Modern Processor Design (III): Whenever You're Ready

Hung-Wei Tseng

Recap: But A is faster!

```
d. /* one line statement using bit-wise operators */ (most efficient)
a^=b^=a^=b;
```

The order of evaluation is from right to left. This is same as in approach (c) but the three statements are compounded into one statement.

```
void regswap(int* a, int* b) {
   int temp = *a;
   *a = *b;
   *b = temp;
}
```

```
void xorswap(int* a, int* b) {
    *a ^= *b = *a = *b;
}
```

 \mathbf{m}

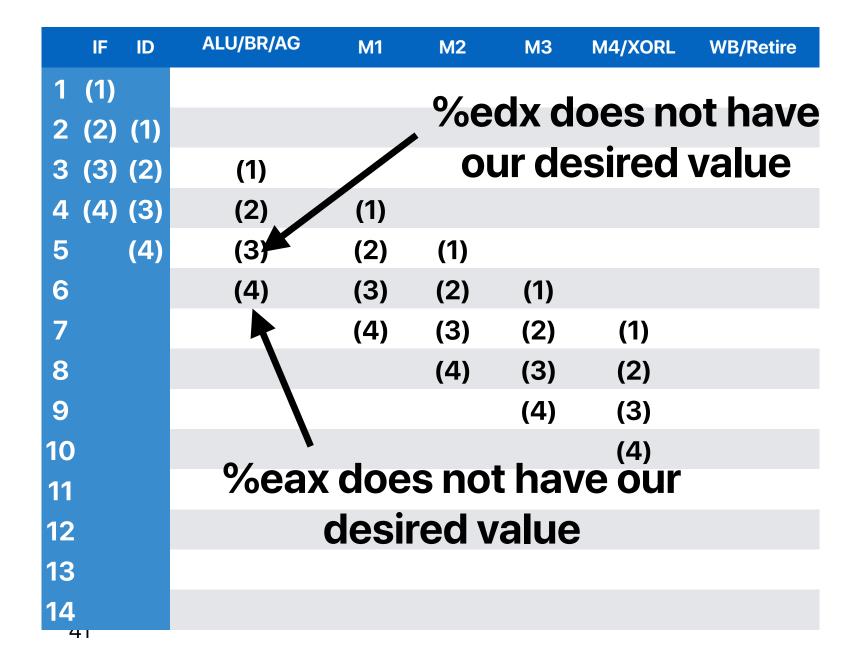
Data hazards

Data hazards

- An instruction currently in the pipeline cannot receive the "logically" correct value for execution
- Data dependencies
 - The output of an instruction is the input of a later instruction
 - May sometimes result in data hazard if the later instruction that consumes the result is still in the pipeline

Data hazards

```
① movl (%rdi), %eax
② movl (%rsi), %edx
③ movl %edx, (%rdi)
④ movl %eax, (%rsi)
```



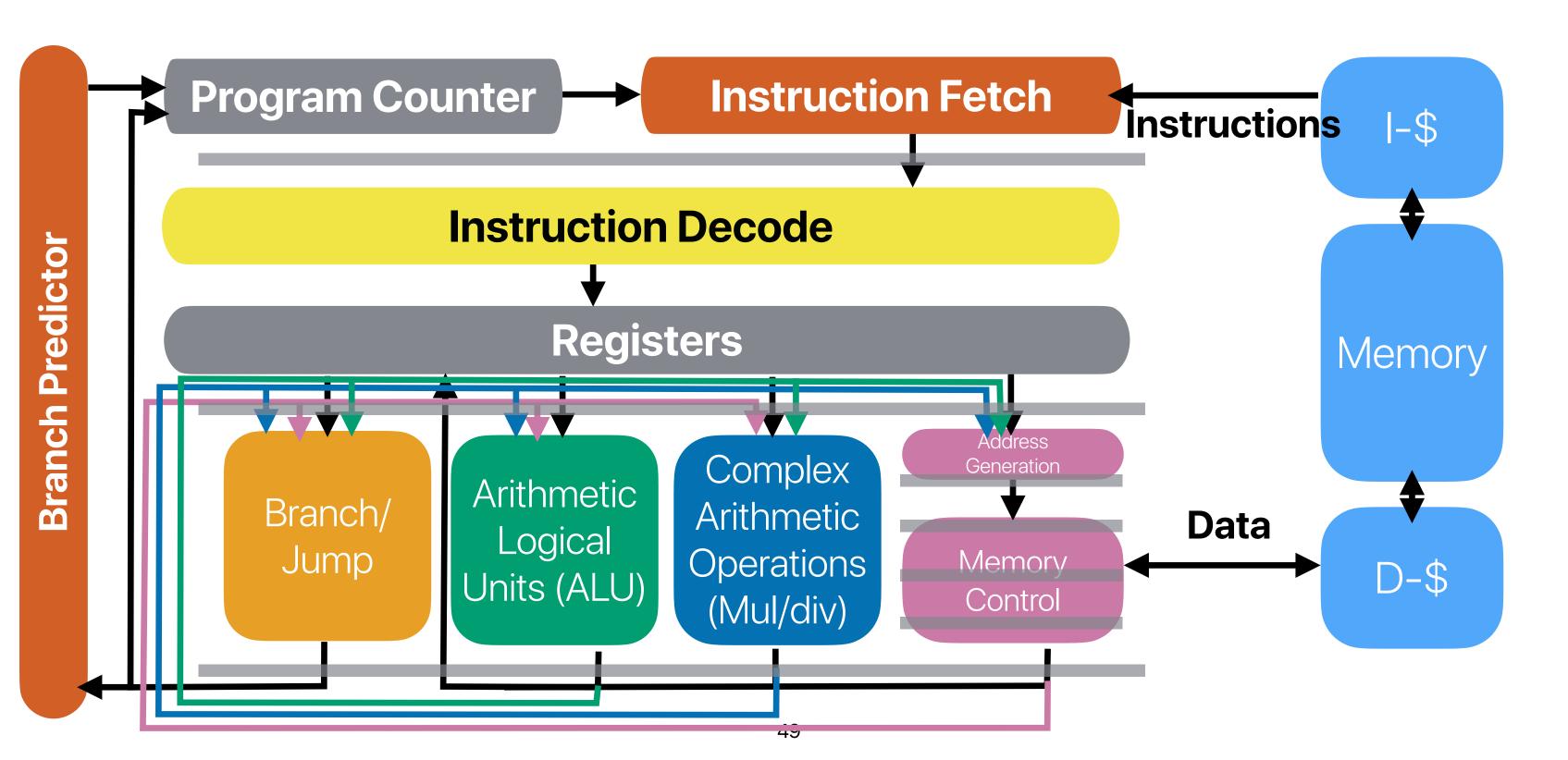
Solution 1: Let's try "stall" again

 Whenever the input is not ready when the consumer is decoding, just stall — the consumer stays at ID.

Solution 2: Data forwarding

 Add logics/wires to forward the desired values to the demanding instructions

Data "forwarding"



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

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	(7)	(6)	(5)	(4)	(3)			
12	(8)	(7)	(6)		(4)	(3)		
13	(9)	(8)	(7)			(4)	(3)	
14	(10)	(9)	(8)				(4)	(3)
15		(10)	(9)	(8)	4.00			(4)
16		(10)		(9)	(8)			(5)
17	(11)		11	cycles	s for	7 (8)	(0)	(6)
18		(10)				(9)	(8)	(7)
19		(10)		struct	LIONS		(9)	(8)
20	(12)		(10)	CPL=1	1.57			(9)
21		(12)	(11)	(10)				
22	(14)	(13)	(12)	(11)	(10)	(10)		
23		(14)	(13)	(12)	(11)	(10)	(10)	
24 57			(14)	(13)	(12)	(11)	(10)	
• .								

```
for(i = 0; i < count; i++) {

Missing opportunities

to the second of th
                                                                                                                                                                                                                                       M2
                                                                                                                                                                                                                                                        M3
                                                                                                                                                                                                                                                                      M4/XORL WB/Retire
                                                                                                                                                                                                                          M1
                     int64_t temp = a[i];
                                                                                                                                                                             (1)
                     a[i] = b[i];
                                                                                                                                    Compiler can only do this when it's 100% for sure
                     b[i] = temp;
                                                                                                                                                                 always an even number! — loop unrolling
                                                                                                         Compilers are limited by the number of registers available
                                                                                                                                                                             (5)
                                                                                                                                                                                                                                                                          (1)
                                                                                                                                                                             (6)
                                                                                                                                                                                                                                                                                             (1)
                                                                                                                                                                                                       (4)
                                                                                                                                                                                                                                                                          (2)
                                                                                                                                                                                       (5)
                                                                                .L9:
                        (%rcx,%rax), %r8
movq
                                                                                                                                                                                                       (5)
                                                                                                                                                                                                                         (4)
                                                                                                                                                                                                                                                                                             (2)
                                                                                                                  (%rcx,%rax), %r8
                                                                                         movq
                        (%rdi,%rax), %rsi
movq
                                                                                                                                                                                                                                                                                             (3)
                                                                                                                                                                                                                         (5)
                                                                                                                                                                     10 (8)
                                                                                                                                                                                                                                      (4)
                                                                                                                                                                                                       (6)
                                                                                                                 (%rdi,%rax), %rsi
                                                                                         mova
                        $8, %rax
addq
                                                                                                                                                                      11 (9) (2)
                                                                                                                                                                                                        (7)
                                                                                                                                                                                                                         (6)
                                                                                                                                                                                                                                       477 cycles for 7
                                                                                                                 $8, %rax
                                                                                          addq
                       %r8, -8(%rdi,%rax)
movq
                                                                                                                 %r8, -8(%rdi,%rax)12(10)(9)
                                                                                                                                                                                                       (8)
                                                                                                                                                                                                                         (7)
                                                                                         mova
                       %rsi, -8(%rcx,%rax)<sub>⑤</sub>
movq
                                                                                                                 %rsi, -8(%rcx,%rax13 (11) (10)
                                                                                                                                                                                                       (9)
                                                                                                                                                                                                                                                                                             (4)
                                                                                         movq
                       %r9, %rax
cmpq
                                                                                                                                                                     14 (12) (11)
                                                                                                                                                                                                      (10)
                                                                                                                 (%rcx,%rax], %r8
                                                                                                                                                                                                                                                                                             (5)
                                                                                         mova
jne
                        .L9
                                                                                                                  (%rdi,%rax], %rsi 15 (13) (12)
                                                                                                                                                                                                                        (10)
                                                                                                                                                                                                                                                                          (7)
                                                                                                                                                                                                      (11)
                                                                                                                                                                                                                                                                                             (6)
                                                                                         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
                                                                                          addq
                                                                                                                 $8, %rax
                                                                                                                                                                     18
                                                                                                                                                                                                                        (13)
                                                                                                                                                                                                                                                                                             (9)
                                                                                                                                                                                                      (14)
                                                                                                                                                                                                                                                      (11)
                                                                                                                                                                                                                                                                         (10)
                       %r8, -8(%rdi,%rax)
movq
                                                                                                                 %r8, -8(%rdi,%rax)<sub>19</sub>
                                                                                         movq
                                                                                                                                                                                                                        (14)
                                                                                                                                                                                                                                     (13)
                                                                                                                                                                                                                                                                          (11)
                                                                                                                                                                                                                                                                                            (10)
                                                                                                                                                                                                                                                      (12)
                       %rsi, -8(%rcx,%rax
movq
                                                                                                                 %rsi, -8(%rcx,%rax20
                                                                                         movq
                                                                                                                                                                                                                                                                                            (11)
                                                                                                                                                                                                                                     (14)
                                                                                                                                                                                                                                                      (13)
                                                                                                                                                                                                                                                                         (12)
                       %r9, %rax
cmpq
                                                                                                                 %r9, %rax
                                                                                          cmpq
                                                                                                                                                                                                                                                                         (13)
                                                                                                                                                                                                                                                                                            (12)
                                                                                                                                                                     21
                                                                                                                                                                                                                                                      (14)
                        .L9
jne
                                                                                          ine
                                                                                                                  .L9
                                                                                                                                                                     22
                                                                                                                                                                                                                                                                                            (13)
                                                                                                                                                                                                                                                                         (14)
                                                                                                                                                    65
                                                                                                                                                                                                                                                                                            (14)
```

Limitations of Compiler Optimizations

- If the hardware (e.g., pipeline changes), the same compiler optimization may not be that helpful
- The compiler can only optimize on static instructions, but cannot optimize dynamic instruction
 - Compiler cannot predict branches
 - Compiler does not know if cache has the data/instructions

