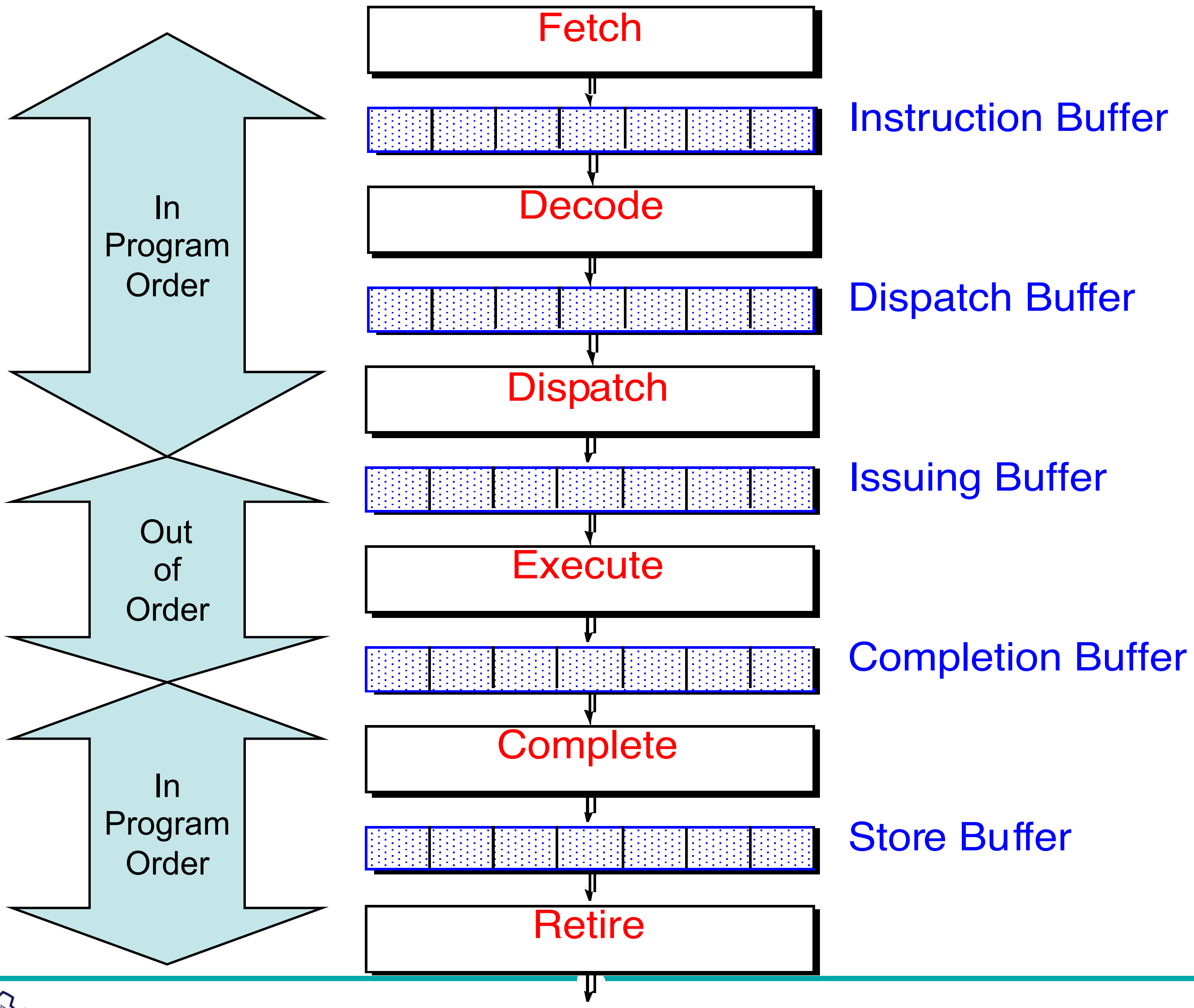
Superscalar Design

Instruction Flow

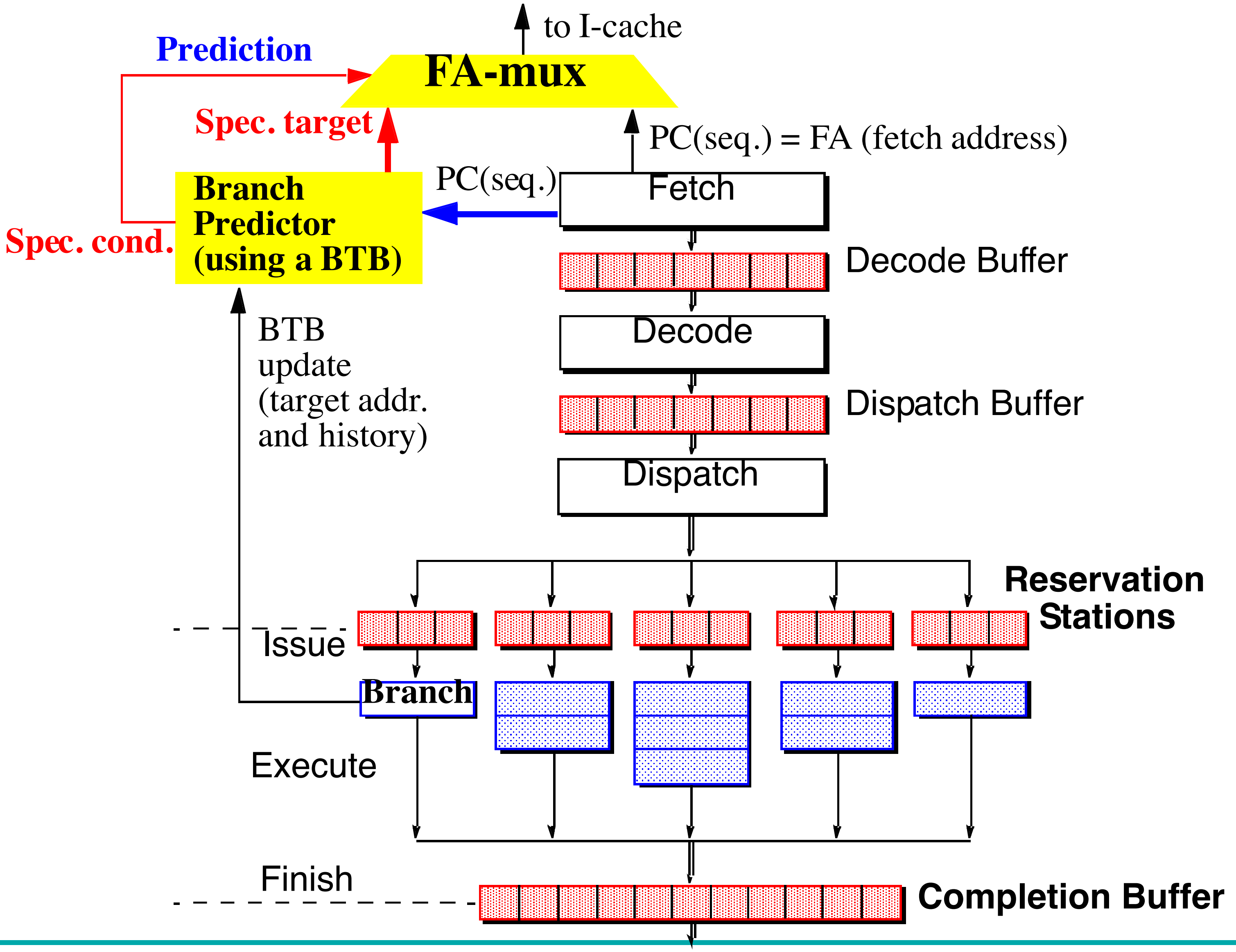
Virendra Singh
Associate Professor
Computer Architecture and Dependable Systems Lab
Department of Electrical Engineering
Indian Institute of Technology Bombay
http://www.ee.iitb.ac.in/~viren/
E-mail: viren@ee.iitb.ac.in

