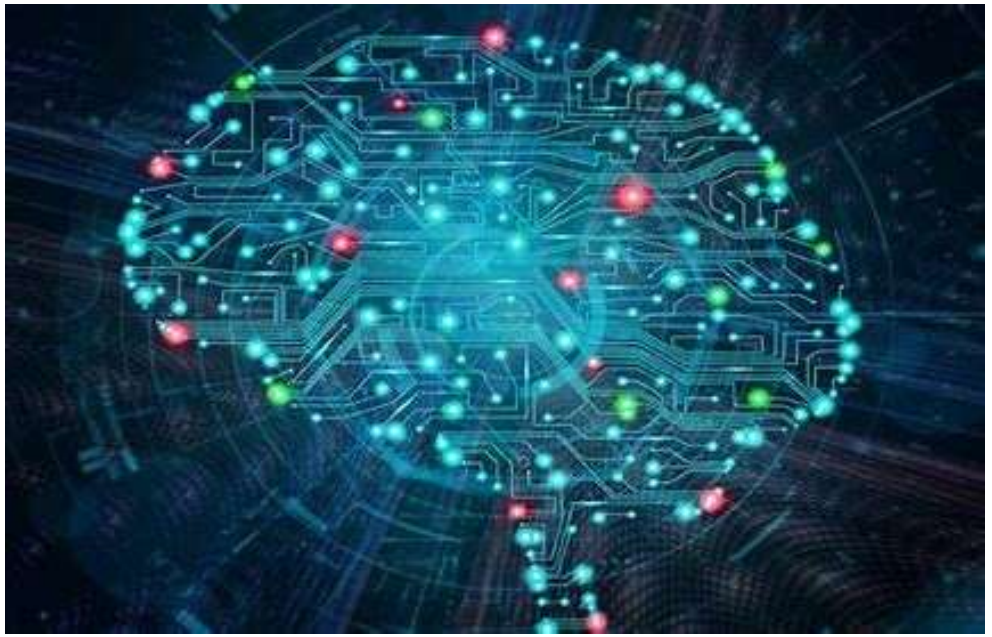
JOINT INSTITUTE
交大密西根学院

RC6



ECE3700J Intro to Computer Organization

2023.11.7

Data Hazard

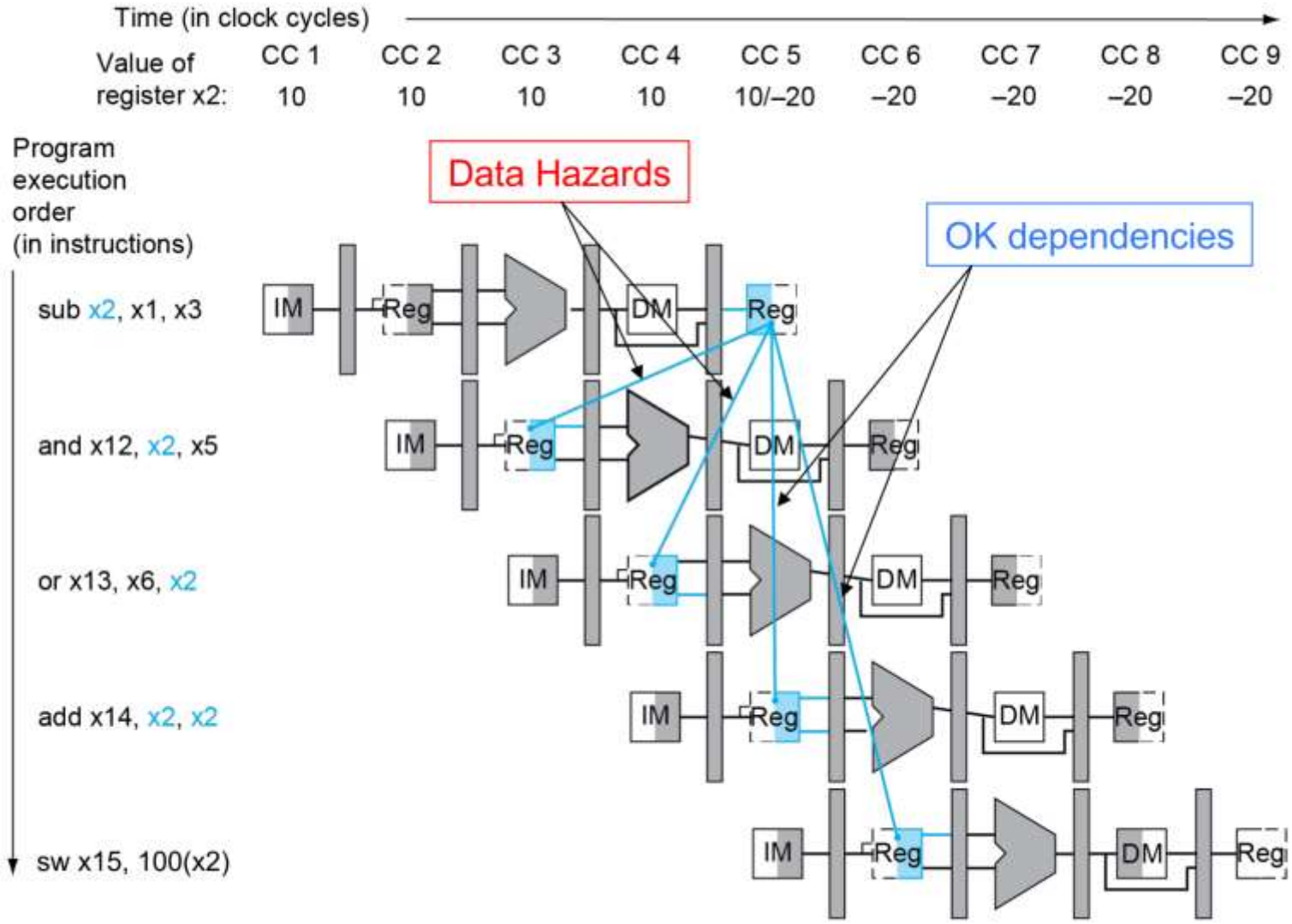
- Consider this sequence:

```
(1)  sub   x2, x1,x3
(2)  and   x12,x2,x5
(3)  or    x13,x6,x2
(4)  add   x14,x2,x2
(5)  sw    x15,100(x2)
```

