

Lecture 8 Branch Prediction (3.4, 3.5)

Instructor: Jun Yang

Nov. 9, 2004

Lec.8

1

About the project

- Change your email address on ilearn to your cs account
- Sign up for the mailing list of CS203A
- Find partner(s)

Nov. 9, 2004

Lec.8

2

Why do we want to predict branches?

- MIPS based pipeline – 1 instruction issued per cycle, branch hazard of 1 cycle.
 - Delayed branch
- Modern processor and next generation – multiple instructions issued per cycle, more branch hazard cycles will incur.
 - Cost of branch misfetch goes up
 - Pentium Pro – 3 instructions issued per cycle, 12+ cycle misfetch penalty
 - ❖ HUGE penalty for a misfetched path following a branch

Nov. 9, 2004

Lec.8

3

Branch Prediction

- Easiest (static prediction)
 - Always taken, always not taken
 - Opcode based
 - Displacement based (forward not taken, backward taken)
 - Compiler directed (branch likely, branch not likely)
- Next easiest
 - **1 bit predictor** – remember last taken/not taken per branch
 - ❖ Use a *branch-prediction buffer* or *branch-history table*
 - ❖ Use part of the PC (low-order bits) to index buffer/table
 - Multiple branches may share the same bit
 - ❖ Invert the bit if the prediction is wrong
 - ❖ Backward branches for loops will be mispredicted twice

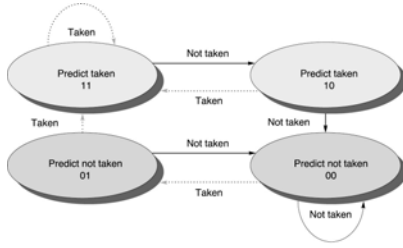
Nov. 9, 2004

Lec.8

4

2-bit Branch Prediction

- Has 4 states instead of 2, allowing for more information about tendencies
- A prediction must miss twice before it is changed
- Good for backward branches of loops



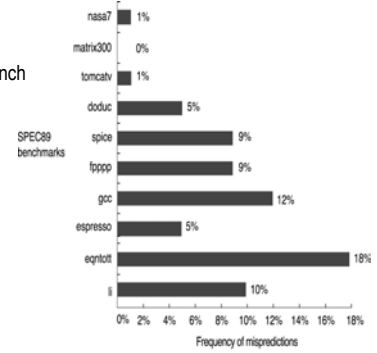
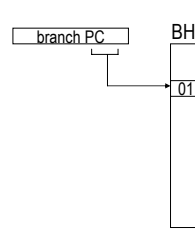
Nov. 9, 2004

Lec.8

5

Branch History Table

- Has limited size
- 2 bits by N (e.g. 4K)
- Uses low-order bits of branch PC to choose entry



Nov. 9, 2004

Lec.8

6

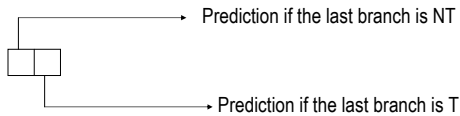
Can we do better ?

- Correlating branch predictors also look at other branches for clues

```

if (aa==2)           T
    aa = 0
if (bb==2)           T
    bb = 0
if (aa!=bb) { ...   NT

```



(1,1) predictor – uses history of 1 branch and uses a 1-bit predictor

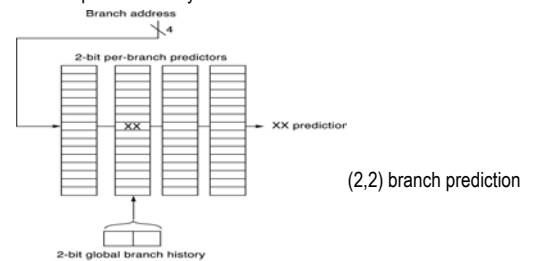
Nov. 9, 2004

Lec.8

7

Correlating Branch Predictor

- If we use 2 branches as histories, then there are 4 possibilities (T-T, NT-T, NT-NT, NT-T).
- For each possibility, we need to use a predictor (1-bit, 2-bit).
- And this repeats for every branch.



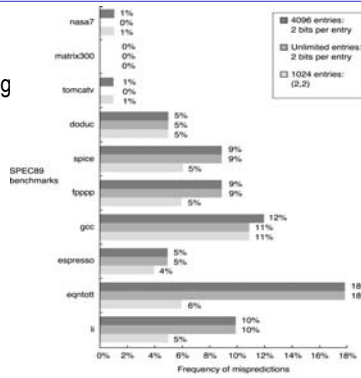
Nov. 9, 2004

Lec.8

8

Performance of Correlating Branch Prediction

- With same number of state bits, (2,2) performs better than noncorrelating 2-bit predictor.
- Outperforms a 2-bit predictor with infinite number of entries



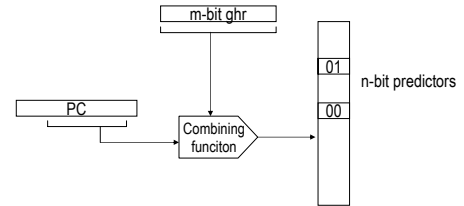
Nov. 9, 2004

Lec.8

9

General (m,n) Branch Predictors

- The *global history register* is an m -bit shift register that records the last m branches encountered by the processor
- Usually use both the PC address and the GHR (2-level)



Nov. 9, 2004

Lec.8

10

Is Branch Predictor Enough?

