Control Hazard

Outline

- Pipeline, Pipelined datapath
- Dependences, Hazards
  - Structural, Data - Stalling, Forwarding
- Control Hazards
- Branch prediction

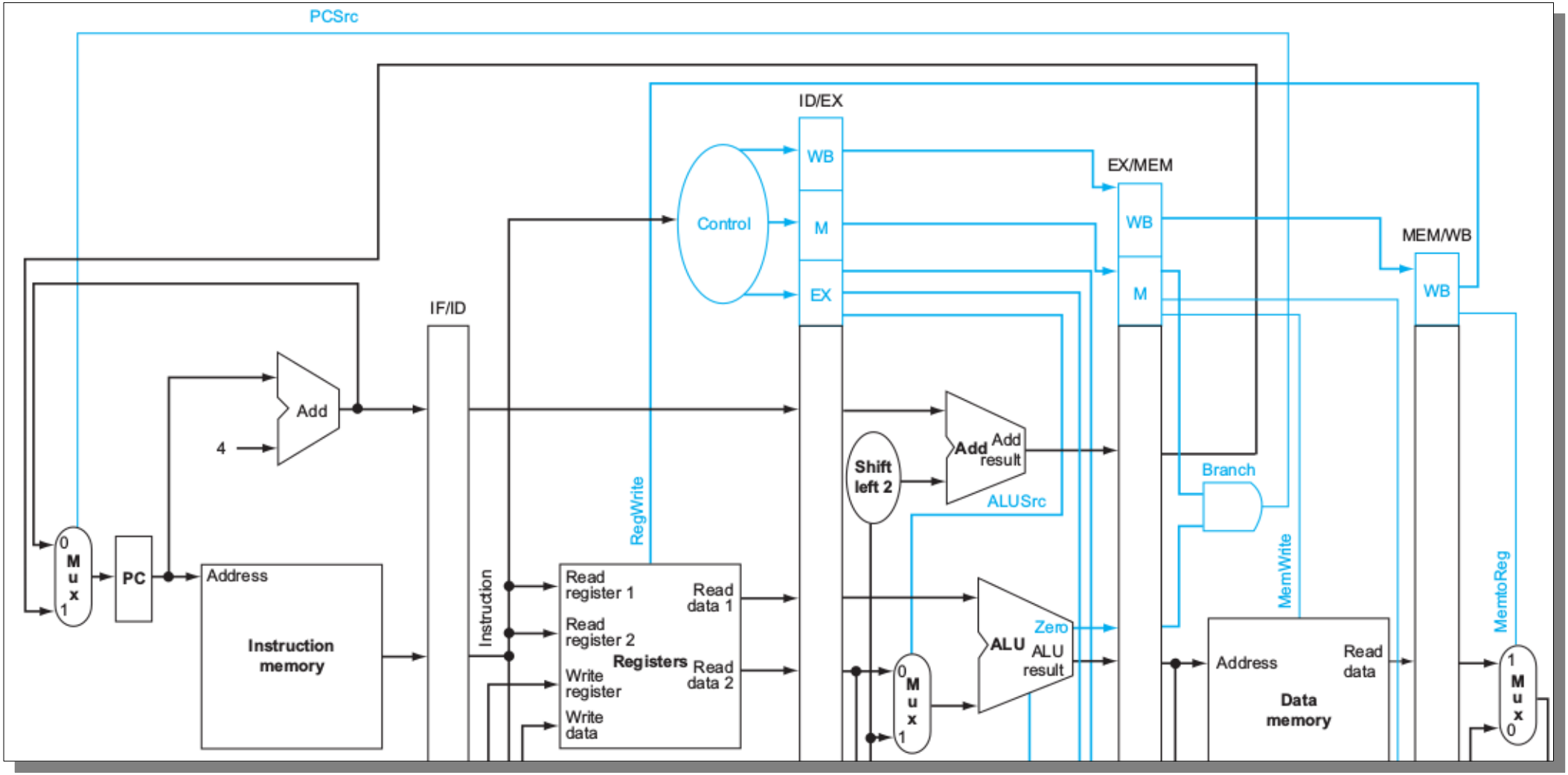
Control Hazard

- Arise from the pipelining of branches and other instructions that change the PC

Control Hazard

- Arise from the pipelining of branches and other instructions that change the PC
- Also called Branch Hazards

Branch Evaluation

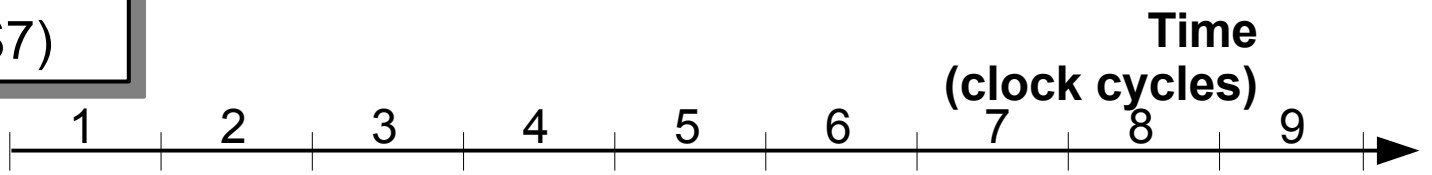


Control Hazard

```
beq     $1, $3, 28
and     $12, $2, $5
or      $13, $6, $2
add     $14, $2, $2
...
lw      $4, 50($7)
```

Control Hazard

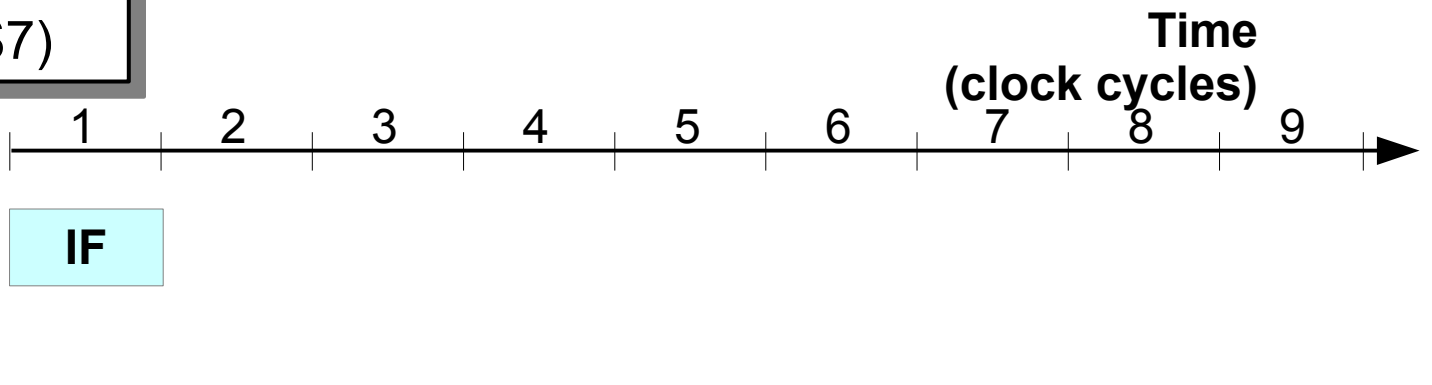
```
beq     $1, $3, 28
and     $12, $2, $5
or      $13, $6, $2
add     $14, $2, $2
...
lw      $4, 50($7)
```



beq

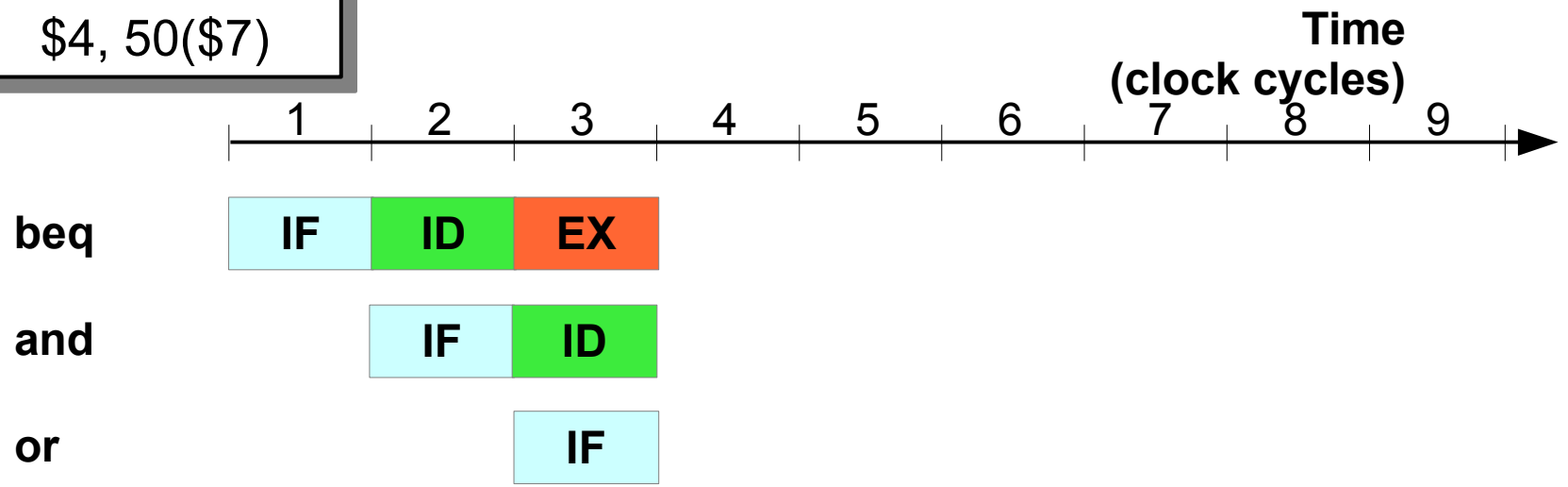
Control Hazard

```
beq     $1, $3, 28
and     $12, $2, $5
or      $13, $6, $2
add     $14, $2, $2
...
lw      $4, 50($7)
```



Control Hazard

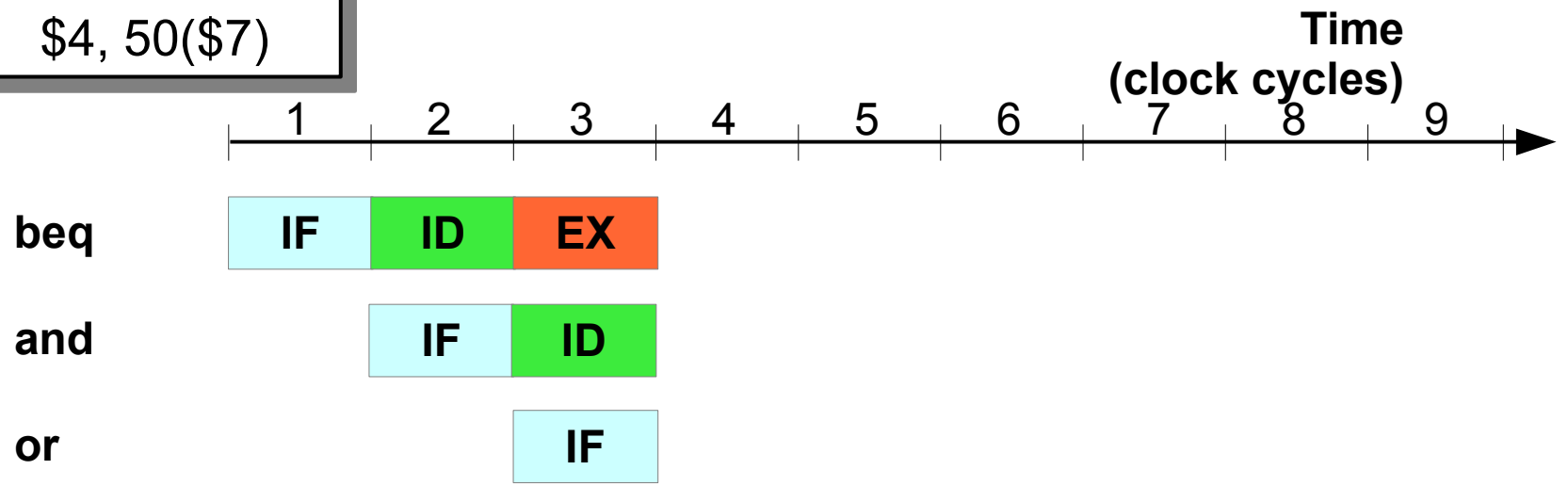
```
beq     $1, $3, 28
and     $12, $2, $5
or      $13, $6, $2
add     $14, $2, $2
...
lw      $4, 50($7)
```



Control Hazard

```
beq     $1, $3, 28
and     $12, $2, $5
or      $13, $6, $2
add     $14, $2, $2
...
lw      $4, 50($7)
```

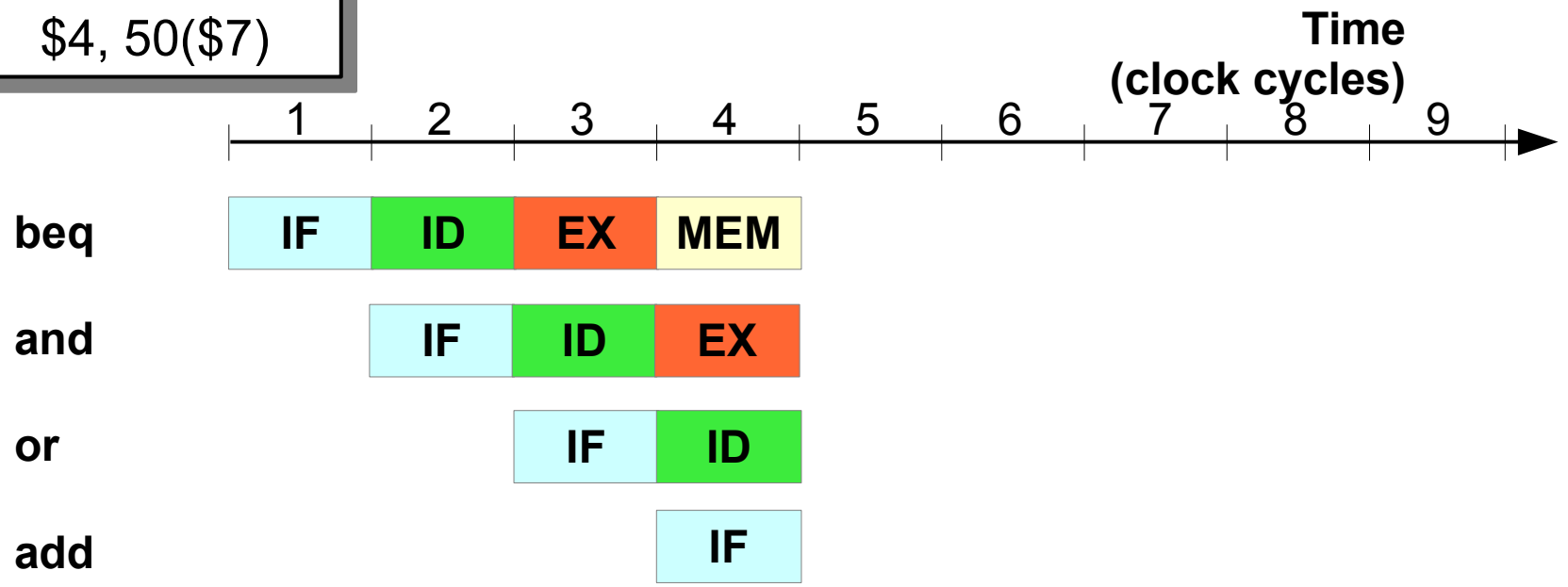
Branch Taken



Control Hazard

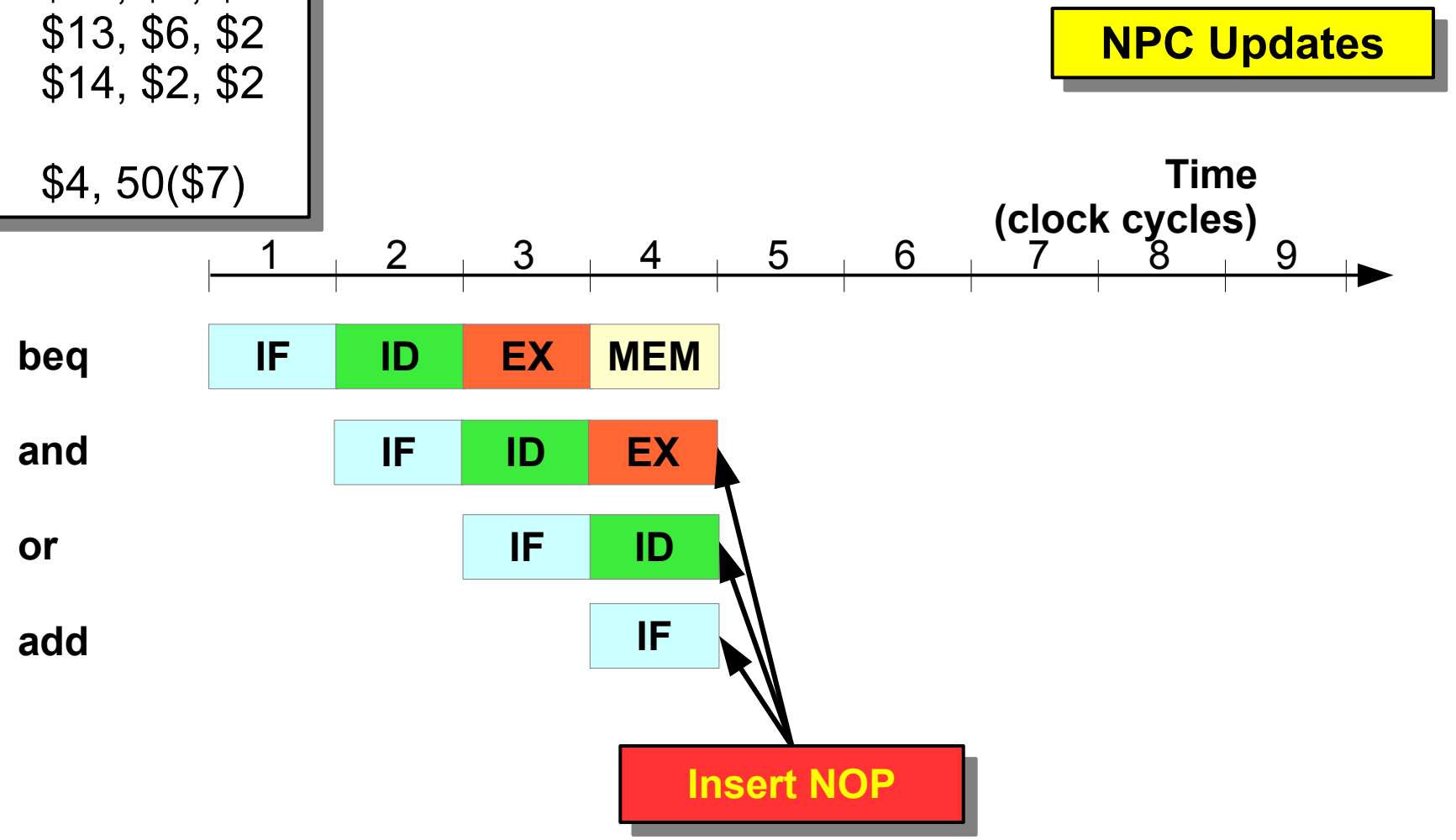
```
beq     $1, $3, 28
and     $12, $2, $5
or      $13, $6, $2
add     $14, $2, $2
...
lw      $4, 50($7)
```

NPC Updates



Control Hazard

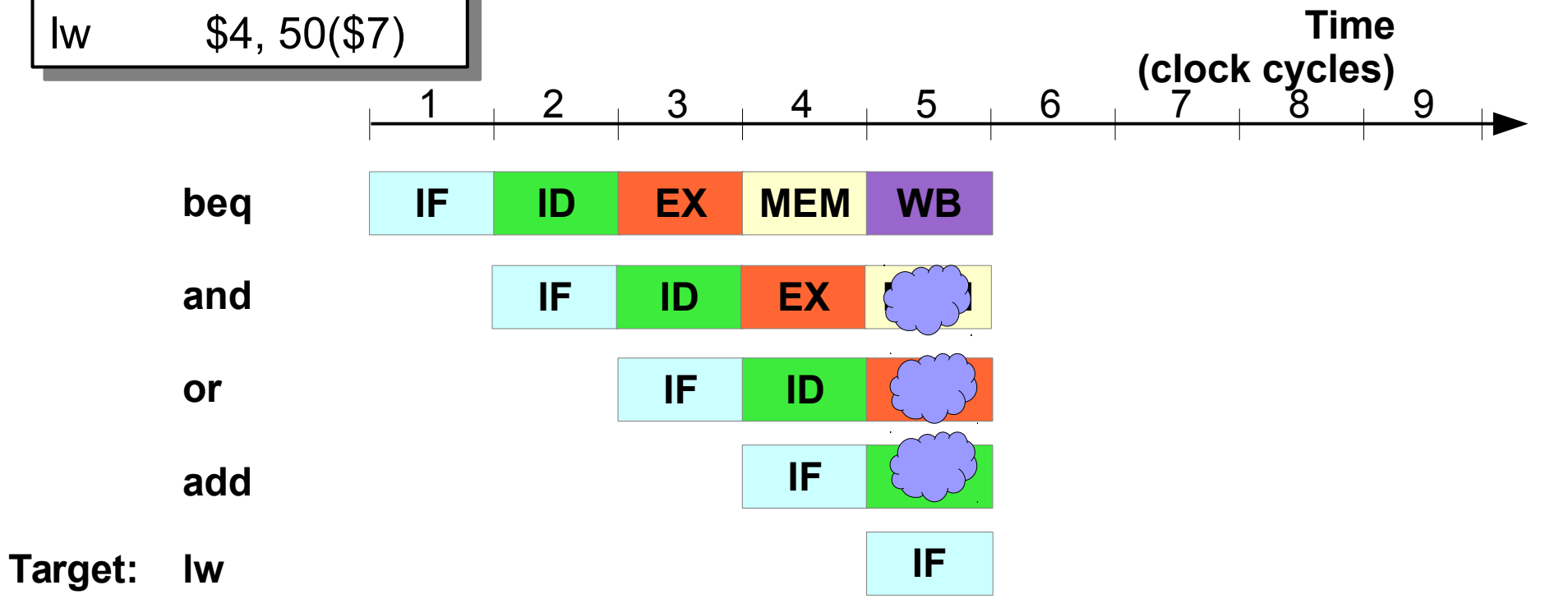
```
beq     $1, $3, 28
and     $12, $2, $5
or      $13, $6, $2
add     $14, $2, $2
...
lw      $4, 50($7)
```



Control Hazard

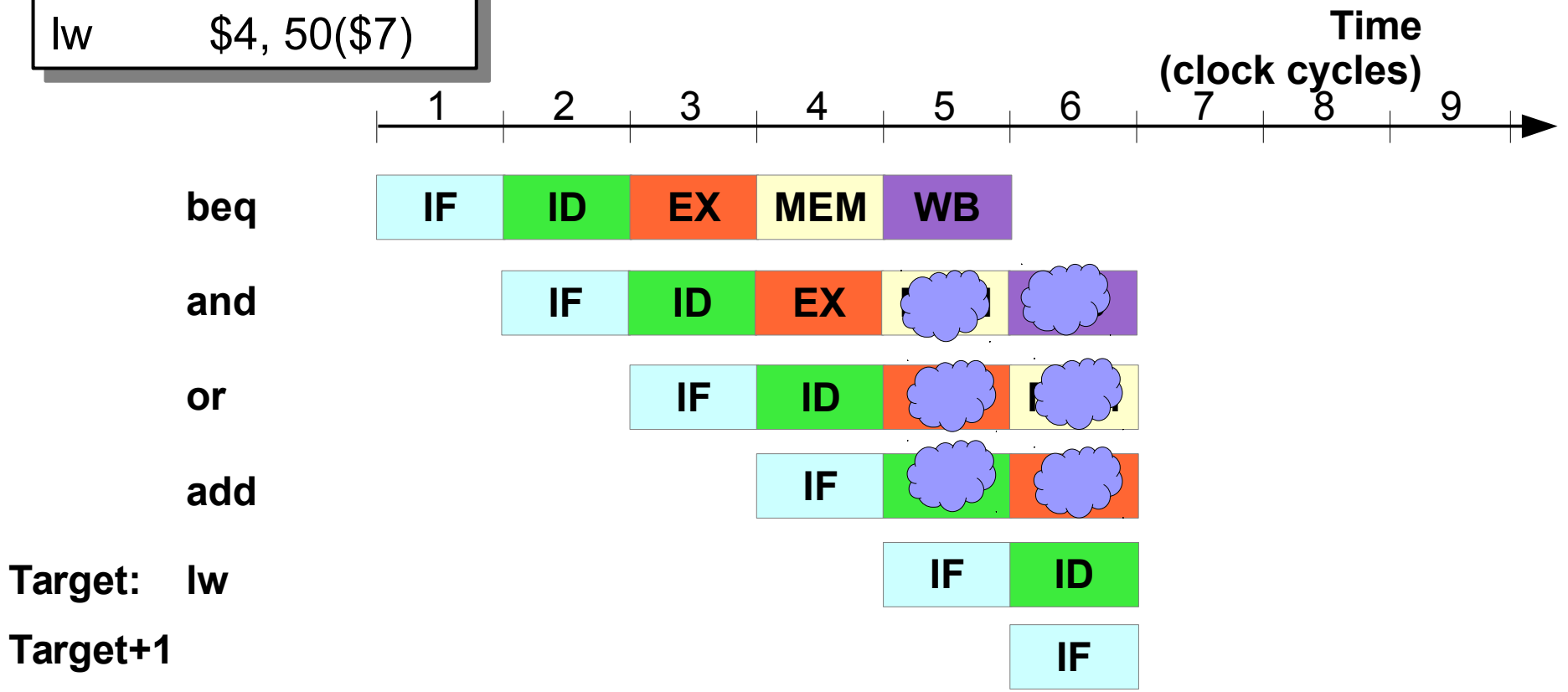
```
beq     $1, $3, 28
and     $12, $2, $5
or      $13, $6, $2
add     $14, $2, $2
...
lw      $4, 50($7)
```

Branch target enters the pipeline



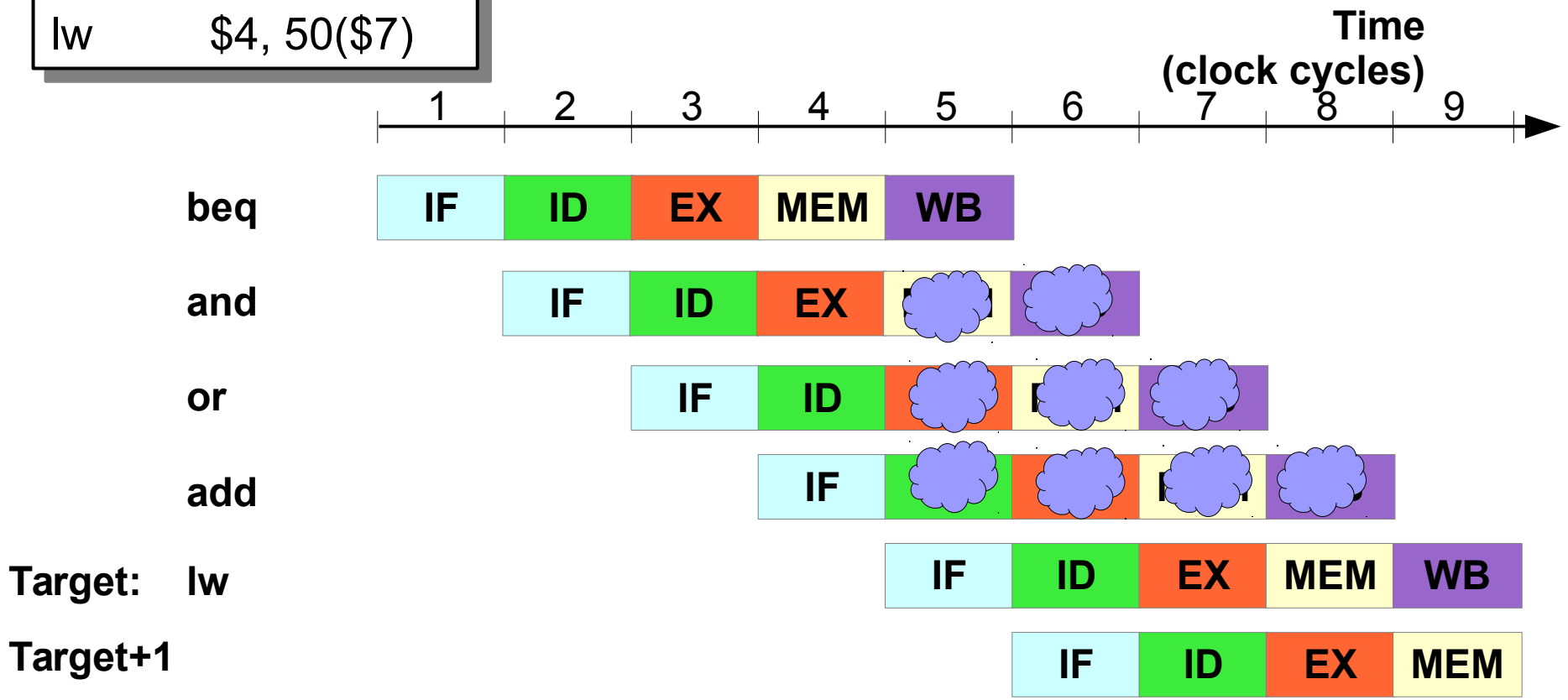
Control Hazard

```
beq     $1, $3, 28
and     $12, $2, $5
or      $13, $6, $2
add     $14, $2, $2
...
lw      $4, 50($7)
```



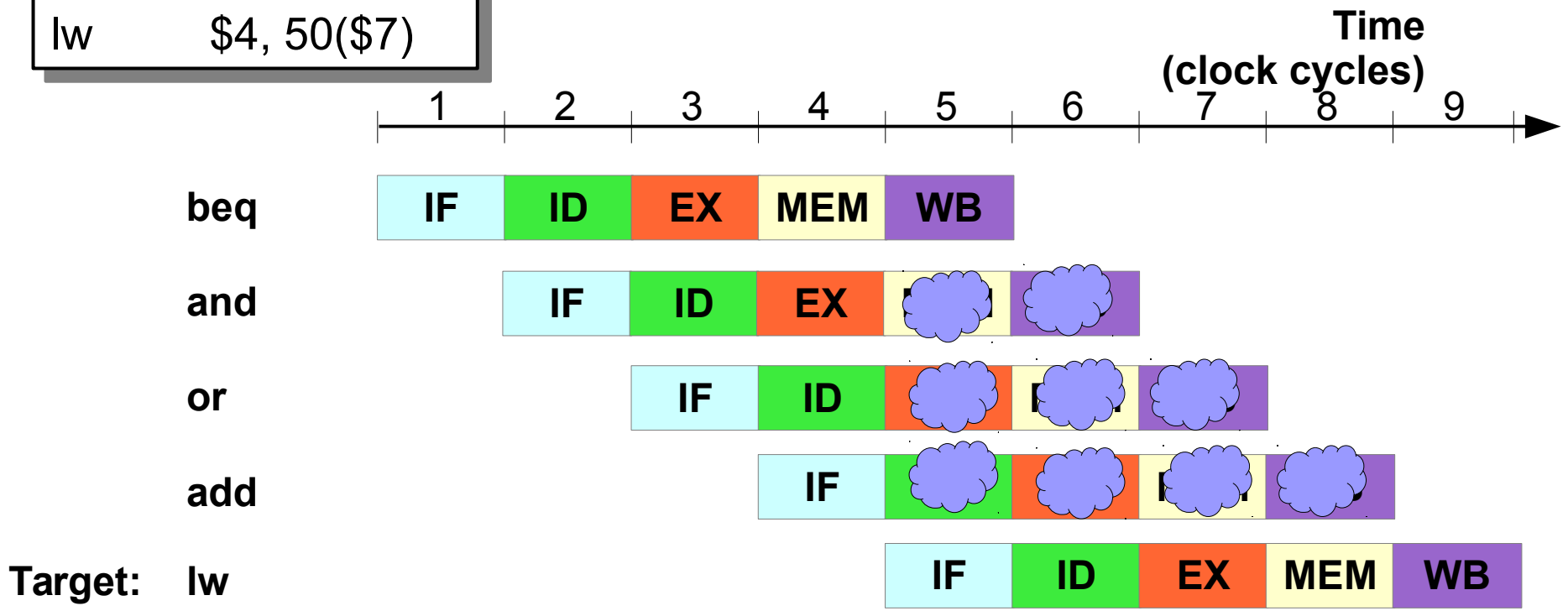
Control Hazard

```
beq     $1, $3, 28
and     $12, $2, $5
or      $13, $6, $2
add     $14, $2, $2
...
lw      $4, 50($7)
```



Control Hazard

```
beq     $1, $3, 28
and     $12, $2, $5
or      $13, $6, $2
add     $14, $2, $2
...
lw      $4, 50($7)
```



Branch Penalty = 3 cycles

Control Hazard

- Branch evaluation occurs in EX stage.
- PC updates in MEM stage.
- Branch target loads in the next cycle
- The delay in determining the proper instruction to fetch is called a Control Hazard or Branch Hazard.