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

iviiooii iy oppoi tui iitico													
		-			IF	ID	ALU/BR/AG	M1	M2	МЗ	M4/XORL	WB/Retire	
<pre>int64 t_tems</pre>) =	a[i]	•		1 (1)								
ofil - Dro			or o	an n	KO		ot v	Wh	at				
		G33	OI C	ан р	3 (3)	42,		VII	at				
b[i] = temp;	· ·			_	4 (4)	(3)	(2)	(1)					
shoi	ш	d ha	anna	an al	nd	4	nro	(P)	tha	2			
3110			app			\-,			(2)	(1)			
										(2)			
(%rov %rov) %ro		000		/nan	nic		4	(4)			(2)	(1)	
	1	movq					/ - >		(4)			(2)	
	2	movq			10 (8)	(/)					- f	(3)	
%r8. -8 (%rdi.%rax)	3	addq											
%rsi, -8 (%rcx,%rax	()	•						(/)	(6) ir	ist ru	ctions	(4)	
%r9, %rax									(/)				
.L9		•		•				(10)		CPI	(=)	(5)	
(%rcx,%rax), %r			•	•					(10)		(/)	(6) (7)	
(%rdi,%rax), %rsi	<u> </u>	•	_	IX						(10)		(7)	
\$8, %rax	10			,	10	(14)					(10)	(8)	
%r8, -8 (%rdi,%rax)	(11)	•	•		10		(14)					(9) (10)	
%rs1, -8 (%rcx,%rax	(12)	mova						(14)				(10) (11)	
/01 7 / /01 d X	(13)		•	•					(14)			(11) (12)	
.L9	<u>14</u>)	jne	.L9							(14)		(12)	
				68	23						(17)	(14)	
	<pre>int64_t temp a[i] = P(0) b[i] = temp; Shoot (%rcx,%rax), %r8 (%rdi,%rax), %rsi \$8, %rax %r8, -8(%rdi,%rax) %rsi, -8(%rcx,%rax) %r9, %rax .L9 (%rcx,%rax), %rsi \$8, %rax %r9, %rax .L9</pre> (%rcx,%rax), %rsi \$8, %rax %r8, -8(%rdi,%rax) \$8, %rax %r8, -8(%rdi,%rax) \$8, %rax %r8, -8(%rdi,%rax)	<pre>int64_t temp = a[i] = Proc b[i] = temp;</pre>	<pre>int64_t temp = a[i] a[i] = Process b[i] = temp;</pre>	<pre>int64_t temp = a[i]; a[i] = Processor Ca b[i] = temp;</pre>	<pre>int64_t temp = a[i]; a[i] = Processor can p b[i] = temp;</pre>	<pre>int64_t temp = a[i]; a[i] = Processor can page b[i] = temp;</pre>	int64_t temp = a[i]; a[i] = Processor can predict b[i] = temp; should happen and temp; should happen a	int64_t temp = a[i]; a[i] = Processor can predict, v b[i] = temp; should happen and unro (%rcx,%rax), %r8 (%rdi,%rax), %rsi \$8, %rax \$8, %rax \$10 (8) (7) (6) \$10 (8) (7) (6) \$10 (8) (7) (6) \$11 (9) (8) (7) \$11 (9) (8) (7) \$12 (10) (9) (8) \$13 (12) (11) \$14 (12) (11) (10) \$15 (13) (12) (11) \$15 (13) (12) (11) \$15 (14) (13) (12) \$15 (15) (15	int64_t temp = a[i]; a[i] = Processor can predict, who b[i] = temp; should happen and throw the should happen and	int64_t temp = a[i]; a[i] = Processor can predict what b[i] = temp;	int64_t temp = a[i]; a[i] = Processor can predict, what b[i] = temp;	int64_t temp = a[i]; a[i] = Processor can predict, what b[i] = temp; should happen and unroll the (%rcx,%rax), %r8 (%rdi,%rax), %rsi \$8, %rax \$\text{movq} (\text{%rdi,%rax}), \text{msi} (\$\text{\$\te	

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

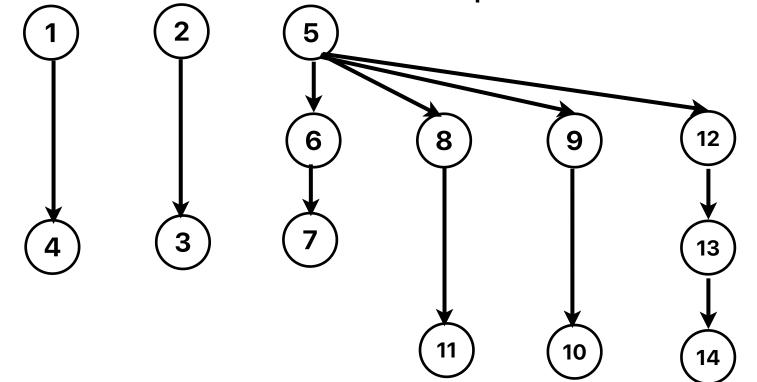
What do you need to execution an instruction?

- 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

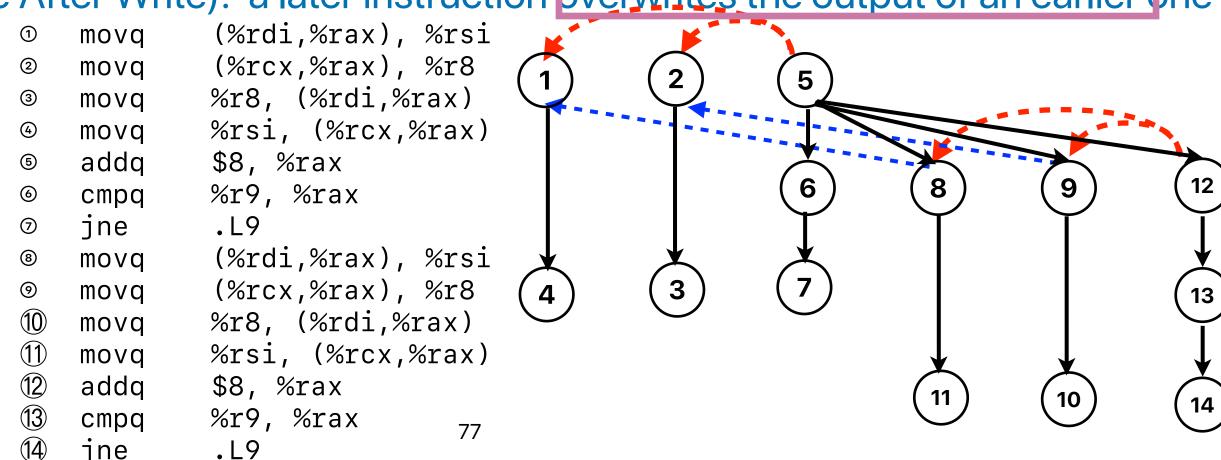
False dependencies

- 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 <u>bverwrites the source of an earlier one</u>
 - 5 and 1, 5 and 2, 12 and 8, 12 and 9

ine

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

• 8 and 1 9 and 2



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 78

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
                                         addq
                                                  $8, %rax, %t2
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

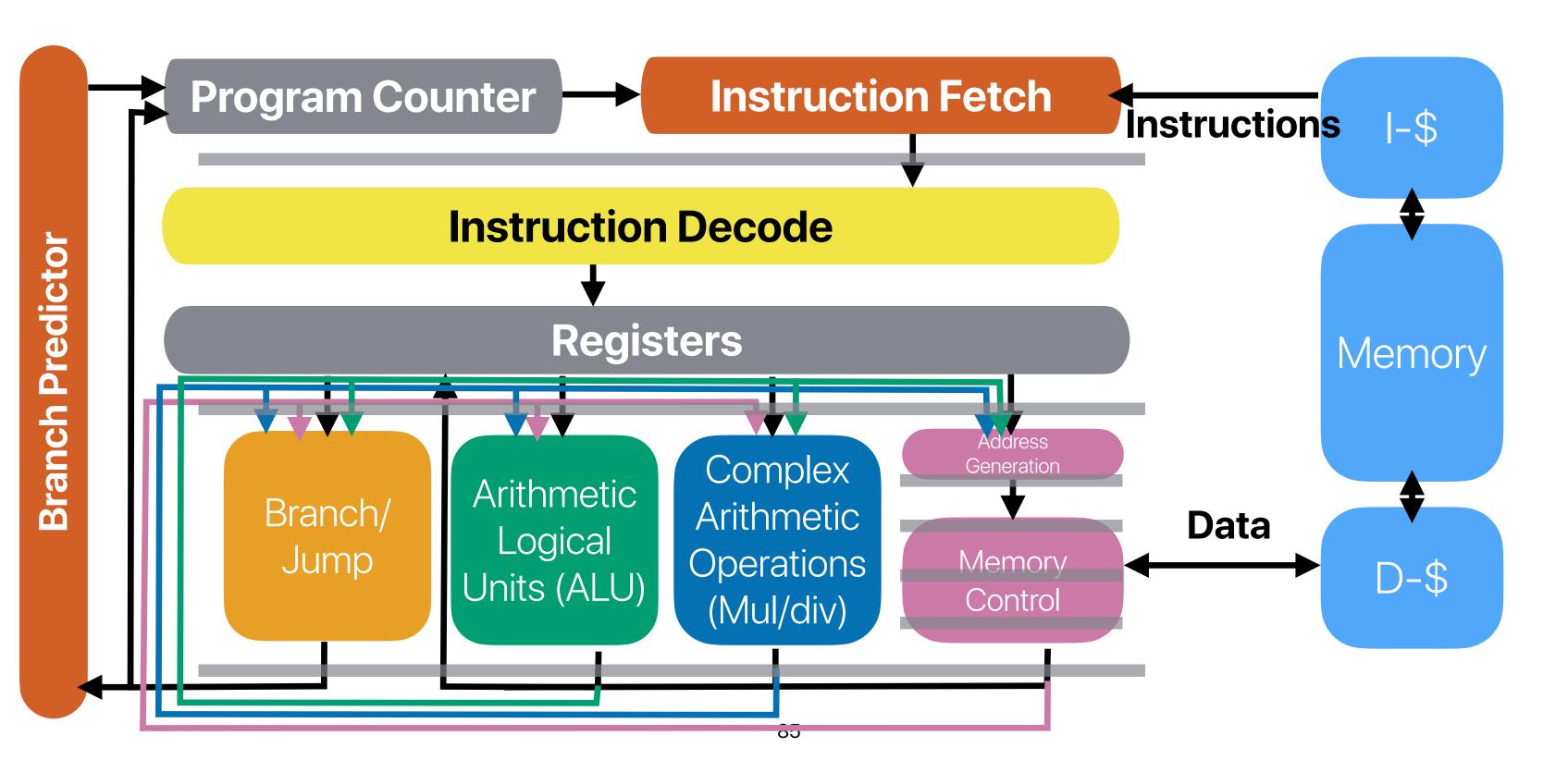
Can we really execute instructions OoO?