- When is using branch prediction beneficial?
 - When the outcome is known later than the target
 - For example, in our standard MIPS pipeline, we compute the target in ID stage but testing the branch condition incur a structure hazard in register file.
- If we predict the branch is taken and suppose it is correct, what is the target address?
 - Need a mechanism to provide target address as well
- Can we eliminate the one cycle delay for the 5-stage pipeline?
 - Need to fetch from branch target immediately after branch

Nov. 9, 2004

Lec.8

11

Branch Target Buffer (BTB)

Is the current instruction a branch ?

- BTB provides the answer before the current instruction is decoded and therefore enables **fetching to begin** after IF-stage .

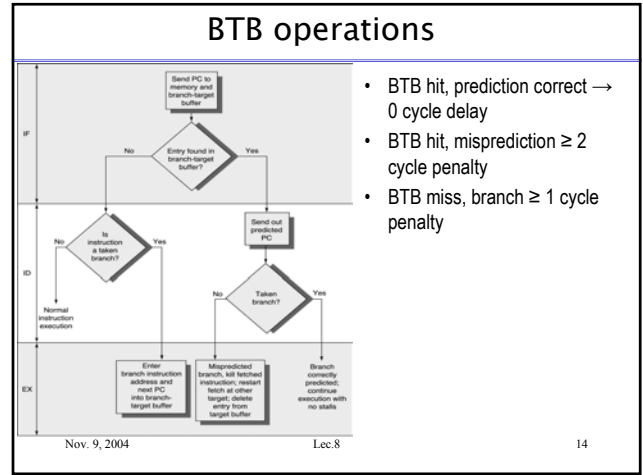
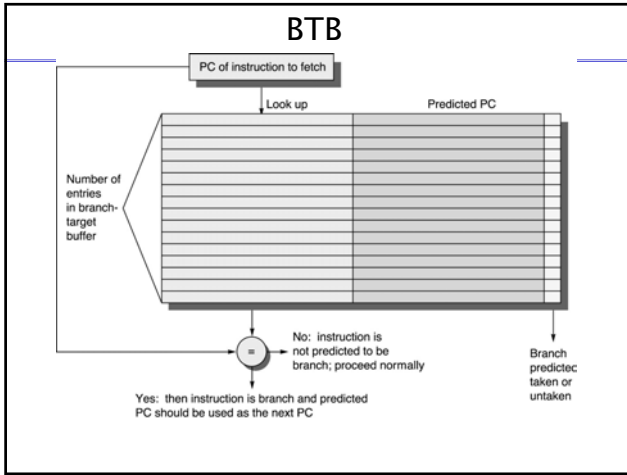
What is the branch target ?

- BTB provides the branch target if the prediction is a **taken direct branch** (for not taken branches the target is simply $PC+4$) .

Nov. 9, 2004

Lec.8

12



BTB Performance

- Two things can go wrong
 - BTB miss (mismatch)
 - Mispredicted a branch (mispredict)
- Suppose for branches, BTB hit rate of 85% and predict accuracy of 90%, mismatch penalty of 2 cycles and mispredict penalty of 5 cycles. What is the average branch penalty?

$$2 \cdot (15\%) + 5 \cdot (85\% \cdot 10\%)$$
 see also the example on Pg. 211
- BTB and BPT can be used together to perform better prediction

Nov. 9, 2004
Lec. 8
15

Indirect jumps/returns

- BPT does well with conditional branches
- BTB does well with unconditional direct jumps (jumps etc)
- Indirect jumps often jump to different destinations, even from the same instruction.
 - Procedure returns (most often used)
 - select, or case statements
- Return can easily be handled by stack
 - jal → push PC+4
 - return → predict jump to address on top of stack, pop stack

Nov. 9, 2004
Lec. 8
16

Branch Prediction Summary

- The better we predict, the higher penalty we might incur
- 2-bit predictors capture tendencies well
- Correlating predictors improve accuracy, particularly when combined with 2-bit predictors
- Accurate branch prediction does no good if we don't know there was a branch to predict
- BTB identifies branches in IF stage
- BTB combined with branch prediction table identifies branches to predict, and predicts them well