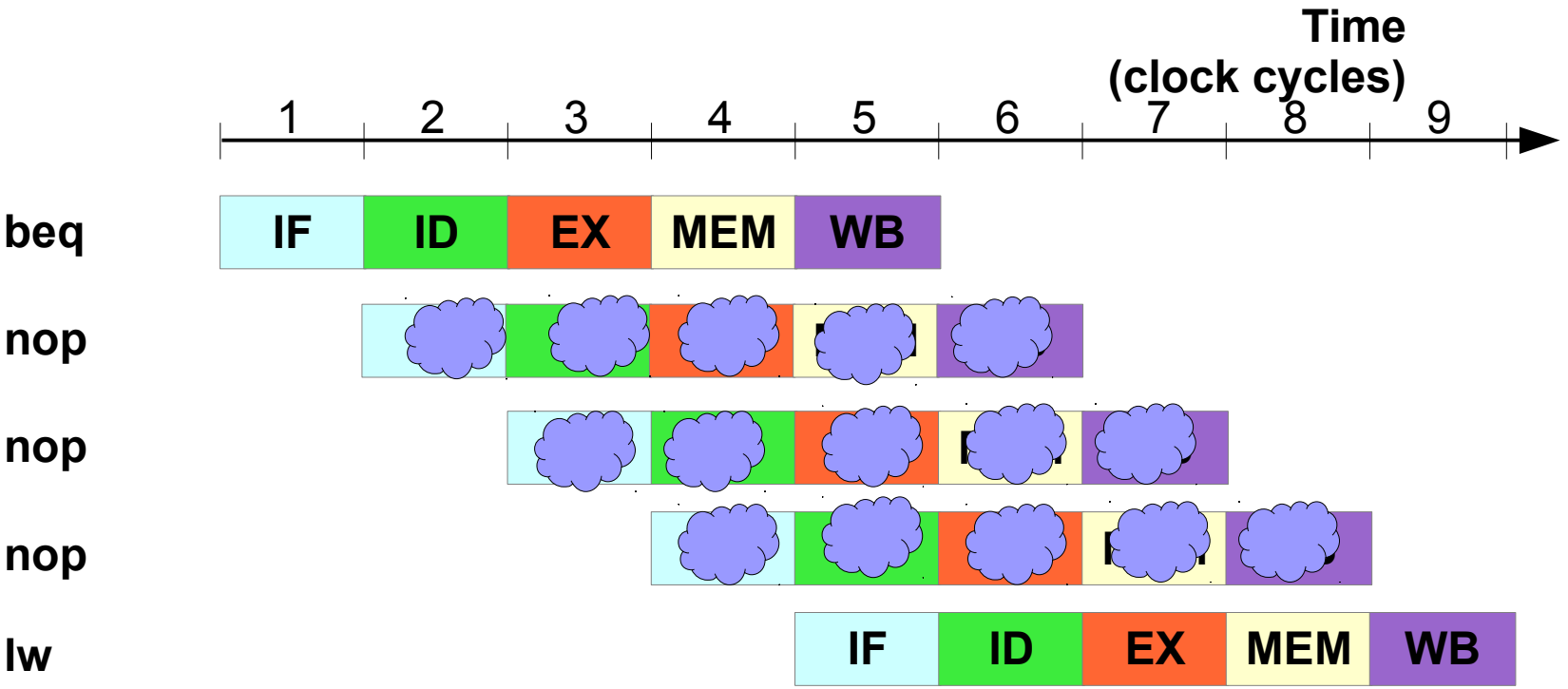
Reducing Pipeline Branch Penalties

Reducing Pipeline Branch Penalties

- Freeze the pipeline

Reducing Pipeline Branch Penalties

- Freeze the pipeline



Reducing Branch Delay

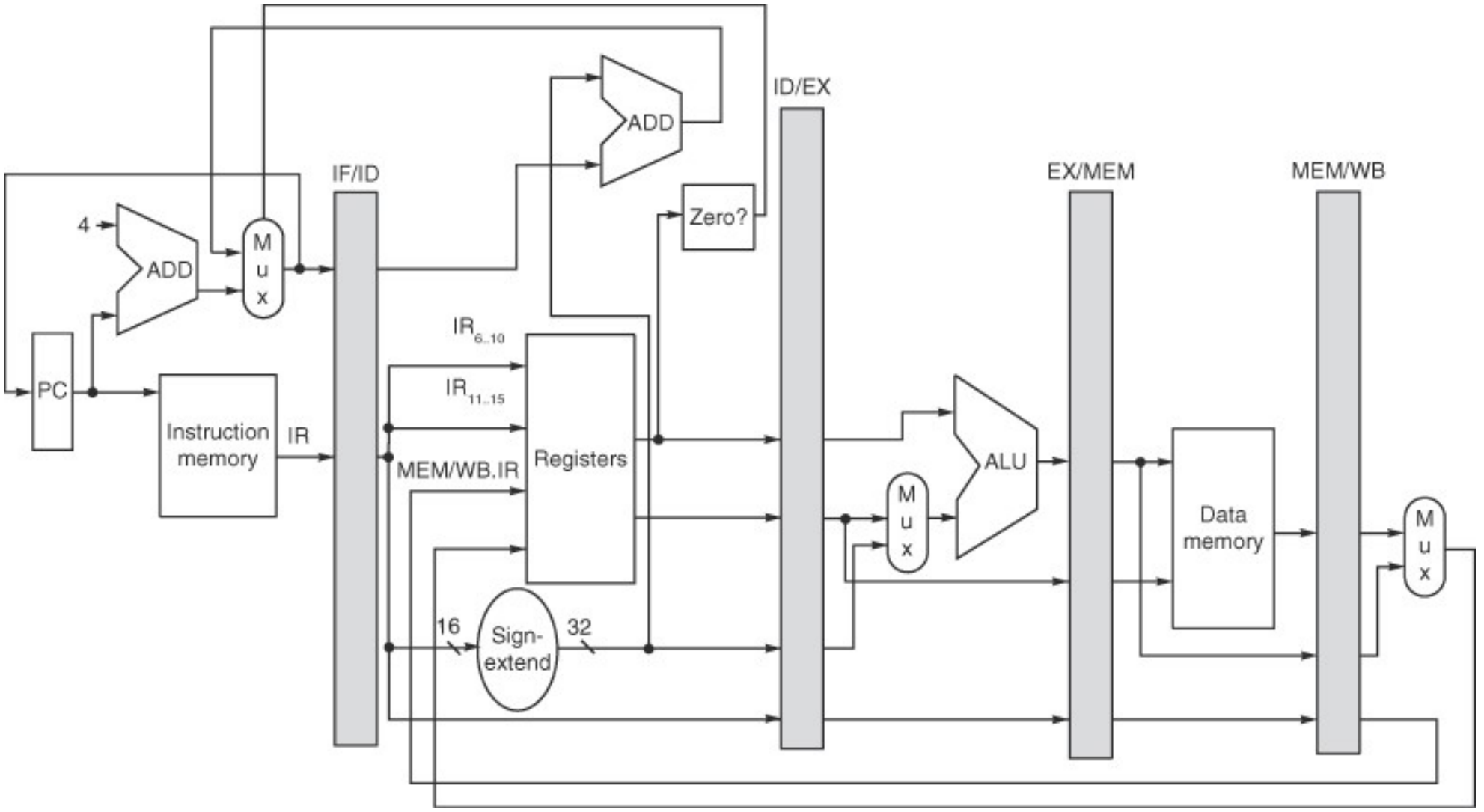
Reducing Branch Delay

- Most branches rely on simple tests (equality or sign)
- Move the branch decision up
-

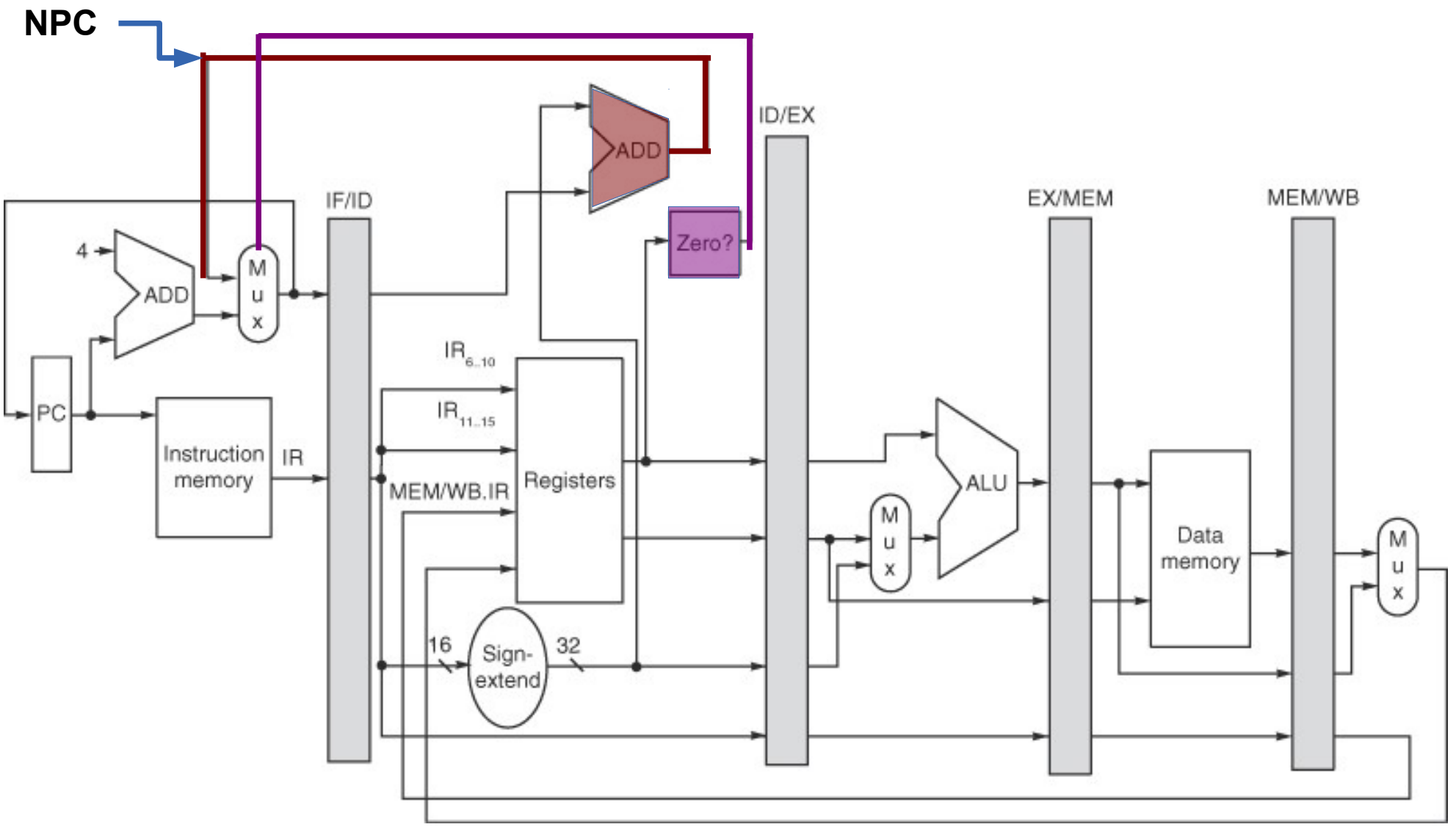
Reducing Branch Delay

- Most branches rely on simple tests (equality or sign)
- Move the branch decision up
  - Compute NPC earlier
  - Evaluate branch at the earliest

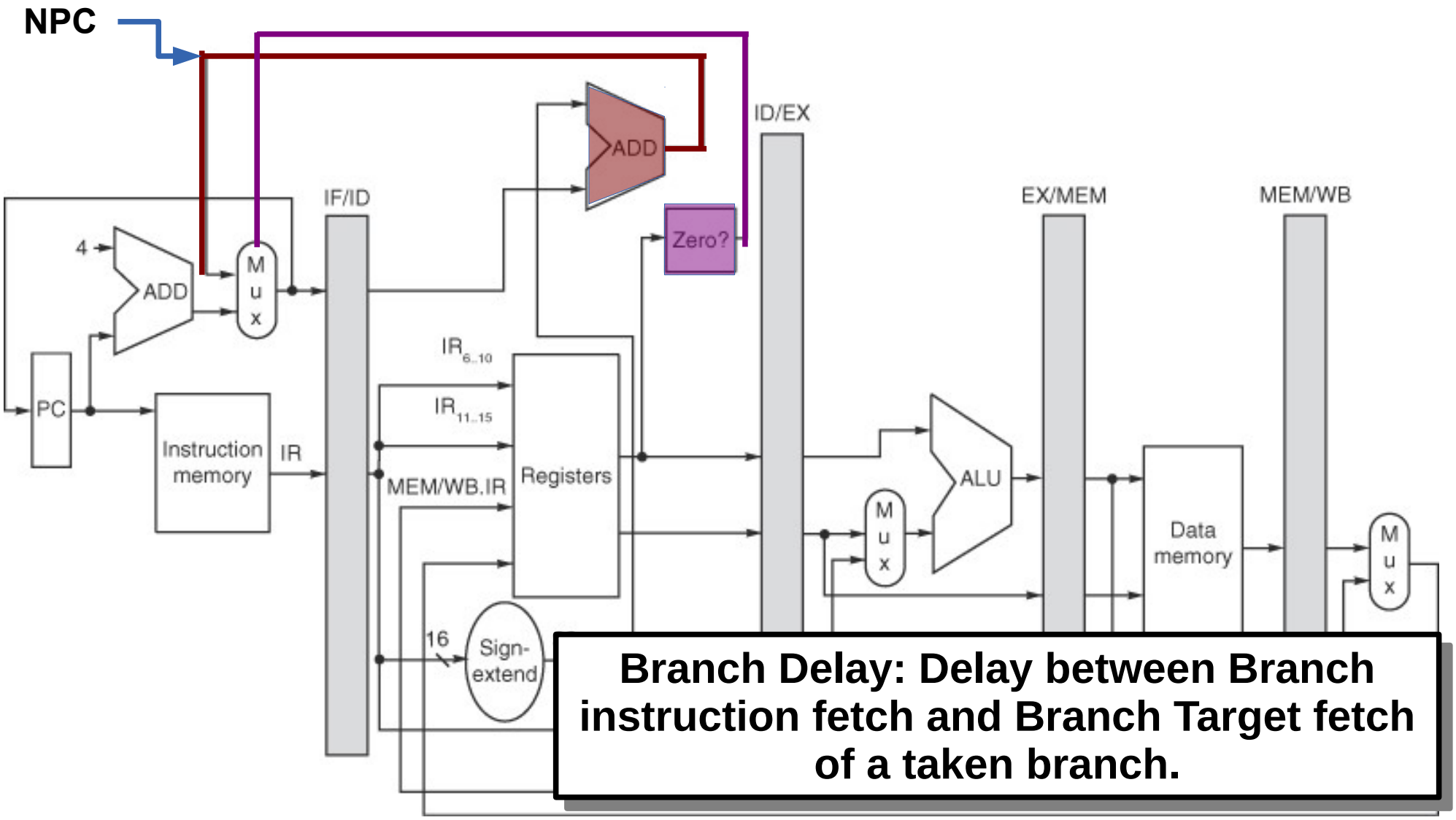
Reducing Branch Delay



Reducing Branch Delay

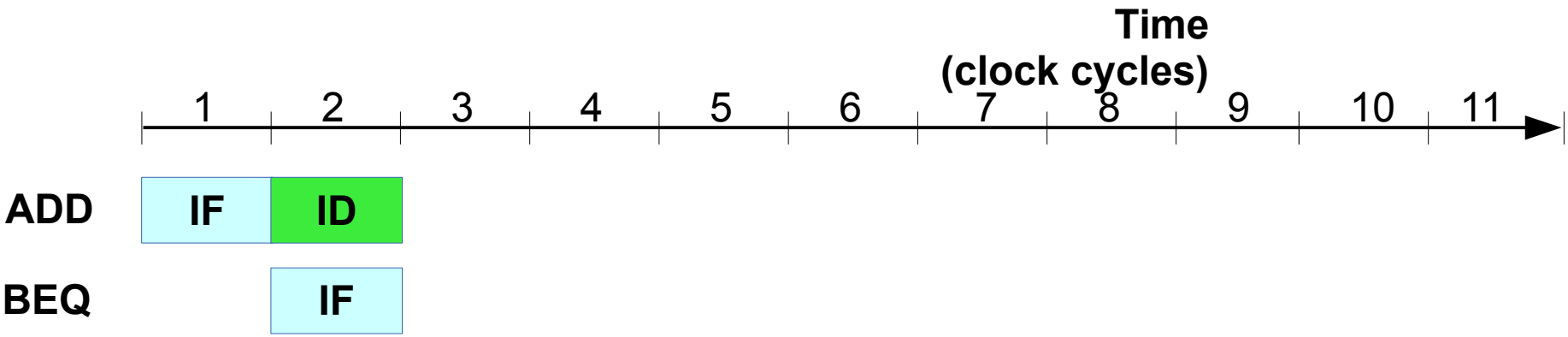


Reducing Branch Delay



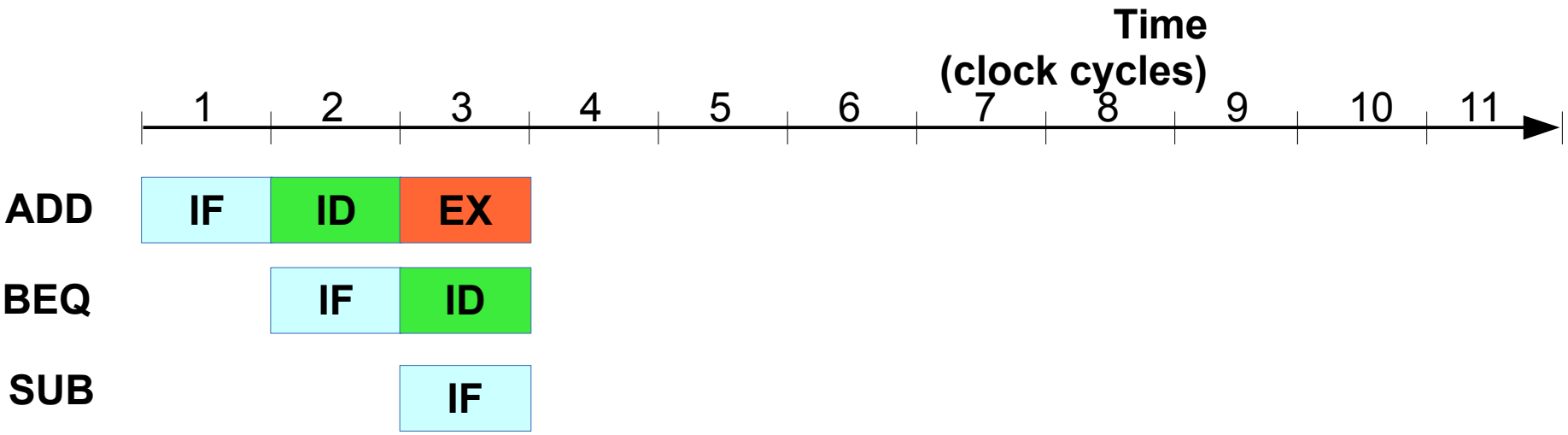
Branch Delay

```
ADD   R2, R3,R4
BEQ   R2, R4, loop
SUB   R5, R5,R4
...
```



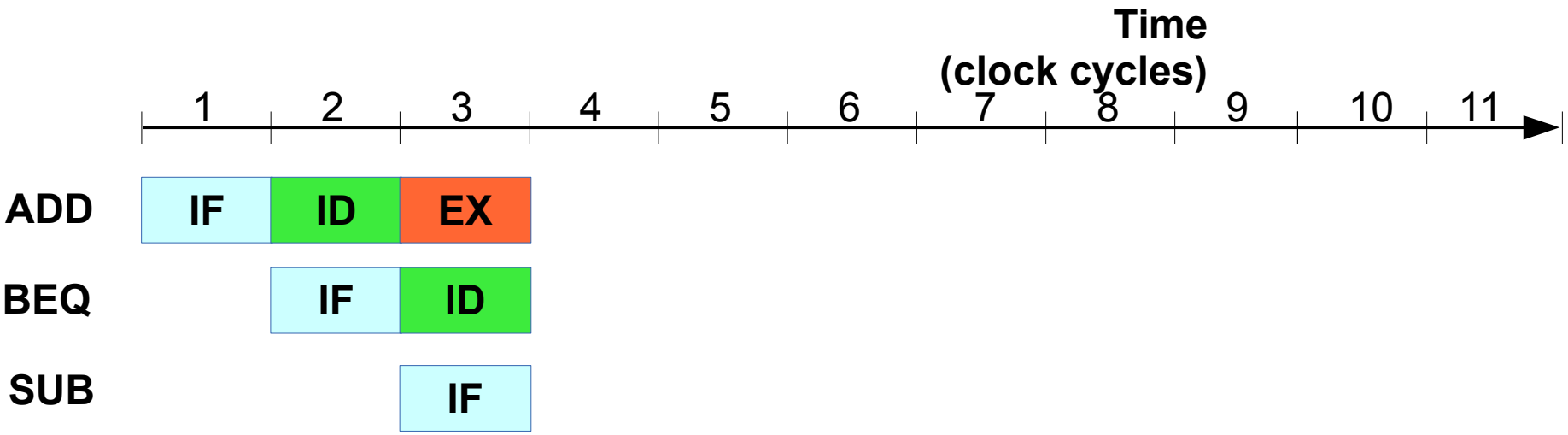
Branch Delay

```
ADD   R2, R3,R4
BEQ   R2, R4, loop
SUB   R5, R5,R4
...
```



Branch Delay

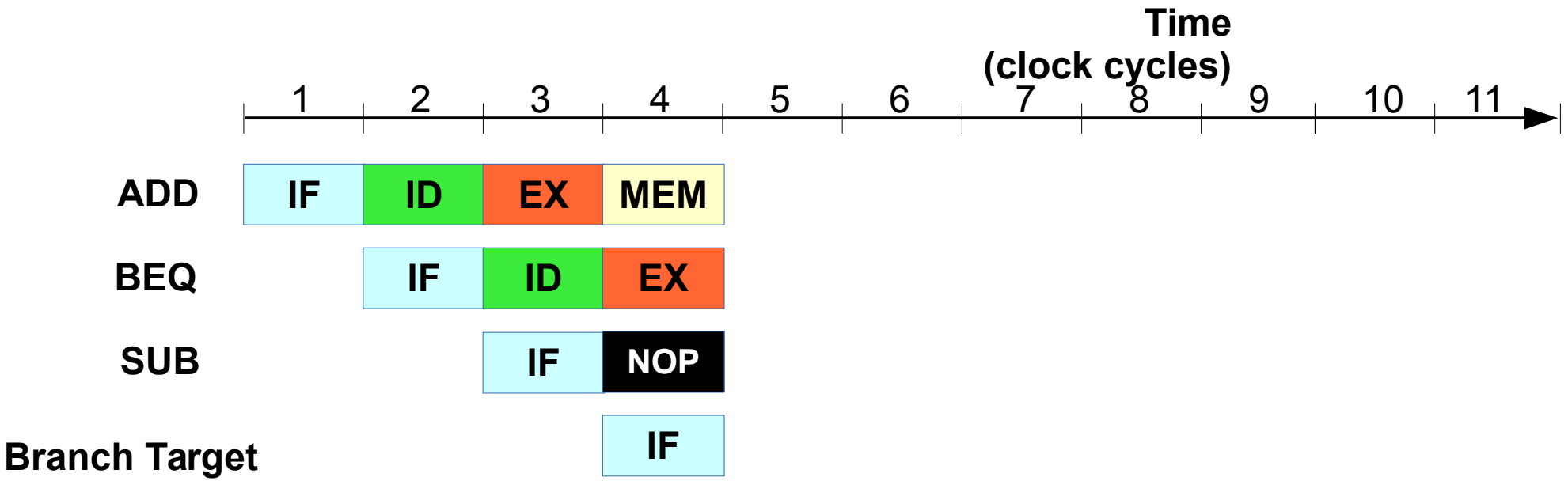
```
ADD   R2, R3,R4
BEQ   R2, R4, loop
SUB   R5, R5,R4
...
```



PC = PC + Offset

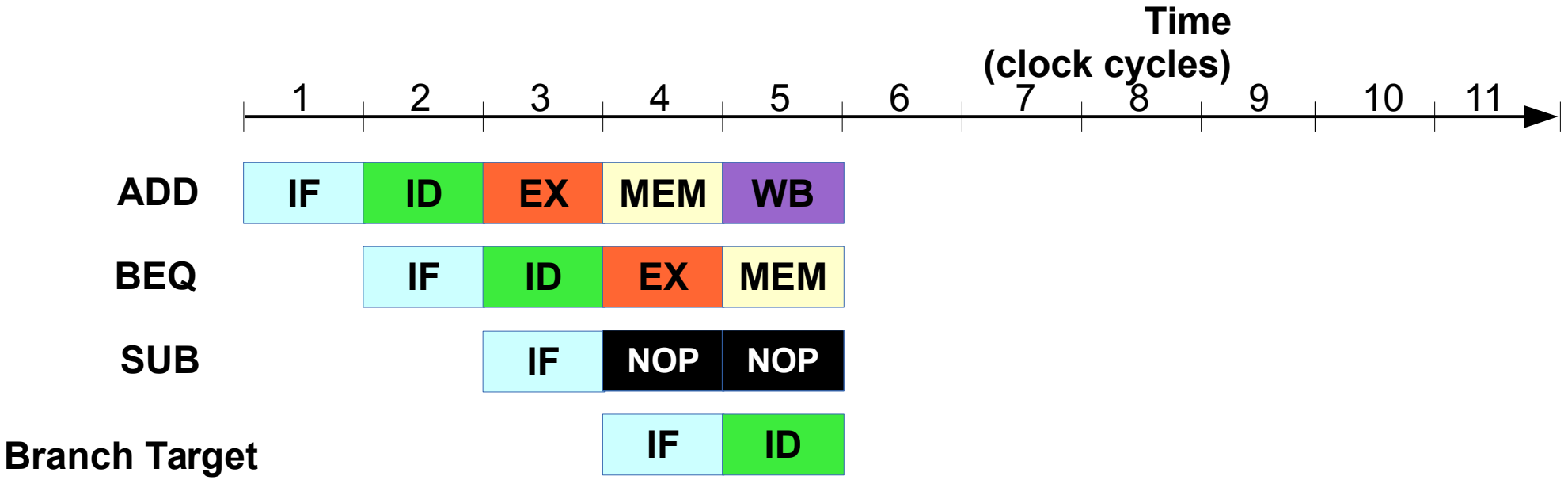
Branch Delay

```
ADD   R2, R3,R4
BEQ   R2, R4, loop
SUB   R5, R5,R4
...
```



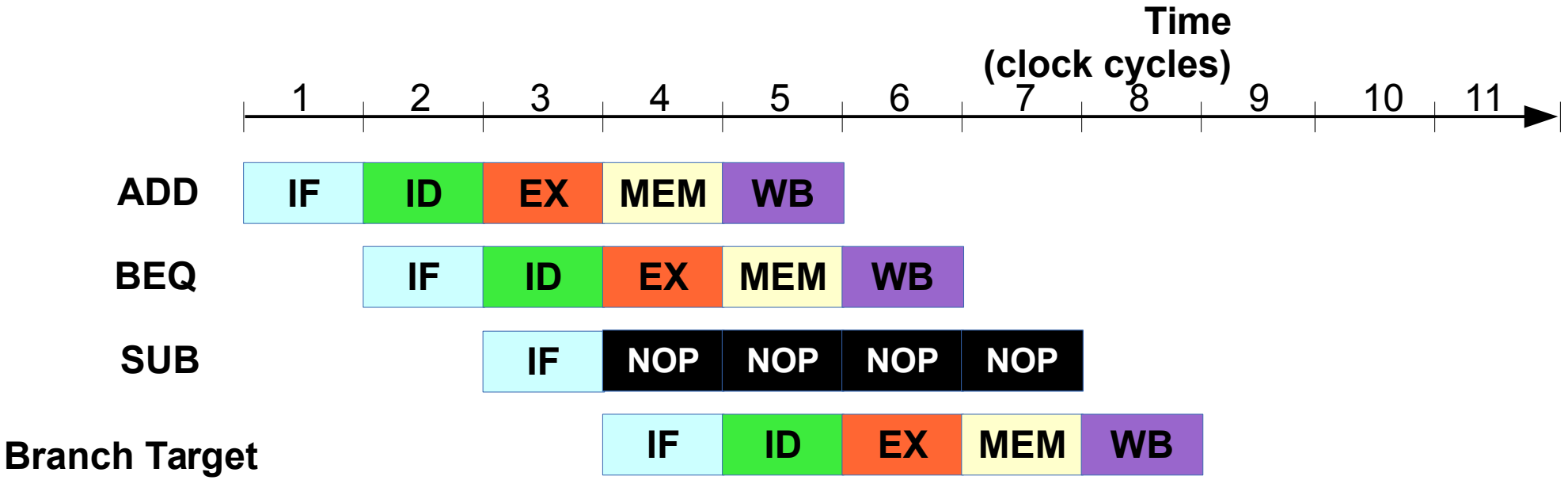
Branch Delay

```
ADD   R2, R3,R4
BEQ   R2, R4, loop
SUB   R5, R5,R4
...
```

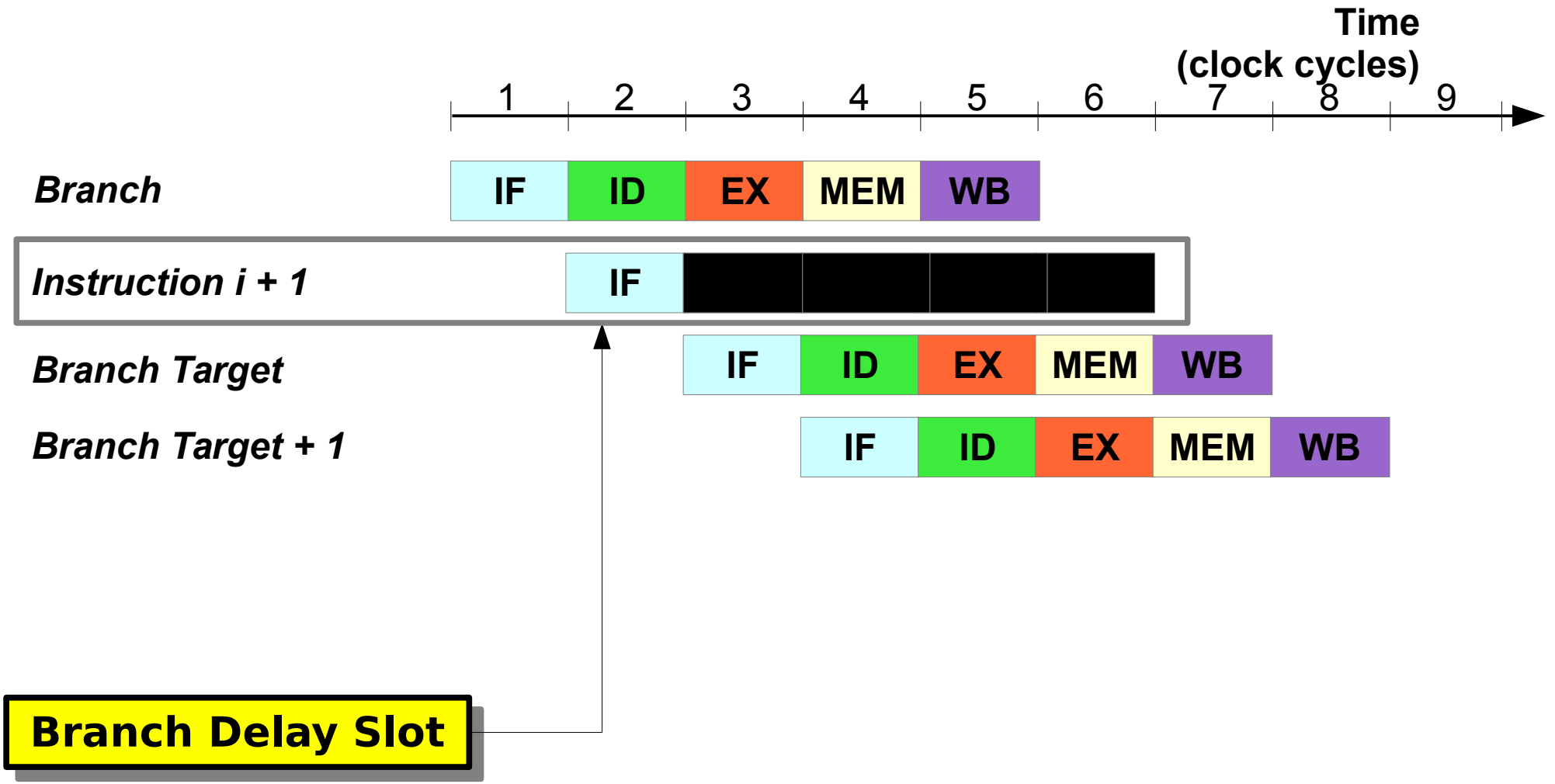


Branch Delay

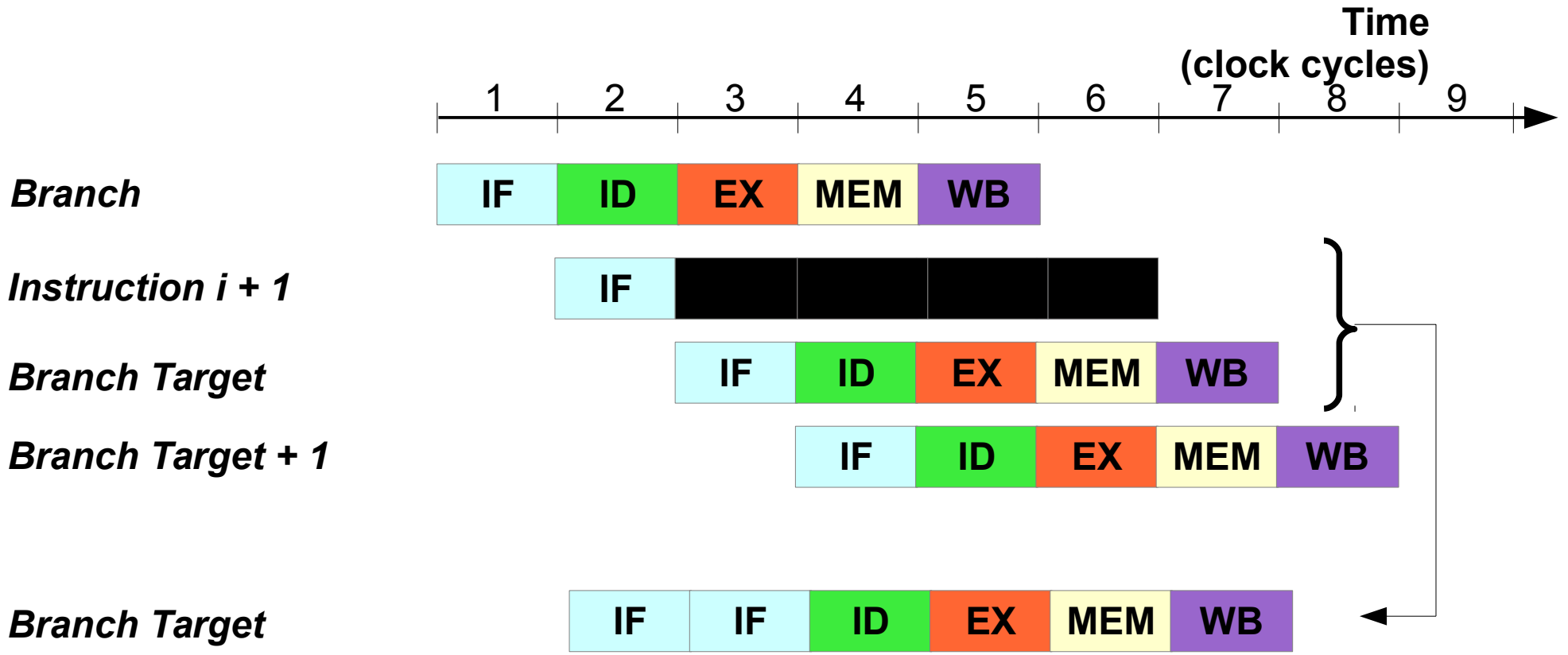
```
ADD    R2, R3,R4
BEQ    R2, R4, loop
SUB    R5, R5,R4
...
```



