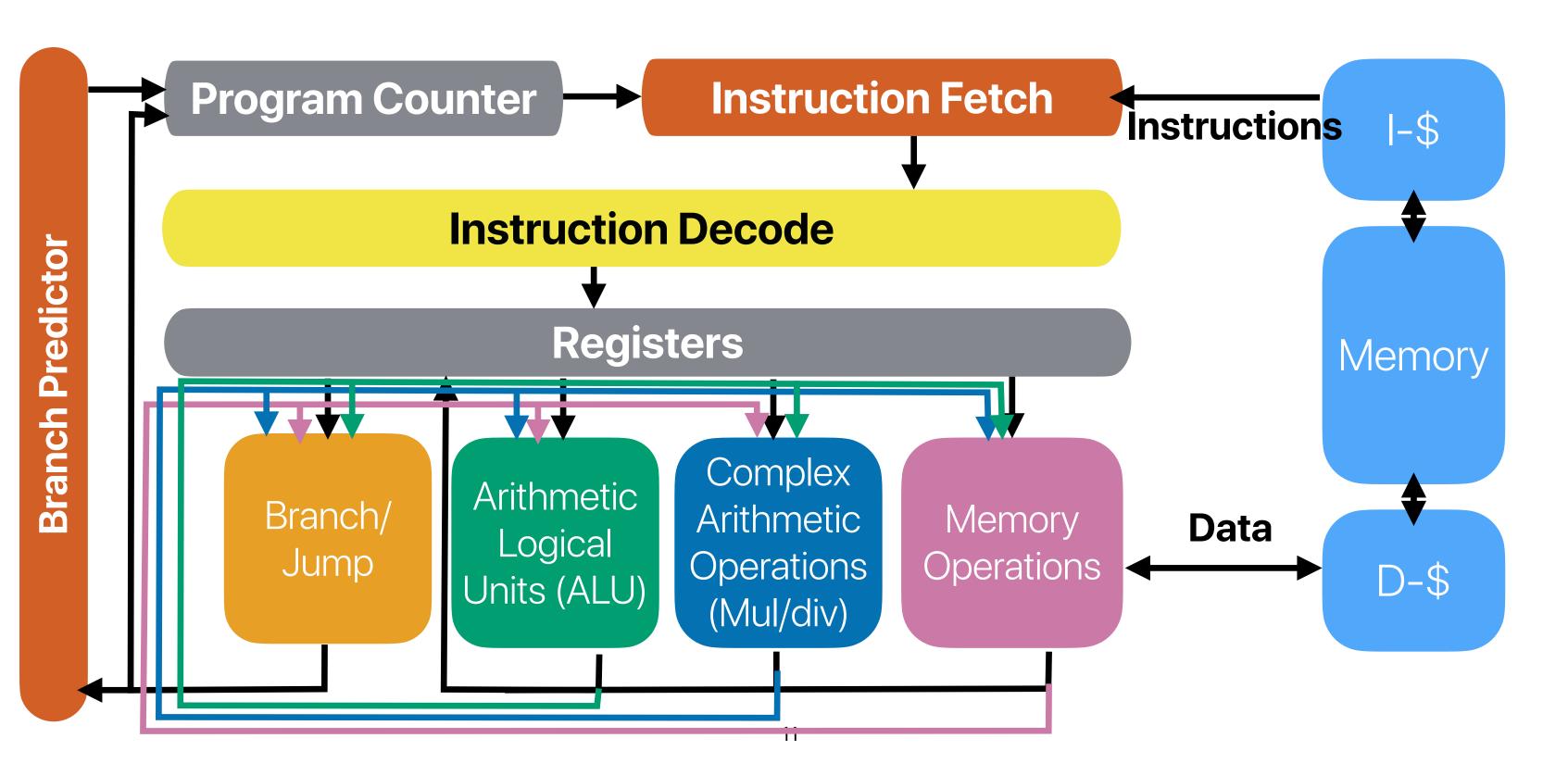
Can we really execute instructions OoO?

- 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

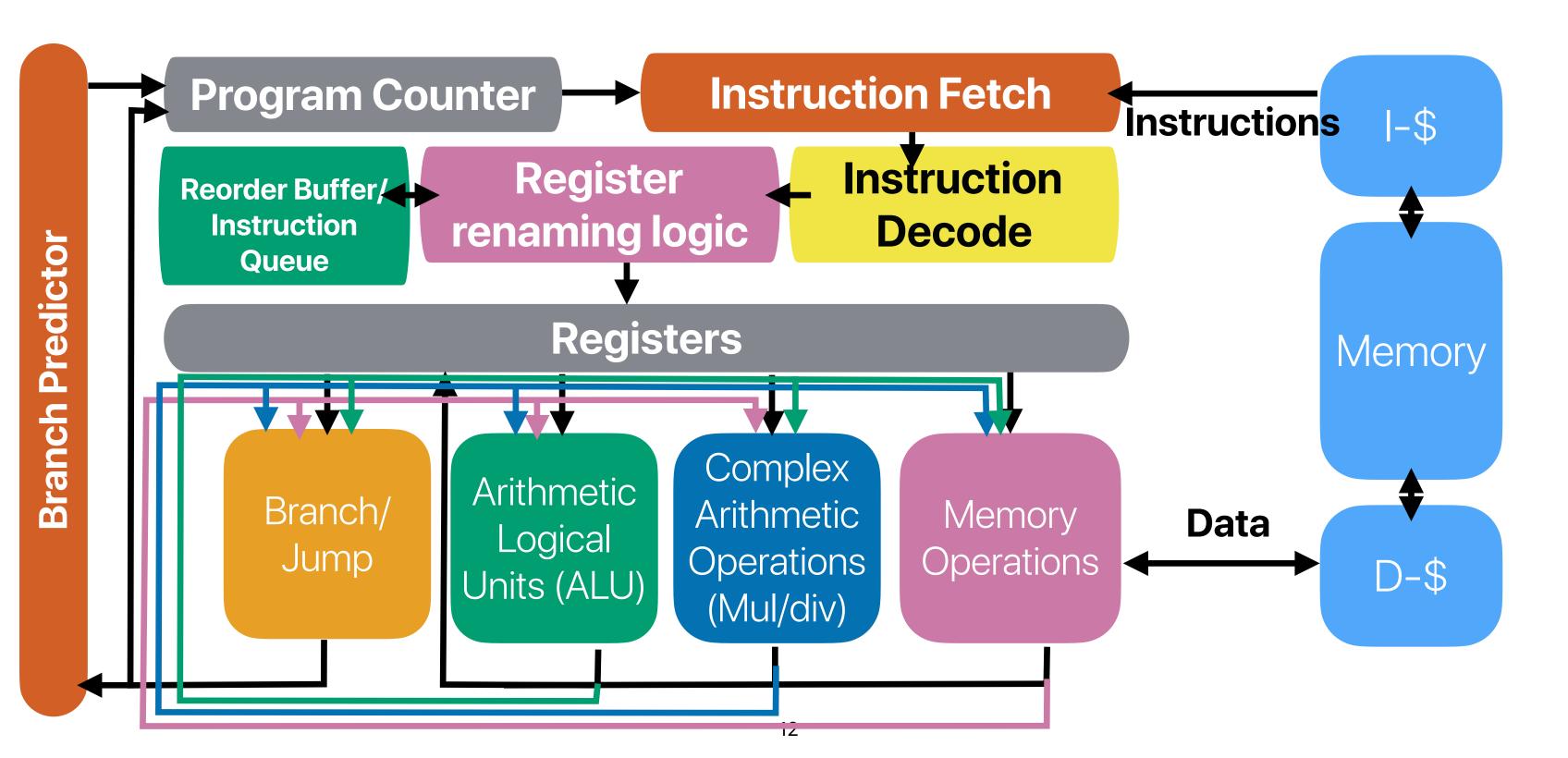
Speculative Execution

- **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



```
movq (%rdi,%rax), %rsi
movq (%rcx,%rax), %r8
movq %r8, (%rdi,%rax)
movq %rsi, (%rcx,%rax)
addq $8, %rax
cmpq %r9, %rax
jne .L9
movq (%rdi,%rax), %rsi
movq (%rcx,%rax), %r8
movq %r8, (%rdi,%rax)
movq %rsi, (%rcx,%rax)
addq $8, %rax
cmpq %r9, %rax
```

jne .L9

	IF	ID	REN	M1	M2 M3 N	14 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			
Р3				P8			
P4				P9			
P5				P10			

```
o movq (%rdi,%rax), %rsi → P1
```

- @ movq (%rcx,%rax), %r8
- movq %r8, (%rdi,%rax)
- movq %rsi, (%rcx,%rax)
- ⑤ addq \$8, %rax
- © cmpq %r9, %rax
- [→] jne .L9
- ® movq (%rdi,%rax), %rsi
- 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 ha	ave a instruc	ction 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			
Р3				P8			
P4				P9			
P5				P10			

```
    movq (%rdi,%rax), %rsi → P1
```

- ② movq (%rcx,%rax), %r8 → P2
- movq %r8, (%rdi,%rax)
- movq %rsi, (%rcx,%rax)
- ⊚ addq \$8, %rax
- © cmpq %r9, %rax
- [⊙] jne .L9
- ® movq (%rdi,%rax), %rsi
- 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 ha	ave a instruc	ction 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			
Р3				P8			
P4				P9			
P5				P10			

```
  movq (%rdi,%rax), %rsi → P1
```

- ② movq (%rcx,%rax), %r8 → P2
- movq %r8, (%rdi,%rax)
- movq %rsi, (%rcx,%rax)
- ⑤ addq \$8, %rax
- © cmpq %r9, %rax
- [⊙] jne .L9
- ® movq (%rdi,%rax), %rsi
- 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 instruc	tion 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			
Р3				P8			
P4				P9			
P5				P10			

```
    movq (%rdi,%rax), %rsi → P1
    movq (%rcx,%rax), %r8 → P2
    movq %r8, (%rdi,%rax)
    movq %rsi, (%rcx,%rax)
```

- ⑤ addq \$8, %rax → P3
- © cmpq %r9, %rax
- [⊙] jne .L9
- ® movq (%rdi,%rax), %rsi
- movq (%rcx,%rax), %r8
- 10 movq %r8, (%rdi,%rax)
- 10 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	М3	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	Р3
rcx	
rdi	
rsi	P1
r8	P2

	Valid	Value	In use		Valid	Value	In use
P1	0		1	P6			
P2	0		1	P7			
Р3	0		1	P8			
P4				P9			
P5				P10			

```
movq (%rdi,%rax), %rsi → P1
movq (%rcx,%rax), %r8 \rightarrow P2
movq %r8, (%rdi,%rax)
movq %rsi, (%rcx,%rax)
addq $8, %rax
```

- cmpq %r9, %rax
- jne .L9
- movq (%rdi,%rax), %rsi
- movq (%rcx,%rax), %r8
- 10 movq %r8, (%rdi,%rax)
- movq %rsi, (%rcx,%rax)
- addq \$8, %rax
- cmpq %r9, %rax
- jne .L9

	Offiny 1 Of					arrio oyolo
IF ID	REN	M1 M2 M3	M4 ALU	MUL	BR	ROB

	IF	ID	REN	M1	M2	M3	MA	ALU	MUL	ЬR	ROB
1			KLIN	IVI	1712	IVIO	101-4	ALO	IVIOL	- DK	- KOD
	(1)	(4)									
2	(2)	(1)						Ins	struct	ion (4) is
3	(3)	(2)	(1)							_	
4	(4)	(3)	(2)	(1)				rur	nning	ahead	10 k
5	(5)	(4)	(3)	(2)	(1)					21	
6	(6)	(5)	(3)(4)		(2)	(1)			(,	3)	
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											