EE-739: Processor Design

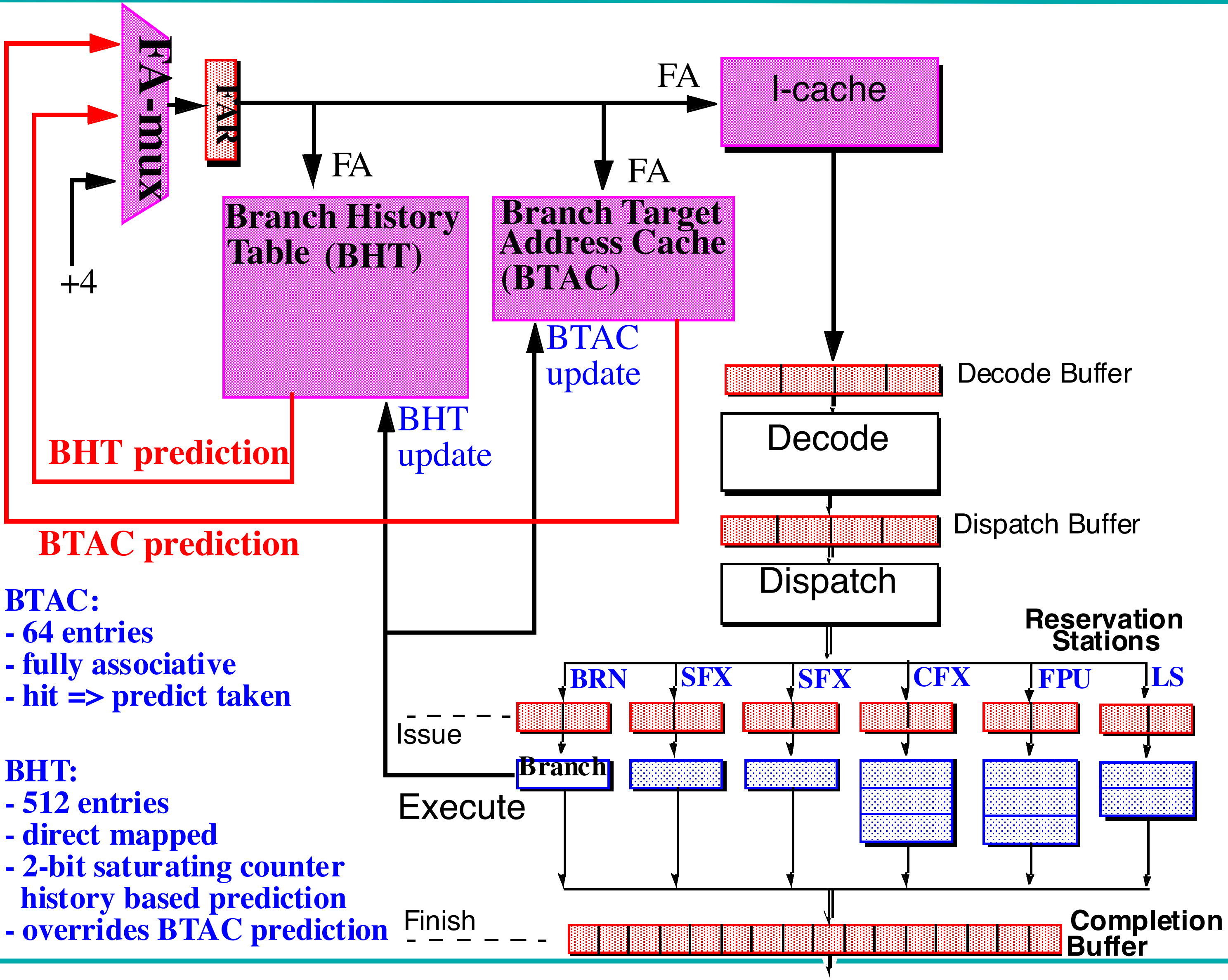
Superscalar Pipeline Stages



Branch Instruction Speculation



BTAC and BHT Design (PPC 604)



BTAC:
- **64 entries**
- **fully associative**
- **hit => predict taken**

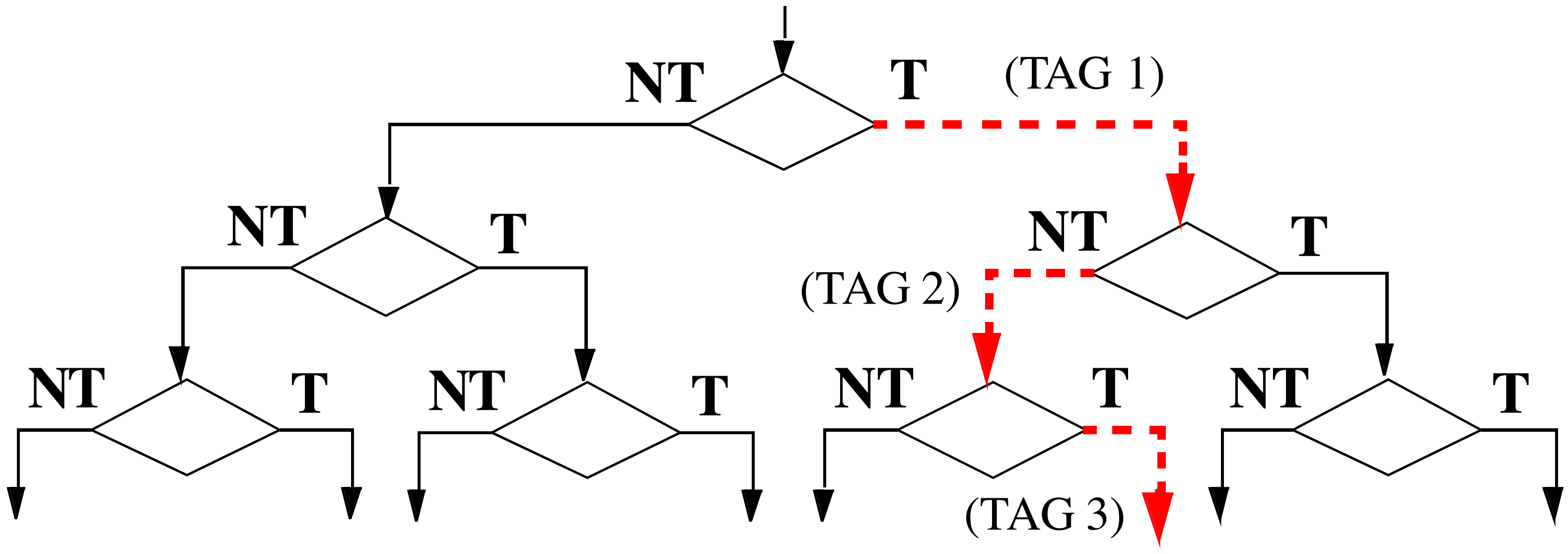
BHT:
- **512 entries**
- **direct mapped**
- **2-bit saturating counter history based prediction**
- **overrides BTAC prediction**

