CASPER

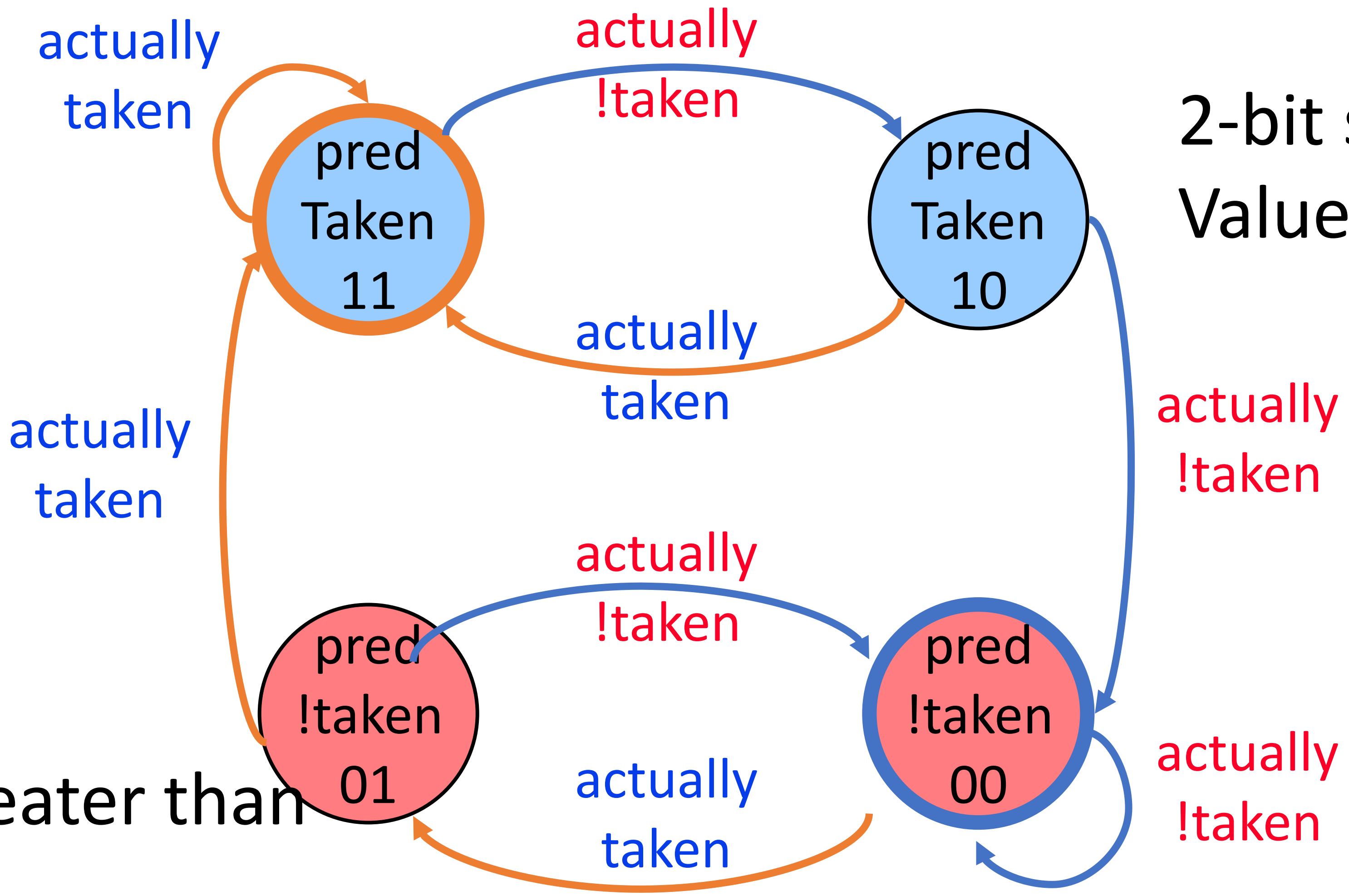
CS305: Computer Architecture

Branch Prediction-II

https://www.cse.iitb.ac.in/~biswa/courses/CS305/main.html

https://www.cse.iitb.ac.in/~biswa/

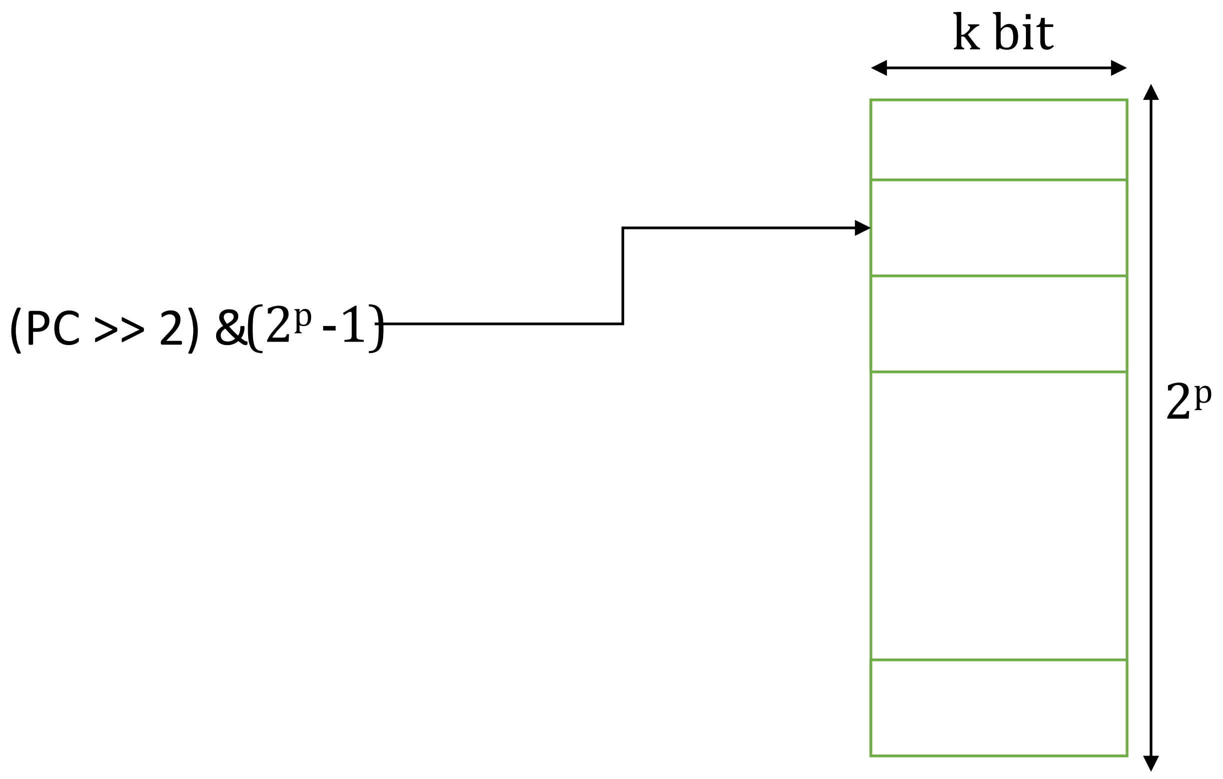
Recap: 2-bit Bimodal Predictors: A bit better



2-bit saturating counter:
Values from 00 to 11

Counter greater than equal to 10,
prediction: taken

No history predictor: 2 bit predictor



Bimodal predictor: Good for biased branches

Is this enough?

- Control flow instructions (branches) are frequent
  - 20% of all instructions
- Problem: Next fetch address after a control-flow instruction is not determined after N cycles in a pipelined processor
  - N cycles: (minimum) branch resolution latency
- How do we keep the pipeline full after a branch?

Let’s understand the utility

Branch prediction accuracy 98% to 99%, Assume a pipeline with 20-cycle branch resolution latency

Is it a big deal? It is 2% misprediction rate → 1%

- That’s a halving of the number of mispredictions

  Example: one billion branches
  99% accuracy: 10000000 mis-predictions ☹ X 20 cycles
  98% accuracy: 20000000 mis-predictions ☹ ☹ X 20 cycles

Local and global history

- Local Behavior

  What is the predicted direction of Branch A given the outcomes of previous instances of Branch A ?

- Global Behavior

  What is the predicted direction of Branch Z given the outcomes of *all** previous branches A, B, …, X and Y?

  Number of previous branches tracked limited by the history length

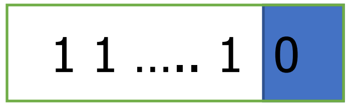
Two Level Branch Predictors

First level: Global branch history register (N bits)

The direction of last N branches

Second level: Table of saturating counters for each history entry

The direction the branch took the last time the same history was seen



GHR
(global history register)