	Physical Register
rax	Р3
rcx	
rdi	
rsi	P1
r8	P2

	Valid	Value	In use		Valid	Value	In use
P1	1		1	P6			
P2	0		1	P7			
Р3	0		1	P8			
P4				P9			
P5				P10			

```
movq (%rdi,%rax), %rsi → P1
movq (%rcx,%rax), %r8 → P2
movq %r8, (%rdi,%rax)
movq %rsi, (%rcx,%rax)
addq $8, %rax → P3
cmpq %r9, %rax
```

- jne .L9
- ® movq (%rdi,%rax), %rsi
- 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

Offig 1 Of t	Helli Call Have a	IIISH UCHOH at	tile Saille Cycle

	IF	ID	REN	M1	M2	М3	M4	ALU	MUL	ЬR	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	Р3
rcx	
rdi	
rsi	P1
r8	P2

	Valid	Value	In use		Valid	Value	In use
P1	1		1	P6			
P2	1		1	P7			
Р3	0		1	P8			
P4				P9			
P5				P10			

```
    movq (%rdi,%rax), %rsi → P1
    movq (%rcx,%rax), %r8 → P2
    movq %r8, (%rdi,%rax)
    movq %rsi, (%rcx,%rax)
    addq $8, %rax → P3
    cmpq %r9, %rax
```

- ∅ jne .L9
- ® movq (%rdi,%rax), %rsi
- movq (%rcx,%rax), %r8
- 10 movq %r8, (%rdi,%rax)
- 10 movq %rsi, (%rcx,%rax)
- 12 addq \$8, %rax
- ① cmpq %r9, %rax
- 14 jne .L9

	Offiny 1 of					
IF ID	REN	M1 M2 M3 M4	ALU	MUL	ЬR	ROB

	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	Р3								
rcx									
rdi									
rsi	P1								
r8	P2								

	Valid	Value	In use		Valid	Value	In use
P1	1		1	P6			
P2	1		1	P7			
Р3	0		1	P8			
P4				P9			
P5				P10			

```
    movq (%rdi,%rax), %rsi → P1
    movq (%rcx,%rax), %r8 → P2
    movq %r8, (%rdi,%rax)
    movq %rsi, (%rcx,%rax)
    addq $8, %rax → P3
    cmpq %r9, %rax
```

- ® movq (%rdi,%rax), %rsi → P4
- movq (%rcx,%rax), %r8
- 10 movq %r8, (%rdi,%rax)
- 11 movq %rsi, (%rcx,%rax)
- 12 addq \$8, %rax

jne .L9

- ① cmpq %r9, %rax
- 14 jne .L9

IF.	ID	REN	M1	M2	М3	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										

	Physical Register
rax	Р3
rcx	
rdi	
rsi	P4
r8	P2

13

14

15

16

	Valid	Value	In use		Valid	Value	In use
P1	1		1	P6			
P2	1		1	P7			
Р3	0		1	P8			
P4	0		1	P9			
P5				P10			

```
    movq (%rdi,%rax), %rsi → P1
    movq (%rcx,%rax), %r8 → P2
    movq %r8, (%rdi,%rax)
    movq %rsi, (%rcx,%rax)
    addq $8, %rax → P3
```

- ⑦ jne .L9
- ® movq (%rdi,%rax), %rsi → P4
- \odot movq (%rcx,%rax), %r8 \rightarrow P5
- 10 movq %r8, (%rdi,%rax)

cmpq %r9, %rax

- 11) movq %rsi, (%rcx,%rax)
- 12 addq \$8, %rax
- ① cmpq %r9, %rax
- 14 jne .L9

Only 1 of the	em can nave a	Instruc	tion at	tne same cycle
	L ,			

	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	Р3
rcx	
rdi	
rsi	P4
r8	P5

	Valid	Value	In use		Valid	Value	In use
P1	1		1	P6			
P2	1		1	P7			
Р3	1		1	P8			
P4	0		1	P9			
P5	0		1	P10			

(9)(10)

```
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
```

- ® movq (%rdi,%rax), %rsi → P4

 ® movq (%rcx,%rax), %r8 → P5
- 10 movq %r8, (%rdi,%rax)
- 10 movq %rsi, (%rcx,%rax)
- 12 addq \$8, %rax
- 13 cmpq %r9, %rax
- 14 jne .L9

	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)

(3)

	Physical Register
rax	Р3
rcx	
rdi	
rsi	P4
r8	P5

12 (12) (11)

13

14

15

16

	Valid	Value	In use		Valid	Value	In use
P1	1		1	P6			
P2	1		1	P7			
Р3	1		1	P8			
P4	0		1	P9			
P5	0		1	P10			

(7)

(4)(5)(6)

```
movq (%rdi,%rax), %rsi → P1
movq (%rcx,%rax), %r8 → P2
movq %r8, (%rdi,%rax)
movq %rsi, (%rcx,%rax)
addq $8, %rax → P3
cmpq %r9, %rax
```

- ⑦ jne .L9
 ⑧ movq (%rdi,%rax), %rsi → P4
- \odot movq (%rcx,%rax), %r8 \rightarrow P5
- 10 movq %r8, (%rdi,%rax)
- 11 movq %rsi, (%rcx,%rax)
- 12 addq \$8, %rax
- 13 cmpq %r9, %rax
- 14 jne .L9

	IF	ID	REN	M1	M2	М3	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)

	Physical Register
rax	Р3
rcx	
rdi	
rsi	P4
r8	P5

(10)(11)

(8)

13 (13) (12)

14

15

16

	Valid	Value	In use		Valid	Value	In use
P1	1		1	P6			
P2	1		1	P7			
Р3	1		1	P8			
P4	0		1	P9			
P5	0		1	P10			

(3)(4)(5)(6)(7)

```
    movq (%rdi,%rax), %rsi → P1
    movq (%rcx,%rax), %r8 → P2
    movq %r8, (%rdi,%rax)
    movq %rsi, (%rcx,%rax)
    addq $8, %rax → P3
    cmpq %r9, %rax
    jne .L9
```

- ® movq (%rdi,%rax), %rsi → P4

 ® movq (%rcx,%rax), %r8 → P5
- 10 movq %r8, (%rdi,%rax)
- ① movq %rsi, (%rcx,%rax)
- 12 addq \$8, %rax
- 13 cmpq %r9, %rax
- 14 jne .L9

			•								
	IF	ID	REN	M1	M2	М3	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											

Physical Register
P3
P4
P5

	Valid	Value	In use		Valid	Value	In use
P1	1		1	P6			
P2	1		1	P7			
Р3	1		1	P8			
P4	0		1	P9			
P5	0		1	P10			

```
movq (%rdi,%rax), %rsi → P1
movq (%rcx,%rax), %r8 \rightarrow P2
movq %r8, (%rdi,%rax)
movq %rsi, (%rcx,%rax)
addq $8, %rax
cmpq %r9, %rax
jne .L9
movq (%rdi,%rax), %rsi → P4
movq (%rcx,%rax), %r8 \rightarrow P5
movq %r8, (%rdi,%rax)
movq %rsi, (%rcx,%rax)
addq $8, %rax
                        → P6
cmpq %r9, %rax
```

jne .L9

			•								
	IF	ID	REN	M1	M2	М3	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)					

Physical Register
P6
P4
P5

16

	Valid	Value	In use		Valid	Value	In use
P1	1		1	P6	0		1
P2	1		1	P7			
Р3	1		1	P8			
P4	0		1	P9			
P5	0		1	P10			

```
movq (%rdi,%rax), %rsi → P1
movq (%rcx,%rax), %r8 \rightarrow P2
movq %r8, (%rdi,%rax)
movq %rsi, (%rcx,%rax)
addq $8, %rax
cmpq %r9, %rax
jne .L9
movq (%rdi,%rax), %rsi → P4
movq (%rcx,%rax), %r8 \rightarrow P5
movq %r8, (%rdi,%rax)
movq %rsi, (%rcx,%rax)
                       → P6
addq $8, %rax
```

cmpq %r9, %rax

jne .L9

		Only 1 of the	n <u>e</u> m can nave	truction	ruction at the same cycle			
		•						
IF	ID	REN	M1 M2 M3 M4	ALU	MUL	BR	ROB	
7.43								

			DEM		140	140	2.4.4				DOD
	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			
Р3	1		1	P8			
P4	0		1	P9			
P5	0		1	P10			

			16
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	
<i>-</i>	•		

jne

.L9

			Only 1 of the	nem	ı ca	n na	ave	a ins	truction	at the	e same cycle
	IF	ID	REN	M1	M2	М3	M4	ALU	MJL	ЬR	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)(0)(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)

			16
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 the	nem	ca	n ha	ave	a ins	truction	at the	e same cycle
	IF	ID	REN	M1	M2	M3	M4	ALU	MJL	ЬR	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)			
		(10)	(7)(8)(9)			(3)	(4)	(6)			(5)
	(12)		(9)(10)				(3)			(7)	(4)(5)(6)
		(12)	(10) (11)	(8)							(3)(4)(5)(0)(7)
		(13)	(10)(11)(12)	(9)	(8)						
		(14)	(10) (11) (13)		(9)	(8)	-	(12)			
		(15)	(10)(11)(14)			(9)	(8)	(13)			(12)
			(10)(14)(15)	(11)			(9)				(8) (12) (13)
18	(18)	(17)	(14)(15)(16)	(10)	(11)						(9) (12)(13)

			10
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 the	hem	ca	n ha	ave	a ins	truction	at the	e same cycle
	IF	ID	REN	M1	M2	M3	M4	ALU	MJL	ЬR	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)		(10) (11) (14)			(9)	(8)	(13)			(12)
	(17)		(10) (14) (15)	(11)			(9)				(8)(12)(13)
			(14) (15) (16)	(10)	(11)						(0) (12)(13)
19	(19)	(18)	(15) (16) (17)		(10)	(11)				(14)	(12)(13)

			TE
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		
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 the	hem	ca	n ha	ave	a ins	truction a	at the	e same cycle
	IF	ID	REN	M1	M2	M3	M4	ALU	MJL	ЬR	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)(0)(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)		(10) (14) (15)	(11)			(9)				(8)(12)(13)
		(17)	(14) (15) (16)	(10)	(11)						- (0) (12)(13)
		(18)	(15) (16) (17)		(10)	(11)				(14)	(12)(13)
20	(20)	(19)	(16) (17) (18)	(15)		(10)	(11)				(12)(13)(14)

		16
movq	(%rdi,%rax), %rsi →	P1
movq	(%rcx,%rax), %r8 →	P2
movq	%r8, (%rdi,%rax)	
movq	%rsi, (%rcx,%rax)	
addq	\$8, %rax →	P3
cmpq	%r9, %rax	
jne	.L9	
movq	(%rdi,%rax), %rsi →	P4
movq	(%rcx,%rax), %r8 →	P5
movq	%r8, (%rdi,%rax)	
movq	%rsi, (%rcx,%rax)	
addq	\$8, %rax →	P6
cmpq	%r9, %rax	
jne	.L9	
movq	(%rdi,%rax), %rsi	
movq	(%rcx,%rax), %r8	
movq	%r8, (%rdi,%rax)	
movq	%rsi, (%rcx,%rax)	
addq	\$8, %rax	
cmpq	%r9, %rax	
jne	.L9	
	movq movq addq cmpq jne movq movq addq cmpq jne movq movq movq addq cmpq	<pre>movq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rdi,%rax), %rsi → movq (%rcx,%rax), %r8 → movq %r8, (%rdi,%rax) movq %rsi, (%rcx,%rax) addq \$8, %rax →</pre>

			Only 1 of the	nem	ca	n ha	ave	a inst	truction a	at the	e same cycle
	IF	ID	REN	M1	M2	M3	M4	ALU	MJL	ЬR	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)		(9)(10)				(3)			(7)	(4)(5)(6)
13	(13)	(12)	(10)(11)	(8)							(3)(4)(5)(0)(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)

			10
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)	
1	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)	ine	19	

			Only 1 of t	hem	n ca	n ha	ave	a inst	truction	at the	same cycle
	IF	ID	REN	M1	M2	M3	M4	ALU	MJL	ЬR	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)(0)(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)