Branch/Jump Target Prediction

Branch inst. address	Information for predict.	**Branch target address** (most recent)
0x0348	0101 (NTNT)	0x0612

- Branch Target Buffer: small cache in fetch stage
  - Previously executed branches, address, taken history, target(s)
- Fetch stage compares current FA against BTB
  - If match, use prediction
  - If predict taken, use BTB target
- When branch executes, BTB is updated
- Optimization:
  - Size of BTB: increases hit rate
  - Prediction algorithm: increase accuracy of prediction

Branch Speculation

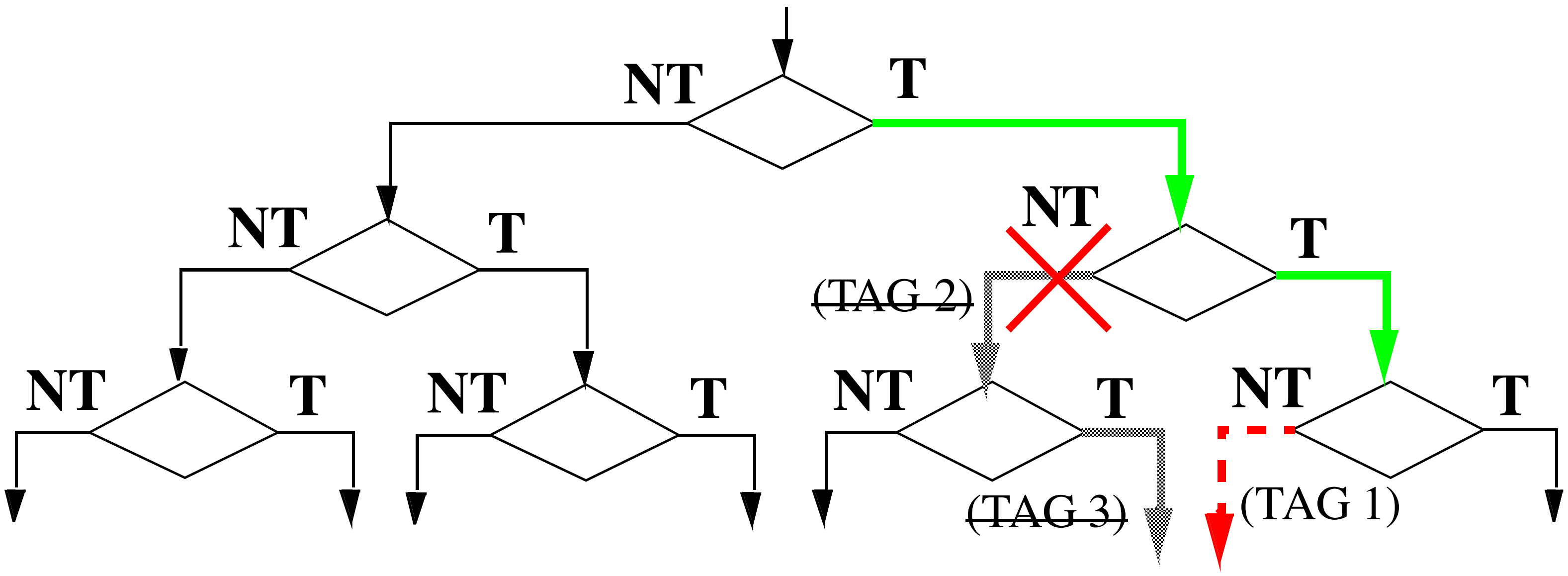


- Leading Speculation
  - Typically done during the Fetch stage
  - Based on potential branch instruction(s) in the current fetch group
- Trailing Confirmation
  - Typically done during the Branch Execute stage
  - Based on the next Branch instruction to finish execution

Branch Speculation

- <u>Leading Speculation</u>
  1. Tag speculative instructions
  2. Advance branch and following instructions
  3. Buffer addresses of speculated branch instructions
- <u>Trailing Confirmation</u>
  1. When branch resolves, remove/deallocate speculation tag
  2. Permit completion of branch and following instructions

Branch Speculation



- Start new correct path
  - Must remember the alternate (non-predicted) path
- Eliminate incorrect path
  - Must ensure that the mis-speculated instructions produce no side effects

