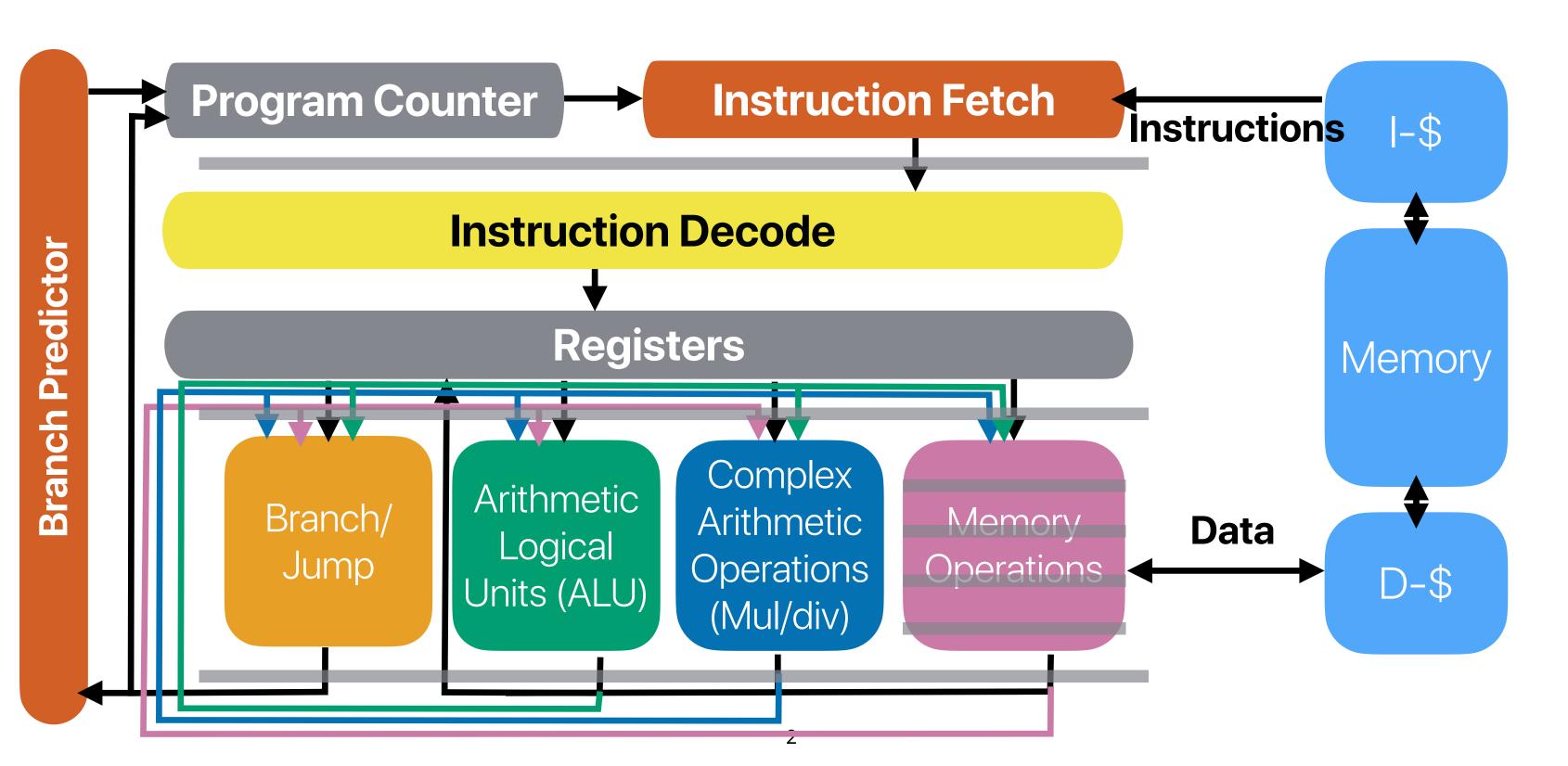
## Modern Processor Design (IV): Try everything

Hung-Wei Tseng

#### Recap: Data "forwarding"



#### Recap: 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:
            (%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/	AG M1	M2	М3	M4/XORL	WB/Retire
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	(4)	(3)					(2)	(1)
9	(5)	(4)	(3)					(2)
10	(6)	(5)	(4)	(3)				
11	<b>(7)</b>	(6)	(5)	(4)	(3)			
12	(8)	(7)	(6)	<b>7</b>	(4)	(3)		
13	(9)	(8)	(7)	<b>V</b>		(4)	(3)	
14	(10)	(9)	(8)				(4)	(3)
15	(11)	(10)	(9)	(8)				(4)
16	(11)	(10)		(9)	(8)			(5)
17	(11)	(10)	11	1 cycles	<b>for 7</b>	(8)		(6)
18		(10)				(9)	(8)	(7)
19		(10)		instructi	ons		(9)	(8)
20	(12)		(10)	CPI = 1	57			(9)
21		(12)	(11)	(10)				
22	(14)	(13)	(12)		(10)			
23		(14)	(13)		(11)	(10)		
<b>24</b> 3			<b>(14)</b>	(13)	(12)	(11)	(10)	
3								

```
ID ALU/BR/AG M1 M2 M3 M4/XORL WB/Retire
                                                                   (1)
       int64_t temp = a[i];
                                                                                           9 cycles for 7
                                                                   (2)
                                                                      (1)
       a[i] = b[i];
                                                                   (3)
                                                                             (1)
                                                                                            instructions
                                                                   (4)
                                                                             (2)
                                                                                   (1)
       b[i] = temp;
                                                                                             CPI = 1.29
                                                                   (5)
                                                                             (3)
                                                                                   (2)
                                                                                       (1)
                                                                   (5)
                                                                                        (2)
                                                                                           (1)
                                                                   (5)
                                                                                           (2)
                                                                                                 (1)
                                                                                                 (2)
                                                                                                        (1)
                                                                   (6)
                                                                             (4)
                                                                                                        (2)
                                                                                   (4)
                                                                             (5)
                                .L9:
                                                                   (8)
                                                                                                        (3)
                                                                             (6)
                                                                                   (5)
                                                                                       (4)
         (%rdi,%rax), %rsi
                                            (%rcx,%rax), %r8
movq
                                   movq
                                                                  (9)
                                                                                           (4)
                                                                             (7)
                                                                                        (5)
                                            (%rdi,%rax), %rsi
         (%rcx,%rax), %r8
movq
                                   mova
                                                                12 (10)
                                                                             (8)
                                                                                           (5)
                                                                                                 (4)
                                            $8, %rax
         %r8, (%rdi,%rax)
                                   addq
                                                                13 (11)
                                                                             (9)
                                                                                                        (4)
movq
                                                                      (10)
                                                                                   (8)
                                                                                                 (5)
                                            %r8, -8(%rdi,%rax)
                                                               14 (11)
                                                                                                        (5)
         %rsi, (%rcx,%rax)
                                   mova
                                                                      (10)
                                                                            (10)
                                                                                   (9)
                                                                                       (8)
movq
                                            %rsi, -8(%rcx,%rax 15 (11)
                                   mova
                                                                                                        (6)
                                                                                       (9)
                                                                                           (8)
                                            %r9, %rax
                                                                16 (11)
                                                                                           (9)
                                                                                                        (7)
                                   cmpq
                                                                      (10)
                                                                                                 (8)
         %r9, %rax
cmpq
                                            .L9
                                                                17 (12)
                                   jne
                                                                            (11)
                                                                                                        (8)
                                                                                                 (9)
         .L9
jne
                                            (%rcx,%rax), %r8
                                                                                                        (9)
                               8
                                   mova
                                                                18 (12)
                                                                            (12)
                                                                                   (11)
         (%rdi,%rax), %rsi
movq
                                            (%rdi,%rax), %rsi
                                                                19 (13) (12)
                                                                                       (11)
                               9
                                                                            (13)
                                                                                   (12)
                                                                                                        (10)
                                   mova
         (%rcx,%rax), %r8
mova
                                            $8, %rax
                                                               20 (14) (13)
                                   addq
                                                                                       (12) (11)
                                                                            (14)
         %r8, (%rdi,%rax)
movq
                               (11)
                                            %r8, -8(%rdi,%rax)
                                                                                                 (11)
                                                                      (14)
                                                                                           (12)
                                   movq
         %rsi, (%rcx,%rax)
movq
                               (12)
                                                                                                        (11)
                                            %rsi, -8(%rcx,%rax 22
                                                                                                 (12)
                                   movq
addq
         $8, %rax
                                                                                                        (12)
                                                                23
                                            %r9, %rax
                                   cmpq
                                                               24
                                                                                                        (13)
         %r9, %rax
cmpq
                                            .L9
                                   ine
                                                               25
                                                                                                        (14)
         .L9
jne
```

```
for(i = 0; i < count; i++) {</pre>
                                                                               M1
                                                                                    M2
                                                                                          M3
                                                                                                M4/XORL WB/Retire
       int64_t temp = a[i];
       a[i] = b[i];
                                                                         (1)
                = temp;
                        The compiler can only do this when it's 100% for sure count is
                                  always an even number! — loop unrolling
              Compilers are limited by the number of registers available to the software!
         (%rcx,%rax), %r8
movq
                                                                        (5)
                                                                               (4)
                                                                                                        (2)
                                         (%rcx,%rax), %r8
                                mova
         (%rdi,%rax), %rsi
movq
                                         (%rdi,%rax), %rsi 10 (8)
                                                                                                        (3)
                                                                        (6)
                                                                               (5)
                                                                                    (4)
                                mova
addq
        $8, %rax
                                                                        (7)
                                                                               (6)
                                                                                    (5)7 cycles for 7
                                                            11 (9)
                                         $8, %rax
                                 addq
        %r8, -8(%rdi,%rax)
movq
                                         %r8, -8(%rdi,%rax)12(10)(9)
                                                                        (8)
                                                                               (7)
        %rsi, -8(%rcx,%rax)<sub>⑤</sub>
                                mova
movq
                                                                        (9)
                                         %rsi, -8(%rcx,%rax13 (11) (10)
                                                                                                        (4)
                                movq
        %r9, %rax
cmpq
                                                            14 (12) (11)
                                                                        (10)
                                         (%rcx,%rax), %r8
                            6
                                                                                                        (5)
                                mova
jne
         .L9
                                         (%rdi,%rax), %rsi 15 (13) (12)
                                                                        (11)
                                                                               (10)
                                                                                                 (7)
                                                                                                        (6)
                             7
                                mova
        (%rcx,%rax), %r
movq
                                         %r9, %rax
                                                            16 (14) (13)
                                                                        (12)
                                                                               (11)
                                                                                   (10)
                                                                                                        (7)
                                cmpq
        (%rdi,%rax), %rsi
movq
                                         .L9
                                 jne
                                                                        (13)
                                                                               (12)
                                                                                    (11)
                                                                                          (10)
                                                                                                        (8)
                                                                  (14)
        $8, %rax
addq
                                         $8, %rax
                                 addq
                                                                                                        (9)
                                                                        (14)
                                                                               (13)
                                                                                   (12)
                                                                                          (11)
                                                                                                 (10)
        %r8, -8(%rdi,%rax)
movq
                                         %r8, -8(%rdi,%rax)<sub>19</sub>
                                mova
                                                                               (14)
                                                                                   (13)
                                                                                          (12)
                                                                                                 (11)
                                                                                                        (10)
        %rsi, -8(%rcx,%rax
movq
                                         %rsi, -8(%rcx,%rax20
                                movq
                                                                                                        (11)
                                                                                    (14)
                                                                                          (13)
                                                                                                 (12)
        %r9, %rax
cmpq
                                         %19, %1ax
                                 CIIIPY
                                                            21
                                                                                                 (13)
                                                                                                        (12)
                                                                                          (14)
         .L9
jne
                                         .L9
                                 jne
                                                            22
                                                                                                       (13)
                                                                                                 (14)
                                                      5
                                                                                                       (14)
```

#### Takeaways: data hazards

- More data dependencies, more likelihood of data hazards
- Stalls and data forwarding can both address data hazards to generate correct code execution results — but not very efficient
- Compiler optimizations can help, but to a limited extent

Missing opportunities

			AIIƏŻ		J, Uh					IUC	<b>3</b>				
for	(i = 0; i < c	ou	nt; i	++)	_{	_		IF	ID	ALU/BR/AG	M1	M2	МЗ	M4/XORL	WB/Retire
	int64_t_temp	) =	a[i]	•			1	(1)						,	
	oria - Dro			Or		2 D	VC		Ji.	ot v	MA	at			
	a[i] = Pro		<b>C</b> 33	<b>UI</b>	Gal		3		(2,		VII	al			
	b[i] = temp;					_	4	<b>(4)</b>	(3)	(2)	(1)	_			
}	b[i] = temp; Shot		d ha	ani	OPT	1 21	7		4]	ngo	(2)				
J	3110		M I I C		<b>3</b> 61			(3)	C,			(2)	(1)		
													(2)	(1)	
· move	(9/360) 9/360) 9/360				nvr	an	$\mathbf{n}$	C		147				(2)	(1)
movq	(%rcx,%rax), %r8 (%rdi,%rax), %rsi	1	movq							/ 6 >		(4)			(2)
addq	\$8, %rax	2	movq		(,%rax),	, %rsi	10	(8)	(/)	(6) (7)	(5) (C)	(4)		o for	(3)
movq	%r8, <b>-8</b> (%rdi,%rax)	3	addq	\$8, %				(9) (10)		(7) (8)	(6)			es for 7	
movq	%rsi, -8(%rcx,%rax	) (4)	movq		-8(%rdi					(8) (9)	(7)	(7)	nstru	ctions	(4)
cmpq	%r9, %rax	6	movq		// // // // // // // // // // // // //					(10)		( / )		(5)   = (6)	(5)
jne	.L9	7	movq		[,%rax),	•				(11)	(10)		<b>U</b> P	(7)	(6)
movq	(%rcx,%rax), %r	8	cmpq		%rax	•		(14)		(12)		(10)		ζ- ,	<b>(7)</b>
movq	(%rdi,%rax), %rsi	9	jne	.L9			17		(14)	(13)	(12)	(11)	(10)		(8)
addq movq	\$8, %rax %r8, -8(%rdi,%rax)	10	addq	\$8, %			18			(14)	(13)	(12)	(11)	(10)	(9)
movq	%rsi, <b>-8</b> (%rcx,%rax		movq		<b>-8</b> (%rdi						(14)	(13)	(12)	(11)	(10)
cmpq	%r9, %rax		movq	•	<b>-8</b> (%rd	cx,%ra>	20					(14)	(13)	(12)	(11)
jne	.L9	(13) (14)	ino ino	%19,	∕01 a X		21						(14)	(13)	(12)
		(14)	jne	.L9		_	22							(14)	(13)
						9	23							4	<b>14</b> )

### Dynamic instruction scheduling/ Out-of-order (OoO) execution

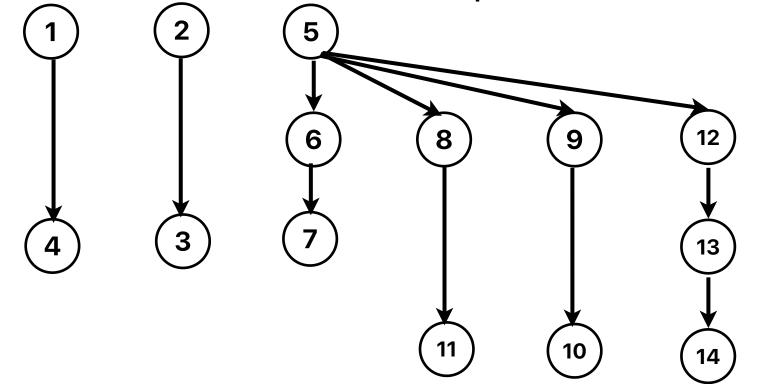
#### What do you need to execution an instruction?

- Whenever the instruction is decoded put decoded instruction somewhere
- Whenever the inputs are ready all data dependencies are resolved
- Whenever the target functional unit is available

#### Scheduling instructions: based on data dependencies

• Draw the data dependency graph, put an arrow if an instruction depends on the other.

```
(%rdi,%rax), %rsi
  movq
          (%rcx,%rax), %r8
  movq
          %r8, (%rdi,%rax)
  movq
          %rsi, (%rcx,%rax)
  movq
  addq
          $8, %rax
          %r9, %rax
  cmpq
  jne
           .L9
          (%rdi,%rax), %rsi
  movq
          (%rcx,%rax), %r8
  movq
          %r8, (%rdi,%rax)
10 movq
① movq
          %rsi, (%rcx,%rax)
          $8, %rax
12 addq
13 cmpq
          %r9, %rax
14 jne
           .L9
```



- In theory, instructions without dependencies can be executed in parallel or out-of-order
- Instructions with dependencies (on the same path) can never be reordered



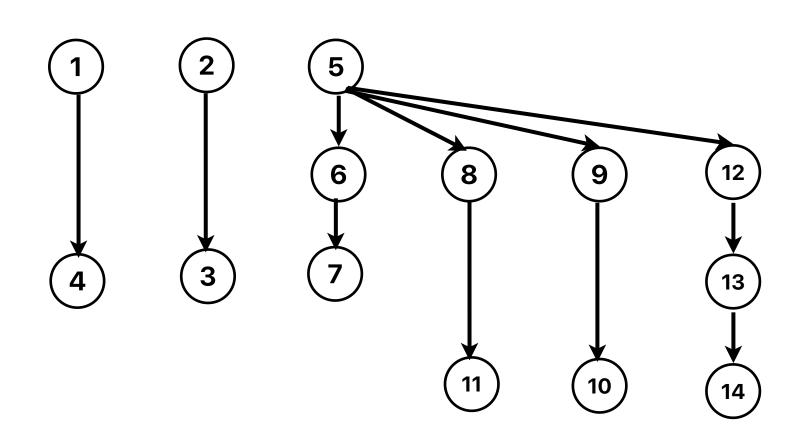
#### If we can predict the future ...

Consider the following dynamic instructions:

```
(%rdi,%rax), %rsi
① 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

10 movq
          %r8, (%rdi,%rax)
          %rsi, (%rcx,%rax)
11) movq
12 addq
          $8, %rax
13 cmpq
          %r9, %rax
14 jne
           .L9
```



Which of the following pair can we reorder without affecting the correctness if the **branch prediction is perfect**?

- A. (1) and (2)
- B. (3) and (5)
- C. (4) and (6)
- D. (6) and (8)
- E. (3) and (8)



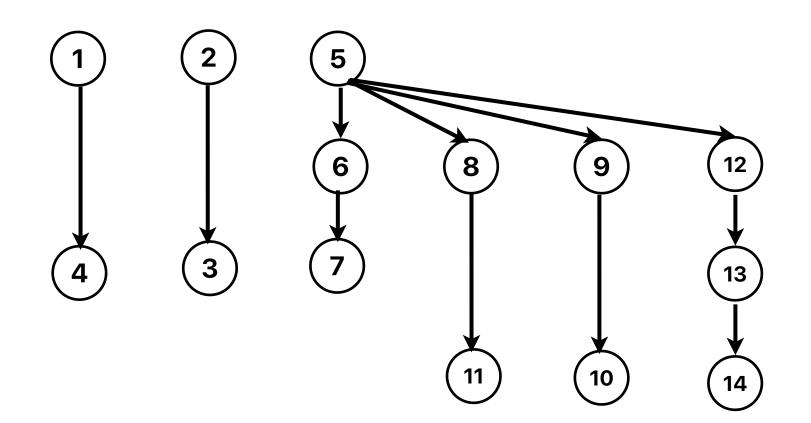
#### If we can predict the future ...

