CPT_S 260 Intro to Computer Architecture
Lecture 38

Forwarding and Hazard Detection
April 18, 2022

Ganapati Bhat
School of Electrical Engineering and Computer Science
Washington State University

Announcements

- Quiz 5 is posted
- Homework 6 is also posted
- We will have another short quiz next week
- I'll discuss homework 7 in class for final exam
- Exam 2 grades will be available this week

Recap: Double Data Hazard

- **Consider the sequence:**

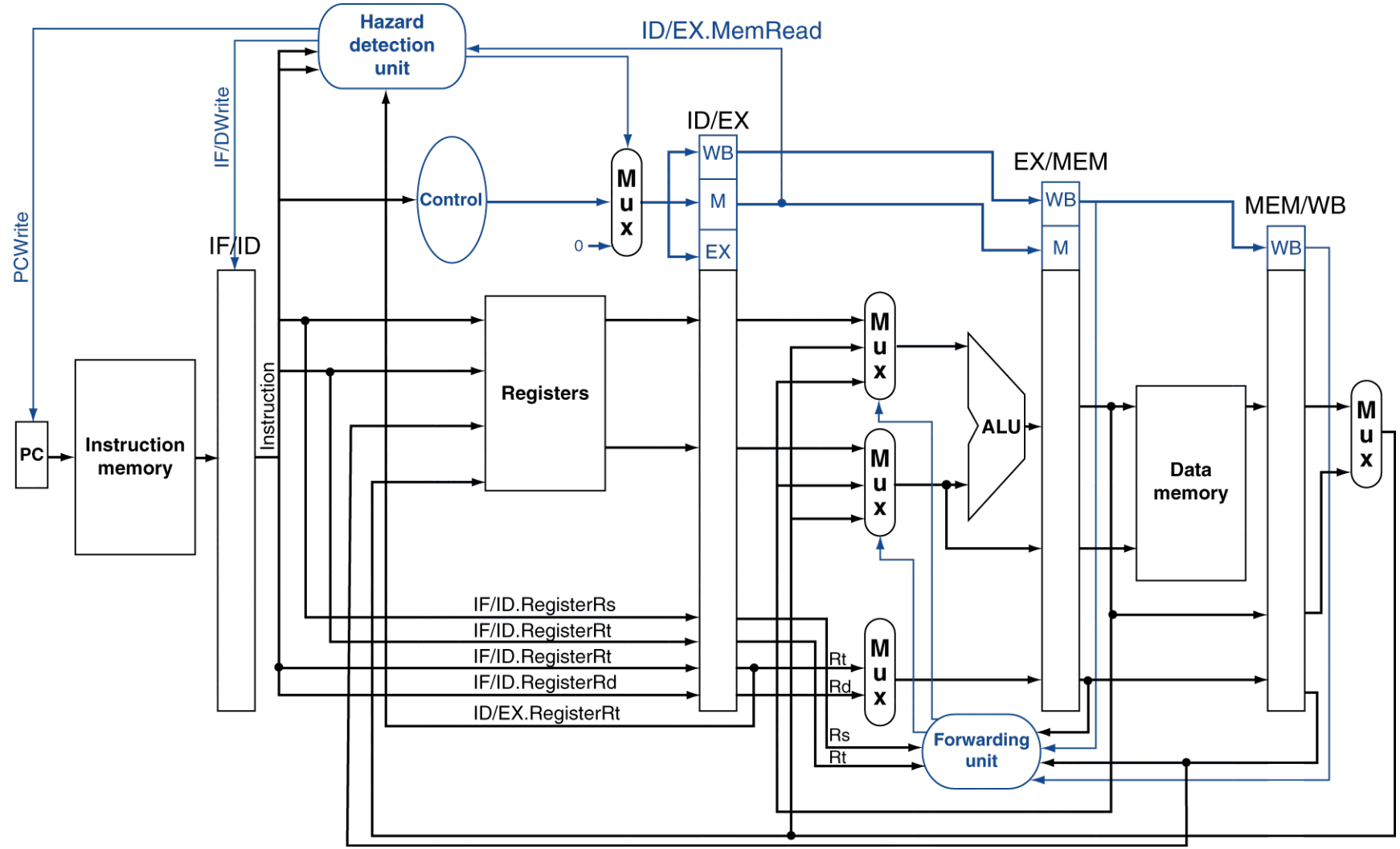
```
add $1,$1,$2
add $1,$1,$3
add $1,$1,$4
```

- **Both hazards occur**
  - Want to use the most recent
- **Revise MEM hazard condition**
  - Only fwd if EX hazard condition isn't true