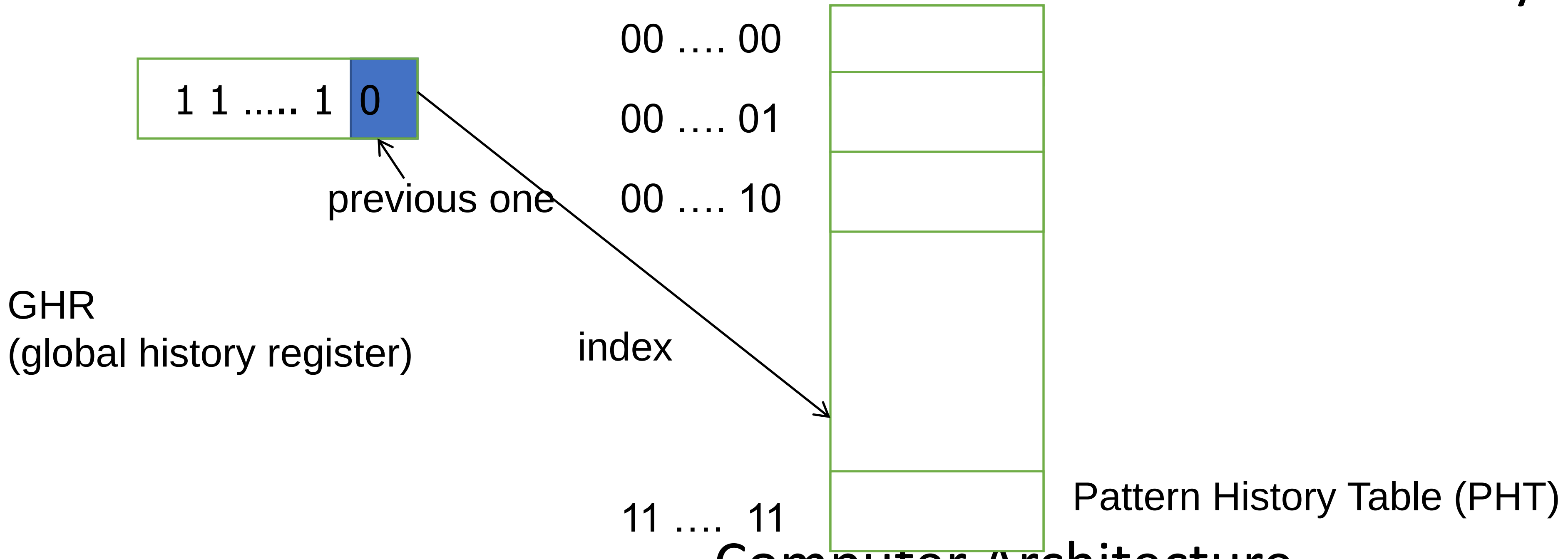
Two Level Branch Predictors

First level: Global branch history register (N bits)

The direction of last N branches

Second level: Table of saturating counters for each history entry

The direction the branch took the last time the same history was seen



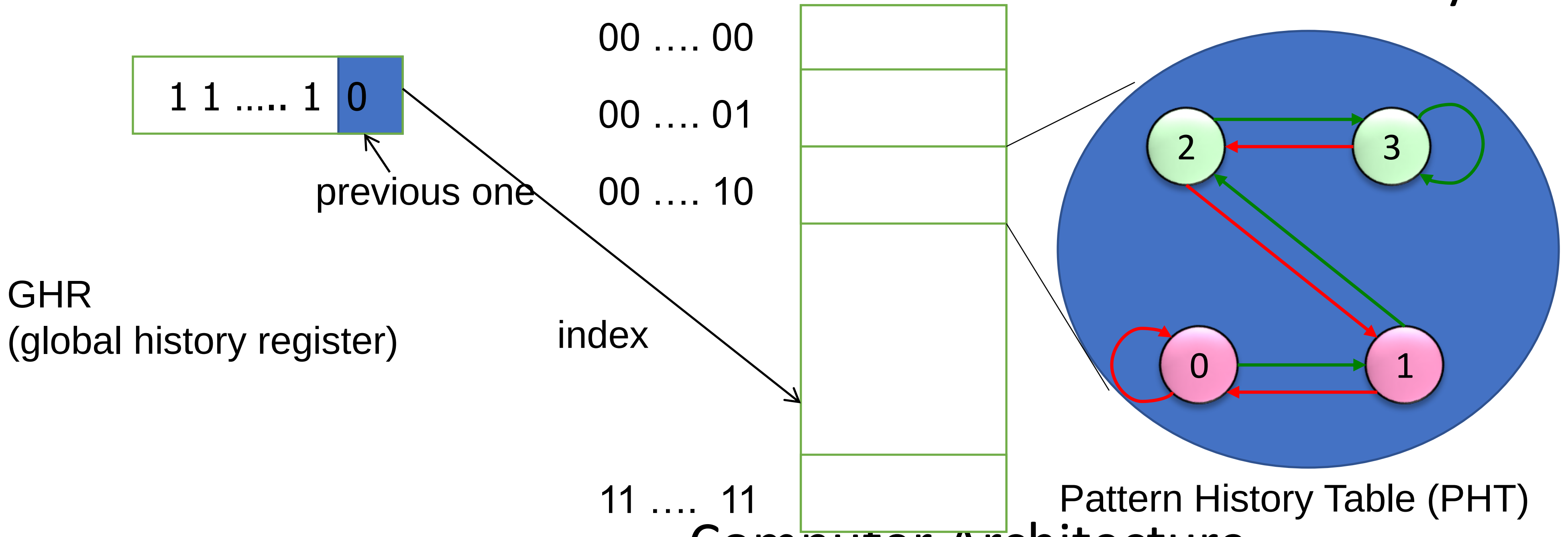
Two Level Branch Predictors

First level: Global branch history register (N bits)