Consider the following dynamic instructions:

```
(%rdi,%rax), %rsi
① movq
           (%rcx,%rax), %r8
② movq
          %r8, (%rdi,%rax)
3 movq
          %rsi, (%rcx,%rax)
@ movq
          $8, %rax
 addq
          %r9, %rax
© cmpq
⑦ jne
           .L9
          (%rdi,%rax), %rsi
® movq
          (%rcx,%rax), %r8

  movq

10 movq
          %r8, (%rdi,%rax)
          %rsi, (%rcx,%rax)
11) movq
12 addq
          $8, %rax
13 cmpq
          %r9, %rax
14 jne
           .L9
```



Which of the following pair can we reorder without affecting the correctness if the **branch prediction is perfect**?

- A. (1) and (2)
- B. (3) and (5)
- C. (4) and (6)
- D. (6) and (8)
- E. (3) and (8)

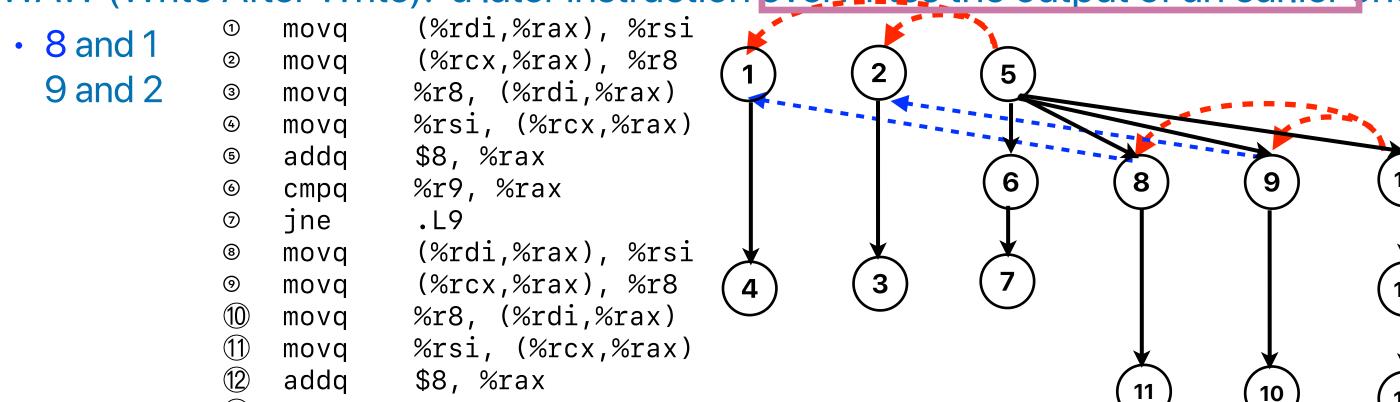
#### False dependencies

- We are still limited by false dependencies
- They are not "true" dependencies because they don't have an arrow in data dependency graph
  - WAR (Write After Read): a later instruction powerwrites the source of an earlier one
    - 5 and 1, 5 and 2, 12 and 8, 12 and 9

cmpq

ine

WAW (Write After Write): a later instruction overwrites the output of an earlier one



18

%r9, %rax

.L9

#### False dependencies

We are still limited by false dependencies

.L9

- They are not "true" dependencies because they don't have an arrow in data dependency graph
  - WAR (Write After Read): a later instruction powerwrites the source of an earlier one
    - 5 and 1, 5 and 2, 12 and 8, 12 and 9

ine

 WAW (Write After Write): a later instruction overwrites the output of an earlier one (%rdi,%rax), %rsi mova • 8 and 1 (%rcx,%rax), %r8 mova cmpq jne movq 7 3 (%rcx,%rax), %r8 movq 13 %r8, (%rdi,%rax) movq %rsi, (%rcx,%rax) movq \$8, %rax addq 11 %r9, %rax cmpq 19

#### Takeaways: data hazards

- More data dependencies, more likelihood of data hazards
- Stalls and data forwarding can both address data hazards to generate correct code execution results — but not very efficient
- Compiler optimizations can help, but to a limited extent
- False dependencies limits the freedom of out-of-order execution

#### What if we can use more registers...

```
(%rdi,%rax), %t0
        (%rdi,%rax), %rsi
movq
                                         movq
        (%rcx,%rax), %r8
                                                  (%rcx,%rax), %t1
                                         movq
movq
        %r8, (%rdi,%rax)
                                                  %t1, (%rdi,%rax)
                                         movq
movq
                                                  %t0, (%rcx,%rax)
        %rsi, (%rcx,%rax)
movq
                                         movq
        $8, %rax
                                                  $8, %rax, %t2
addq
                                         addq
                                                  %r9, %t2
        %r9, %rax
                                         cmpq
cmpq
         .L9
                                                  .L9
jne
                                         jne
        (%rdi,%rax), %rsi
                                                  (%rdi, %t2), %t3
movq
                                         movq
        (%rcx,%rax), %r8
                                                  (%rcx, %t2), %t4
movq
                                         movq
        %r8, (%rdi,%rax)
                                                  %t4, (%rdi,%t2)
                                         movq
movq
        %rsi, (%rcx,%rax)
                                                  %t3, (%rcx, %t2)
                                         movq
movq
        $8, %rax
                                         addq
                                                  $8, %t2, %t5
addq
        %r9, %rax
                                                  %r9, %t5
                                         cmpq
cmpq
        .L9
                                         jne
                                                  .L9
jne
```

All false dependencies are gone!!!

# 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

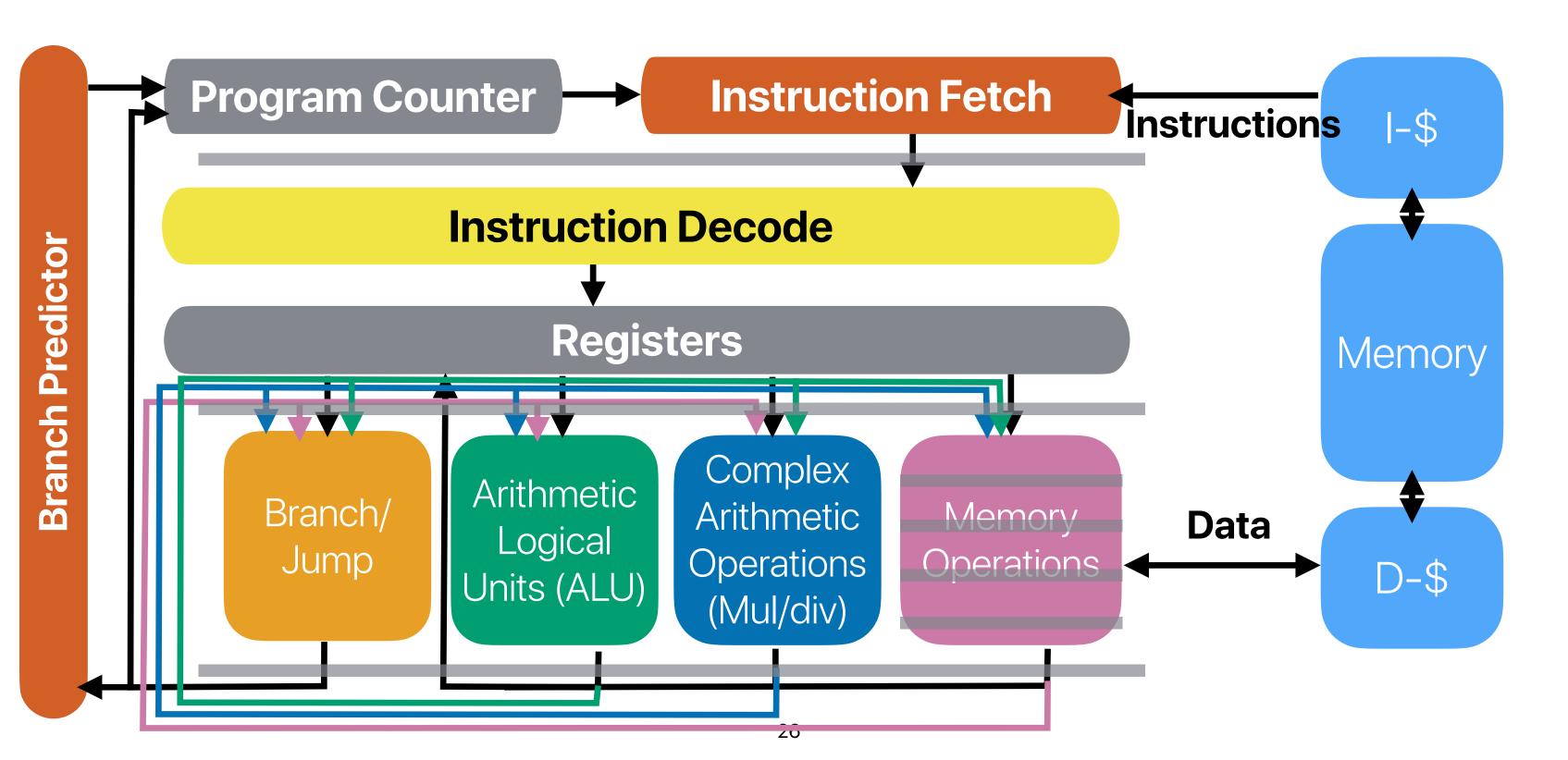
#### Can we really execute instructions OoO?

- 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

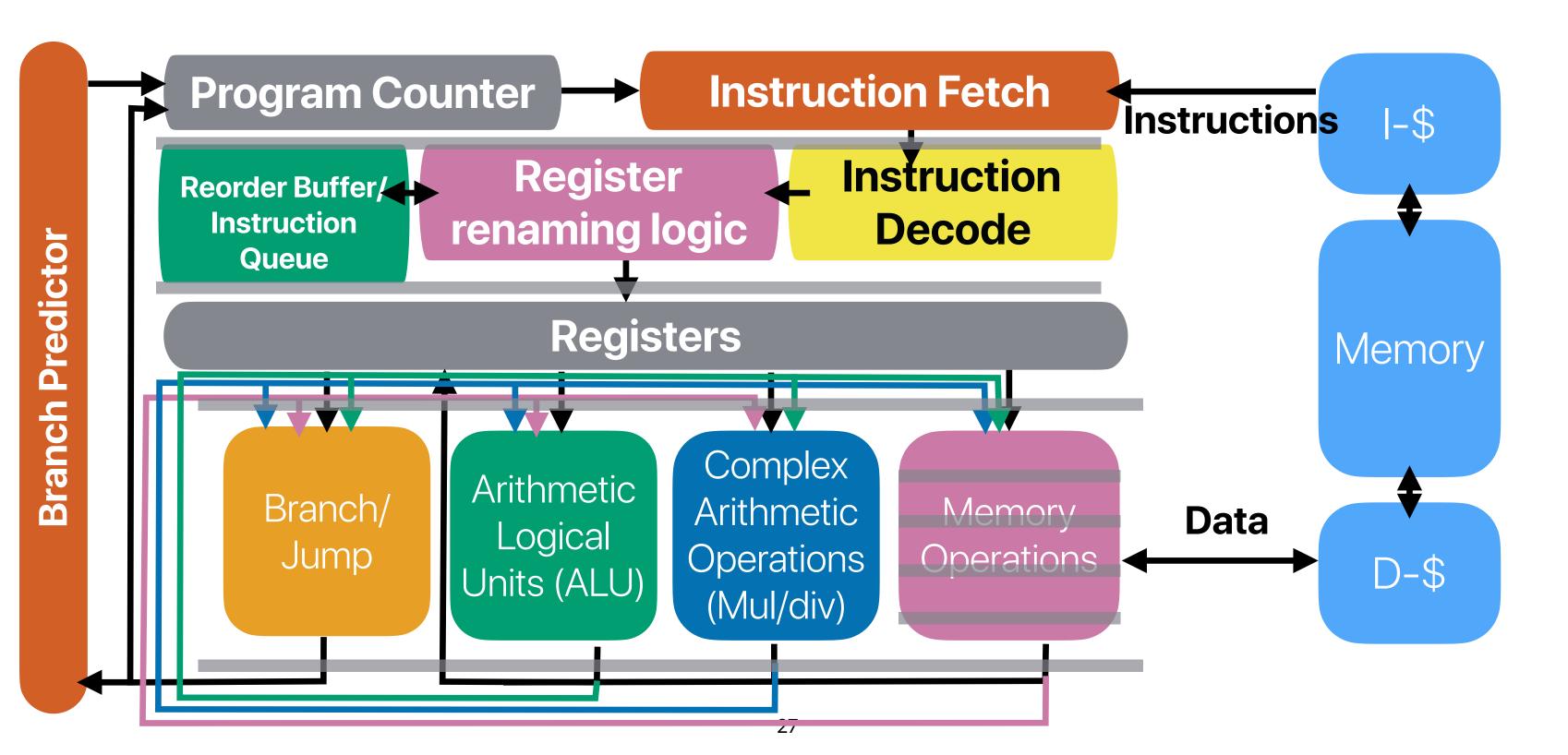
#### **Speculative Execution**

- **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	AG	M1	M2	M3 N	<b>M4</b>	ALU	MUL	BR	ROB
1	(1)											
2	(2)	(1)										
3			(1)									
4			(-)									
5												
6												
7												
7												
8												
9												
10												
11												
12												
13												
14												
15												
16												
10												

	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			

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

- <sup>⊙</sup> 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

	IF	ID	REN	AG	M1	M2	М3	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	)											
1												
12	2											
13												
14												
1	5											
10	3											

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

1	movq	%rsi	., (%rcx,%rax)
12	addq	\$8,	%rax

13 cmpq %r9, %rax

14 jne .L9

	IF	ID	REN	AG	M1	M2	M3 M	14	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	<b>P2</b>

	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 \rightarrow P2
movq %r8 (P1), (%rdi,%rax)
movq %rsi(P2), (%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	AG	M1	M2	M3 I	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
P	1	0		1	P6			
P	2	0		1	P7			
P	3				P8			
P	4				P9			
P!	5				P10			

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

- 12 addq \$8, %rax
- ① cmpq %r9, %rax
- 14 jne .L9

	IF	ID	REN	AG	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	<b>(7)</b>	(6)	(3)(4)(5)			(2)	(1)				
8											
9											
10											
11											
12											
13											
14											
15											

	Physical Register
rax	
rcx	
rdi	
rsi	P1
r8	<b>P2</b>

16

	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 \rightarrow P2
movq %r8 (P1), (%rdi,%rax)
movq %rsi(P2), (%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	AG	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	<b>(7)</b>	(6)	(3)(4)(5)			(2)	(1)					
8												
9												
10												
11												
12												
13												
14												
15												
16												

	Physical Register
rax	Р3
rcx	
rdi	
rsi	P1
r8	<b>P2</b>

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

```
AG M1 M2 M3 M4 ALU
   IF ID
               REN
                                                                     ROB
1 (1)
2 (2) (1)
                                         Instruction (5) is
3 (3) (2)
                (1)
                                      running ahead of (3)
               (2)
                        (1)
4 (4)
      (3)
                (3)
                        (2) (1)
5 (5)
              (3)(4)
                            (2) (1)
  (6)
      (5)
      (6)
             (3)(4)(5)
                                (2) (1)
8 (8) (7)
                                    (2) (1)
                                            (5)
             (3)(4)(6)
10
12
13
14
15
16
```

	Physical Register
rax	Р3
rcx	
rdi	
rsi	P1
r8	<b>P2</b>

	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 (P1), (%rdi,%rax)
movq %rsi(P2), (%rcx,%rax)
                       → P3
addq $8, %rax
cmpq %r9, %rax (P3)
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	AG M1					MUL	BR	ROB
1	(1)			nstru	ICti	on	(4	) is			
2	(2)	(1)		ning	ıak	102	nd c	f (3	\ Ins	struct	tion (5) is
3	(3)	(2)	(1)	9			id C	) (S			
4	(4)	(3)	(2)	(1)					runn	jing a	head of (3
5	(5)	(4)	(3)	(2) (1)							
6	(6)	(5)	(3)(4)	(2)	(1)						
7	<b>(7)</b>	(6)	(3)(4)(5)		(2)	(1)					
8	(8)	<b>(7)</b>	(3)(4)(6)			(2)	(1)	(5)			
9	(9)	(8)	(3)(6)(7)	(4)			(2)				(1)(5)
10											
11											
12											
13											
14											
15											
16											

	Physical Register
rax	Р3
rcx	
rdi	
rsi	P1
r8	<b>P2</b>

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

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

				77								
	IF	ID	REN	AG	M1	M2	М3	M4	ALU	MUL	BR	ROB
1	(1)											
2	(2)	(1)										
3	(3)	(2)	(1)					R	atire	e/Con	omit (	1)
4	(4)	(3)	(2)	(1)						,, 0011	, , , , , , , , , , , , , , , , , , ,	•
5	(5)	(4)	(3)	(2)	(1)							
6	(6)	(5)	(3)(4)		(2)	(1)						
7	<b>(7)</b>	(6)	(3)(4)(5)			(2)	(1)					
8	(8)	<b>(7)</b>	(3)(4)(6)				(2)	(1)	(5)			
9	(9)	(8)	(3)(6)(7)	(4)				(2)				<del>(1)(</del> 5)
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	1		1	P8			
P4				P9			
P5				P10			

```
movq (%rdi,%rax), %rsi → P1
  movq (%rcx,%rax), %r8 \rightarrow P2
  movq %r8 (P1), (%rdi,%rax)
  movq %rsi(P2), (%rcx,%rax)
   addq $8, %rax
   cmpq %r9, %rax (P3)
  jne .L9
  movq (%rdi,%rax), %rsi → P4
  movq (%rcx,%rax), %r8
  movq %r8, (%rdi,%rax)
  movq %rsi, (%rcx,%rax)
   addq $8, %rax
① cmpq %r9, %rax
   jne .L9
```

	IF		ID	REN	AG	M1	M2	М3	M4	ALU	MUL	BR	ROB
1	(1)	)											
2	(2	2)	(1)										
3	(3	()	(2)	(1)									
4	(4	-)	(3)	(2)	(1)								
5	(5	5)	(4)	(3)	(2)	(1)							
6	(6	5)	(5)	(3)(4)		(2)	(1)						
7	(7	<b>'</b> )	(6)	(3)(4)(5)			(2)	(1)					
8	(8	3)	<b>(7)</b>	(3)(4)(6)				(2)	(1)	(5)			
9	(9	)	(8)	(3)(6)(7)	(4)				(2)				<del>(1)(</del> 5)
10	(10	<b>D)</b>	(9)	(6)(7)(8)	(3)	(4)							(2)(5)
11													
12	2												
13	3												
14	1												
15	5												

