Computer Architecture
ELE 475 / COS 475
Slide Deck 4: Superscalar 1

David Wentzlaff
Department of Electrical Engineering
Princeton University

PRINCETON UNIVERSITY

PRINCETON
School of Engineering and Applied Science

Types of Data Hazards

Consider executing a sequence of

$$r_k \leftarrow r_i \text{ op } r_j$$

type of instructions

Data-dependence

$$\begin{tikzcd}
r_3 & \leftarrow & r_1 \text{ op } r_2 \\
r_5 & \leftarrow & r_3 \text{ op } r_4
\arrow[from=1-1, to=2-3]
\end{tikzcd}$$

Read-after-Write (RAW) hazard

Anti-dependence

$$\begin{tikzcd}
r_3 & \leftarrow & r_1 \text{ op } r_2 \\
r_1 & \leftarrow & r_4 \text{ op } r_5
\arrow[from=1-3, to=2-1]
\end{tikzcd}$$

Write-after-Read (WAR) hazard

Output-dependence

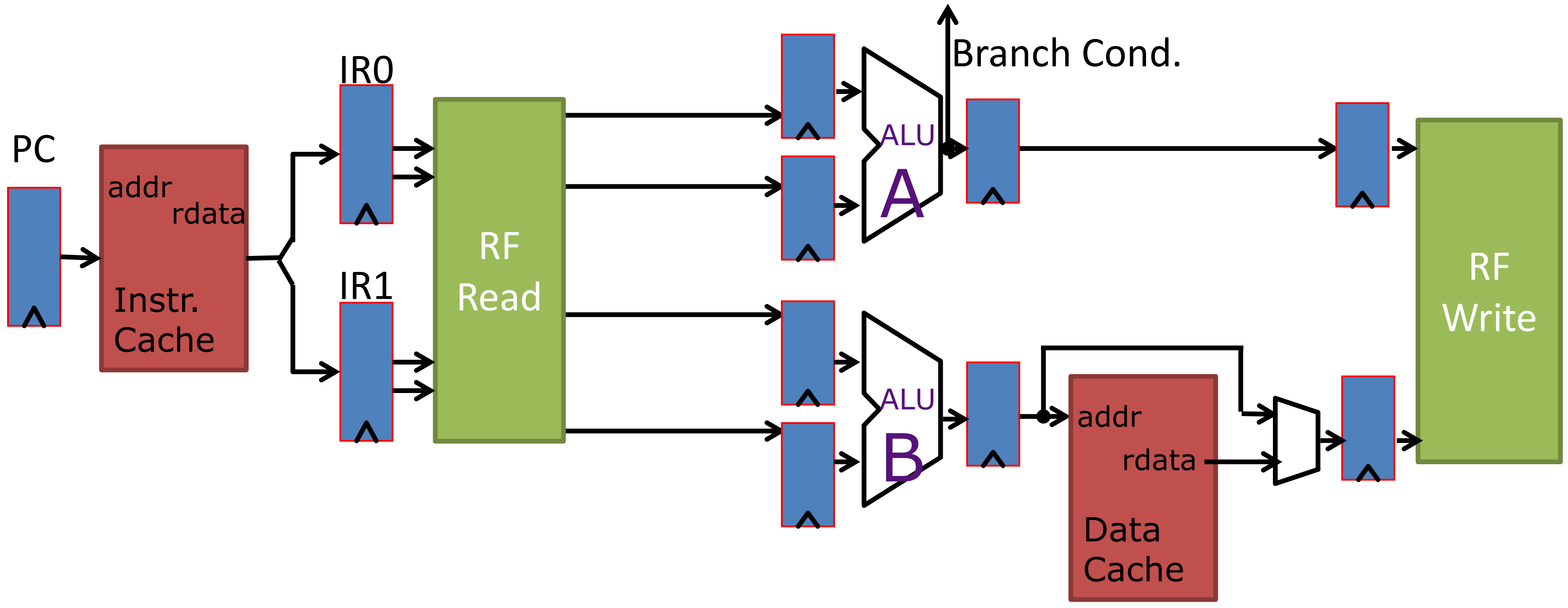
$$\begin{tikzcd}
r_3 & \leftarrow & r_1 \text{ op } r_2 \\
r_3 & \leftarrow & r_6 \text{ op } r_7
\arrow[from=1-1, to=2-1, bend right]
\end{tikzcd}$$