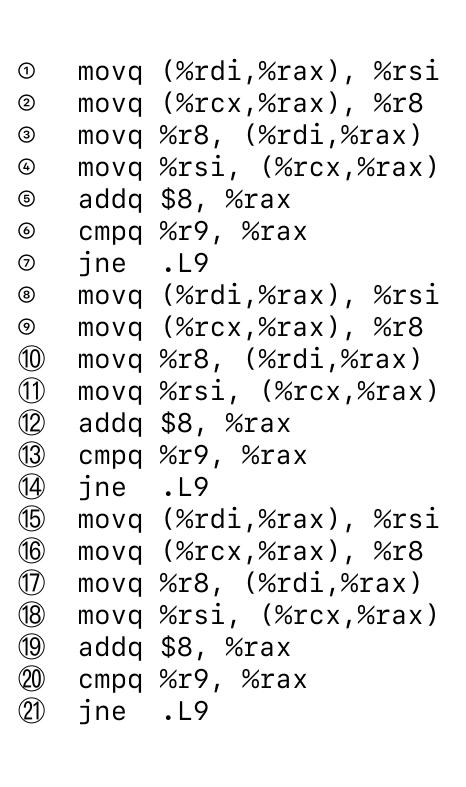
			10
1	movq	(%rdi,%rax), %rsi →	P1
2	movq	(%rcx,%rax), %r8 →	P2
3	movq	%r8, (%rdi,%rax)	
4	movq	%rsi, (%rcx,%rax)	
5	addq	\$8, %rax →	P3
6	cmpq	%r9, %rax	
7	jne	.L9	
8	movq	(%rdi,%rax), %rsi →	P4
9	movq	(%rcx,%rax), %r8 →	P5
10	movq	%r8, (%rdi,%rax)	
1	movq	%rsi, (%rcx,%rax)	
12	addq	\$8, %rax →	P6
13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

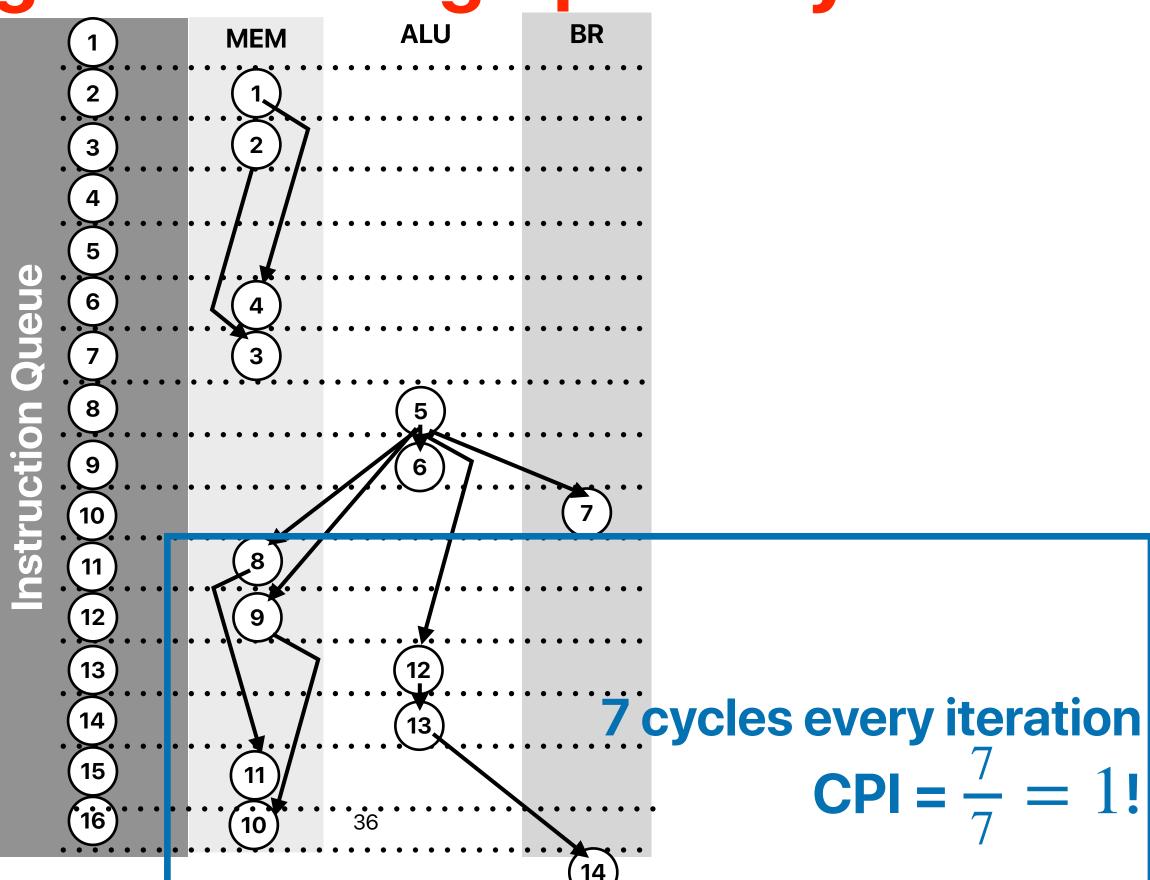
			Only 1 of the	nem	ca	n ha	ave	a ins	truction	at the	e same cycle
	IF	ID	REN	M1	M2	M3	M4	ALU	MJL	ЬR	ROB
1	(1)										
2	(2)	(1)									
3	(3)	(2)	(1)								
4	(4)	(3)	(2)	(1)							
5	(5)	(4)	(3)	(2)	(1)						
6	(6)	(5)	(3)(4)		(2)	(1)					
7	(7)	(6)	(3)(4)(5)			(2)	(1)				
8	(8)	(7)	(3)(5)(6)	(4)			(2)				(1)
9	(9)	(8)	(5)(6)(7)	(3)	(4)						(2)
10	(10)	(9)	(6)(7)(8)		(3)	(4)		(5)			
11	(11)		(7)(8)(9)			(3)	(4)	(6)			(5)
12	(12)	(11)	(9)(10)				(3)			(7)	(4)(5)(6)
	(13)		(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)		(10) (14) (15)	(11)			(9)				(8)(12)(13)
	(18)		(14) (15) (16)	(10)	(11)						(0) (12)(13)
19	(19)	(18)	(15) (16) (17)		(10)	(11)				(14)	(12)(13)
	(20)		(16) (17) (18)	(15)			(11)				(12)(13)(14)
	(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)

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	
<u>17</u>)	movq	%r8, (%rdi,%rax)	
18	movq	%rsi, (%rcx,%rax)	
19	addq	\$8, %rax	
20	cmpq	%r9, %rax	
21)	jne	.L9	

			Only 1 of the	nem	ca	n ha	ave	a ins	struction	at the	e same cycle
	IF	ID	REN	M1	M2	M3	M4	ALU	MJL	ЬR	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)(0)(7)
14	(14)	(13)	(10)(11)(12)	(9)	(8)				7 cycle	es to	r /
15	(15)	(14)	(10)(11)(13)		(9)	(8)		(12)	inctru	otio	-
16	(16)	(15)	(10)(11)(14)			(9)	(8)	(13)	instru	Guoi	(12)
17	(17)	(16)	(10)(14)(15)	(11)			(9)		CPI	= 1	(8) (12)(13)
18	(18)	(17)	(14)(15)(16)	(10)	(11)				O		(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)

Through data flow graph analysis





If CPI==1 the limitation?