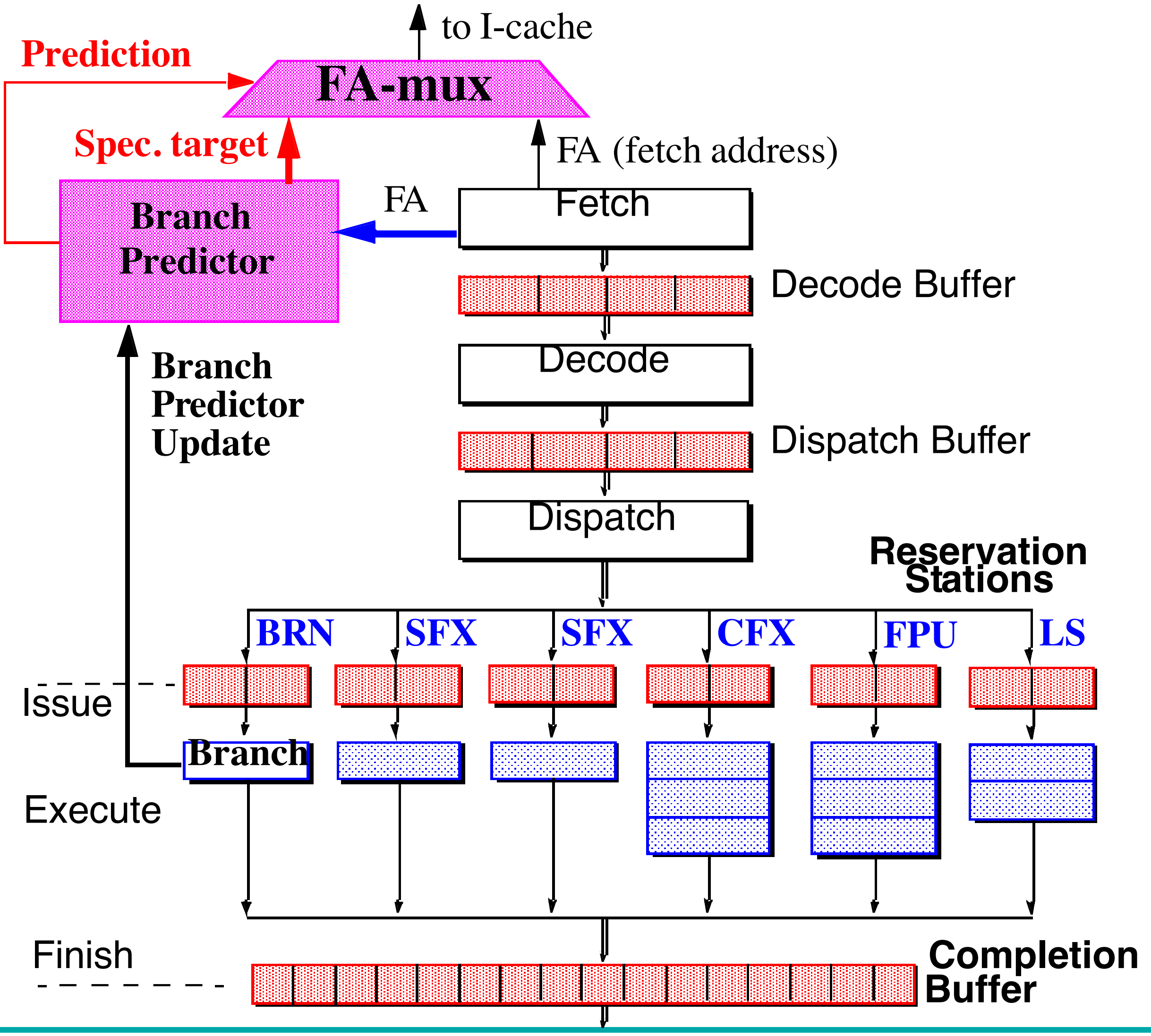
Mis-speculation Recovery

- Start new correct path
  1. Update PC with computed branch target (if predicted NT)
  2. Update PC with sequential instruction address (if predicted T)
  3. Can begin speculation again at next branch
- Eliminate incorrect path
  1. Use tag(s) to deallocate ROB entries occupied by speculative instructions
  2. Invalidate all instructions in the decode and dispatch buffers, as well as those in reservation stations

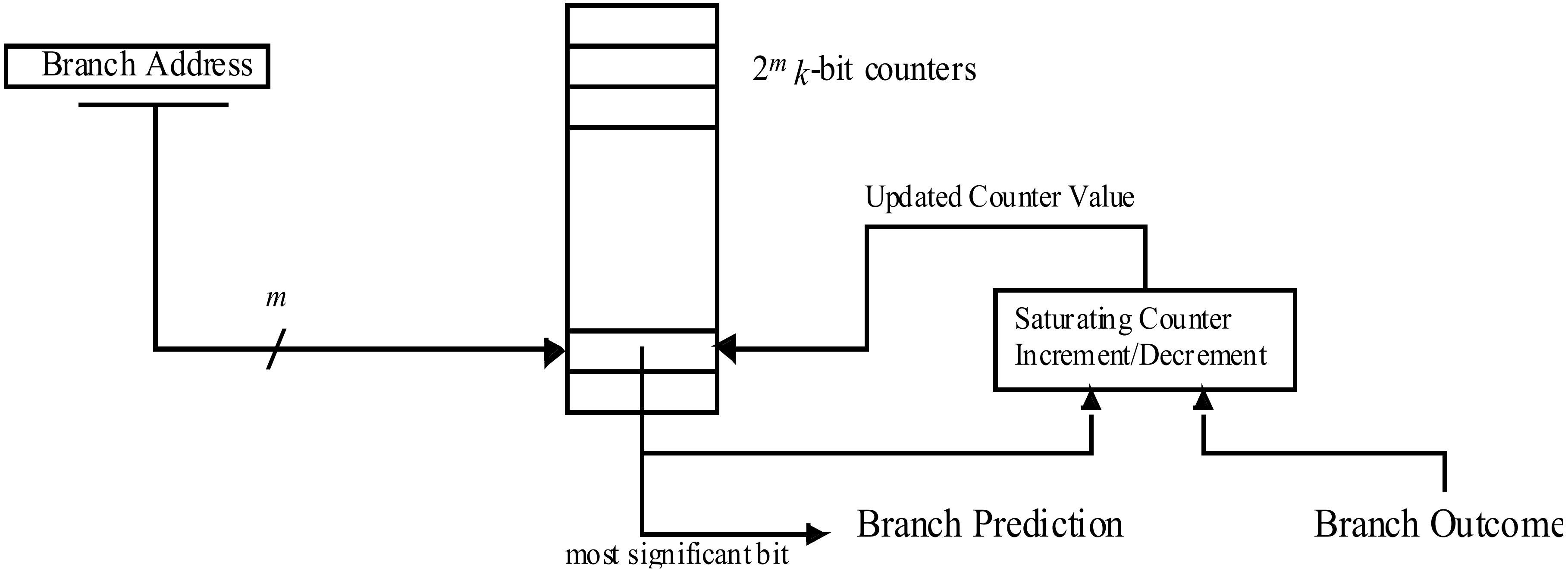
Tracking Instructions

- Assign branch tags
  - Allocated in circular order
  - Instruction carries this tag throughout processor
- Track instruction groups
  - Instructions managed in groups, max. one branch per group
  - ROB structured as groups
    - Leads to some inefficiency
    - Simpler tracking of speculative instructions