Revised Forwarding Condition

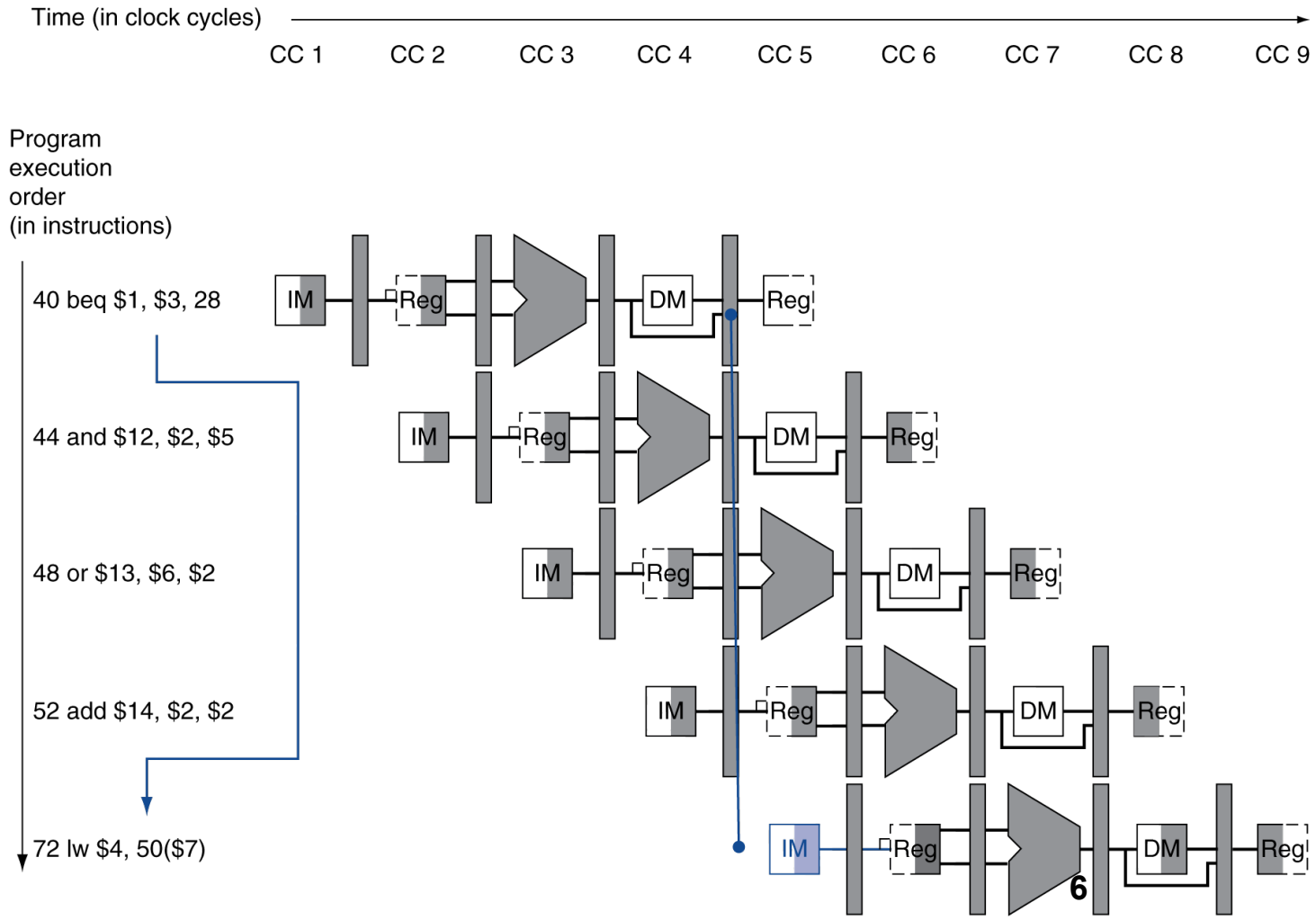
- **MEM hazard**
  - if (MEM/WB.RegWrite and (MEM/WB.RegisterRd ≠ 0)
    and not (EX/MEM.RegWrite and (EX/MEM.RegisterRd ≠ 0)
    and (EX/MEM.RegisterRd = ID/EX.RegisterRs))
    and (MEM/WB.RegisterRd = ID/EX.RegisterRs))
    ForwardA = 01
  - if (MEM/WB.RegWrite and (MEM/WB.RegisterRd ≠ 0)
    and not (EX/MEM.RegWrite and (EX/MEM.RegisterRd ≠ 0)
    and (EX/MEM.RegisterRd = ID/EX.RegisterRt))
    and (MEM/WB.RegisterRd = ID/EX.RegisterRt))
    ForwardB = 01