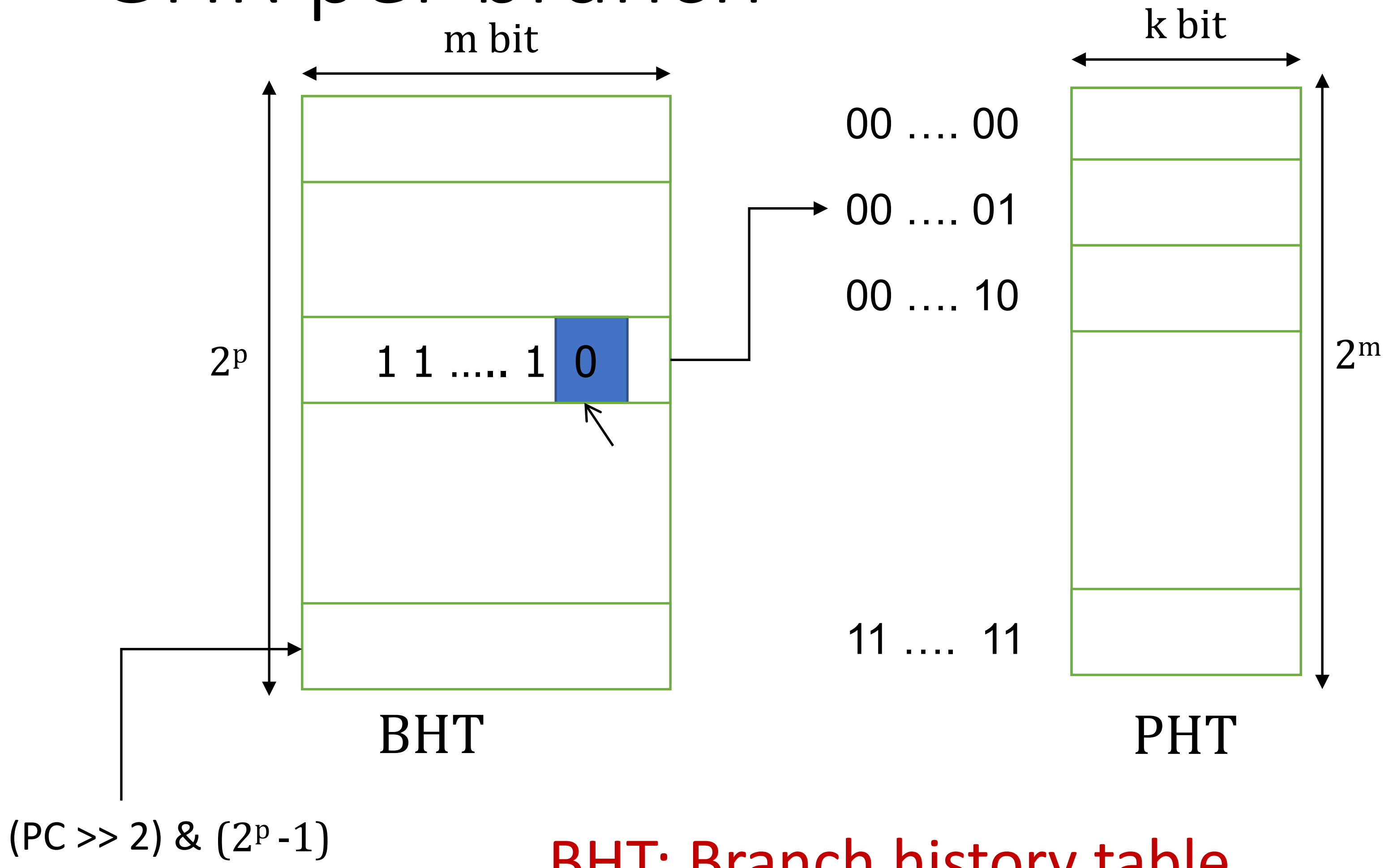
The direction of last N branches

Second level: Table of saturating counters for each history entry

The direction the branch took the last time the same history was seen