- Data dependencies between:

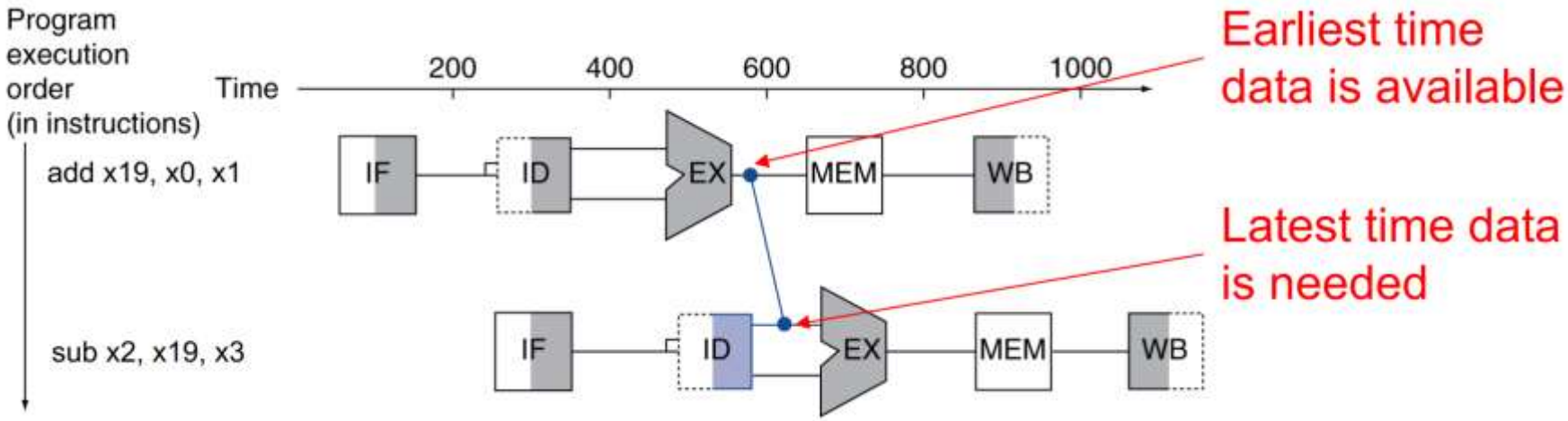
```
(1)  &  (2)
(1)  &  (3)
(1)  &  (4)
(1)  &  (5)
```