- 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

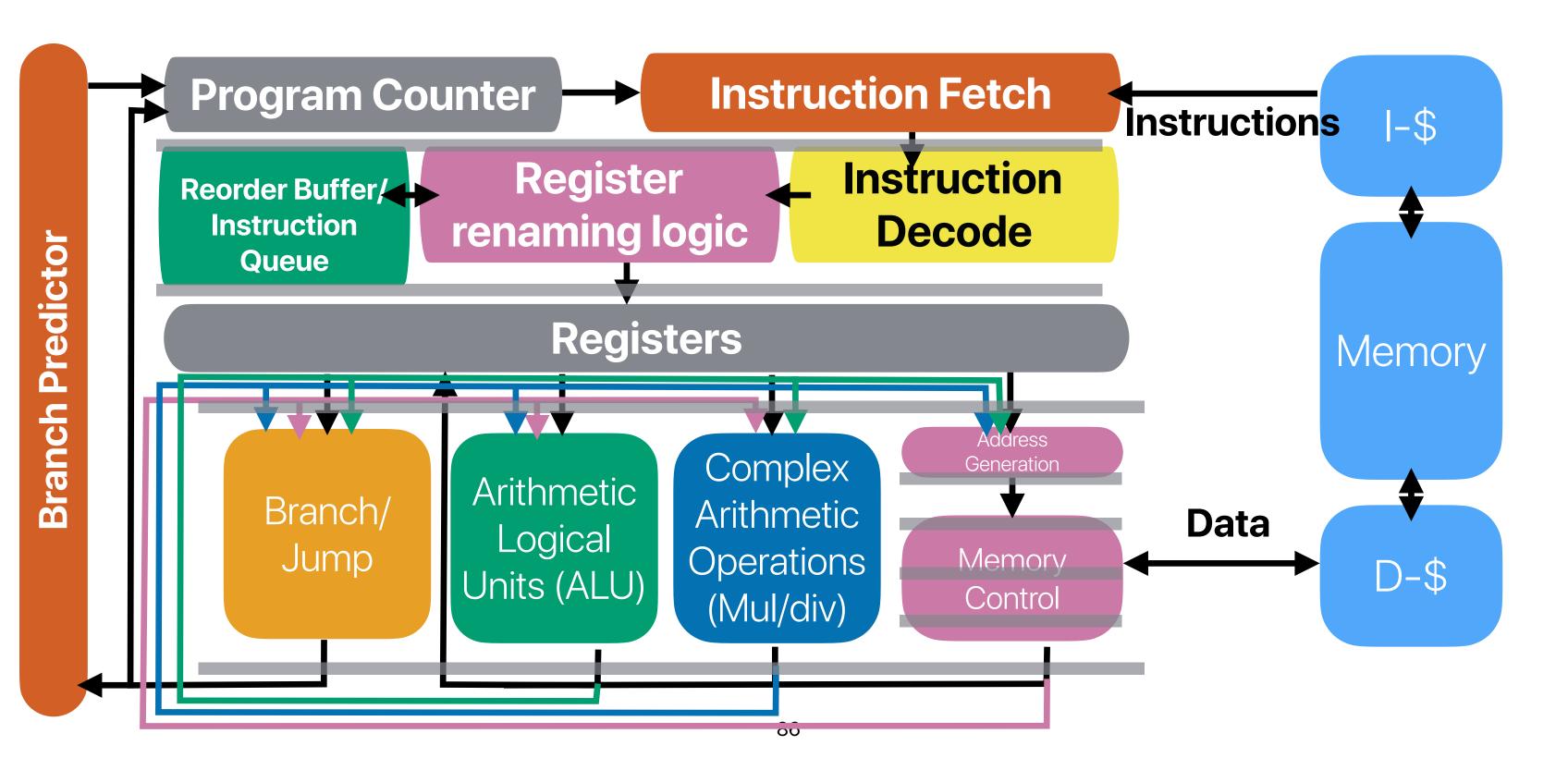
Speculative Execution

- 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	M4	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
```

- [⊙] 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			

Register renaming

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

- ② movq (%rcx,%rax), %r8 → P2
- movq %r8, (%rdi,%rax)
- movq %rsi, (%rcx,%rax)
- ⑤ addq \$8, %rax
- © cmpq %r9, %rax
- [⊙] jne .L9
- ® movq (%rdi,%rax), %rsi
- movq (%rcx,%rax), %r8
- 10 movq %r8, (%rdi,%rax)
- 11) movq %rsi, (%rcx,%rax)
- 12 addq \$8, %rax
- 13 cmpq %r9, %rax
- 14 jne .L9

Only 1 of th	em can have a	instruction	on at th	ne same cy	/cle
					•

	IF.	ID	DEN		N 44	140	MO	N 4 4	ALVI			DOD
	<u>IF</u>	ID	REN	AG	IVI	MZ	M3	M4	ALU	MUL	BR	ROB
1	(1)											
2	(2)	(1)										
3	(3)	(2)	(1)									
4	(4)	(3)	(2)	(1)								
5	(5)		(3)	(2)	(1)							
6												
7												
8												
9												
10												
11												
12												
13												
14												
15												
16												

	Physical Register
rax	
rcx	
rdi	
rsi	P1
r8	P2

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

Register renaming

```
  movq (%rdi,%rax), %rsi → P1
```

- ② movq (%rcx,%rax), %r8 → P2
- movq %r8, (%rdi,%rax)
- movq %rsi, (%rcx,%rax)
- ⑤ addq \$8, %rax
- © cmpq %r9, %rax
- [⊙] jne .L9
- ® movq (%rdi,%rax), %rsi
- movq (%rcx,%rax), %r8
- 10 movq %r8, (%rdi,%rax)
- 11) movq %rsi, (%rcx,%rax)
- 12 addq \$8, %rax
- 13 cmpq %r9, %rax
- 14 jne .L9

Only 1 of th	iem can ha ve a	instruction	on at th	ne 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												
8												
9												
10												
11												
12												
13												
14												
15												
16												

	Physical Register
rax	
rcx	
rdi	
rsi	P1
r8	P2

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

Register renaming

```
① movq (%rdi,%rax), %rsi → P1
② movq (%rax, %rax), %rsi → P2
```

- @ movq (%rcx,%rax), %r8 \rightarrow P2
- movq %r8, (%rdi,%rax)
- movq %rsi, (%rcx,%rax)
- ⑤ addq \$8, %rax
- © cmpq %r9, %rax
- [⊙] jne .L9
- ® movq (%rdi,%rax), %rsi
- movq (%rcx,%rax), %r8
- 10 movq %r8, (%rdi,%rax)
- 11) movq %rsi, (%rcx,%rax)
- 12 addq \$8, %rax
- 13 cmpq %r9, %rax
- 14 jne .L9

Only 1 of th	iem can have a	instructio	n at the	same cycl	e

				Δ								
	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												
9												
10												
11												
12												
13												
14												
15												
16												

	Physical Register
rax	
rcx	
rdi	
rsi	P1
r8	P2

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

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

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

	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												
9												
10												
11												
12												
13												
14												
15												
16												

	Physical Register
rax	Р3
rcx	
rdi	
rsi	P1
r8	P2

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

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

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

										7		
	IF	ID	REN	AG	M1	M2	M3	M4	ALU	MUL	BR	ROB
1	(1)											
2	(2)	(1)						Ir	netri	ictio	า (5) is	9
3	(3)	(2)	(1)									
4	(4)	(3)	(2)	(1)				run	ning	, ahe	ad of	(3)
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												
10												
11												
12												
13												
14												
15												
16												

	Physical Register
rax	Р3
rcx	
rdi	
rsi	P1
r8	P2

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

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

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

	IF	ID	REN	AG M1	M2	M3	M4	ALU	MUL	BR	ROB
1	(1)			nstru	icti	on	(4) is			
2	(2)	(1)		ning	ıak	102	nd c	of (3)	\ Ins	struct	ion (5) is
3	(3)	(2)	(1)	9			id C	71 (5			
4	(4)	(3)	(2)	(1)					runr	jing a	head of (3
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)				(1)(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 → P2
    movq %r8, (%rdi,%rax)
    movq %rsi, (%rcx,%rax)
    addq $8, %rax → P3
    cmpq %r9, %rax
    jne .L9
```

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

	IF	ID	REN	AG	M1	M2	М3	M4	ALU	MUL	BR	ROB
1	(1)											
2	(2)	(1)										
3	(3)	(2)	(1)					R	atire	/Com	mit (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)				(1)(5)
10												
11												
12												
13												

	Physical Register
rax	Р3
rcx	
rdi	
rsi	P1
r8	P2

14

15

16

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

- ® movq (%rdi,%rax), %rsi → P4
- movq (%rcx,%rax), %r8
- 10 movq %r8, (%rdi,%rax)
- ① movq %rsi, (%rcx,%rax)
- 12 addq \$8, %rax

jne .L9

- 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	(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)				(1)(5)
10	(10)	(9)	(6)(7)(8)	(3)	(4)							(2)(5)
11												
12												
13												
14												
15												

	Physical Register
rax	Р3
rcx	
rdi	
rsi	P4
r8	P2

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			

(7)(8)(9)

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

- ® movq (%rdi,%rax), %rsi → P4

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

	IF	ID	REN	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)	(7)	(3)(4)(6)				(2)	(1)	(5)			
9	(9)	(8)	(3)(6)(7)	(4)				(2)				(1)(5)
10	(10)	(9)	(6)(7)(8)	(3)	(4)							(2)(5)

(6)

(3) (4)

	Physical Register
rax	P3
rcx	
rdi	
rsi	P4
r8	P5

11 (11) (10)

12

13

14

15

16

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

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

- ® movq (%rdi,%rax), %rsi → P4

 ® movq (%rcx,%rax), %r8 → P5
- 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	М3	M4	ALU	MUL	BR	ROB
•		(1)											
2	2 ((2)	(1)										
3	3 ((3)	(2)	(1)									
4	1 (4)	(3)	(2)	(1)								
Ę	5 ((5)	(4)	(3)	(2)	(1)							
6	6 ((6)	(5)	(3)(4)		(2)	(1)						
	7 ((7)	(6)	(3)(4)(5)			(2)	(1)					
8	3 (8)	(7)	(3)(4)(6)				(2)	(1)	(5)			
Ş	9 (9)	(8)	(3)(6)(7)	(4)				(2)				(1)(5)
1	0 (10)	(9)	(6)(7)(8)	(3)	(4)							-(2)(5)
1	1 (11)	(10)	(7)(8)(9)		(3)	(4)			(6)			
1	2 (12)	(11)	(8)(9)(10)			(3)	(4)				(7)	(5)(6)
1	3												
1	4												
1	5												
1	6												

	Physical Register
rax	Р3
rcx	
rdi	
rsi	P4
r8	P5

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

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

- ® movq (%rdi,%rax), %rsi → P4

 ® movq (%rcx,%rax), %r8 → P5

 10 movq %r8, (%rdi,%rax)
- 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	(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)				(1)(5)
10	(10)	(9)	(6)(7)(8)	(3)	(4)							(2)(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												
15												

Physical Register
P3
P4
P5

16

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

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

jne .L9

	IF	ID	REN	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)				(1)(5)
10	(10)	(9)	(6)(7)(8)	(3)	(4)							-(2)(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												
16												

Physical Register
P6
P4
P5

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

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

jne .L9

	IF	ID	REN	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)				(1)(5)
10	(10)	(9)	(6)(7)(8)	(3)	(4)							-(2)(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												

Physical Register
P6
P4
P5

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

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

	IF	ID	REN	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)				(1)(5)
10	(10)	(9)	(6)(7)(8)	(3)	(4)							-(2)(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)

	Physical Register
rax	P6
rcx	
rdi	
rsi	P4
r8	P5

	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			

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

			Only 1 of	tne	m c	an	nav	ea	Instru	iction a	t the	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	(7)	(6)	(3)(4)(5)			(2)	(1)					
8	(8)	(7)	(3)(4)(6)				(2)	(1)	(5)			
9	(9)	(8)	(3)(6)(7)	(4)				(2)				-(1)(5)
10	(10)	(9)	(6)(7)(8)	(3)	(4)							(2)(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												
19												
20												
21												
22												
00				A .								

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

			Only 1 of	tne	m c	an	nav	ea	Instru	iction 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	(7)	(6)	(3)(4)(5)			(2)	(1)					
8	(8)	(7)	(3)(4)(6)				(2)	(1)	(5)			
9	(9)	(8)	(3)(6)(7)	(4)				(2)				-(1)(5)
10	(10)	(9)	(6)(7)(8)	(3)	(4)							(2)(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)				(8)(12)(13)(14)
19												
20												
21												
22												
22												

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

			Only 1 of	tne	m c	an I	nav	ea	Instru	iction a	ttne	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	(7)	(6)	(3)(4)(5)			(2)	(1)					
8	(8)	(7)	(3)(4)(6)				(2)	(1)	(5)			
9	(9)	(8)	(3)(6)(7)	(4)				(2)				-(1)(5)
10	(10)	(9)	(6)(7)(8)	(3)	(4)							-(2) (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)				(8) (12)(13)(14)
19	(19)	(18)	(15)(16)(17)	(10)	(11)							(9)(12)(13)(14)
20												
21												
22												
22												

		•	
1	movq	(%rdi,%rax), %rsi →	P1
2	movq	(%rcx,%rax), %r8 →	P2
3	movq	%r8, (%rdi,%rax)	
4	movq	%rsi, (%rcx,%rax)	
5	addq	\$8, %rax →	P3
6	cmpq	%r9, %rax	
7	jne	.L9	
8	movq	(%rdi,%rax), %rsi →	P4
9	movq	(%rcx,%rax), %r8 →	P5
10	movq	%r8, (%rdi,%rax)	
1	movq	%rsi, (%rcx,%rax)	
12	addq	\$8, %rax →	P6
13	cmpq	%r9, %rax	
14	jne	.L9	
15	movq	(%rdi,%rax), %rsi	
16	movq	(%rcx,%rax), %r8	
17)	movq	%r8, (%rdi,%rax)	
18	movq	%rsi, (%rcx,%rax)	
19	addq	\$8, %rax	
20	cmpq	%r9, %rax	
21)	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	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)				-(1)(5)
10	(10)	(9)	(6)(7)(8)	(3)	(4)							(2)(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)				(8)(12)(13)(14)
19	(19)	(18)	(15)(16)(17)	(10)	(11)							(9)(12)(13)(14)
20	(20)	(19)	(16) (17) (18)	(15)	(10)	(11)						(12)(13)(14)
21												
22												

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

			Only 1 of	the	m c	an	hav	ea	instru	uction a	t the	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	(7)	(6)	(3)(4)(5)			(2)	(1)					
8	(8)	(7)	(3)(4)(6)				(2)	(1)	(5)			
9	(9)	(8)	(3)(6)(7)	(4)				(2)				-(1)(5)
10	(10)	(9)	(6)(7)(8)	(3)	(4)							-(2)(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)				(8)(12)(13)(14)
19	(19)	(18)	(15) (16) (17)	(10)	(11)							(9)(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												

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

			Only 1 of	the	m c	an	hav	ea	instr	uction at	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	(7)	(6)	(3)(4)(5)			(2)	(1)					
8	(8)	(7)	(3)(4)(6)				(2)	(1)	(5)			
9	(9)	(8)	(3)(6)(7)	(4)				(2)				-(1)(5)
10	(10)	(9)	(6)(7)(8)	(3)	(4)							-(2)(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)				(8)(12)(13)(14)
19	(19)	(18)	(15) (16) (17)	(10)	(11)							(9) (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)
00												