Super Scalar

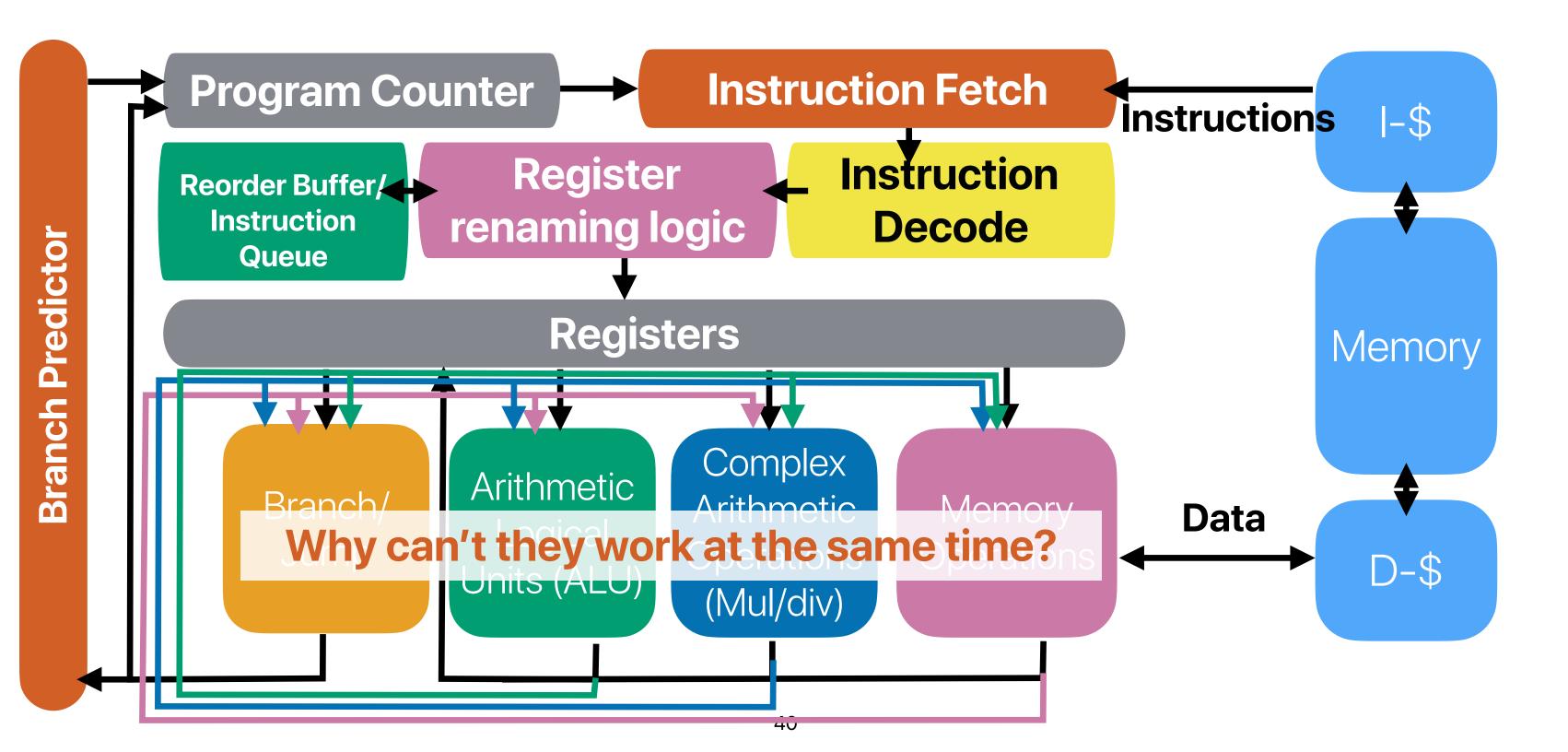
Superscalar

- 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

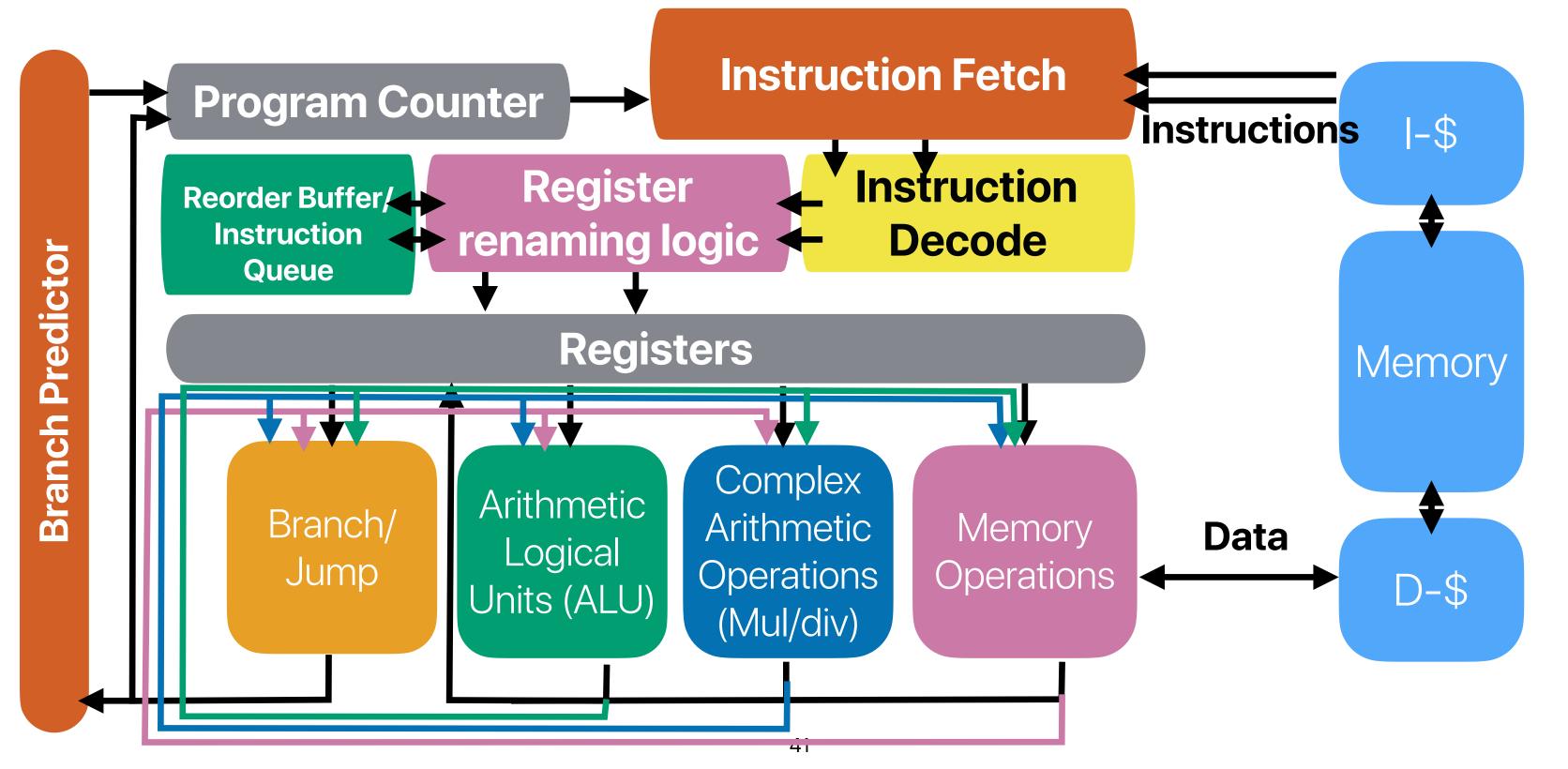
1

min(issue width, fetch width, decode width)

Register renaming + OoO + RoB



Register renaming + SuperScalar



		Z 13346 00			913		f thom can be	ave a instruction a	at the come evole
1	movq	(%rdi,%rax), %rsi→	P1			Jilly 2 Oi	Them can na	ave a instruction a	at the same cycle
2	movq	(%rcx,%rax), %r8 →	P2	IF	ID	REN	M1 M2 M3 M	4 ALU MUL BR	ROB
3	movq	%r8, (%rdi,%rax)		1 (1)(2)					
4	movq	%rsi, (%rcx,%rax)			(1)(2)				
5	addq	\$8, %rax →	Р3	3 (5)(6)	(3)(4)	(1)(2)			
6	cmpq	%r9, %rax		5					
7	jne	•		6					
8		(%rdi,%rax), %rsi →							
9	movq	(%rcx,%rax), %r8 →	P5	8					
10	movq	%r8, (%rdi,%rax)		9					
11	movq	%rsi, (%rcx,%rax)		10					
12	addq	\$8, %rax →	P6	11 12					
13	cmpq	%r9, %rax		13					
14	jne	.L9		14					
15	movq	(%rdi,%rax), %rsi		15					
16	movq	(%rcx,%rax), %r8		16					
17	movq	%r8, (%rdi,%rax)							
18	movq	%rsi, (%rcx,%rax)							
19	addq	\$8, %rax							

② cmpq %r9, %rax

jne

	movq (%rdi,%rax), %rsi → P1 Only "2" of them can have a instruction at the same cycle													
1	movq (%ra)	L,%rax), %	rsi 🔻	PT		O	riiy 2 Oi		m can nav	eam	Struction	Tat the Sa	ame cycle	
2	movq (%rc)	k,%rax), %	r8 →	P2	IF	ID	REN	M1	M2 M3 M4	ALU	MUL BR		ROB	
3	movq %r8,	(%rdi,%ra	x)		1 (1)(2)									
4	movq %rsi,	(%rcx,%r			2 (3)(4)		(4) (4)							
5	addq \$8, %	%rax	→	P3	3 (5)(6)		(1)(2)	(1)						
6	cmpq %r9,	%rax			4 (/)(o) 5	(5)(6)	(2)(3)(4)	(1)						
7	jne .L9				6									
	movq (%rdi													
9	movq (%rc)	k,%rax), %	r8 →	P5	8									
10	movq %r8,	(%rdi,%ra	x)		9									
11	movq %rsi,	(%rcx,%r	ax)		10									
12	addq \$8, %	%rax	>	P6	12									
13	cmpq %r9,	%rax			13									
14	jne .L9				14									
15	movq (%rdi	i,%rax), %	rsi		15									
16	movq (%rc)	k,%rax), %	r8		16									
17)	movq %r8,	(%rdi,%ra	x)											
18	movq %rsi,	(%rcx,%r	ax)											
19	addq \$8, %	%rax												

20 cmpq %r9, %rax

②1) jne

		2-155UE 3			·K	yış		the	Hall			9	TU			olo
1	movq	(%rdi,%rax), %rsi	\rightarrow	P1		U	illy 2 Oi	the	em can r	iave	a in			at the S	ame cyc	cie
2	movq	(%rcx,%rax), %r8	\rightarrow	P2	IF	ID	REN	M1	M2 M3	M4	ALU	MUL	BR		ROB	
3	movq	%r8, (%rdi,%rax)			1 (1)											
4	movq	%rsi, (%rcx,%rax)	•		2 (3)		(4) (0)									
⑤	addq	\$8, %rax	\rightarrow	Р3	3 (5) 4 (7)	(3)(4)	(1)(2) (2)(3)(4)	(1)								
6	cmpq	%r9, %rax			5 (9)(°	(3)(0) (7)(8)	(3)(4)(5)(6)	(2)	(1)							
7	jne					(2)(0)		(-)	(-)							
8		(%rdi,%rax), %rsi														
9	movq	(%rcx,%rax), %r8	\rightarrow	P5												
10	movq	%r8, (%rdi,%rax)														
11	movq	%rsi, (%rcx,%rax)														
12	addq	\$8, %rax	\rightarrow	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														

cmpq %r9, %rax

jne .L9

						913		the		CIIII			on of the o	omo ovolo
1	movq	(%rdi,%rax), %rsi→	P1			O	rily Z Oi	The	, ,	an nave		Structi	Tat the s	arrie Cycle
2	movq	(%rcx,%rax), %r8 →	P2		IF	ID	REN	M1	M2	M3 M4	ALU	MUL E	BR	ROB
3	movq	%r8, (%rdi,%rax)			(1)(2)									
4	movq	%rsi, (%rcx,%rax)	D O	_		(1)(2)	(4) (0)							
⑤	addq	\$8, %rax	P3	3	(5)(6) (7)(8)	(3)(4) (5)(6)	(1)(2) (2)(3)(4)	(1)						
6	cmpq	%r9, %rax		5	(<i>7</i>)(0) (9)(10)	(7)(8)	(3)(4)(5)(6)		(1)					
7	jne		D (6	(11)(12)	(9)(10)	(3)(4)(6)(7)(8)	(_/		(1)	(5)			
8		(%rdi,%rax), %rsi 🤻												
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												

45

② cmpq %r9, %rax

jne

			T	_		Ais		the						T U	t the		lo.
1	movq	(%rdi,%rax), %rsi→	P1			O	rily 2 Oi	The		an r	lave		Struc		at the	same cyci	le
2	movq	(%rcx,%rax), %r8 →	P2		IF	ID	REN	M1	M2	МЗ	M4	ALU	MUL	BR		ROB	
3	movq	%r8, (%rdi,%rax)			(1)(2)												
4	movq	%rsi, (%rcx,%rax)			(3)(4)												
(5)	addq	\$8, %rax →	P3	3		(3)(4)	(1)(2)	(4)									
6	cmpq	%r9, %rax		4 5	(7)(8) (9)(10)	(5)(6) (7)(8)	(2)(3)(4) (3)(4)(5)(6)	(1) (2)	(1)								
7	jne	. L9				(9)(10)	(3)(4)(6)(7)(8)	(2)		(1)		(5)					
8	movq	(%rdi,%rax), %rsi →	P4	7	(13)(14)	(11)(12)	(3)(4)(7)(9)(10)	(8)	(-)	(2)	(1)	(6)				(5)	
9		(%rcx,%rax), %r8 →															
		%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															
		(%rcx,%rax), %r8															
_	•	%r8, (%rdi,%rax)															
_	movq	%rsi, (%rcx,%rax)															
	-	\$8, %rax															

46

20 cmpq %r9, %rax

jne

				_		Ais		the			hov.			tion at the	oomo ovolo
1	movq	(%rdi,%rax), %rsi→	P1			O	rily 2 Oi			idi i	IIdV		Struc	tion at the	Same Cycle
2	movq	(%rcx,%rax), %r8 →	P2		IF	ID	REN	M1	M2	М3	M4	ALU	MUL	BR	ROB
3	movq	%r8, (%rdi,%rax)			(1)(2)										
4	movq	%rsi, (%rcx,%rax)			(3)(4)										
⑤	addq	\$8, %rax →	P3			(3)(4)	(1)(2)	(4)							
6	cmpq	%r9, %rax			(7)(8) (9)(10)	(5)(6) (7)(8)	(2)(3)(4) (3)(4)(5)(6)	(1) (2)	(1)						
7	jne	.L9			(11)(12)	(9)(10)	(3)(4)(6)(7)(8)	(2)		(1)		(5)			
8	movq	(%rdi,%rax), %rsi →	P4	7	(13)(14)	(11)(12)	(3)(4)(7)(9)(10)	(8)	(-)	(2)	(1)	(6)			(5)
9	movq	(%rcx,%rax), %r8 →	P5	8	(15)(16)	(13)(14)	(3)(9)(10)(11)(12)		(8)		(2)			(7)	(1)(5)(6)
10		%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		(%rcx,%rax), %r8													
17)	movq	%r8, (%rdi,%rax)													
	-	%rsi, (%rcx,%rax)													