Recap: Datapath with Hazard Detection



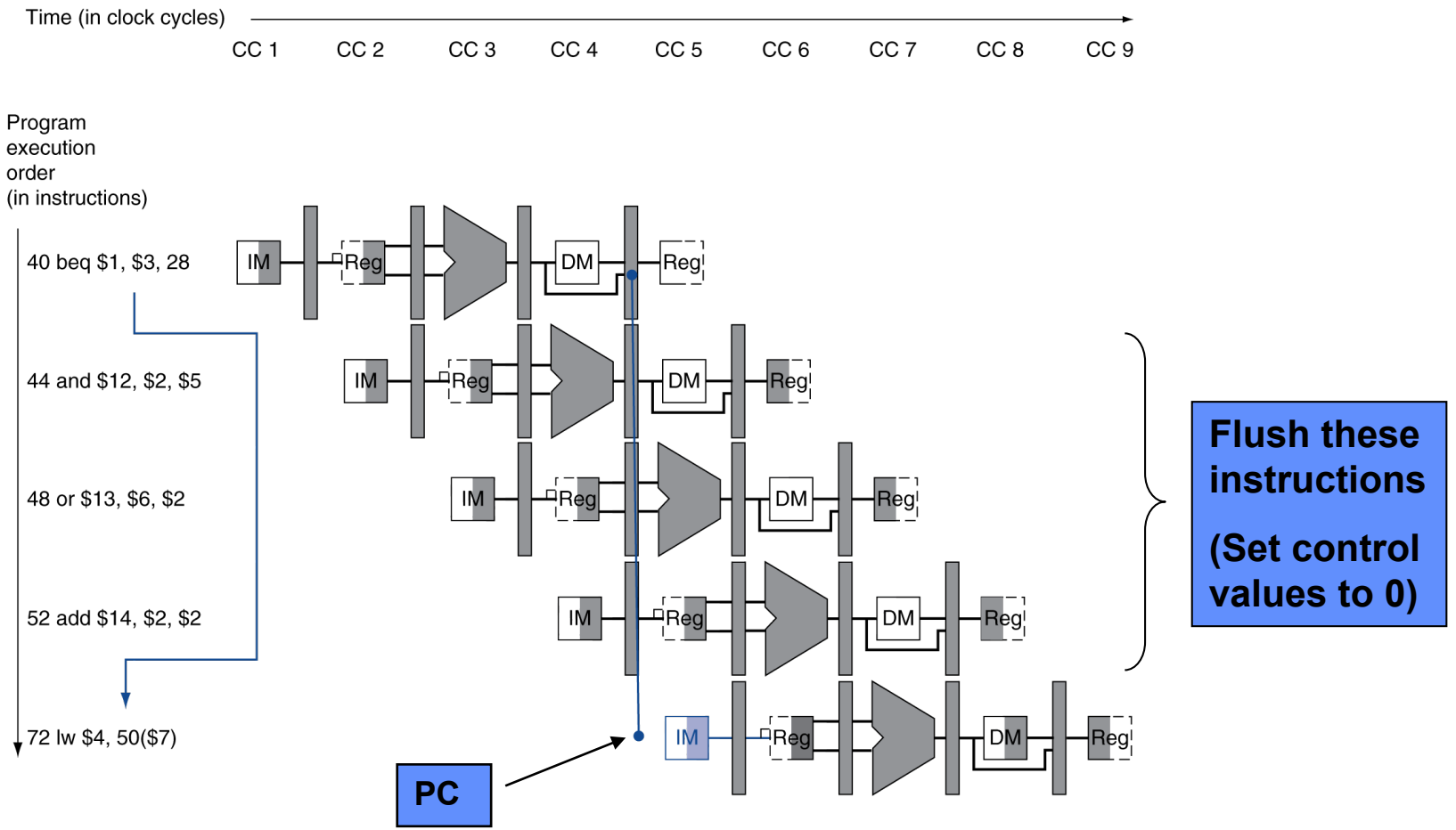
Branch Hazards

- **If branch outcome determined in MEM**