Program Control Flow

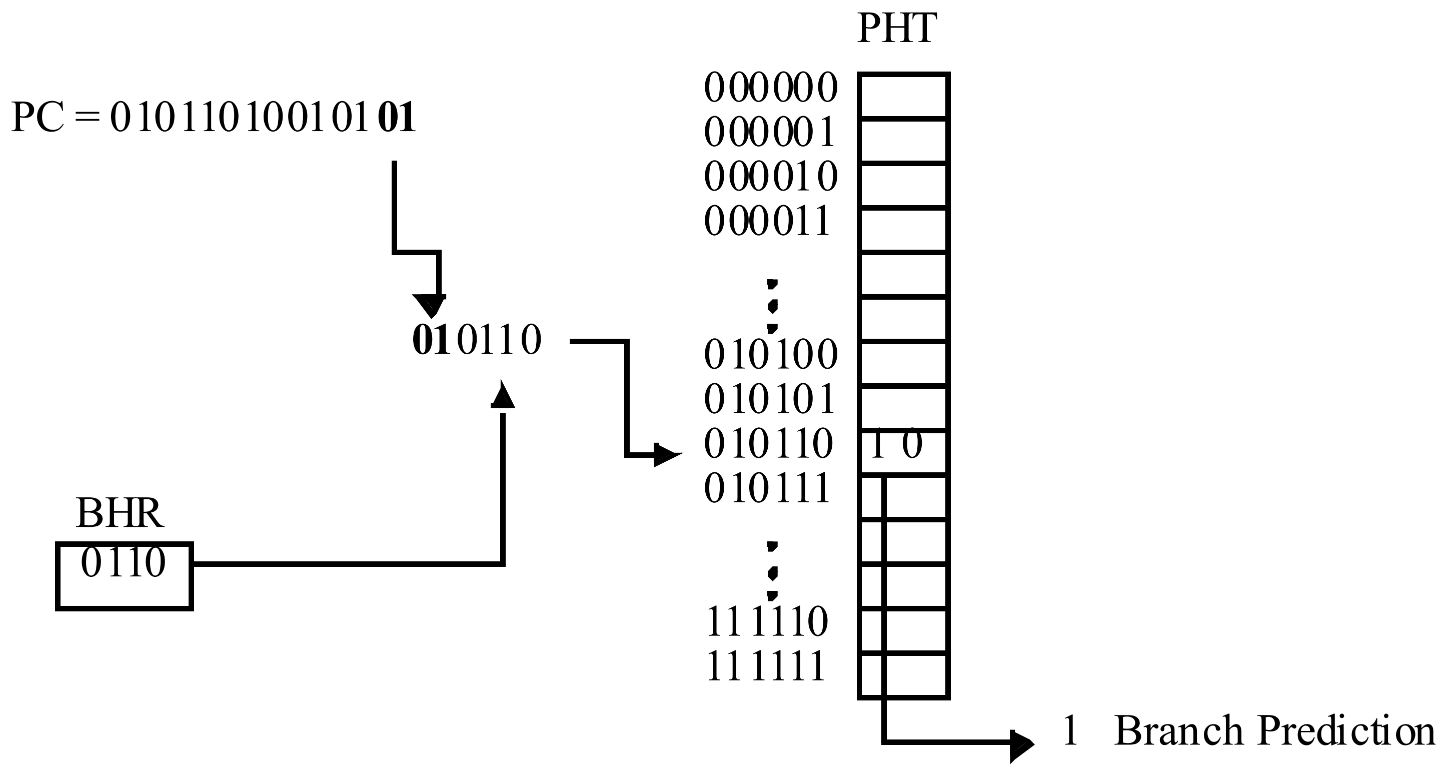


Smith Predictor Hardware



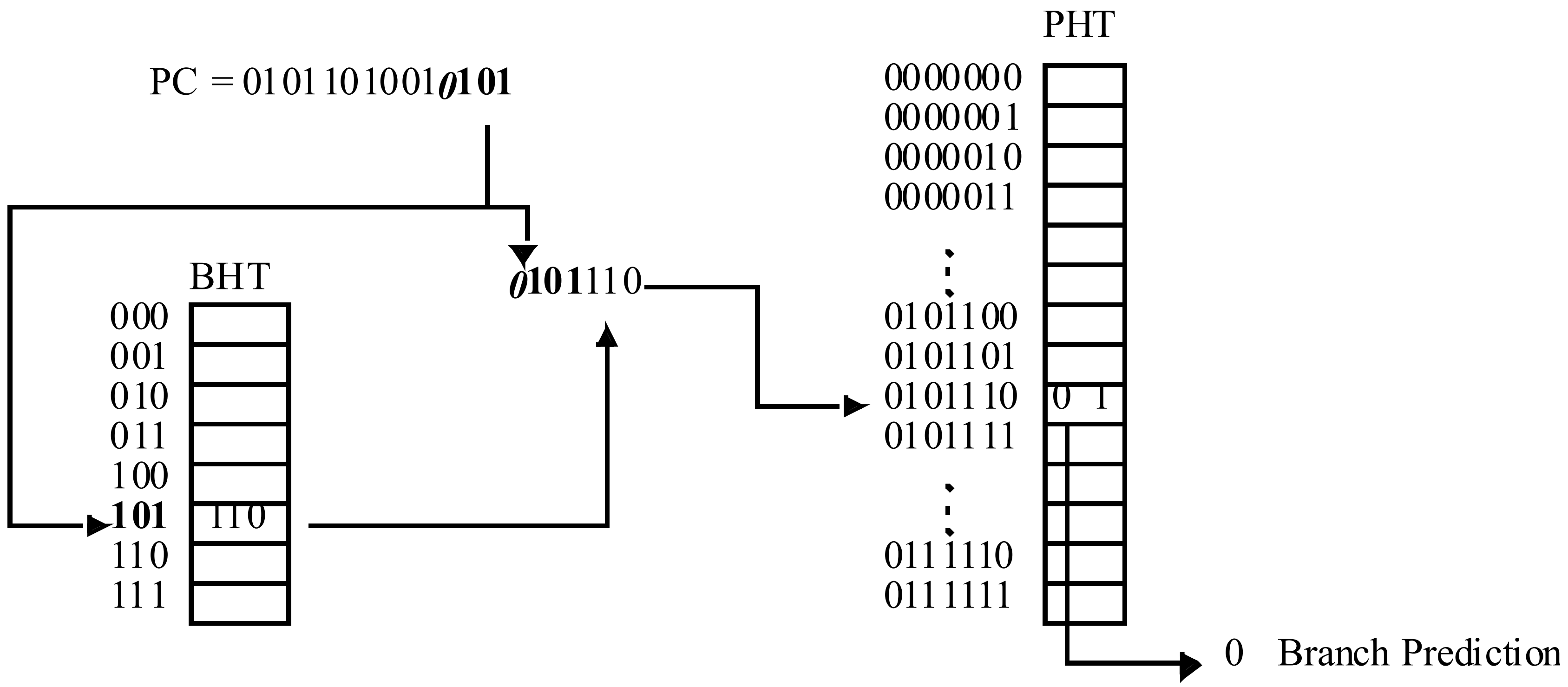
- Jim E. Smith. A Study of Branch Prediction Strategies. International Symposium on Computer Architecture, pages 135-148, May 1981
- Widely employed: Intel Pentium, PowerPC 604, PowerPC 620, etc.

Two-level Branch Prediction



- BHR adds *global* branch history
  - Provides more context
  - Can differentiate multiple instances of the same static branch
  - Can correlate behavior across multiple static branches

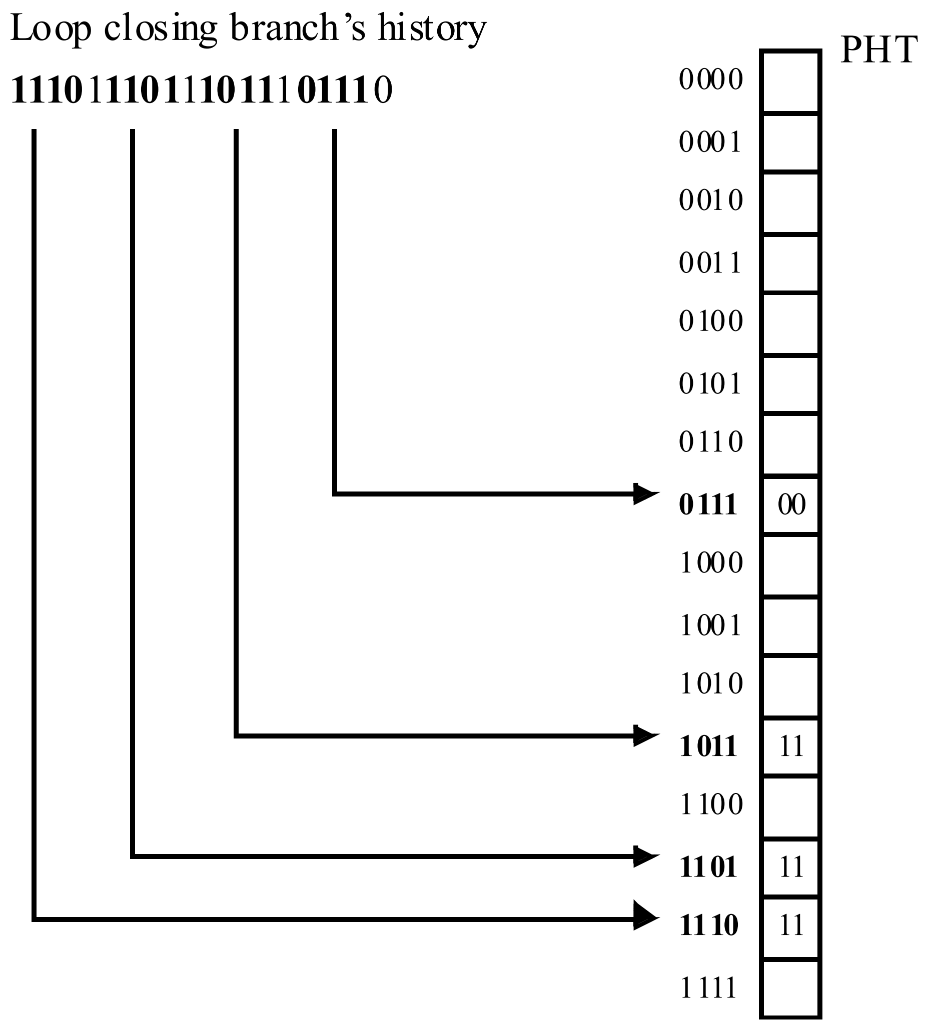
Two-level Prediction: Local History



- Detailed local history can be useful

Local History Predictor Example

- Loop closing branches
  - Must identify last instance
- Local history dedicates PHT entry to each instance
  - '0111' entry predicts not taken