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

				•				111 C		liav	С а				
			Ke		F ID	REN	AG	M1	M2	М3	M4	ALU	MUL	ЬR	ROB
1	mova	(%rdi,%rax), %rsi		1 (1)										
2		(%rcx,%rax), %r8			2) (1)										
3	•	%r8, (%rdi,%rax)	<i>,</i> , ,	3 ((4)								
	-	· · · · · · · · · · · · · · · · · · ·		4 ((1)	141							
4	•	%rsi, (%rcx,%rax)	N DO	5 ((2)	(1)							
5	addq	\$8, %rax	→ P3	6 ((2)	(1)	(4)					
6	cmpq	%r9, %rax			7) (6				(2)		(4)	(C)			
7	jne	.L9			8) (7 ₎		(4)			(2)		(5)			(4) (5)
8	mova	(%rdi,%rax), %rsi	→ P4	•	9) (8		(4)	(4)			(2)				(1)(5)
9	-	(%rcx,%rax), %r8			(0) (9 (1) (10		(3)	(4)				(6)			-(2)(5)
_	•	·	<i>,</i> L2					(3)	(4)	(4)		(6)		(7)	(E)(G)
10	•	%r8, (%rdi,%rax)		_	2) (11 2) (12		(0)		(3)	(4)	(4)			(7)	(5)(6) (5)(6)(7)
$\underbrace{11}_{\bigcirc}$	•	%rsi, (%rcx,%rax)			(12) (13)		(8)	(0)		(3)					(5)(6)(7)
(12)	addq	\$8, %rax	→ P6		4) (13 5) (1 <i>4</i>		(9)	(8)			(3)	(12)			(4)(5)(6)(7)
13	cmpq	%r9, %rax			5) (14 6) (15			(9)	(8)	(9)		(12)			(12)
14	jne	.L9			17) (16				(9)	(8) (9)	(8)	(13)		(14)	(12)(13)
15	_	(%rdi,%rax), %rsi			8) (17		(11)			(3)	(9)			(14)	(12)(13) (8)(12)(13)(14)
16	_	(%rcx,%rax), %r8			9) (18			(11)			(5)				(9)(12)(13)(14)
	•	•			20) (19				(11)						(12)(13)(14)
17	•	%r8, (%rdi,%rax)			21) (20				(10)						(12)(13)(14)
18	movq	%rsi, (%rcx,%rax)		22	(21		(10)				(11)	(19)			(12)(13)(14)
19	addq	\$8, %rax		23		(17)(20)(21)	(18)			(15)		(10)			(11)(12)(13)(14)(19)
20	cmpq	%r9, %rax		24			(10)		()	(:-)	(· •)				,
21)	jne	.L9		25											

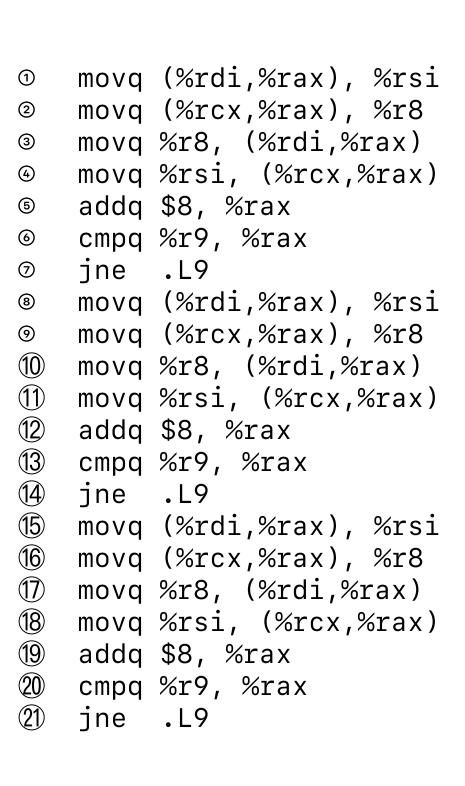
Only 1 of them can have a instruction at the same cycle

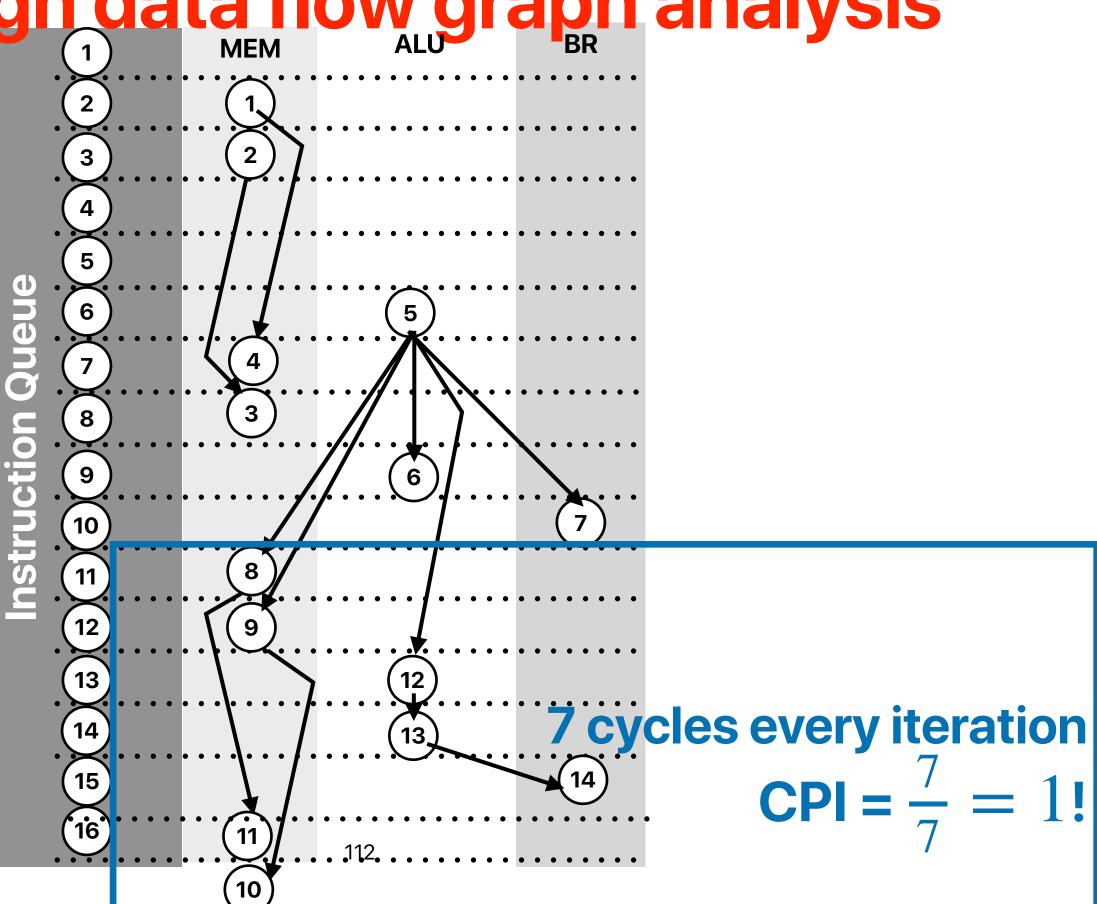
							· · · ·	•		-				
		Ke		F ID	REN	AG	M1	M2	МЗ	M4	ALU	MUL	ЬR	ROB
1	movq (%rdi,%rax), %rsi-	• D1	1 (1											
2	movq (%rcx,%rax), %r8 -		2 (2											
3	movq (%16X,%1dX), %10 movq %r8, (%rdi,%rax)	<i>,</i> , ,	3 (3		(1)	(4)								
	•		4 (4		(2)	(1)	(4)							
4	movq %rsi, (%rcx,%rax)	N D2	5 (5		(3)	(2)	(1)	(4)						
5		→ P3	6 (6		(3)(4)		(2)		(1)					
6	cmpq %r9, %rax		7 (7 8 (8		(3)(4)(5)			(2)		(1)	(5)			
7	jne .L9		9 (9		(3)(4)(6) (3)(6)(7)	(4)			(2)	(2)	(5)			-(1)(5)
8	<pre>movq (%rdi,%rax), %rsi</pre>	> P4		0) (9)	(6)(7)(8)		(4)			(2)				(1) (5) -(2)(5)
9	movq (%rcx,%rax), %r8 -	→ P5		1) (10)		(0)		(4)			(6)			(2)(0)
10	movq %r8, (%rdi,%rax)			2) (11)	(8)(9)(10)		(-)		(4)		(0)		(7)	(5)(6)
11	movq %rsi, (%rcx,%rax)			3) (12)		(8)			(3)	(4)			,	(5)(6)(7)
12		> P6		4) (13)			(8)			(3)				(4)(5)(6)(7)
_	• •	7 PO	15 (1	5) (14)	(10)(11)(13)		(9)	(8)			(12)			(2)(4)(5)(6)(7)
13	cmpq %r9, %rax		16 (1	6) (15)	(10)(11)(14)			(9)	(8)		(13)			(12)
_	jne .L9		17 (1	7) (16)	(10)(11)(15)				(9)	(8)			(14)	(12)(13)
15	movq (%rdi,%rax), %rsi		18 (1	8) (17)	(10)(15)(16)	(11)				(9)				(8) (12)(13)(14)
16	movq (%rcx,%rax), %r8		19 (1	9) (18)	(15)(16)(17)	(10)	(11)							(9)(12)(13)(14)
17)	<pre>movq %r8, (%rdi,%rax)</pre>		20 (2	0) (19)		(15)	(10)	(11)						(12)(13)(14)
18	movq %rsi, (%rcx,%rax)			1) (20)		(16)		(10)						(12)(13)(14)
19	addq \$8, %rax		22	(21)	(17)(18)(20)	4.5.5					(19)			(12)(13)(14)
20	cmpq %r9, %rax		23		(17)(20)(21)	(18)			(15)					(11)(12)(13)(14)(19)
_			24		(20)(21)	(17)	(18)		(16)	(15)				(10)(11)(12)(13)(14)(19)
(21)	ine .L9		25											

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

				•	_	Only 1 of	tne	m c	an i	nav	e a		ion at ti	1e s L	same cycle
			Rec	IF	ID	REN	AG	M1	M2	M3	M4	ALU N	MUL B	R	ROB
1	movq	(%rdi,%rax), %rsi	→ P1	1 (1)											
2		(%rcx,%rax), %r8		2 (2) 3 (3)	(1) (2)	(1)									
3	movq	%r8, (%rdi,%rax)		4 (4)	(3)	(2)	(1)								
4	movq	%rsi, (%rcx,%rax)		5 (5)	(4)	(3)		(1)							
(5)	addq	\$8, %rax	P3	6 (6)	(5)	(3)(4)		(2)	(1)						
6	cmpq	%r9, %rax		7 (7)	(6)	(3)(4)(5)			(2)						
7	jne	.L9		8 (8)	(7)	(3)(4)(6)				(2)	(1)	(5)			44.45
8		(%rdi,%rax), %rsi	P4	9 (9)	(8)	(3)(6)(7)	(4)	(4)			(2)				-(1)(5)
9	•	(%rcx,%rax), %r8		10 (10)		(6)(7)(8)	(3)	(4)	(4)			(0)			-(2)(5)
_	-	-	PJ	11 (11)		(7)(8)(9)		(3)	(4)	(4)		(6)	<i>(</i> -	7 \	(E)(G)
10	•	%r8, (%rdi,%rax)		12 (12)		(8)(9)(10)	(0)		(3)		(4)			7)	(5)(6) (5)(6)(7)
$\underbrace{11}_{\widehat{\Omega}}$	•	%rsi, (%rcx,%rax)		13 (13)		(9)(10)(11)	(8)	(0)		(3)					(5)(6)(7)
(12)	addq	\$8, %rax	P6	14 (14)		(10)(11)(12)	(9)	(8)	(0)		(3)	(12)			(4)(5)(6)(7)
13	cmpq	%r9, %rax		15 (15)		(10)(11)(13)		(9)	(8)	(0)		(12)			(0)(-)(0)(0)(/)
14	jne	.L9		16 (16) 17 (17)		(10)(11)(14) (10)(11)(15)			(9)		(8)	(13)	(1	/ 1)	(12) (12)(13)
15	_	(%rdi,%rax), %rsi		18 (18)		(10)(11)(13)	(11)				(8)			4)	(12)(13) (12)(13)(14)
_	•	(%rcx,%rax), %r8		19 (19)			(11) (10)				(3)	7 cycle	es tor		(9)(12)(13)(14)
16	•			20 (20)					(11)			instru	ctions	3	(12)(13)(14)
<u>17</u>	•	%r8, (%rdi,%rax)		21 (21)											(12)(13)(14)
18	movq	%rsi, (%rcx,%rax)		22	(21)	(17)(18)(20)	(10)	(16)	(15)	(10)	(11)	(19) ^{CP}	I = 1		(12)(13)(14)
19	addq	\$8, %rax		23	(21)		(18)	(10)	(16)	(15)	(10)	(10)			(11)(12)(13)(14)(19)
20	cmpq	%r9, %rax		24		(20)(21)		(18)			(15)				(10)(11)(12)(13)(14)(19)
<u>21</u>	jne	.L9		25		(21)	()		(18)			(20)			(15)(19)