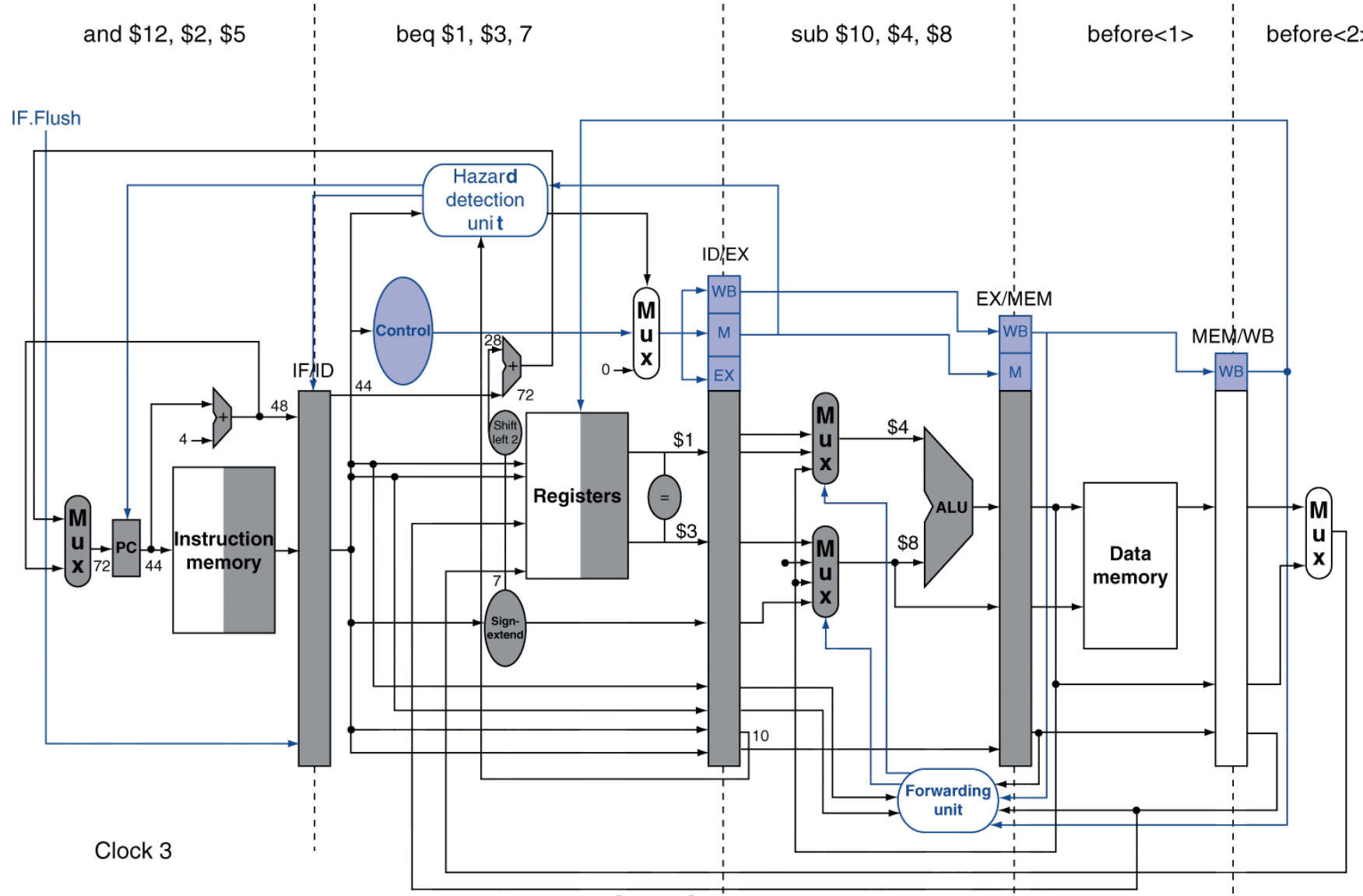


Branch Hazards

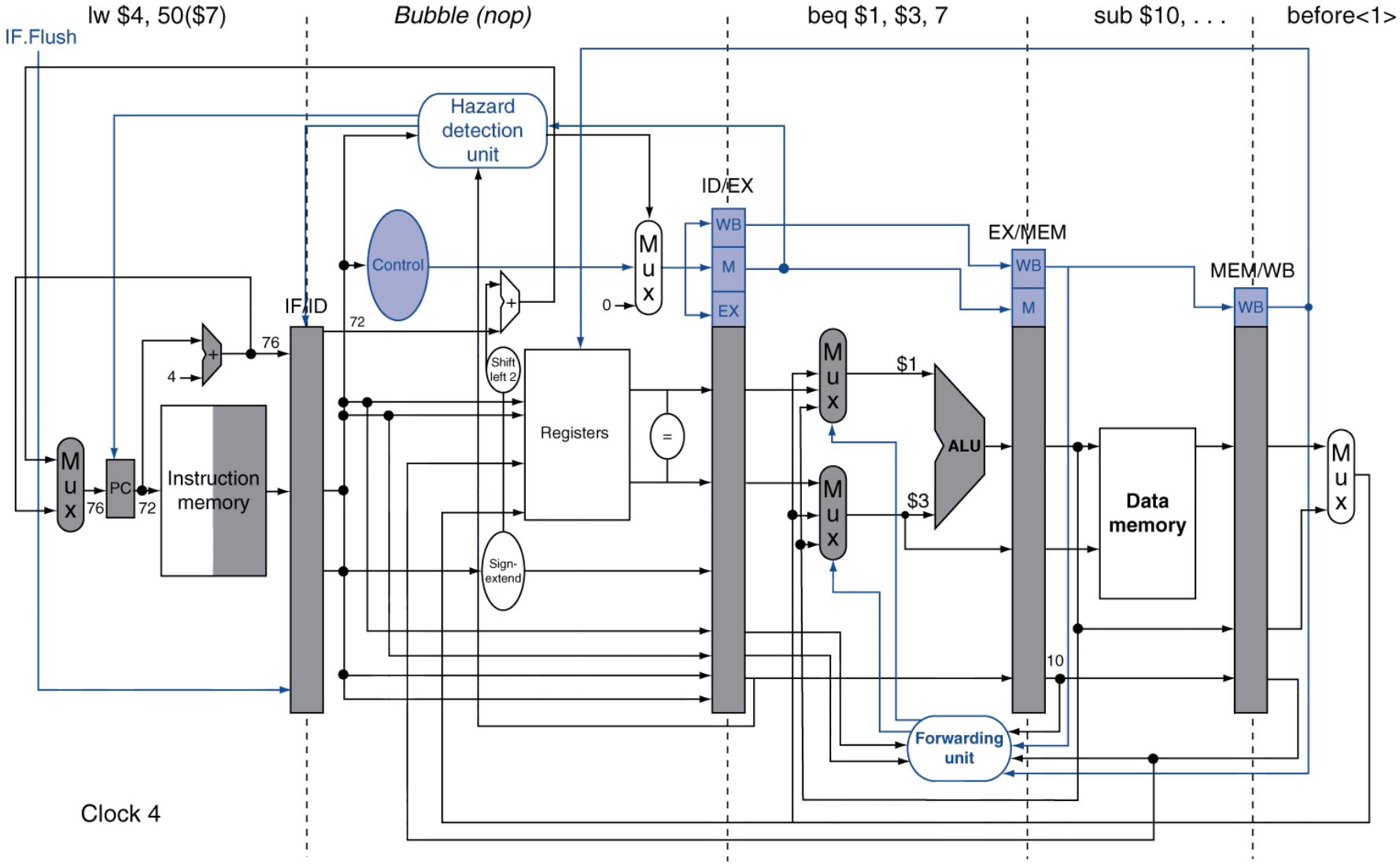
- **If branch outcome determined in MEM**