Data Hazard

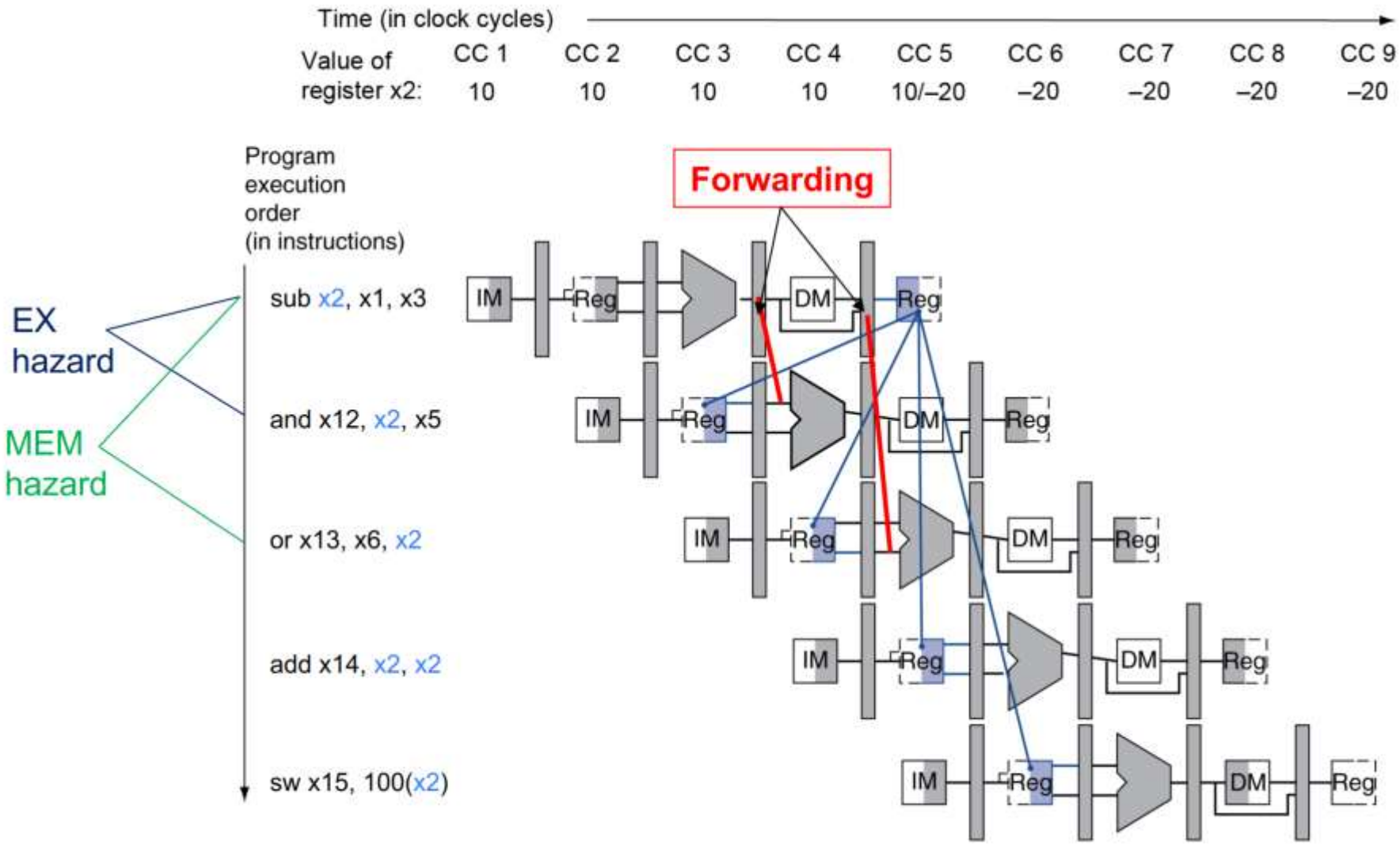

Time (in clock cycles)
Value of register x2:
CC 1
CC 2
CC 3
CC 4
CC 5
CC 6
CC 7
CC 8
CC 9
10
10
10
10
10/–20
–20
–20
–20
–20
Program execution order (in instructions)
Data Hazards
OK dependencies
sub x2, x1, x3
and x12, x2, x5
or x13, x6, x2
add x14, x2, x2
sw x15, 100(x2)
IM
Reg
DM