Write-after-Write (WAW) hazard

Introduction to Superscalar Processor

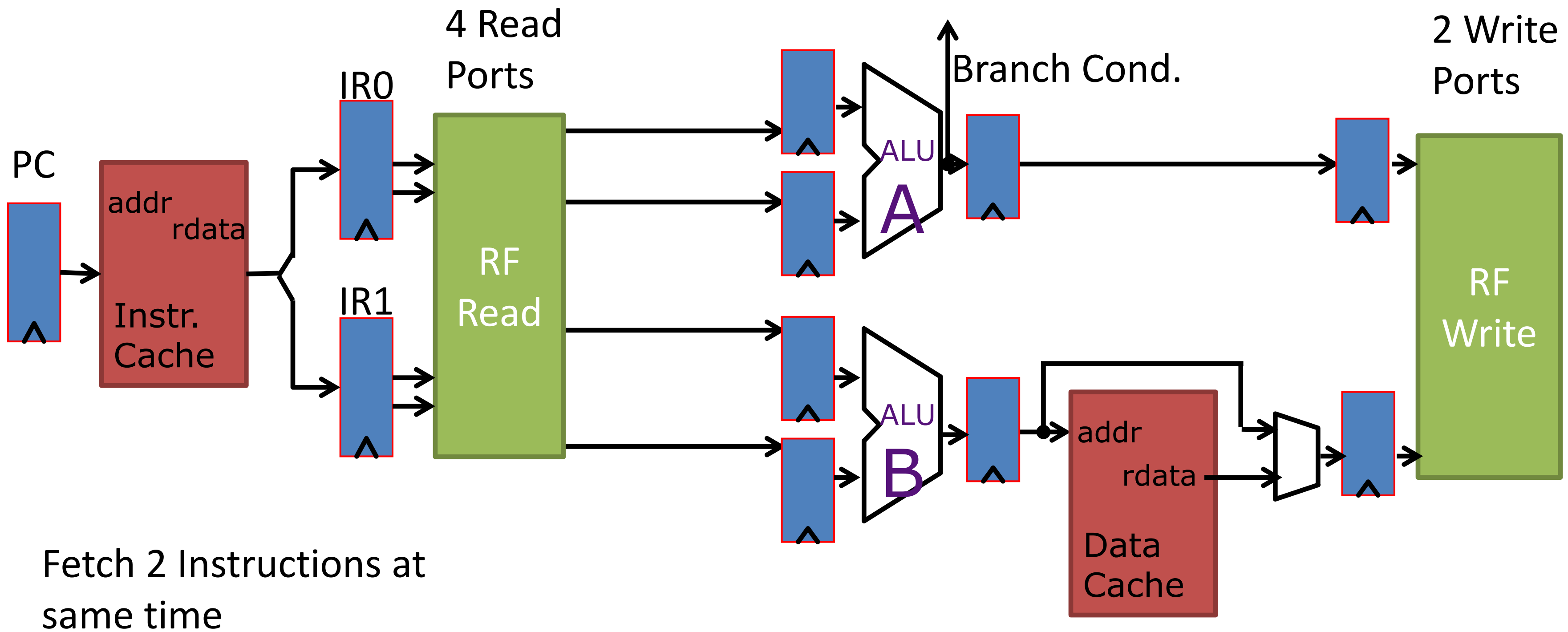
- Processors studied so far are fundamentally limited to CPI >= 1
- Superscalar processors enable CPI < 1 (IPC > 1) by executing multiple instructions in parallel
- Can have both in-order and out-of-order superscalar processors. We will start with in-order.