Example: Branch Taken (Clock 3)



Example: Branch Taken (Clock 4)



Dynamic Branch Prediction

Use history to make future predictions!

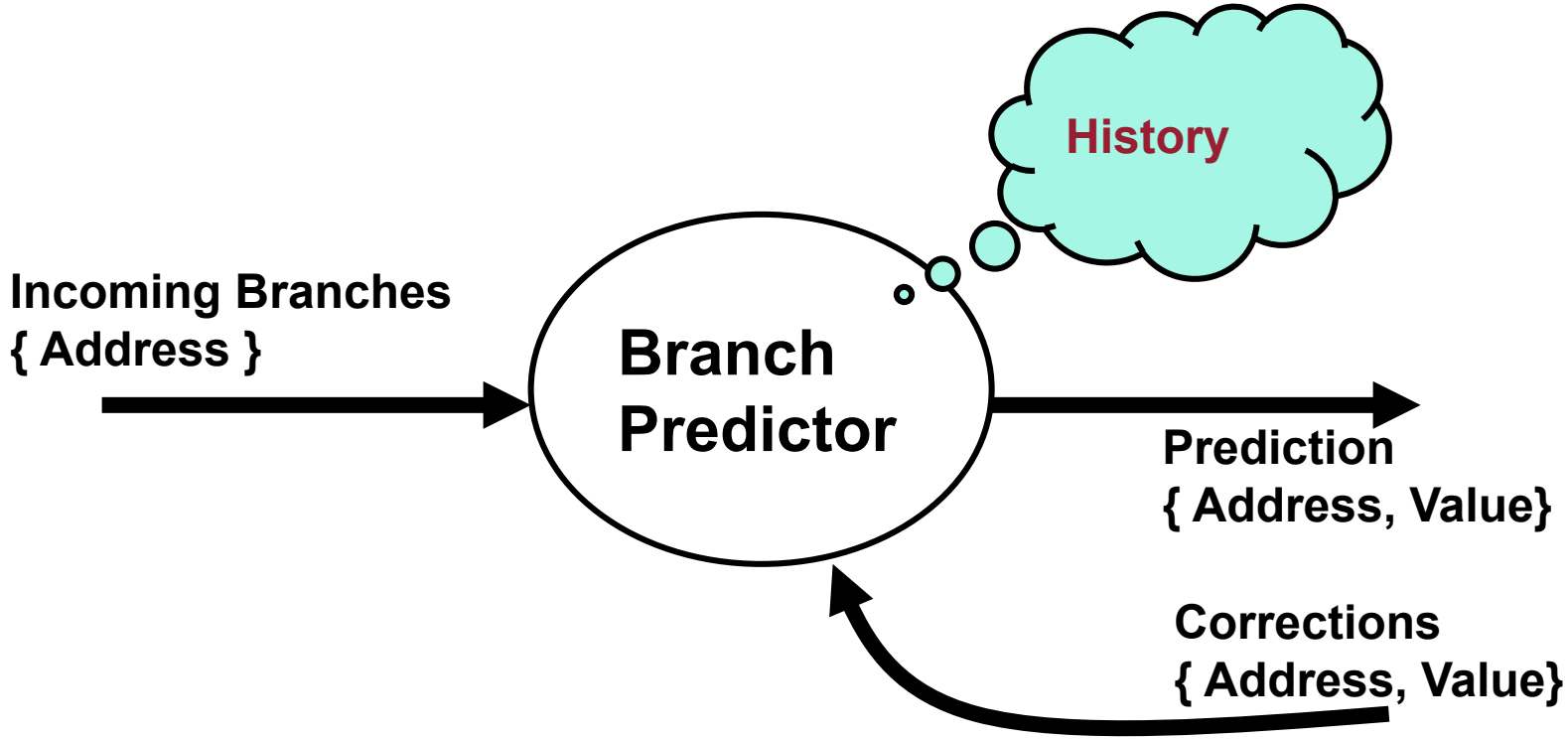
Temporal correlation

The way a branch resolves may be a good predictor of the way it will resolve at the next execution