Data Hazard

The most important part to understand how to solve a hazard by forwarding

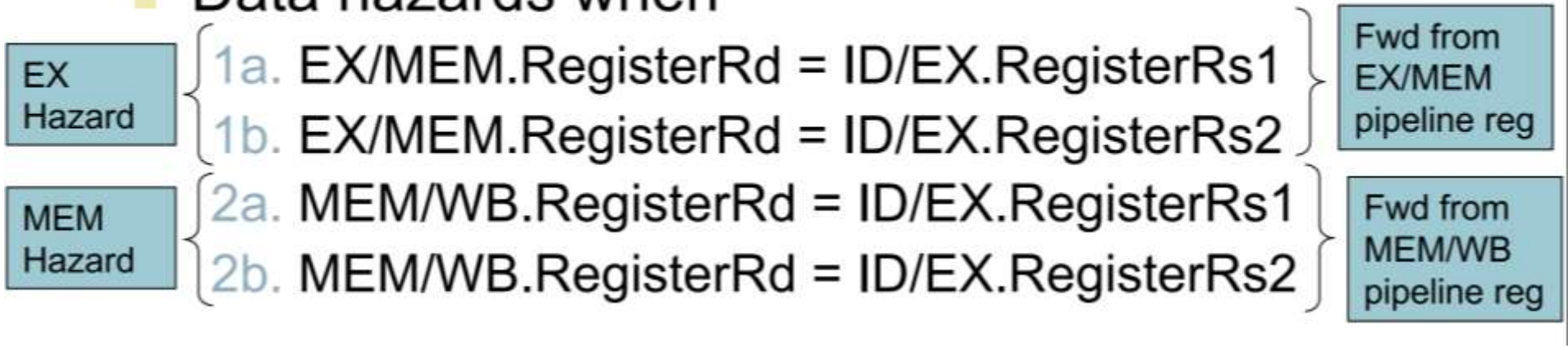


Data Hazard

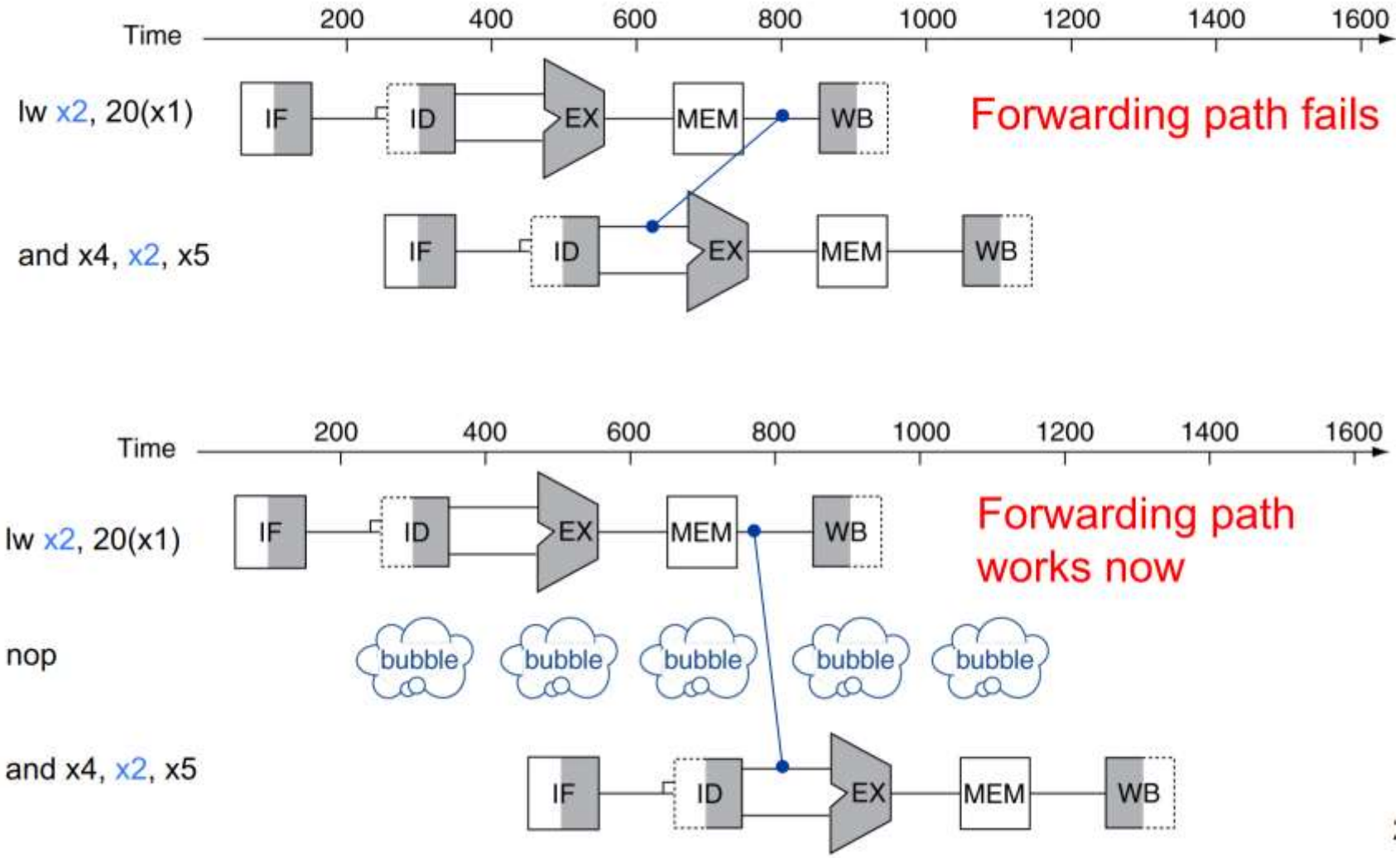

Time (in clock cycles)
CC 1
CC 2
CC 3
CC 4
CC 5
CC 6
CC 7
CC 8
CC 9
Value of register x2:
10
10
10
10
10/–20
–20
–20
–20
–20
Program execution order (in instructions)
Forwarding
EX hazard
MEM hazard
sub x2, x1, x3
and x12, x2, x5
or x13, x6, x2
add x14, x2, x2
sw x15, 100(x2)
IM
Reg
DM

Data Hazard

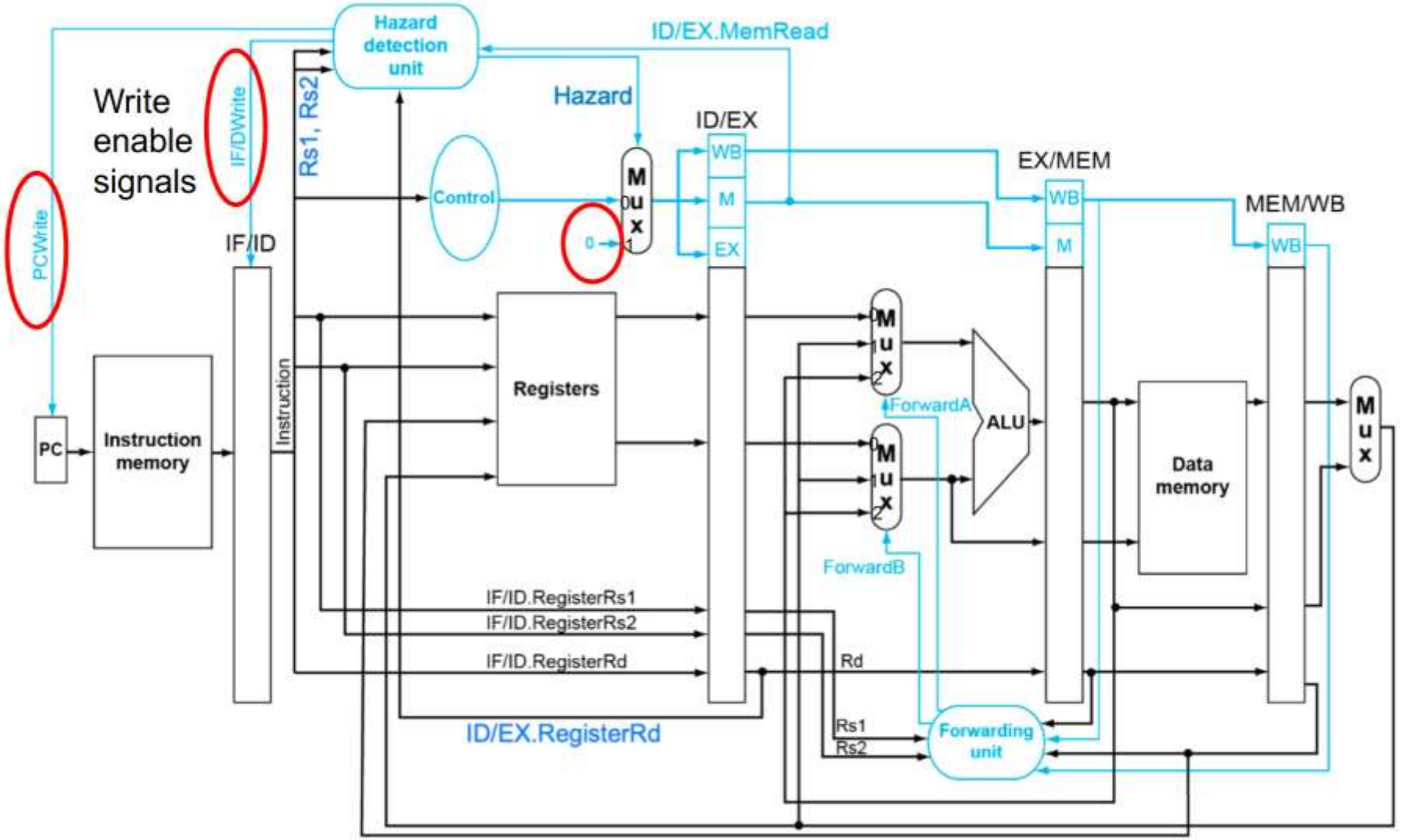
- Denotation:
  - e.g., ID/EX.RegisterRs1 = register number for Rs1 sitting in ID/EX pipeline register
- ALU operand register numbers in EX stage are given by
  - ID/EX.RegisterRs1, ID/EX.RegisterRs2
- Data hazards when