Baseline 2-Way In-Order Superscalar Processor



Pipe A: Integer Ops., Branches
Pipe B: Integer Ops., Memory

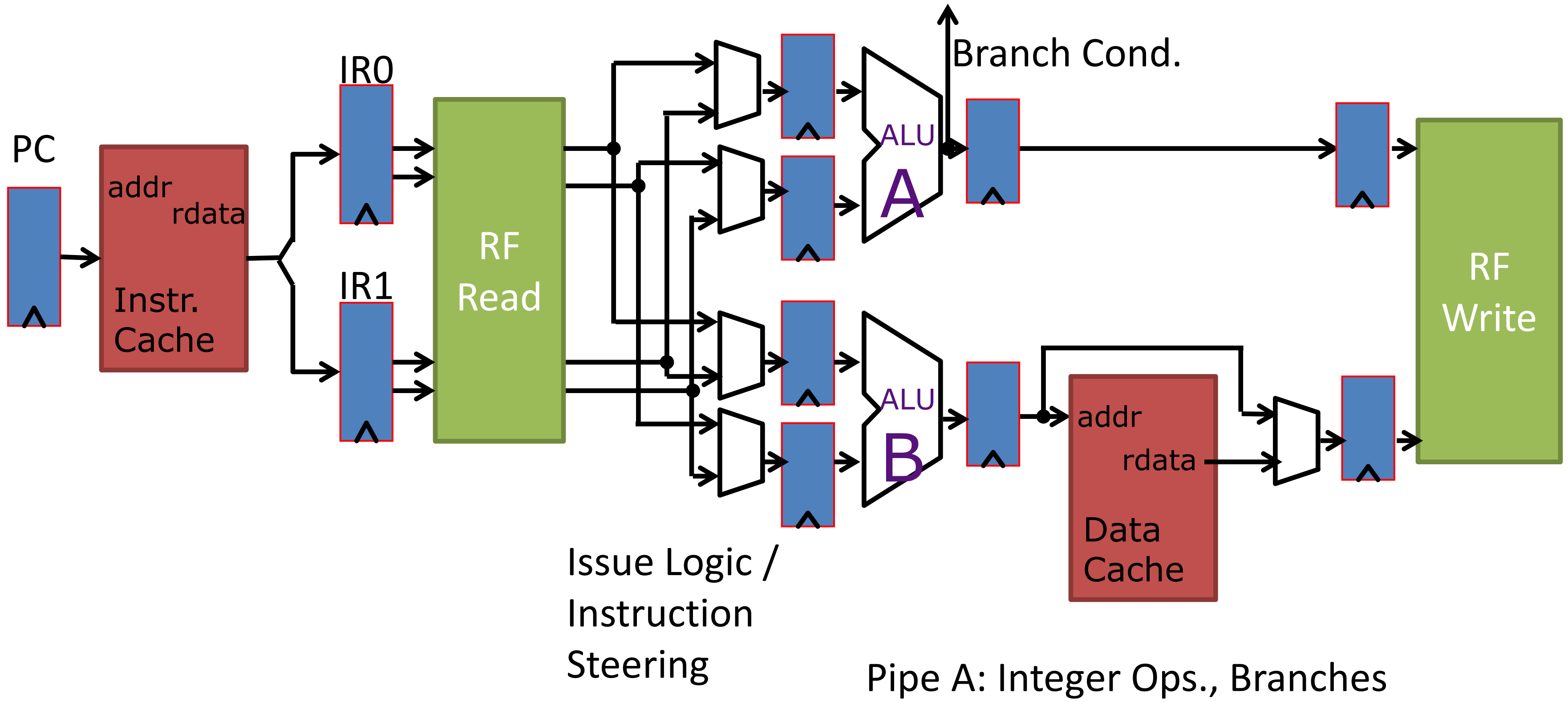
Baseline 2-Way In-Order Superscalar Processor