	Physical Register
rax	Р3
rcx	
rdi	
rsi	P4
r8	<b>P2</b>

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				P10			

(3)(4)(6)

(3)(6)(7)

(6)(7)(8)

(7)(8)(9)

**(4)** 

(3) (4)

(3) (4)

8 (8) (7)

9 (9) (8)

10 (10) (9)

11 (11) (10)

12

13

14

15

16

```
movq (%rdi,%rax), %rsi → P1
movq (%rcx,%rax), %r8 \rightarrow P2
movq %r8 (P1), (%rdi,%rax)
movq %rsi(P2), (%rcx,%rax)
addq $8, %rax
cmpq %r9, %rax (P3)
jne .L9
movq (%rdi,%rax), %rsi → P4
movg (%rcx,%rax), %r8 \rightarrow P5
movq %r8, (%rdi,%rax)
movq %rsi, (%rcx,%rax)
addq $8, %rax
cmpq %r9, %rax
jne .L9
```

					•							
	IF	ID	REN	AG	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)					

(2) (1)

**(2)** 

(5)

(6)

	Physical Register
rax	Р3
rcx	
rdi	
rsi	P4
r8	<b>P5</b>

	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			

(1)(5)

```
movq (%rdi,%rax), %rsi → P1
movq (%rcx,%rax), %r8 \rightarrow P2
movq %r8 (P1), (%rdi,%rax)
movq %rsi(P2), (%rcx,%rax)
addq $8, %rax
cmpq %r9, %rax (P3)
jne .L9
movq (%rdi,%rax), %rsi → P4
movq (%rcx,%rax), %r8 \rightarrow P5
movq %r8 (P4), (%rdi,%rax)
movq %rsi, (%rcx,%rax)
addq $8, %rax
cmpq %r9, %rax
jne .L9
```

	IF	ID	REN	AG	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	<b>(7)</b>	(6)	(3)(4)(5)			(2)	(1)					
8	(8)	<b>(7)</b>	(3)(4)(6)				(2)	(1)	(5)			
9	(9)	(8)	(3)(6)(7)	(4)				(2)				<del>(1)(</del> 5)
10	(10)	(9)	(6)(7)(8)	(3)	(4)							<del>(2)(</del> 5)
11	(11)	(10)	(7)(8)(9)		(3)	<b>(4)</b>			(6)			
12	(12)	(11)	(8)(9)(10)			(3)	<b>(4)</b>				(7)	(5)(6)
13												
14												
15												
16												

	Physical Register
rax	Р3
rcx	
rdi	
rsi	P4
r8	<b>P5</b>

	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			

(8)(9)(10)

(9)(10)(11)

(8)

```
movq (%rdi,%rax), %rsi → P1
movq (%rcx,%rax), %r8 \rightarrow P2
movq %r8 (P1), (%rdi,%rax)
movq %rsi(P2), (%rcx,%rax)
                        → P3
addq $8, %rax
cmpq %r9, %rax (P3)
jne .L9
movq (%rdi,%rax), %rsi → P4
movq (%rcx,%rax), %r8 \rightarrow P5
movq %r8 (P4), (%rdi,%rax)
movq %rsi(P11), (%rcx,%rax)
addq $8, %rax
cmpq %r9, %rax
jne .L9
```

	IF_	ID	REN	AG	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)	<b>(7)</b>	(3)(4)(6)				(2)	(1)	(5)			
9	(9)	(8)	(3)(6)(7)	(4)				(2)				<del>(1)(</del> 5)
10	(10)	(9)	(6)(7)(8)	(3)	<b>(4)</b>							<del>(2)(</del> 5)
11	(11)	(10)	(7)(8)(9)		(3)	<b>(4)</b>			(6)			

(3) (4)

(3) (4)

	Physical Register
rax	P3
rcx	
rdi	
rsi	P4
r8	<b>P5</b>

12 (12) (11)

14

15

16

13 (13) (12)

	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)** 

(5)(6)

(5)(6)(7)

```
movq (%rdi,%rax), %rsi → P1
movq (%rcx,%rax), %r8 \rightarrow P2
movq %r8 (P1), (%rdi,%rax)
movq %rsi(P2), (%rcx,%rax)
addq $8, %rax
cmpq %r9, %rax (P3)
jne .L9
movq (%rdi,%rax), %rsi → P4
movq (%rcx,%rax), %r8 \rightarrow P5
movq %r8 (P4), (%rdi,%rax)
movq %rsi(P11), (%rcx,%rax)
addq $8, %rax → P6
cmpq %r9, %rax
jne .L9
```

IF ID	REN	AG	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)(4)(6)				(2)	(1)	(5)			
9 (9) (8)	(3)(6)(7)	(4)				(2)				<del>(1)(</del> 5)
10 (10) (9)	(6)(7)(8)	(3)	<b>(4)</b>							<del>-(2)(</del> 5)
11 (11) (10)	(7)(8)(9)		(3)	(4)			(6)			
12 (12) (11)	(8)(9)(10)			(3)	<b>(4)</b>				(7)	(5)(6)
13 (13) (12)	(9)(10)(11)	(8)			(3)	(4)				(5)(6)(7)
14 (14) (13)	(10)(11)(12)	(9)	(8)			(3)				(4)(5)(6)(7)
15										

	Physical Register
rax	P6
rcx	
rdi	
rsi	P4
r8	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 (P1), (%rdi,%rax)
movq %rsi(P2), (%rcx,%rax)
addq $8, %rax
cmpq %r9, %rax (P3)
jne .L9
movq (%rdi,%rax), %rsi → P4
movq (%rcx,%rax), %r8 \rightarrow P5
movq %r8 (P4), (%rdi,%rax)
movq %rsi(P11), (%rcx,%rax)
addq $8, %rax → P6
cmpq %r9, %rax (P6)
jne .L9
```

IF	ID	REN	AG	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)	<b>(7)</b>	(3)(4)(6)				(2)	(1)	(5)			
9 (9)	(8)	(3)(6)(7)	(4)				(2)				<del>(1)(</del> 5)
10 (10)	(9)	(6)(7)(8)	(3)	(4)							<del>-(2)(</del> 5)
11 (11)	(10)	(7)(8)(9)		(3)	(4)			(6)			
12 (12)	(11)	(8)(9)(10)			(3)	(4)				<b>(7)</b>	(5)(6)
13 (13)	(12)	(9)(10)(11)	(8)			(3)	(4)				(5)(6)(7)
14 (14)	(13)	(10)(11)(12)	(9)	(8)			(3)				(4)(5)(6)(7)
15 (15)	(14)	(10)(11)(13)		(9)	(8)			(12)			(3)(4)(5)(6)(7)
16											

	Physical Register
rax	P6
rcx	
rdi	
rsi	P4
r8	<b>P5</b>

	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 (P1), (%rdi,%rax)
movq %rsi(P2), (%rcx,%rax)
addq $8, %rax
cmpq %r9, %rax (P3)
jne
     .L9
movq (%rdi,%rax), %rsi → P4
movq (%rcx,%rax), %r8 \rightarrow P5
movq %r8 (P4), (%rdi,%rax)
movq %rsi(P11), (%rcx,%rax)
addq $8, %rax
                        → P6
cmpq %r9, %rax (P6)
jne
    .L9
```

```
AG M1 M2 M3 M4 ALU
    IF ID
                  REN
                                                                                  ROB
1 (1)
2 (2) (1)
                   (1)
        (2)
3 (3)
                   (2)
                              (1)
        (3)
                   (3)
                             (2) (1)
5 (5)
6 (6)
                 (3)(4)
                                  (2) (1)
        (5)
7 (7)
        (6)
                (3)(4)(5)
                                      (2) (1)
8 (8)
                (3)(4)(6)
                                           (2) (1)
        (7)
                                                     (5)
9 (9)
        (8)
                (3)(6)(7)
                             (4)
                                               (2)
                                                                                <del>(1)(</del>5)
                                                                                 (2)(5)
10 (10) (9)
                (6)(7)(8)
                             (3) (4)
11 (11) (10)
                                                     (6)
                (7)(8)(9)
                                  (3) (4)
12 (12) (11)
               (8)(9)(10)
                                      (3) (4)
                                                                      (7)
                                                                                 (5)(6)
13 (13) (12)
               (9)(10)(11)
                                                                                (5)(6)(7)
                                           (3) (4)
                             (8)
14 (14) (13)
              (10)(11)(12)
                                               (3)
                             (9) (8)
                                                                              (4)(5)(6)(7)
15 (15) (14)
              (10)(11)(13)
                                  (9) (8)
                                                    (12)
                                                                             (3)(4)(5)(0)(7)
```

(9) (8)

	Physical Register
rax	P6
rcx	
rdi	
rsi	P4
r8	P5

(10)(11)(14)

16 (16) (15)

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

(12)

(13)

1	movq	(%rdi,%rax), %rsi → <b>P1</b>
2	movq	(%rcx,%rax), %r8 → P2
3	movq	%r8 ( <b>P1</b> ), (%rdi,%rax)
4	movq	%rsi(P2), (%rcx,%rax)
<b>⑤</b>	addq	\$8, %rax → <b>P3</b>
6	cmpq	%r9, %rax (P3)
7	jne	.L9
8	movq	(%rdi,%rax), %rsi → <b>P4</b>
9	movq	(%rcx,%rax), %r8 → P5
10	movq	%r8 (P4), (%rdi,%rax)
11	movq	%rsi(P11), (%rcx,%rax)
12	addq	\$8, %rax → P6
13	cmpq	%r9, %rax (P6)
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	m c	an	na⊽	ea	Instru	uction a	t the	same cycle
	IF	ID	REN	AG	M1	M2	МЗ	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	<b>(7)</b>	(6)	(3)(4)(5)			(2)	(1)					
8	(8)	<b>(7)</b>	(3)(4)(6)				(2)	(1)	(5)			
9	(9)	(8)	(3)(6)(7)	(4)				(2)				<del>-(1)(</del> 5)
10	(10)	(9)	(6)(7)(8)	(3)	(4)							<del>-(2)(</del> 5)
11	(11)	(10)	(7)(8)(9)		(3)	(4)			(6)			
12	(12)	(11)	(8)(9)(10)			(3)	(4)				(7)	(5)(6)
13	(13)	(12)	(9)(10)(11)	(8)			(3)	(4)				(5)(6)(7)
14	(14)	(13)	(10)(11)(12)	(9)	(8)			(3)				(4)(5)(6)(7)
15	(15)	(14)	(10)(11)(13)		(9)	(8)			(12)			<del>(2)(4)(5)(6)(7)</del>
16	(16)	(15)	(10)(11)(14)			(9)	(8)		(13)			(12)
17	(17)	(16)	(10)(11)(15)				(9)	(8)			(14)	(12)(13)
18												
19												
20												
21												
22												
22												

movq	(%rdi,%rax), %rsi → <b>P1</b>
movq	(%rcx,%rax), %r8 → P2
movq	%r8 ( <b>P1</b> ), (%rdi,%rax)
movq	<pre>%rsi(P2), (%rcx,%rax)</pre>
addq	\$8, %rax → <b>P3</b>
cmpq	%r9, %rax (P3)
jne	.L9
movq	(%rdi,%rax), %rsi → <b>P4</b>
movq	(%rcx,%rax), %r8 → P5
movq	%r8 (P4), (%rdi,%rax)
movq	%rsi(P11), (%rcx,%rax)
addq	\$8, %rax → P6
cmpq	%r9, %rax (P6)
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	1.0
	movq movq movq addq cmpq ine movq movq addq cmpq ine movq movq movq addq cmpq

			Only 1 of	the	m c	an	nav	ea	Instru	iction a	t the	same cycle
	IF	ID	REN	AG	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	<b>(7)</b>	(6)	(3)(4)(5)			(2)	(1)					
8	(8)	<b>(7)</b>	(3)(4)(6)				(2)	(1)	(5)			
9	(9)	(8)	(3)(6)(7)	(4)				(2)				<del>-(1)(</del> 5)
10	(10)	(9)	(6)(7)(8)	(3)	<b>(4)</b>							<del>(2)(</del> 5)
11	(11)	(10)	(7)(8)(9)		(3)	(4)			(6)			
12	(12)	(11)	(8)(9)(10)			(3)	(4)				<b>(7)</b>	(5)(6)
13	(13)	(12)	(9)(10)(11)	(8)			(3)	(4)				(5)(6)(7)
14	(14)	(13)	(10)(11)(12)	(9)	(8)			(3)				(4)(5)(6)(7)
15	(15)	(14)	(10)(11)(13)		(9)	(8)			(12)			<del>(2)(4)(5)(6)(7)</del>
16	(16)	(15)	(10)(11)(14)			(9)	(8)		(13)			(12)
17	(17)	(16)	(10) (11) (15)				(9)	(8)			(14)	(12)(13)
18	(18)	(17)	(10)(15)(16)	(11)				(9)				(8)(12)(13)(14)
19												
20												
21												
22												
22												

Register renaming

1	movq	(%rdi,%rax), %rsi → <b>P1</b>
2	movq	(%rcx,%rax), %r8 → P2
3	movq	%r8 ( <b>P1</b> ), (%rdi,%rax)
4	movq	%rsi(P2), (%rcx,%rax)
<b>⑤</b>	addq	\$8, %rax → <b>P3</b>
6	cmpq	%r9, %rax (P3)
7	jne	.L9
8	movq	(%rdi,%rax), %rsi → <b>P4</b>
9	movq	(%rcx,%rax), %r8 → P5
10	movq	%r8 (P4), (%rdi,%rax)
11	movq	%rsi(P11), (%rcx,%rax)
12	addq	\$8, %rax → P6
13	cmpq	%r9, %rax (P6)
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	m c	an	hav	ea	instr	uction a	t the	same cycle
	IF	ID	REN	AG	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	<b>(7)</b>	(6)	(3)(4)(5)				(1)					
8	(8)	<b>(7)</b>	(3)(4)(6)				(2)	(1)	(5)			
9	(9)	(8)	(3)(6)(7)	(4)				(2)				<del>-(1)(</del> 5)
10	(10)	(9)	(6)(7)(8)	(3)	(4)							<del>-(2)(</del> 5)
11	(11)	(10)	(7)(8)(9)		(3)	(4)			(6)			
12	(12)	(11)	(8)(9)(10)			(3)	(4)				(7)	(5)(6)
13	(13)	(12)	(9)(10)(11)	(8)			(3)	(4)				(5)(6)(7)
14	(14)	(13)	(10)(11)(12)	(9)	(8)			(3)				(4)(5)(6)(7)
15	(15)	(14)	(10)(11)(13)		(9)	(8)			(12)			(3)(4)(5)(6)(7)
16	(16)	(15)	(10)(11)(14)			(9)	(8)		(13)			(12)
17	(17)	(16)	(10)(11)(15)				(9)	(8)			(14)	(12)(13)
18	(18)	(17)	(10)(15)(16)	(11)				(9)				<del>(8)(</del> 12)(13)(14)
19	(19)	(18)	(15)(16)(17)	(10)	(11)							(9)(12)(13)(14)
20												
21												
22												
00												

1	movq	(%rdi,%rax), %rsi → <b>P1</b>
2	movq	(%rcx,%rax), %r8 → P2
3	movq	%r8 ( <b>P1</b> ), (%rdi,%rax)
4	movq	%rsi(P2), (%rcx,%rax)
5	addq	\$8, %rax → P3
6	cmpq	%r9, %rax (P3)
7	jne	.L9
8	movq	(%rdi,%rax), %rsi → <b>P4</b>
9	movq	(%rcx,%rax), %r8 → P5
10	movq	%r8 (P4), (%rdi,%rax)
11	movq	%rsi(P11), (%rcx,%rax)
12	addq	\$8, %rax → P6
13	cmpq	%r9, %rax (P6)
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
<u>21</u> )	jne	.L9

			Only 1 of	the	m c	anı	nav	ea	Instru	uction a	tine	same cycle
	IF	ID	REN	AG	M1	M2	МЗ	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	<b>(7)</b>	(6)	(3)(4)(5)			(2)	(1)					
8	(8)	<b>(7)</b>	(3)(4)(6)				(2)	(1)	(5)			
9	(9)	(8)	(3)(6)(7)	(4)				(2)				<del>-(1)(</del> 5)
10	(10)	(9)	(6)(7)(8)	(3)	(4)							<del>-(2)(</del> 5)
11	(11)	(10)	(7)(8)(9)		(3)	(4)			(6)			
12	(12)	(11)	(8)(9)(10)			(3)	(4)				(7)	(5)(6)
13	(13)	(12)	(9)(10)(11)	(8)			(3)	(4)				(5)(6)(7)
14	(14)	(13)	(10)(11)(12)	(9)	(8)			(3)				(4)(5)(6)(7)
15	(15)	(14)	(10)(11)(13)		(9)	(8)			(12)			(2)(4)(5)(6)(7)
16	(16)	(15)	(10)(11)(14)			(9)	(8)		(13)			(12)
17	(17)	(16)	(10)(11)(15)				(9)	(8)			(14)	(12)(13)
18	(18)	(17)	(10)(15)(16)	(11)				(9)				<del>(8)(</del> 12)(13)(14)
19	(19)	(18)	(15)(16)(17)	(10)	(11)							<del>(9)(</del> 12)(13)(14)
20	(20)	(19)	(16)(17)(18)	(15)	(10)	(11)						(12)(13)(14)
21												
22												
00												

movq	(%rdi,%rax), %rsi → <b>P1</b>
movq	(%rcx,%rax), %r8 → P2
movq	%r8 ( <b>P1</b> ), (%rdi,%rax)
movq	%rsi(P2), (%rcx,%rax)
addq	\$8, %rax → <b>P3</b>
cmpq	%r9, %rax (P3)
jne	.L9
movq	(%rdi,%rax), %rsi → <b>P4</b>
movq	(%rcx,%rax), %r8 → P5
movq	%r8 (P4), (%rdi,%rax)
movq	%rsi(P11), (%rcx,%rax)
addq	\$8, %rax → P6
cmpq	%r9, %rax (P6)
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 ine movq movq addq cmpq ine movq movq movq addq cmpq

			Only 1 of	tne	m c	anı	na⊽	e a	Instru	uction a	ttne	same cycle
	IF	ID	REN	AG	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	<b>(7)</b>	(6)	(3)(4)(5)			(2)	(1)					
8	(8)	<b>(7)</b>	(3)(4)(6)				(2)	(1)	(5)			
9	(9)	(8)	(3)(6)(7)	(4)				(2)				<del>-(1)(</del> 5)
10	(10)	(9)	(6)(7)(8)	(3)	(4)							<del>-(2)(</del> 5)
11	(11)	(10)	(7)(8)(9)		(3)	(4)			(6)			
12	(12)	(11)	(8)(9)(10)			(3)	(4)				(7)	(5)(6)
13	(13)	(12)	(9)(10)(11)	(8)			(3)	(4)				(5)(6)(7)
14	(14)	(13)	(10)(11)(12)	(9)	(8)			(3)				(4)(5)(6)(7)
15	(15)	(14)	(10)(11)(13)		(9)	(8)			(12)			(3)(4)(5)(6)(7)
16	(16)	(15)	(10)(11)(14)			(9)	(8)		(13)			(12)
<b>17</b>	(17)	(16)	(10)(11)(15)				(9)	(8)			(14)	(12)(13)
18	(18)	(17)	(10)(15)(16)	(11)				(9)				<del>(8)</del> (12)(13)(14)
19	(19)	(18)	(15)(16)(17)	(10)	(11)							<del>(9)(</del> 12)(13)(14)
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												