② cmpq %r9, %rax

19 addq \$8, %rax

②1 jne .L9

	Z-13346 3	JT			Ais							9	T L	et the same avale
movq	(%rdi,%rax), %rsi→	P1			U	niy 2 Oi	the		an I	nave	e a in	Struc		at the same cycle
movq	(%rcx,%rax), %r8 →	P2		IF	ID	REN	M1	M2	М3	M4	ALU	MUL	BR	ROB
movq	%r8, (%rdi,%rax)													
movq	%rsi, (%rcx,%rax)													
addq	\$8, %rax	P3	3											
-	-		4					(4)						
• •	•						(2)		(1)		(5)			
_		P4					(8)	(2)		(1)				(5)
						(3)(9)(10)(11)(12)		(8)	(-)		(0)		(7)	(3) (5)(6)
			9	(17)(18)	(15) (16)	(9)(10)(11)(12)(13) (14)			(8)	\ _/	(12)		(-)	(2)(5)(6)(7)
•						(1-7)								
•		D ₆												
•	40 70 = 0.71	PO												
	-													
jne	.L9													
movq	(%rdi,%rax), %rsi													
mova	%r8, (%rdi,%rax)													
•	•													
-														
	movq movq addq cmpq jne movq movq addq cmpq jne movq movq movq movq movq	movq (%rdi,%rax), %rsi > movq (%rcx,%rax), %r8 > movq %r8, (%rdi,%rax) movq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rdi,%rax), %rsi > movq (%rcx,%rax), %r8 > movq %r8, (%rdi,%rax) movq %rsi, (%rcx,%rax)	movq (%rd1,%rax), %rs1 → P1 movq (%rcx,%rax), %rs1 → P2 movq %rs, (%rdi,%rax) movq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rdi,%rax), %rsi → P4 movq (%rcx,%rax), %rs → P5 movq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rdi,%rax), %rsi movq (%rdi,%rax), %rsi movq (%rcx,%rax), %rsi movq (%rcx,%rax), %rsi movq (%rcx,%rax), %rsi movq %rsi, (%rdi,%rax) movq %rsi, (%rdi,%rax) movq %rsi, (%rcx,%rax)	movq (%rdi,%rax), %rsi → P1 movq (%rcx,%rax), %r8 → P2 movq %r8, (%rdi,%rax) movq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rdi,%rax), %rsi → P4 movq (%rcx,%rax), %r8 → P5 movq %r8, (%rdi,%rax) movq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rdi,%rax), %rsi movq (%rdi,%rax), %rsi movq (%rdi,%rax), %rsi movq (%rcx,%rax), %r8 movq %r8, (%rdi,%rax) movq %rsi, (%rcx,%rax) movq %rsi, (%rcx,%rax)	movq (%rdi,%rax), %rsi → P1 movq (%rcx,%rax), %rs → P2 movq %rs, (%rdi,%rax) movq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rdi,%rax), %rsi → P4 movq (%rcx,%rax), %rs → P5 movq %rsi, (%rdi,%rax) movq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rdi,%rax), %rsi movq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rdi,%rax), %rsi movq (%rdi,%rax), %rsi movq (%rdi,%rax), %rsi movq (%rsi, (%rdi,%rax)) movq %rsi, (%rdi,%rax) movq %rsi, (%rdi,%rax) movq %rsi, (%rdi,%rax) movq %rsi, (%rcx,%rax)	movq (%rdi,%rax), %rsi > P1 movq (%rcx,%rax), %r8 > P2 movq %r8, (%rdi,%rax) movq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rdi,%rax), %rsi > P4 movq (%rdi,%rax), %rsi > P5 movq (%rcx,%rax), %r8 > P5 movq %r8, (%rdi,%rax) movq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rdi,%rax), %rsi movq (%rdi,%rax), %r8 movq %r8, (%rdi,%rax) movq %r8, (%rdi,%rax) movq %r8, (%rdi,%rax) movq %rsi, (%rcx,%rax)	movq (%rcx, %rax), %rsi → P1 movq (%rcx, %rax), %rsi → P2 movq %rsi, (%rcx, %rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rcx, %rax), %rsi → P4 movq (%rcx, %rax), %rsi → P5 movq (%rcx, %rax) movq %rsi, (%rcx, %rax) movq %rsi, (%rcx, %rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rcx, %rax) movq %rsi, (%rcx, %rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rdi, %rax) movq (%rdi, %rax) movq %rsi, (%rcx, %rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rdi, %rax), %rsi movq (%rcx, %rax), %rsi movq (%rcx, %rax), %rsi movq (%rcx, %rax), %rsi movq (%rsi, (%rdi, %rax)) movq %rsi, (%rdi, %rax) movq %rsi, (%rcx, %rax) movq %rsi, (%rcx, %rax) movq %rsi, (%rcx, %rax)	movq (%rcx,%rax), %rs1 → P1 movq (%rcx,%rax), %rs1 → P2 movq %rs, (%rdi,%rax) movq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rdi,%rax), %rsi → P4 movq (%rcx,%rax), %rsi → P5 movq (%rcx,%rax), %rsi → P5 movq %rsi, (%rcx,%rax) movq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rcx,%rax) movq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rdi,%rax), %rsi movq (%rcx,%rax), %rsi movq (%rcx,%rax), %rsi movq (%rcx,%rax), %rsi movq (%rsi, (%rcx,%rax)) movq %rsi, (%rcx,%rax) movq %rsi, (%rdi,%rax) movq %rsi, (%rcx,%rax) movq %rsi, (%rcx,%rax) movq %rsi, (%rcx,%rax) movq %rsi, (%rcx,%rax)	movq (%rd1,%rax), %rs1 → P1 movq (%rcx,%rax), %r8 → P2 movq %r8, (%rdi,%rax) movq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rdi,%rax), %rsi → P4 movq (%rcx,%rax), %r8 → P5 movq (%rdi,%rax), %rsi → P5 movq %r8, (%rdi,%rax) movq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %r9, %rax addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rdi,%rax), %rsi movq (%rcx,%rax), %rsi movq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rdi,%rax), %rsi movq (%rcx,%rax), %r8 movq %r8, (%rdi,%rax) movq %r8, (%rdi,%rax) movq %rsi, (%rcx,%rax)	movq (%rcx,%rax), %rs1 → P2 movq (%rcx,%rax), %rs1 → P2 movq %rs, (%rdi,%rax) movq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rdi,%rax), %rsi → P4 movq (%rdi,%rax), %rsi → P4 movq (%rcx,%rax), %rsi → P5 movq (%rcx,%rax), %rsi → P5 movq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rdi,%rax) movq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rdi,%rax), %rsi movq (%rsi, (%rcx,%rax)) movq %rsi, (%rcx,%rax) movq %rsi, (%rcx,%rax) movq %rsi, (%rcx,%rax) movq %rsi, (%rcx,%rax)	movq (%rd1,%rax), %rs1 → P1 movq (%rcx,%rax), %r8 → P2 movq %r8, (%rdi,%rax) movq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rdi,%rax), %rsi → P4 movq (%rcx,%rax), %r8 → P5 movq (%rcx,%rax), %r8 → P5 movq %r8, (%rdi,%rax) movq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rdi,%rax) movq %rsi, (%rcx,%rax) addq \$8, %rax provq %rsi, (%rcx,%rax) movq %rsi, (%rcx,%rax) movq (%rdi,%rax), %rsi movq (%rdi,%rax), %rsi movq (%rcx,%rax), %r8 movq %r8, (%rdi,%rax) movq (%rcx,%rax), %r8 movq %r8, (%rdi,%rax) movq (%rcx,%rax), %r8 movq %rsi, (%rcx,%rax) movq %rsi, (%rcx,%rax) movq %rsi, (%rcx,%rax) movq %rsi, (%rcx,%rax)	movq (%rd1,%rax), %rs1 → P1 movq (%rcx,%rax), %r8 → P2 movq %rs, (%rdi,%rax) movq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rdi,%rax), %rsi → P4 movq (%rcx,%rax), %rsi → P5 movq (%rdi,%rax), %rsi → P5 movq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rcx,%rax), %r8 → P5 movq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq %rsi, (%rcx,%rax) addq \$8, %rax cmpq %r9, %rax jne .L9 movq (%rdi,%rax), %rsi movq (%rcx,%rax), %rsi movq (%rcx,%rax), %r8 movq %rsi, (%rdi,%rax) movq %rsi, (%rdi,%rax) movq %rsi, (%rdi,%rax) movq %rsi, (%rcx,%rax) movq %rsi, (%rcx,%rax)	movq (%rcx, %rax), %r8 → P2 IF ID REN M1 M2 M3 M4 ALU MUL movq %r8, (%rdi, %rax) movq %r8, (%rdi, %rax) movq %rsi, (%rcx, %rax) → P3 addq \$8, %rax → P3 cmpq %r9, %rax → P4 jne .L9 → P4 movq (%rdi, %rax), %rsi → P4 → P5 movq (%rcx, %rax), %r8 → P5 → P5 movq %r8, (%rdi, %rax) → P6 cmpq %r9, %rax → P6 cmpq %r9, %rax → P6 movq (%rdi, %rax), %rsi movq (%rdi, %rax), %rsi movq (%rcx, %rax), %r8 movq %r8, (%rdi, %rax), %rsi movq (%rsi, (%rdi, %rax), %r8 movq %r8, (%rdi, %rax), movq %rsi, (%rcx, %rax) movq %r8, (%rdi, %rax)	movq (%rcx, %rax), %r8 → P2

48

20 cmpq %r9, %rax

jne

		(0/mali 0/max) 0/mai	9 7		NC	Ais		the			l I I		9	T U	t the same avale
1	movq	(%rdi,%rax), %rsi	> P1			O	rily 2 Oi			jan i	nav	e a in	Struc	tion a	it the same cycle
2	movq	(%rcx,%rax), %r8	→ P2		IF	ID	REN	M1	M2	М3	M4	ALU	MUL	BR	ROB
3	movq	%r8, (%rdi,%rax)			(1)(2)										
4	movq	%rsi, (%rcx,%rax)			(3)(4)										
(5)	addq	\$8, %rax	→ P3	3	(5)(6)	(3)(4)	(1)(2)	(4)							
6	cmpq	%r9, %rax		5	(7)(8) (9)(10)	(5)(6) (7)(8)	(2)(3)(4) (3)(4)(5)(6)	(1) (2)	(1)						
7	jne			6	(11) (12)	(9)(10)	(3)(4)(6)(7)(8)	(2)		(1)		(5)			
8	movq	(%rdi,%rax), %rsi	→ P4	7	(13)(14)	(11) (12)	(3)(4)(7)(9)(10)	(8)		(2)	(1)	(6)			(5)
9	movq	(%rcx,%rax), %r8	→ P5	8	(15) (16)	(13) (14)	(3)(9)(10)(11)(12)	(4)	(8)		(2)			(7)	-(1)(5)(6)
10		%r8, (%rdi,%rax)		9	(17)(18)	(15) (16)	(9)(10)(11)(12)(13) (14)		(4)			(12)			(2) (5)(6)(7)
11	movq	%rsi, (%rcx,%rax)		10	(19)(20)	(17) (18)	(10)(11)(12)(13)(14) (15)(16)	(9)	(3)	(4)	(8)	(13)			(5)(6)(7)(12)
12	addq	\$8, %rax	> P6												
13	cmpq	%r9, %rax													
14	jne	.L9													
15)	movq	(%rdi,%rax), %rsi													
_		(%rcx,%rax), %r8													
17	movq	%r8, (%rdi,%rax)													
_	-	%rsi, (%rcx,%rax)													