Fetch 2 Instructions at same time

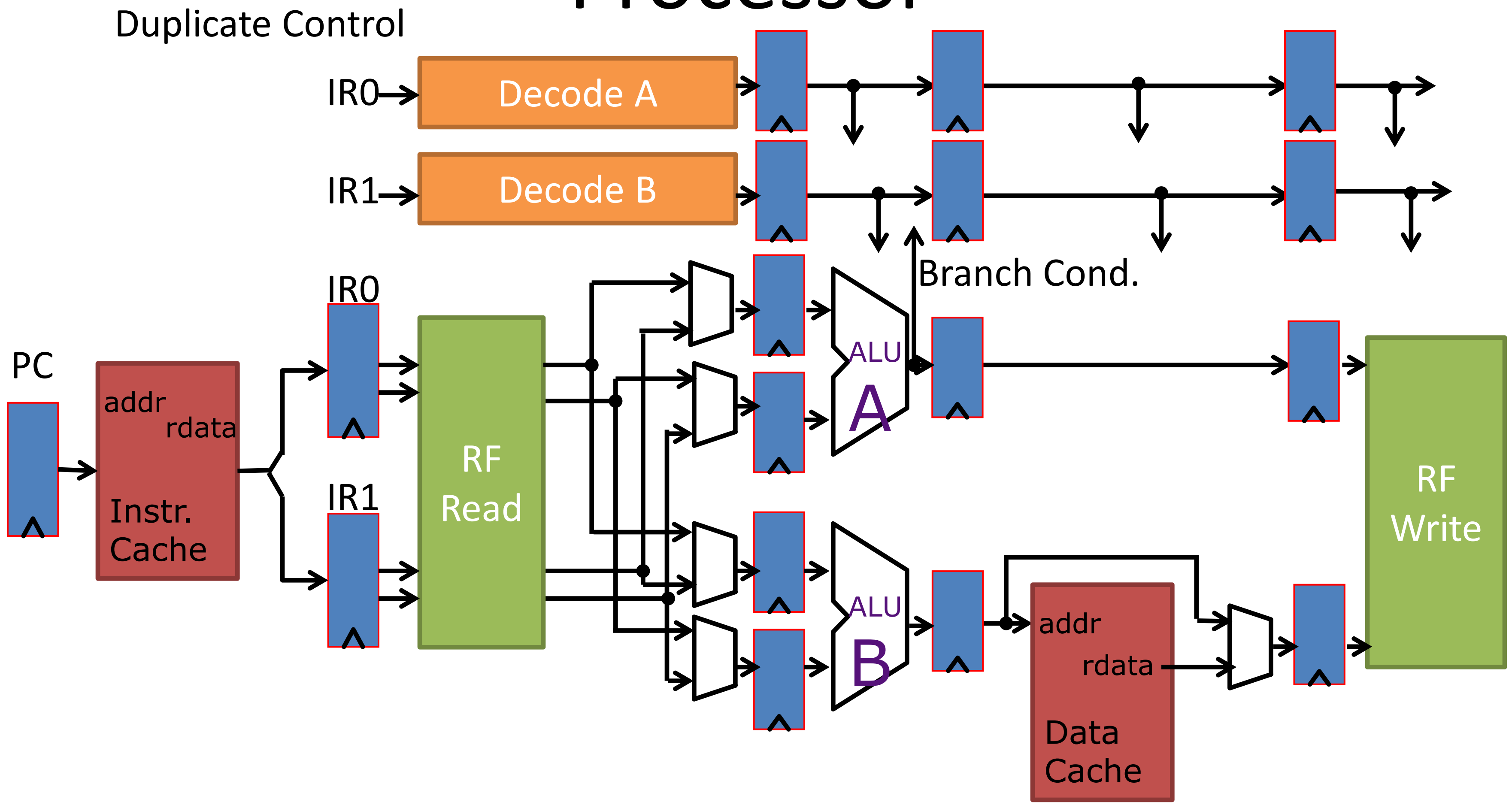
Pipe A: Integer Ops., Branches
Pipe B: Integer Ops., Memory

Baseline 2-Way In-Order Superscalar Processor



Pipe A: Integer Ops., Branches
Pipe B: Integer Ops., Memory

Baseline 2-Way In-Order Superscalar Processor



Pipe A: Integer Ops., Branches
Pipe B: Integer Ops., Memory

Issue Logic Pipeline Diagrams

```
OpA    F   D   A0  A1  W
OpB    F   D   B0  B1  W
OpC        F   D   A0  A1  W
OpD        F   D   B0  B1  W
OpE            F   D   A0  A1  W
OpF            F   D   B0  B1  W
```

CPI = 0.5 (IPC = 2)

Double Issue Pipeline
Can have two instructions in same stage at same time

```
ADDIU  F   D   A0  A1  W
LW     F   D   B0  B1  W
LW         F   D   B0  B1  W
ADDIU      F   D   A0  A1  W
LW             F   D   B0  B1  W
LW             F   D   D   B0  B1  W
```