Register renaming

1	movq	(%rdi,%rax), %rsi → <b>P1</b>
2	movq	(%rcx,%rax), %r8 → P2
3	movq	%r8 ( <b>P1</b> ), (%rdi,%rax)
4	movq	%rsi(P2), (%rcx,%rax)
5	addq	\$8, %rax → P3
6	cmpq	%r9, %rax ( <b>P3</b> )
7	jne	.L9
8	movq	(%rdi,%rax), %rsi → <b>P4</b>
9	movq	(%rcx,%rax), %r8 → P5
10	movq	%r8 (P4), (%rdi,%rax)
11	movq	%rsi( <b>P11</b> ), (%rcx,%rax)
12	addq	\$8, %rax → P6
13	cmpq	%r9, %rax (P6)
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
<b>21</b> )	jne	.L9

			Only 1 of	the	m c	an I	hav	ea	instru	uction a	t the	same cycle
	IF	ID	REN	AG	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	<b>(7)</b>	(6)	(3)(4)(5)			(2)	(1)					
8	(8)	<b>(7)</b>	(3)(4)(6)				(2)	(1)	(5)			
9	(9)	(8)	(3)(6)(7)	(4)				(2)				<del>-(1)(</del> 5)
10	(10)	(9)	(6)(7)(8)	(3)	(4)							<del>-(2)(</del> 5)
11	(11)	(10)	(7)(8)(9)		(3)	(4)			(6)			
12	(12)	(11)	(8)(9)(10)			(3)	(4)				(7)	(5)(6)
13	(13)	(12)	(9)(10)(11)	(8)			(3)	(4)				(5)(6)(7)
14	(14)	(13)	(10) (11) (12)	(9)	(8)			(3)				(4)(5)(6)(7)
15	(15)	(14)	(10) (11) (13)		(9)	(8)			(12)			(2)(4)(5)(6)(7)
16	(16)	(15)	(10)(11)(14)			(9)	(8)		(13)			(12)
<b>17</b>	(17)	(16)	(10) (11) (15)				(9)	(8)			(14)	(12)(13)
18	(18)	(17)	(10)(15)(16)	(11)				(9)				<del>(8)(</del> 12)(13)(14)
19	(19)	(18)	(15)(16)(17)	(10)	(11)							<del>(9)(</del> 12)(13)(14)
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)	(10)	(11)	(19)			(12)(13)(14)

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

		- <u>-</u> - <u>-</u> .	01111/101		<u> </u>						
	Red	IF ID	REN	AG	M1	M2	M3 M4	ALU	MUL	BR	ROB
1	movq (%rdi,%rax), %rsi → P1	1 (1)									
2	movq (%rcx,%rax), %r8 $\rightarrow$ P2	2 (2) (1)									
3	movq %r8 (P1), (%rdi,%rax)	3 (3) (2)		(4)							
	• • • • • • • • • • • • • • • • • • • •	4 (4) (3)		(1)	(4)						
4	movq %rsi(P2), (%rcx,%rax)	5 (5) (4)		(2)	(1)						
5	addq \$8, %rax → P3	6 (6) (5)			(2)	(1)	(4)				
6	cmpq %r9, %rax (P3)	7 (7) (6)				(2)		(5)			
7	jne .L9	8 (8) (7)		(4)			(2) (1)	(5)			(4) (5)
8	movq (%rdi,%rax), %rsi → P4	9 (9) (8)		(4)			(2)				<del>(1)(</del> 5)
9		10 (10) (9) 11 (11) (10		(3)	(4)			(6)			<del>(2)(</del> 5)
_					(3)	(4)	(4)	(6)		(7)	(E)(G)
10		12 (12) (11		(0)		(3)	(4)			(7)	(5)(6) (5)(6)(7)
$\underbrace{11}_{\widehat{\Omega}}$	movq /0131(1 11), (/010x//01dx)	13 (13) (12		(8)			(3) (4)				(5)(6)(7)
12	addq \$8, %rax → P6	14 (14) (13		(9)	(8)		(3)	(12)			(4)(5)(6)(7)
13	cmpq %r9, %rax (P6)	15 (15) (14 16 (16) (15			(9)	(8)	(0)	(12)			<del>(2)(4)(5)(6)(7)</del>
(14)	jne .L9	17 (17) (16				(9)		(13)		(14)	(12) (12)(13)
15	movq (%rdi,%rax), %rsi	18 (18) (17		(11)			(9) (8) (9)			(14)	(12)(13) - <del>(8)</del> (12)(13)(14)
	movq (%rax,%rax), %r8	19 (19) (18 19 (19) (18		(10)			(9)				<del>(3)(</del> 12)(13)(14) <del>(3)(</del> 12)(13)(14)
<u>16</u>		20 (20) (19				(11)					(12)(13)(14)
<u>17</u>	1110 V q 701 O , (701 O 1 , 701 O X )	21 (21) (20				(10)					(12)(13)(14)
18	movq %rsi, (%rcx,%rax)	22 (21)		(10)			(10) (11)	(19)			(12)(13)(14)
19	addq \$8, %rax	23	(17)(10)(20)	(18)			(15) (11)				(12)(13)(14)(19)
20	cmpq %r9, %rax	24	(17)(20)(21)	(10)		(10)	(10) (10)				
_	ine .L9	25									

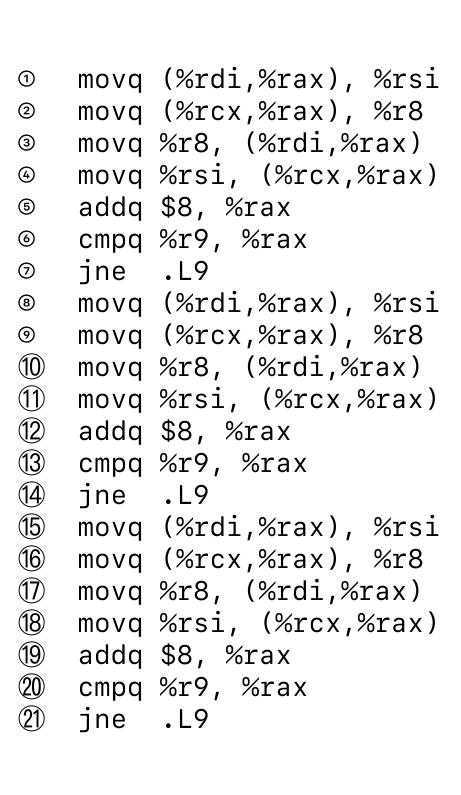
Only 1 of them can have a instruction at the same cycle

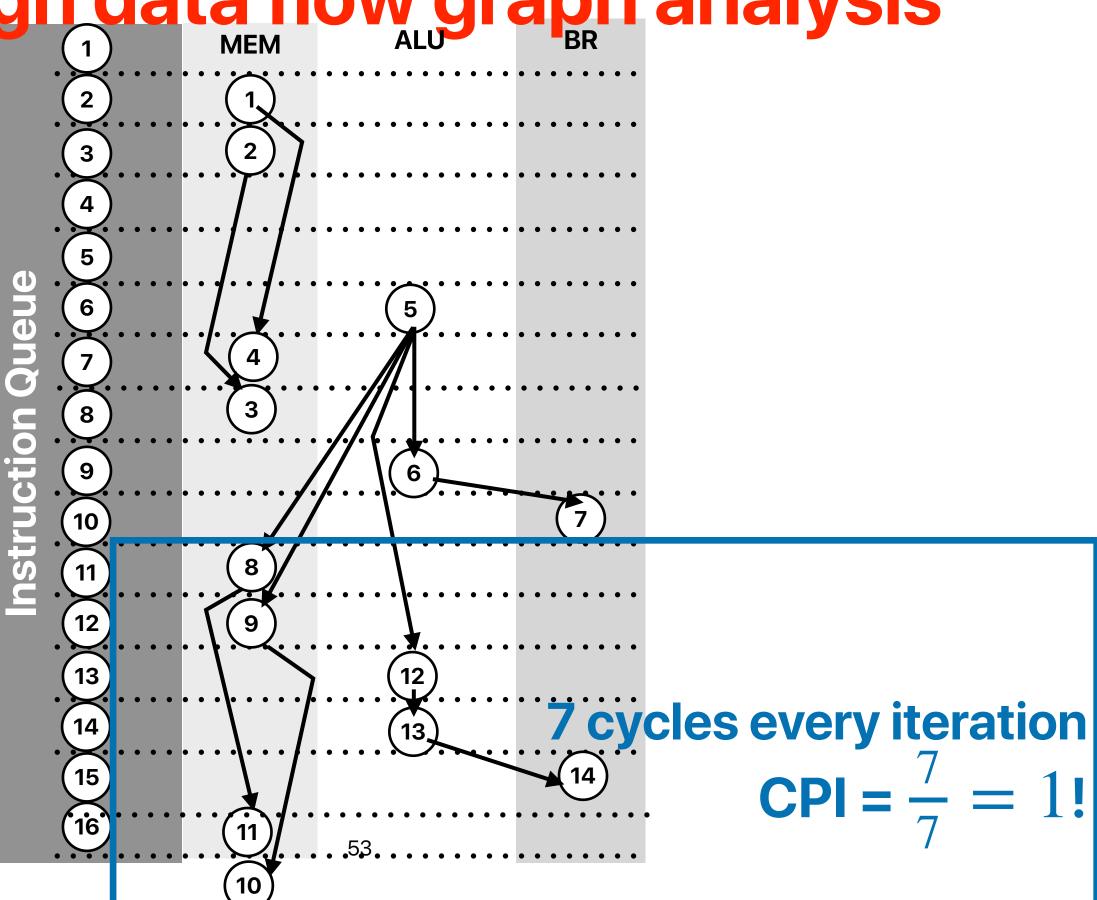
			Keg	IF	ID	REN	AG	M1	M2	М3	M4	ALU	MUL	BR	ROB
1	mova	(%rdi,%rax), %rsi →	P1	1 (1)											
2		(%rcx,%rax), %r8 →	D2	2 (2)		(4)									
3	•	%r8 (P1), (%rdi,%ra		3 (3) 4 (4)	(2) (3)	(1)	(1)								
4	•	%rsi(P2), (%rcx,%ra		5 (5)	(4)	(2) (3)	(1) (2)	(1)							
<u></u>	•	,	P3	6 (6)	( <del>-</del> )	(3)(4)	(2)		(1)						
<b>6</b>	•	%r9, %rax (P3)		7 (7)	(6)	(3)(4)(5)		<b>(—)</b>	(2)	(1)					
	• ' '	•		8 (8)	(7)	(3)(4)(6)					(1)	(5)			
⑦	jne	.L9	D/	9 (9)	(8)	(3)(6)(7)	(4)				(2)				<del>-(1)(</del> 5)
8	•	(%rdi,%rax), %rsi →		10 (10)	(9)	(6)(7)(8)	(3)	(4)							<del>-(2)(</del> 5)
9	•	(%rcx,%rax), %r8 →		11 (11)		(7)(8)(9)		(3)	(4)			(6)			
10	•	%r8 (P4), (%rdi,%ra		12 (12)		(8)(9)(10)			(3)					(7)	(5)(6)
1	movq	%rsi(P11), (%rcx,%)	LUX	13 (13)		(9)(10)(11)	(8)			(3)					(5)(6)(7)
12	addq	\$8, %rax →	PO	14 (14)		(10)(11)(12)	(9)	(8)	(0)		(3)	(40)			(4)(5)(6)(7)
13	cmpq	%r9, %rax (P6)		15 (15) 16 (16)		(10)(11)(13)		(9)	(8)	<b>/0</b> \		(12)			<del>(3)(4)(5)(6)(7)</del>
(14)	jne	.L9		16 (16) 17 (17)		(10)(11)(14) (10)(11)(15)			(9)	( <del>0</del> )	(8)	(13)		(14)	(12) (12)(13)
<u>15</u>		(%rdi,%rax), %rsi		18 (18)		(10)(11)(16)	(11)				(9)			(17)	(12)(13) (8)(12)(13)(14)
<u>16</u>	-	(%rcx,%rax), %r8		19 (19)		(15)(16)(17)		(11)			(-)				<del>(9)</del> (12)(13)(14)
17	-	%r8, (%rdi,%rax)		20 (20)		(16)(17)(18)			(11)						(12)(13)(14)
18	•	%rsi, (%rcx,%rax)		21 (21)	(20)	(17)(18)(19)	(16)	(15)	(10)	(11)					(12)(13)(14)
_	•	•		22	(21)	(17)(18)(20)		(16)	(15)	(10)	(11)	(19)			(12)(13)(14)
19	•	\$8, %rax		23		(17)(20)(21)	(18)		(16)	(15) (	(10)				(11)(12)(13)(14)(19)
<b>20</b>	cmpq	•		24		(20)(21)	(17)	(18)		(16)	(15)				(10)(11)(12)(13)(14)(19)
21)	Jne	.L9		25											

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

					Offiny 1 of	uje	ШС	anı	IIav	ea	Instruc	stion at	une	Same Cycle
					<b>M M O IO</b>	7								
		RE	IF	ID	REN	AG	M1	M2	M3	M4	ALU	MUL	BR	ROB
1	mova	(%rdi,%rax), %rsi <b>→ P1</b>	1 (1)											
2	-	(%rcx,%rax), %r8 → P2		(1)	<b></b>									
3	•	%r8 (P1), (%rdi,%rax)		(2)	(1)									
	<del>-</del>			(3)	(2)	(1)								
4	movq	%rsi(P2), (%rcx,%rax)		(4)	(3)	(2)	(1)							
5	addq	\$8, %rax → <b>P3</b>	6 (6)	(5)	(3)(4)		<b>(2)</b>	(1)						
6	cmpa	%r9, %rax (P3)	7 (7)	(6)	(3)(4)(5)			(2)	(1)					
7	jne	.L9	8 (8)	<b>(7)</b>	(3)(4)(6)				<b>(2)</b>	(1)	(5)			
			9 (9)	(8)	(3)(6)(7)	(4)				(2)				<del>-(1)(</del> 5)
8	-	(%rdi,%rax), %rsi → <b>P4</b>	10 (10)	(9)	(6)(7)(8)	(3)	<b>(4)</b>							<del>-(2)(</del> 5)
9	movq	(%rcx,%rax), %r8 → P5	11 (11) (	10)	(7)(8)(9)		(3)	(4)			(6)			
10	movq	%r8 (P4), (%rdi,%rax)	12 (12) (	(11)	(8)(9)(10)			(3)	<b>(4)</b>				<b>(7)</b>	(5)(6)
11	mova	%rsi(P11), (%rcx,%rax)	13 (13) (	12)	(9)(10)(11)	(8)			(3)	(4)				(5)(6)(7)
12	-	\$8, %rax → P6	14 (14) (	13)	(10)(11)(12)	(9)	(8)			(3)				(4)(5)(6)(7)
_	-		<b>15 (15) (</b> ′	14)	(10)(11)(13)		(9)	(8)			(12)			(3)(4)(5)(6)(7)
13	•	%r9, %rax (P6)	16 (16) (	15)	(10)(11)(14)			(9)	(8)		(13)			(12)
<b>14</b> )	Jne	.L9	17 (17) (°	16)	(10)(11)(15)				(9)	(8)			(14)	(12)(13)
15	movq	(%rdi,%rax), %rsi	18 (18) (	17)	(10)(15)(16)	(11)				(9)	7 cvc	les fo	r 7	<del>(8)(</del> 12)(13)(14)
<u>16</u> )	movq	(%rcx,%rax), %r8	19 (19) (	18)	(15)(16)(17)	(10)	(11)							<del>(9)(</del> 12)(13)(14)
<u>17</u>	•	%r8, (%rdi,%rax)	20 (20) (	19)	(16)(17)(18)	(15)	(10)	(11)			instr	uctio	ns	(12)(13)(14)
18	•	%rsi, (%rcx,%rax)	21 (21) (2	20)	(17)(18)(19)	(16)	(15)	(10)	(11)			PI = 1		(12)(13)(14)
_	•		22 (	21)	(17)(18)(20)		(16)	(15)	(10)	(11)	(19)	-1 - 1		(12)(13)(14)
19	•	\$8, %rax	23		(17)(20)(21)	(18)		(16)	(15)	(10)				(11)(12)(13)(14)(19)
20	cmpq	%r9, %rax	24		(20)(21)	(17)	(18)		(16)					(10)(11)(12)(13)(14)(19)
<u>21</u> )	jne	.L9	25		(21)		(17)	(18)		(16)	(20)			(15) (19)

Through data flow graph analysis



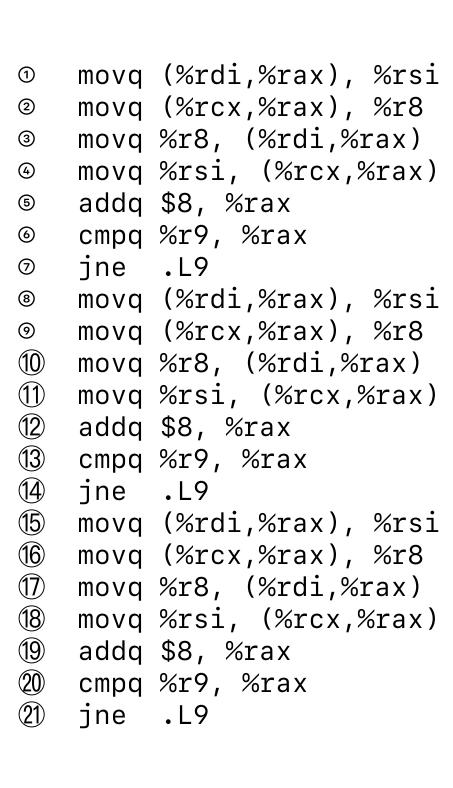


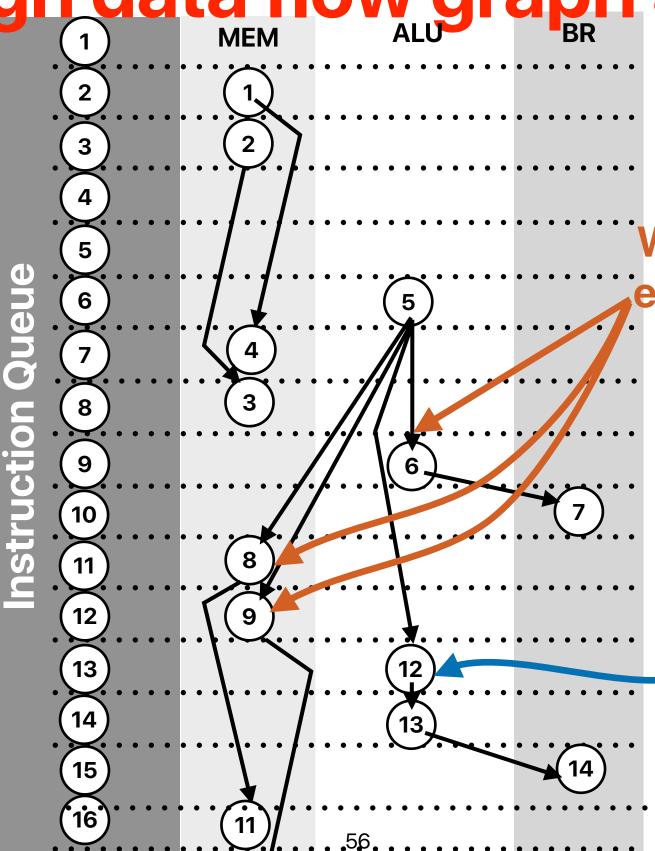
#### Takeaways: data hazards

- More data dependencies, more likelihood of data hazards
- Stalls and data forwarding can both address data hazards to generate correct code execution results — but not very efficient
- Compiler optimizations can help, but to a limited extent
- False dependencies limits the freedom of out-of-order execution
- Register renaming + Speculative execution enables more efficient execution by dynamically scheduling instructions whenever their data dependencies are resolved

# If CPI==1 the limitation?

Through data flow graph analysis





10

We cannot issue them earlier simply because structural hazards!