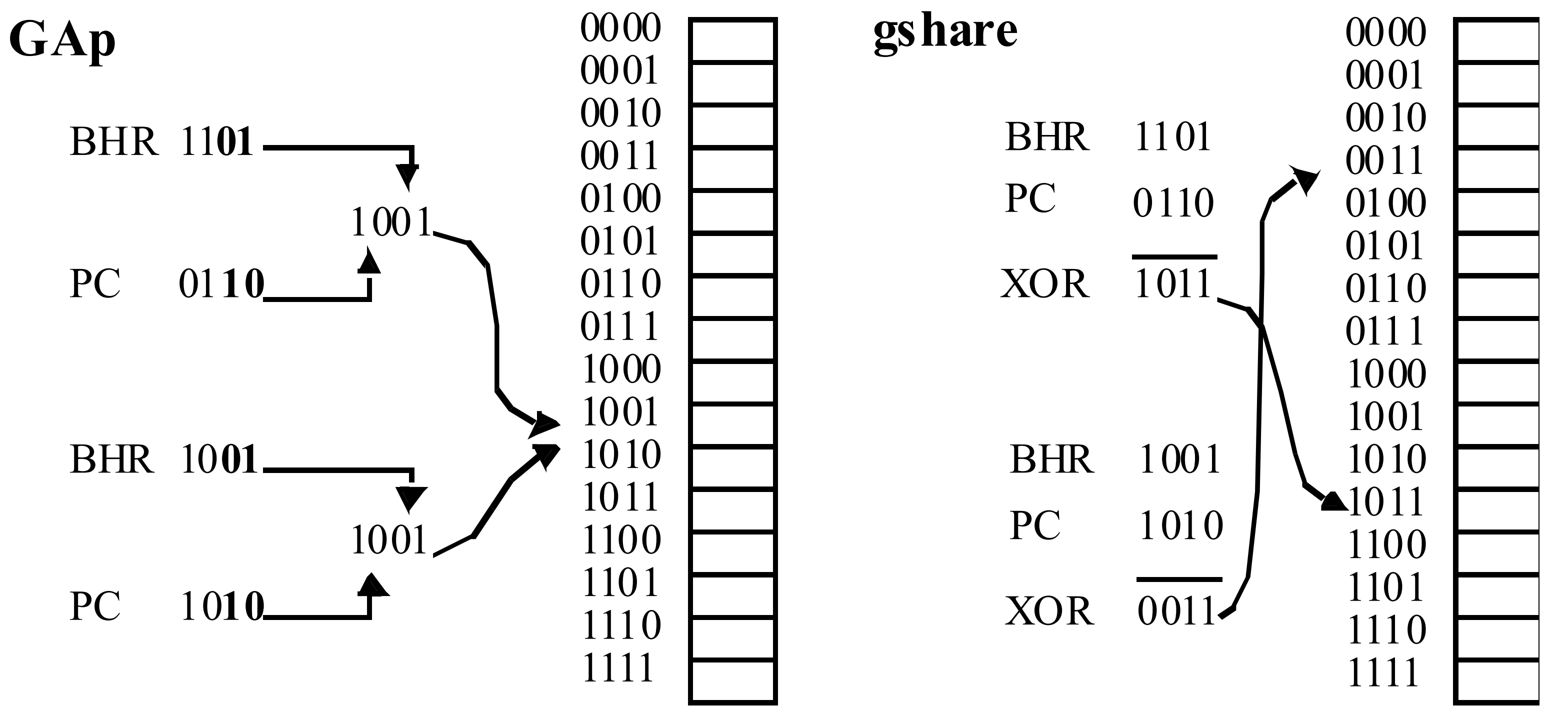


Two-level Taxonomy

- Based on indices for branch history and pattern history
  - BHR: {G,P,S}: {Global, Per-address, Set}
  - PHT: {g,p,s}: {Global, Per-address, Set}
  - 9 combinations: GAg, GAp, GAs, PAg, PAp, PAs, SAg, SAp and Sas

- Tse-Yu Yeh and Yale N. Patt. Two-Level Adaptive Branch Prediction. International Symposium on Microarchitecture, pages 51-61, November 1991.

Index Sharing in Two-level Predictors



- Use XOR function to achieve better utilization of PHT
- Scott McFarling. Combining Branch Predictors. TN-36, Digital Equipment Corporation Western Research Laboratory, June 1993.