Instruction Issue Logic swaps from natural position

Structural Hazard

Dual Issue Data Hazards

```
No Bypassing:
ADDIU R1,R1,1  F  D  A0 A1 W
ADDIU R3,R4,1  F  D  B0 B1 W
ADDIU R5,R6,1     F  D  A0 A1 W
ADDIU R7,R5,1     F  D  D  D  D  A0 A1 W

Full Bypassing:
ADDIU R1,R1,1  F  D  A0 A1 W
ADDIU R3,R4,1  F  D  B0 B1 W
ADDIU R5,R6,1     F  D  A0 A1 W
ADDIU R7,R5,1     F  D  D  A0 A1 W
```

Dual Issue Data Hazards

```
Order Matters:
ADDIU R1,R1,1  F  D  A0 A1 W
ADDIU R3,R4,1  F  D  B0 B1 W
ADDIU R7,R5,1     F  D  A0 A1 W
ADDIU R5,R6,1     F  D  B0 B1 W
```

WAR Hazard Possible?

Fetch Logic and Alignment

Cyc	Addr	Instr
0	0x000	OpA
0	0x004	OpB
1	0x008	OpC
1	0x00C	J 0x100
…		
2	0x100	OpD
2	0x104	J 0x204
…		
3	0x204	OpE
3	0x208	J 0x30C
…		
4	0x30C	OpF
4	0x310	OpG
5	0x314	OpH

0x000	0	0	1	1
…				
0x100	2	2		
…				
0x200		3	3	
…				
0x300				4
0x310	4	5		

Fetching across cache Lines is very hard. May need extra ports

Fetch Logic and Alignment

```
Cyc Addr Instr
0   0x000 OpA
0   0x004 OpB
1   0x008 OpC
1   0x00C J 0x100
…
2   0x100 OpD
2   0x104 J 0x204
…
3   0x204 OpE
3   0x208 J 0x30C
…
4   0x30C OpF
4   0x310 OpG
5   0x314 OpH
```

```
Ideal, No Alignment Constraints

OpA F   D   A0  A1  W
OpB F   D   B0  B1  W
OpC     F   D   B0  B1  W
J       F   D   A0  A1  W
OpD         F   D   B0  B1  W
J           F   D   A0  A1  W
OpE             F   D   B0  B1  W
J               F   D   A0  A1  W
OpF                 F   D   A0  A1  W
OpG                 F   D   B0  B1  W
OpH                     F   D   A0  A1  W
```

With Alignment Constraints

```
Cyc Addr Instr
?   0x000 OpA
?   0x004 OpB
?   0x008 OpC
?   0x00C J 0x100
…
?   0x100 OpD
?   0x104 J 0x204
…
?   0x204 OpE
?   0x208 J 0x30C
…
?   0x30C OpF
?   0x310 OpG
?   0x314 OpH
```

0x000	0	0	1	1
…				
0x100	2	2		
…				
0x200	3 ✖	3	4	4 ✖
…				
0x300			5 ✖	5
0x310	6	6		

With Alignment Constraints

```
Cyc Addr Instr
1   0x000 OpA       F  D  A0 A1 W
1   0x004 OpB       F  D  B0 B1 W
2   0x008 OpC          F  D  B0 B1 W
2   0x00C J 0x100      F  D  A0 A1 W
3   0x100 OpD             F  D  B0 B1 W
3   0x104 J 0x204         F  D  A0 A1 W
4   0x200 ?                  F  -  -  -  -
4   0x204 OpE                F  D  A0 A1 W
5   0x208 J 0x30C               F  D  A0 A1 W
5   0x20C ?                     F  -  -  -  -
6   0x308 ?                        F  -  -  -  -
6   0x30C OpF                      F  D  A0 A1 W
7   0x310 OpG                         F  D  A0 A1 W
7   0x314 OpH                         F  D  B0 B1 W
```