We could have this executed earlier if it's in the queue earlier

# Super Scalar

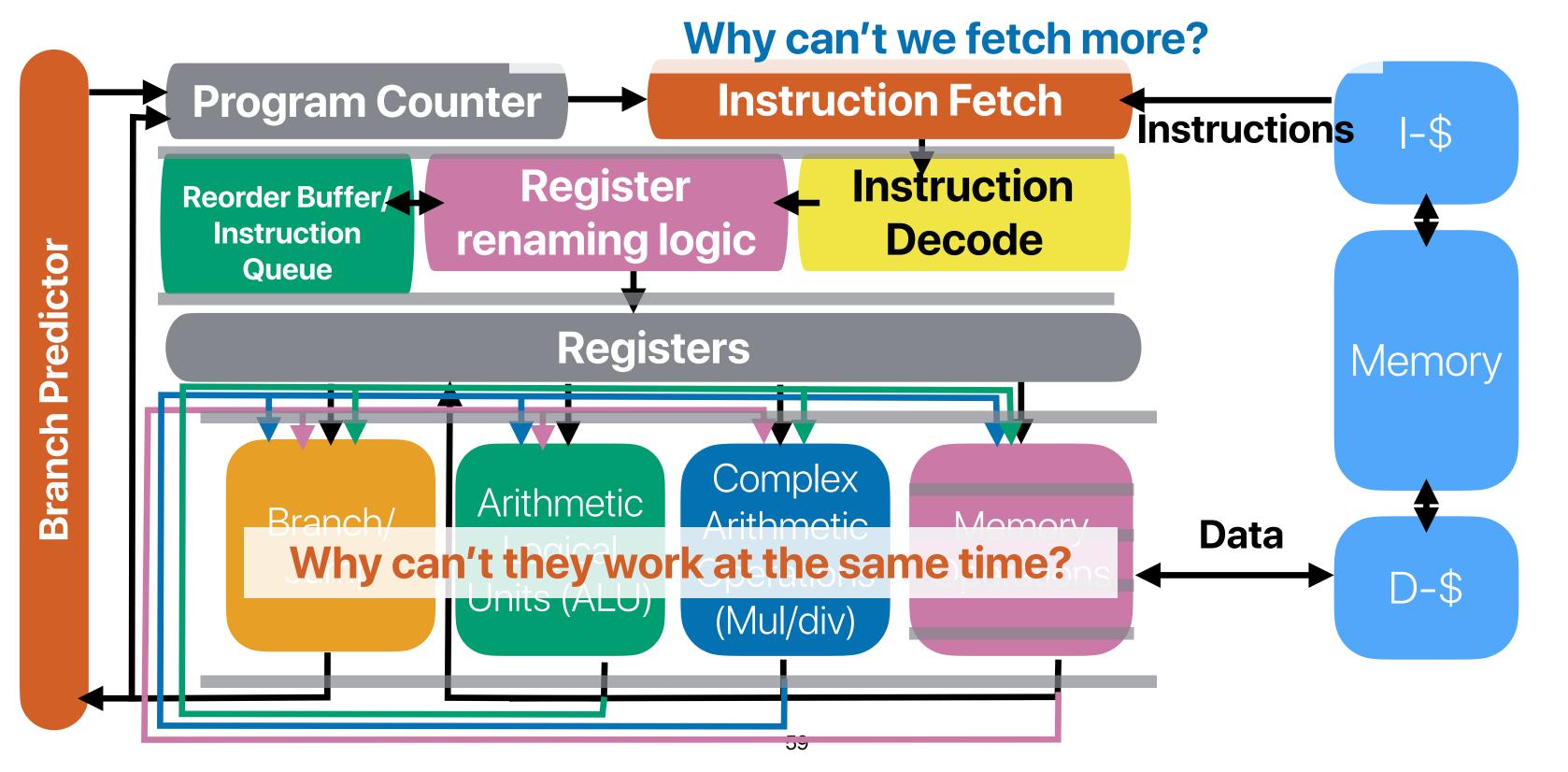
#### Superscalar

- 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

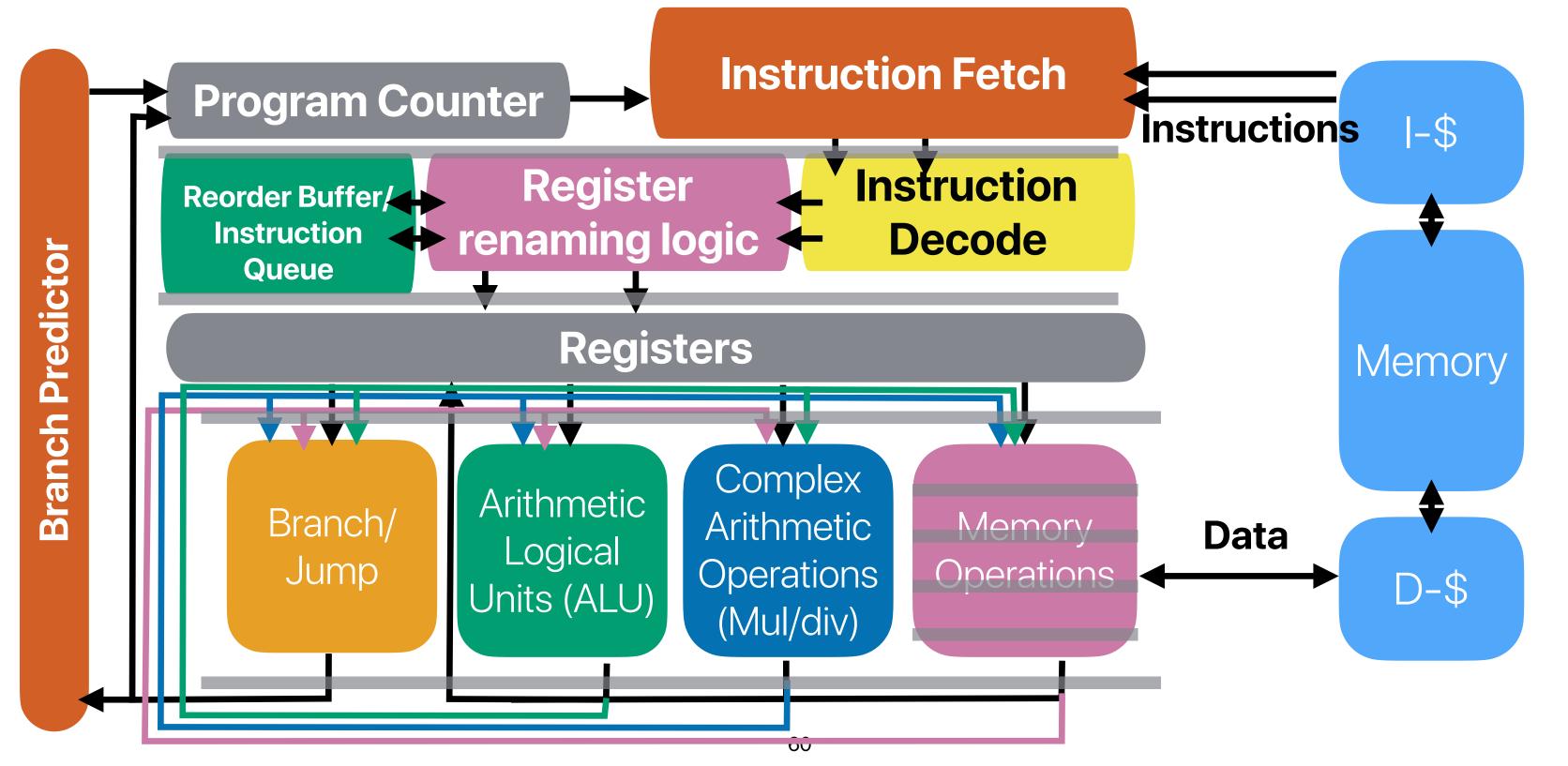
1

min(issue width, fetch width, decode width)

## Register renaming + OoO + RoB



## Register renaming + SuperScalar



1	movq	(%rdi,%rax), %rsi	→ P:	L			issue:	4	oi ti	nem	can	nav	eal	nstruc	tion a	t the same cycle
2	movq	(%rcx,%rax), %r8	→ P2	2	IF	ID	REN	AG	M1	M2	МЗ	M4	ALU	MUL	BR	ROB
3	movq	%r8, (%rdi,%rax)		1	(1)(2)											
4	movq	%rsi, (%rcx,%rax)			(3)(4)											
<b>5</b>	addq	\$8, %rax	→ P;	3	(5)(6)	(3)(4)	(1)(2)									
6	cmpq	%r9, %rax		4												
7	jne			5												
8	movq	(%rdi,%rax), %rsi	$\rightarrow P'$	+ 6 7												
9	movq	(%rcx,%rax), %r8	→ P!	8												
10	movq	%r8, (%rdi,%rax)		9												
11	movq	%rsi, (%rcx,%rax)		10												
12	addq	\$8, %rax	$\rightarrow$ P(	11												
13	cmpq	%r9, %rax		12												
14	jne	.L9		13												
15	movq	(%rdi,%rax), %rsi		14												
16	movq	(%rcx,%rax), %r8		15												
17)	movq	%r8, (%rdi,%rax)		16												
18	movq	%rsi, (%rcx,%rax)		17												
19	addq	\$8, %rax		18 19												
20	cmpq	%r9, %rax		20												
<b>(21)</b>	ine	<b>.</b> L 9		20			61									

1	movq	(%rdi,%rax), %rsi	→ P1			_	155ue.	4	or tr	ICIII	Cai	IIIav	e a i	ristique	tion at	tile Saille	cycle
2	movq	(%rcx,%rax), %r8	$\rightarrow P2$		IF	ID	REN	AG	M1	M2	М3	M4	ALU	MUL	BR	ROB	
3	movq	%r8, (%rdi,%rax)		1	(1)(2)												
4	movq	%rsi, (%rcx,%rax)			(3)(4)												
5	addq	\$8, %rax	→ P3				(1)(2)										
6	cmpq	%r9, %rax		4	(7)(8)	(5)(6)	(2)(3)(4)	(1)									
7	jne	.L9		5													
8	movq	(%rdi,%rax), %rsi	→ P4	5 7													
9	movq	(%rcx,%rax), %r8	→ P5	8													
_		%r8, (%rdi,%rax)		9													
11	movq	%rsi, (%rcx,%rax)		10													
12	addq	\$8, %rax	$\rightarrow$ P6	11													
13	cmpq	%r9, %rax		12													
14	jne	.L9		13													
15	movq	(%rdi,%rax), %rsi		14													
16	movq	(%rcx,%rax), %r8		15													
17	movq	%r8, (%rdi,%rax)		16													
18	movq	%rsi, (%rcx,%rax)		17													
		\$8, %rax		18 19													
		%r9, %rax		20													
<b>21</b> )		.L9		20			62										

1	movq	(%rdi,%rax), %rsi	$\rightarrow$ P:	L		_	10040.	1			oai	iiiav				the dame by	
2	movq	(%rcx,%rax), %r8	$\rightarrow$ P:	2	IF	ID	REN	AG	M1	M2	М3	M4	ALU	MUL	BR	ROB	
3	movq	%r8, (%rdi,%rax)		1	(1)(2)												
4	movq	%rsi, (%rcx,%rax)	_		(3)(4)												
<b>5</b>	addq	\$8, %rax	→ P;		(5)(6)		(1)(2)										
6	cmpq	%r9, %rax			(7)(8)		(2)(3)(4)	(1)									
7	jne	.L9			(9)(10)	(7)(8)	(3)(4)(5)(6)	(2)	(1)								
8	movq	(%rdi,%rax), %rsi	$\rightarrow P$	4 6													
9	movq	(%rcx,%rax), %r8	$\rightarrow$ P	5 /													
_		%r8, (%rdi,%rax)		9													
11	movq	%rsi, (%rcx,%rax)		10													
12	addq	\$8, %rax	$\rightarrow$ P														
13	cmpq	%r9, %rax		12	2												
_	jne			13	3												
15	movq	(%rdi,%rax), %rsi		14													
16	movq	(%rcx,%rax), %r8		15													
17)	movq	%r8, (%rdi,%rax)		16													
18	movq	%rsi, (%rcx,%rax)		17													
	<del>-</del>	\$8, %rax		18													
		%r9, %rax		19 20													
_	ine			20			63										

1	movq	(%rdi,%rax), %rsi	→ P1				. 133uc.	4	oi ti	ICIII	Cai	IIIAV			LIOILA	' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' '	CIC
2	movq	(%rcx,%rax), %r8	→ P2		IF	ID	REN	AG	M1	M2	МЗ	M4	ALU	MUL	BR	ROB	
3	movq	%r8, (%rdi,%rax)		1	(1)(2)												
4	movq	%rsi, (%rcx,%rax)			(3)(4)												
(5)	addq	\$8, %rax	→ P3				(1)(2)										
6	cmpq	%r9, %rax			(7)(8)			(1)									
7	jne						(3)(4)(5)(6)	(2)		44X							
8	movq	(%rdi,%rax), %rsi	$\rightarrow PZ$	6	(11)(12)	(9)(10)	(3)(4)((6)(7)(8)		(2)	(1)							
9	movq	(%rcx,%rax), %r8	→ P5	/ 0													
_		%r8, (%rdi,%rax)		9													
_	<del>-</del>	%rsi, (%rcx,%rax)		10													
12	addq	\$8, %rax	$\rightarrow P6$														
13	cmpq	%r9, %rax		12													
14	jne	.L9		13													
15	movq	(%rdi,%rax), %rsi		14													
16	movq	(%rcx,%rax), %r8		15													
	<del>-</del>	%r8, (%rdi,%rax)		16													
	<del>-</del>	%rsi, (%rcx,%rax)		17													
_	•	\$8, %rax		18													
		%r9, %rax		19													
$\overline{(21)}$		19		20			64										

1	movq	(%rdi,%rax), %rsi	<ul><li>P1</li></ul>				. 10040.	1			- Cai	iiiav				the same sy	
2	movq	(%rcx,%rax), %r8	P2		IF	ID	REN	AG	M1	M2	МЗ	M4	ALU	MUL	BR	ROB	
3	movq	%r8, (%rdi,%rax)		1	(1)(2)												
4	movq	%rsi, (%rcx,%rax)			(3)(4)												
<b>(5)</b>	addq	\$8, %rax	• P3														
6	cmpq	%r9, %rax				(5)(6)		(1)									
	jne					(7)(8)		(2)		141			<b>(</b> =)				
8	movq	(%rdi,%rax), %rsi	• P4	-			(3)(4)((6)(7)(8)		(2)		(1)		(5)			<b>(E)</b>	
9	movq	(%rcx,%rax), %r8	<b>P5</b>	2	(13)(14)	(11)(12)	(3)(4)((6)(7)(9) (10)	(0)		(2)	(1)					(5)	
		%r8, (%rdi,%rax)		9													
11	movq	%rsi, (%rcx,%rax)		10													
12	addq	\$8, %rax	<b>P6</b>														
13	cmpq	%r9, %rax		12													
_	jne			13													
15	movq	(%rdi,%rax), %rsi		14													
_	_	(%rcx,%rax), %r8		15													
17)	movq	%r8, (%rdi,%rax)		16													
18	movq	%rsi, (%rcx,%rax)		17													
	-	\$8, %rax		18													
		%r9, %rax		19 20													
	ine			20			65										

1	movq	(%rdi,%rax), %rsi-	<ul><li>P1</li></ul>				. 10000.	4			Cai	iiiav				ne dame eye	
2	movq	(%rcx,%rax), %r8	<b>P2</b>		IF	ID	REN	AG	M1	M2	МЗ	M4	ALU	MUL	BR	ROB	
3	movq	%r8, (%rdi,%rax)		1	(1)(2)												
4	movq	%rsi, (%rcx,%rax)			(3)(4)												
<b>5</b>	addq	\$8, %rax	<b>P3</b>			(3)(4)											
6	cmpq	%r9, %rax				(5)(6)		(1)									
7						(7)(8)		(2)		141			<b>(</b> =)				
8	movq	(%rdi,%rax), %rsi	<b>→</b> P4	-			(3)(4)((6)(7)(8)		(2)		(4)		(5)			<b>(E)</b>	
9	movq	(%rcx,%rax), %r8	<b>P5</b>	•			(3)(4)((6)(7)(9) (10) (3)(4)(7)(10)(11) (12)				(1)	(1)	(6)		(7)	(5) (5)(6)	
		%r8, (%rdi,%rax)		9	(10)(10)	(10)(14)	(12)	(9)	(0)		(2)	(1)			(7)	(3)(0)	
11	movq	%rsi, (%rcx,%rax)		10													
12	addq	\$8, %rax	<b>P6</b>														
13	cmpq	%r9, %rax		12													
_	jne			13													
15	movq	(%rdi,%rax), %rsi		14													
16	movq	(%rcx,%rax), %r8		15													
17)	movq	%r8, (%rdi,%rax)		16													
18	movq	%rsi, (%rcx,%rax)		17													
	-	\$8, %rax		18													
		%r9, %rax		19 20													
	ine			20			66										