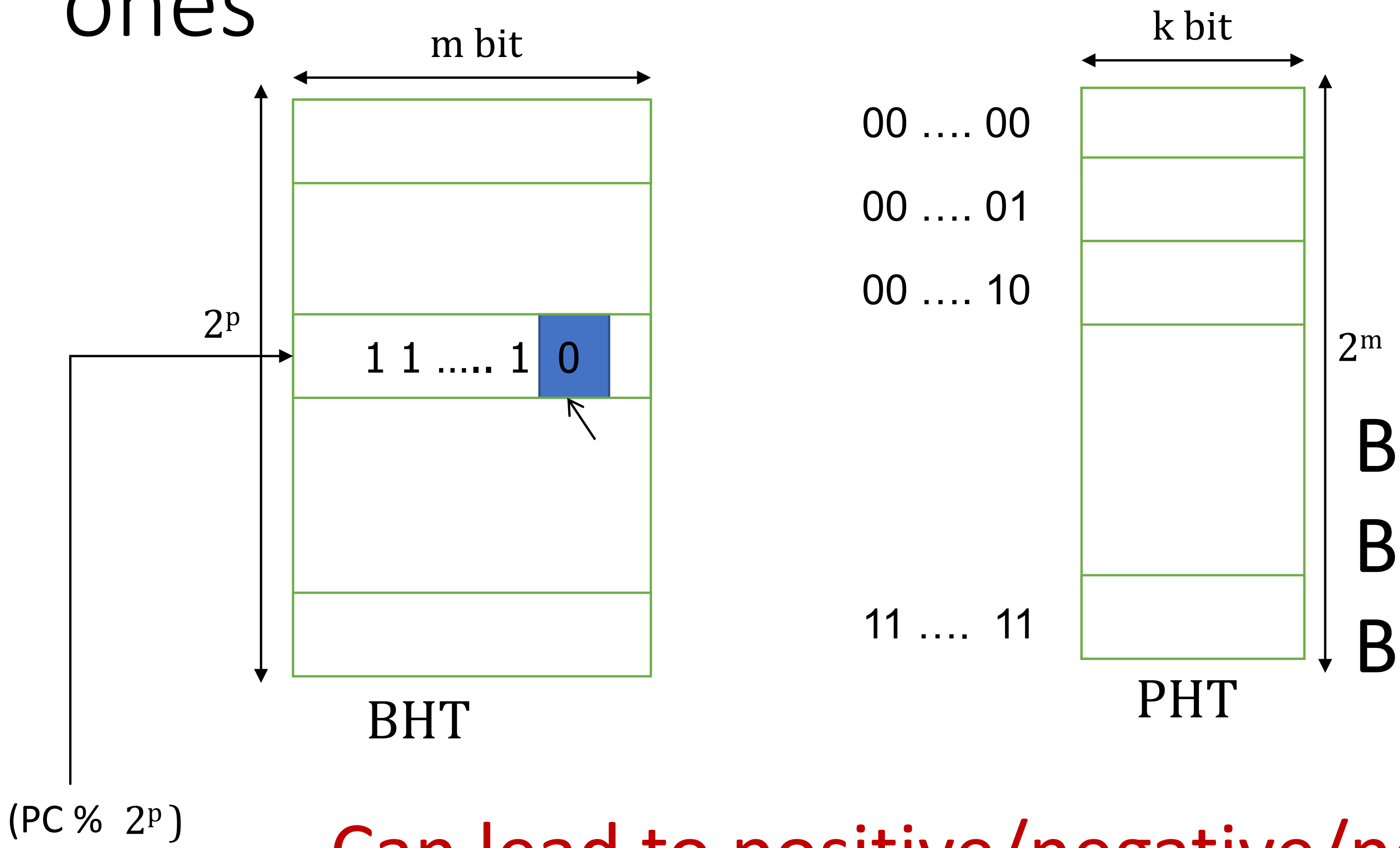
GHR per branch



BHT: Branch history table

Mostly K=2, m =12 for example

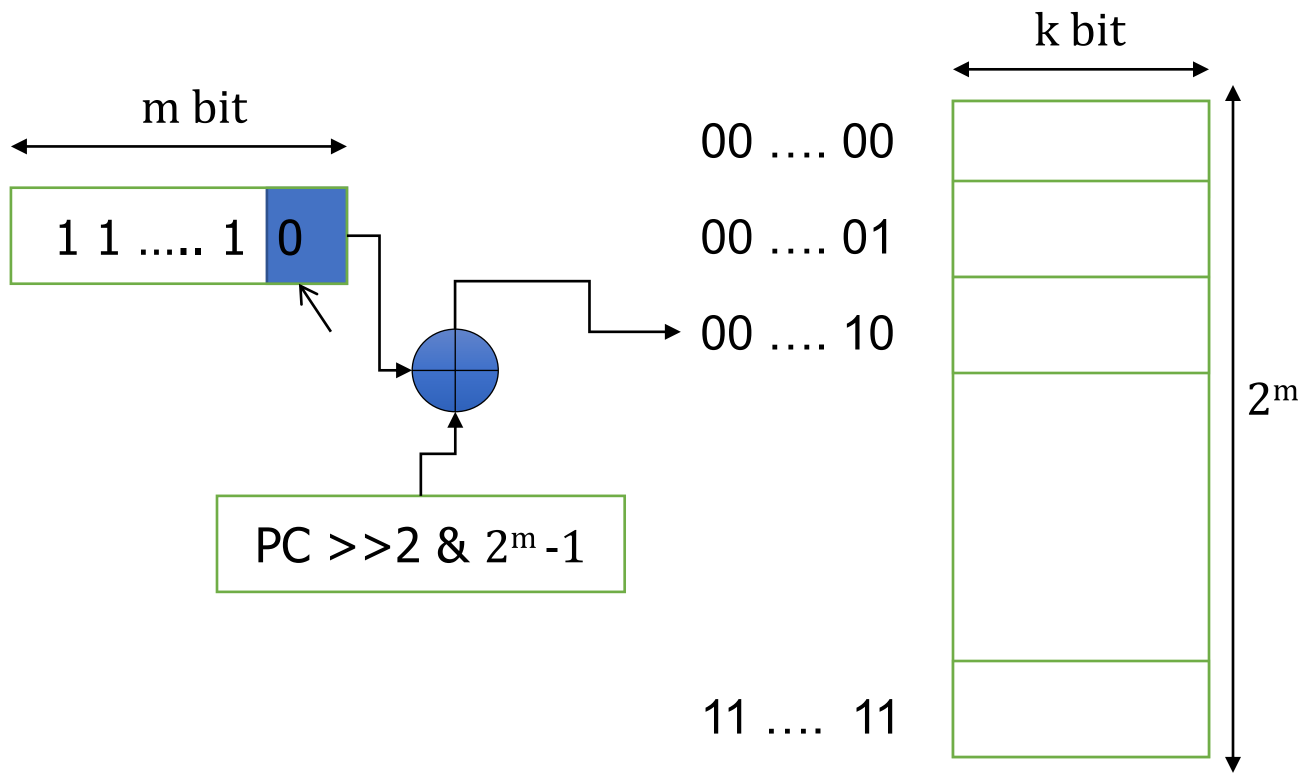
Set of branches: One register for correlated ones