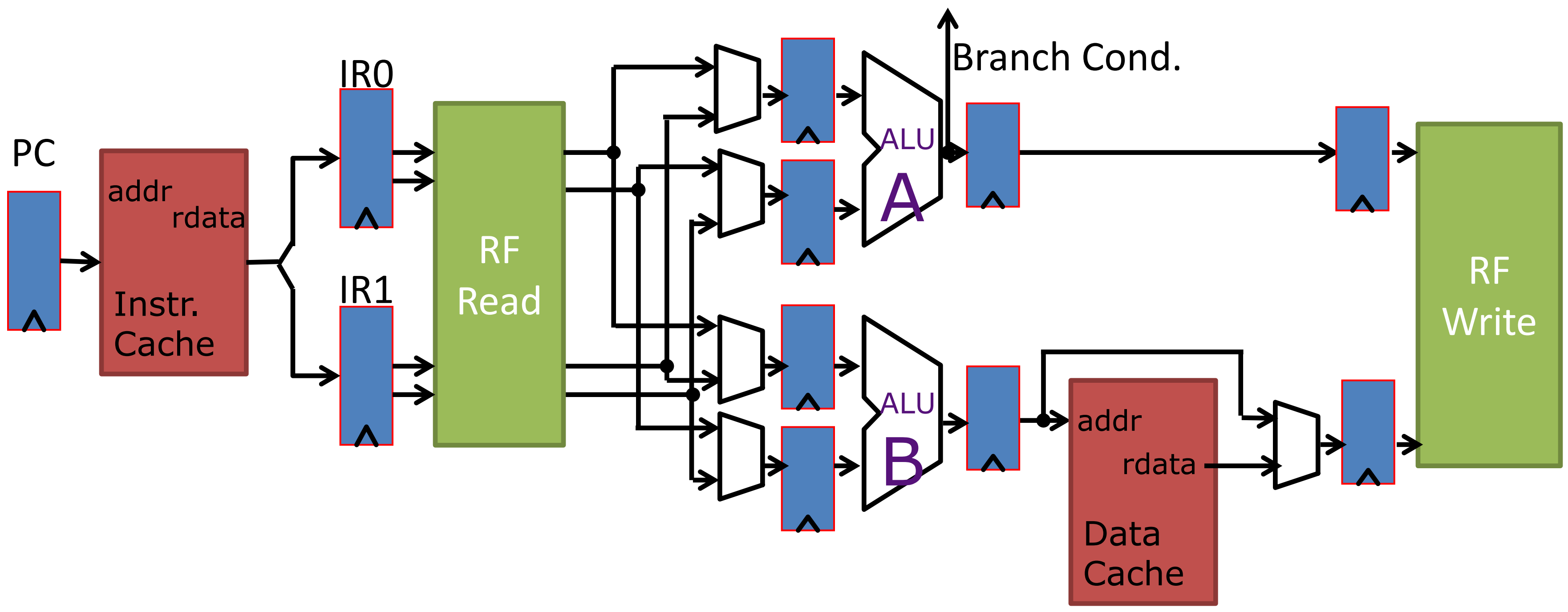
Precise Exceptions and Superscalars

- Similar to tracking program order for data dependencies, we need to track order for exceptions

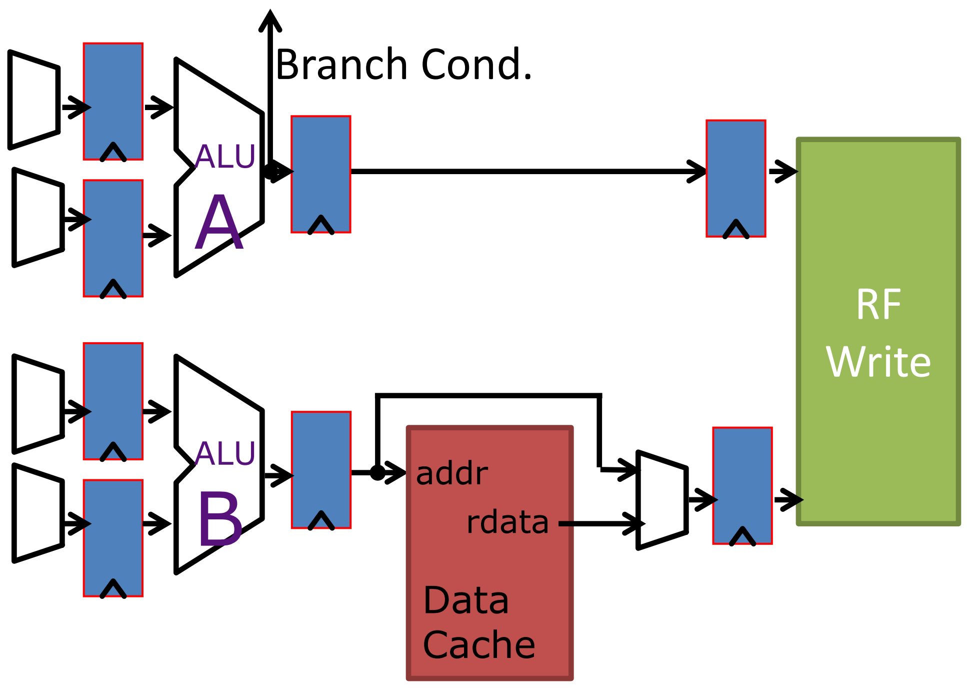
```
LW         F  D B0 B1 W
SYSCALL    F  D A0 A1 W
```