- 19 addq \$8, %rax
- 20 cmpq %r9, %rax
- ②1) jne .L9

					916	nly "2" of	tho						tion	at the same cycle
1	movq	(%rdi,%rax), %rsi→	P1		O	illy 2 Oi			all	llave		Struc		at the Same Cycle
2	movq	(%rcx,%rax), %r8 →	P2	IF	ID	REN	M1	M2	M3	M4	ALU	MUL	BR	ROB
3	movq	%r8, (%rdi,%rax)		1 (1)(2)										
4	movq	%rsi, (%rcx,%rax)			(1)(2)	/// // /								
5	addq	\$8, %rax →	P3		(3)(4)	(1)(2)	(4)							
6	cmpq	%r9, %rax		4 (7)(8) 5 (9)(10)	(5)(6) (7)(8)	(2)(3)(4) (3)(4)(5)(6)	(1) (2)	(1)						
7	jne	.L9		6 (11)(12)	(<i>7</i>)(<i>0</i>) (9)(10)	(3)(4)(6)(7)(8)	(2)	(1) (2)	(1)		(5)			
8	movq	(%rdi,%rax), %rsi→	P4		(11) (12)	(3)(4)(7)(9)(10)	(8)	(-)	(2)	(1)	(6)			(5)
9		(%rcx,%rax), %r8 →			(13) (14)	(3)(9)(10)(11)(12)		(8)		(2)			(7)	(1)(5)(6)
10		%r8, (%rdi,%rax)		9 (17)(18)	(15) (16)	(9)(10)(11)(12)(13) (14)	(3)	(4)	(8)		(12)			(2) (5)(6)(7)
$\widetilde{11}$	•	%rsi, (%rcx,%rax)		10 (19)(20)	(17) (18)	(10)(11)(12)(13)(14) (15)(16)		(3)			(13)			(5)(6)(7)(12)
12	•	•	P6	11	(19) (20)	(10)(12)(13)(15)(16) (17)(18)	(11)	(9)	(3)	(4)			(14)	(5)(6)(7)(8)(12)(13)
13	-	%r9, %rax												
14	jne	.L9												
15		(%rdi,%rax), %rsi												
16	_	(%rcx,%rax), %r8												
_	•	%r8, (%rdi,%rax)												
_		%rsi, (%rcx,%rax)												
	-	\$8, %rax												
	aaaq	ΨΟ, /0Ι α Λ												

cmpq %r9, %rax

jne

		Z 13346 06				Sign	ply "2" of								at the same cycle
1	movq	(%rdi,%rax), %rsi→	P1			O	illy 2 Oi			Jall	llavt		Struc		of the Same Cycle
2	movq	(%rcx,%rax), %r8 →	P2		IF	ID	REN	M1	M2	М3	M4	ALU	MUL	BR	ROB
3	movq	%r8, (%rdi,%rax)		1	(1)(2)										
4	movq	%rsi, (%rcx,%rax)		2		(1)(2)									
(5)	addq	\$8, %rax →	P3	3	(5)(6)	(3)(4)	(1)(2)	443							
6	-	%r9, %rax		4	(7)(8)	(5)(6)	(2)(3)(4)	(1)	(4)						
7	jne	•		5	(9)(10) (11)(12)	(7)(8) (9)(10)	(3)(4)(5)(6) (3)(4)(6)(7)(8)	(2)	(1)	(1)		(5)			
8	_	(%rdi,%rax), %rsi→	P4		(13)(14)	(11)(12)	(3)(4)(7)(9)(10)	(8)	(2)	(2)	(1)	(5) (6)			(5)
9	mova	(%rcx,%rax), %r8 →	P5	8	(15) (16)	(13)(14)	(3)(9)(10)(11)(12)	(4)	(8)	(-)	(2)	(0)		(7)	- (1)(5)(6)
10		%r8, (%rdi,%rax)		9	(17)(18)	(15)(16)	(9)(10)(11)(12)(13) (14)	(3)		(8)		(12)			(2) (5)(6)(7)
	•	%rsi, (%rcx,%rax)		10	(19)(20)	(17)(18)	(10)(11)(12)(13)(14) (15)(16)	(9)		(4)	(8)	(13)			(5)(6)(7)(12)
_	•	ΦΟ 0/τον	P6	11		(19)(20)	(10)(12)(13)(15)(16) (17)(18)	(11)	(9)	(3)	(4)			(14)	(5)(6)(7)(8)(12)(13)
12	-	\$8, %rax	PO	12			(12)(13)(15)(16)(17) (18)(19)(20)	(10)	(11)	(9)	(3)				(4)(5)(6)(7)(8)(12)(13)(14)
13	•	%r9, %rax													
14	jne	.L9													
15	movq	(%rdi,%rax), %rsi													
16	movq	(%rcx,%rax), %r8													
17	movq	%r8, (%rdi,%rax)													
_	=	%rsi, (%rcx,%rax)													

@ cmpq %r9, %rax

19 addq \$8, %rax

① jne .L9

					GIZ							Gruetien	ot the come evelo
1	movq	(%rdi,%rax), %rsi→	P1		U	riiy 2 Oi	The		anı	IIav	eam	Struction	rat the same cycle
2	movq	(%rcx,%rax), %r8 🗦	P2	IF	ID	REN	M ₁	M2	М3	M4	ALU	MUL BR	ROB
3	movq	%r8, (%rdi,%rax)		1 (1)(2)									
4	movq	%rsi, (%rcx,%rax)		2 (3)(4)	(1)(2)								
5	addq	\$8, %rax	P3		(3)(4)	(1)(2)	(4)						
6	-	%r9, %rax		4 (7)(8) F (9)(10)	(5)(6)	(2)(3)(4)	(1)	(4)					
7		.L9		5 (9)(10) 6 (11)(12)	(7)(8) (9)(10)	(3)(4)(5)(6) (3)(4)(6)(7)(8)	(2)	(2)	(1)		(5)		
8	_	(%rdi,%rax), %rsi	P4		(11) (12)	(3)(4)(7)(9)(10)	(8)	(2)	(2)	(1)	(6)		(5)
9		(%rcx,%rax), %r8		8 (15)(16)	(13) (14)	(3)(9)(10)(11)(12)		(8)	(-/	(2)	(0)	(7)	- (1)(5)(6)
10		%r8, (%rdi,%rax)		9 (17)(18)	(15) (16)	(9)(10)(11)(12)(13) (14)		(4)	(8)		(12)	. ,	(2) (5)(6)(7)
11	-	%rsi, (%rcx,%rax)		10 (19)(20)	(17) (18)	(10)(11)(12)(13)(14) (15)(16)	(9)	(3)	(4)	(8)	(13)		(5)(6)(7)(12)
12	-	\$8, %rax	D6	11	(19) (20)	(10)(12)(13)(15)(16) (17)(18)	(11)	(9)	(3)	(4)		(14)	(5)(6)(7)(8)(12)(13)
_	•	ΨΟ, /01 d X	P6	12		(12)(13)(15)(16)(17) (18)(19)(20)			(9)				(4)(5)(6)(7)(8)(12)(13)(14)
13	•	%r9, %rax		13		(12)(13)(16)(17)(18) (20)	(15)	(10)	(11)	(9)	(19)		(3)(4)(5)(6)(7)(6)(12)(13)(14)
14	jne	.L9											
_	_	(%rdi,%rax), %rsi											
16	movq	(%rcx,%rax), %r8											
	_	%r8, (%rdi,%rax)											
18	movq	%rsi, (%rcx,%rax)											

52

② cmpq %r9, %rax ② jne .L9

19 addq \$8, %rax

2-issue SS + Register renaming + OoO
Only "2" of them can have a instruction at the same cycle

		Z 1334C 00 1		
1	movq	(%rdi,%rax), %rsi → P1		
2	movq	(%rcx,%rax), %r8 → P2		IF
3		%r8, (%rdi,%rax)	1	(1)(2
4	movq	%rsi, (%rcx,%rax)	2	(3)(4
⑤	addq	\$8, %rax → P3	3	(5)(6
6	cmpq	%r9, %rax	4 5	(7)(8 (9)(10
7	jne	.L9	6	(11) (1
8	movq	(%rdi,%rax), %rsi → P4	7	(13)(1
9		(%rcx,%rax), %r8 → P5	8	(15)(1
10	movq	%r8, (%rdi,%rax)	9	(17)(1
11	movq	%rsi, (%rcx,%rax)		(19)(2
12	addq	\$8, %rax → P6	11 12	
13	cmpq	%r9, %rax	13	
14	jne	.L9	14	
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		

				1	•						7
	IF	ID	REN	M ₁	M2	М3	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)(0)(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
Only "2" of them can have a instruction at the same cycle

		Z 1334C CC		
1	movq	(%rdi,%rax), %rsi →	P1	
2			P2	
3	movq	%r8, (%rdi,%rax)		1
4	movq	%rsi, (%rcx,%rax)		2
5	addq	\$8, %rax →	P3	3
6	cmpq	%r9, %rax		4 5
7	jne	.L9		6
8	movq	(%rdi,%rax), %rsi →	P4	7
9	movq	(%rcx,%rax), %r8 →	P5	8
10	movq	%r8, (%rdi,%rax)		9
11	movq	%rsi, (%rcx,%rax)		10 11
12	addq	\$8, %rax →	P6	12
13	cmpq	%r9, %rax		13
14	jne	.L9		14
15	movq	(%rdi,%rax), %rsi		15
16	movq	(%rcx,%rax), %r8		
17	movq	%r8, (%rdi,%rax)		
18	movq	%rsi, (%rcx,%rax)		
19	addq	\$8, %rax		
20	cmpq	%r9, %rax		

•				, – 0.	1	,					4	7
2		IF	ID	REN	M1	M2	М3	M4	ALU	MUL	BR	ROB
	1	(1)(2)										
	2	(3)(4)	(1)(2)									
3	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)			
H	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)(0)(7)(8)(12)(13)(14)
	14			(12)(13)(17)(18)	(16)	(15)	(10)	(11)	(20)			(3) (12)(13)(14)(19)
	15					(16)	(15)	(10)			(21)	(11)(12)(13)(14)(19)(20)

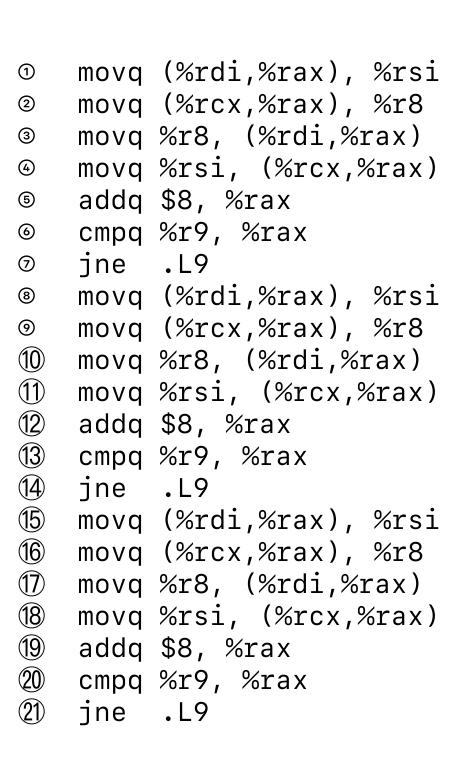
2-issue SS + Register renaming +0o0 e same cycle