Branch Delay Slot



Branch Hazards

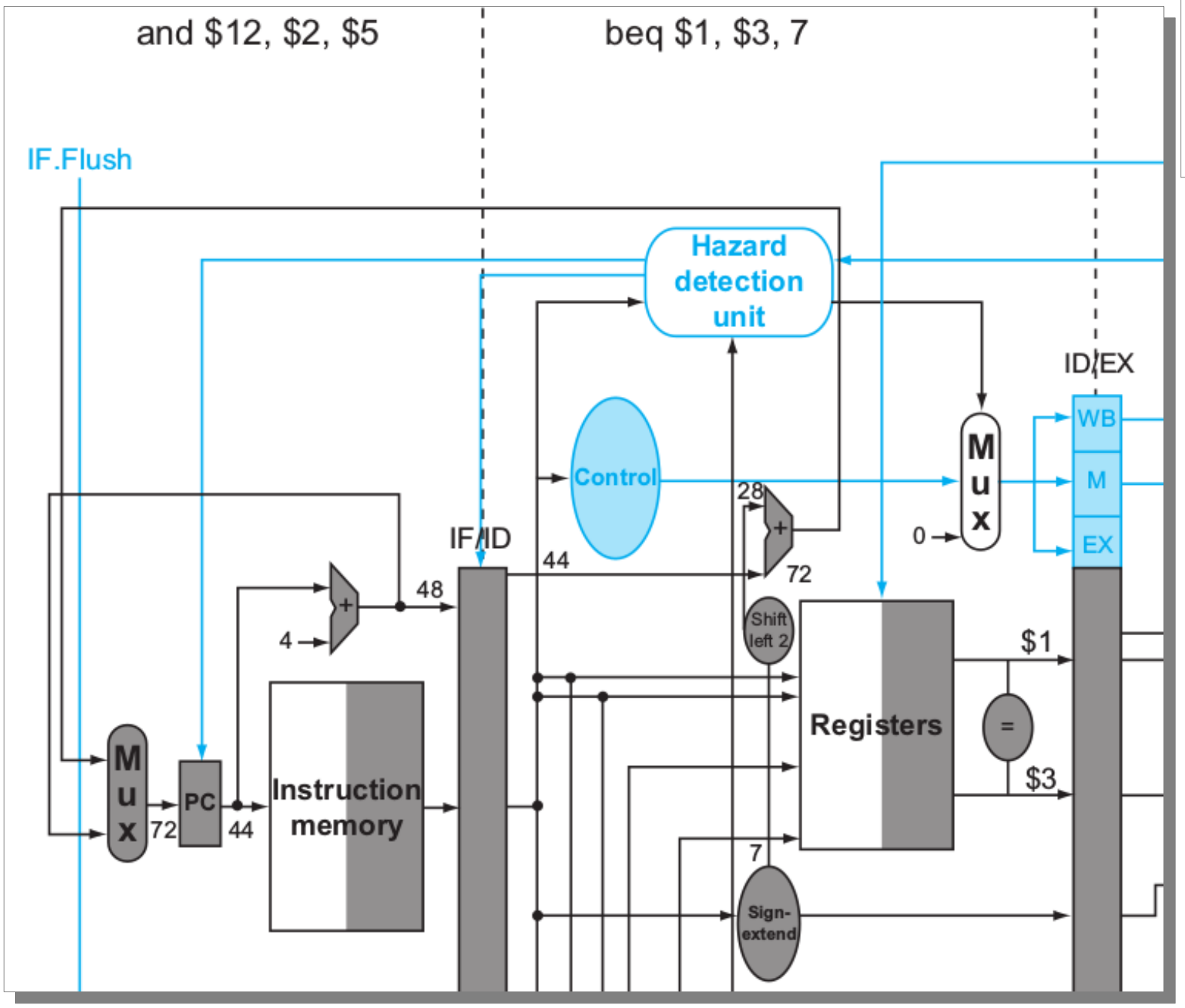


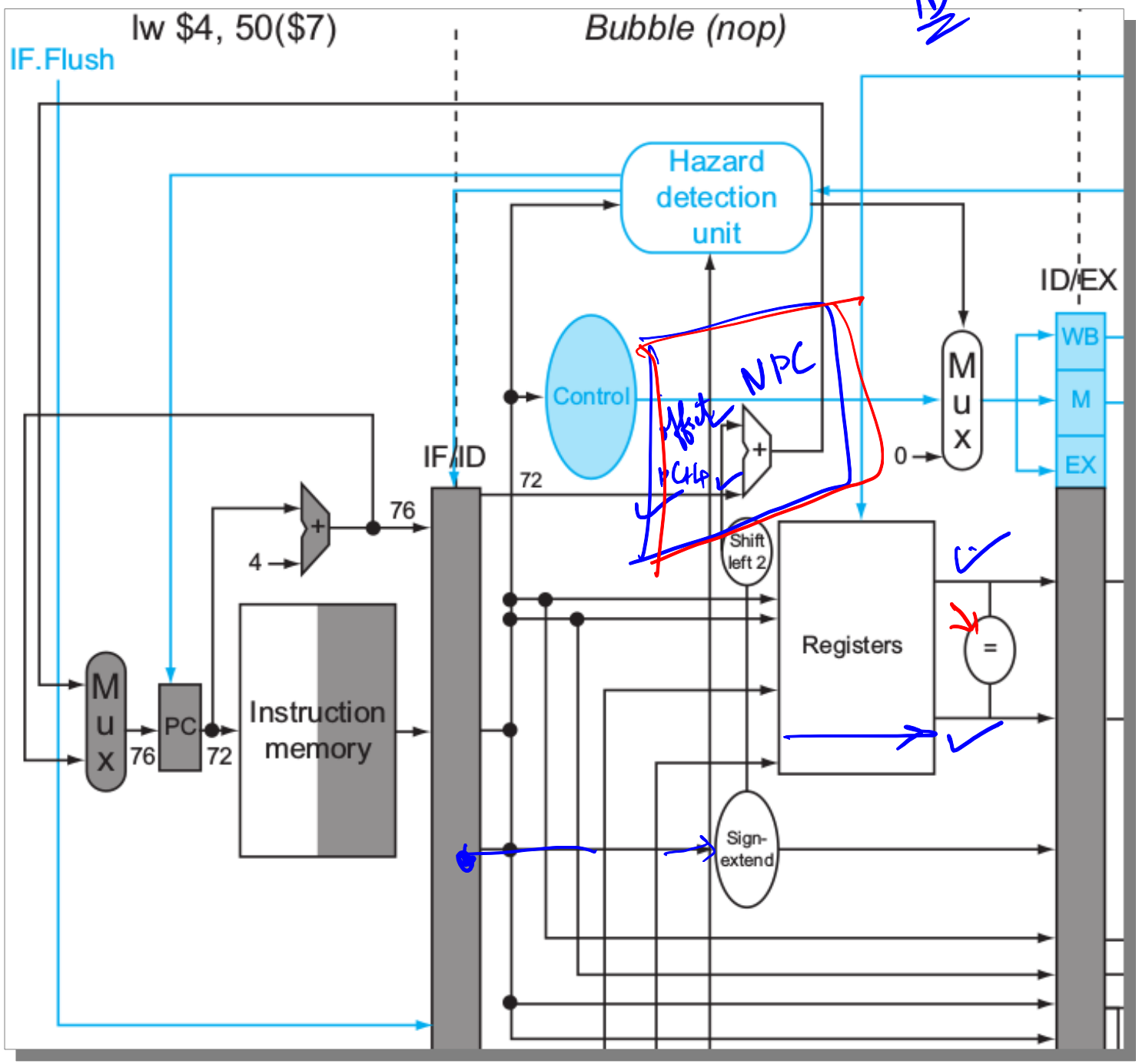
- 1 stall cycle for every branch yields a performance loss of 10% to 30%!

Example – Branch Taken

```
36 sub $10, $4, $8
40 beq $1, $3, 7
44 and $12, $2, $5
48 or  $13, $2, $6
52 add $14, $4, $2
56 slt $15, $6, $7
. . .
72 lw $4, 50($7)
```

```
36 sub $10, $4, $8
40 beq $1, $3, 7 #
44 and $12, $2, $5
48 or  $13, $2, $6
52 add $14, $4, $2
56 slt $15, $6, $7
. . .
72 lw $4, 50($7)
```






```
36 sub $10, $4, $8
40 beq $1, $3, 7 #
44 and $12, $2, $5
48 or  $13, $2, $6
52 add $14, $4, $2
56 slt $15, $6, $7
. . .
72 lw $4, 50($7)
```

Branch Evaluation in ID – Steps

- Are beq inputs ready?

Branch Evaluation in ID – Steps

- Decode the instruction

Branch Evaluation in ID – Steps

- Decode the instruction
  - RRead
- Decide whether to bypass/forward to the equality unit
  - Forwarding Logic has to be modfied

bypass from EX, MEM, WB

Branch Evaluation in ID – Steps

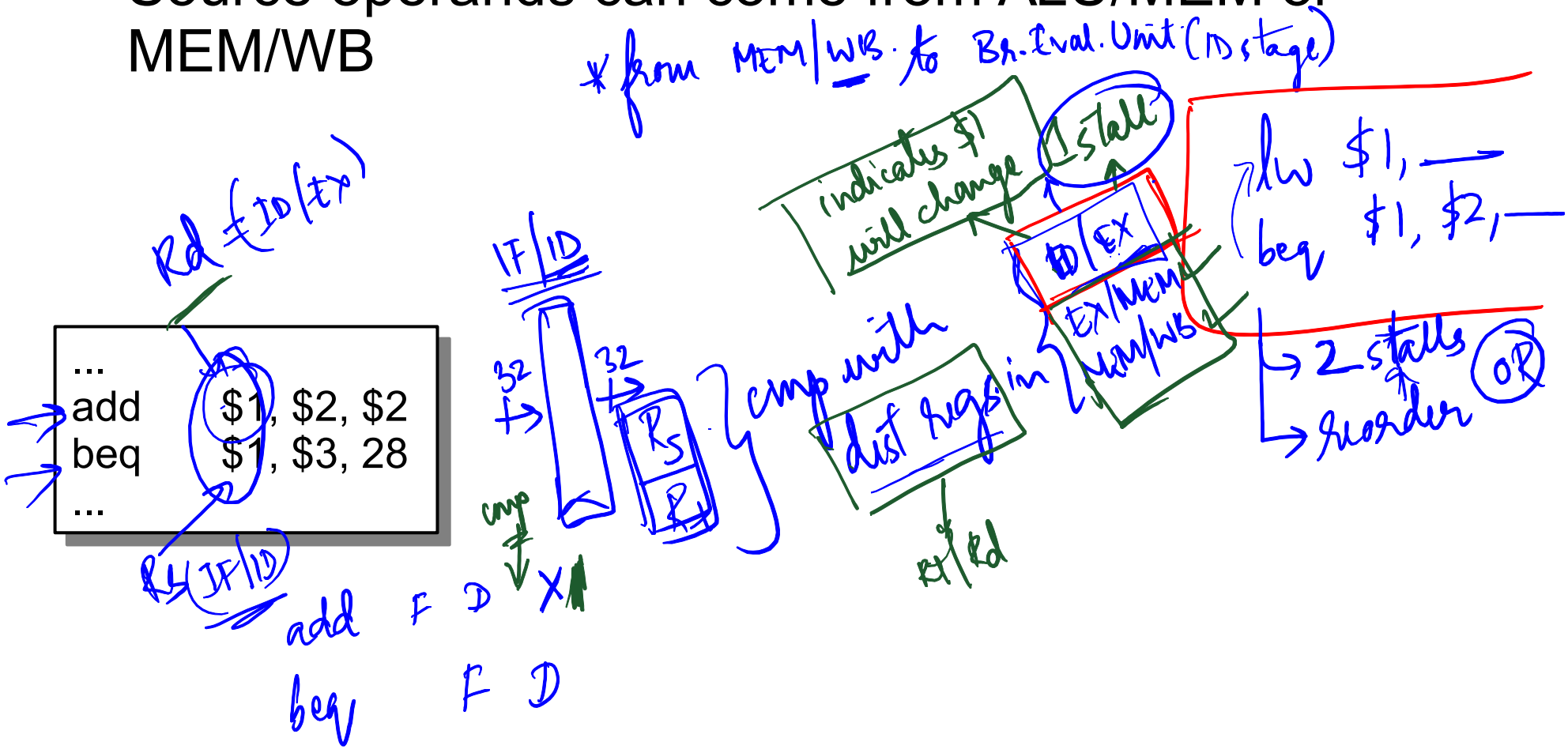
- Decode the instruction
- Decide whether to bypass/forward to the equality unit
  - Forwarding Logic has to be modfied
- Complete the comparison

Branch Evaluation in ID – Steps

- Decode the instruction
- Decide whether to bypass/forward to the equality unit
  - Forwarding Logic has to be modfied
- Complete the comparison
- If branch taken, set the PC to the branch target address.

Branch Evaluation – Forwarding

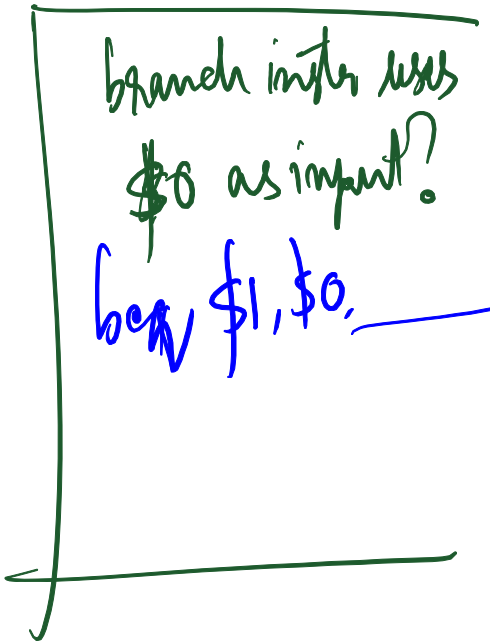
- New forwarding logic required
  - Source operands can come from ALU/MEM or MEM/WB



Branch Evaluation – Forwarding

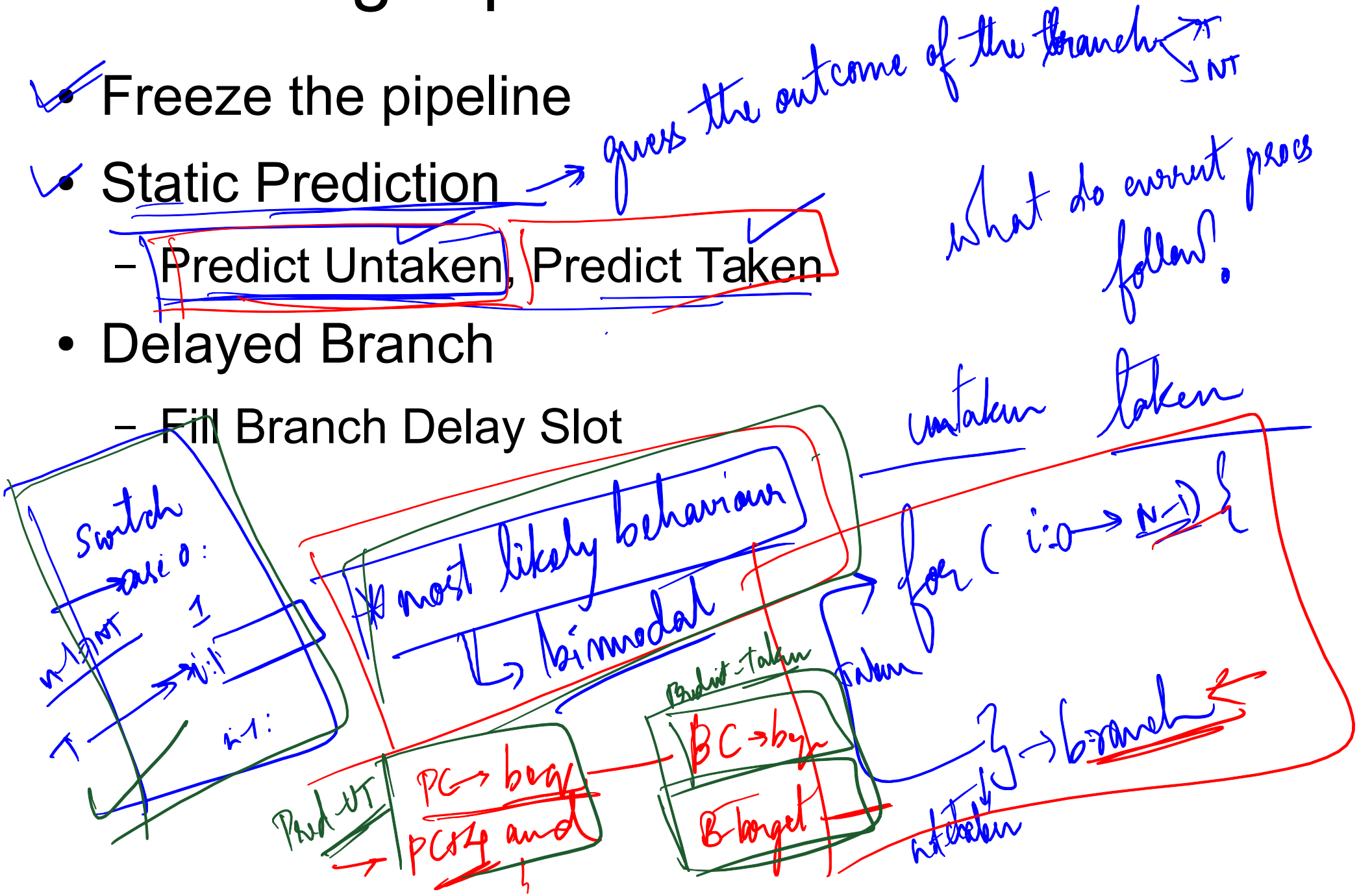
- New forwarding logic required
  - Source operands can come from ALU/MEM or MEM/WB
- If source operands are not ready, data hazard can occur and a stall will be needed.

```
...
add     $1, $2, $2
beq     $1, $3, 28
...
```

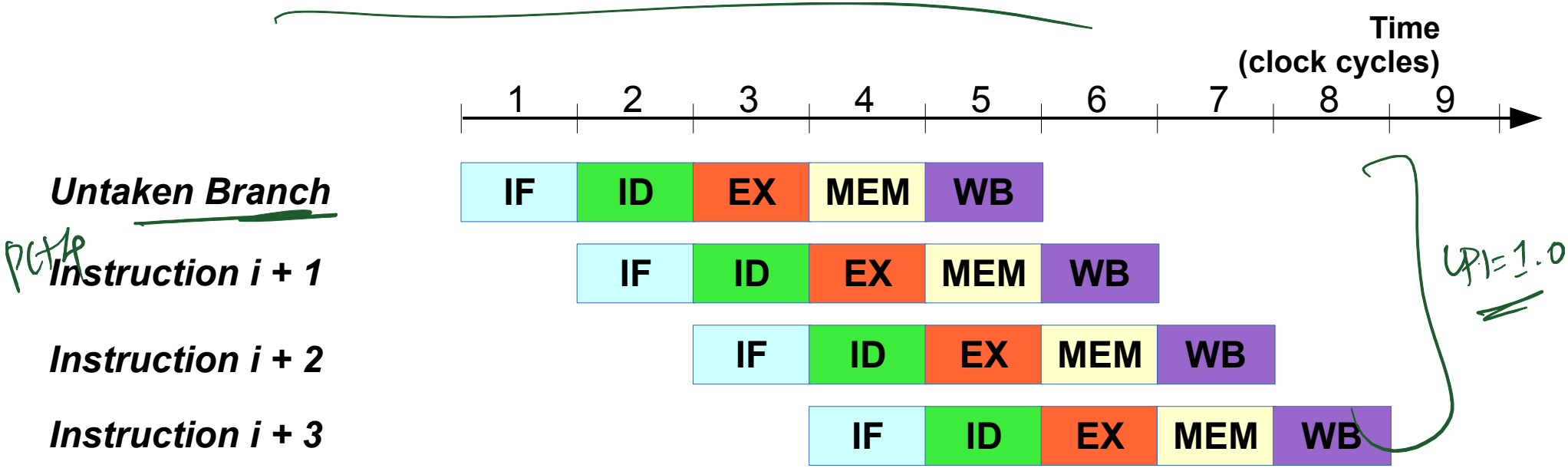


Reducing Pipeline Branch Penalties

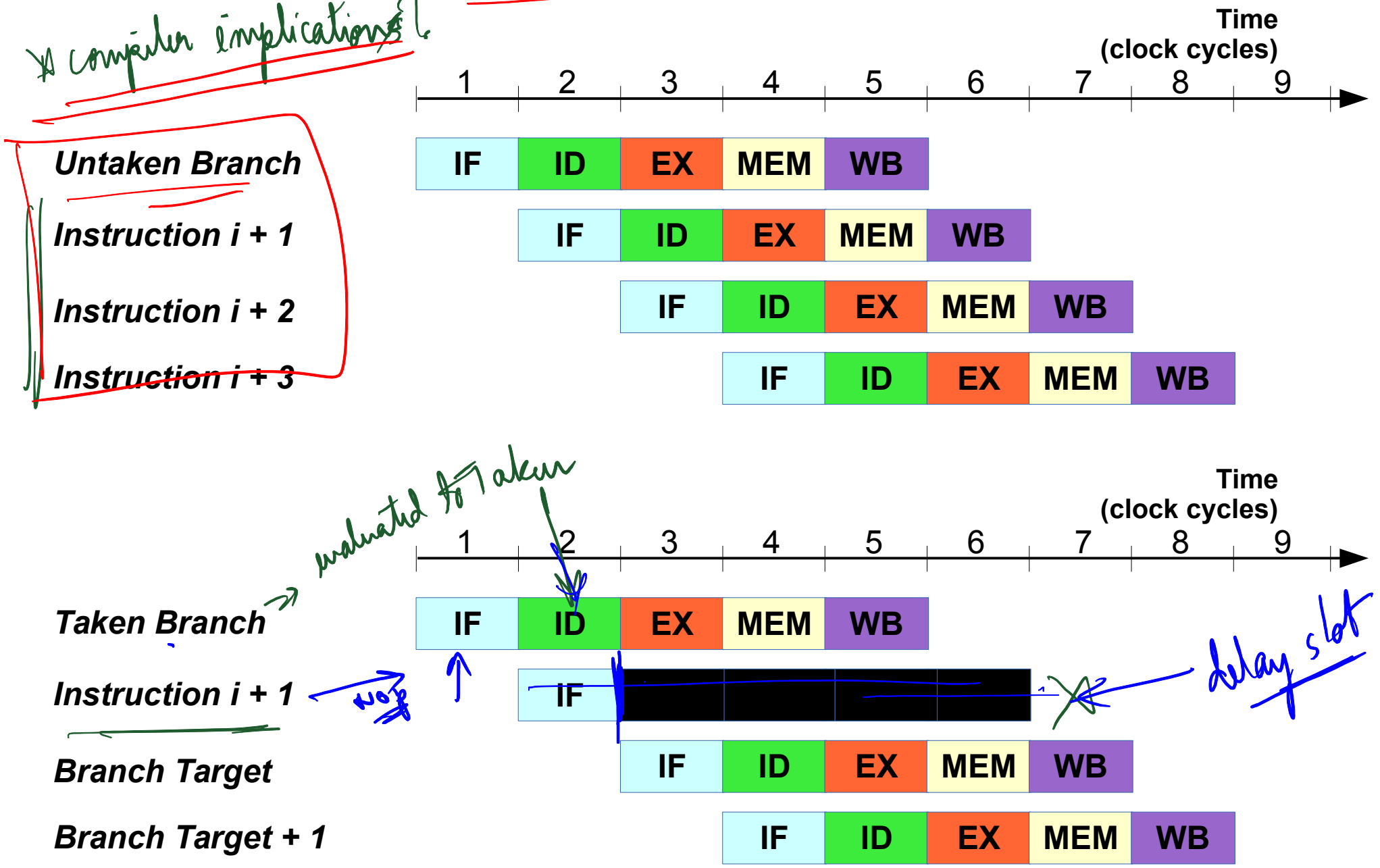
- Freeze the pipeline
- Static Prediction
  - Predict Untaken, Predict Taken
- Delayed Branch
  - Fill Branch Delay Slot



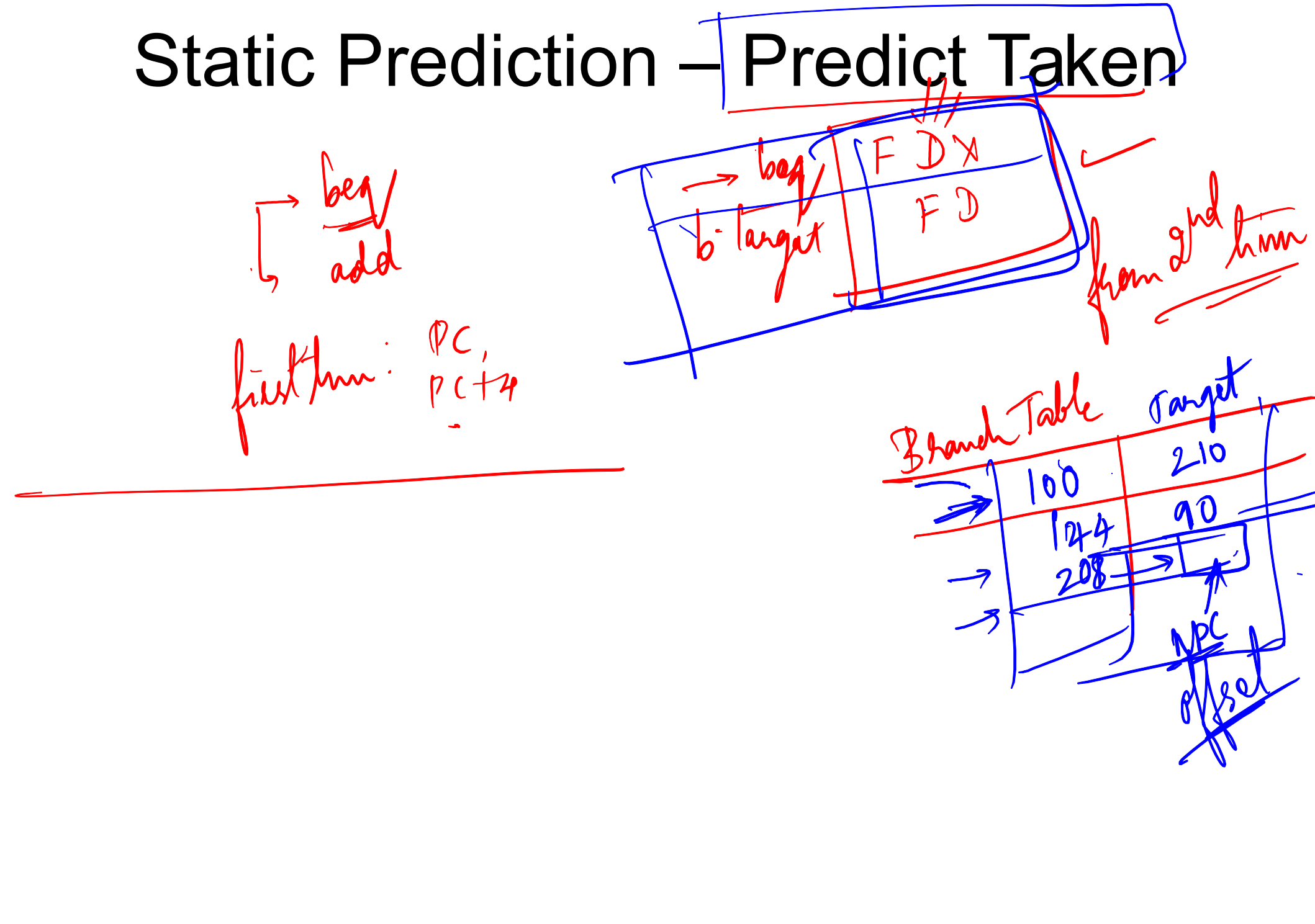
Predict Untaken Scheme



Predict Untaken Scheme



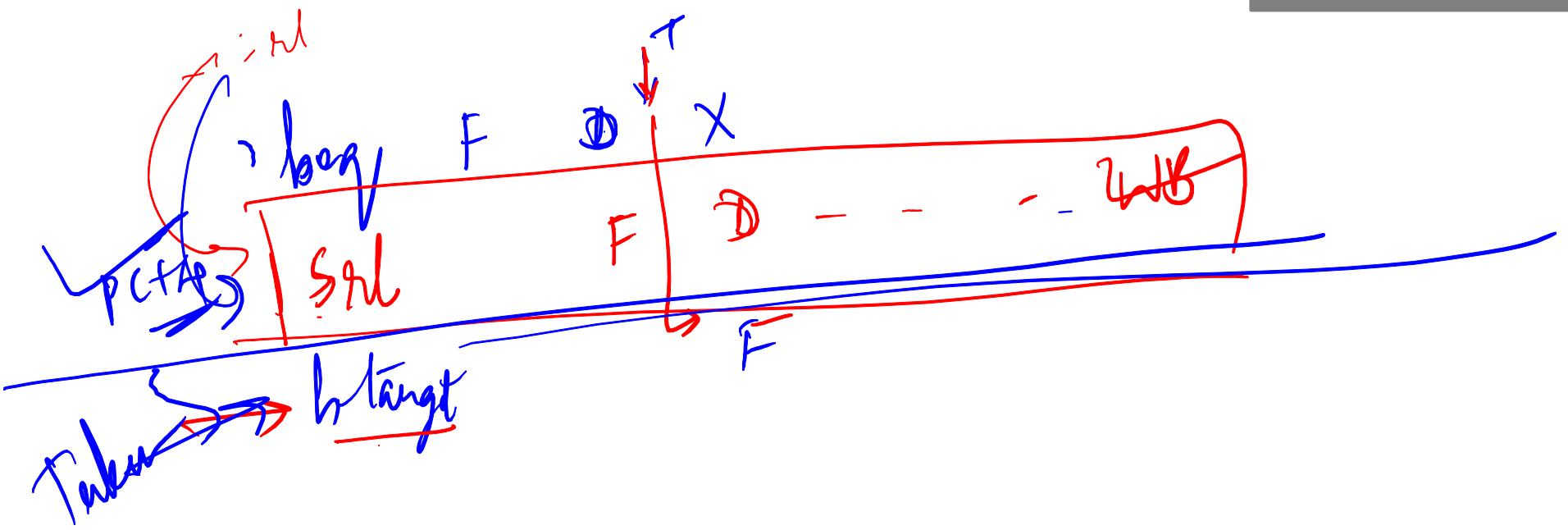
Static Prediction – Predict Taken



Reducing Pipeline Branch Penalties

- Fill the branch delay slot

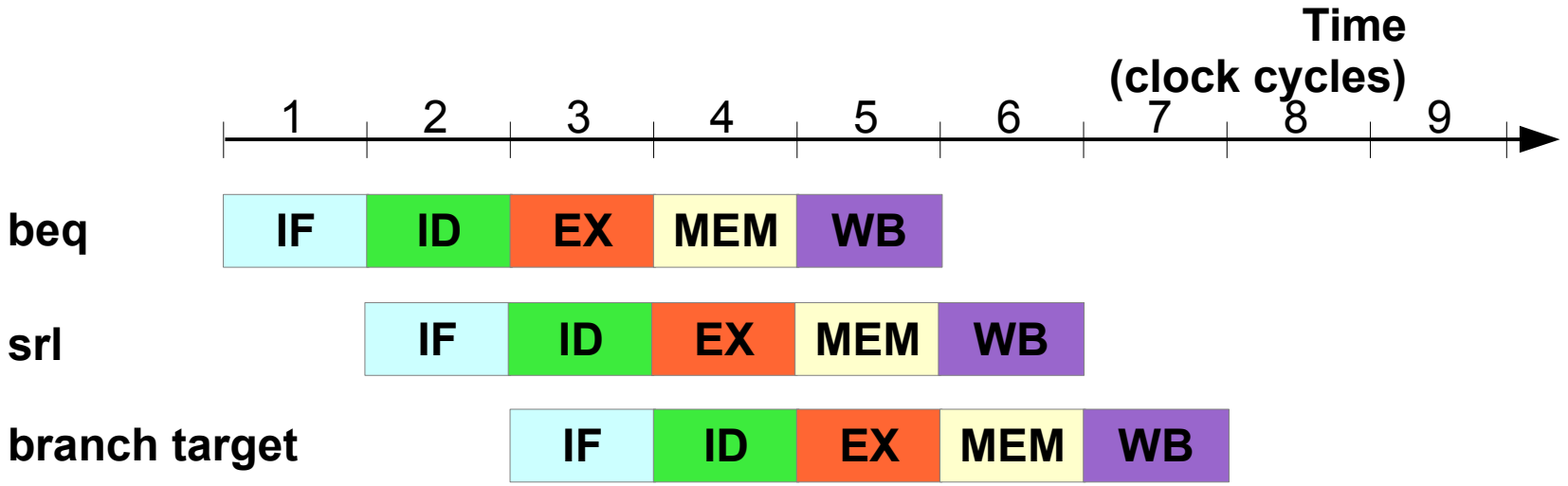
```
sll     ...
srl     $14, $2, $2
beq     $1, $3, 28
and     $12, $2, $5
...
```