Data Hazard

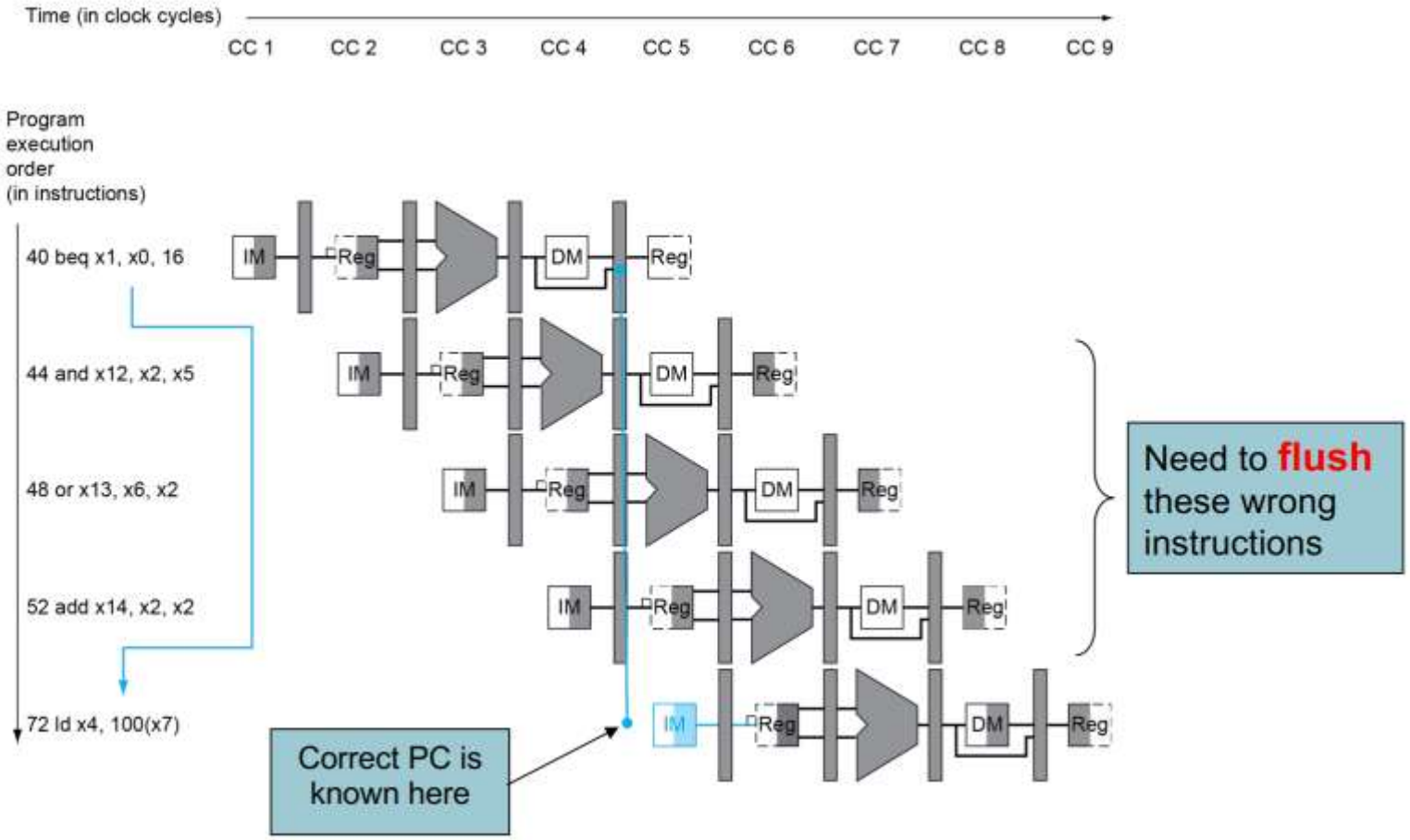

Time
200
400
600
800
1000
1200
1400
1600
lw x2, 20(x1)
IF
ID
EX
MEM
WB
Forwarding path fails
and x4, x2, x5
IF
ID
EX
MEM
WB
Time
200
400
600
800
1000
1200
1400
1600
lw x2, 20(x1)
IF
ID
EX
MEM
WB
Forwarding path works now
nop
bubble
bubble
bubble
bubble
bubble
and x4, x2, x5
IF
ID
EX
MEM
WB

Data Hazard


Hazard detection unit
ID/EX.MemRead
Write enable signals
IF/DWrite
Rs1, Rs2
Hazard
ID/EX
WB
M
EX
EX/MEM
MEM/WB
Control
PCWrite
IF/ID
Instruction
PC
Instruction memory
Registers
ALU
Data memory
ForwardA
ForwardB
IF/ID.RegisterRs1
IF/ID.RegisterRs2
IF/ID.RegisterRd
Rd
ID/EX.RegisterRd
Rs1
Rs2
Forwarding unit