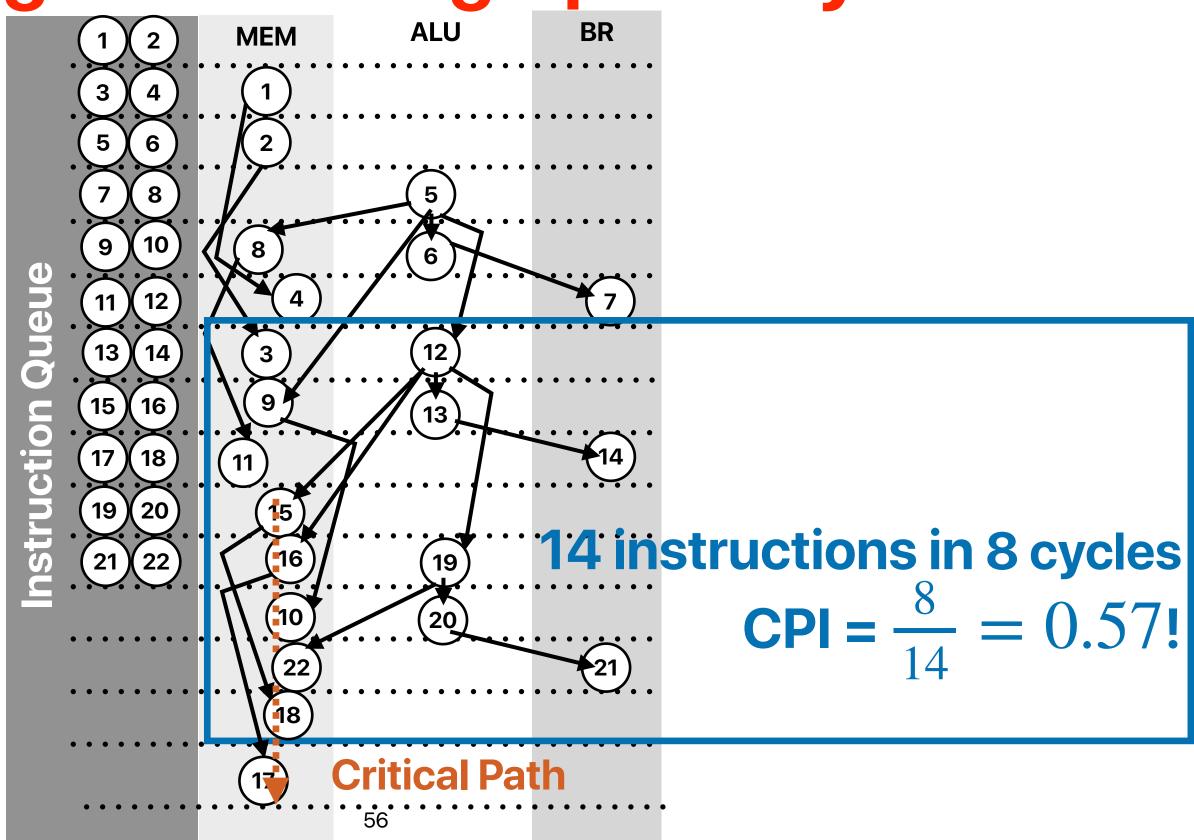
		Z-135UC 3				y iş		the						T \	at the same cycle
1	movq	(%rdi,%rax), %rsi	\rightarrow P	1			Jilly Z Oi		·	all	IIdV		Struc		Tat the Same Cycle
2	movq	(%rcx,%rax), %r8	\rightarrow P	2	IF	ID	REN	M1	M2	M3	M4	ALU	MUL	BR	ROB
3	movq	%r8, (%rdi,%rax)			1 (1)(2)										
4	movq	%rsi, (%rcx,%rax)			2 (3)(4)	(1)(2)									
(5)	addq	\$8, %rax	\rightarrow P	3	3 (5)(6)	(3)(4)	(1)(2)	(4)							
6	•	%r9, %rax			4 (/)(8) 5 (9)(10)	(5)(6)	(2)(3)(4) (3)(4)(5)(6)	(1)	(1)						
7	jne	.L9			5 (9)(10) 6 (11)(12)	(7)(8) (9)(10)	(3)(4)(6)(7)(8)	(2)	(1)	(1)		(5)			
8		(%rdi,%rax), %rsi	\rightarrow P		7 (13)(14)	(11)(12)	(3)(4)(7)(9)(10)	(8)	(2)	(2)	(1)	(6)			(5)
9	-	(%rcx,%rax), %r8		5	8 (15)(16)	(13)(14)	(3)(9)(10)(11)(12)		(8)	\- /	(2)	(0)		(7)	-(1)(5)(6)
10		%r8, (%rdi,%rax)			9 (17)(18)	(15) (16)	(9)(10)(11)(12)(13) (14)		(4)	(8)		(12)			(2) (5)(6)(7)
11	•	%rsi, (%rcx,%rax)		•	(19)(20)	(17) (18)	(10)(11)(12)(13)(14) (15)(16)	(9)	(3)	(4)	(8)	(13)			(5)(6)(7)(12)
12	•	\$8, %rax	→ P	6	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)			(9)					(4)(5)(6)(7)(8)(12)(13)(14)
13	•	%r9, %rax		1	13		(12)(13)(16)(17)(18) (20)	(15)	(10)	(11)	(9)	(19)			(3)(4)(5)(6)(7)(8)(12)(13)(14)
14)	jne	.L9		1	14		(12)(13)(17)(18)	(16)	(15)	(10)	(11)	(20)			-(3) (12)(13)(14)(19)
15	movq	(%rdi,%rax), %rsi		•	15				(16)	(15)	(10)			(21)	(11)(12)(13)(14)(19)(20)
16	mova	(%rcx,%rax), %r8		•	16					(16)	(15)				(10)(11)(12)(13)(14)(19)(20)(21)
_	•	%r8, (%rdi,%rax)													
	•	%rsi, (%rcx,%rax)													
$\overline{}$	•	\$8, %rax													
20	•	%r9, %rax													

jne

.L9

Through data flow graph analysis

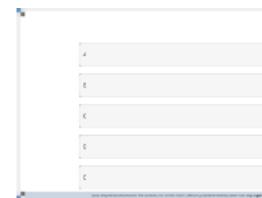






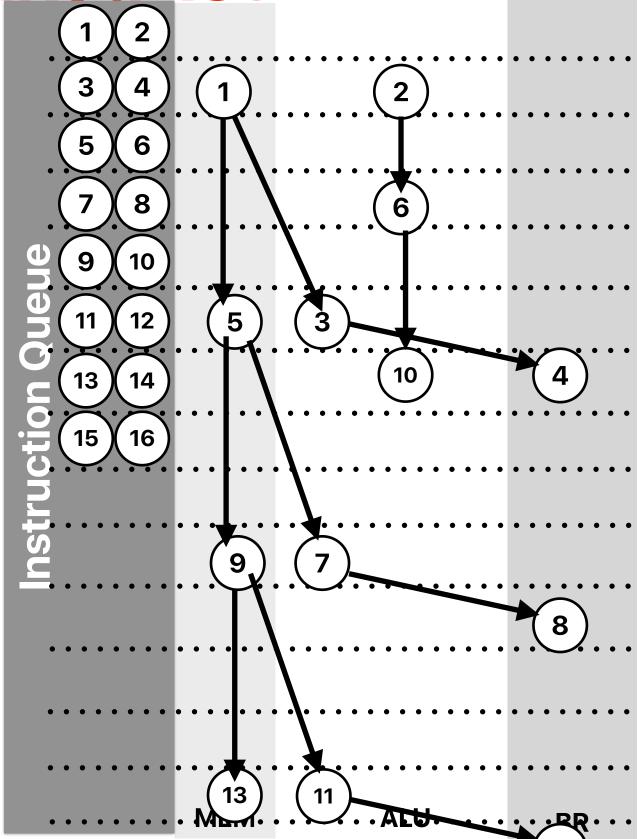
Assume the current PC is already at instruction (1) and this linked list
has only three nodes. This processor can fetch and issue 2
instructions per cycle, with exactly the same register renaming
hardware and pipeline as we showed previously.
Which of the following C state of the
code snippet determines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

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



Dvnamic instructions

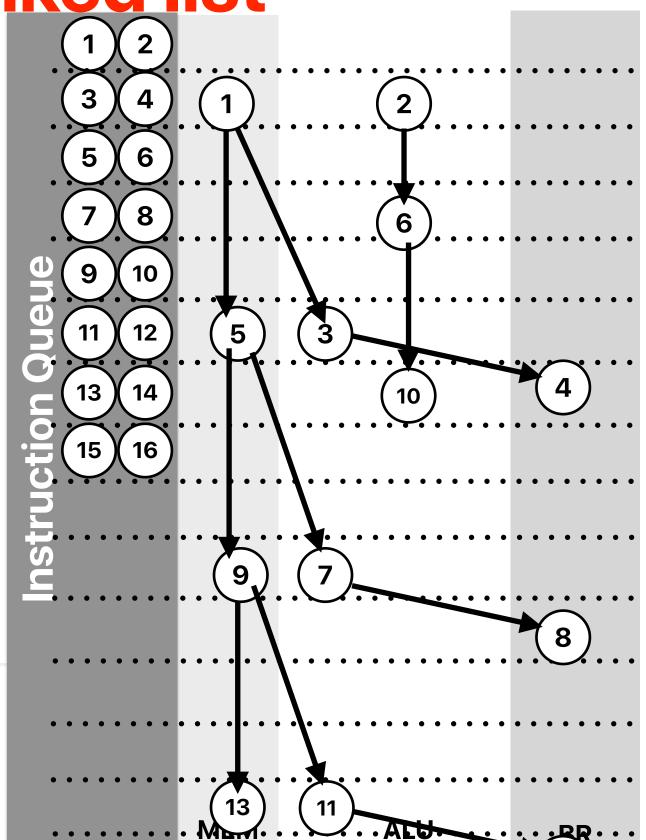
		ymanic	II 13ti uctio	71 IO
① .L	3:	movq	8(%rdi),	%rdi
2		addl	\$1, %eax	
3		testq	%rdi, %rd	di
4		jne	.L3	
⑤ .L	3:	movq	8(%rdi),	%rdi
6		addl	\$1, %eax	
7		testq	%rdi, %rd	di
8		jne	.L3	
9 .L	3:	movq	8(%rdi),	%rdi
10		addl	\$1, %eax	
11)		testq	%rdi, %rd	di
12		jne	.L3	
13 .L	3:	movq	8(%rdi),	%rdi
14		addl	\$1, %eax	
15)		testq	%rdi, %rd	di
16)		jne	.L3	61