Through data flow graph analysis



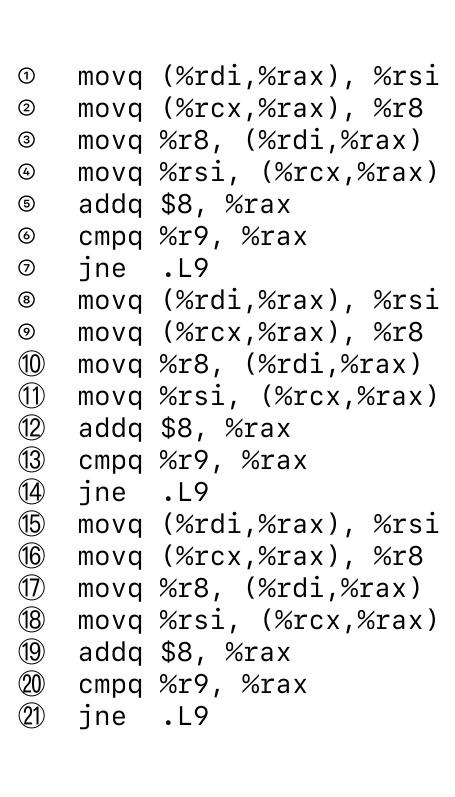


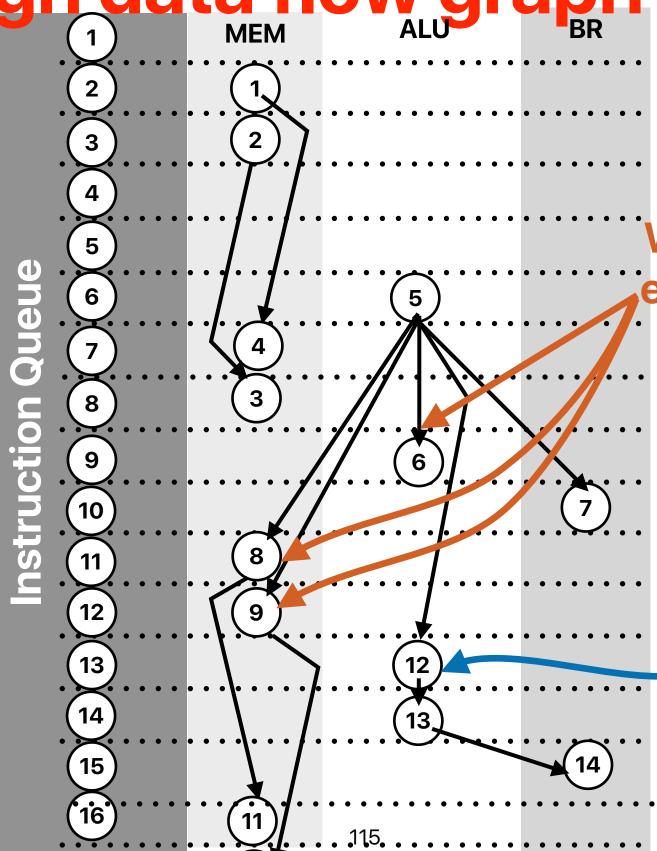
Takeaways: data hazards

- 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

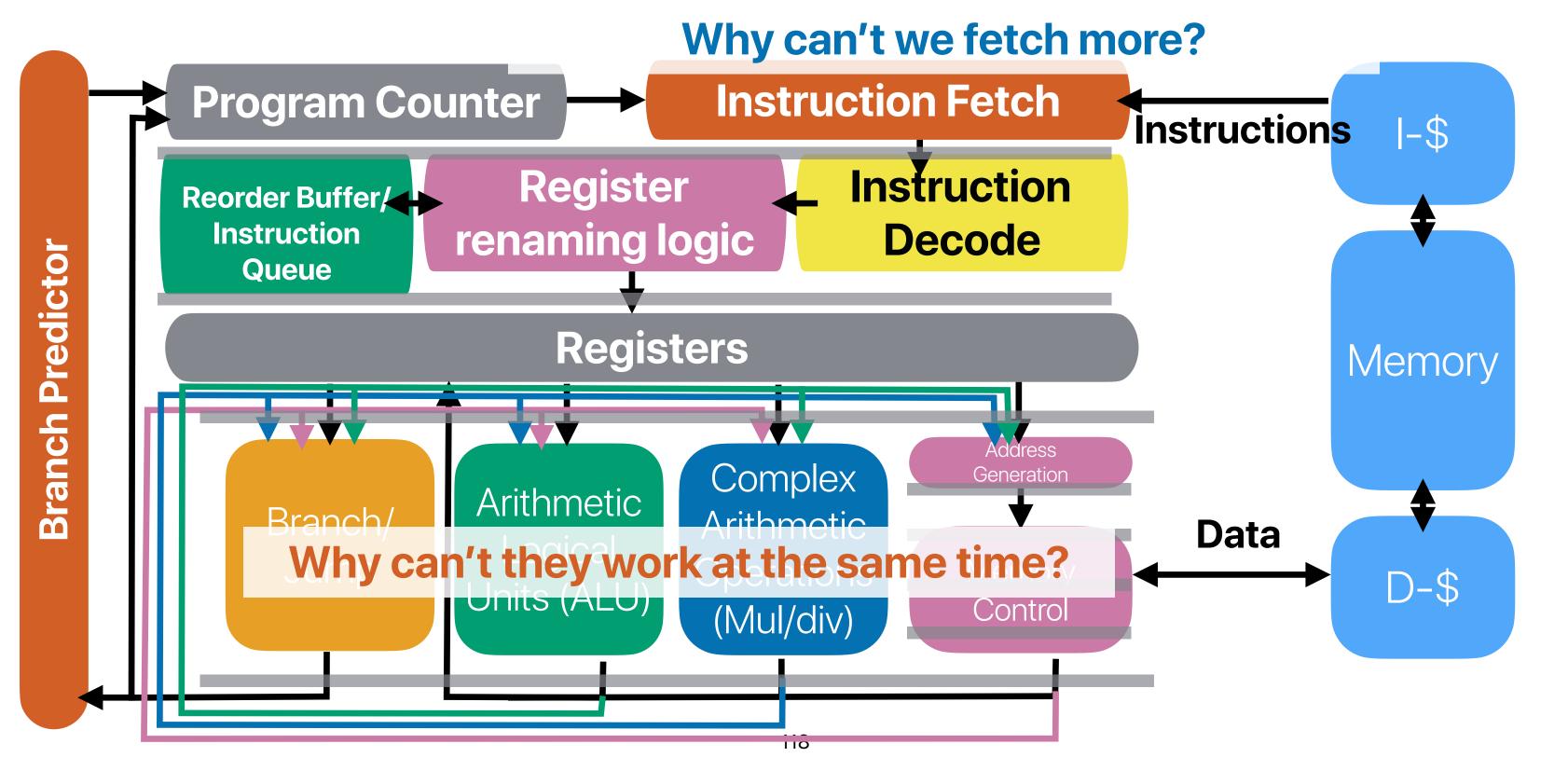
Superscalar

- 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

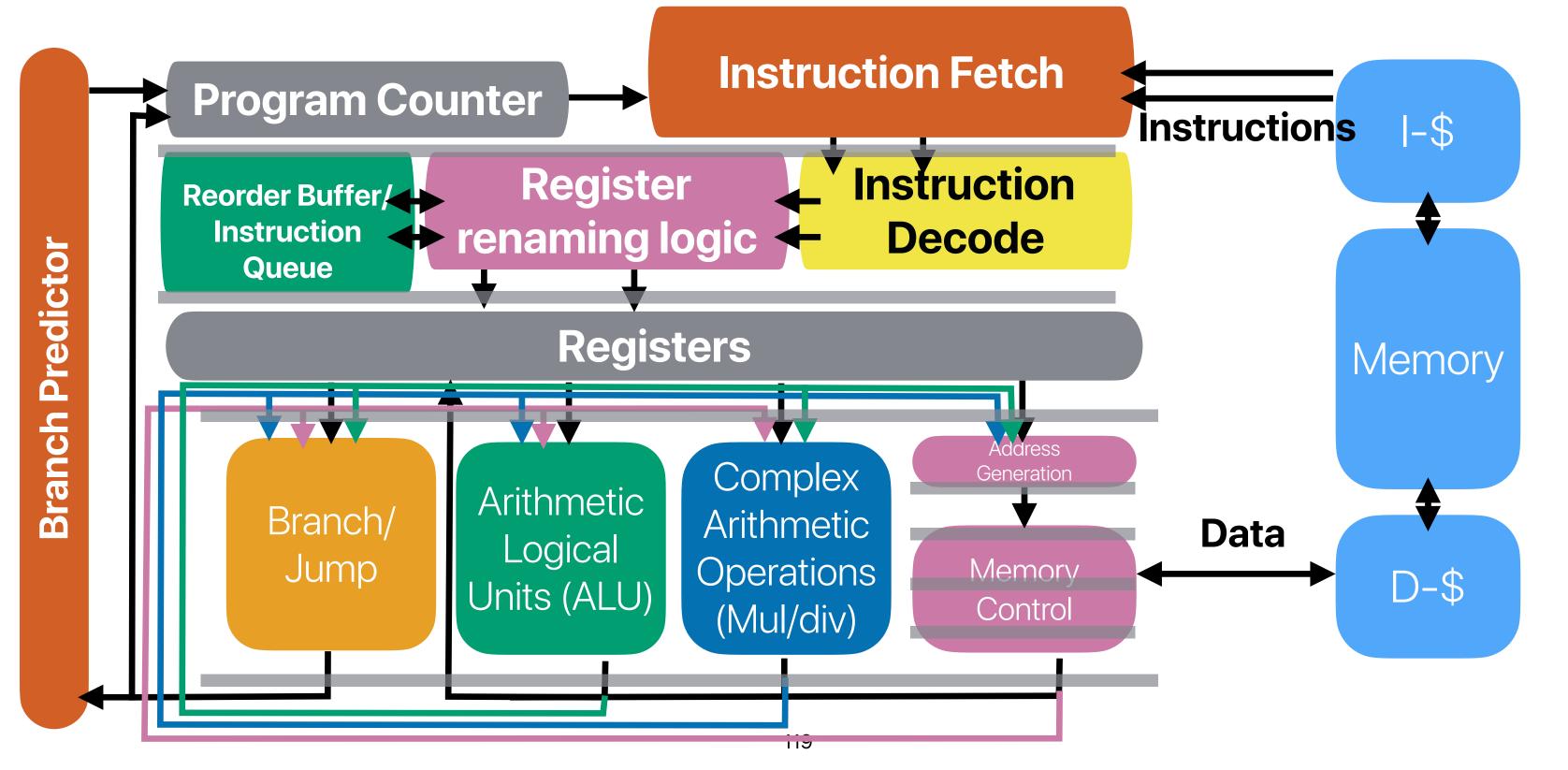
1

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	ınav	eal	nstruc	uon a	t the same cycle
2	movq	(%rcx,%rax), %r8	→ P2	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)	. 5		(3)(4)											
(5)	addq	\$8, %rax	→ P3	3	(5)(6)	(3)(4)	(1)(2)									
6	cmpq	%r9, %rax		4												
7	jne			5												
8	movq	(%rdi,%rax), %rsi	$\rightarrow P^{\prime}$	H 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												
(21)	ine	. L 9					120									

1	movq	(%rdi,%rax), %rsi	→ P1			_	issue.	4	or tr	EIII	Gai	IIIav	ear	ristique	tion	l lile Saili	e Cycle
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)												
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	6													
9	movq	(%rcx,%rax), %r8	→ P5	9													
_		%r8, (%rdi,%rax)		9													
11	movq	%rsi, (%rcx,%rax)		10													
12	addq	\$8, %rax	→ 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													
19	addq	\$8, %rax		18 19													
20	cmpq	%r9, %rax		20													
21)	jne	.L9		20			121										

1	movq	(%rdi,%rax), %rsi	→ P;	L		_	10040.	1			Odi	iiiav				the dame by	
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)												
(5)	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	5 11													
13	cmpq	%r9, %rax		12													
14	jne	.L9		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													
19	addq	\$8, %rax		18 19													
		%r9, %rax		20													
_	ine			20			122										

1	movq	(%rdi,%rax), %rsi	• P1				. 155uc.	4	OI LI	HeIII	Cai	IIIav	E A I	iistiuc	tion at t	ne same cycle
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		(5)(6)											
6	cmpq	%r9, %rax			(7)(8)			(1)								
7	jne				(9)(10)				(1)	(4)						
8	movq	(%rdi,%rax), %rsi	P4	7	(11)(12)	(9)(10)	(3)(4)((6)(7)(8)		(2)	(1)						
9	movq	(%rcx,%rax), %r8	P5	8												
10	movq	%r8, (%rdi,%rax)		9												
11	movq	%rsi, (%rcx,%rax)		10												
12	addq	\$8, %rax	• P6	11												
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												
21)	jne	.L9		20			123									

1	movq	(%rdi,%rax), %rsi	→ P1				. 155ue.	4	or u	ICIII	Cai	IIIav	E d I	ristitut	tion at	the Same Cych
2	movq	(%rcx,%rax), %r8	→ 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													
6	cmpq	%r9, %rax			(7)(8)			(1)								
7	jne	.L9			(9)(10)			(2)								
8	movq	(%rdi,%rax), %rsi	→ P4				(3)(4)((6)(7)(8)		(2)		(4)		(C)			(E)
9	movq	(%rcx,%rax), %r8	→ P5	0	(13)(14)	(11)(12)	(3)(4)((6)(7)(9) (10)	(8)		(2)	(1)		(5)			(5)
		%r8, (%rdi,%rax)		9												
_	-	%rsi, (%rcx,%rax)		10												
12	addq	\$8, %rax	→ P6													
_	-	%r9, %rax		12												
14	jne	.L9		13												
15	movq	(%rdi,%rax), %rsi		14												
16	movq	(%rcx,%rax), %r8		15												
_	-	%r8, (%rdi,%rax)		16												
	' - '	%rsi, (%rcx,%rax)		17												
_		\$8, %rax		18												
	=	%r9, %rax		19												
	ine			20			124									