Spatial correlation

Several branches may resolve in a highly correlated manner *(a preferred path of execution)*

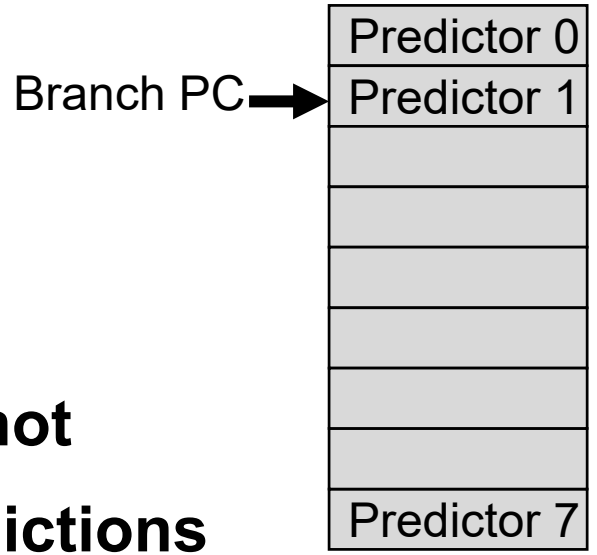
Dynamic Branch Prediction Problem



- Incoming stream of addresses
- Fast outgoing stream of predictions
- Correction information returned from pipeline

One-level Branch History Table (BHT)

- **Each branch given its own predictor state machine**
- **BHT is table of "Predictors"**
  - Could be 1-bit, could be complex state machine
- **Indexed by PC address of Branch – without tags**
  - Lower bits of the PC address are used
  - No address check (saves HW, but may not be the right address)
- **1-bit BHT keeps says whether branch was taken or not**
- **Problem: In a loop, 1-bit BHT will cause two mispredictions**
  - End of loop case: when it exits instead of looping as before
  - First time through loop on *next* time through code, when it predicts exit instead of looping
- **Solution: Use at least two bits for predictor**



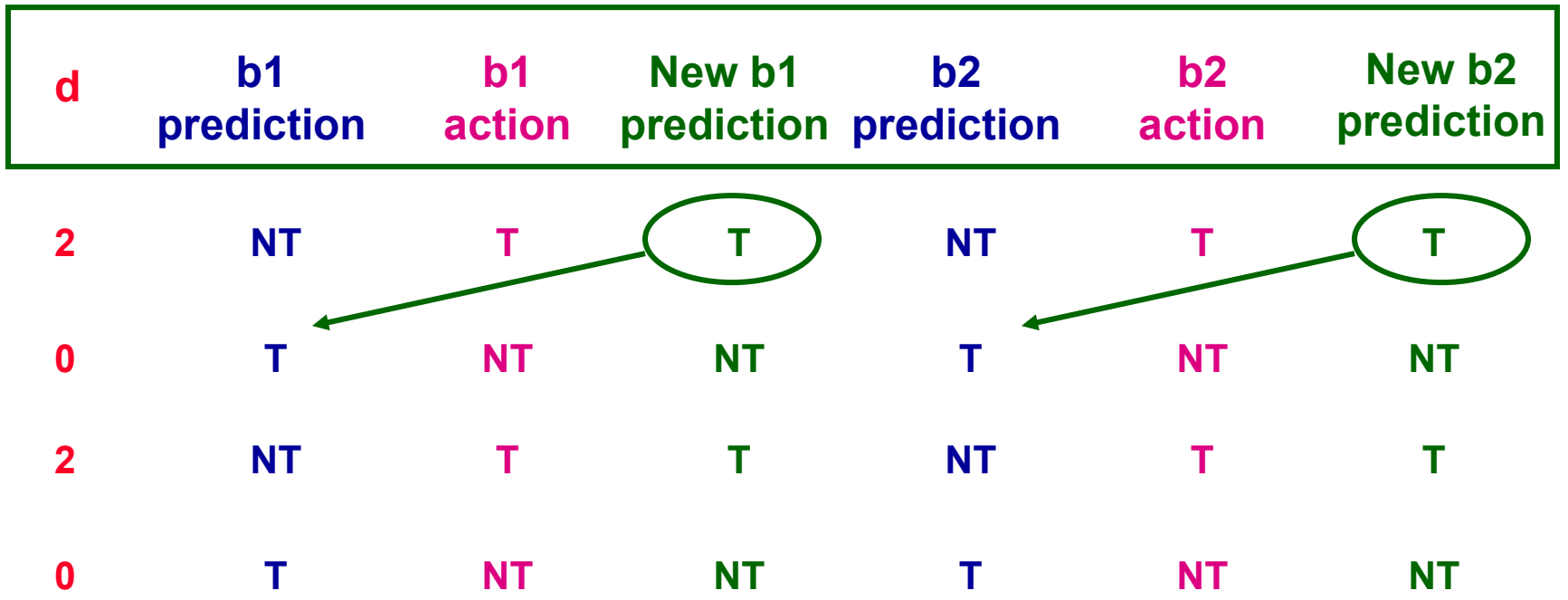
Example: Possible Sequence

```
if      ( d == 0 )      b1
                d = 1
if      ( d == 1)       b2
```