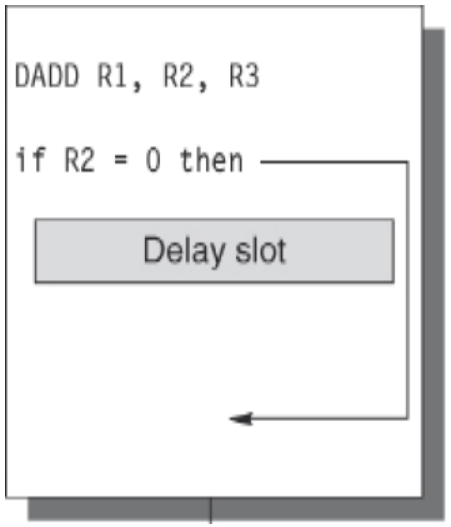
Reducing Pipeline Branch Penalties

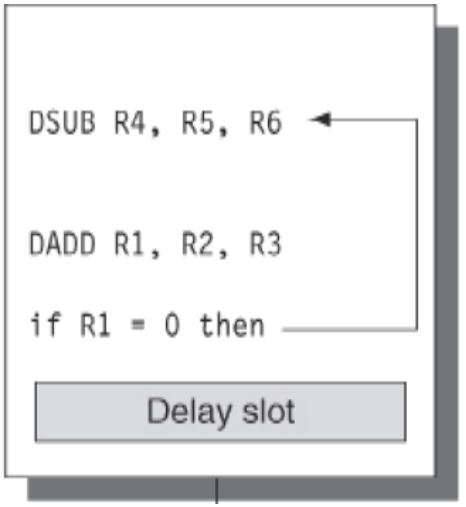
- Fill the branch delay slot

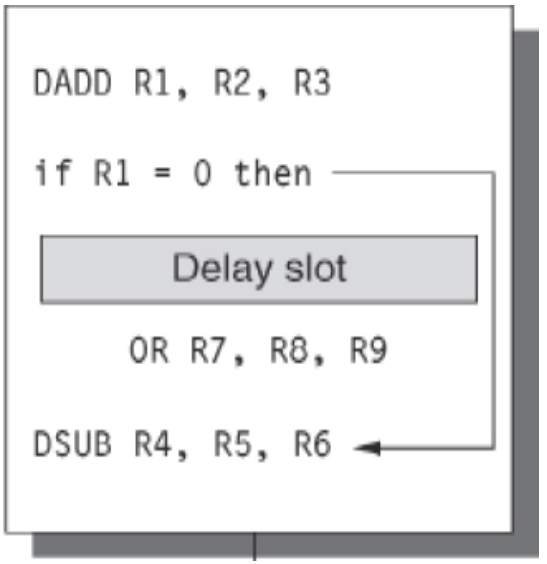
```
sll     ...
srl     $14, $2, $2
beq     $1, $3, 28
and     $12, $2, $5
...
```



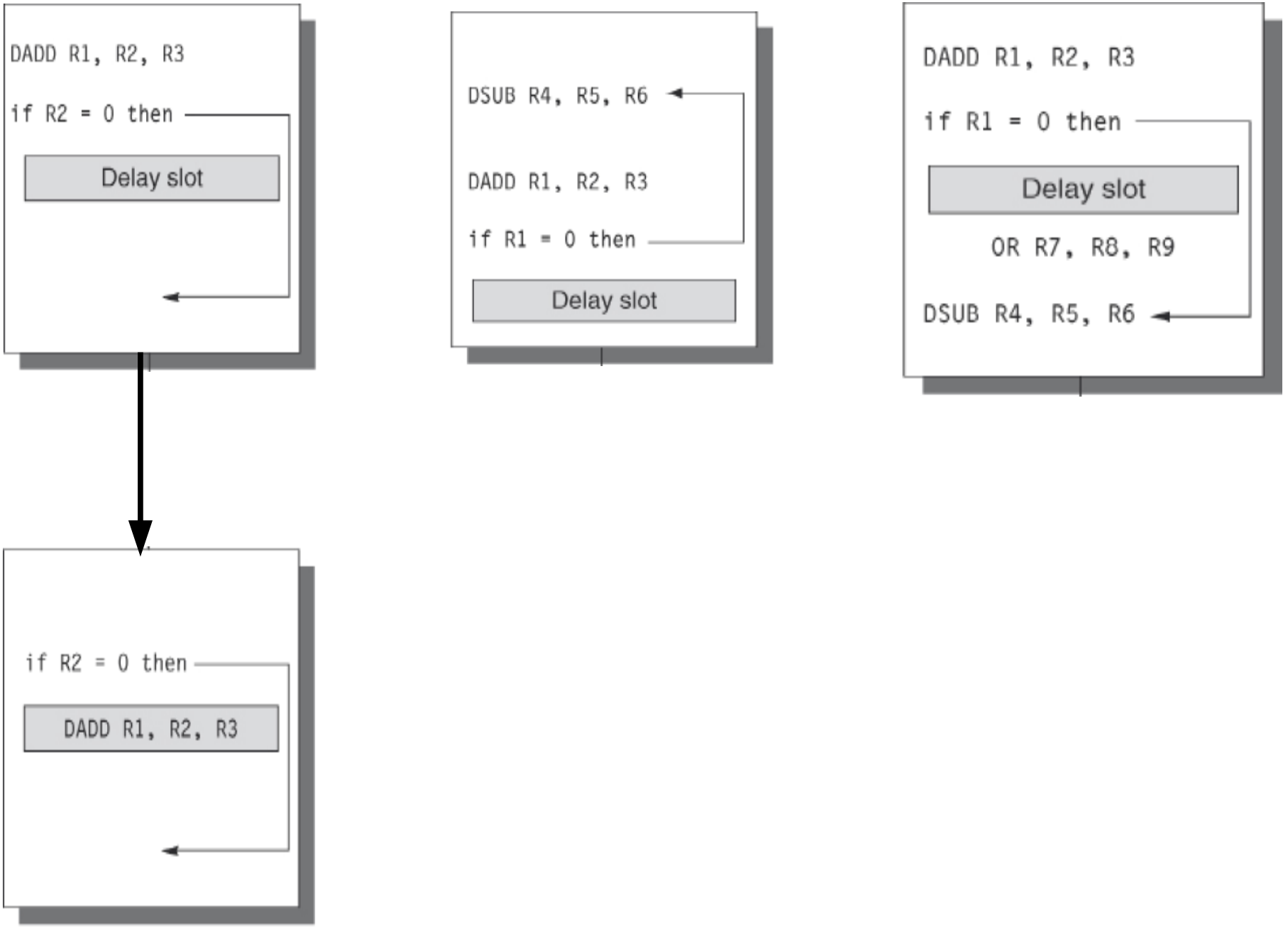
Branch Delay Slot



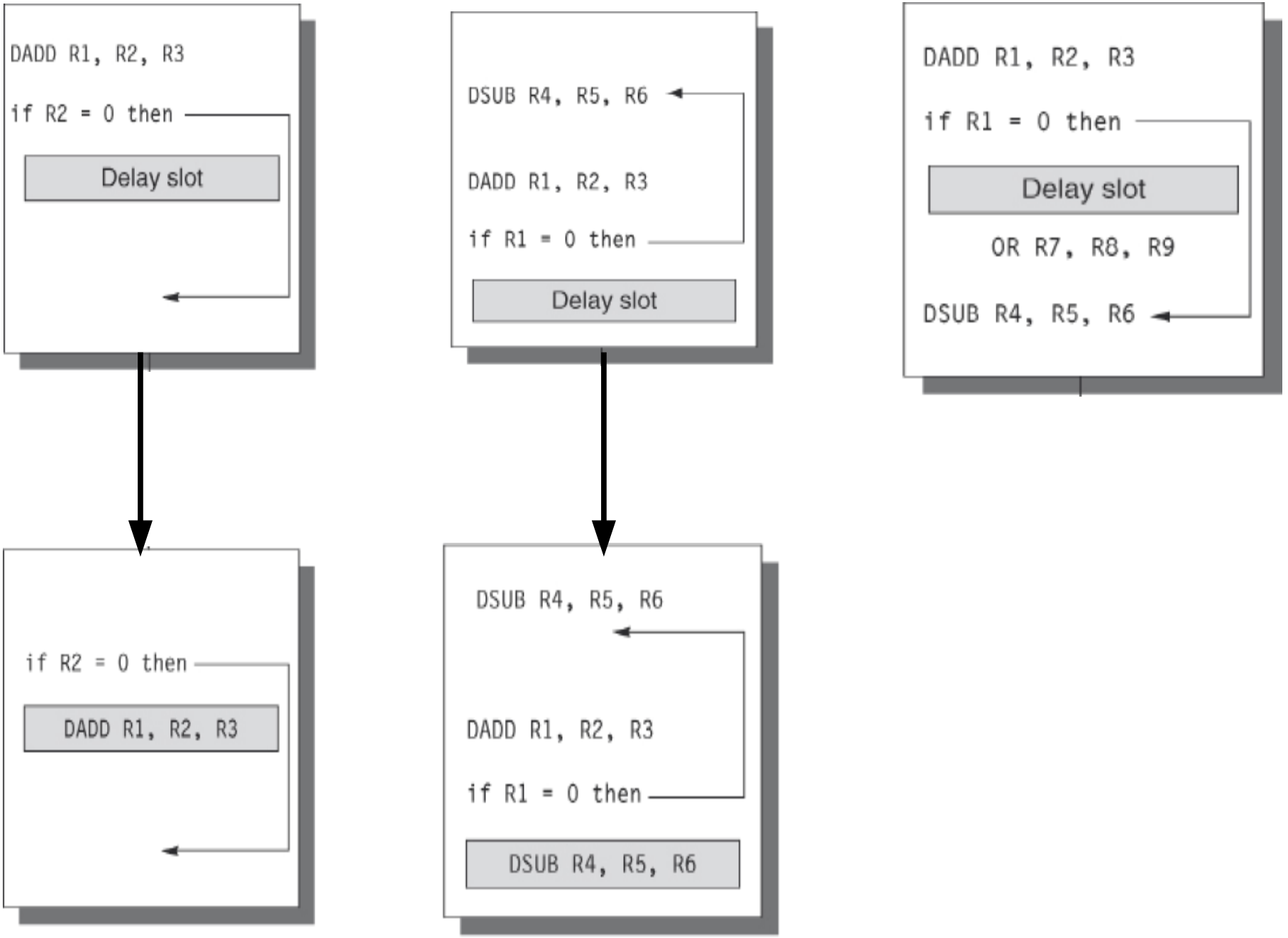




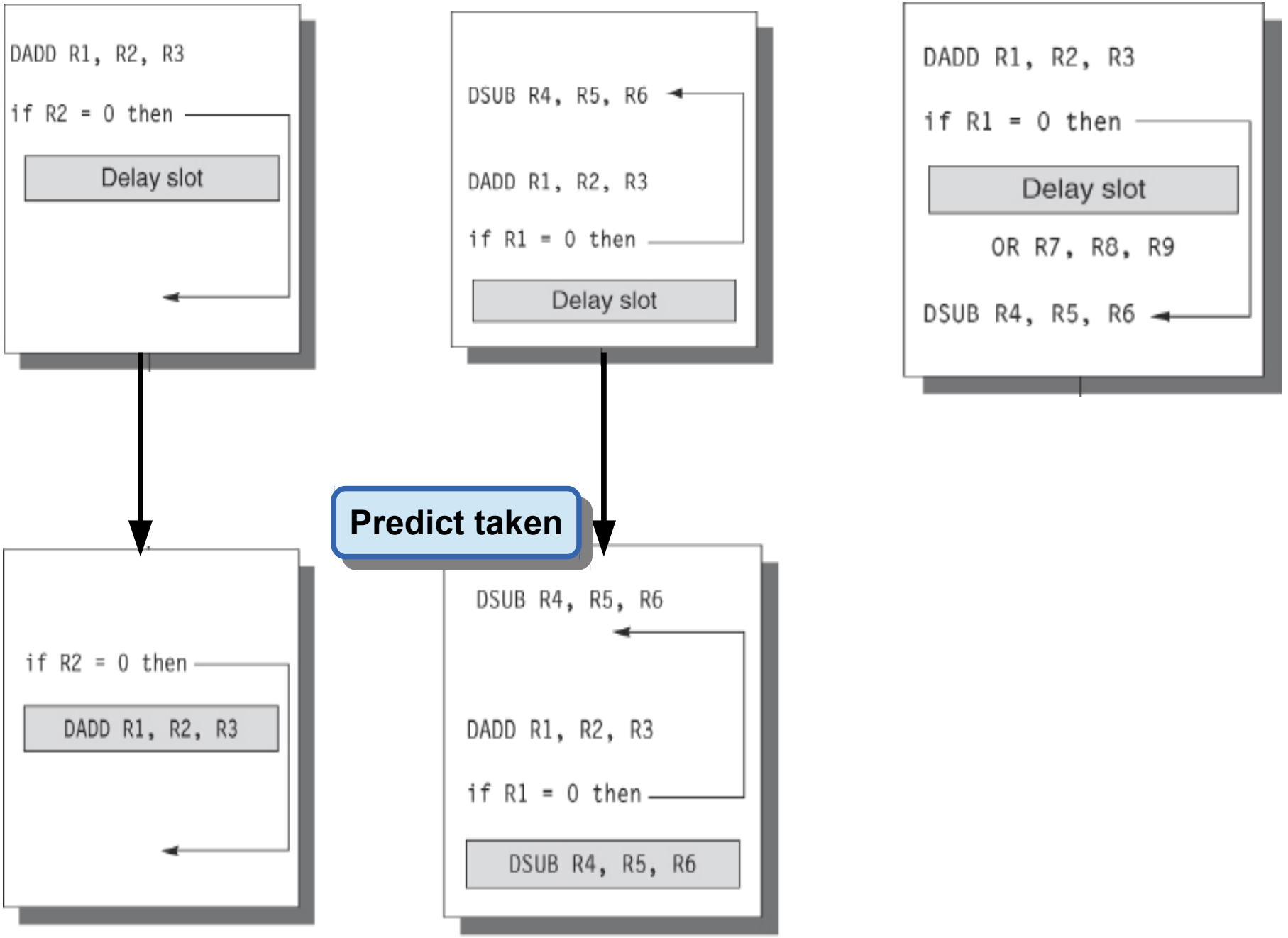
Branch Delay Slot



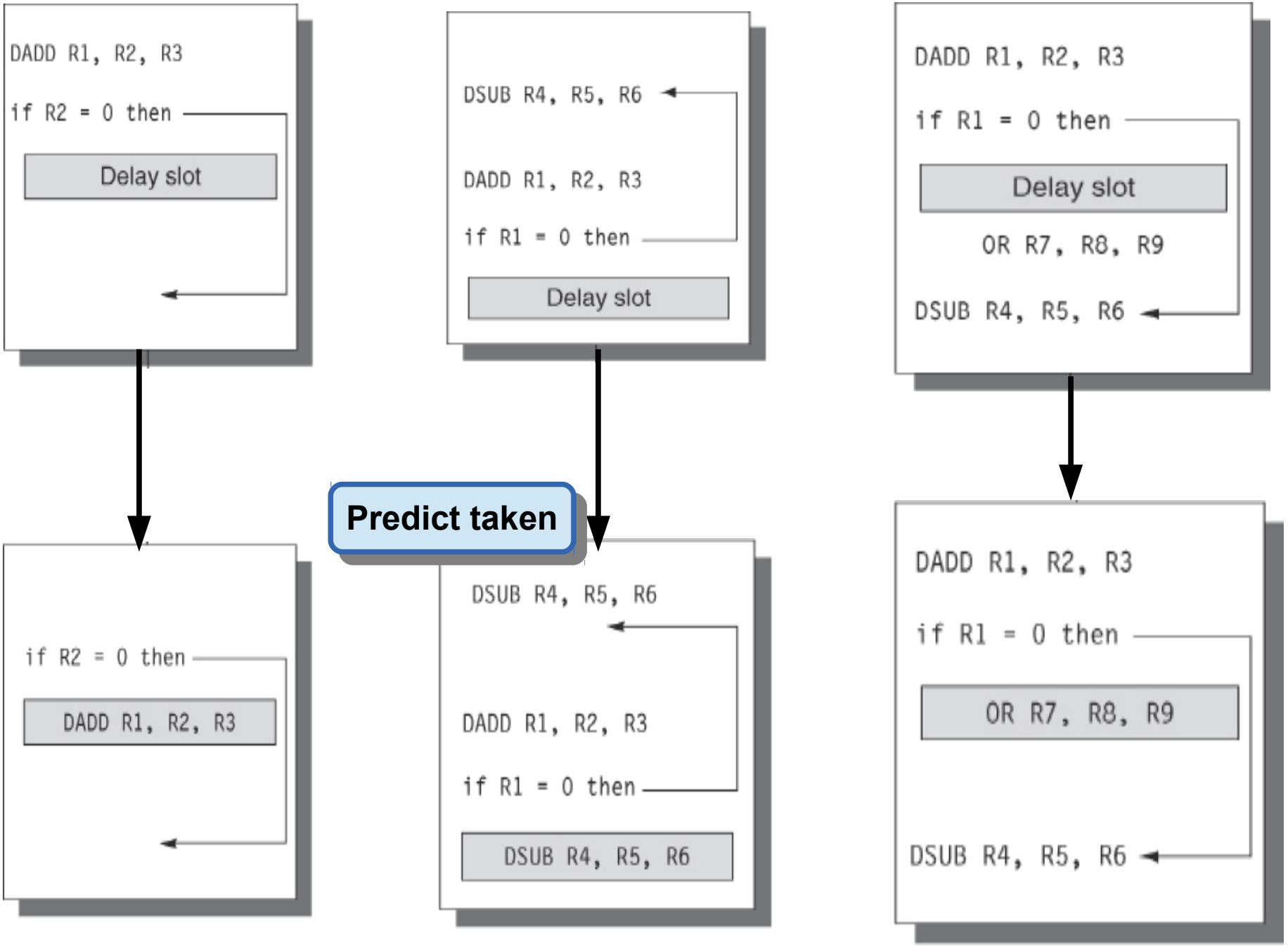
Branch Delay Slot



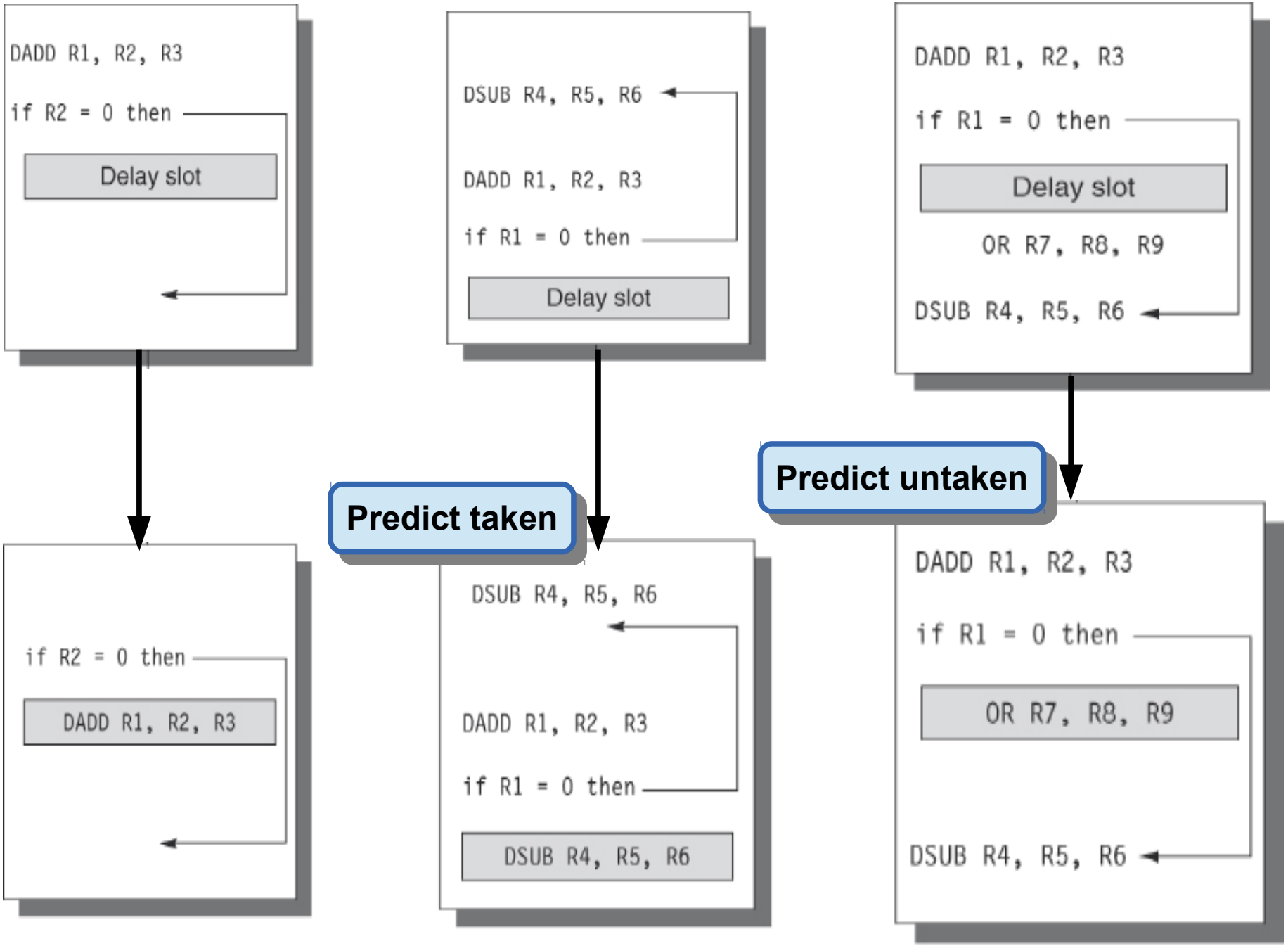
Branch Delay Slot



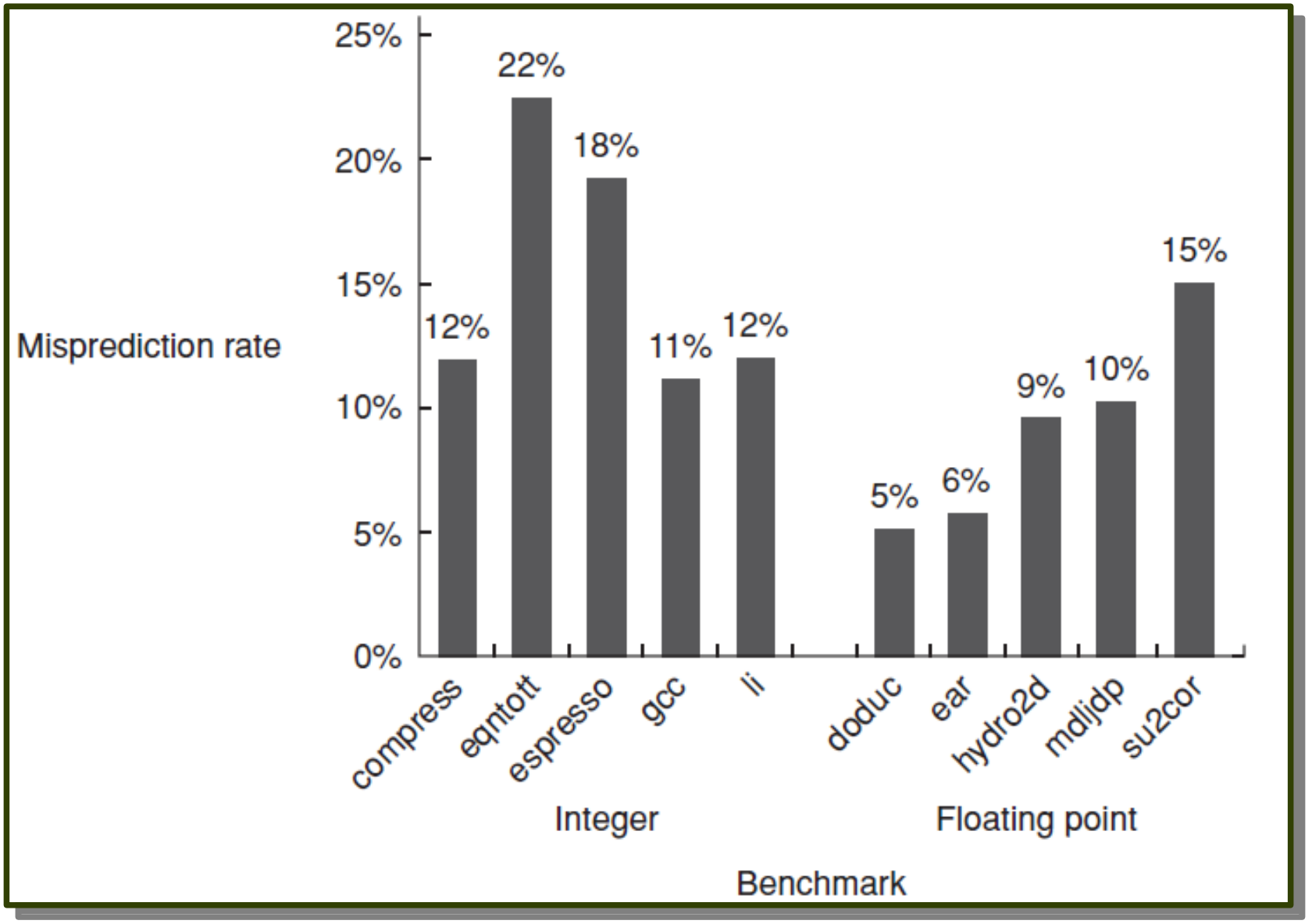
Branch Delay Slot



Branch Delay Slot



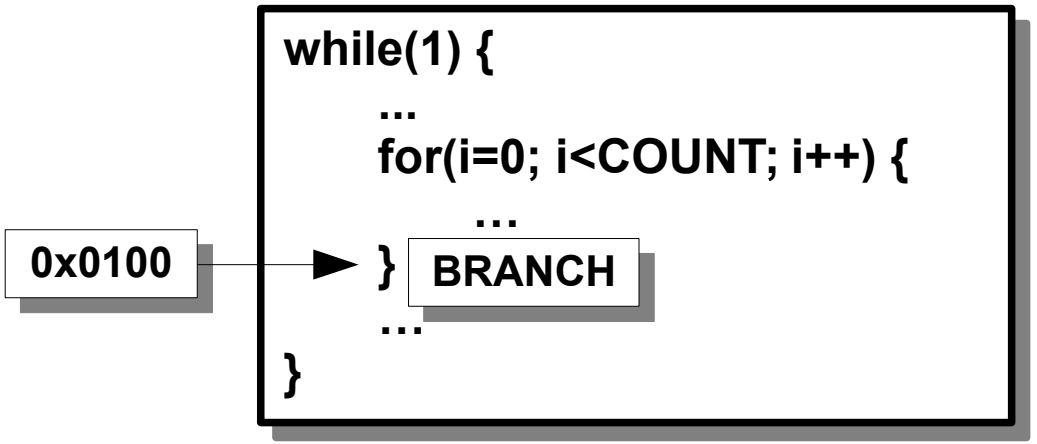
Static Branch Prediction



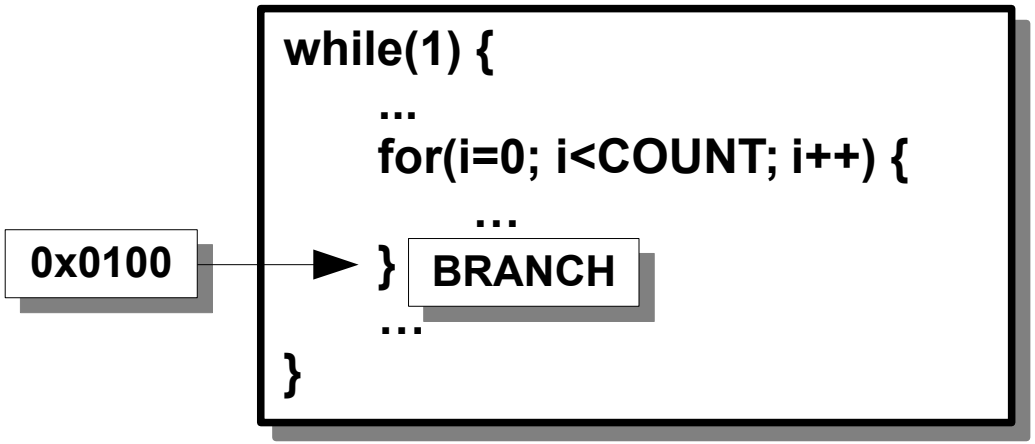
Static Branch Prediction

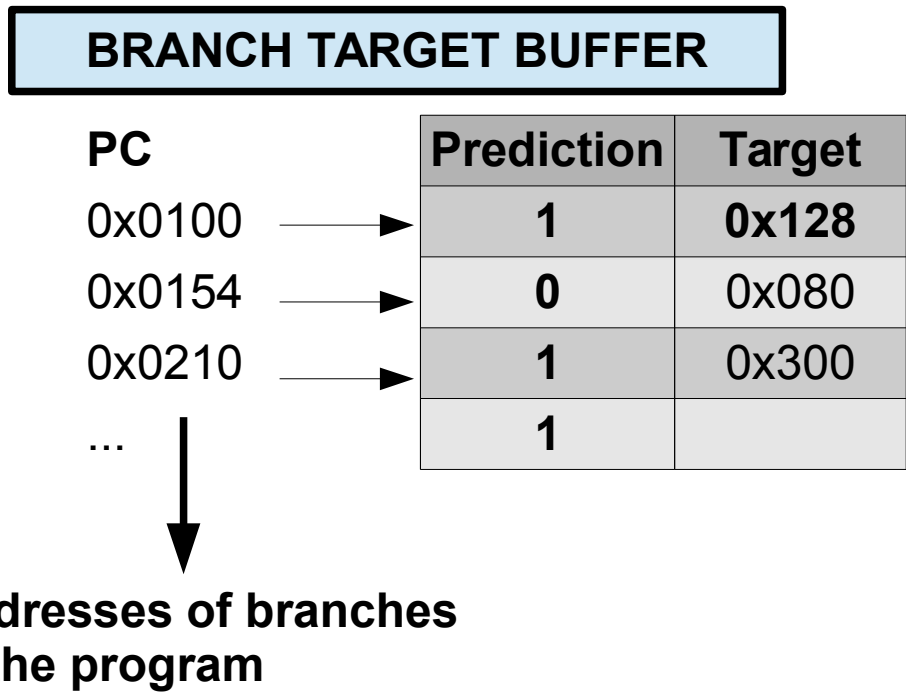
- Reduce mispredictions?