1	movq	(%rdi,%rax), %rsi	P1				. 10040.	1			Cai	iiiav				ne same cy	
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	P 3														
6	cmpq	%r9, %rax					(2)(3)(4)	(1)									
7						(7)(8)		(2)		141							
8	movq	(%rdi,%rax), %rsi	▶ P4	_			(3)(4)((6)(7)(8)		(2)		(4)		(E)			(E)	
9	movq	(%rcx,%rax), %r8	→ P5				(3)(4)((6)(7)(9) (10) (3)(4)(7)(10)(11) (12)				(1)	(1)	(5) (6)			(5) (5)(6)	
		%r8, (%rdi,%rax)		9	(10)(10)	(10)(14)	(12)	(9)	(0)		(2)	(1)	(6)			(3)(0)	
11	movq	%rsi, (%rcx,%rax)		10													
12	addq	\$8, %rax	> P6														
13	cmpq	%r9, %rax		12													
_	jne			13													
15	movq	(%rdi,%rax), %rsi		14													
_	_	(%rcx,%rax), %r8		15													
17)	movq	%r8, (%rdi,%rax)		16													
	-	%rsi, (%rcx,%rax)		17													
	-	\$8, %rax		18													
		%r9, %rax		19													
	ine			20			125										

1	movq	(%rdi,%rax), %rsi	• P1				2 issue: "	4	ot tr	nem	car	nav	еат	nstruc	tion at	tne same cycle
2	movq	(%rcx,%rax), %r8	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		(5)(6)											
6	cmpq	%r9, %rax			(7)(8)			(1)								
7	jne	.L9			(9)(10)				(1)							
8	movq	(%rdi,%rax), %rsi	P4				(3)(4)((6)(7)(8)		(2)		(4)		(E)			(E)
9	movq	(%rcx,%rax), %r8	P5		(13)(14)		(3)(4)((6)(7)(9) (10) (3)(4)(7)(10)(11) (12)			(2)	(1) (2)	(1)	(5) (6)			(5) (5)(6)
		%r8, (%rdi,%rax)					(12) (3)(10)(11)(12) (13)(14)		(9)	(8)	(2)	(1) (2)	(6)		(7)	(1)(5)(6)
11	movq	%rsi, (%rcx,%rax)		10	(==)(==)	(10)(10)	(13)(14)	()	(0)	(0)		(-)			(7)	(1)(0)(0)
12	addq	\$8, %rax	P6													
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												
20	cmpq	%r9, %rax		19 20												
_	ine			20			126									

1	movq	(%rdi,%rax), %rsi	→ P1					7								
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													
6	cmpq	%r9, %rax			(7)(8)			(1)								
7	jne	.L9			(9)(10)			(2)		448						
8	movq	(%rdi,%rax), %rsi	→ P4				(3)(4)((6)(7)(8)		(2)		(4)		(5)			(E)
9	movq	(%rcx,%rax), %r8	→ P5				(3)(4)((6)(7)(9) (10) (3)(4)(7)(10)(11)			(2)		(1)	(5) (6)			(5) (5)(6)
		%r8, (%rdi,%rax)					(3)(4)(7)(10)(11) (12) (3)(10)(11)(12) (13)(14)		(9)	(8)	(2)	(2)	(6)		(7)	(5)(6) (1)(5)(6)
_	•	%rsi, (%rcx,%rax)					(13)(14) (10)(11)(13)(14) (15)(16)			(9)	(8)	(2)	(12)		(7)	(2)(5)(6)(7)
_	•		→ P6	11	(2/(2/	(/(- /	(15)(16)	(0)	()	(0)	(0)		(12)			(2)(3)(3)(7)
	-	%r9, %rax		12												
_	jne	-		13												
_	_	(%rdi,%rax), %rsi		14												
_	-	(%rcx,%rax), %r8		15												
		%r8, (%rdi,%rax)		16												
_	-	%rsi, (%rcx,%rax)		17												
		\$8, %rax		18												
		%r9, %rax		19												
21	jne			20			127									

1	movq	(%rdi,%rax), %rsi	→ P	1			issue: "	4	ot tr	nem	car	n nav	еат	nstruc	tion at i	tne same cycle
2	movq	(%rcx,%rax), %r8	→ 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)											
5	addq	\$8, %rax	→ P		(5)(6)											
6	cmpq	%r9, %rax			(7)(8)			(1)								
7	jne		_		(9)(10)			(2)		(4)						
8	movq	(%rdi,%rax), %rsi	→ P				(3)(4)((6)(7)(8)		(2)		(4)		(E)			(E)
9	movq	(%rcx,%rax), %r8	\rightarrow P	5 ((3)(4)((6)(7)(9) (10) (3)(4)(7)(10)(11) (12)			(2)		(1)	(5) (6)			(5) (5)(6)
10		%r8, (%rdi,%rax)					(12) (3)(10)(11)(12) (13)(14)			(8)	(2)	(2)	(6)		(7)	(3)(6)
11	movq	%rsi, (%rcx,%rax)					(13)(14) (10)(11)(13)(14) (15)(16)			(9)	(8)	(-)	(12)		(*)	(2)(5)(6)(7)
12	addq	\$8, %rax	\rightarrow P	6 11	(21)(22)	(19)(20)	(15)(16) (10)(11)(14)(16) (17)(18)	(15)				(8)	(13)			(5)(6)(7)(12)
13	cmpq	%r9, %rax		12			(1/)(10)									
14	jne	.L9		13	3											
15)	movq	(%rdi,%rax), %rsi		14	ļ											
16	movq	(%rcx,%rax), %r8		15												
_	-	%r8, (%rdi,%rax)		16												
18	movq	%rsi, (%rcx,%rax)		17												
	-	\$8, %rax		18												
	-	%r9, %rax		19 20												
_	jne			20			128									

1	•	(%rd1,%rax), %rs1	_					,	7								
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)	(1)(2)										
(5)	addq	\$8, %rax	→	P3	3	(5)(6)	(3)(4)	(1)(2)									
6	cmpq	%r9, %rax				(7)(8)		(2)(3)(4)	(1)								
7	jne	.L9				(9)(10)		(3)(4)(5)(6)	(2)								
8	movq	(%rdi,%rax), %rsi	→	P4				(3)(4)((6)(7)(8)		(2)		(4)		(E)			(E)
9	movq	(%rcx,%rax), %r8	→	P5				(3)(4)((6)(7)(9) (10) (3)(4)(7)(10)(11)		(0)	(2)	(1)	(4)	(5)			(5)
10		%r8, (%rdi,%rax)						(3)(4)(7)(10)(11) (12) (3)(10)(11)(12)		(8)	(9)	(2)	(1)	(6)		(7)	(5)(6) (1)(5)(6)
_	•	%rsi, (%rcx,%rax)						(3)(10)(11)(12) (13)(14) (10)(11)(13)(14) (15)(16)		(9) (4)		(8)	(2)	(12)		(7)	(1)(5)(6) (2)(5)(6)(7)
_	-	-	>	P6		(21)(22)		(15) (16) (10) (11) (14) (16) (17) (18)				(9)	(8)	(12)			(5)(6)(7)(12)
_	•	%r9, %rax			12			(17) (18) (16) (17) (18) (19) (20						(10)		(14)	(5)(6)(7)(8)(12)(13)
_	jne				13			(20	,		(-)	(- /	(-)			()	
	_	(%rdi,%rax), %rsi			14												
	-	(%rcx,%rax), %r8			15												
_		%r8, (%rdi,%rax)			16												
	-	%rsi, (%rcx,%rax)			17												
	_	\$8, %rax			18												
	_	%r9, %rax			19												
_					20			120									
21)	jne	.LY						129									

<pre>② movq (%rcx,%rax), %r8 → P2 ③ movq %r8, (%rdi,%rax) ④ movq %rsi, (%rcx,%rax) ⑤ addq \$8, %rax → P3</pre> IF ID REN AG M1 M2 M3 M4 ALU MUL BR 1 (1)(2) 2 (3)(4) (1)(2) 3 (5)(6) (3)(4) (1)(2)	ROB
@ movq %rsi, (%rcx,%rax) 2 (3)(4) (1)(2)	
$\begin{array}{c} \mathbf{D} \mathbf{C} = \mathbf{C} \mathbf{C} \mathbf{C} \mathbf{C} \mathbf{C} \mathbf{C} \mathbf{C} \mathbf{C}$	
(a) add $88. \%$ rax $\Rightarrow P3 = 3 = (5)(6)(3)(4) = (1)(2)$	
addy \$67 701 dx	
© cmpq %r9, %rax 4 (7)(8) (5)(6) (2)(3)(4) (1)	
<pre>7 jne .L9</pre> 5 (9)(10) (7)(8) (3)(4)(5)(6) (2) (1)	
® movq (%rdi,%rax), %rsi → P4 $\frac{6}{7}$ (13)(14) (11)(12) (9)(10) (3)(4)((6)(7)(8) (2) (1) (2) (1) (2) (3)(4)((6)(7)(9) (8) (1) (1) (1) (1) (1) (1) (1) (1) (1) (1	
	(5)
10^{10} may a $0/20^{10}$ $10/20^{10}$ $10/20^{10}$ $10/20^{10}$	(5)(6)
$\begin{array}{cccccccccccccccccccccccccccccccccccc$	(1)(5)(6) (2)(5)(6)(7)
$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	(5)(6)(7)(12)
	(5)(6)(7)(8)(12)(13)
(20 (40) (47) (40) (20)	5)(6)(7)(8)(9)(12)(13)(14)
15 movq (%rdi,%rax), %rsi 14	
15 movq (%rcx,%rax), %r8	
① movq (%16x, %16x), %16 ① movq %r8, (%rdi, %rax)	
17 movq %ro, (%rcx,%rax)	
18 19 addq \$8, %rax	
19 addy \$6, %18X 20 cmpq %r9, %rax	
20 21) ine .L9 130	

1	movq	(%rdi,%rax), %rsi	\rightarrow	P1				. 133UC.	4	or ti	ICIII	Cari	IIIav		istruc	CIOILA	r tille 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	\rightarrow	P3	3	(5)(6)	(3)(4)	(1)(2)									
6	cmpq	%r9, %rax				(7)(8)		(2)(3)(4)	(1)								
7	jne	.L9				(9)(10)		(3)(4)(5)(6)	(2)		7.43						
8	movq	(%rdi,%rax), %rsi	\rightarrow	P4				(3)(4)((6)(7)(8)		(2)		(4)		(E)			(5)
9	movq	(%rcx,%rax), %r8	\rightarrow	P5				(3)(4)((6)(7)(9) (10) (3)(4)(7)(10)(11)			(2)		(1)	(5)			(5)
10		%r8, (%rdi,%rax)				(17)(18)		(3)(4)(7)(10)(11) (12) (3)(10)(11)(12)		(8) (9)	(8)	(2)	(1) (2)	(6)		(7)	(5)(6) (1)(5)(6)
$\overline{11}$	•	%rsi, (%rcx,%rax)				(19)(20)		(3)(10)(11)(12) (13)(14) (10)(11)(13)(14) (15)(16)			(9)	(8)	(2)	(12)		(7)	(2)(5)(6)(7)
	•	\$8, %rax	\rightarrow	P6	11	(21)(22)		(15)(16) (10)(11)(14)(16) (17)(18)			(4)		(8)	(13)			(5)(6)(7)(12)
	•	%r9, %rax			12								(9)	(10)		(14)	(5)(6)(7)(8)(12)(13)
		.L9			13						(15)		(4)	(19)			(5)(6)(7)(8)(9)(12)(13)(14)
<u>(15)</u>		(%rdi,%rax), %rsi			14			(=:)(==)				(15)	(3)	(20)			(4)(5)(6)(7)(8)(9)(12)(13)(14) (19)
_	•	(%rcx,%rax), %r8			15												
	-	%r8, (%rdi,%rax)			16												
	-	%rsi, (%rcx,%rax)			17												
_	•	\$8, %rax			18												
		%r9, %rax			19												
<u>21</u>)	_	.L9			20			131									