B1: if (aa == 2) aa = 0;
B2: if (bb == 2); bb=0;
B3: if (aa != bb) { … }

Can lead to positive/negative/neutral interference

Gshare is the answer



For a given history and for a given branch (PC) counters are trained

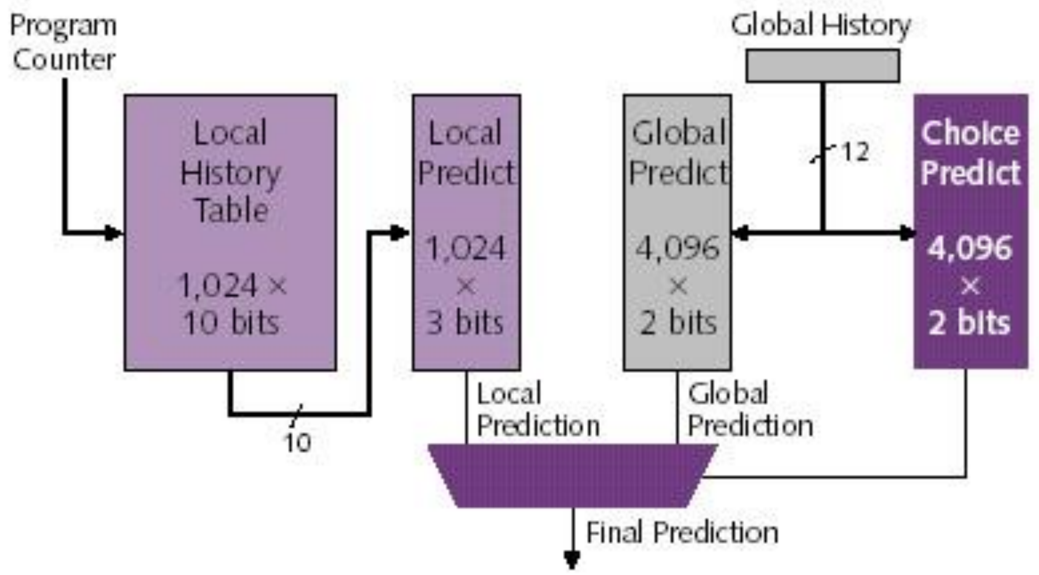
Few Important Points

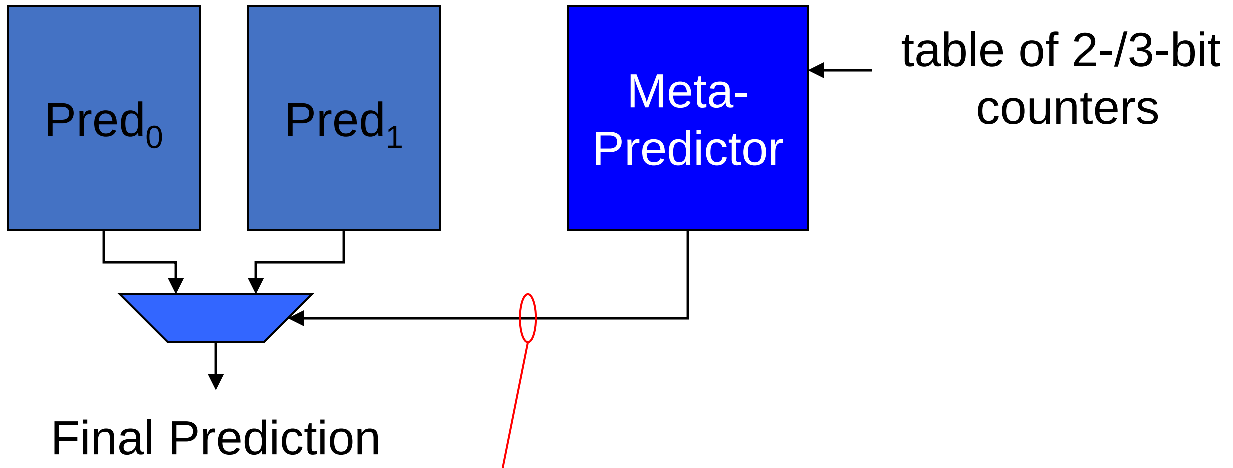
Branch prediction happens at the IF stage.