Dynamic Branch Prediction



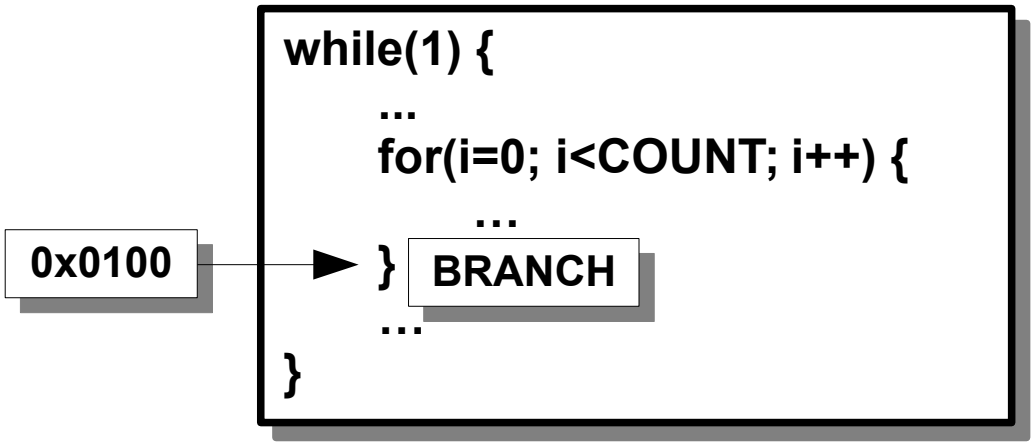
Dynamic Branch Prediction

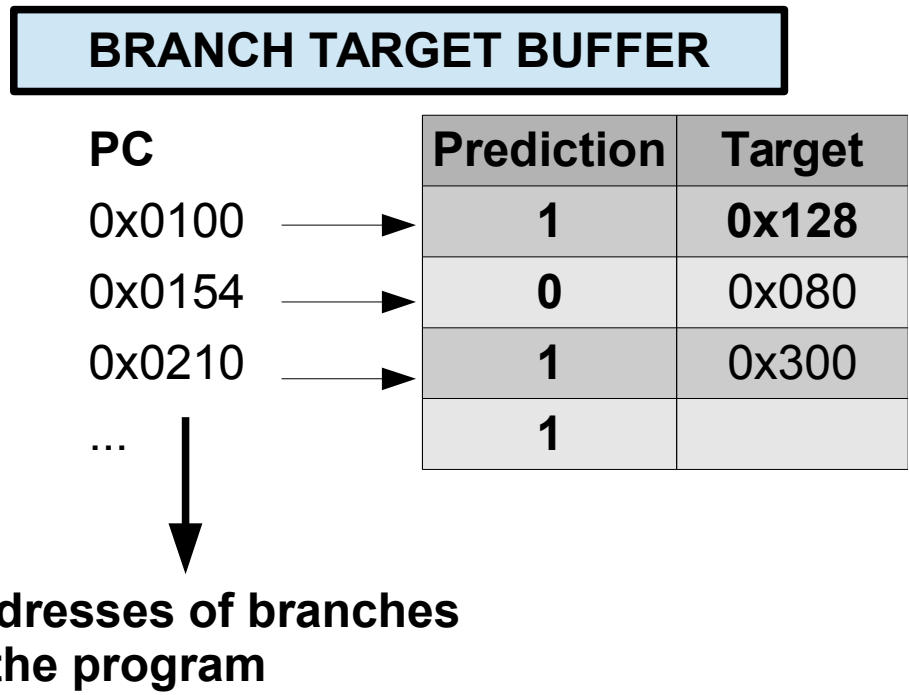




Dynamic Branch Prediction

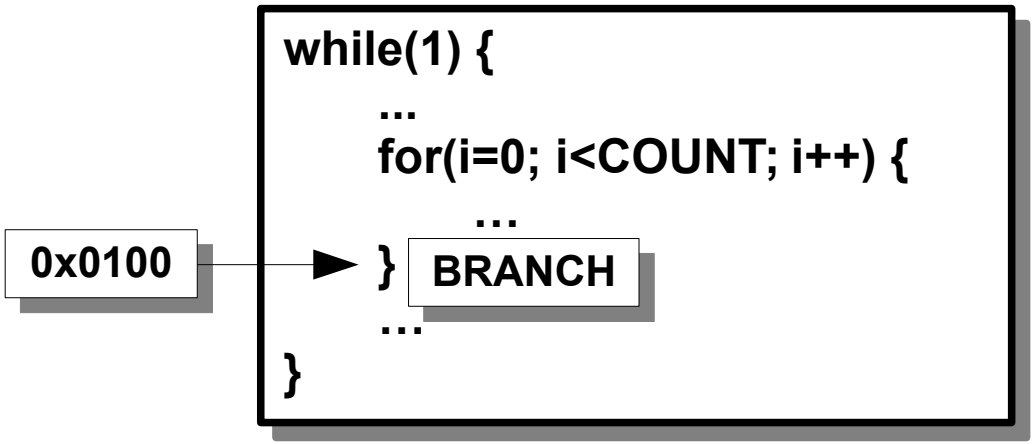
- Branch Target Buffer
  - Single bit predictors (1-bit bimodal predictor)
  - Change prediction with branch behaviour
  - No. of wrong predictions?





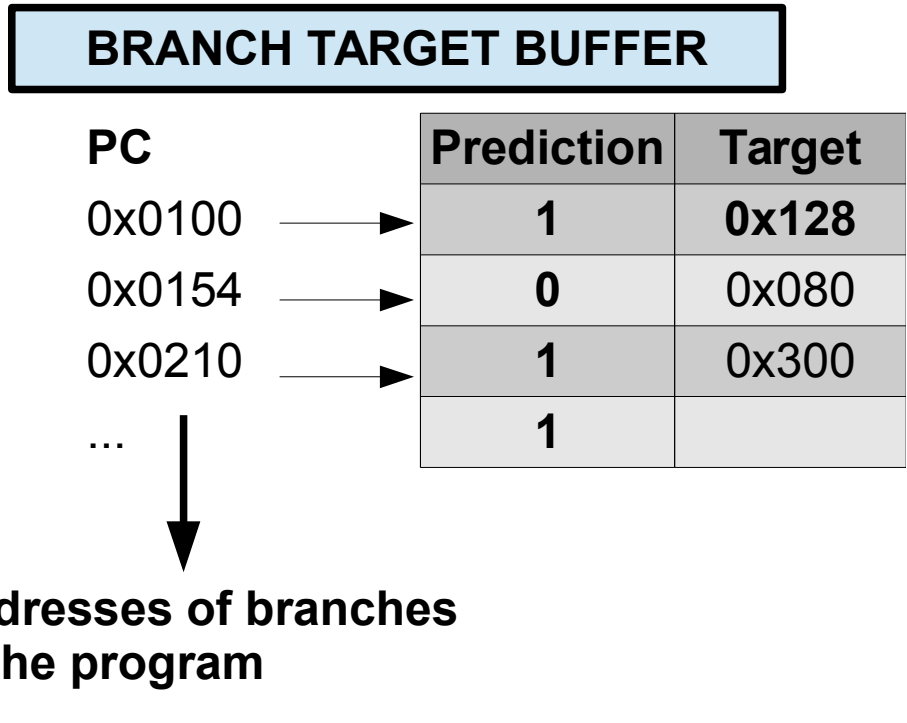
Dynamic Branch Prediction

- Branch Target Buffer
  - Single bit predictors (1-bit bimodal predictor)
  - Change prediction with branch behaviour
  - No. of wrong predictions?



Branch instruction behaviour

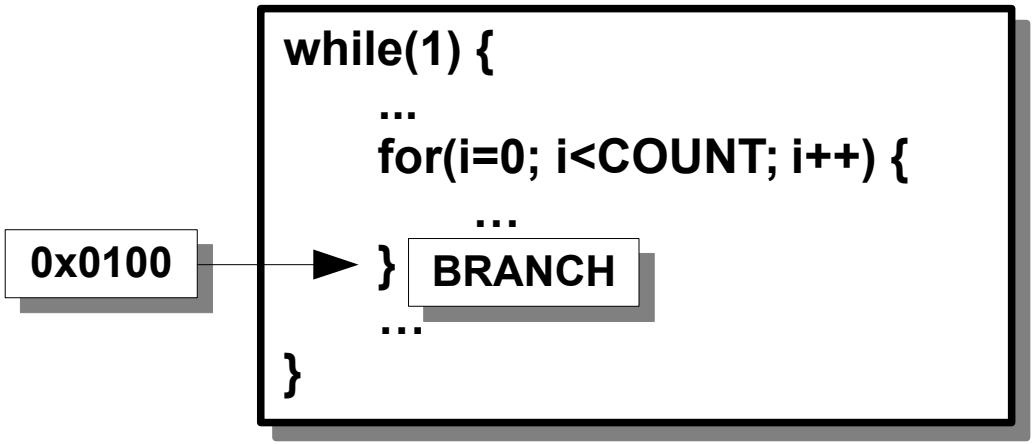
Wrong Predictions



Addresses of branches in the program

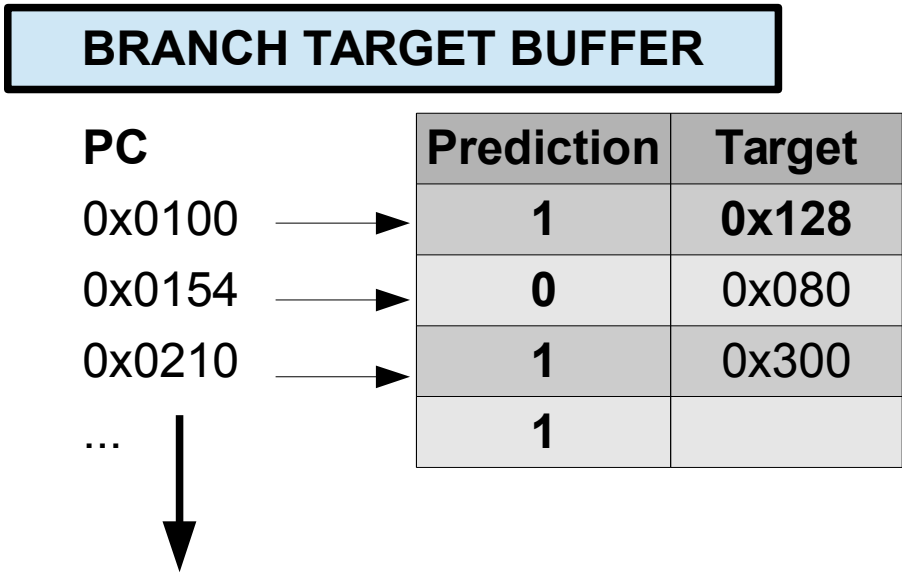
Dynamic Branch Prediction

- Branch Target Buffer
  - Single bit predictors (1-bit bimodal predictor)
  - Change prediction with branch behaviour
  - No. of wrong predictions?



Branch instruction behaviour

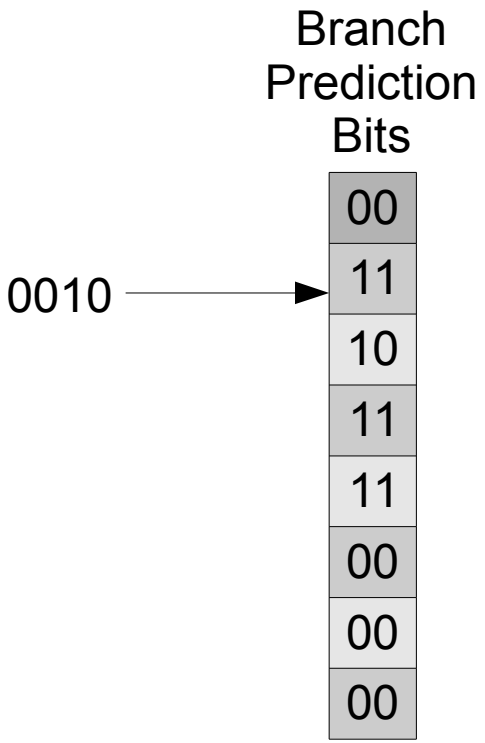
Wrong Predictions



Addresses of branches in the program

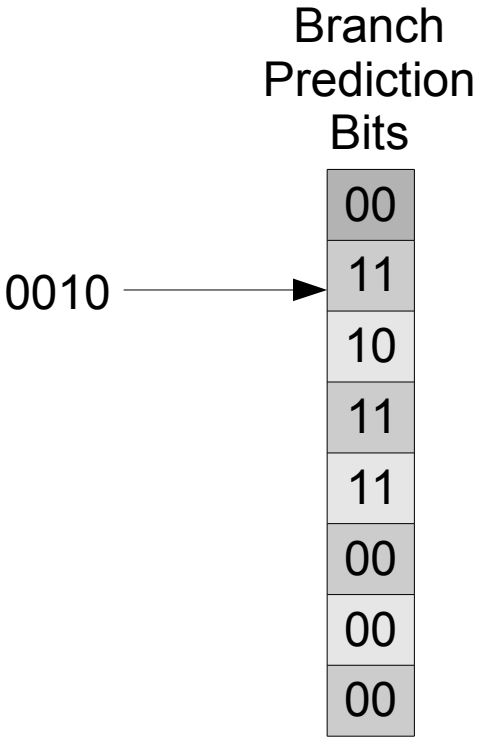
Can we do better?

Dynamic Branch Prediction



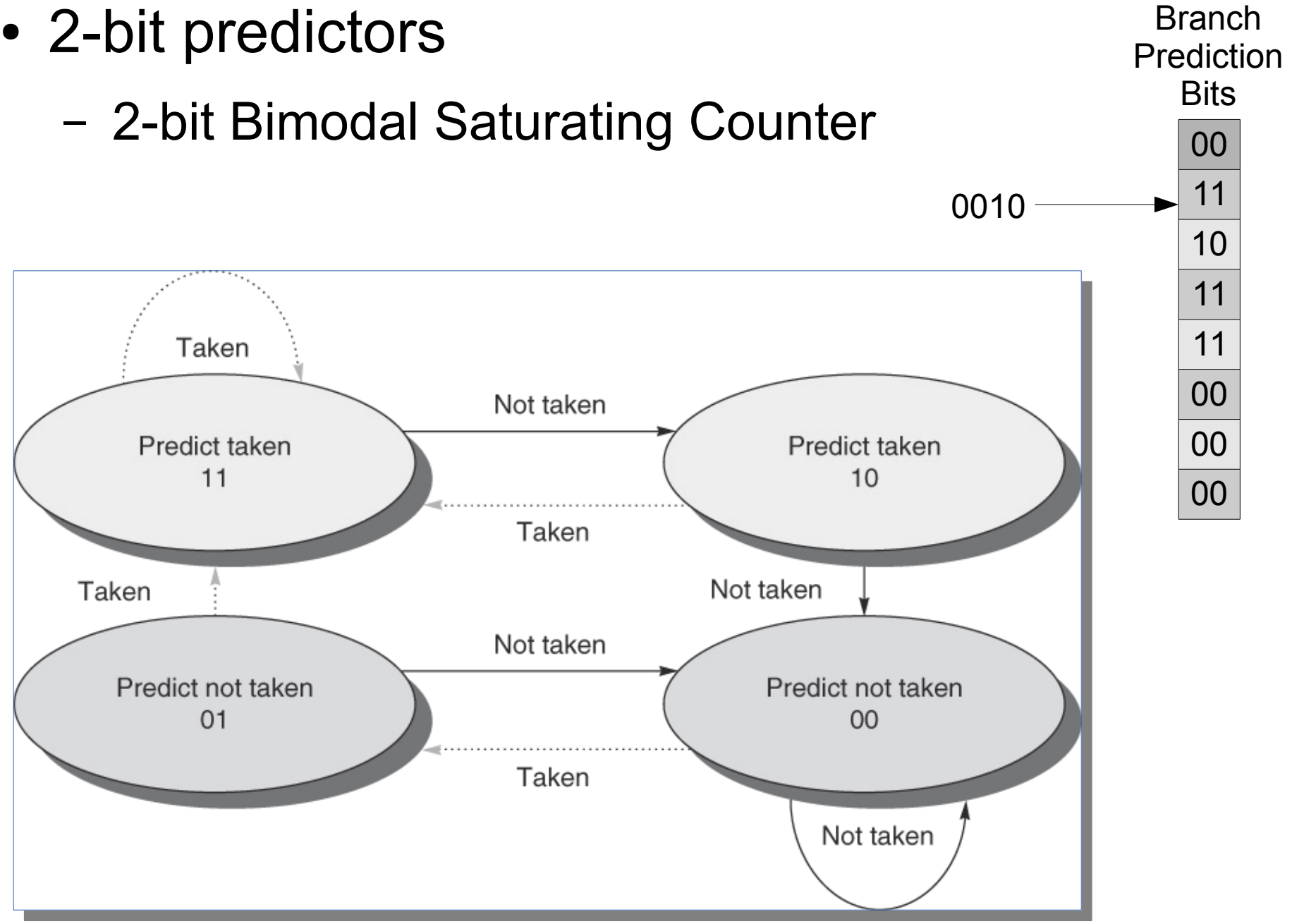
Dynamic Branch Prediction

- 2-bit predictors
  - 2-bit Bimodal Saturating Counter



Dynamic Branch Prediction

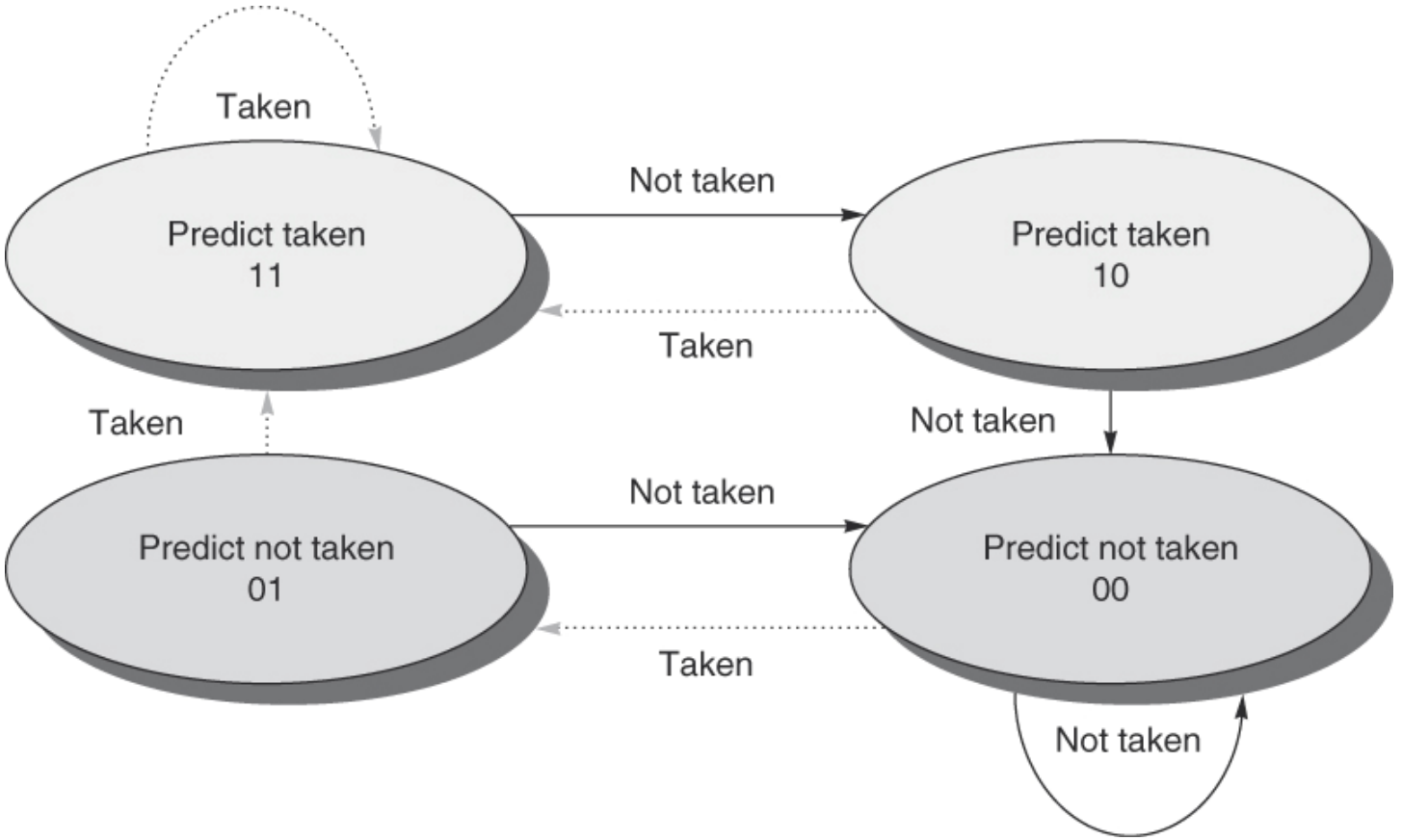
- 2-bit predictors
  - 2-bit Bimodal Saturating Counter



Dynamic Branch Prediction

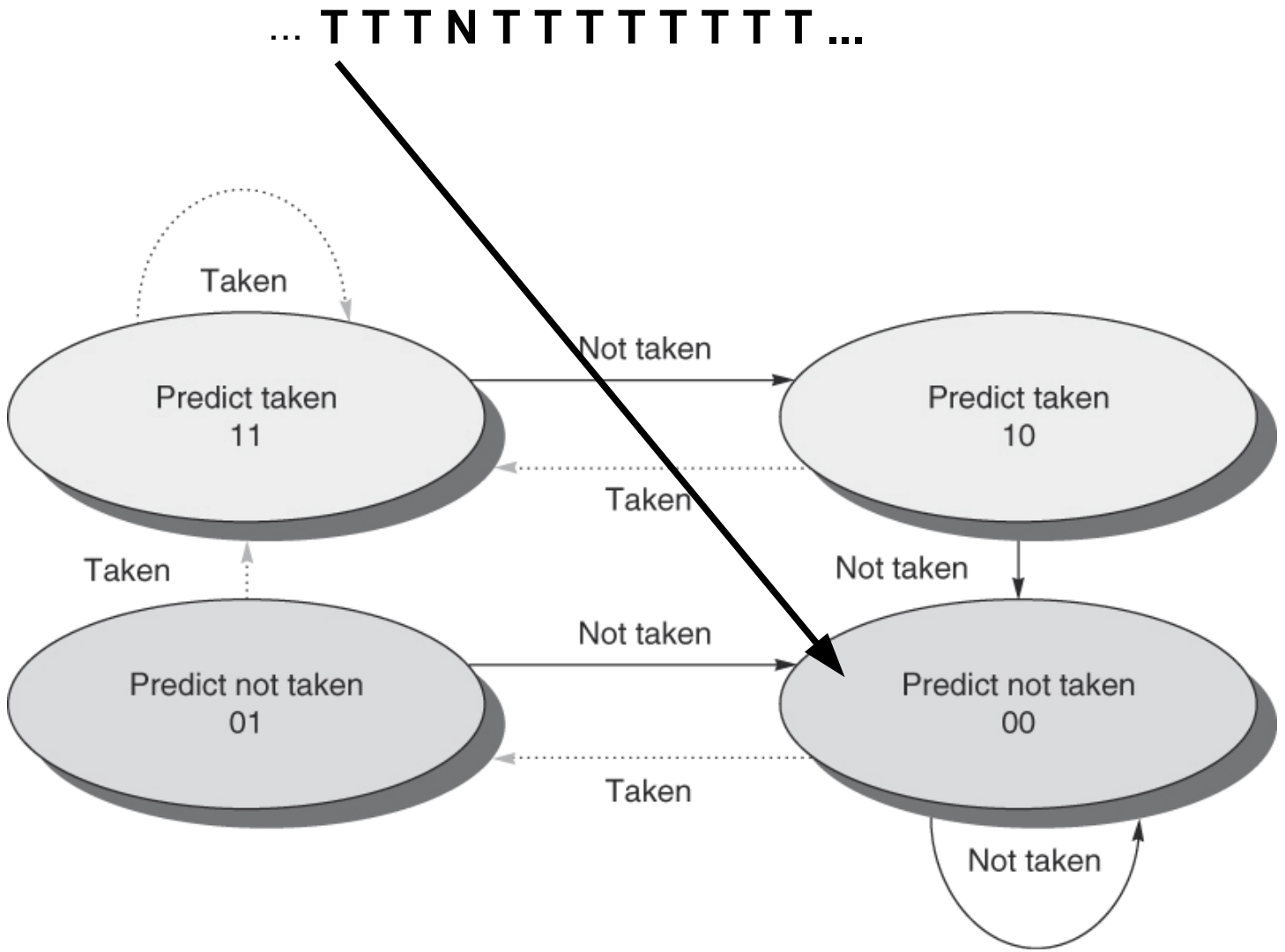
- 2-bit predictors

... **T T T N T T T T T T T T** ...