LW is in B pipeline, but commits first in logical order!

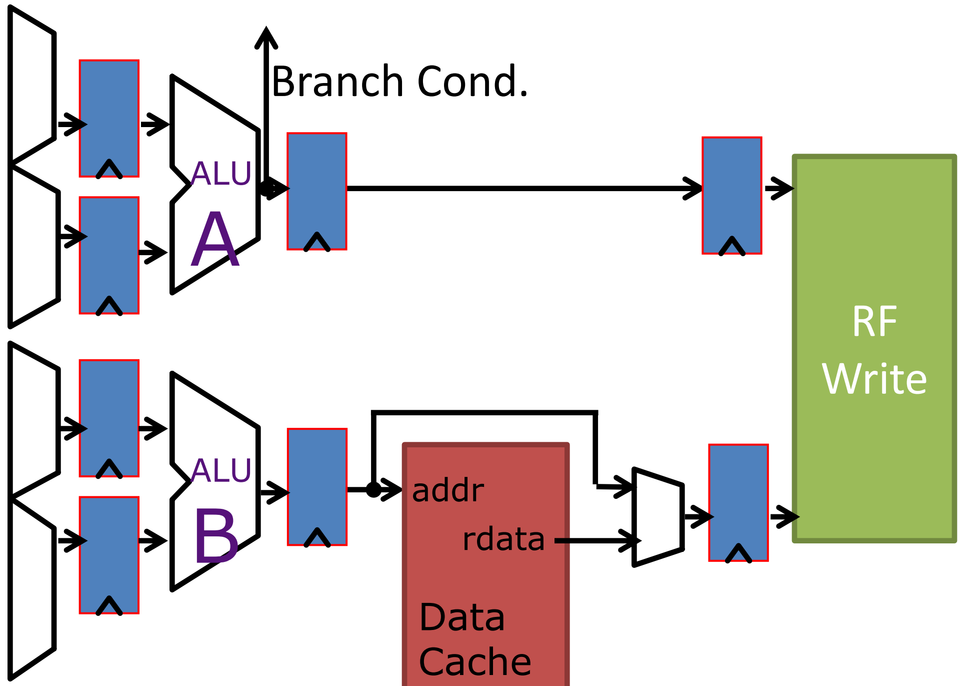
Bypassing in Superscalar Pipelines



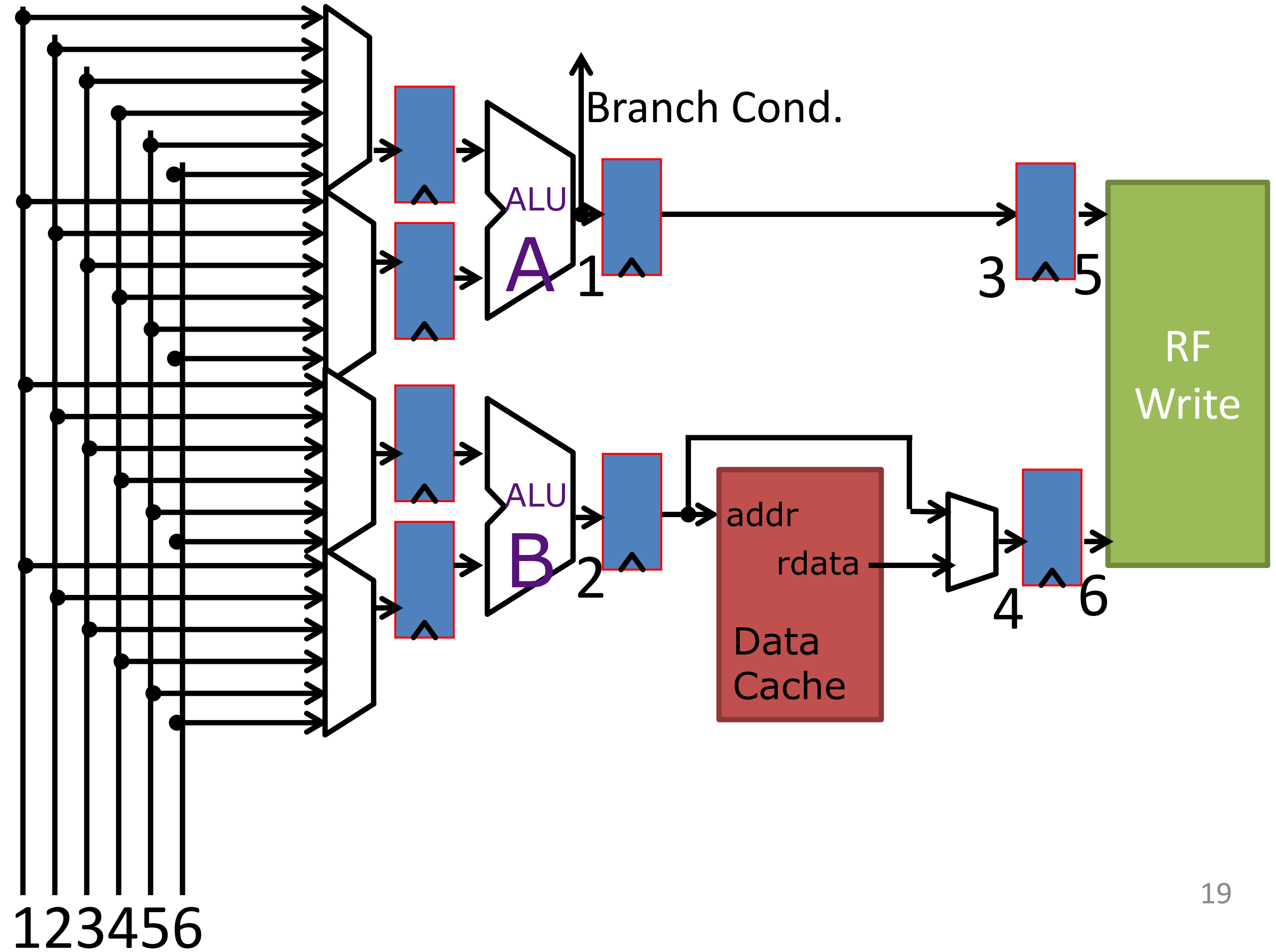
Bypassing in Superscalar Pipelines



Bypassing in Superscalar Pipelines



Bypassing in Superscalar Pipelines



Breaking Decode and Issue Stage

- Bypass Network can become very complex
- Can motivate breaking Decode and Issue Stage

D = Decode, Possibly resolve structural Hazards

I = Register file read, Bypassing, Issue/Steer Instructions to proper unit

```
OpA  F  D  I  A0 A1 W
OpB  F  D  I  B0 B1 W
OpC     F  D  I  A0 A1 W
OpD     F  D  I  B0 B1 W
```

Superscalars Multiply Branch Cost

BEQZ	F	D	I	A0	A1	W				
OpA	F	D	I	B0	-	-				
OpB		F	D	I	-	-	-			
OpC		F	D	I	-	-	-			
OpD			F	D	-	-	-	-		
OpE			F	D	-	-	-	-		
OpF				F	-	-	-	-	-	
OpG				F	-	-	-	-	-	
OpH					F	D	I	A0	A1	W
OpI					F	D	I	B0	B1	W

Acknowledgements

- These slides contain material developed and copyright by:
  - Arvind (MIT)
  - Krste Asanovic (MIT/UCB)
  - Joel Emer (Intel/MIT)
  - James Hoe (CMU)
  - John Kubiatowicz (UCB)
  - David Patterson (UCB)
  - Christopher Batten (Cornell)

- MIT material derived from course 6.823
- UCB material derived from course CS252 & CS152
- Cornell material derived from course ECE 4750