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
```



Performance determined by the critical path

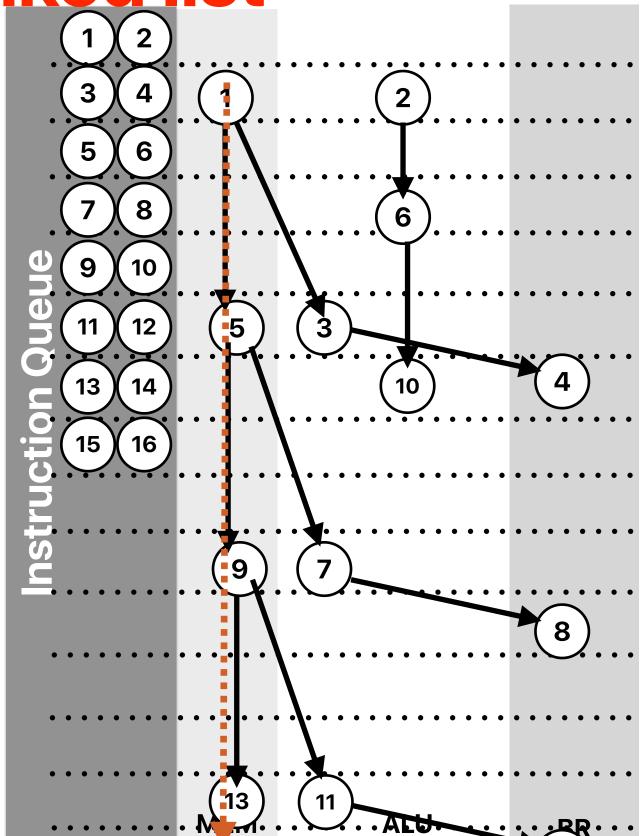
4 cycles each iteration

4 instructions per iteration

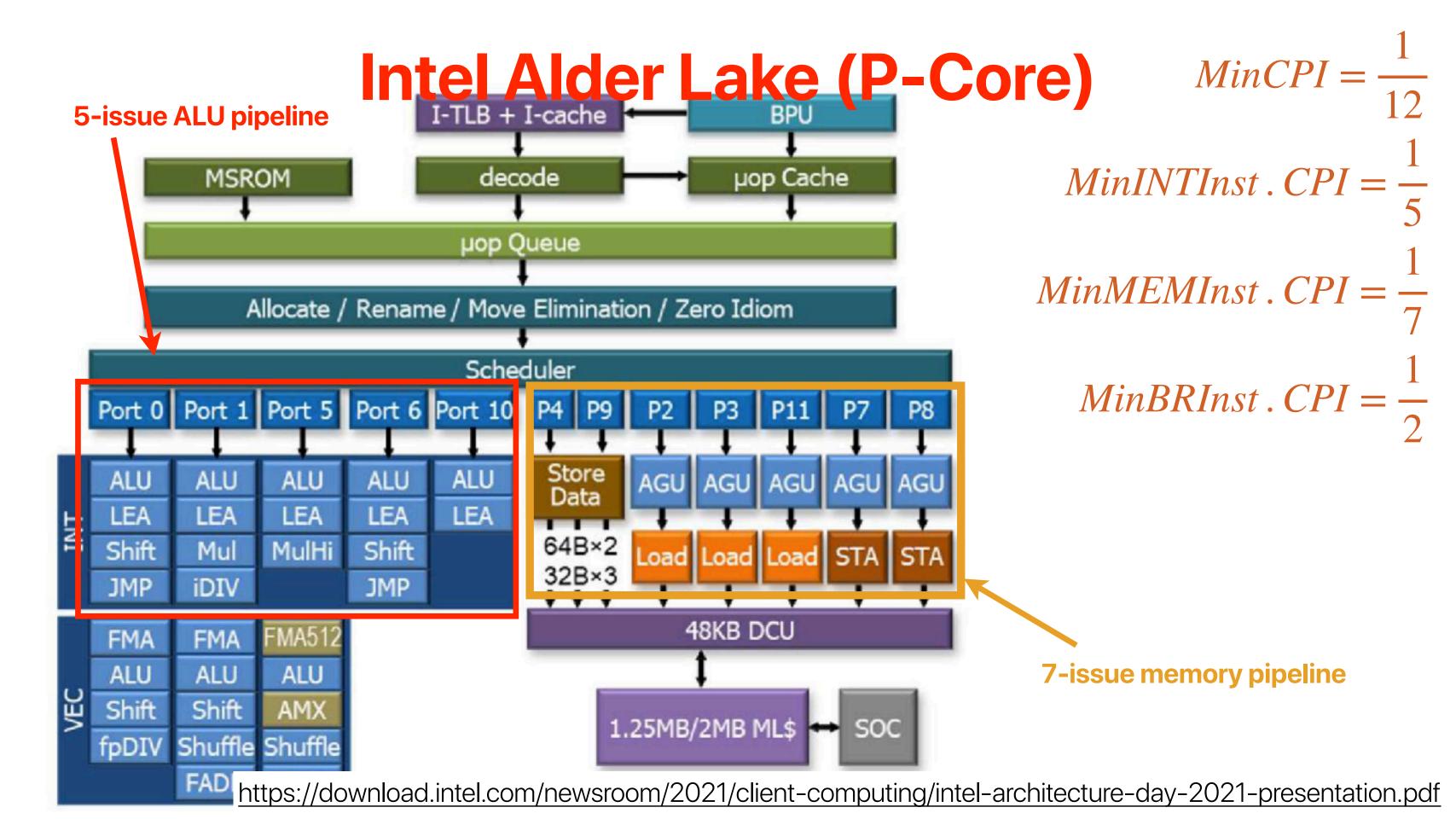
$$CPI = \frac{4}{4} = 1$$

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

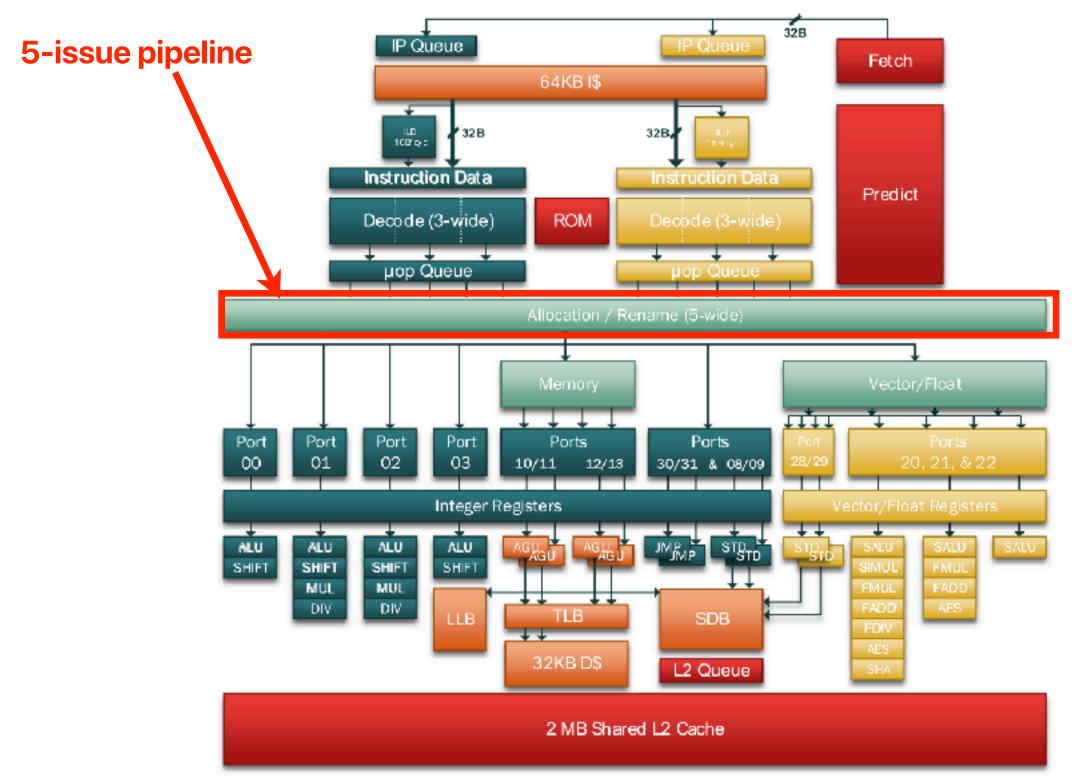
① .L3:    movq    8(%rdi), %rdi
②    addl    $1, %eax
③    testq    %rdi, %rdi
④    jne    .L3
```



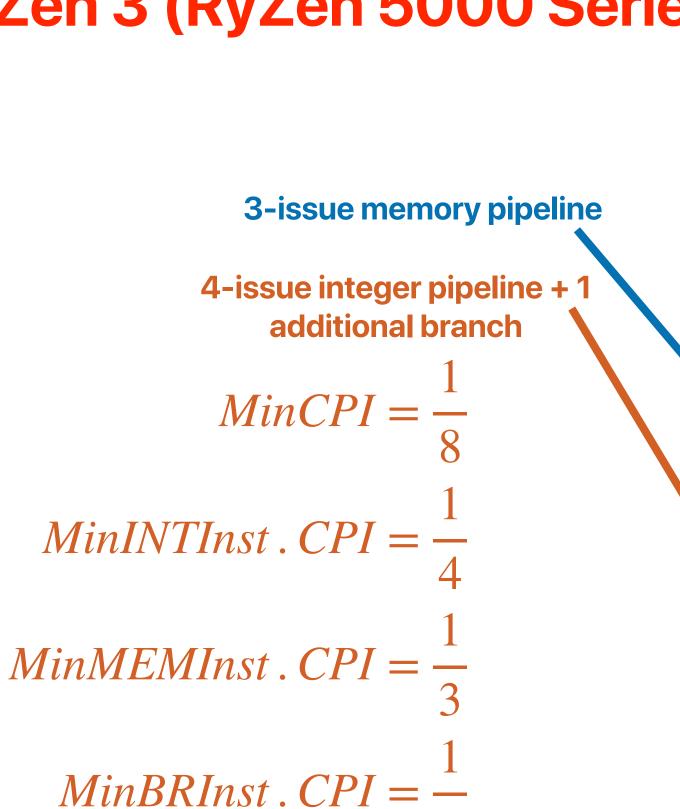
The pipelines of Modern Processors

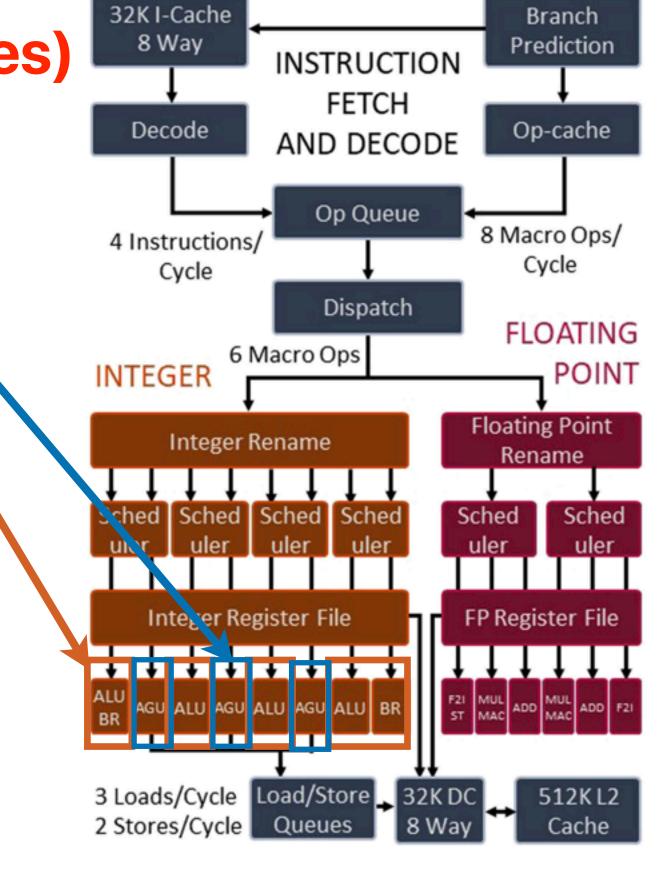


Intel Alder Lake (E-Core)



AMD Zen 3 (RyZen 5000 Series)





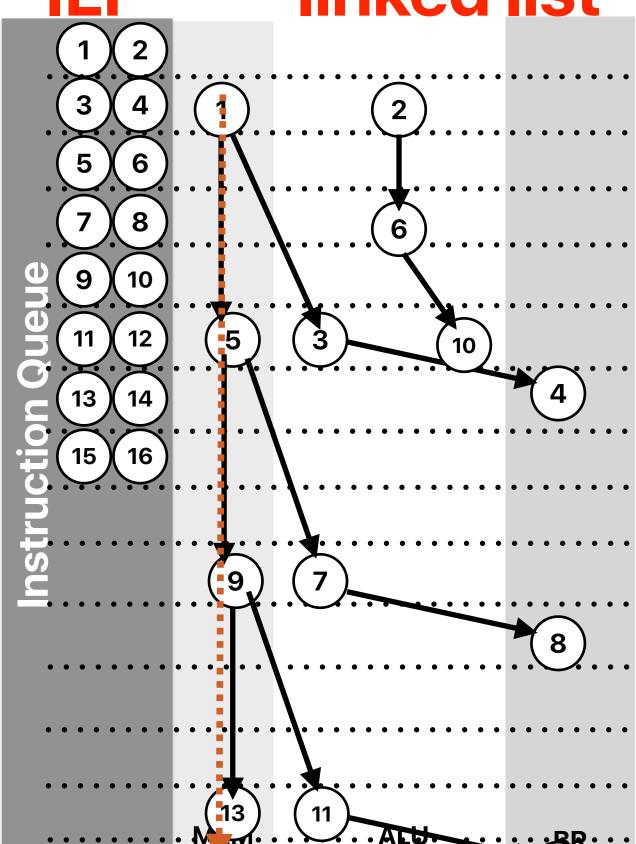
What if we have "unlimited" ILP — "linked list"

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

Ob1110100001011111

```
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;
```



Five implementations

Which of the following implementations will perform the best on modern

return c;

pipeline processors?

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

```
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;
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 \gg 4;
```

```
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 \gg 4;
     return c;
```

Five implementations

Which of the following implementations will perform the best on modern pipeline

processors?

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

```
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;
    c += x & 1;
    x = x >> 1;
    c += x & 1;
    x = x >> 1;
}
return c;
}
```

```
line 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;
}
```

```
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-todate 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

203