We know the target outcome at the end of EX stage.

So BHT and PHT will be updated after EX stage for the corresponding PC. Any issues here?

Tournament Predictor

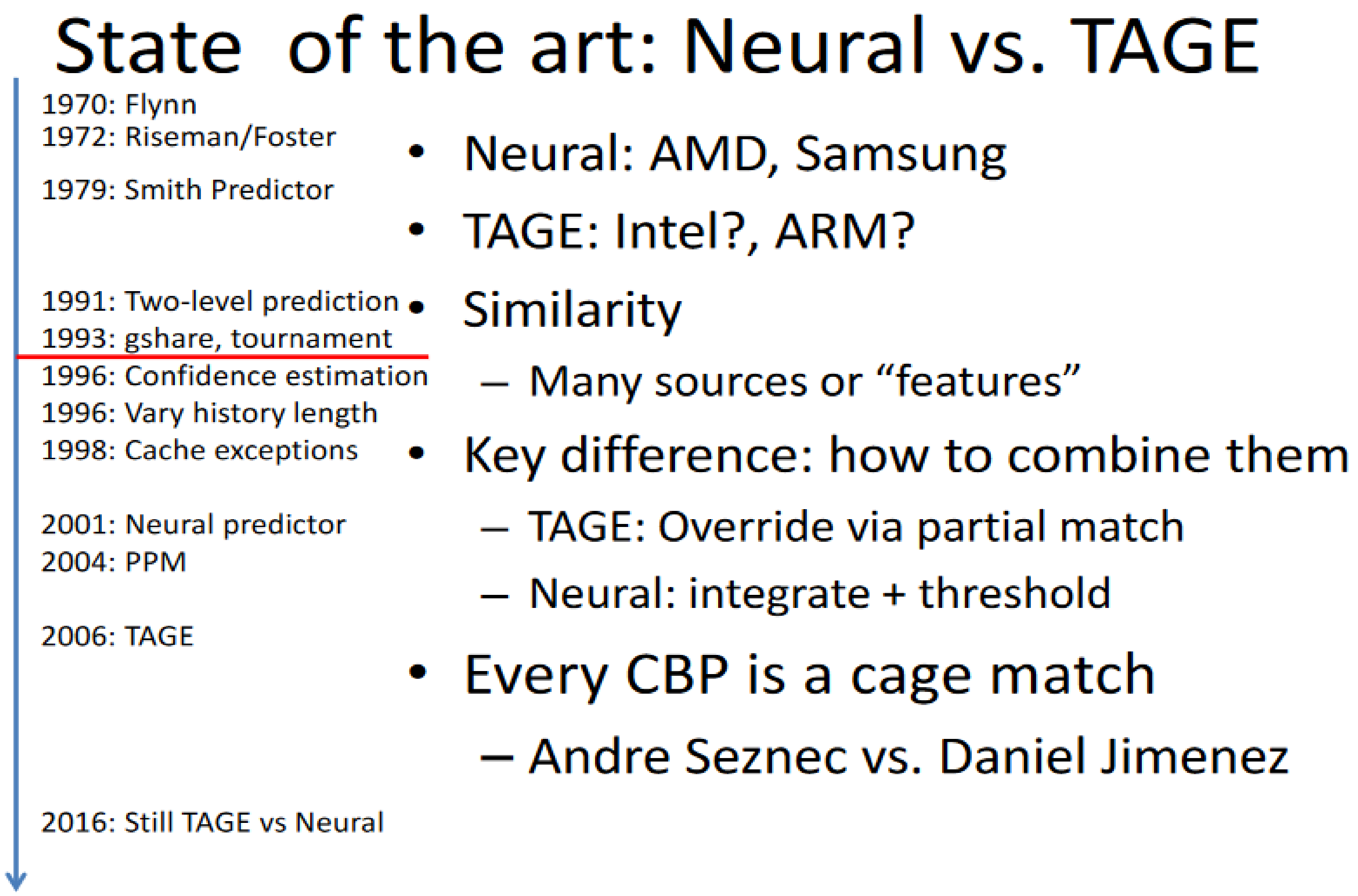

Program Counter
Local History Table 1,024 × 10 bits
10
Local Predict 1,024 × 3 bits
Global History
12
Global Predict 4,096 × 2 bits
Choice Predict 4,096 × 2 bits
Local Prediction
Global Prediction
Final Prediction