➢ Used in e.g. IBM Power 4, Alpha 21264

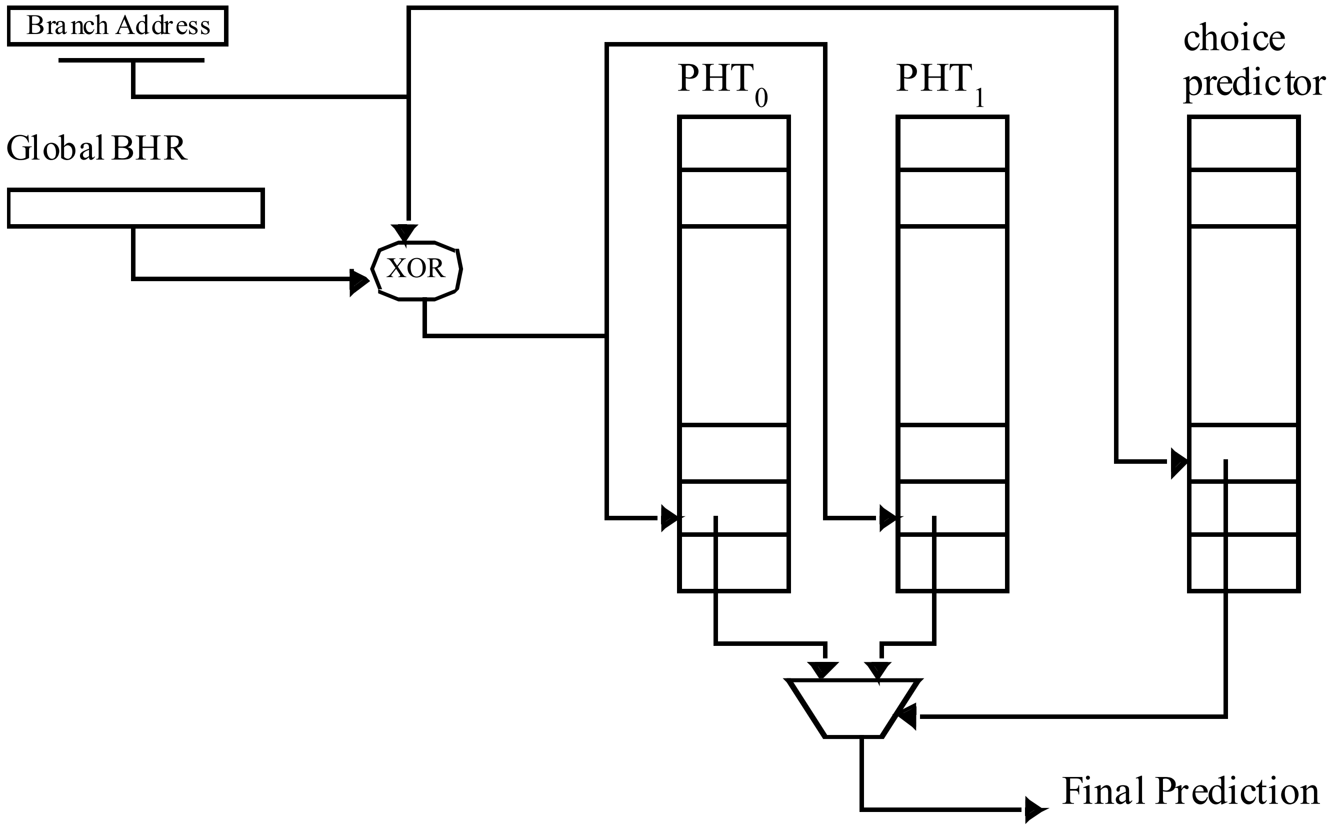
Sources of Mispredictions

- Lack of history (training time)
- Randomized behavior
  - Usually due to randomized input data
  - Surprisingly few branches depend on input data values
- BHR capacity
  - Correlate to branch that already shifted out
  - e.g., loop count > BHR width
- PHT capacity
  - Aliasing/interference
    - Positive
    - Negative

Reducing Interference

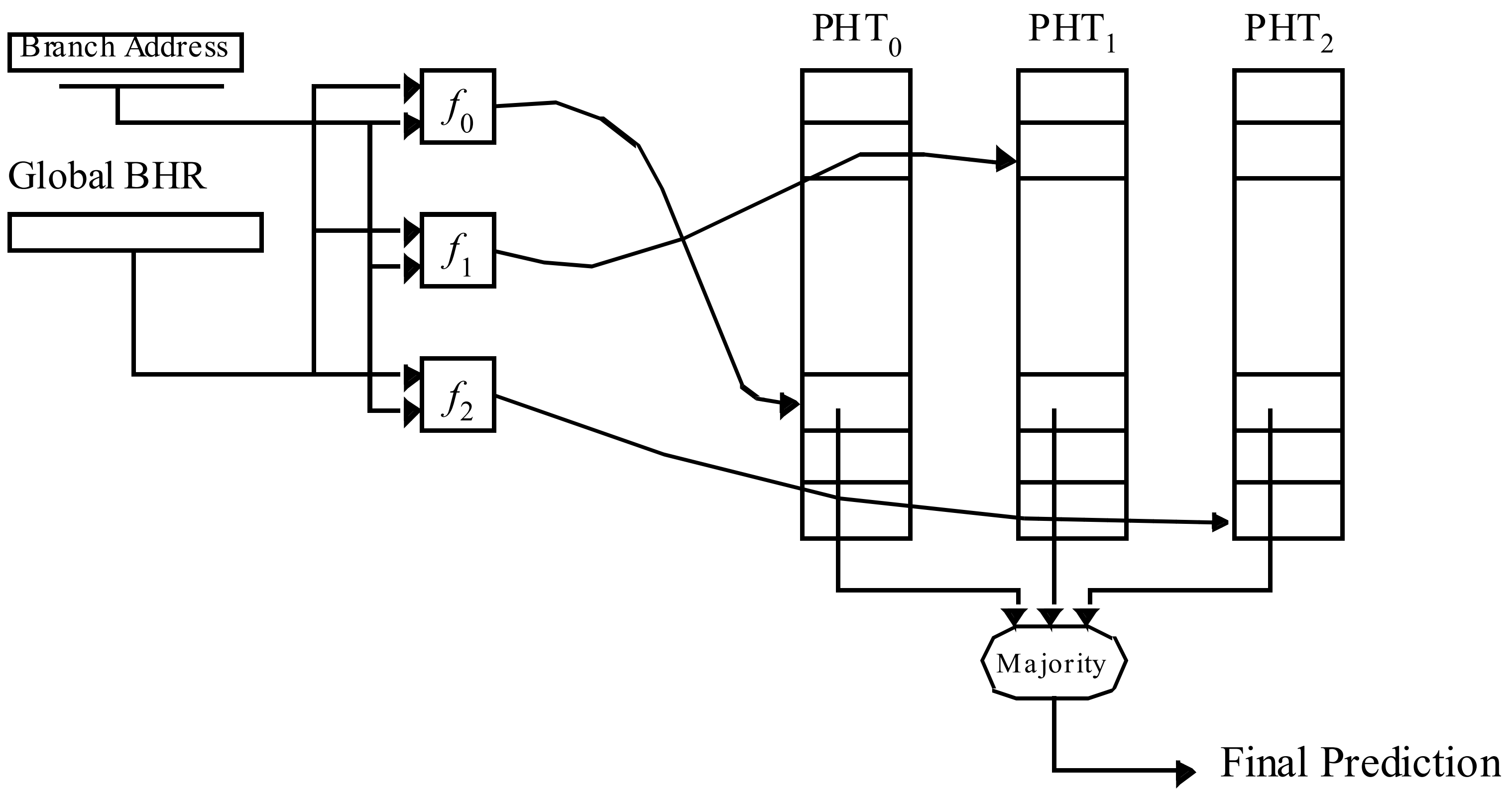
- Compulsory aliasing (*cold miss*)
  - Not important (less than 1%)
  - Only remedy is to set appropriate initial value
  - Also: beware indexing schemes with high training cost (e.g. very long branch history)
- Capacity aliasing (*capacity miss*)
  - Increase PHT size
- Conflict aliasing (*conflict miss*)
  - Change indexing scheme or partition PHT in a clever fashion