d initial value	d==0?	b1	d value before b2	d==1?	b2
0	Y	NT	1	Y	NT
1	N	T	1	Y	NT
2	N	T	2	N	T

1-bit Branch Predictor

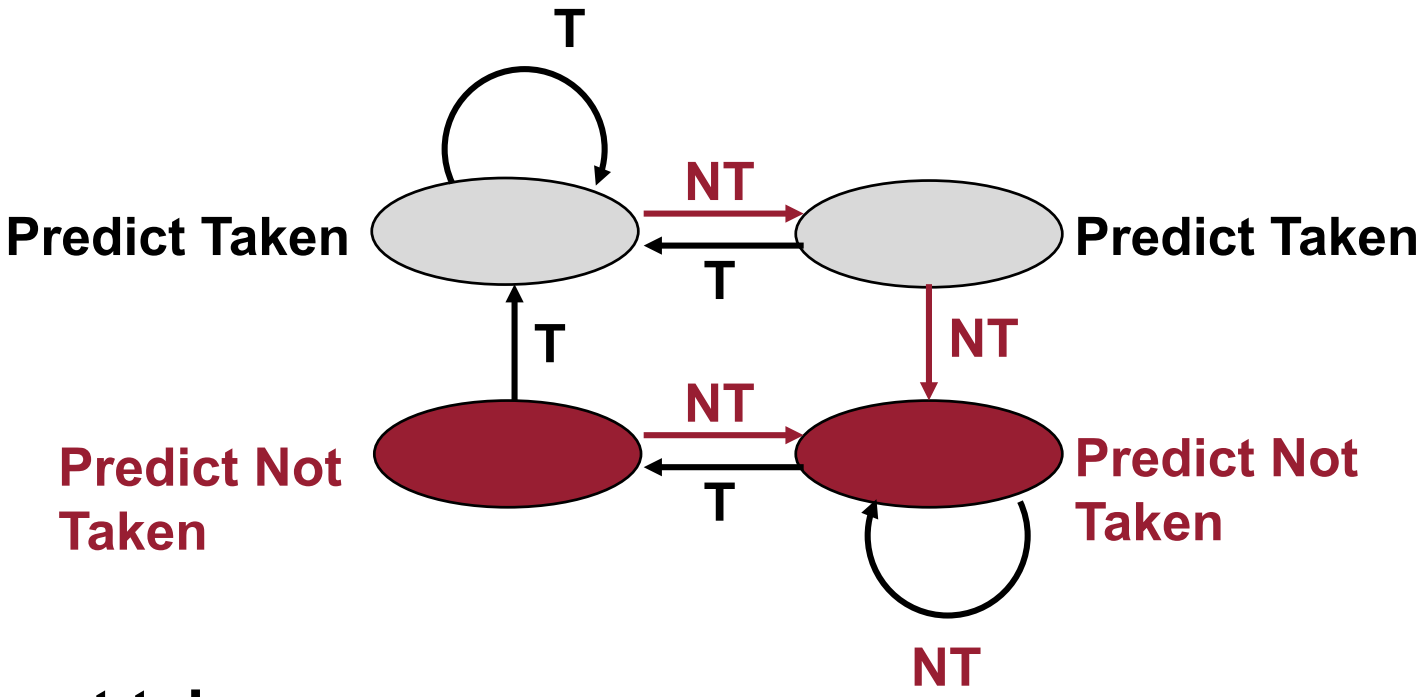
```
if      ( d == 0 )      b1
                d = 1
if      ( d == 1)       b2
```



d	b1 prediction	b1 action	New b1 prediction	b2 prediction	b2 action	New b2 prediction
2	NT	T	T	NT	T	T
0	T	NT	NT	T	NT	NT
2	NT	T	T	NT	T	T
0	T	NT	NT	T	NT	NT

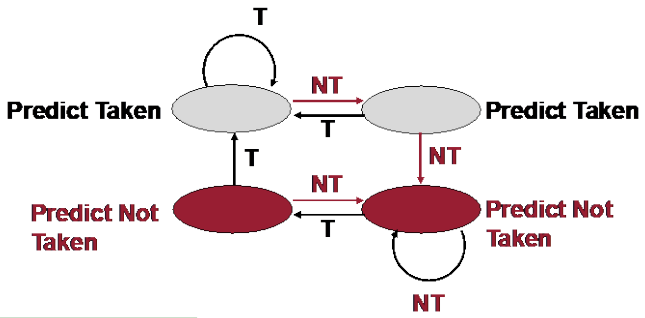
2-bit Branch Predictor

- Solution: 2-bit scheme where change prediction only if get misprediction *twice:*