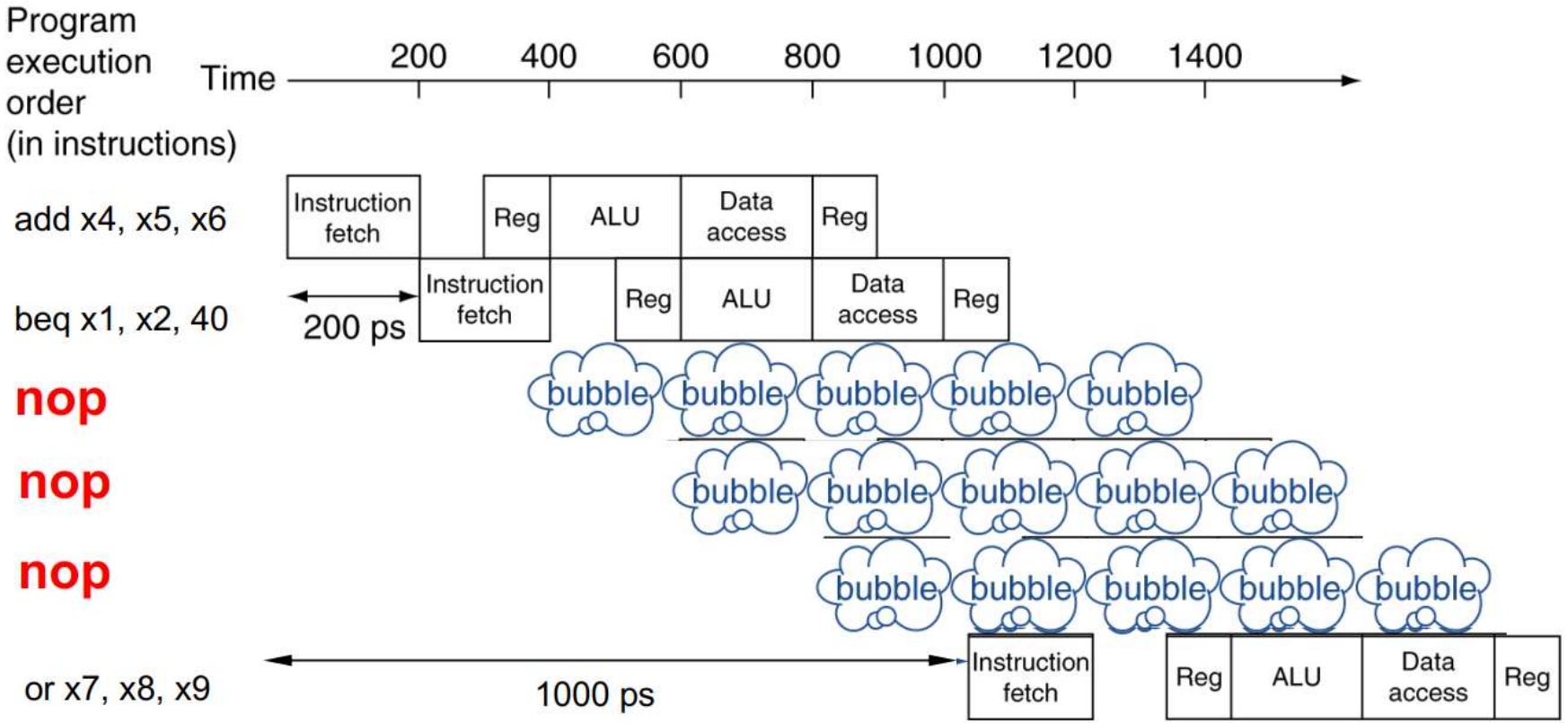
Control Hazard

- Branch outcome determined in MEM

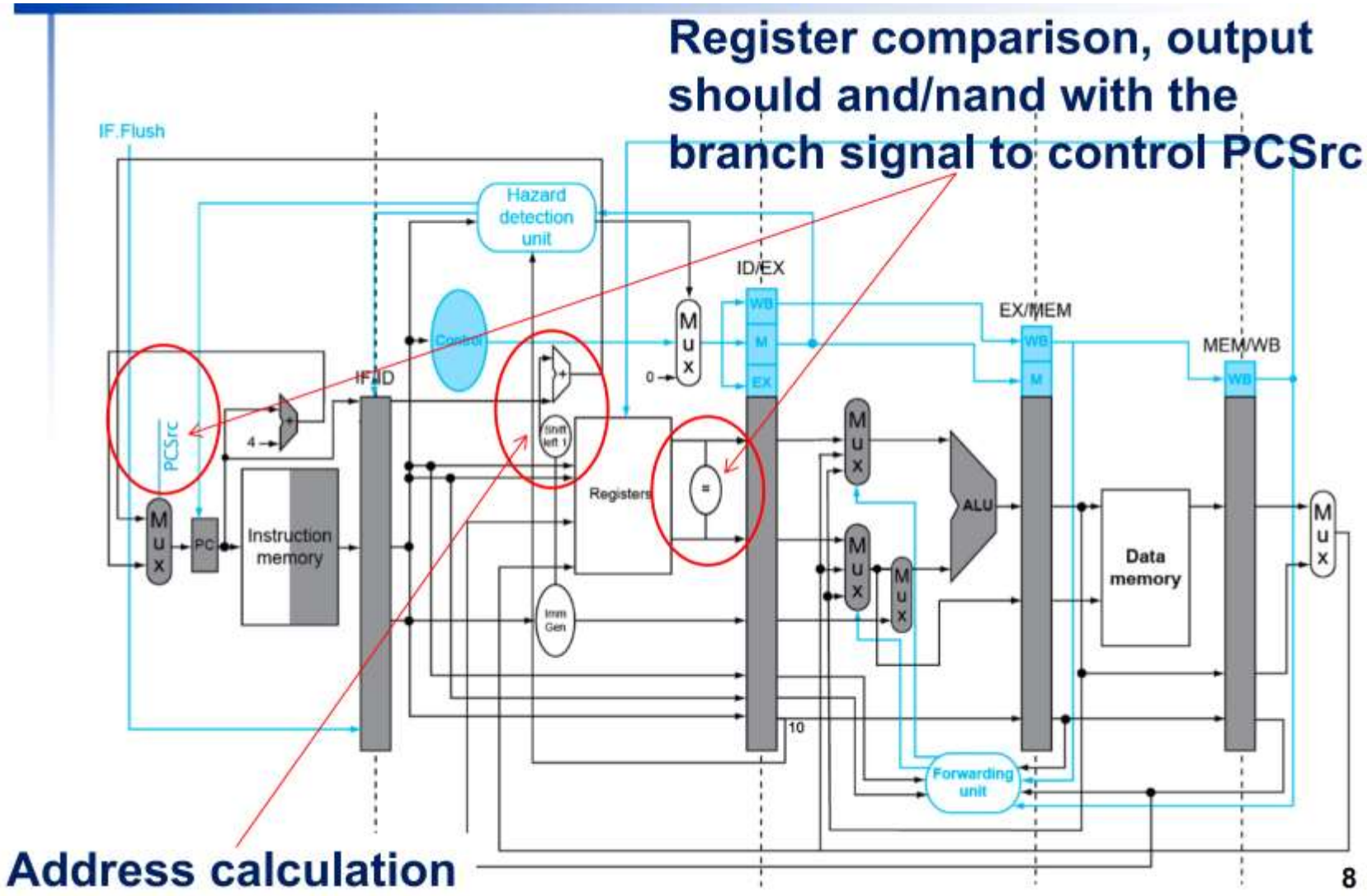


Control Hazard

- Wait until branch outcome is determined before fetching the next instruction