Pred$_0$	Pred$_1$	Meta Update
✗	✗	---
✗	✓	Inc
✓	✗	Dec
✓	✓	---

If meta-counter MSB = 0, use pred$_0$ else use pred$_1$

State-of-the-art



State of the art: Neural vs. TAGE

1970: Flynn
1972: Riseman/Foster
1979: Smith Predictor
1991: Two-level prediction
1993: gshare, tournament
1996: Confidence estimation
1996: Vary history length
1998: Cache exceptions
2001: Neural predictor
2004: PPM
2006: TAGE
2016: Still TAGE vs Neural

- Neural: AMD, Samsung
- TAGE: Intel?, ARM?
- Similarity
  - Many sources or "features"
- Key difference: how to combine them
  - TAGE: Override via partial match
  - Neural: integrate + threshold
- Every CBP is a cage match
  - Andre Seznec vs. Daniel Jimenez

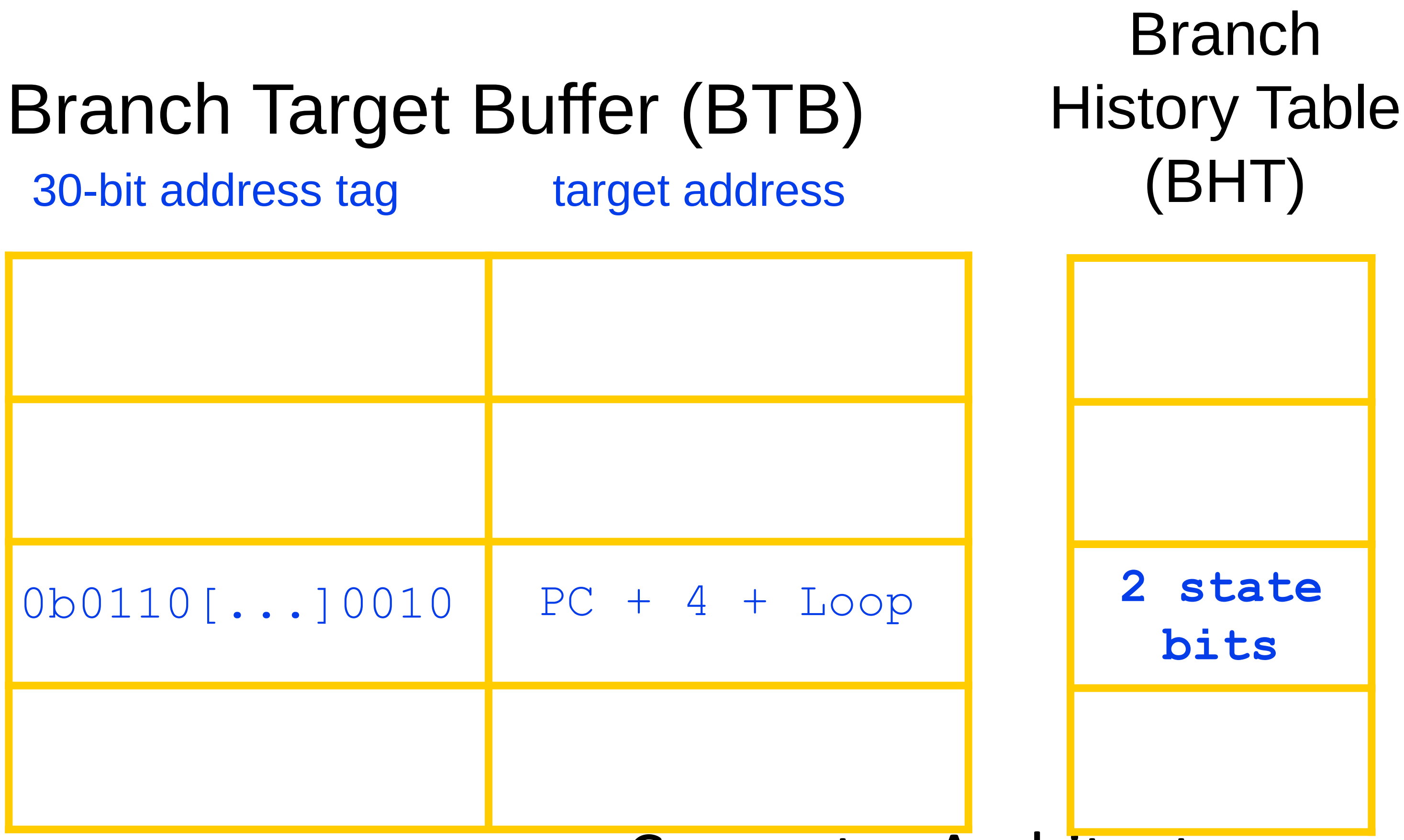
BTB (Target Address Predictor)

Address of branch instruction

`0b0110[...]01001000`

Branch instruction

`BNEZ R1 Loop`



BTB is probed in the fetch stage along with the direction predictor.

A hit in the BTB means the PC is a branch PC.

Grazie