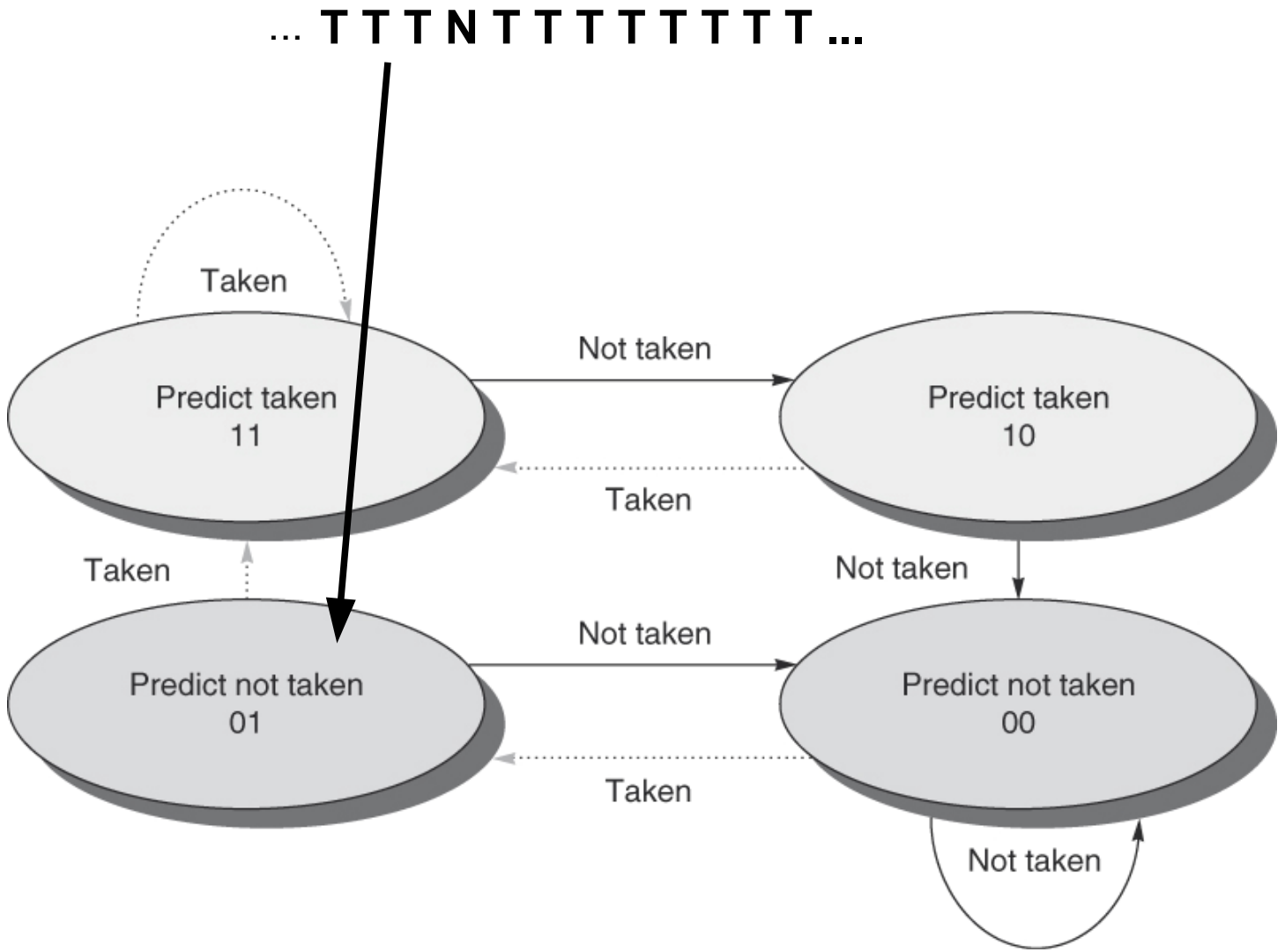
Dynamic Branch Prediction

- 2-bit predictors



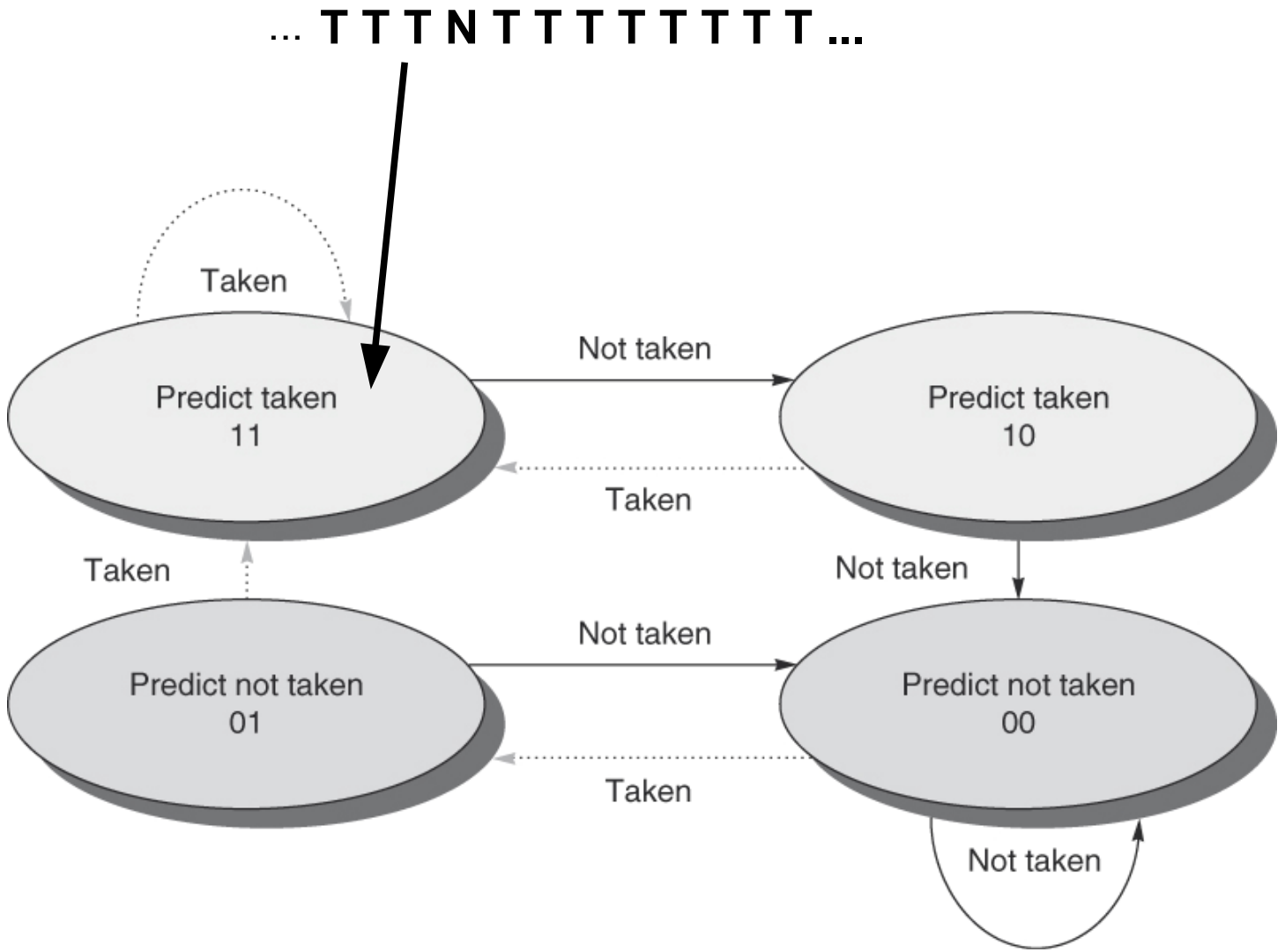
Dynamic Branch Prediction

- 2-bit predictors



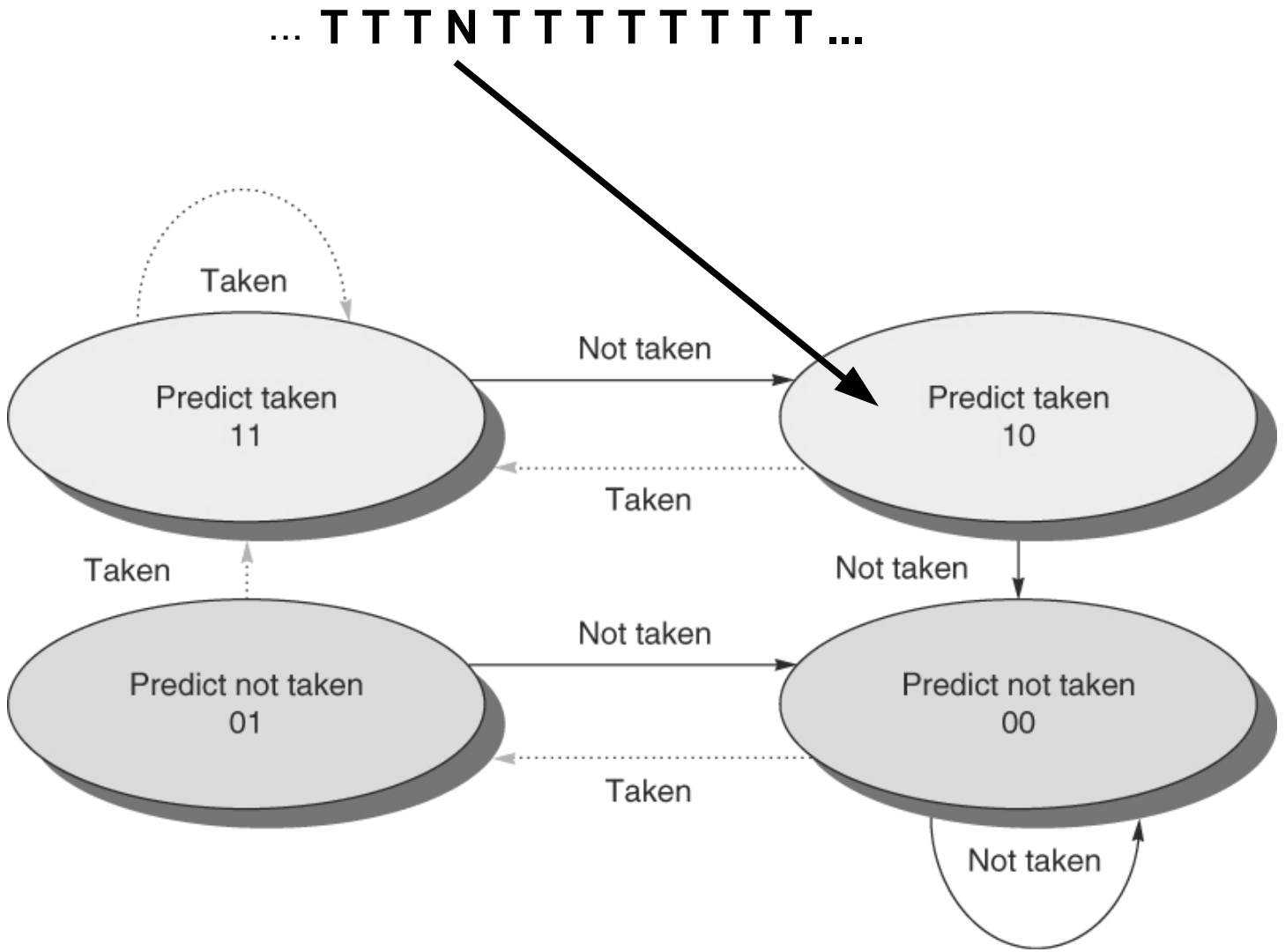
Dynamic Branch Prediction

- 2-bit predictors



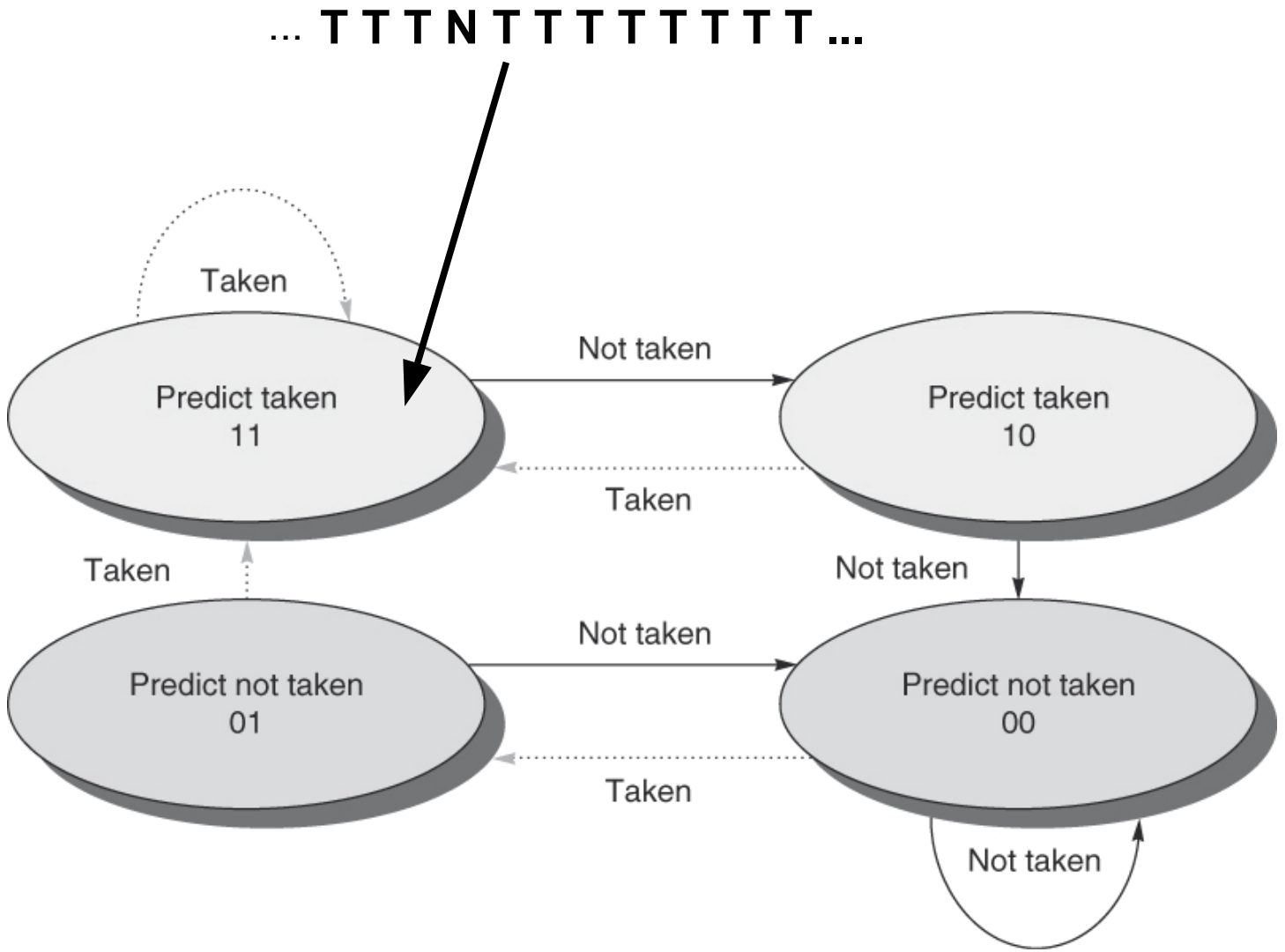
Dynamic Branch Prediction

- 2-bit predictors



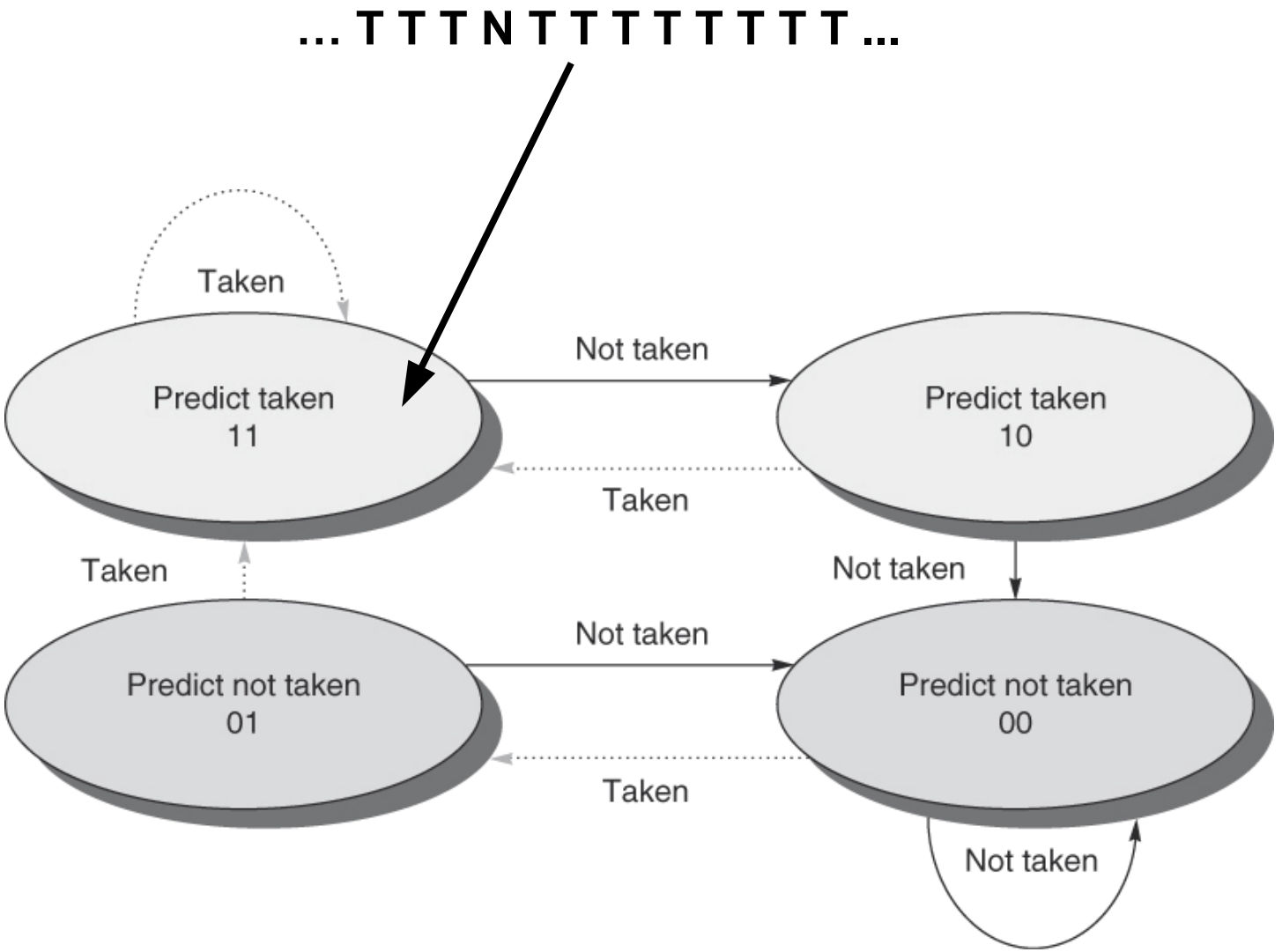
Dynamic Branch Prediction

- 2-bit predictors



Dynamic Branch Prediction

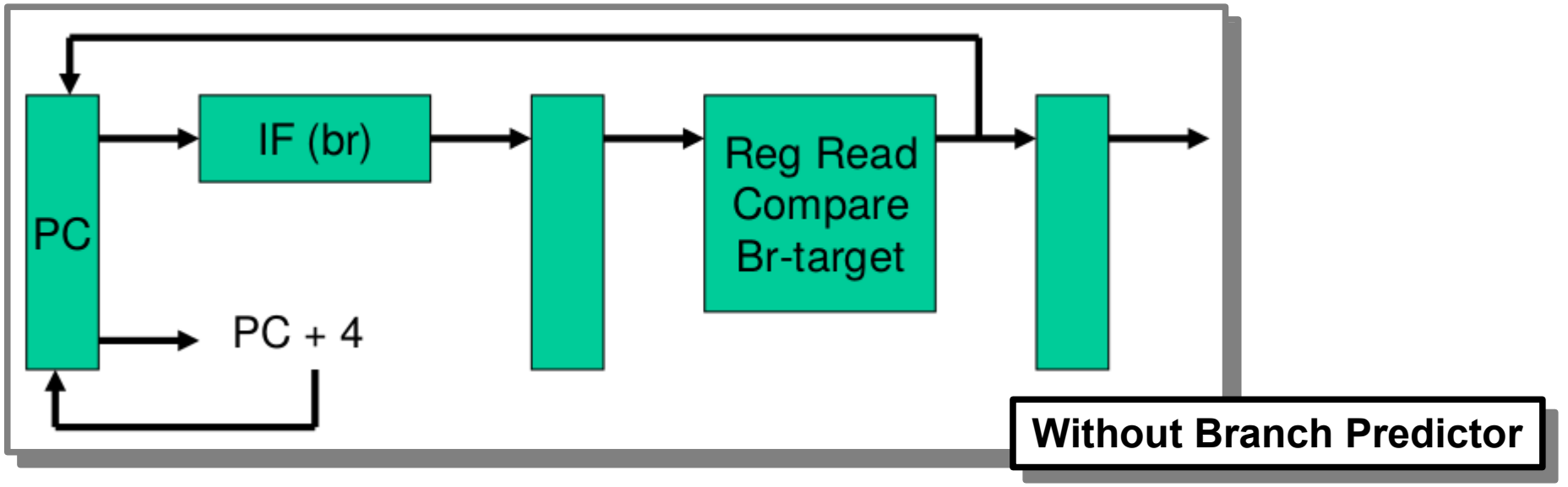
- 2-bit predictors



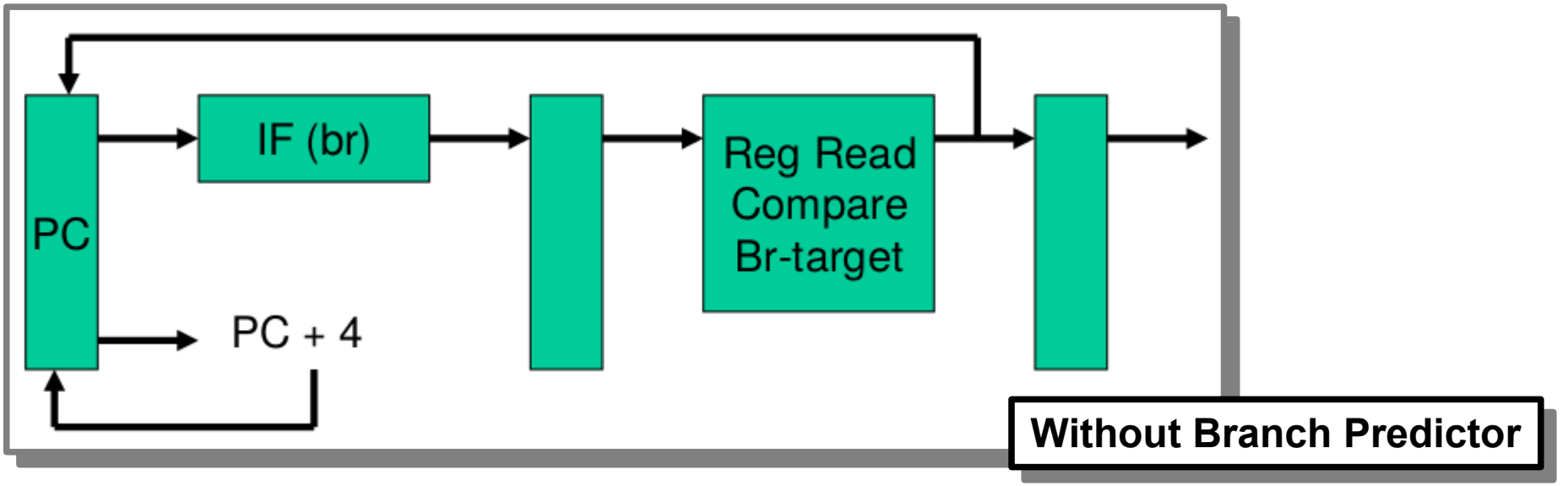
Dynamic Branch Prediction

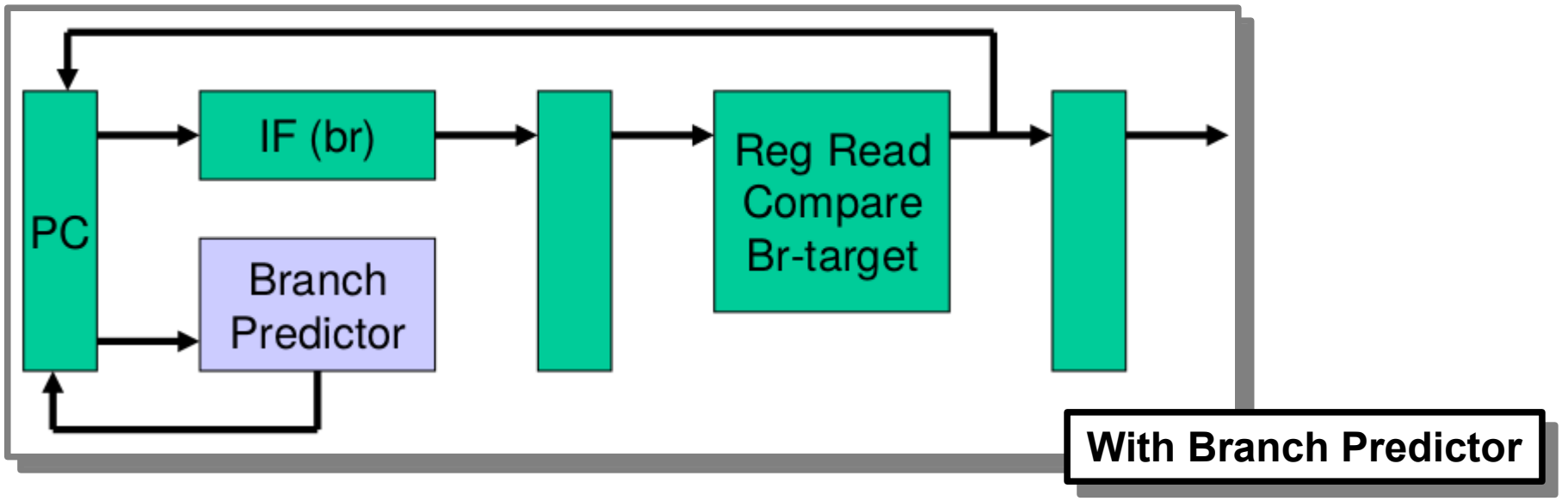
- n-bit saturating counters

Branch Predictors

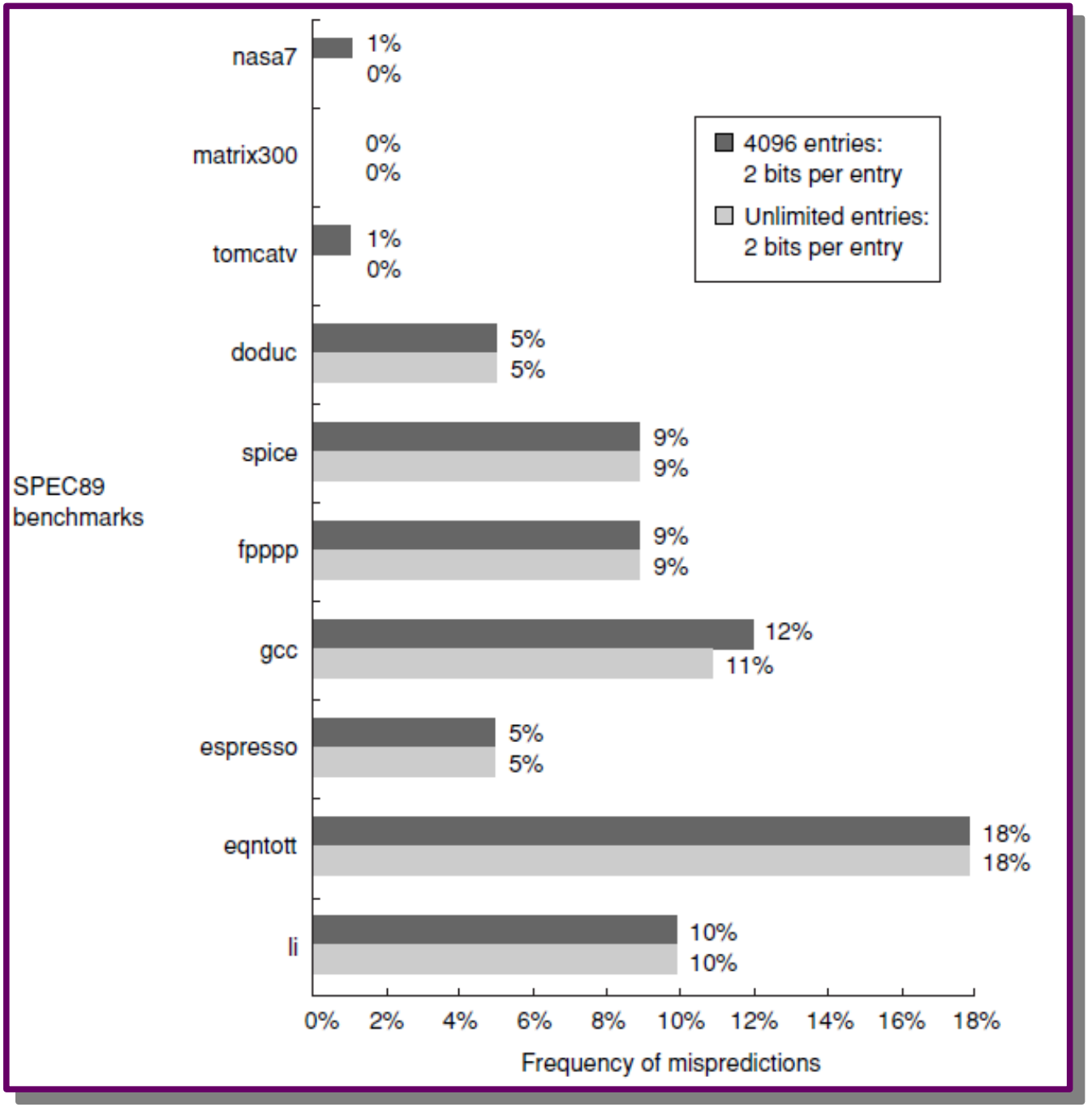


Branch Predictors

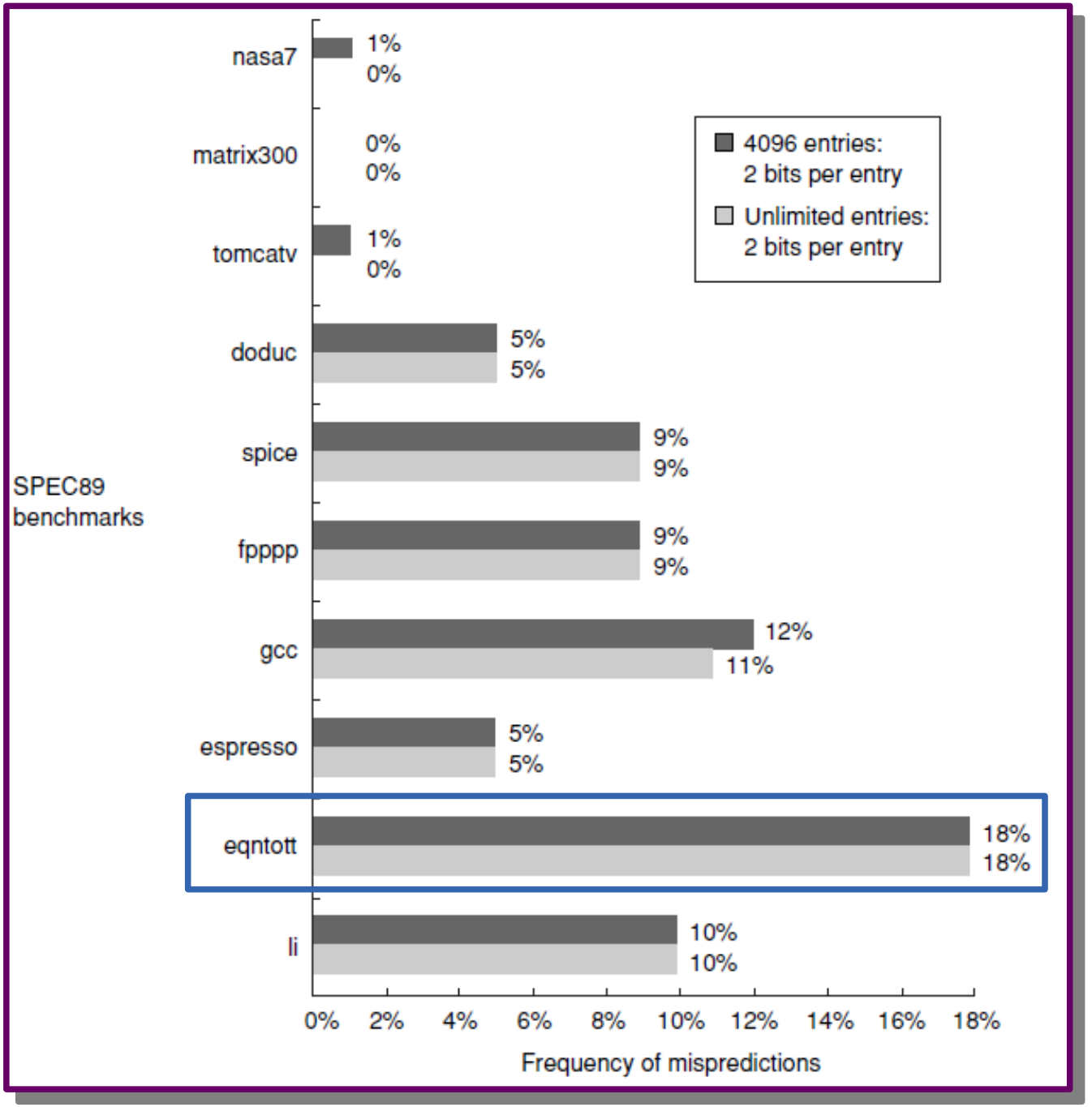




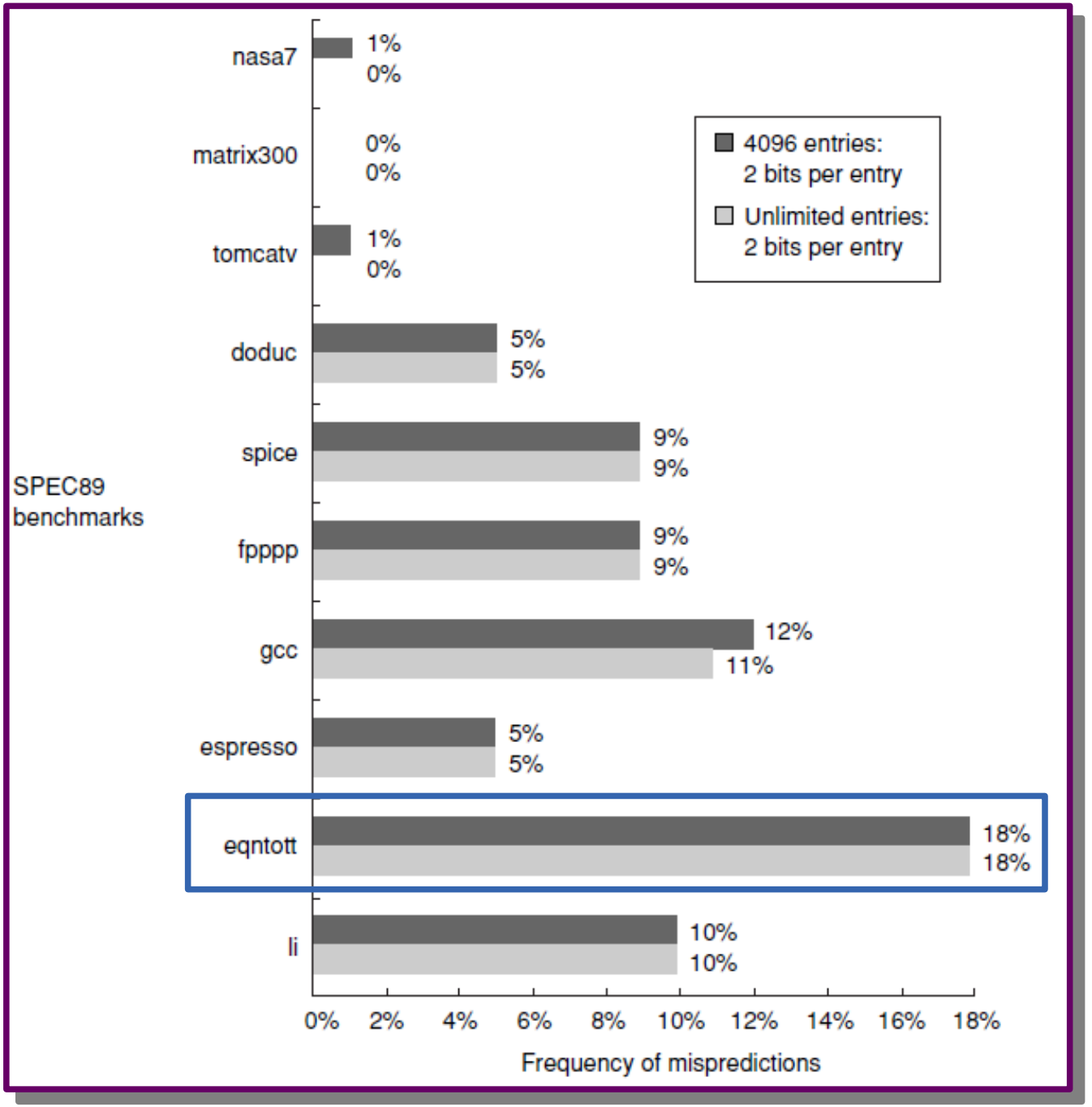
Dynamic Branch Prediction



Dynamic Branch Prediction



Dynamic Branch Prediction



Observations?

Correlating Branch Predictors

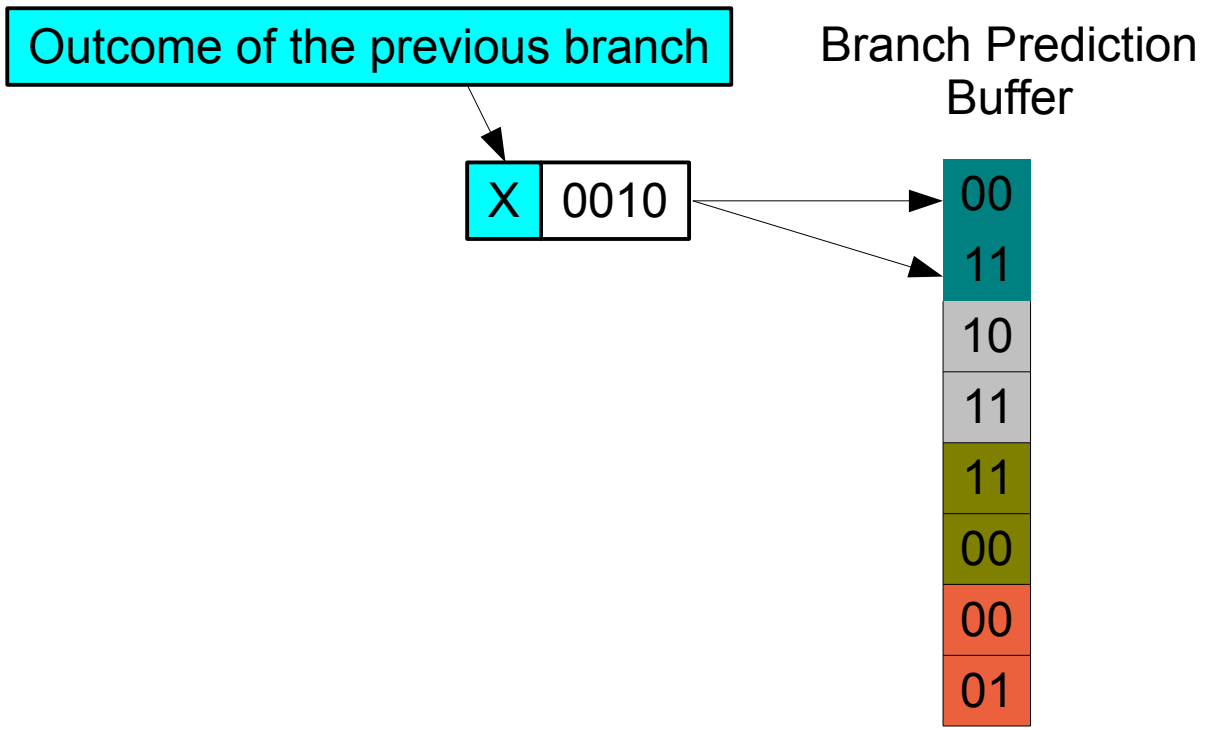
eqntott code

```
if (aa == 2)
    aa = 0;
if (bb == 2)
    bb = 0;
if (aa!=bb) {
```

Correlating Branch Predictors

eqntott code

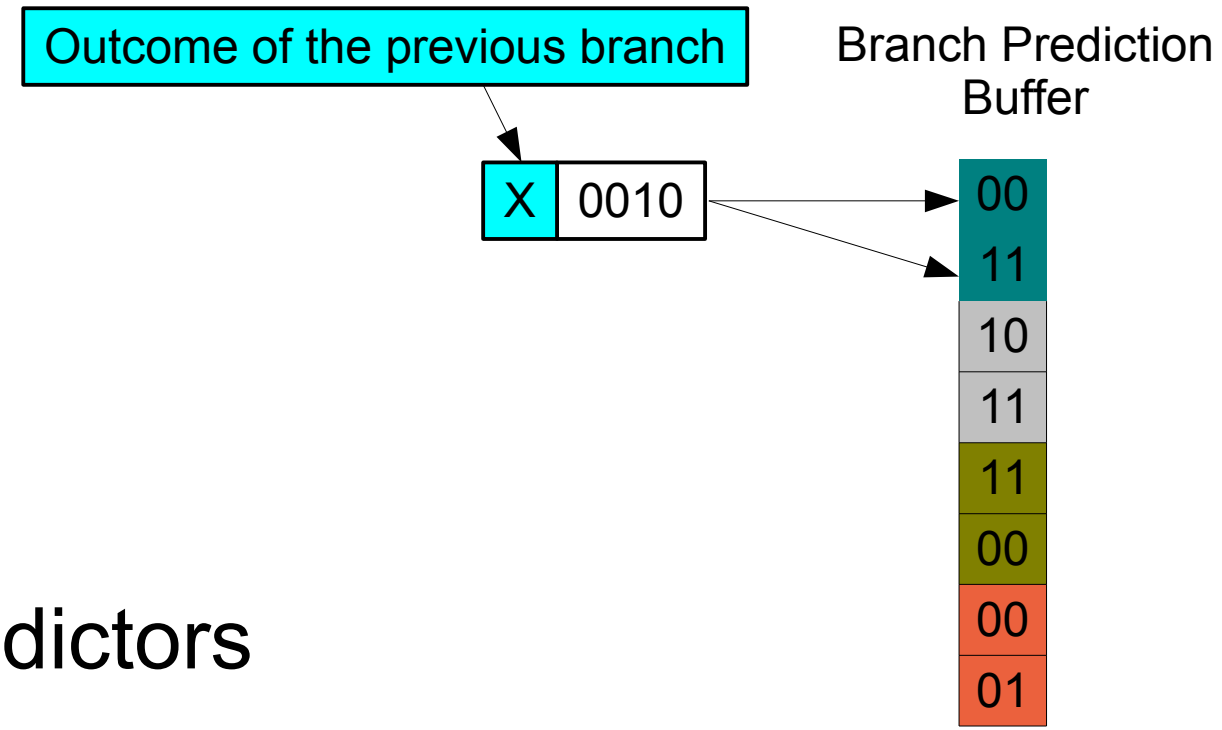
```
if (aa == 2)
    aa = 0;
if (bb == 2)
    bb = 0;
if (aa!=bb) {
```