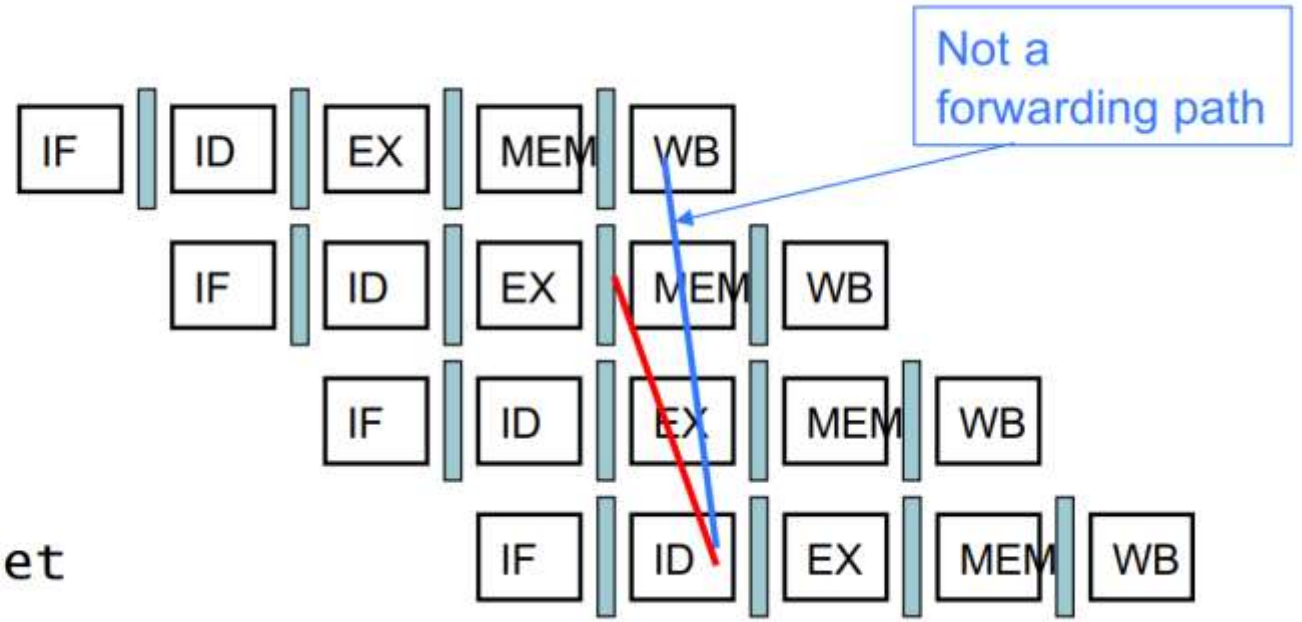
Control Hazard



Only one stall

Control Hazard

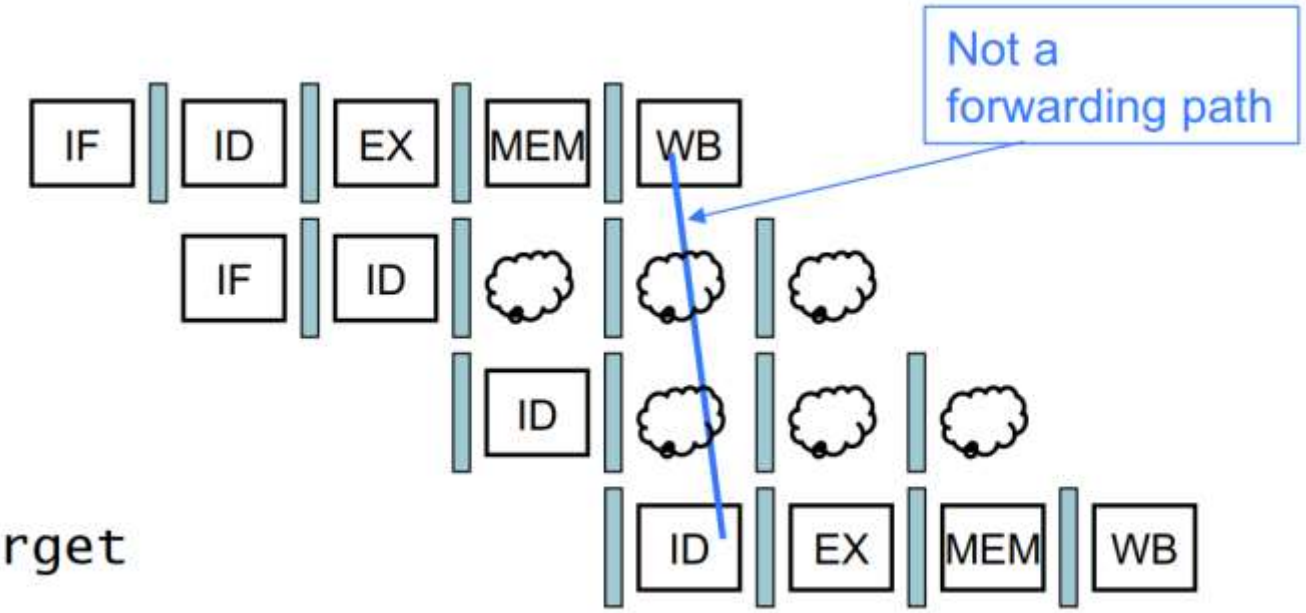
```
add x1, x2, x3
add x4, x5, x6
...
beq x1, x4, target
```