1	movq	(%rdi,%rax), %rsi	\rightarrow P	1			. 155ue.	4	oi ti	lem	Cai	IIIav	e a l	IISTITUC	tion at	tile Same Cycl
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)										
5	addq	\$8, %rax	→ P	3 3	(5)(6)	(3)(4)	(1)(2)									
6	cmpq	%r9, %rax			(7)(8)		(2)(3)(4)	(1)								
7	jne	.L9	_		(9)(10)		(3)(4)(5)(6)	(2)		(4)						
8	movq	(%rdi,%rax), %rsi	\rightarrow P	٠,	(12)(14)				(2)		(1)		(E)			(E)
9	movq	(%rcx,%rax), %r8	\rightarrow P				(3)(4)((6)(7)(9) (10) (3)(4)(7)(10)(11) (12)		(8)	(2)	(2)	(1)	(5) (6)			(5) (5)(6)
10		%r8, (%rdi,%rax)			(17)(18)				(9)	(8)	(2)	(2)	(0)		(7)	(1)(5)(6)
11	movq	%rsi, (%rcx,%rax)			(19)(20)					(9)	(8)	ν—,	(12)		(2)	(2)(5)(6)(7)
12	addq	\$8, %rax	\rightarrow P	6 1	1 (21)(22)	(19)(20)	(10)(11)(14)(16)	(15)		(4)		(8)	(13)			(5)(6)(7)(12)
13	cmpq	%r9, %rax		1:	2	(21)(22)		(11)	(15)	(3)	(4)	(9)			(14)	(5)(6)(7)(8)(12)(13)
14	jne	.L9		1	3		(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	4			(16)	(10)	(11)	(15)	(3)	(20)			(4)(5)(6)(7)(8)(9)(12)(13)(14) (19)
16	movq	(%rcx,%rax), %r8		1					(16)	(10)	(11)	(15)			(21)	(3)(4)(5)(6)(7)(8)(9)(12)(13) (14)(19)(20)
17)	movq	%r8, (%rdi,%rax)		10												
18	movq	%rsi, (%rcx,%rax)		1												
19	addq	\$8, %rax		18												
	=	%r9, %rax		19												
21)		.L9		Z			132									

1	movq	(%rdi,%rax), %rsi	\rightarrow	P1				. 155UC.	4	Ji ti	ICIII	Cai	IIIav		istique	tionat	the same cycli
2	movq	(%rcx,%rax), %r8	\rightarrow	P2		IF	ID	REN	AG	M1	M2	M3	M4	ALU	MUL	BR	ROB
3		%r8, (%rdi,%rax)				(1)(2)											
4	movq	%rsi, (%rcx,%rax)				(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					(9)(10)		(3)(4)(5)(6)	(2)								
8	movq	(%rdi,%rax), %rsi	\rightarrow	P4		(11) (12)		(3)(4)((6)(7)(8)		(2)		(4)		(E)			(E)
9	movq	(%rcx,%rax), %r8	\rightarrow	P5				(3)(4)((6)(7)(9) (10) (3)(4)(7)(10)(11)		(0)	(2)		(1)	(5) (6)			(5)
10		%r8, (%rdi,%rax)				(17)(18)		(3)(4)(7)(10)(11) (12) (3)(10)(11)(12) (13)(14)		(8) (9)	(8)	(2)	(1) (2)	(6)		(7)	(5)(6) (1)(5)(6)
11	•	%rsi, (%rcx,%rax)				(19)(20)					(9)	(8)	(2)	(12)		(7)	(2)(5)(6)(7)
12	-		\rightarrow	P6	11	(21)(22)		(15)(16) (10)(11)(14)(16) (17)(18)			(4)		(8)	(13)			(5)(6)(7)(12)
13	•	%r9, %rax			12			(17)(18) (16)(17)(18)(19) (20			(3)		(9)	(,		(14)	(5)(6)(7)(8)(12)(13)
<u>14</u>)	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			(=1/(==/		(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>21</u>)	jne				20			133									

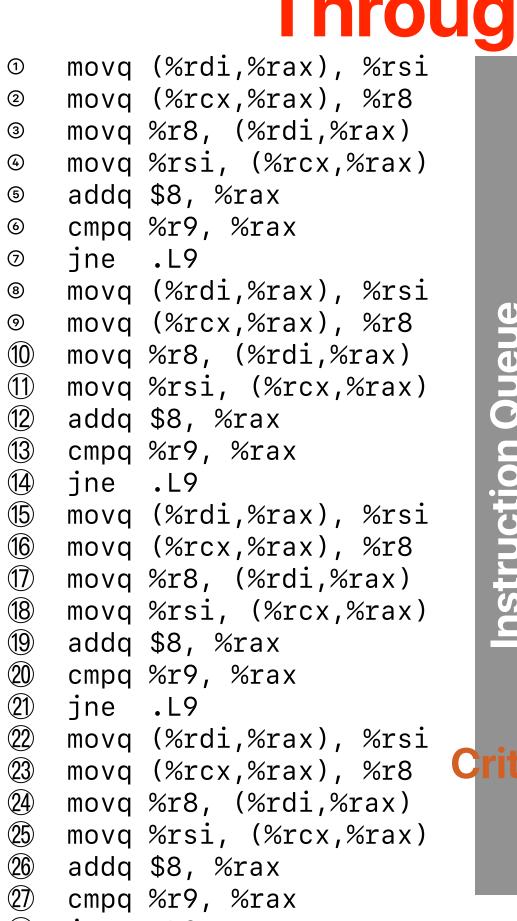
1	movq	(%rdi,%rax), %rsi	→ P1				issue:	4	oi ti	iem	Car	ınav	eall	nstruc	tion at	the same cycle
2	movq	(%rcx,%rax), %r8	→ P2		IF	ID	REN	AG	M1	M2	МЗ	M4	ALU	MUL	BR	ROB
3	movq	%r8, (%rdi,%rax)		1	(1)(2)											
4	movq	%rsi, (%rcx,%rax)			(3)(4)	(1)(2)										
⑤	addq	\$8, %rax	→ P3	3	(5)(6)	(3)(4)	(1)(2)									
6	cmpq	%r9, %rax		4		(5)(6)	(2)(3)(4)	(1)								
7	jne	.L9		5	(9)(10)		(3)(4)(5)(6)	(2)		(4)						
8	movq	(%rdi,%rax), %rsi	→ P4		(11)(12)		(3)(4)((6)(7)(8)	(0)			(1)		(E)			(E)
9	movq	(%rcx,%rax), %r8	→ P5	2	(15)(14)		(3)(4)((6)(7)(9) (10) (3)(4)(7)(10)(11)		(8)	(2)	(2)	(1)	(5) (6)			(5) (5)(6)
10	movq	%r8, (%rdi,%rax)		9	(17)(18)		(12) (3)(10)(11)(12) (13)(14)		(9)	(8)	(2)	(2)	(0)		(7)	(3)(6)
11	movq	%rsi, (%rcx,%rax)		10	(19)(20)		(13)(14) (10)(11)(13)(14) (15)(16)			(9)	(8)	(-)	(12)		(2)	(2)(5)(6)(7)
12	addq	\$8, %rax	→ P6	11	(21)(22)	(19)(20)	(10)(11)(14)(16) (17)(18)		(3)		(9)	(8)	(13)			(5)(6)(7)(12)
13	cmpq	%r9, %rax		12		(21)(22)	(16)(17)(18)(19) (20			(3)		(9)			(14)	(5)(6)(7)(8)(12)(13)
14	jne	.L9		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)	(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)	(12)(13)(14)(19)(20)
17)	movq	%r8, (%rdi,%rax)		16				(17)				(11)				(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)
_	•	\$8, %rax		18												
20	-	%r9, %rax		19 20												
(21)	ine	•		20			134									

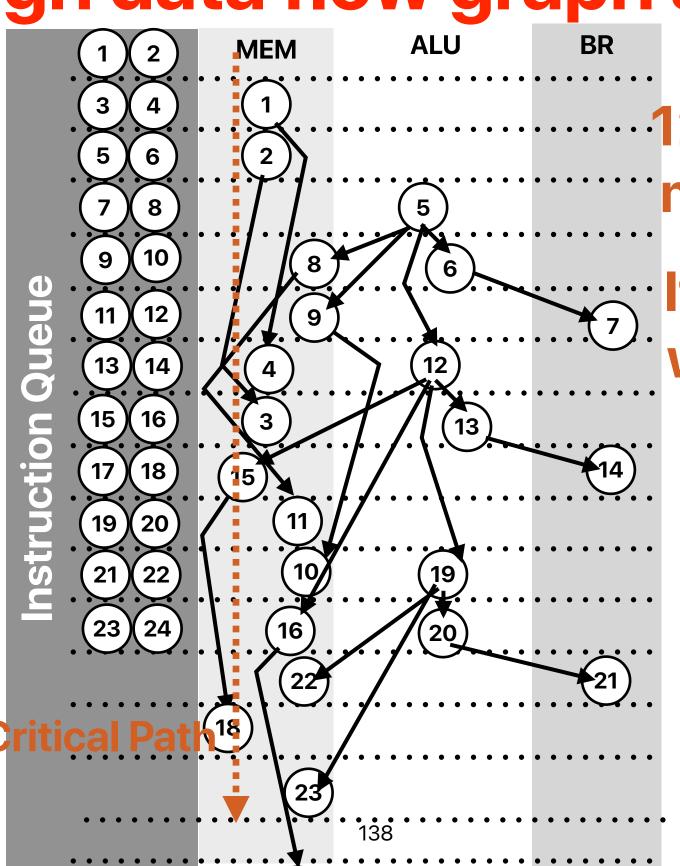
1	movq	(%rdi,%rax), %rsi	\rightarrow	P1			_	. 133UC.	4	or ti	ICIII	Cai	IIIAV		istinc	CIOILS	r tille Saille Cycle
2	movq	(%rcx,%rax), %r8	\rightarrow	P2		IF	ID	REN	AG	M1	M2	МЗ	M4	ALU	MUL	BR	ROB
3	movq	%r8, (%rdi,%rax)			1	(1)(2)											
4	movq	%rsi, (%rcx,%rax)				(3)(4)	(1)(2)										
5	addq	\$8, %rax	\rightarrow	P3	3	(5)(6)	(3)(4)	(1)(2)									
6	cmpq	%r9, %rax			4		(5)(6)	(2)(3)(4)	(1)								
7	jne	.L9			5		(7)(8)	(3)(4)(5)(6)	(2)								
8	movq	(%rdi,%rax), %rsi	\rightarrow	P4				(3)(4)((6)(7)(8)		(2)	(1)	(4)		(E)			(5)
9	movq	(%rcx,%rax), %r8	\rightarrow	P5		(13)(14)		(3)(4)((6)(7)(9) (10) (3)(4)(7)(10)(11)		(0)	(2)		(1)	(5)			(5)
10		%r8, (%rdi,%rax)				(17)(18)		(12)	(0)	(8) (9)	(8)	(2)	(1) (2)	(6)		(7)	(5)(6) (1)(5)(6)
$\overline{11}$	•	%rsi, (%rcx,%rax)				(19)(20)		(3)(10)(11)(12) (13)(14) (10)(11)(13)(14) (15)(16)			(9)	(8)	(2)	(12)		(7)	(2)(5)(6)(7)
	•	\$8, %rax	\rightarrow	P6	11	(21)(22)		(15)(16) (10)(11)(14)(16) (17)(18)			(4)		(8)	(13)			(5)(6)(7)(12)
	•	%r9, %rax			12			(17)(18) (16)(17)(18)(19) (20				(4)	(9)	(10)		(14)	(5)(6)(7)(8)(12)(13)
		.L9			13						(15)		(4)	(19)		,	(5)(6)(7)(8)(9)(12)(13)(14)
<u>(15)</u>		(%rdi,%rax), %rsi			14			(21)(22)			(11)		(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			135									

1	movq	(%rdi,%rax), %rsi	$\rightarrow P$	L			. 10000.	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)										
5	addq	\$8, %rax	→ P:	3 3	(5)(6)	(3)(4)	(1)(2)									
6	cmpq	%r9, %rax			(7)(8)		(2)(3)(4)	(1)								
7	jne	.L9			(9)(10)		(3)(4)(5)(6)	(2)								
8	movq	(%rdi,%rax), %rsi	\rightarrow P	4 6	(11)(12)		(3)(4)((6)(7)(8)	(0)	(2)		(4)		(5)			(5)
9	mova	(%rcx,%rax), %r8	\rightarrow P	5 /	(13)(14)		(3)(4)((6)(7)(9) (10) (3)(4)(7)(10)(11)	(-)	(0)	(2)		(4)	(5)			(5)
10		%r8, (%rdi,%rax)		0	(15)(16) (17)(18)		(12)		(8)	(0)	(2)	(1)	(6)		(7)	(5)(6)
$\widetilde{11}$	•	%rsi, (%rcx,%rax)		10		(17)(18)	(13)(14)			(8) (9)	(8)	(2)	(12)		(7)	(1)(5)(6) (2)(5)(6)(7)
12	-	\$8, %rax	\rightarrow P			(19)(20)	(10)(11)(13)(14) (15)(16) (10)(11)(14)(16) (17)(18)					(8)	(12)			(5)(6)(7)(12)
13	•	%r9, %rax		12		(21)(22)	(17) (18) (16) (17) (18) (19) (20			(3)		(9)	(10)		(14)	(5)(6)(7)(8)(12)(13)
14	jne	.L9		13			(20 (16)(17)(18)(20) (21)(22)			(15)		(4)	(19)		()	(5)(6)(7)(8)(9)(12)(13)(14)
15		(%rdi,%rax), %rsi		14			(21)(22)					(3)	(20)			(4)(5)(6)(7)(8)(9)(12)(13)(14) (19)
_	•	(%rcx,%rax), %r8		15								(15)			(21)	(3)(4)(5)(6)(7)(8)(9)(12)(13) (14)(19)(20)
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)
19	•	\$8, %rax		18	}					(17)		(16)				(10)(11)(12)(13)(14)(15)(19) (20)(21)
20	-	%r9, %rax		19				(18)			(17)					(16)(19)(20)(21)
20	•	.19		20			136									