Correlating Branch Predictors

eqntott code

```
if (aa == 2)
    aa = 0;
if (bb == 2)
    bb = 0;
if (aa!=bb) {
```

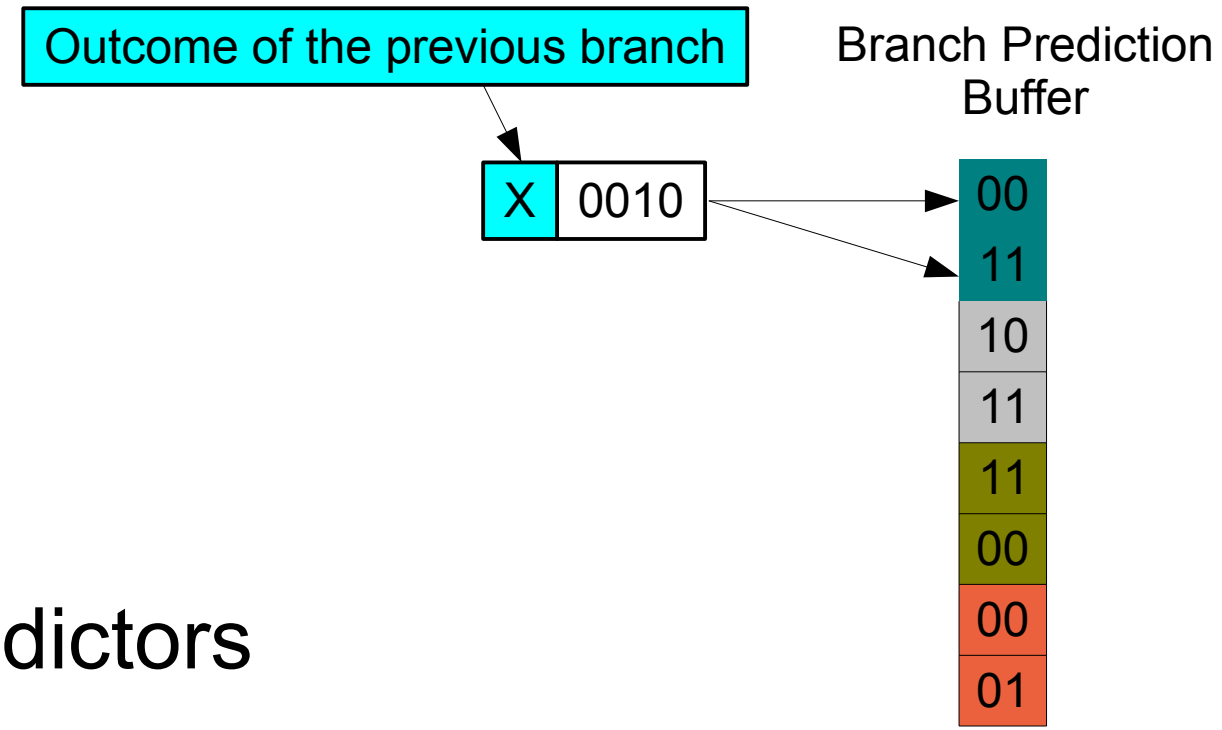


- Two-level predictors
  - (1,2) predictor

Correlating Branch Predictors

eqntott code

```
if (aa == 2)
    aa = 0;
if (bb == 2)
    bb = 0;
if (aa!=bb) {
```

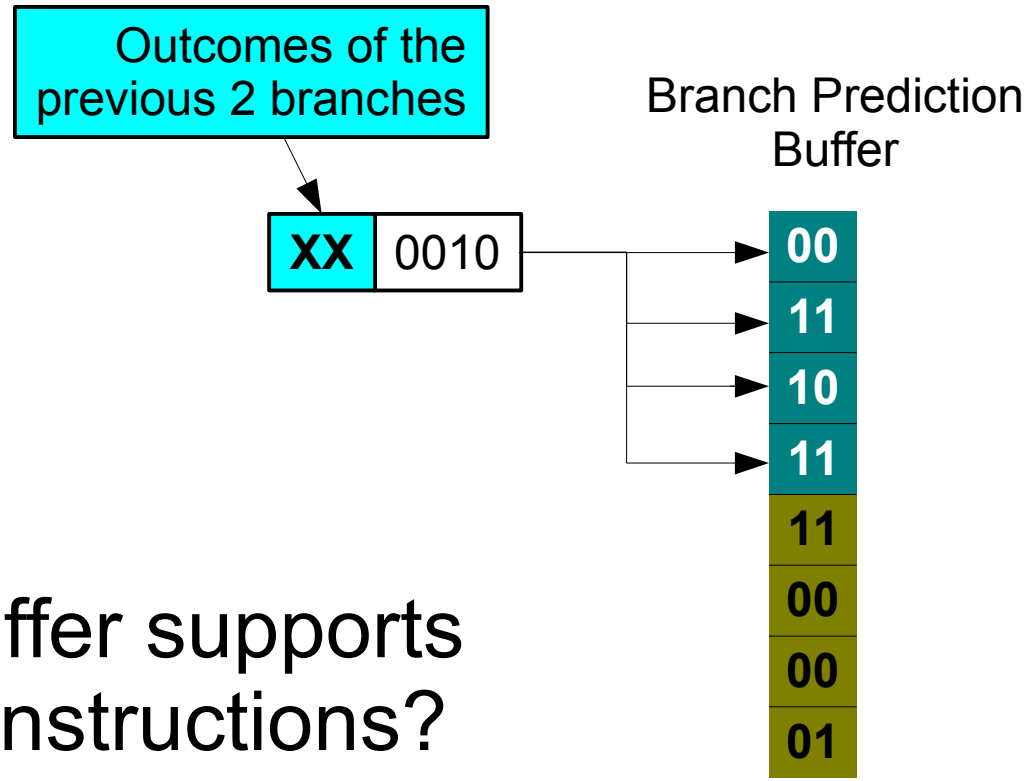


- Two-level predictors
  - (1,2) predictor

Yeh and Patt, Alternative implementations of two-level adaptive branch prediction, ISCA, 1992.

Correlating Branch Predictors

```
if (aa == 2)
    aa = 0;
if (bb == 2)
    bb = 0;
if (aa!=bb) {
```

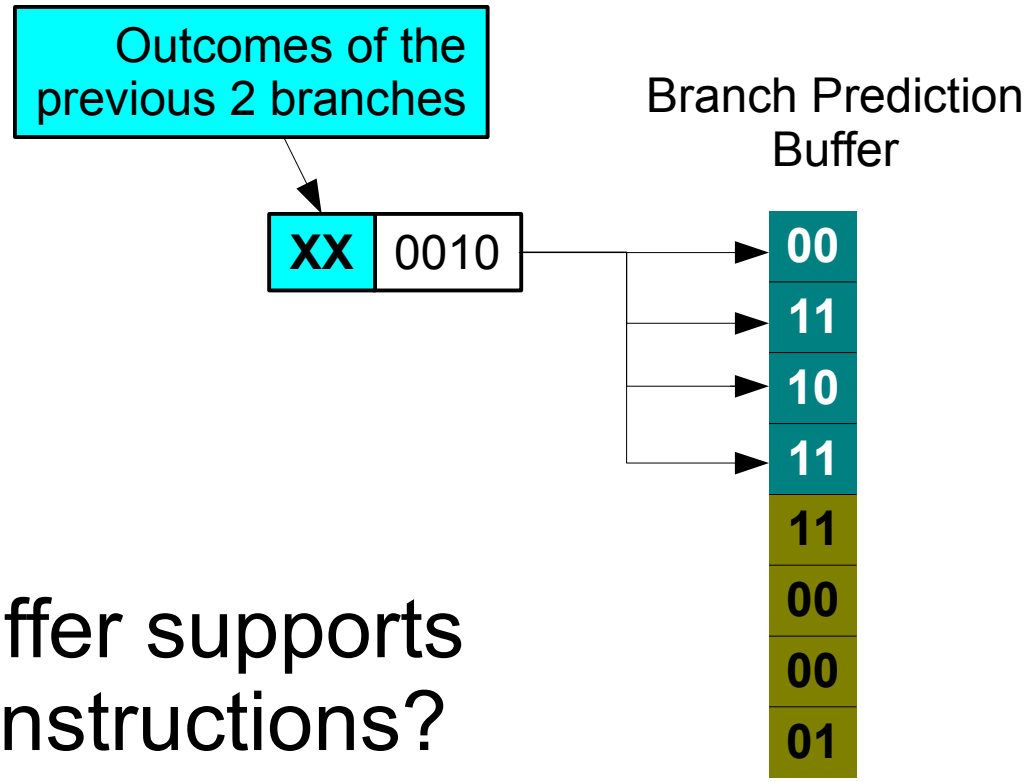


- A 4096 bit, (2,2) buffer supports how many branch instructions?

Yeh and Patt, Alternative implementations of two-level adaptive branch prediction, ISCA, 1992.

Correlating Branch Predictors

```
if (aa == 2)
    aa = 0;
if (bb == 2)
    bb = 0;
if (aa!=bb) {
```

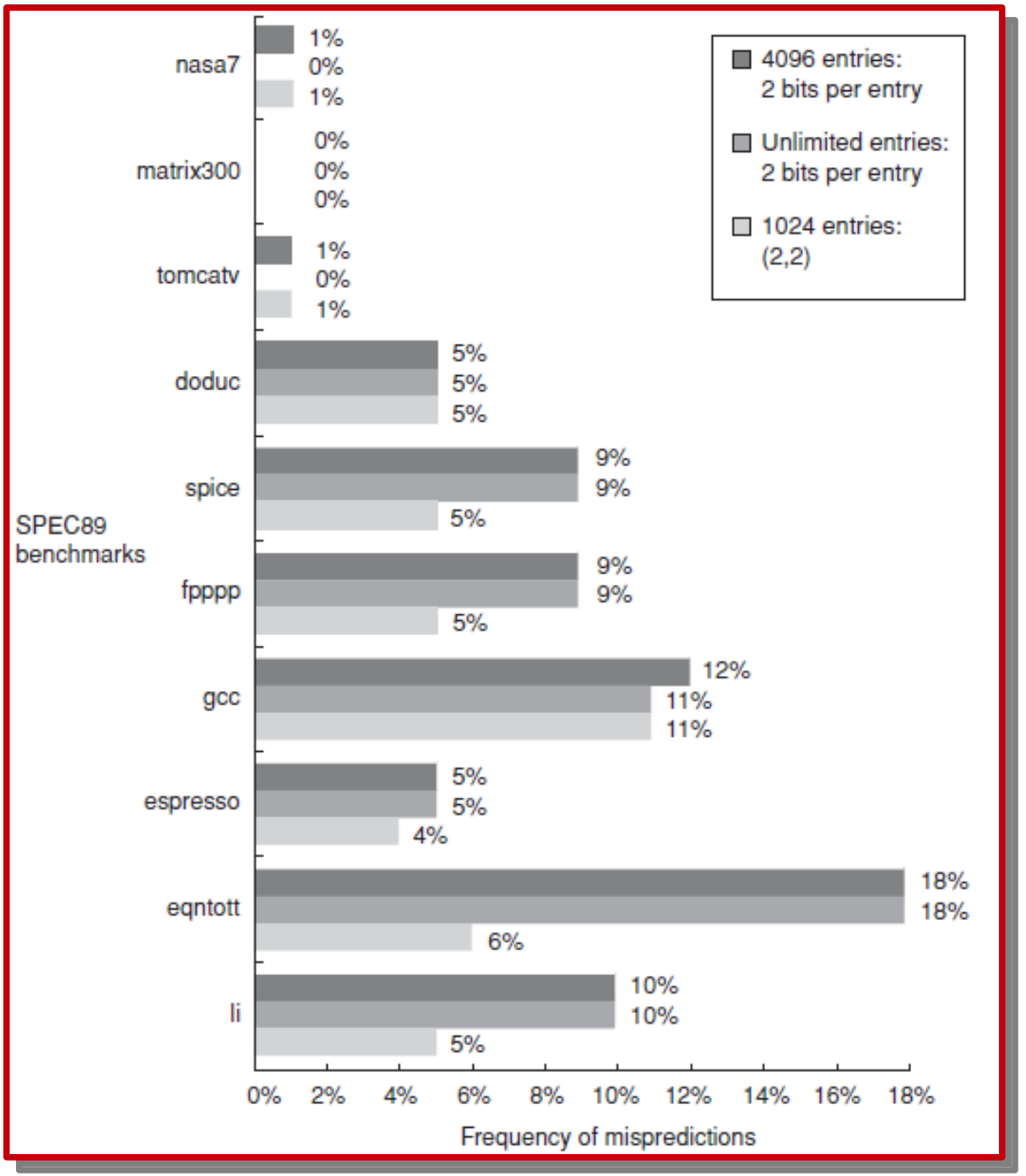


- A 4096 bit, (2,2) buffer supports how many branch instructions?

$$(m, n)\ BPB\ bits = 2^m \times n \times No.\ of\ prediction\ entries$$

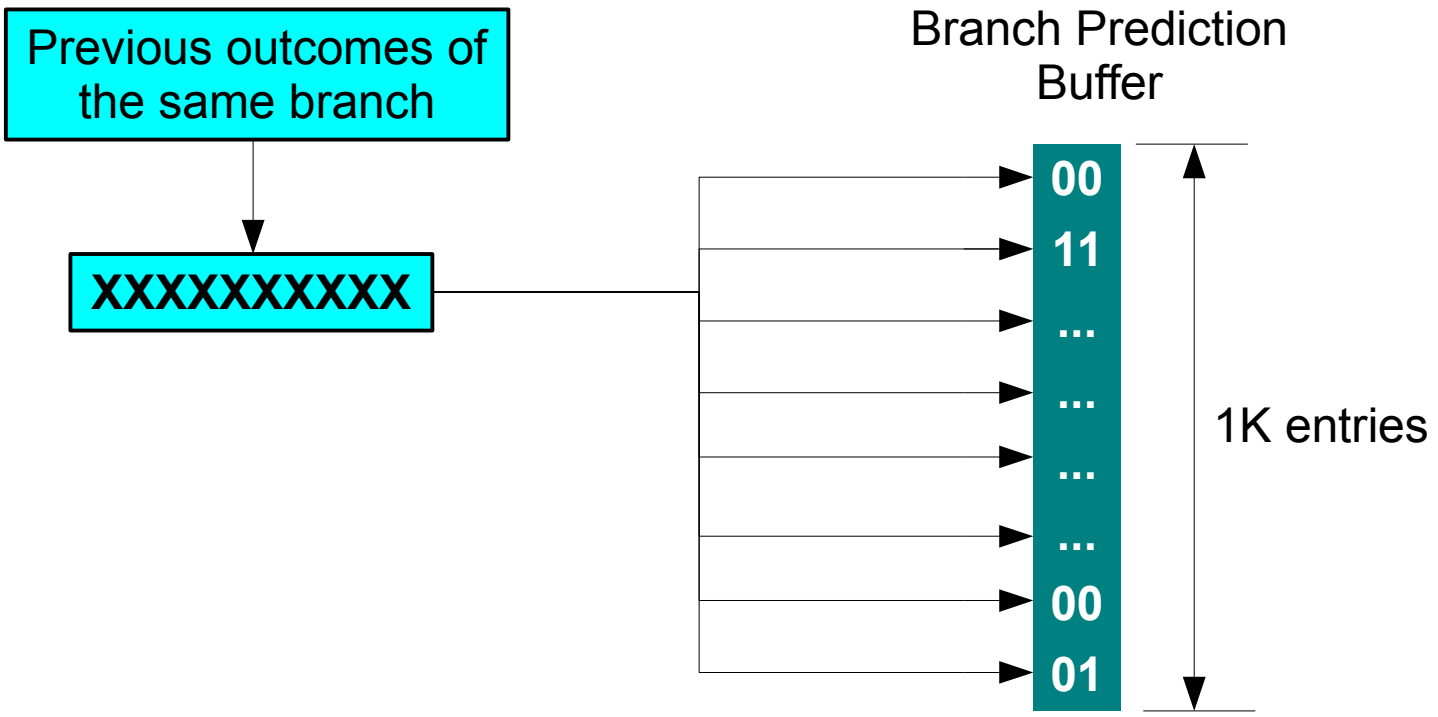
Yeh and Patt, Alternative implementations of two-level adaptive branch prediction, ISCA, 1992.

Correlating Branch Predictors



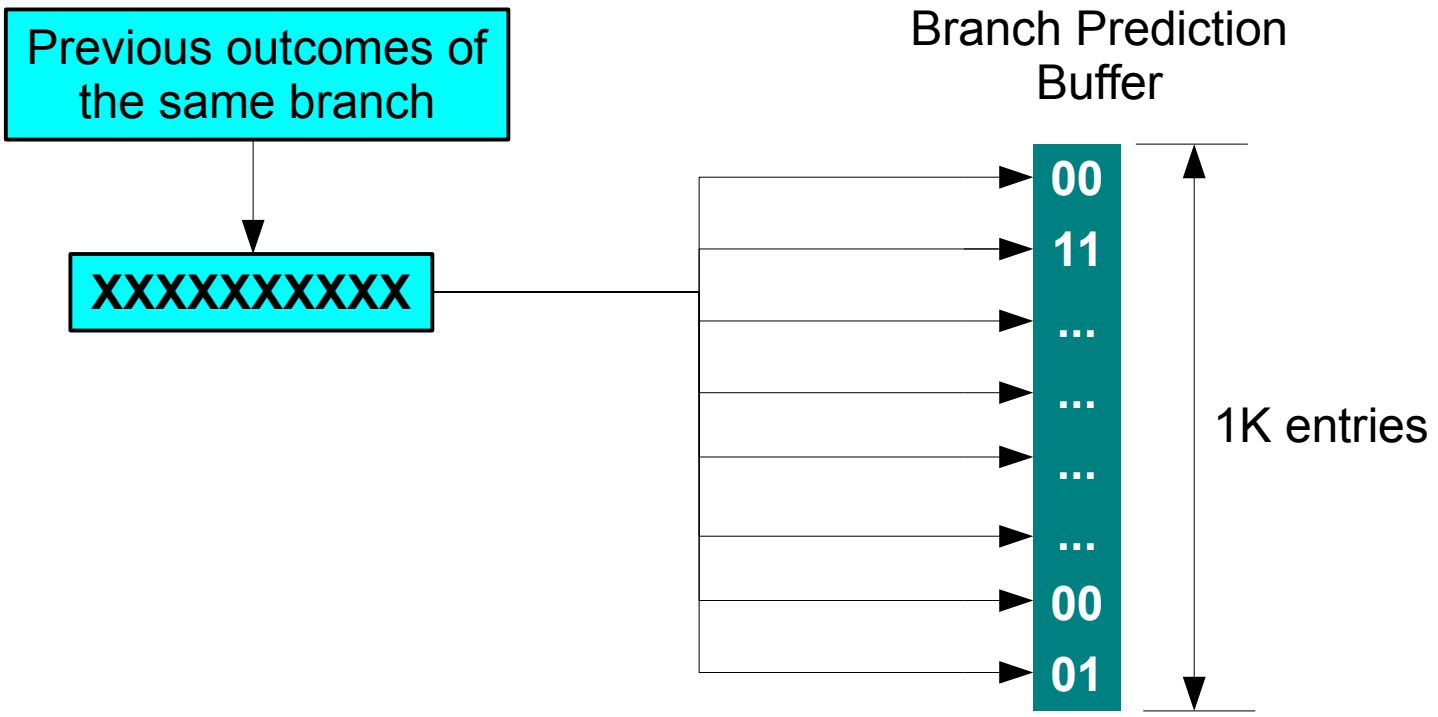
Yeh and Patt, ISCA, 1992.

Local Predictors



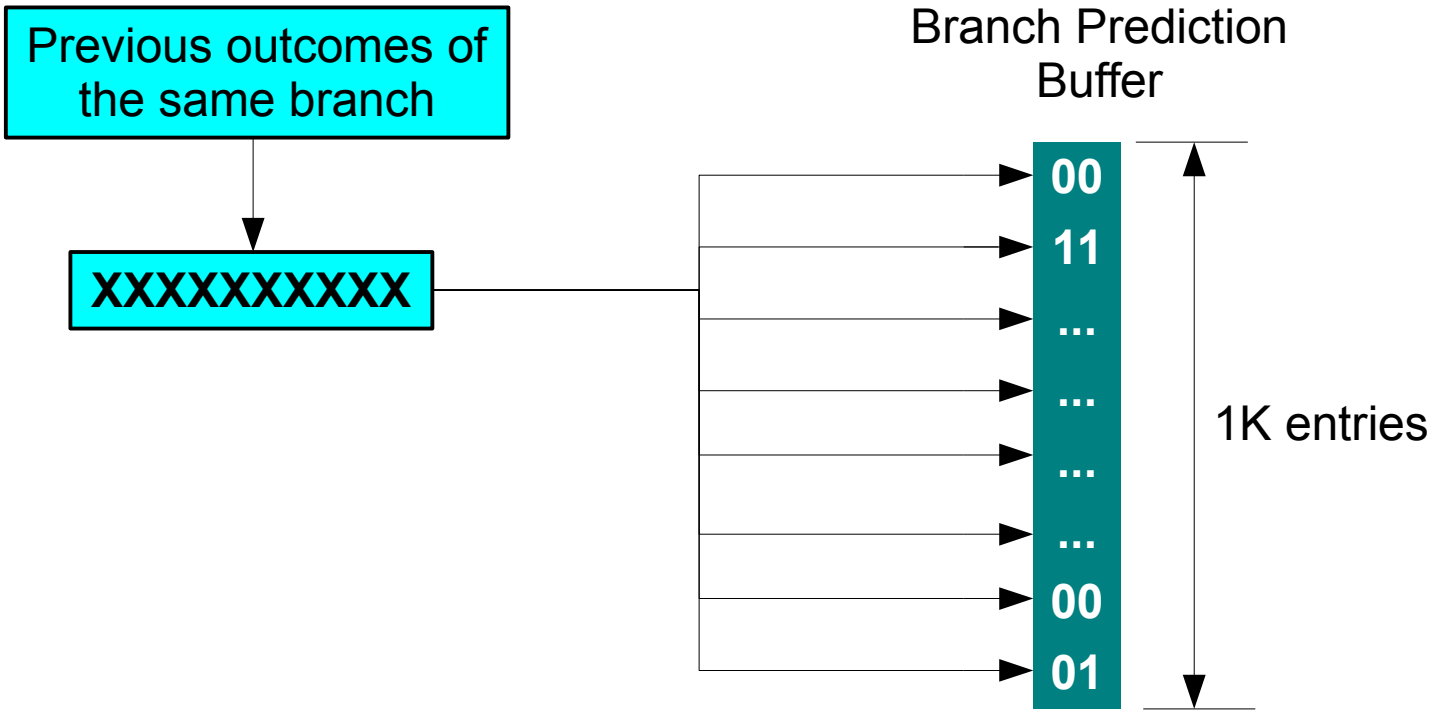
Local Predictors

- A history of branch behaviour is recorded

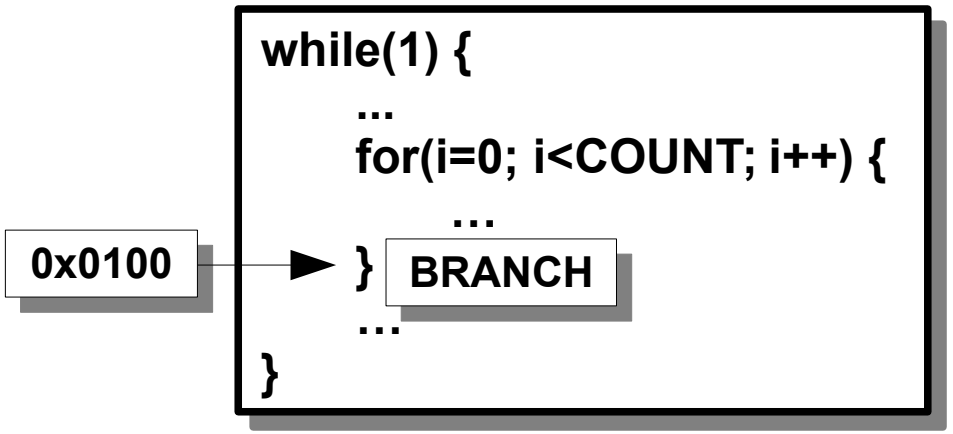


Local Predictors

- A history of branch behaviour is recorded
- One for each possible combination of outcomes for the last *n* occurrences of this branch