Need one stall in between

Control Hazard

```
lw   x1, addr
beq  stalled
beq  stalled
beq  x1, x0, target
```

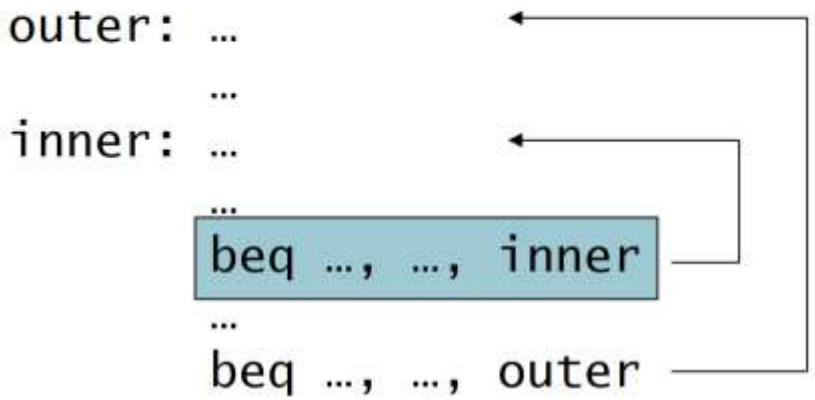


Need two stalls in between

Dynamic prediction

1-bit Predictor

Inner loop branches mispredicted twice!



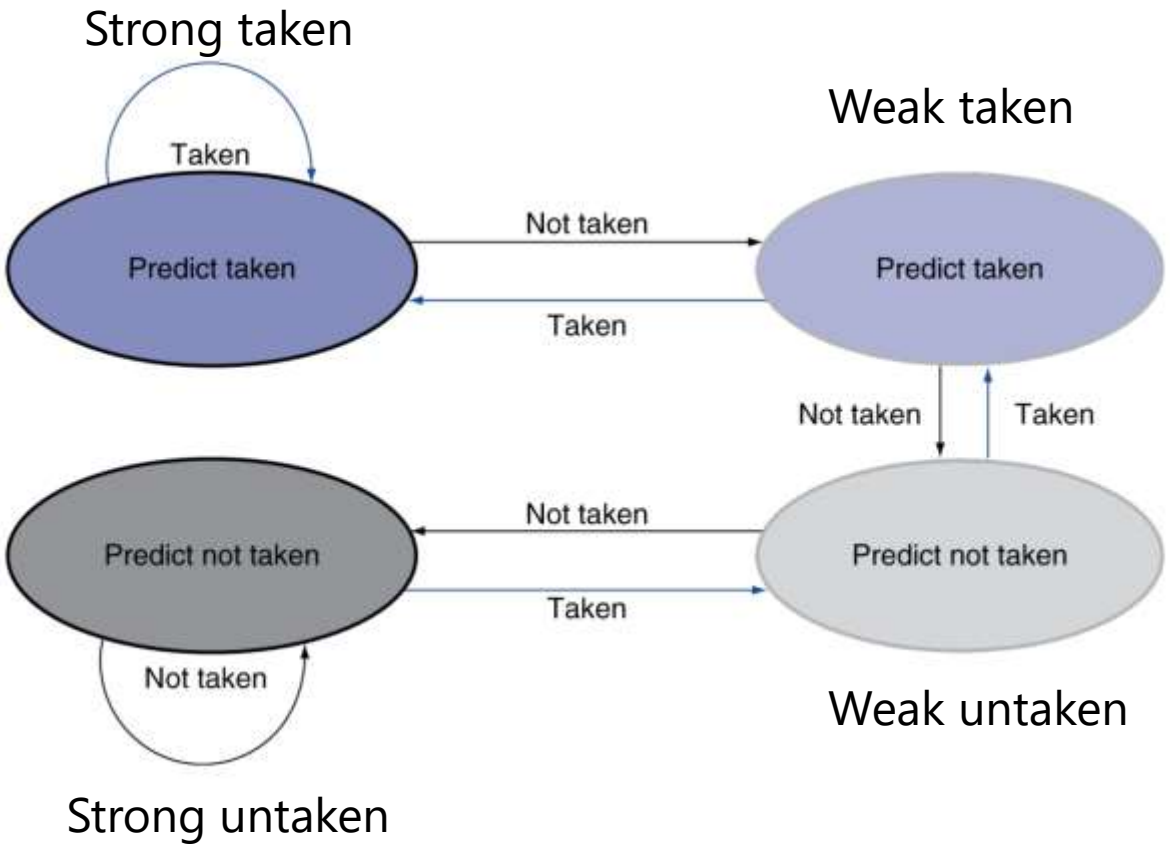
instruction	prediction
beq (inner)	
beq (outer)	

prediction=1: taken
prediction=0: not taken

- Mispredict as taken on last iteration of inner loop
- Then mispredict as not taken on first iteration of inner loop next time around

Dynamic prediction

2-bit Predictor



That's all for today's RC