Bi-Mode Predictor



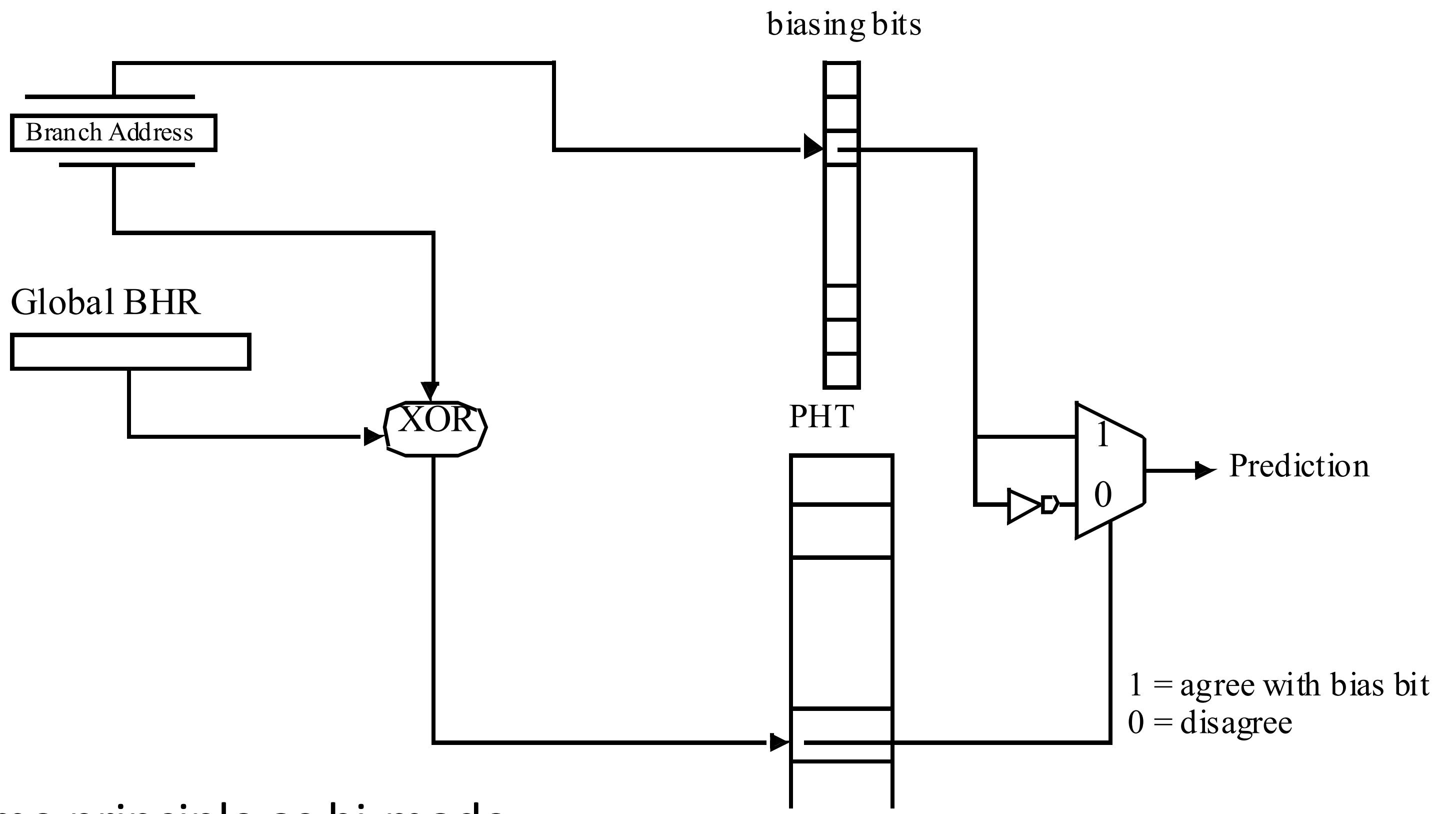
- PHT partitioned into T/NT halves
  - Selector chooses source
- Reduces negative interference, since most entries in PHT_0 tend towards NT, and most entries in PHT_1 tend towards T

gskewed Predictor



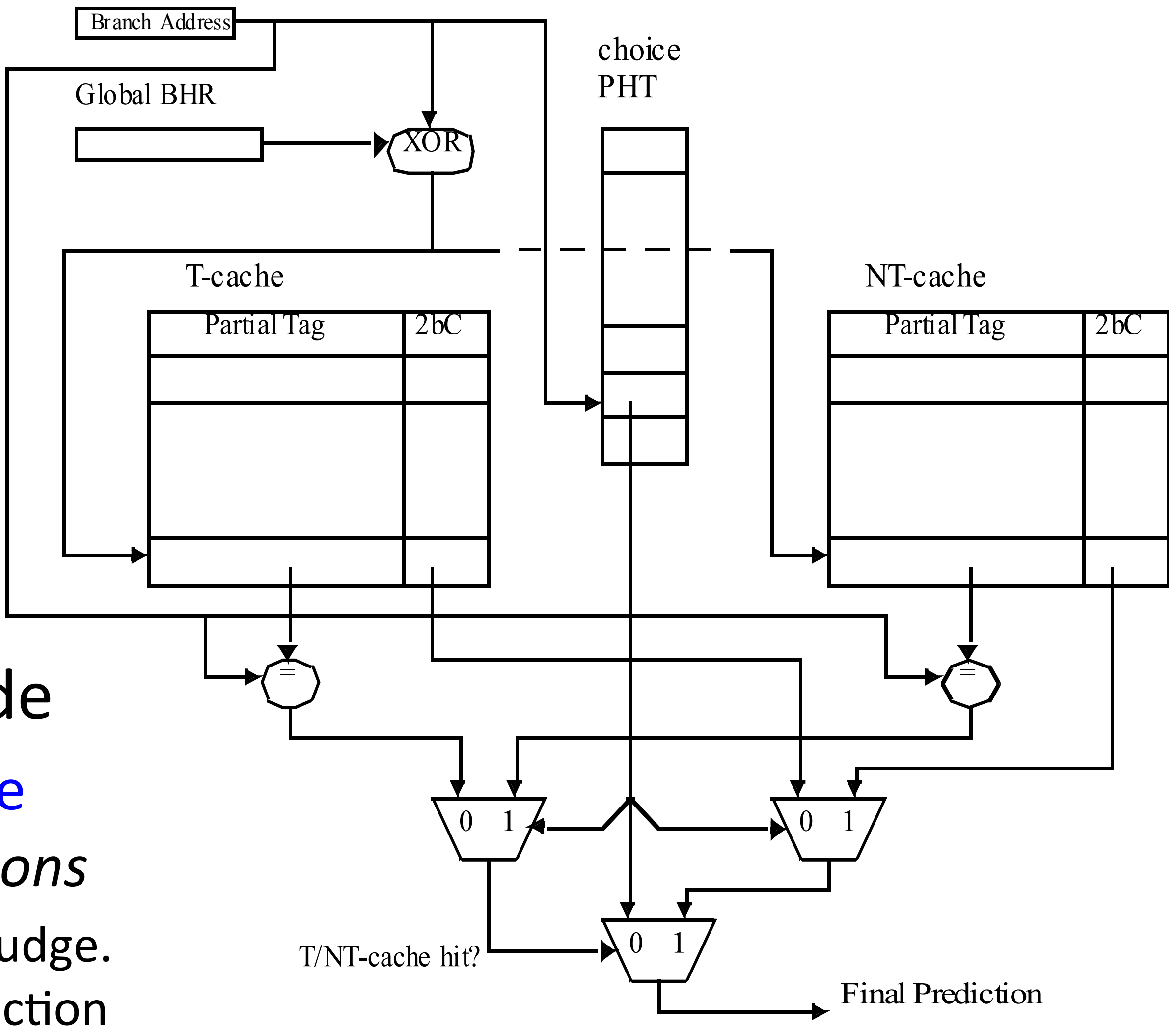
- Multiple PHT banks indexed by different hash functions
  - Conflicting branch pair unlikely to conflict in more than one PHT
- Majority vote determines prediction
- Used in Alpha EV8 (ultimately cancelled)
- P. Michaud, A. Seznec, and R. Uhlig. Trading Conflict and Capacity Aliasing in Conditional Branch Predictors. ISCA-24, June 1997

Agree Predictor



- Same principle as bi-mode
- PHT records whether branch bias matches outcome
  - Exploits 70-80% static predictability
- Used in in HP PA-8700
- E. Sprangle, R. S. Chappell, M. Alsup, and Y. N. Patt. The Agree Predictor: A Mechanism for Reducing Negative Branch History Interference. ISCA-24, June 1997.

YAGS Predictor



- Based on bi-mode
  - T/NT PHTs cache only the *exceptions*
- A. N. Eden and T. N. Mudge. The YAGS Branch Prediction Scheme. MICRO, Dec 1998.

Thank You