1	•	(%rdi,%rax), %rsi						7								
2	movq	(%rcx,%rax), %r8	$\rightarrow P_2$	2	IF	ID	REN	AG	M1	M2	МЗ	M4	ALU	MUL	BR	ROB
3	movq	%r8, (%rdi,%rax)		1	(1)(2)											
4	movq	%rsi, (%rcx,%rax)			(3)(4)											
<b>(5)</b>	addq	\$8, %rax	→ P;		(5)(6)											
6	cmpq	%r9, %rax			(7)(8)			(1)								
7	jne	.L9			(9)(10)			(2)		(4)			4>			
8	movq	(%rdi,%rax), %rsi	$\rightarrow P$	•			(3)(4)((6)(7)(8)		(2)		(4)		(5)			<b>(E)</b>
9	movq	(%rcx,%rax), %r8	→ P!				(3)(4)((6)(7)(9) (10) (3)(4)(7)(10)(11)			(2)		(1)	(6)		(7)	(5)
		%r8, (%rdi,%rax)					(3)(4)(7)(10)(11) (12) (3)(10)(11)(13) (14)				(2)		(12)		(7)	(5)(6) (1)(5)(6)(7)
$\overline{11}$	movq	%rsi, (%rcx,%rax)		10		(10)(10)	(14)	()	(3)	(0)		(2)	(12)			(1)(3)(0)(7)
_	<del>-</del>		$\rightarrow$ P(	11												
_	•	%r9, %rax		12												
_	jne '			13												
_	_	(%rdi,%rax), %rsi		14												
_	-	(%rcx,%rax), %r8		15												
_	=	%r8, (%rdi,%rax)		16												
	<del>-</del>	%rsi, (%rcx,%rax)		17												
	_	\$8, %rax		18												
	_	%r9, %rax		19												
	ine			20			67									

1	movq	(%rdi,%rax), %rsi	→ P	1			. 10000.	T								tile same cycle
2	movq	(%rcx,%rax), %r8	$\rightarrow$ P	2	IF	ID	REN	AG	M1	M2	МЗ	M4	ALU	MUL	BR	ROB
3	movq	%r8, (%rdi,%rax)		1	(1)(2)											
4	movq	%rsi, (%rcx,%rax)	_		(3)(4)											
<b>5</b>	addq	\$8, %rax	→ P		(5)(6)											
6	cmpq	%r9, %rax			(7)(8)			(1)								
7	jne	.L9			(9)(10)			(2)								
8	movq	(%rdi,%rax), %rsi	$\rightarrow$ P				(3)(4)((6)(7)(8)		(2)		(4)		(5)			<b>(5</b> )
9	movq	(%rcx,%rax), %r8	$\rightarrow$ P	5 1			(3)(4)((6)(7)(9) (10) (3)(4)(7)(10)(11)			(2)		(4)	(6)		(7)	(5)
		%r8, (%rdi,%rax)					(3)(4)(7)(10)(11) (12) (3)(10)(11)(13)				(2)		(12)		(7)	(5)(6)
_	•	%rsi, (%rcx,%rax)					(3)(10)(11)(13) (14) (10)(11)(14)(15) (16)				(8)	(2)	(12) (13)			(1)(5)(6)(7) (2)(5)(6)(7)(12)
_	•	•	→ P	6 11		(17)(10)	(16)	(3)	(+)	(3)	(0)		(15)			(2)(0)(0)(7)(12)
	-	%r9, %rax		12												
_	jne			13												
_	_	(%rdi,%rax), %rsi		14	ļ											
_	<del>-</del>	(%rcx,%rax), %r8		15	5											
_	-	%r8, (%rdi,%rax)		16	<b>;</b>											
_	-	%rsi, (%rcx,%rax)		17	,											
	<del>-</del>	\$8, %rax		18	3											
		%r9, %rax		19												
_	ine			20	)		68									

1	•	(%rdi,%rax), %rsi →						7								
2	movq	(%rcx,%rax), %r8 →	<b>P2</b>		IF	ID	REN	AG	M1	M2	МЗ	M4	ALU	MUL	BR	ROB
3	movq	%r8, (%rdi,%rax)		1	(1)(2)											
4	movq	%rsi, (%rcx,%rax)			(3)(4)											
<b>(5)</b>	addq	\$8, %rax →	<b>P3</b>	3	(5)(6)	(3)(4)	(1)(2)									
6	cmpq	%r9, %rax			(7)(8)			(1)								
7	jne	.L9			(9)(10)			(2)		(4)						
8	movq	(%rdi,%rax), %rsi→	<b>P4</b>				(3)(4)((6)(7)(8)		(2)		(4)		(5)			<b>(E)</b>
9	movq	(%rcx,%rax), %r8 →	<b>P5</b>				(3)(4)((6)(7)(9) (10) (3)(4)(7)(10)(11)			(2)		(1)	(6)		(7)	(5) (5)(6)
		%r8, (%rdi,%rax)					(3)(4)(7)(10)(11) (12) (3)(10)(11)(13) (14)				(2)	(2)	(12)		(7)	(5)(6) (1)(5)(6)(7)
(11)	movq	%rsi, (%rcx,%rax)					(14) (10)(11)(14)(15) (16)				(8)	(2)	(12)			(2)(5)(6)(7)(12)
_	-	\$8, %rax →	P6				(16) (10)(11)(16)(17) (18)					(8)	(10)		(14)	(5)(6)(7)(12)(13)
_	•	%r9, %rax		12			(18)		<b>\</b> - <b>,</b>	<b>,</b> ,		(-,			,	
_		.L9		13												
_	_	(%rdi,%rax), %rsi		14												
_	-	(%rcx,%rax), %r8		15												
	<del>-</del>	%r8, (%rdi,%rax)		16												
		%rsi, (%rcx,%rax)		17												
		\$8, %rax		18												
		%r9, %rax		19												
	_	.L9		20			69									

1	movq	(%rdi,%rax), %rsi	→ P:	L				7								
2	movq	(%rcx,%rax), %r8	$\rightarrow$ P:	2	IF	ID	REN	AG	M1	M2	МЗ	M4	ALU	MUL	BR	ROB
3	movq	%r8, (%rdi,%rax)		1	(1)(2)											
4	movq	%rsi, (%rcx,%rax)		2	(3)(4)	(1)(2)										
<b>⑤</b>	addq	\$8, %rax	→ P;		(5)(6)		(1)(2)									
6	cmpq	%r9, %rax			(7)(8)			(1)								
7	jne	.L9			(9)(10)		(3)(4)(5)(6)	(2)								
8	movq	(%rdi,%rax), %rsi	$\rightarrow$ P	T .			(3)(4)((6)(7)(8)		(2)		(4)		(5)			
9	movq	(%rcx,%rax), %r8	$\rightarrow$ P	5 64			(3)(4)((6)(7)(9) (10) (3)(4)(7)(10)(11)			(2)	(1)	(1)	(6)		(7)	(5)
_		%r8, (%rdi,%rax)					(3)(4)(7)(10)(11) (12) (3)(10)(11)(13) (14)		(8)	(8)	(2)		(12)		(7)	(5)(6) (1)(5)(6)(7)
_	•	%rsi, (%rcx,%rax)					(14) (10)(11)(14)(15) (16)					(2)	(12)			(2)(5)(6)(7)(12)
_	•	•	$\rightarrow$ P									(8)	(10)		(14)	(5)(6)(7)(12)(13)
	•	%r9, %rax		12			(18) (16)(17)(18)(19) (20								( /	(5)(6)(7)(8)(12)(13)
_	jne	-		13	3		(20					,				(4.1)
_	_	(%rdi,%rax), %rsi		14												
_	-	(%rcx,%rax), %r8		15												
_	_	%r8, (%rdi,%rax)		16												
	-	%rsi, (%rcx,%rax)		17	,											
		\$8, %rax		18												
	=	%r9, %rax		19												
_				20			70									

1	movq	(%rdi,%rax), %rsi	$\rightarrow$	<b>P1</b>				issue:	4	oi u	iem	car	ınav	eall	nstruc	tion at	. the same cycl
2	movq	(%rcx,%rax), %r8	$\rightarrow$	<b>P2</b>		IF	ID	REN	AG	M1	M2	МЗ	M4	ALU	MUL	BR	ROB
3	movq	%r8, (%rdi,%rax)			1	(1)(2)											
4	movq	%rsi, (%rcx,%rax)				(3)(4)											
<b>⑤</b>	addq	\$8, %rax	$\rightarrow$	<b>P3</b>	3	(5)(6)	(3)(4)	(1)(2)									
6	cmpq	%r9, %rax			4	(7)(8)	(5)(6)	(2)(3)(4)	(1)								
7	jne	•				(9)(10)		(3)(4)(5)(6)	(2)								
8		(%rdi,%rax), %rsi	$\rightarrow$	<b>P4</b>		(11)(12)				(2)				(5)			
9	mova	(%rcx,%rax), %r8	$\rightarrow$	<b>P5</b>				(3)(4)((6)(7)(9) (10)		(0)	(2)		(4)	(6)		(7)	(5)
10		%r8, (%rdi,%rax)						(3)(4)(7)(10)(11) (12) (3)(10)(11)(13)		(8)	(0)	(2)	(1)	(40)		(7)	(5)(6)
11	•	%rsi, (%rcx,%rax)						(3)(10)(11)(13) (14) (10)(11)(14)(15)		(9)		(0)	(2)	(12)			(1)(5)(6)(7)
12	•	•	<b>→</b>	<b>P6</b>	10			(10)(11)(14)(15) (16) (10)(11)(16)(17)			(9)		(0)	(13)		(4.4)	(2)(5)(6)(7)(12)
	•	· - 1		. 0		(21)(22)		(10)(11)(16)(17) (18)			(4)		(8)			(14)	(5)(6)(7)(12)(13)
13		%r9, %rax			12		(21)(22)	(16)(17)(18)(19) (20			(3)			(40)			(5)(6)(7)(8)(12)(13)
14	jne				13			(16)(17)(18)(20) (21)(22)	(10)	(11)	(15)	(3)	(4)	(19)			(5)(6)(7)(8)(9)(12)(13)(14)
(15)	movq	(%rdi,%rax), %rsi			14												
16	movq	(%rcx,%rax), %r8			15												
	<del>-</del>	%r8, (%rdi,%rax)			16												
		%rsi, (%rcx,%rax)			17												
		\$8, %rax			18												
		%r9, %rax			19												
_	ine				20			71									
I (ZI)	1116	. L Y						/ I									

1	movq	(%rdi,%rax), %rsi	$\Rightarrow$	<b>P1</b>				. 155ue.	4	oi ti	iem	Car	IIIav	e a II	istituc	tion at	the same cycle
2	movq	(%rcx,%rax), %r8	$\rightarrow$	<b>P2</b>		IF	ID	REN	AG	M1	M2	МЗ	M4	ALU	MUL	BR	ROB
3	movq	%r8, (%rdi,%rax)			1	(1)(2)											
4	movq	%rsi, (%rcx,%rax)				(3)(4)	(1)(2)										
5	addq	\$8, %rax	$\rightarrow$	P3	3	(5)(6)		(1)(2)									
6	cmpq	%r9, %rax			4		(5)(6)	(2)(3)(4)	(1)								
7	jne	.L9				(9)(10)		(3)(4)(5)(6)		(1)				<i>-</i>			
8	movq	(%rdi,%rax), %rsi	$\rightarrow$	<b>P4</b>		(11)(12)		(3)(4)((6)(7)(8)			(1)	(4)		(5)			<b>(E)</b>
9	movq	(%rcx,%rax), %r8	$\rightarrow$	<b>P5</b>				(3)(4)((6)(7)(9) (10) (3)(4)(7)(10)(11)			(2)	(1)	(1)	(6)		(7)	(5) (5)(6)
10		%r8, (%rdi,%rax)				(17)(18)		(3)(4)(7)(10)(11) (12) (3)(10)(11)(13) (14)		(8) (9)	(8)	(2)	(1) (2)	(12)		(7)	(5)(6) (1)(5)(6)(7)
(11)	movq	%rsi, (%rcx,%rax)						(14) (10)(11)(14)(15) (16)			(9)	(8)	(2)	(12)			(2)(5)(6)(7)(12)
12	addq	\$8, %rax	$\rightarrow$	<b>P6</b>	11	(21)(22)		(16) (10)(11)(16)(17) (18)		(3)			(8)	(10)		(14)	(5)(6)(7)(12)(13)
13	-	%r9, %rax			12			(16) (16)(17)(18)(19) (20			(3)		(9)			,	(5)(6)(7)(8)(12)(13)
14	jne	.L9			13			(16)(17)(18)(20) (21)(22)			(15)		(4)	(19)			(5)(6)(7)(8)(9)(12)(13)(14)
<u>(15)</u>		(%rdi,%rax), %rsi			14			(=:/(==/	(16)	(10)	(11)	(15)	(3)	(20)			(4)(5)(6)(7)(8)(9)(12)(13)(14) (19)
_	•	(%rcx,%rax), %r8			15												
_	<del>-</del>	%r8, (%rdi,%rax)			16												
	•	%rsi, (%rcx,%rax)			17												
_	-	\$8, %rax			18												
	<del>-</del>	%r9, %rax			19												
<u>(21)</u>		19			20			72									

1	movq	(%rdi,%rax), %rsi	→ P:	L		_	. 133UC.	4	oi ti	ICIII	Cai	IIIav		i sti-uc	LIOTE	i trie same cyci
2	movq	(%rcx,%rax), %r8	$\rightarrow$ P2		IF	ID	REN	AG	M1	M2	МЗ	M4	ALU	MUL	BR	ROB
3	movq	%r8, (%rdi,%rax)		1	(1)(2)											
4	movq	%rsi, (%rcx,%rax)	_		(3)(4)	(1)(2)										
<b>5</b>	addq	\$8, %rax	→ P3	3	(5)(6)	(3)(4)	(1)(2)									
6	cmpq	%r9, %rax		4	(7)(8)		(2)(3)(4)	(1)								
7	jne	.L9			(9)(10)		(3)(4)(5)(6)	(2)		(4)			<b>4</b> -5			
8	movq	(%rdi,%rax), %rsi	$\rightarrow$ P		(11)(12)		(3)(4)((6)(7)(8)		(2)		(4)		(5)			<b>(E)</b>
9		(%rcx,%rax), %r8			(13)(14)		(3)(4)((6)(7)(9) (10) (3)(4)(7)(10)(11)		(9)	(2)		(1)	(6)		<i>(</i> 7)	(5) (5)(6)
10	movq	%r8, (%rdi,%rax)			(17)(18)		(3)(4)(7)(10)(11) (12) (3)(10)(11)(13) (14)		(8) (9)	(8)	(2)	(1) (2)	(12)		(7)	(5)(6) (1)(5)(6)(7)
11)	movq	%rsi, (%rcx,%rax)			(19)(20)					(9)	(8)	(2)	(12)			(2)(5)(6)(7)(12)
(12)	-	\$8, %rax	$\rightarrow$ P(	_			(10)(11)(16)(17)			(4)		(8)	(10)		(14)	(5)(6)(7)(12)(13)
_	•	%r9, %rax		12		(21)(22)	(18) (16)(17)(18)(19) (20			(3)		(9)				(5)(6)(7)(8)(12)(13)
14	jne	.L9		13						(15)		(4)	(19)			(5)(6)(7)(8)(9)(12)(13)(14)
15	movq	(%rdi,%rax), %rsi		14					(10)	(11)	(15)	(3)	(20)			(4)(5)(6)(7)(8)(9)(12)(13)(14) (19)
_	-	(%rcx,%rax), %r8		15					(16)	(10)	(11)	(15)			(21)	(3)(4)(5)(6)(7)(8)(9)(12)(13) (14)(19)(20)
	-	%r8, (%rdi,%rax)		16												
	=	%rsi, (%rcx,%rax)		17												
	•	\$8, %rax		18												
		%r9, %rax		19												
<u>(21)</u>	jne			20			73									

1	movq	(%rdi,%rax), %rsi	$\Rightarrow$	<b>P1</b>				. 155ue.	4	OI LI	lem	Cai	IIIav	e a II	istituc	tion at	. the same cycle
2	movq	(%rcx,%rax), %r8	$\rightarrow$	<b>P2</b>		IF	ID	REN	AG	M1	M2	МЗ	M4	ALU	MUL	BR	ROB
3	movq	%r8, (%rdi,%rax)			1	(1)(2)											
4	movq	%rsi, (%rcx,%rax)				(3)(4)	(1)(2)										
<b>5</b>	addq	\$8, %rax	$\rightarrow$	<b>P3</b>	3	(5)(6)	(3)(4)	(1)(2)									
6	cmpq	%r9, %rax			4		(5)(6)	(2)(3)(4)	(1)								
7	jne	.L9				(9)(10)		(3)(4)(5)(6)		(1)				<i>-</i> >			
8	movq	(%rdi,%rax), %rsi	$\rightarrow$	<b>P4</b>		(11)(12)		(3)(4)((6)(7)(8)			(1)	(4)		(5)			<b>(E)</b>
9	movq	(%rcx,%rax), %r8	$\rightarrow$	<b>P5</b>		(13)(14)		(3)(4)((6)(7)(9) (10) (3)(4)(7)(10)(11)				(1)	(1)	(6)		(7)	(5) (5)(6)
10		%r8, (%rdi,%rax)				(17)(18)		(12) (3)(10)(11)(13) (14)	(-)	(8) (9)		(2)	(1) (2)	(12)		(7)	(5)(6) (1)(5)(6)(7)
$\overline{11}$	•	%rsi, (%rcx,%rax)						(14) (10)(11)(14)(15) (16)			(9)	(8)	(2)	(12)			(2)(5)(6)(7)(12)
	-	\$8, %rax	$\rightarrow$	<b>P6</b>	11	(21)(22)	(19)(20)	(16) (10)(11)(16)(17)		(3)			(8)	(10)		(14)	(5)(6)(7)(12)(13)
<u>13</u>	-	%r9, %rax			12			(18) (16)(17)(18)(19) (20					(9)			(,	(5)(6)(7)(8)(12)(13)
14	jne	.L9			13			(16)(17)(18)(20) (21)(22)			(15)		(4)	(19)			(5)(6)(7)(8)(9)(12)(13)(14)
<u>(15)</u>		(%rdi,%rax), %rsi			14			(=:)(==)		(10)	(11)	(15)	(3)	(20)			(4)(5)(6)(7)(8)(9)(12)(13)(14) (19)
_	•	(%rcx,%rax), %r8			15					(16)	(10)	(11)	(15)			(21)	(3)(4)(5)(6)(7)(8)(9)(12)(13) (14)(19)(20)
_	_	%r8, (%rdi,%rax)			16				(17)		(16)	(10)	(11)				(12)(13)(14)(15)(19)(20(21)
_	-	%rsi, (%rcx,%rax)			17												
_	-	\$8, %rax			18												
	-	%r9, %rax			19												
<u> </u>		19			20			74									