1	movq	(%rdi,%rax), %rsi	\rightarrow F	1				. 13340.	4	oi ti		Cai	iiiav		Strate		7
2	movq	(%rcx,%rax), %r8	\rightarrow F	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)										
⑤	addq	\$8, %rax	→ F	93	3 ((5)(6)	(3)(4)	(1)(2)									
6	cmpq	%r9, %rax			4	(7)(8)	(5)(6)	(2)(3)(4)	(1)								
7	jne	.L9				(9) (10)		(3)(4)(5)(6)	(2)								
8		(%rdi,%rax), %rsi	\rightarrow F	4				(3)(4)((6)(7)(8)			(1)			<i>,</i> = <i>,</i>			 \
9	mova	(%rcx,%rax), %r8	\rightarrow F	5				(3)(4)((6)(7)(9) (10) (3)(4)(7)(10)(11)			(2)		(4)	(5)			(5)
10		%r8, (%rdi,%rax)						(3)(4)(7)(10)(11) (12) (3)(10)(11)(12)	(0)	(8)	(0)	(2)	(1)	(6)		(7)	(5)(6)
0	•	%rsi, (%rcx,%rax)					(15)(16) (17)(18)	(13)(14)		(9)		(0)	(2)	(12)		(7)	<u>(1)(5)(6)</u>
_	•	\$8, %rax	→ F	_				(10) (11) (13) (14) (15) (16) (10) (11) (14) (16)			(9)		(8)	(12)			(2) (5)(6)(7)
13	-	%r9, %rax	•		12	(21)(22)		(10)(11)(14)(16) (17)(18) (16)(17)(18)(19) (20			(4) (3)		(8) (9)	(13)		(14)	(5)(6)(7)(12) (5)(6)(7)(8)(12)(13)
		.L9			13		(/(/				(15)		(4)	(19)		(17)	(5)(6)(7)(8)(9)(12)(13)(14)
_		(%rdi,%rax), %rsi			14			(21)(22)			(11)			(20)			(4)(5)(6)(7)(8)(9)(12)(13)(14)
_	•	(%rcx,%rax), %r8			15				(10)				(15)	(_0)		(21)	(19) (2)(4)(5)(6)(7)(9)(9)(12)(13)
_	-				16				(17)				(11)			, ,	(14)(19)(20) (12)(13)(14)(15)(19)(20(21)
(17) (10)	•	%r8, (%rdi,%rax)			17					(17)			(10)				(11)(12)(13)(14)(15)(19)(20(21)
18	•	%rsi, (%rcx,%rax)			18						(17)		(16)				(10)(11)(12)(12)(14)(15)(19) (20)(21)
19	•	\$8, %rax			19				(18)			(17)					(16) (19)(20)(21)
20		%r9, %rax			20					(18)			(17)				(19)(20)(21)
(21)	ine	.L9						137									

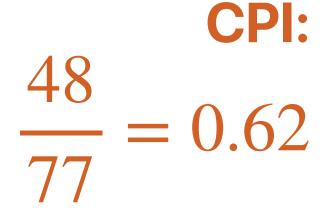
Through data flow graph analysis





12 cycles for every 11memory instructions

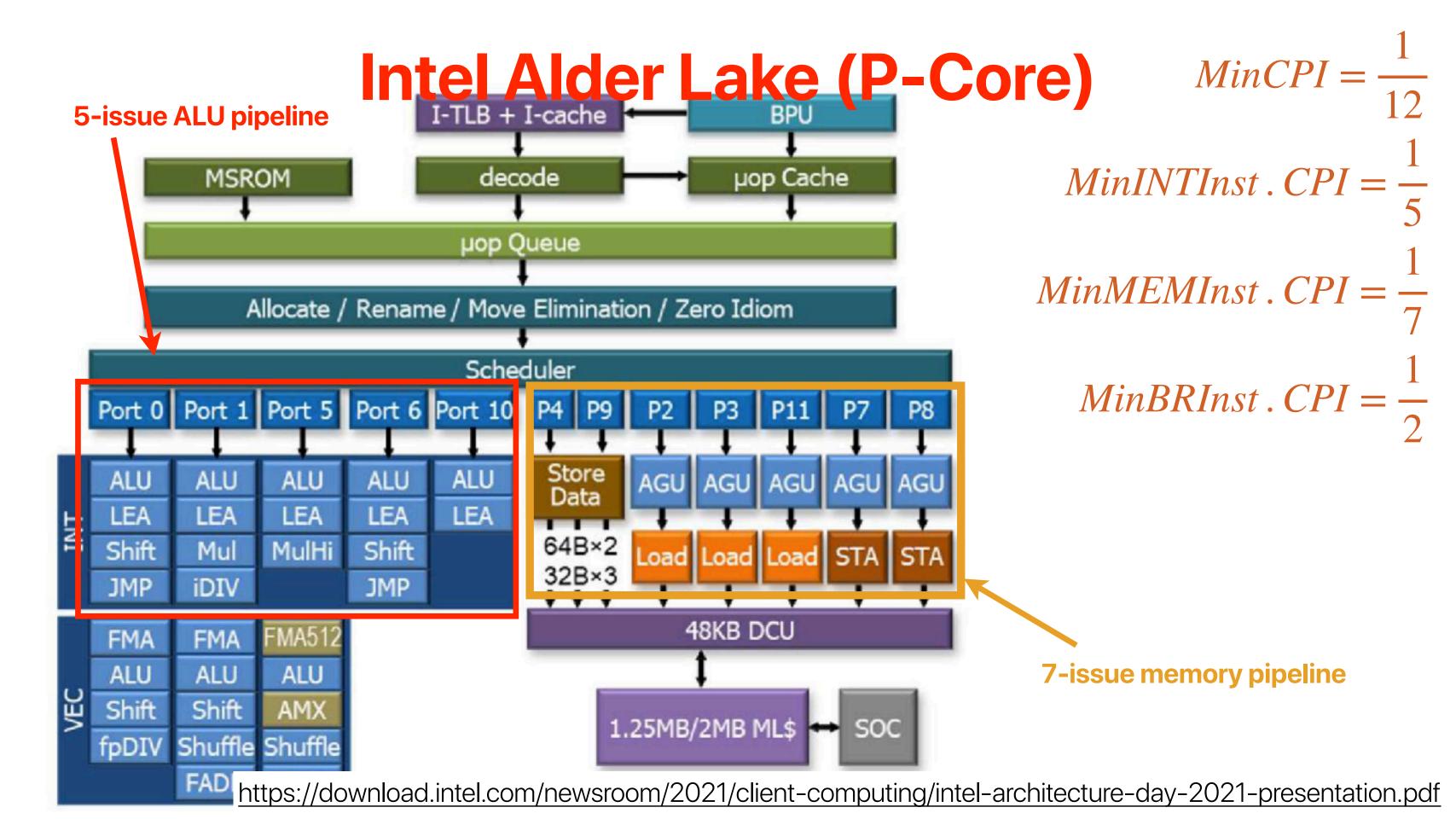
If we have 11 loops, it will have 44 memory instructions, 77 instructions in total and take 48 cycles



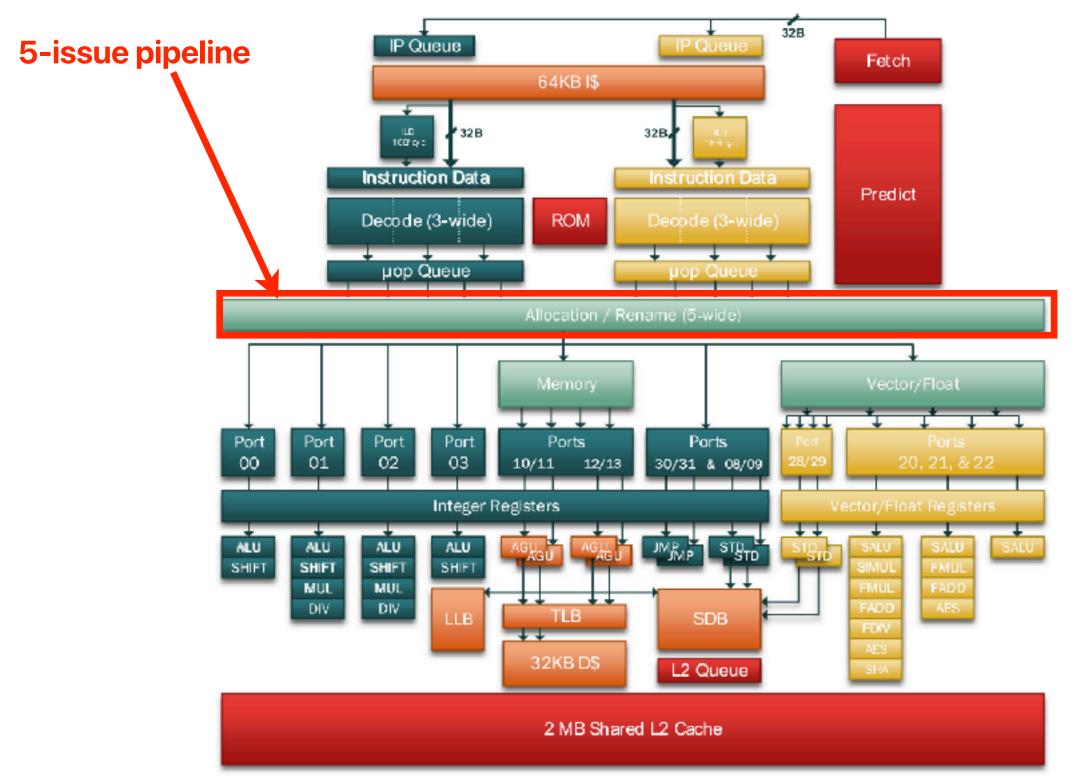
Takeaways: data hazards

- 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

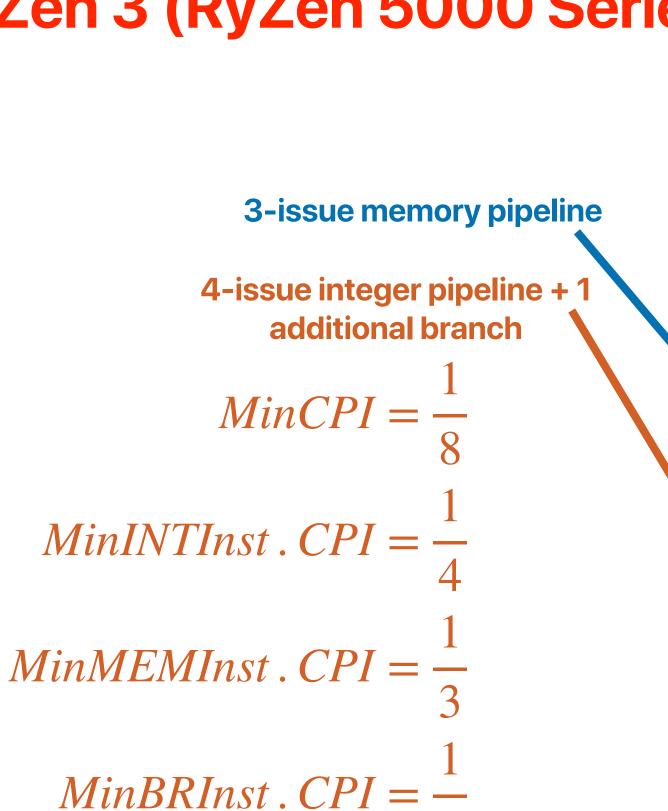
The pipelines of Modern Processors

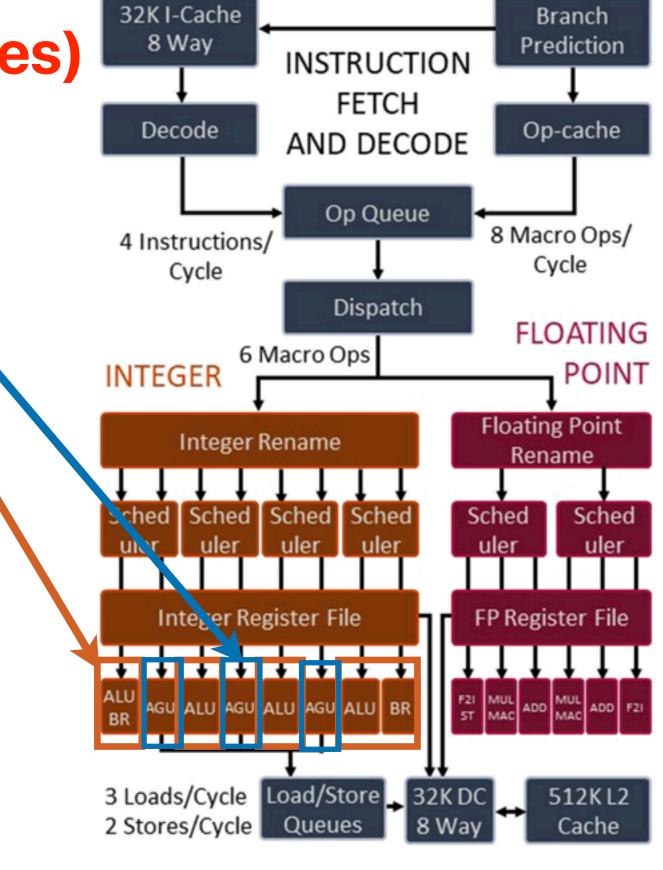


Intel Alder Lake (E-Core)



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