- Red: stop, not taken
- Grey: go, taken
- Adds *hysteresis* to decision making process

2-Bit Branch Predictor



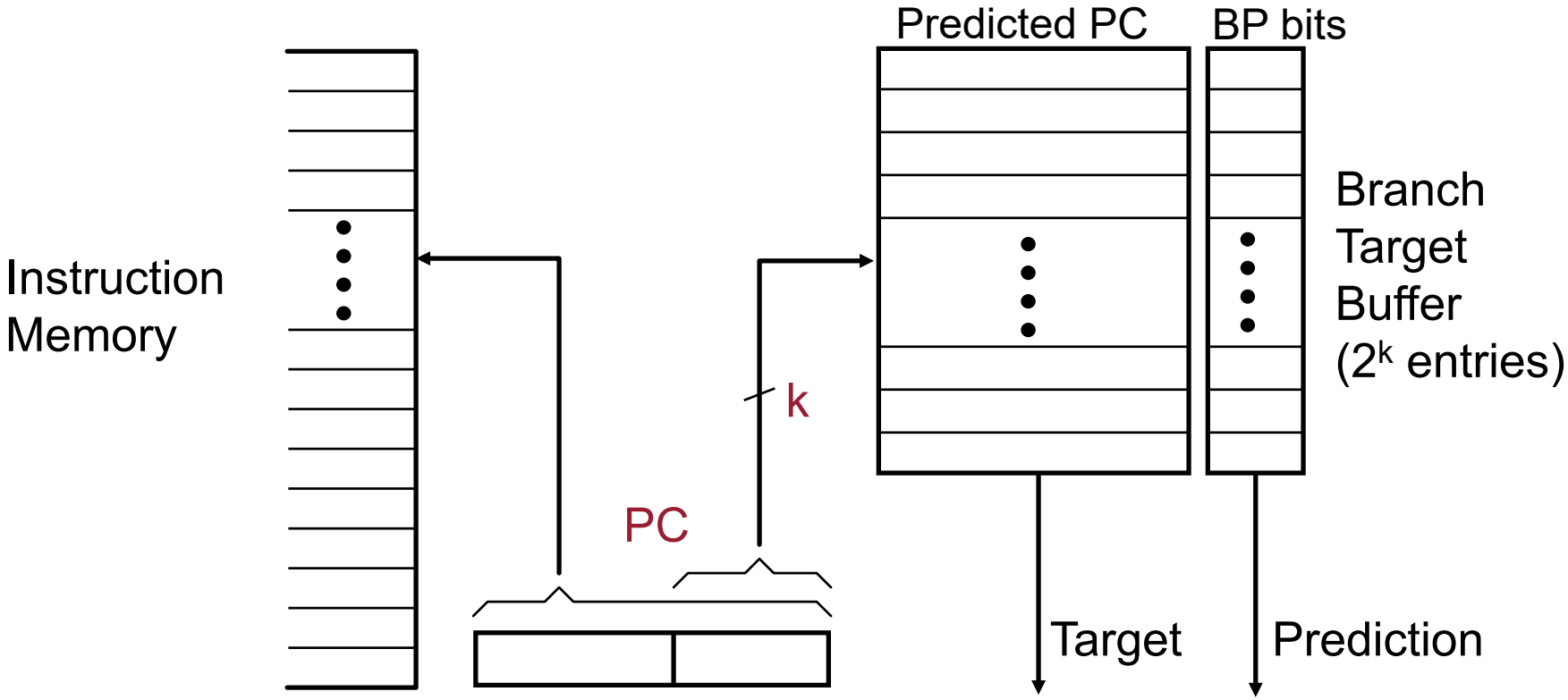
d	b1 prediction	b1 action	New b1 prediction	b2 prediction	b2 action	New b2 prediction
2	NT/NT	T	T/NT	NT/NT	T	NT/T
0	T/NT	NT	T/NT	NT/T	NT	NT/T
2	T/NT	T	T/NT	NT/T	T	NT/T
0	T/NT	NT	T/NT	NT/T	NT	NT/T

```
if      ( d == 0 )      b1
                d = 1
if      ( d == 1)       b2
```

Dynamic Branch Prediction Summary

- **Prediction is critical to minimize branch stalls**
- **Branch History Table: 2 bits for loop accuracy**
- **However, branch prediction only predicts the direction**
  - Still need to figure out the address for taken branches

Branch Target Buffer



BP bits are stored with the predicted target address.

IF stage: *If (BP=taken) then nPC=target else nPC=PC+4*
later: *check prediction, if wrong then kill the instruction and update BTB & BPb else update BPb*