1	movq	(%rdi,%rax), %rsi	$\rightarrow$ P	1				155ue.	4	oi ti	lem	Gai	IIIav		istitue	uortai	tille Same Cych
2	movq	(%rcx,%rax), %r8	$\rightarrow$ P	2	[	F II	D	REN	AG	M1	M2	МЗ	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	→ P	3	3 (5)	(6) (3)		(1)(2)									
6	cmpq	%r9, %rax			4 (7)			(2)(3)(4)	(1)								
7		.L9	_			10) (7)		(3)(4)(5)(6)	(2)		(4)			<b>(</b> E)			
8	movq	(%rdi,%rax), %rsi	→ P	4				(3)(4)((6)(7)(8)		(2)		(1)		(5)			<b>(E)</b>
9	movq	(%rcx,%rax), %r8	$\rightarrow$ P	5				(3)(4)((6)(7)(9) (10) (3)(4)(7)(10)(11) (12)		(8)	(2)	(1)	(1)	(6)		(7)	(5) (5)(6)
10		%r8, (%rdi,%rax)				(18) (15)		(12) (3)(10)(11)(13) (14)		(9)	(8)	(2)	(2)	(12)		(7)	(1)(5)(6)(7)
11	movq	%rsi, (%rcx,%rax)						(14) (10)(11)(14)(15) (16)			(9)	(8)	(-/	(13)			(2)(5)(6)(7)(12)
12	addq	\$8, %rax	$\rightarrow$ P	6				(16) (10)(11)(16)(17) (18)			(4)		(8)	,		(14)	(5)(6)(7)(12)(13)
13	cmpq	%r9, %rax		•	12	(21)		(16)(17)(18)(19) (20		(15)	(3)	(4)	(9)				(5)(6)(7)(8)(12)(13)
14)	jne	.L9		•	13			(16)(17)(18)(20) (21)(22)					(4)	(19)			(5)(6)(7)(8)(9)(12)(13)(14)
15	movq	(%rdi,%rax), %rsi		•	14								(3)	(20)			(4)(5)(6)(7)(8)(9)(12)(13)(14) (19)
16	movq	(%rcx,%rax), %r8		ľ	15					(16)	(10)	(11)	(15)			(21)	(12)(13)(14)(19)(20)
17)	movq	%r8, (%rdi,%rax)			16				(17)		(16)						(3)(4)(5)(6)(7)(8)(9)(12)(13) (14)(19)(20)
18	movq	%rsi, (%rcx,%rax)			17					(17)		(16)	(10)				(11)(12)(13)(14)(15)(19)(20(21)
_	<del>-</del>	\$8, %rax			18 10												
20	<del>-</del>	%r9, %rax			19 20												
<b>21</b> )	jne	.L9			20			75									

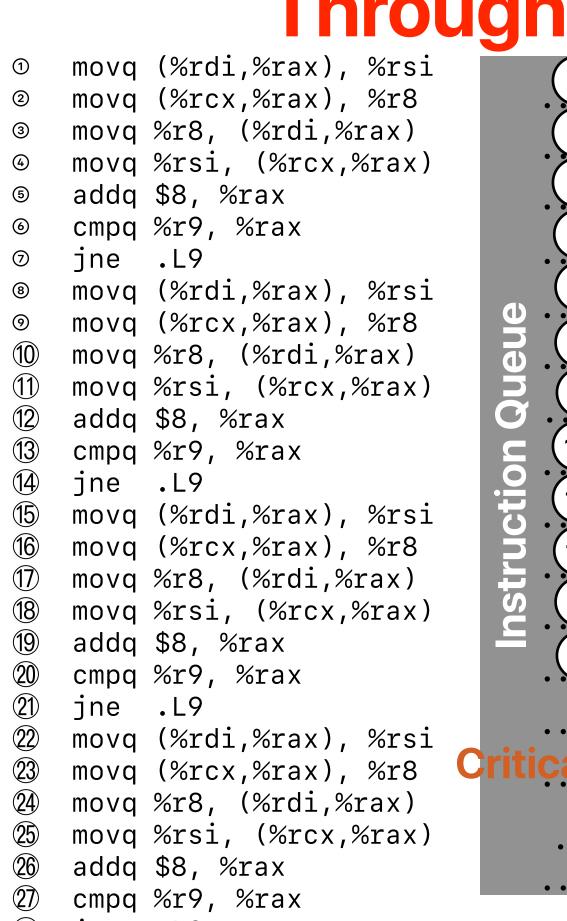
1	movq	(%rdi,%rax), %rsi	$\rightarrow$	<b>P1</b>				. 133UC.	4	oi ti	ICIII	Cai	IIIav			CIOILA	' same cycle
2	movq	(%rcx,%rax), %r8	$\rightarrow$	<b>P2</b>		IF	ID	REN	AG	M1	M2	МЗ	M4	ALU	MUL	BR	ROB
3	movq	%r8, (%rdi,%rax)			1	(1)(2)											
4	movq	%rsi, (%rcx,%rax)				(3)(4)	(1)(2)										
<b>5</b>	addq	\$8, %rax	$\rightarrow$	P3	3	(5)(6)	(3)(4)	(1)(2)									
6	cmpq	%r9, %rax			4		(5)(6)	(2)(3)(4)	(1)								
7	jne	.L9			5		(7)(8)	(3)(4)(5)(6)		(1)							
8	movq	(%rdi,%rax), %rsi	$\rightarrow$	<b>P4</b>		(11)(12)					(1)	(4)		(5)			<b>(F)</b>
9	movq	(%rcx,%rax), %r8	$\rightarrow$	<b>P5</b>				(3)(4)((6)(7)(9) (10) (3)(4)(7)(10)(11)			(2)		(1)	(6)		(7)	(5) (5)(6)
10		%r8, (%rdi,%rax)				(17)(18)		(3)(4)(7)(10)(11) (12) (3)(10)(11)(13) (14)		(8) (9)	(8)	(2)	(1) (2)	(12)		(7)	(5)(6) (1)(5)(6)(7)
(11)	movq	%rsi, (%rcx,%rax)				(19)(20)		(14) (10)(11)(14)(15) (16)			(9)	(8)	(2)	(12)			(2)(5)(6)(7)(12)
_	•	\$8, %rax	$\rightarrow$	<b>P6</b>	11	(21)(22)		(16) (10)(11)(16)(17) (18)			(4)		(8)	(10)		(14)	(5)(6)(7)(12)(13)
	•	%r9, %rax			12			(18) (16)(17)(18)(19) (20				(4)	(9)			<b>V y</b>	(5)(6)(7)(8)(12)(13)
<u>14</u> )		.L9			13						(15)		(4)	(19)			(5)(6)(7)(8)(9)(12)(13)(14)
<u>(15)</u>		(%rdi,%rax), %rsi			14			(= -/(==/		(10)	(11)	(15)	(3)	(20)			(4)(5)(6)(7)(8)(9)(12)(13)(14) (19)
_	-	(%rcx,%rax), %r8			15					(16)	(10)	(11)	(15)			(21)	(3)(4)(5)(6)(7)(8)(9)(12)(13) (14)(19)(20)
<u>17</u>	_	%r8, (%rdi,%rax)			16				(17)		(16)	(10)	(11)				(12)(13)(14)(15)(19)(20(21)
18	-	%rsi, (%rcx,%rax)			17					(17)		(16)	(10)				(11)(12)(13)(14)(15)(19)(20(21)
19	•	\$8, %rax			18						(17)		(16)				(10)(11)(12)(13)(14)(15)(19) (20)(21)
20	-	%r9, %rax			19												
<u>21</u> )	jne	.L9			20			76									

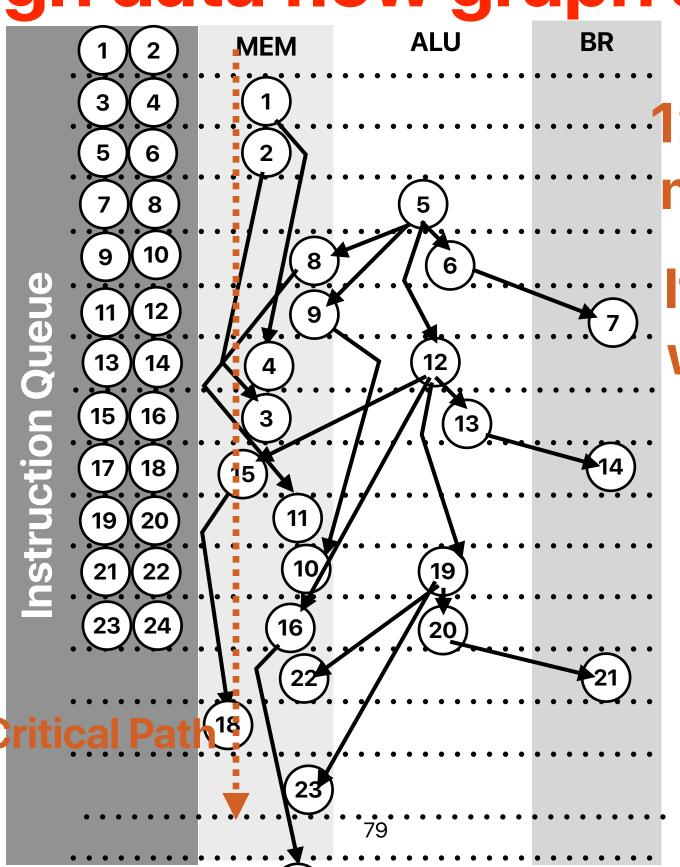
1	movq	(%rdi,%rax), %rsi	<ul><li>P1</li></ul>			_	. 133uc.	4	oi ti	ICIII	Cai	IIIav		IISTIC	tions	i trie same cycle
2	movq	(%rcx,%rax), %r8	<b>P2</b>		IF	ID	REN	AG	M1	M2	МЗ	M4	ALU	MUL	BR	ROB
3	movq	%r8, (%rdi,%rax)		1	(1)(2)											
4	movq	%rsi, (%rcx,%rax)			(3)(4)											
5	addq	\$8, %rax	<b>P3</b>	3	(5)(6)	(3)(4)	(1)(2)									
6	cmpq	%r9, %rax		4	(7)(8)		(2)(3)(4)	(1)								
7		.L9		5			(3)(4)(5)(6)	(2)		(4)			<b>(=</b> )			
8	movq	(%rdi,%rax), %rsi	<b>P4</b>		(11)(12)		(3)(4)((6)(7)(8)	(0)		(1)	(4)		(5)			<b>(E)</b>
9	movq	(%rcx,%rax), %r8	<b>P5</b>				(3)(4)((6)(7)(9) (10) (3)(4)(7)(10)(11) (12)	(8)			(1)	(1)	(6)		(7)	(5) (5)(6)
10		%r8, (%rdi,%rax)		9	(17)(18)		(12) (3)(10)(11)(13) (14)			(8)	(2)	(1) (2)	(12)		(7)	(5)(6) (1)(5)(6)(7)
11	movq	%rsi, (%rcx,%rax)					(14) (10)(11)(14)(15) (16)			(9)	(8)	(-)	(12)			(2)(5)(6)(7)(12)
12	addq	\$8, %rax	▶ P6				(16) (10)(11)(16)(17) (18)			(4)		(8)	(10)		(14)	(5)(6)(7)(12)(13)
13	cmpq	%r9, %rax		12			(16) (17) (18) (19) (20				(4)	(9)				(5)(6)(7)(8)(12)(13)
	jne	.L9		13			(16)(17)(18)(20) (21)(22)		(11)	(15)	(3)	(4)	(19)			(5)(6)(7)(8)(9)(12)(13)(14)
15	movq	(%rdi,%rax), %rsi		14				(16)	(10)	(11)	(15)	(3)	(20)			(4)(5)(6)(7)(8)(9)(12)(13)(14) (19)
<u>16</u>	<del>-</del>	(%rcx,%rax), %r8		15					(16)	(10)	(11)	(15)			(21)	(3)(4)(5)(6)(7)(8)(9)(12)(13) (14)(19)(20)
$\widetilde{17}$	-	%r8, (%rdi,%rax)		16				(17)		(16)	(10)	(11)				(12)(13)(14)(15)(19)(20(21)
18	-	%rsi, (%rcx,%rax)		17					(17)		(16)	(10)				(11)(12)(13)(14)(15)(19)(20(21)
<u>19</u>	•	\$8, %rax		18						(17)		(16)				(10)(11)(12)(13)(14)(15)(19) (20)(21)
20	•	%r9, %rax		19				(18)			(17)					(16)(19)(20)(21)
<u>21</u>	jne	.L9		20			77									

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

1	movq	(%rdi,%rax), %rsi	$\Rightarrow$	<b>P1</b>				. 155ue.	4	oi ti	lem	Cai	IIIav		istinc	tiona	' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' '
2	movq	(%rcx,%rax), %r8	$\rightarrow$	<b>P2</b>		IF	ID	REN	AG	M1	M2	МЗ	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	$\rightarrow$	P3	3	(5)(6)	(3)(4)	(1)(2)									
6	cmpq	%r9, %rax			4		(5)(6)	(2)(3)(4)	(1)								
7	jne	.L9				(9)(10)		(3)(4)(5)(6)		(1)	(4)			<b>(</b> =)			
8	movq	(%rdi,%rax), %rsi	$\rightarrow$	<b>P4</b>		(11)(12)		(3)(4)((6)(7)(8)		(2)		(1)		(5)			(E)
9	movq	(%rcx,%rax), %r8	$\rightarrow$	<b>P5</b>		(15)(14)		(3)(4)((6)(7)(9) (10) (3)(4)(7)(10)(11)		(8)	(2)		(1)	(6)		(7)	(5) (5)(6)
10		%r8, (%rdi,%rax)				(17)(18)		(12) (3)(10)(11)(13) (14)	(0)	(9)	(8)	(2)	(2)	(12)		(7)	(5)(6) (1)(5)(6)(7)
11	movq	%rsi, (%rcx,%rax)				(19)(20)					(9)	(8)	(-/	(13)			<del>(2)(</del> 5)(6)(7)(12)
12	addq	\$8, %rax	$\rightarrow$	P6	11	(21)(22)	(19)(20)	(16) (10)(11)(16)(17)			(4)	(9)	(8)	,		(14)	(5)(6)(7)(12)(13)
13	cmpq	%r9, %rax			12		(21)(22)	(16)(17)(18)(19) (20	(11)	(15)	(3)	(4)	(9)				(5)(6)(7)(8)(12)(13)
14	jne	.L9			13			(16)(17)(18)(20) (21)(22)		(11)	(15)	(3)	(4)	(19)			(5)(6)(7)(8)(9)(12)(13)(14)
15	movq	(%rdi,%rax), %rsi			14				(16)	(10)	(11)	(15)	(3)	(20)			(4)(5)(6)(7)(8)(9)(12)(13)(14) (19)
16	movq	(%rcx,%rax), %r8			15					(16)	(10)	(11)	(15)			(21)	(3)(4)(5)(6)(7)(9)(9)(12)(13) (14)(19)(20)
<u>17</u> )	movq	%r8, (%rdi,%rax)			16				(17)		(16)						(12)(13)(14)(15)(19)(20(21)
18)	-	%rsi, (%rcx,%rax)			17					(17)		(16)	(10)				(11)(12)(13)(14)(15)(19)(20(21)
19	addq	\$8, %rax			18				(40)		(17)	(47)	(16)				(20)(21) (20)(21)
<u>20</u>	<u>-</u>	%r9, %rax			19				(18)			(17)	(17)				<del>(16)</del> (19)(20)(21)
$\widetilde{21}$	jne	.L9			20			78		(18)			(17)				(19)(20)(21)

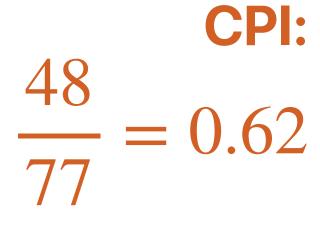
Through data flow graph analysis





12 cycles for every 11memory instructions

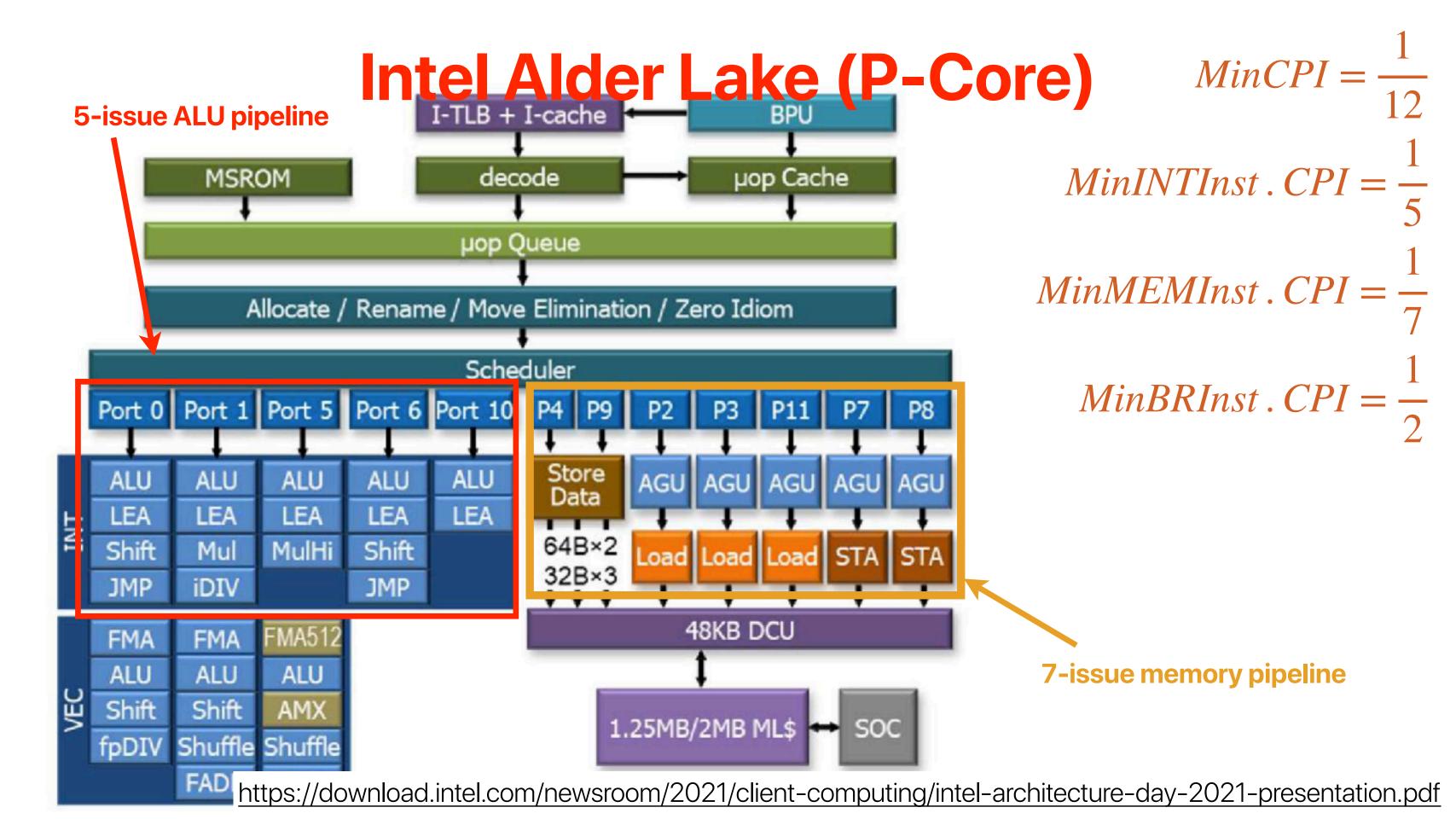
If we have 11 loops, it will have 44 memory instructions, 77 instructions in total and take 48 cycles