Local Predictors – Example

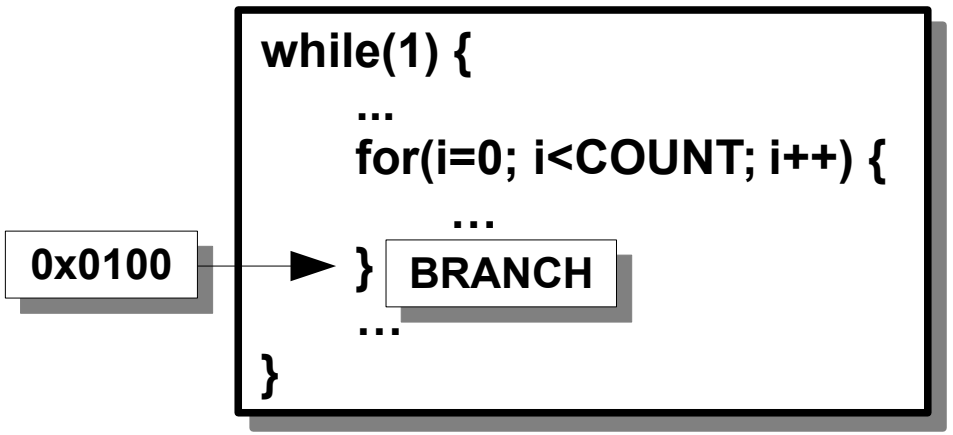


Branch instruction behaviour

… T T N T T T T N T T T T N T T …

… 1 1 0 1 1 1 1 0 1 1 1 1 0 1 1 …

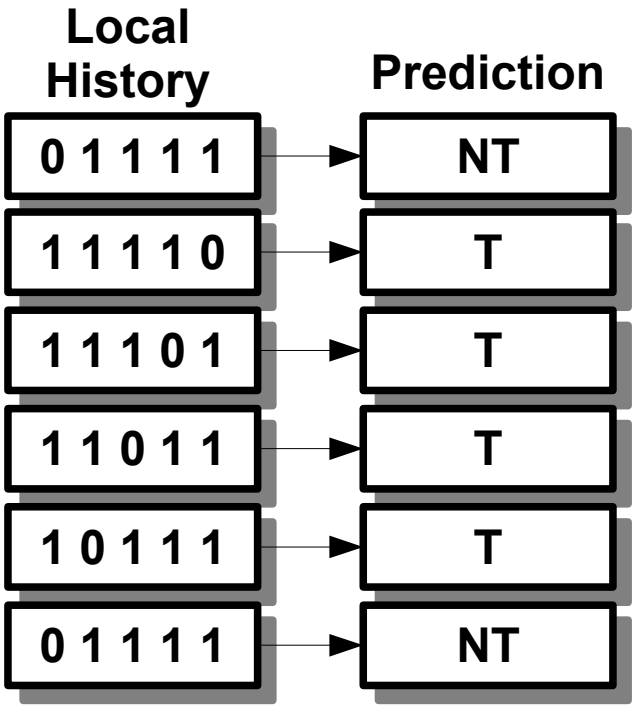
Local Predictors – Example



Branch instruction behaviour

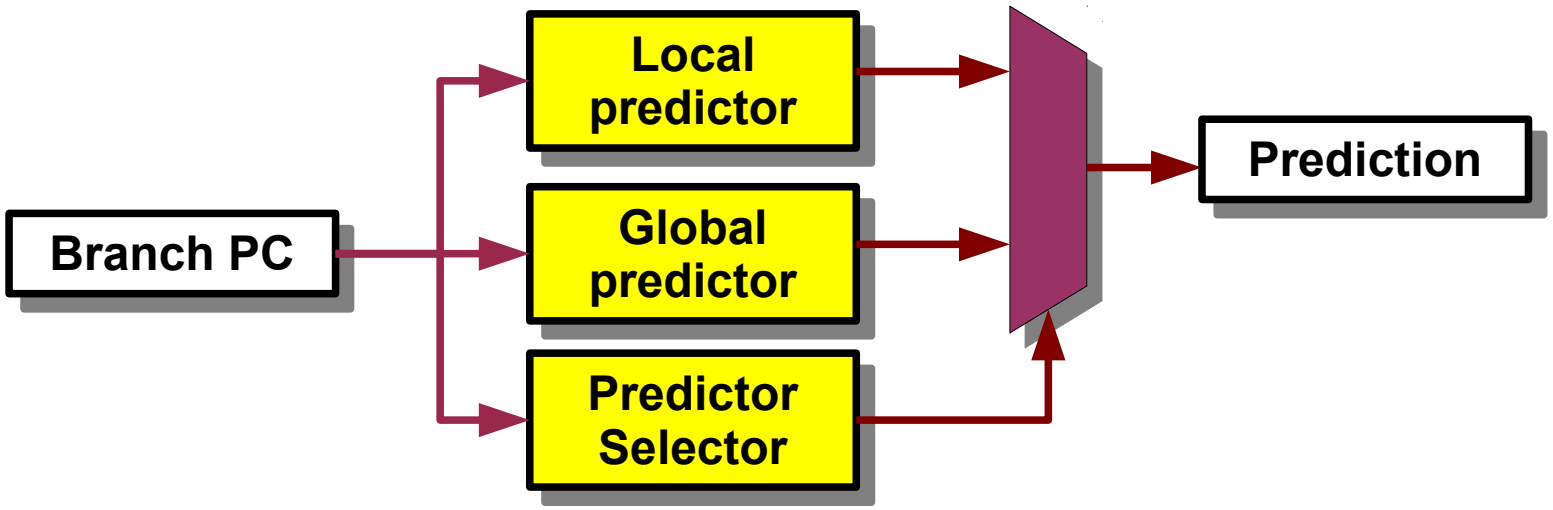
… T T N T T T T N T T T T N T T …

… 1 1 0 1 1 1 1 0 1 1 1 1 0 1 1 …



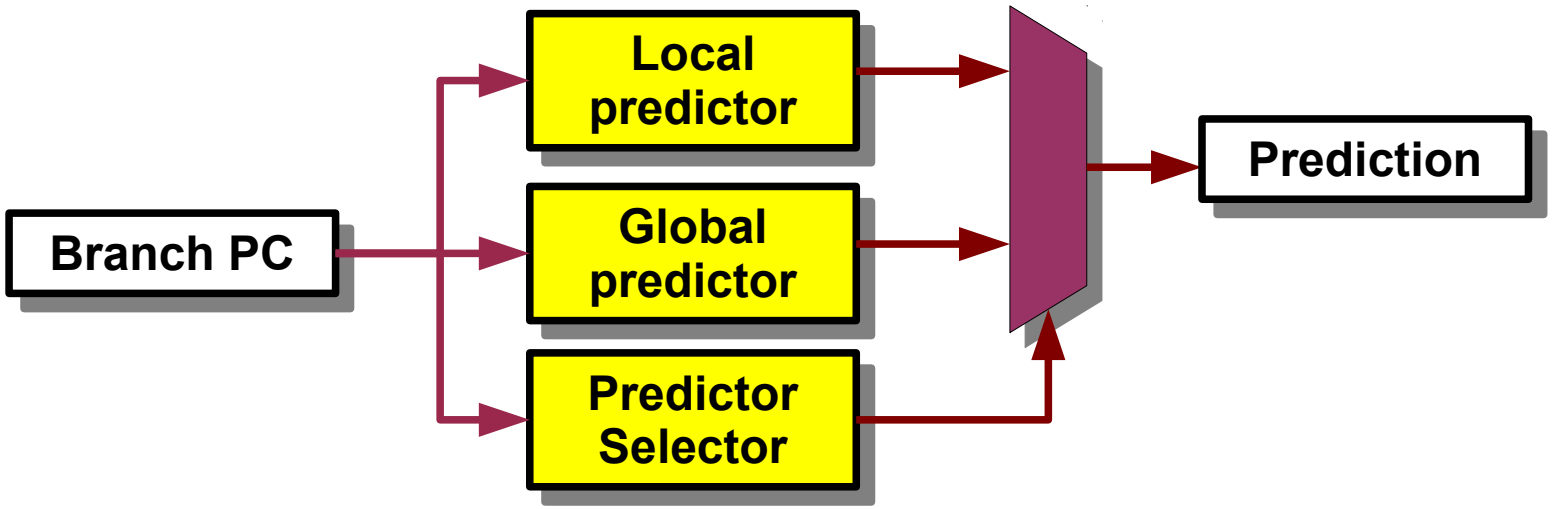
Tournament Predictors

Tournament Predictors



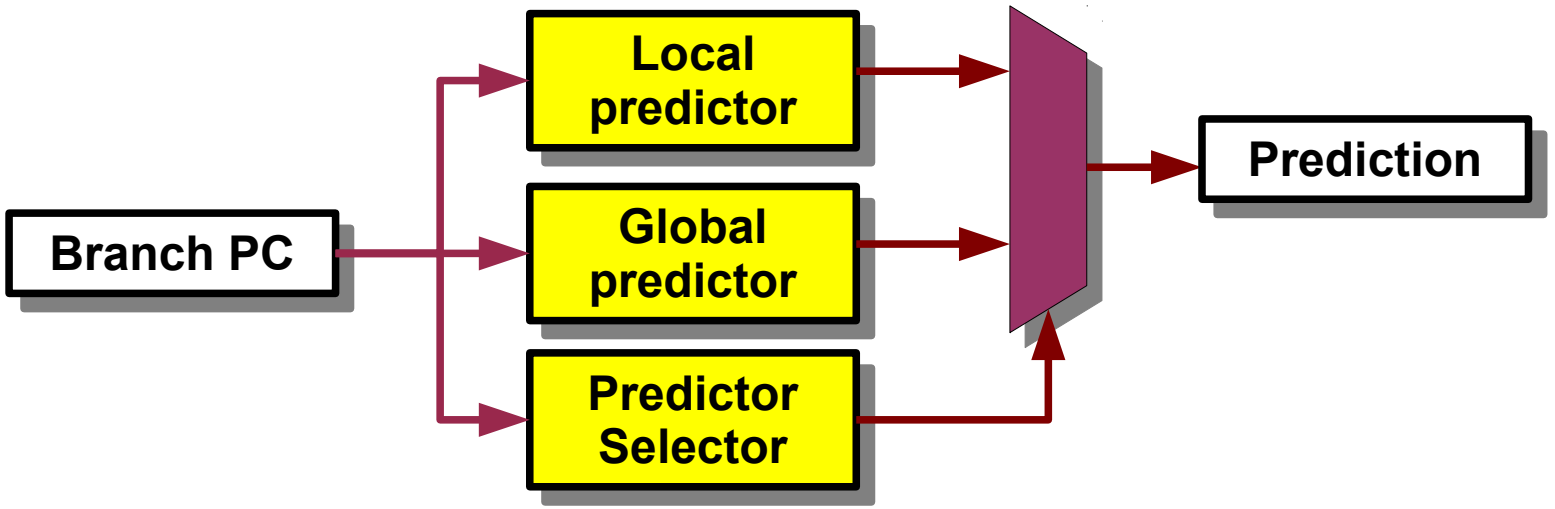
Tournament Predictors

- Use multiple predictors: Global, local or mix

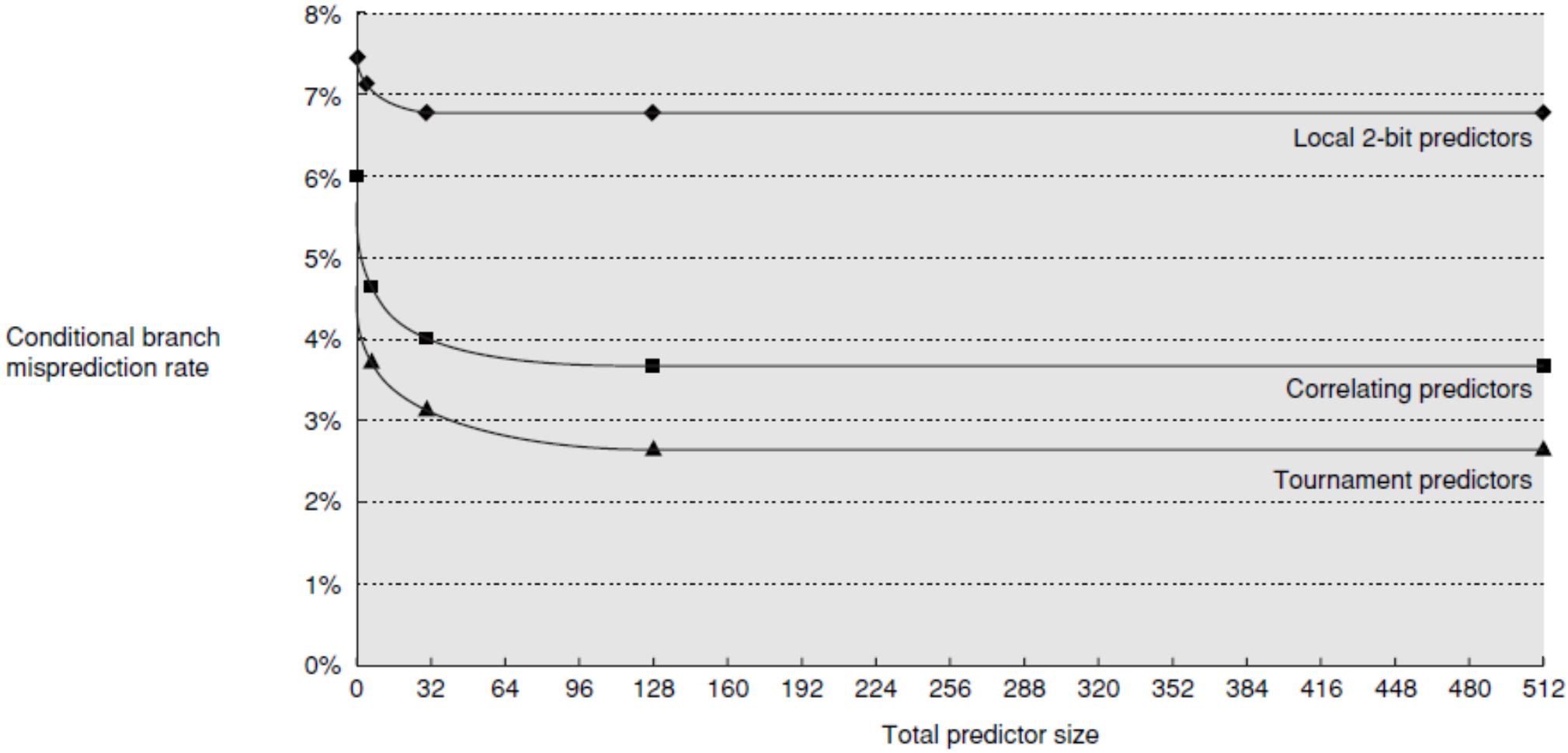


Tournament Predictors

- Use multiple predictors: Global, local or mix
- Combine them with a selector
  - 2 bit saturating counter to select the right predictor for the branch (global vs. local)



Tournament Predictors



Outline

- Pipeline, Pipelined datapath
- Dependences, Hazards
  - Structural, Data - Stalling, Forwarding
- Control Hazards
- Branch prediction

Xtra Slides

Pipelined vs. Nonpipelined Implementation

- Ratio of execution times between the two?
  - For 10^6 instructions?
- Pipelining increases the instruction throughput opposed to individual instruction execution time.

Throughput of the Pipeline

- Number of tasks completed in unit time (one second)

$$w = \eta \times f$$

f: frequency of operation

Efficiency of the Pipeline

- Percentage of stages accomplishing tasks related to the instruction in execution

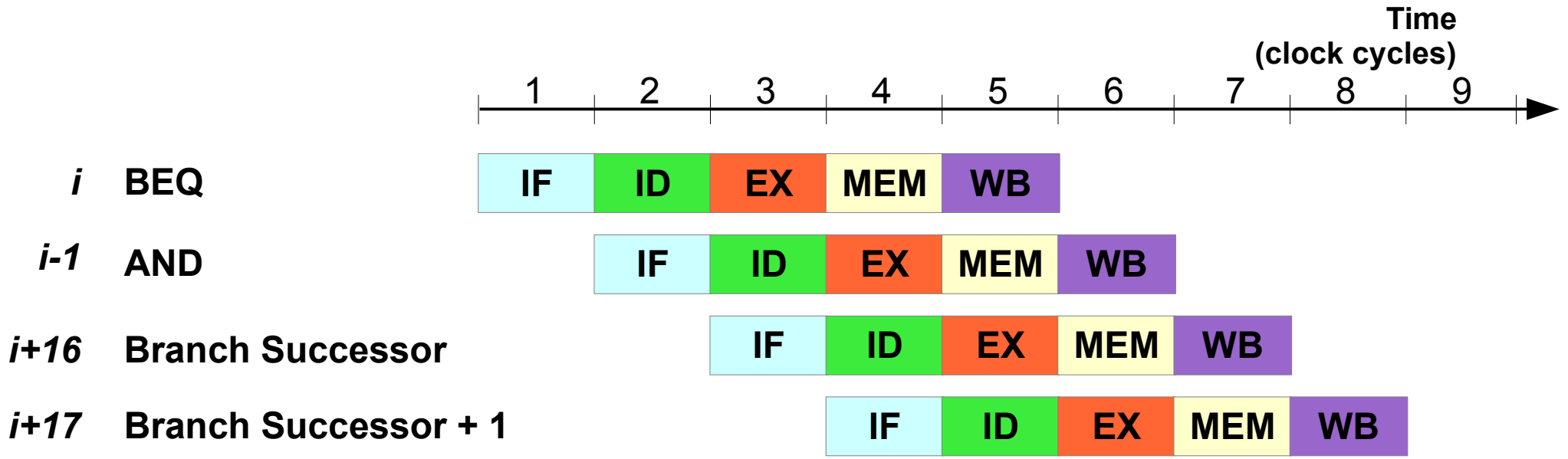
$$\eta = \frac{No.\ of\ \ Instructions}{Instruction\ Execution\ Time}$$

$$\eta = \frac{n}{k + (n-1)}$$

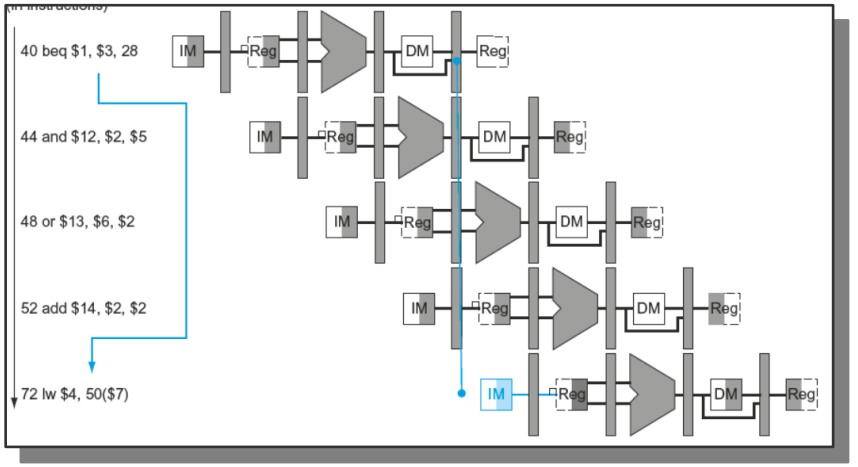
n: number of instructions in the program.
k: number of pipeline stages

Reducing Pipeline Branch Penalties

- Freeze the pipeline
- Predict Taken
- Predict Untaken
- Fill Branch Delay Slot

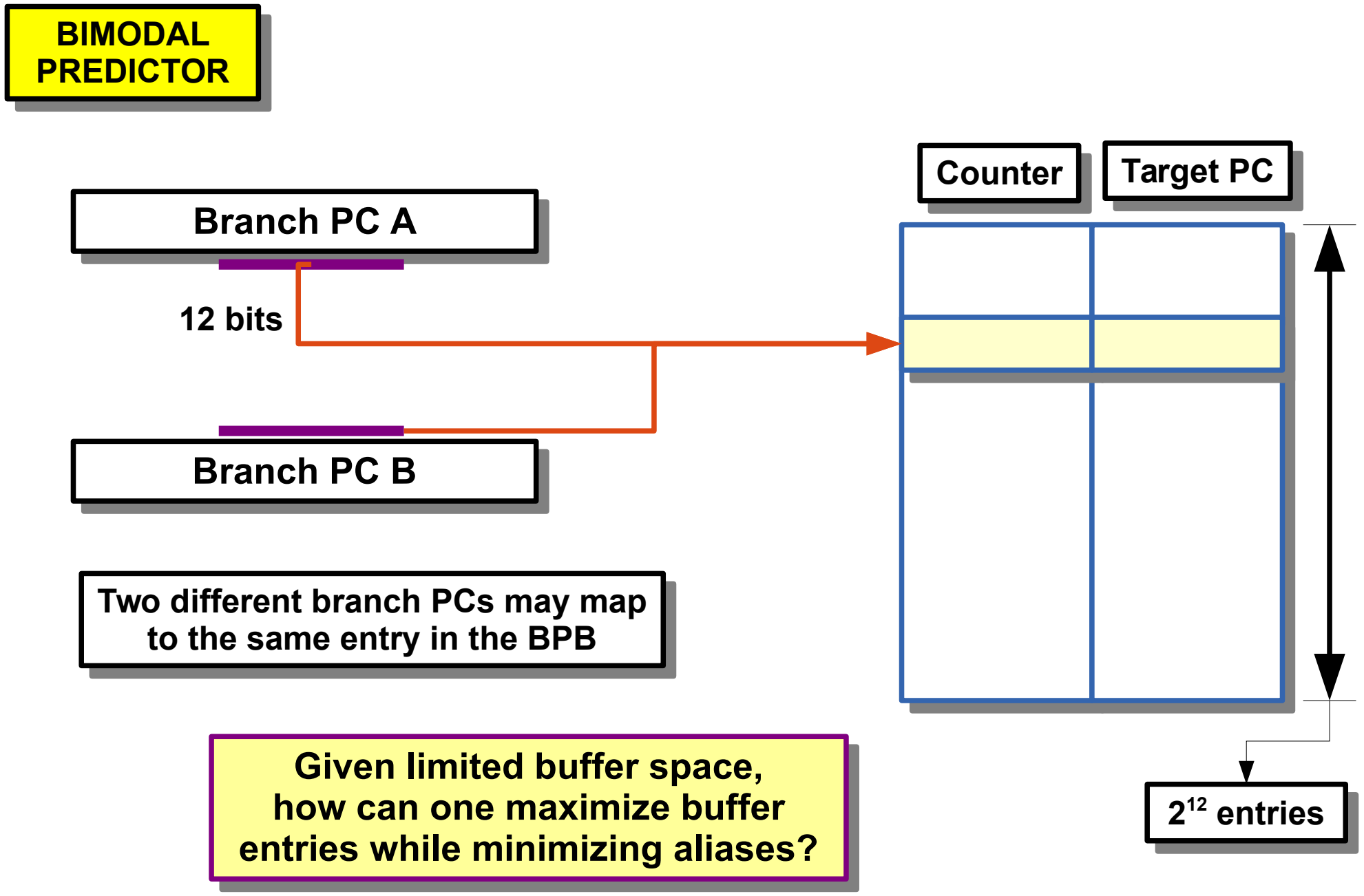


Control Hazard

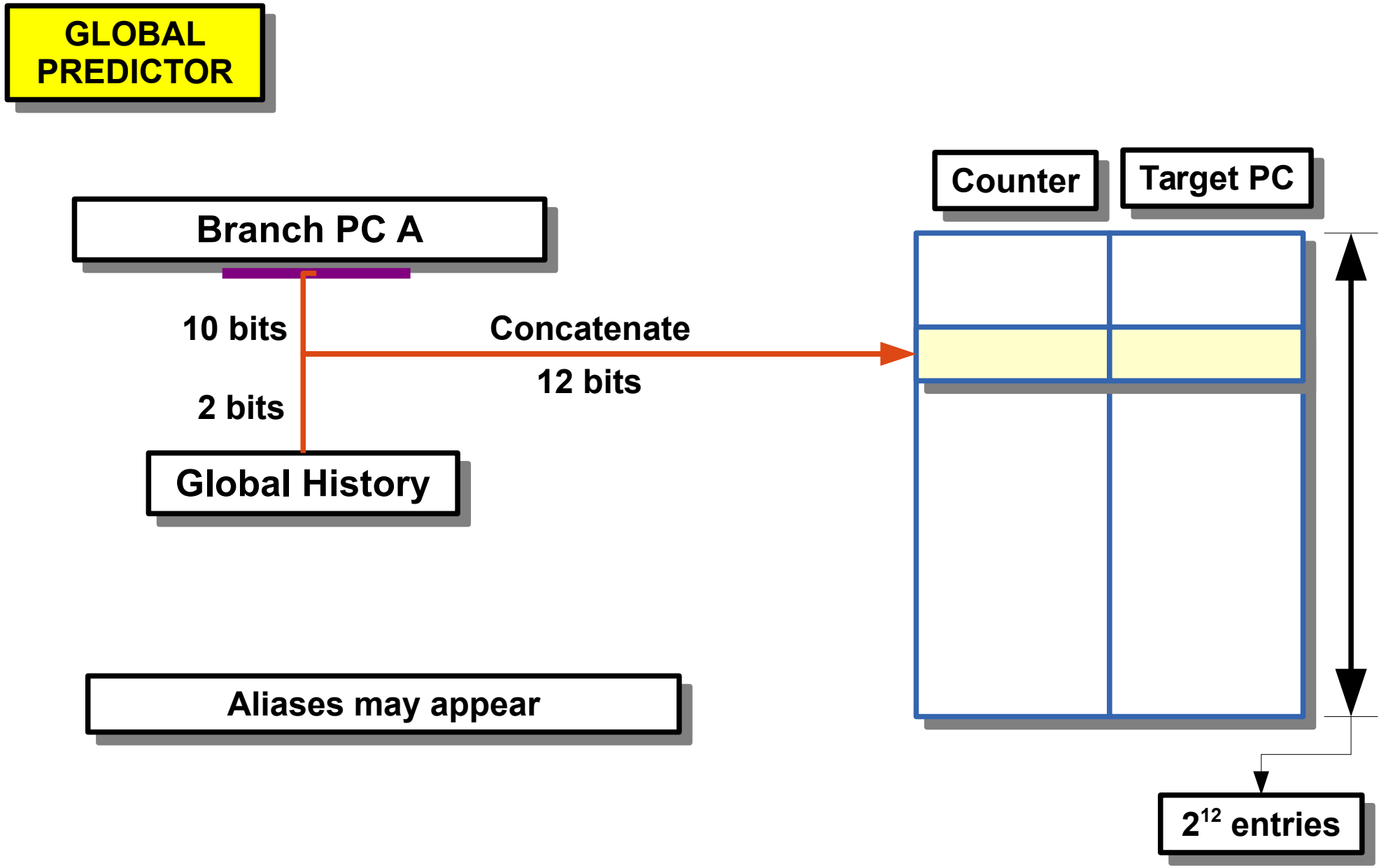


Branch Penalty = 3 cycles

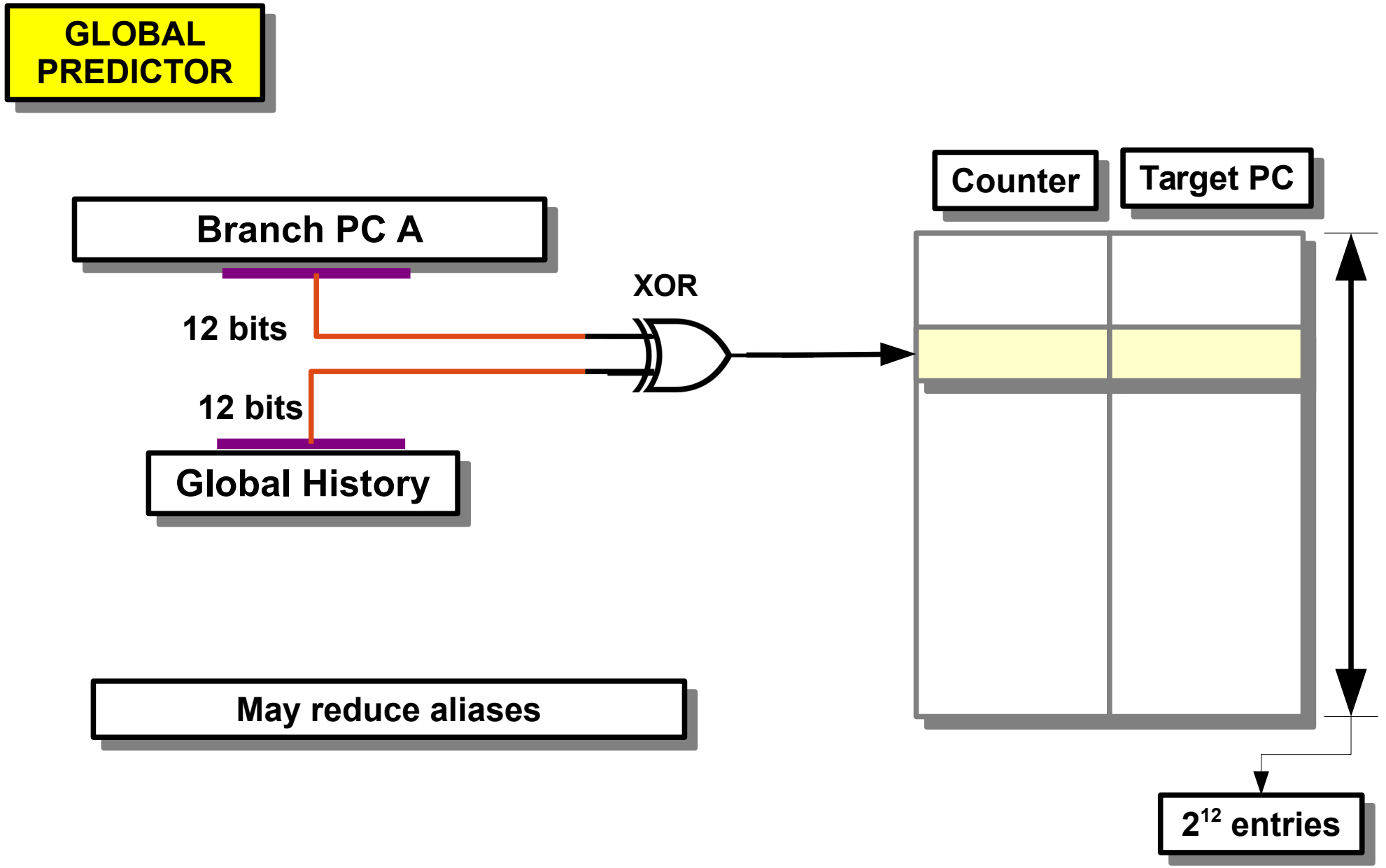
Branch Prediction Buffer



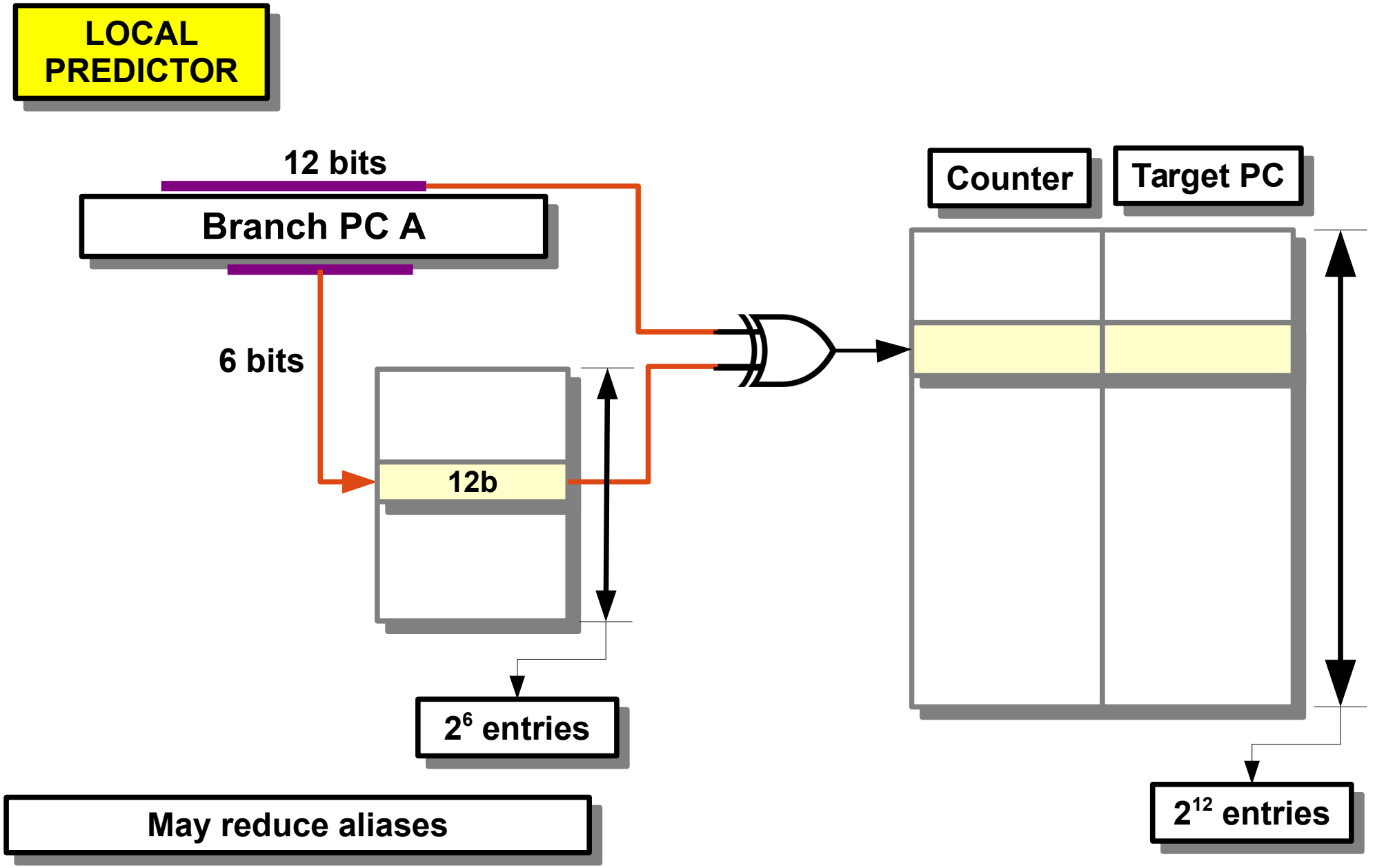
BPB – Global Predictor



BPB – Aliases


GLOBAL PREDICTOR
Branch PC A
12 bits
12 bits
Global History
XOR
Counter
Target PC
2^{12} entries
May reduce aliases

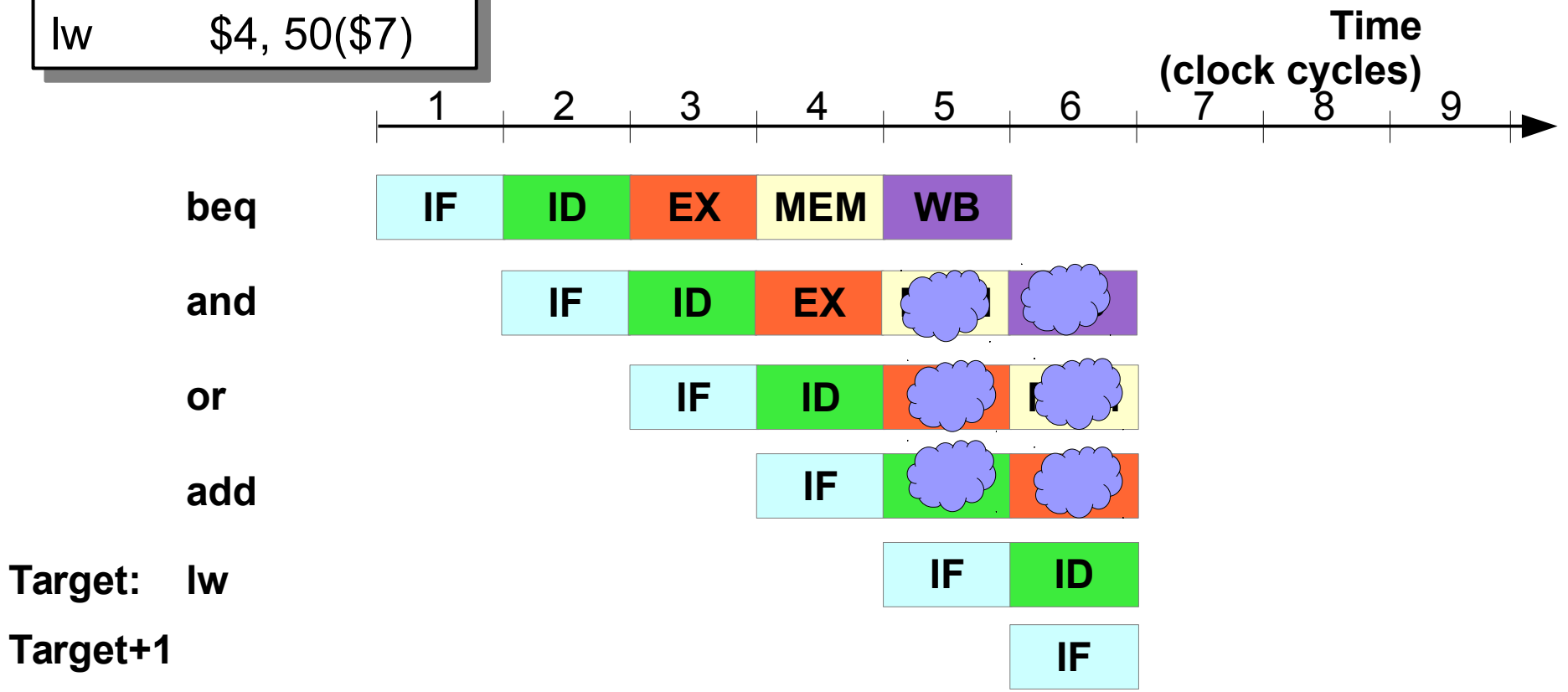
BPB – Local Predictor



Control Hazard

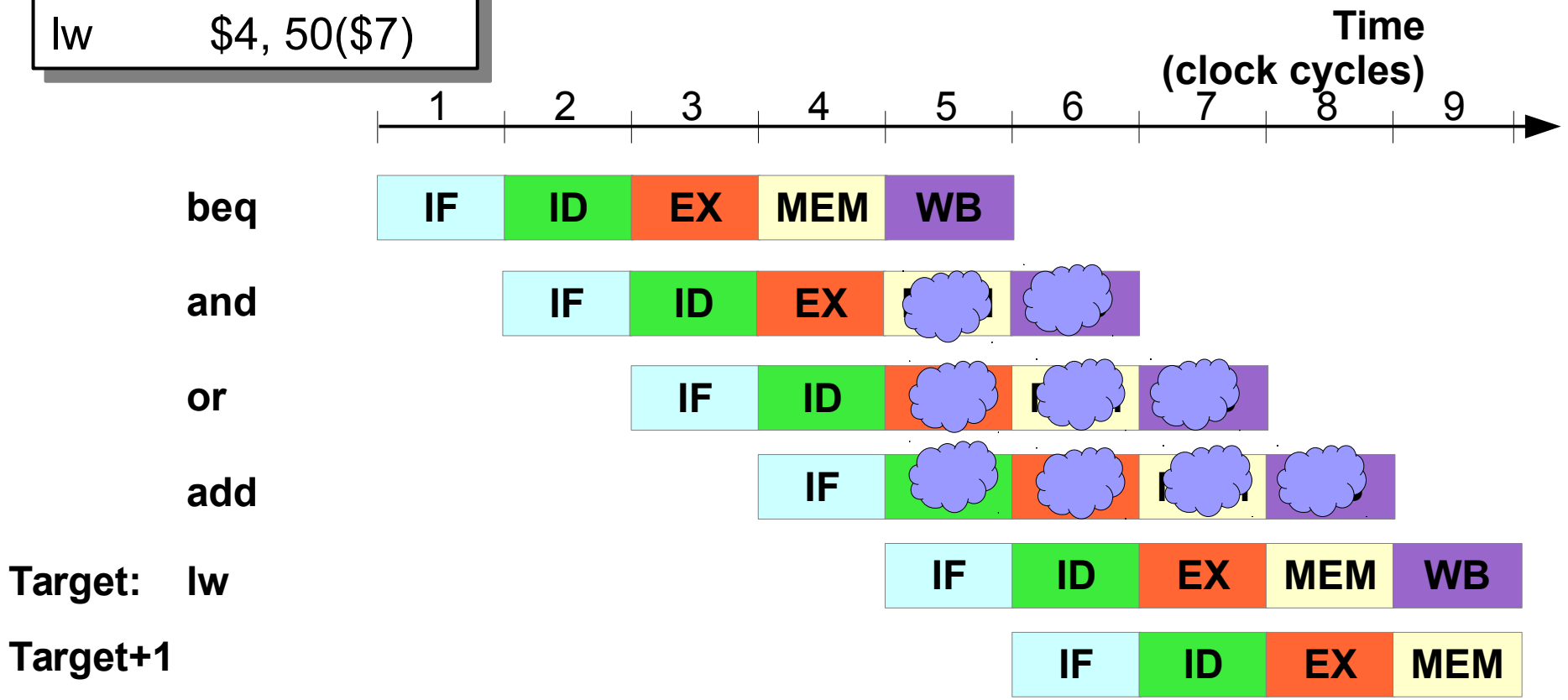
```
beq     $1, $3, 28
and     $12, $2, $5
or      $13, $6, $2
add     $14, $2, $2
...
lw      $4, 50($7)
```

WRONG !!!



Control Hazard

```
beq     $1, $3, 28
and     $12, $2, $5
or      $13, $6, $2
add     $14, $2, $2
...
lw      $4, 50($7)
```



Early Branch Evaluation

```
beq     $1, $3, 28
```

- Evaluate branch at the earliest
- When are the source operands available?
  - After Register Read

BPB – Local Predictor