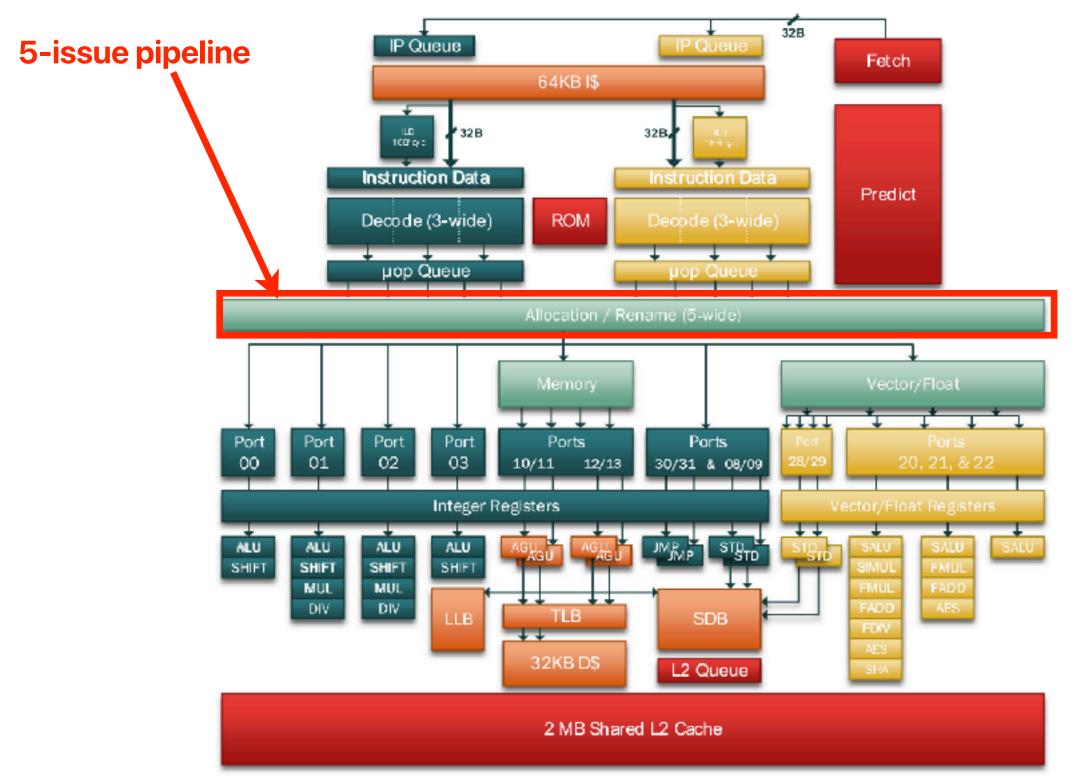
# Takeaways: data hazards

- More data dependencies, more likelihood of data hazards
- Stalls and data forwarding can both address data hazards to generate correct code execution results — but not very efficient
- Compiler optimizations can help, but to a limited extent
- False dependencies limits the freedom of out-of-order execution
- Register renaming + Speculative execution enables more efficient execution by dynamically scheduling instructions whenever their data dependencies are resolved
- Super scalar further improves the utilization of hardware and throughput

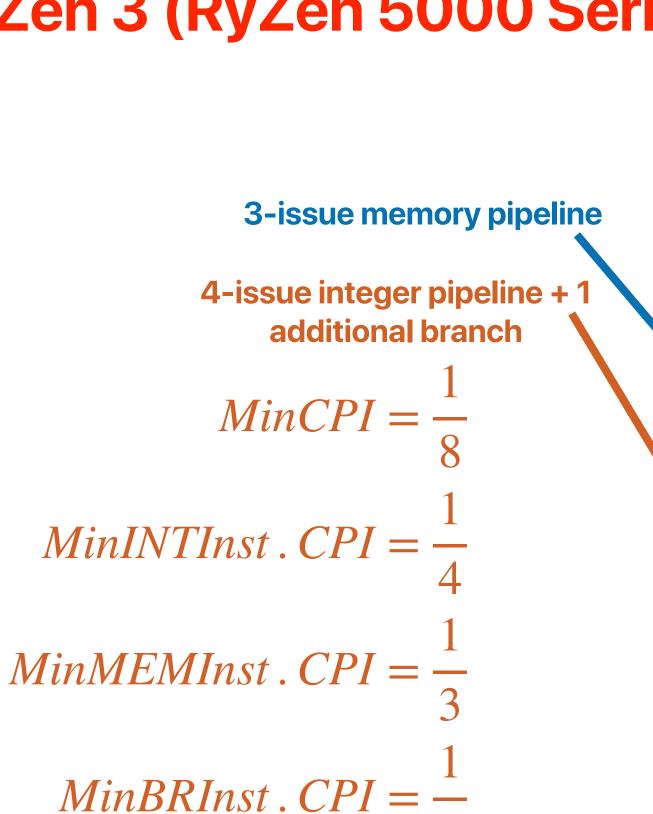
# The pipelines of Modern Processors

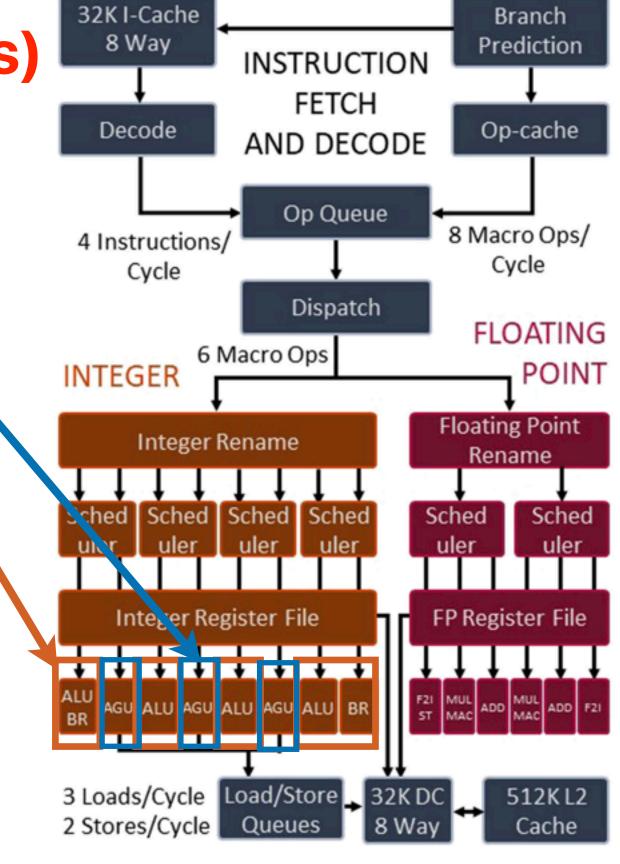


# Intel Alder Lake (E-Core)



# AMD Zen 3 (RyZen 5000 Series)





#### 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

# Takeaways: data hazards

- More data dependencies, more likelihood of data hazards
- Stalls and data forwarding can both address data hazards to generate correct code execution results — but not very efficient
- Compiler optimizations can help, but to a limited extent
- False dependencies limits the freedom of out-of-order execution
- Register renaming + Speculative execution enables more efficient execution by dynamically scheduling instructions whenever their data dependencies are resolved
- Super scalar further improves the utilization of hardware and throughput
- Modern processors are all very wide-issue super scalar processors with OoO capabilities



# 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 **unlimited** number of instructions per cycle, with exactly the same register renaming hardware and pipeline as we showed previously (**5-cycle** memory access latencies, 100% hit rate).

① .L3:

```
do {
    number_of_nodes++;
    current = current->next;
} while ( current != NULL )
A. 0.5
B. 0.75
C. 1.0
D. 1.25
E. 1.5
```

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





# 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 **unlimited** number of instructions per cycle, with exactly the same register renaming hardware and pipeline as we showed previously (**5-cycle** memory access latencies, 100% hit rate).

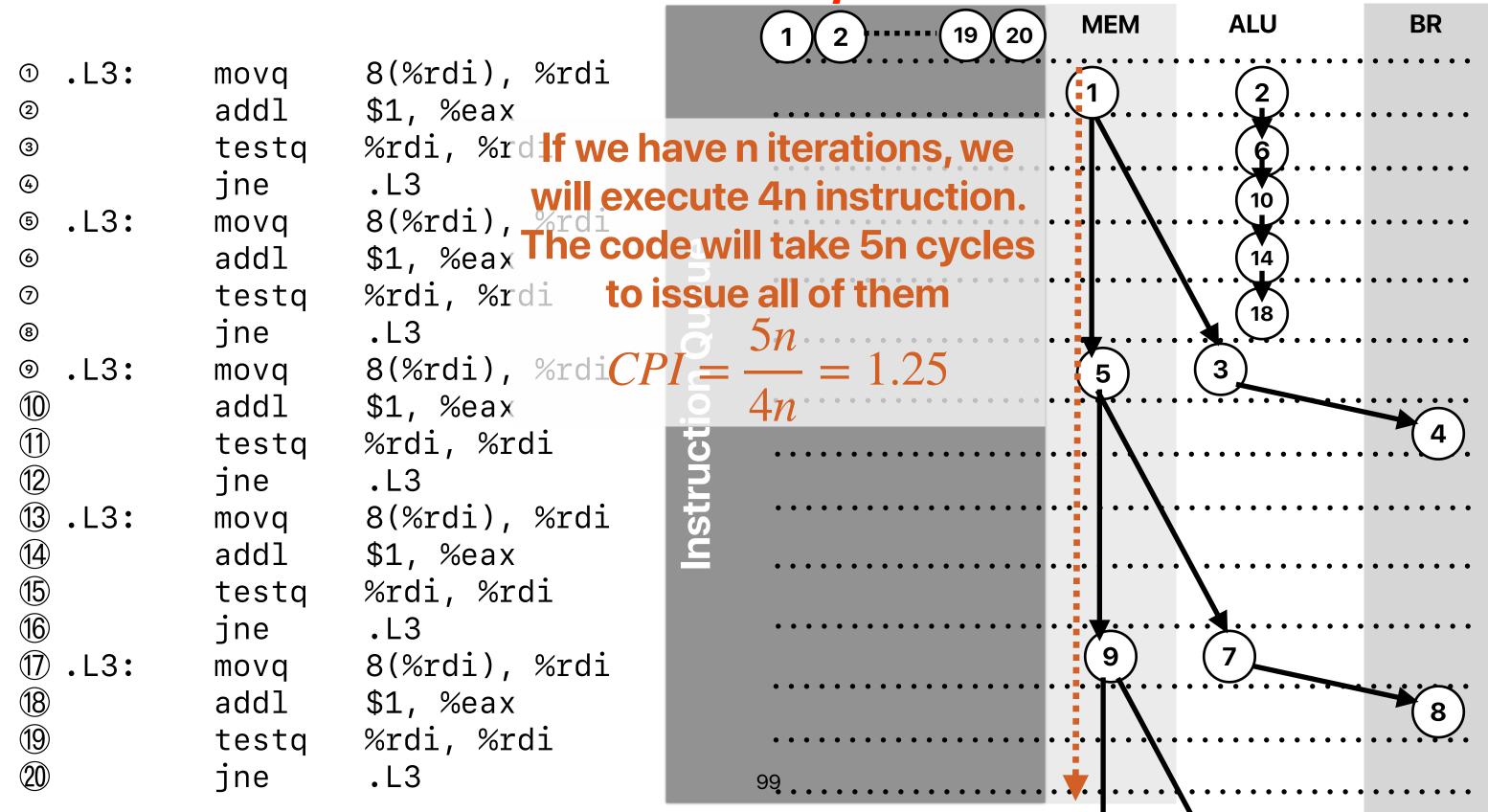
```
do {
    number_of_nodes++;
    current = current->next;
} while ( current != NULL )
A. 0.5
B. 0.75
C. 1.0
D. 1.25
E. 1.5
```

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



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
<pre>    .L3:</pre>	movq	8(%rdi), %rdi
10	addl	<b>\$1,</b> %eax
11)	testq	%rdi, %rdi
12	jne	.L3
① .L3:	movq	8(%rdi), %rdi
14	addl	\$1, %eax
15	testq	%rdi, %rdi
16	jne	.L3
① .L3:	movq	8(%rdi), %rdi
18	addl	<b>\$1,</b> %eax
19	testq	%rdi, %rdi
20	jne	.L3

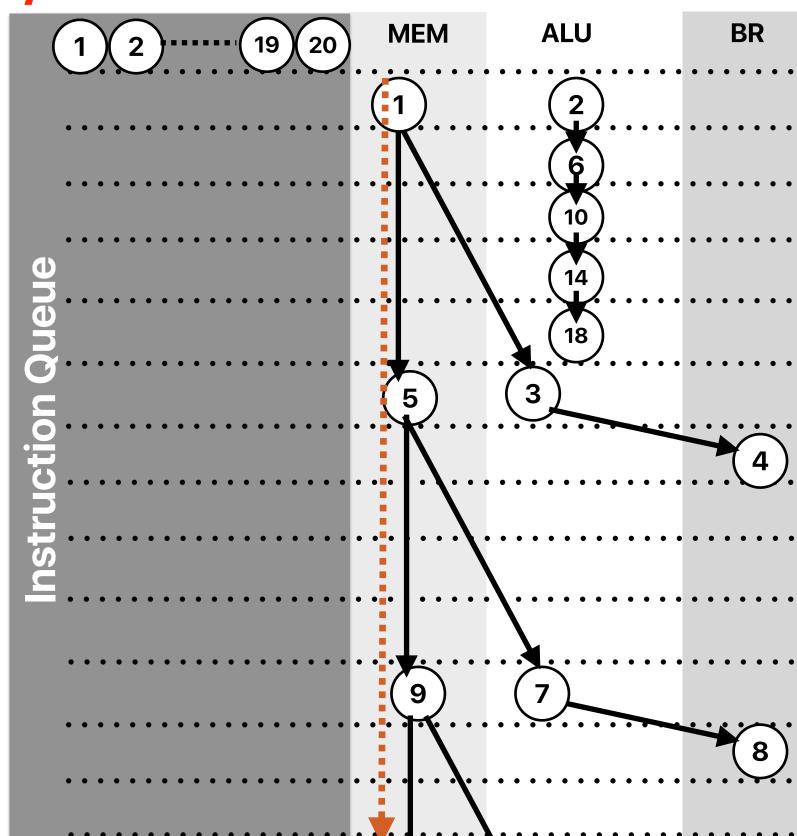
1 17	Ficily 155uc Wil	701 <del>-</del>	- III IKGU	IIISL
	1 2 19 20	MEM	ALU	BR
	• • • • • • • • • • • • • • •	• • • • • •		• • • • • •
	• • • • • • • • • • • • • • • •	• • • • • •	• • • • • • • • • •	• • • • • •
4	• • • • • • • • • • • • • • • • • • • •	• • • • • •	• • • • • • • • • • •	• • • • • •
nstruction Queue	• • • • • • • • • • • • • • • •	• • • • • •	• • • • • • • • • •	• • • • • •
On	• • • • • • • • • • • • • • • • • • • •	• • • • • •		• • • • • •
tion	• • • • • • • • • • • • • • • • •	• • • • • • •	• • • • • • • • • •	• • • • • •
) C	• • • • • • • • • • • • • • • • •	• • • • • • •	• • • • • • • • • • •	• • • • • •
nsti	• • • • • • • • • • • • • • • • •	• • • • • • •	• • • • • • • • • • •	• • • • • •
	• • • • • • • • • • • • • • • • • • • •			
	• • • • • • • • • • • • • • • • • • • •			
	98			



If we cannot improve the performance of executing movq 8(%rdi), %rdi we cannot improve the execution time. That's the "critical path"!

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

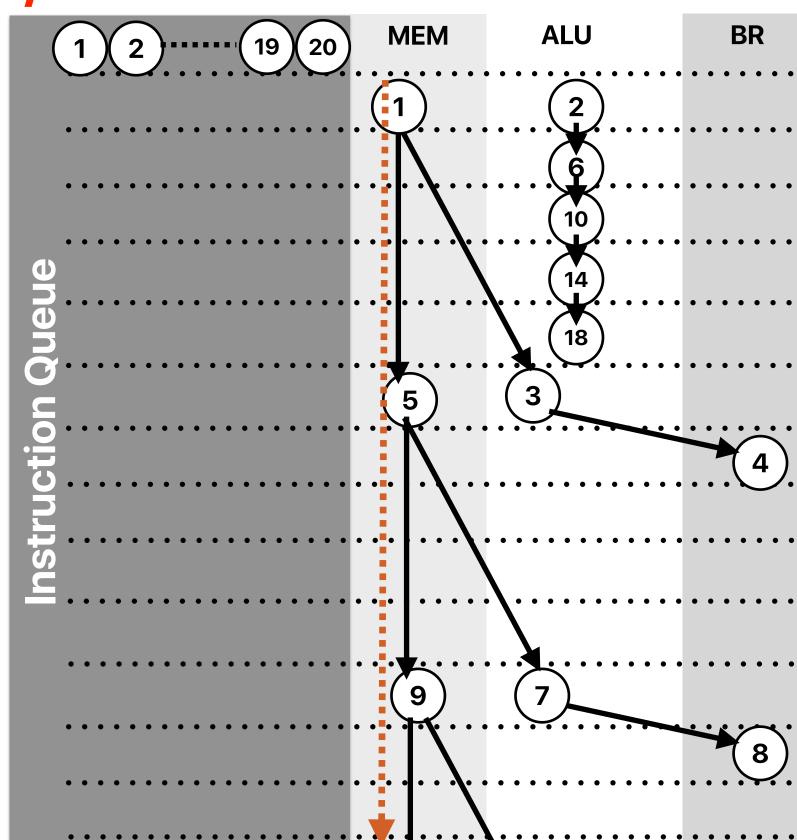
① .L3: movq 8(%rdi), %rdi
② addl $1, %eax
③ testq %rdi, %rdi
④ jne .L3
```



If we cannot improve the performance of executing movq 8(%rdi), %rdi we cannot improve the execution time. That's the "critical path"!

```
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
```



**MEM** 

(19)(20)

**ALU** 

BR

If we cannot improve the performance of executing movq 8(%rdi), %rdi we cannot improve the execution time.

8(%rdi), %rdi

\$1, %eax

.L3

%rdi, %rdi

That's the "critical path"!

mova

addl

jne

testq

.L3:

# Takeaways: data hazards

- More data dependencies, more likelihood of data hazards
- Stalls and data forwarding can both address data hazards to generate correct code execution results — but not very efficient
- Compiler optimizations can help, but to a limited extent
- False dependencies limits the freedom of out-of-order execution
- Register renaming + Speculative execution enables more efficient execution by dynamically scheduling instructions whenever their data dependencies are resolved
- Super scalar further improves the utilization of hardware and throughput
- Modern processors are all very wide-issue super scalar processors with OoO capabilities
- If your code cannot exploit the rich ILP on modern processors, your code cannot be efficient

#### **Announcements**

- Assignment 4 due this Saturday
  - Please reaccept the invitation again we have to scrap the original one due to permission issues
  - You may still keep your current content rename the folder on datahub to a different name, copy your answers to the newly created notebook
- Reading Quiz 7 due Tomorrow before the lecture

# Computer Science & Engineering

142



