18-547 Lecture 19

Instruction Level Parallelism

Instructor: Prof. Falsafi

Electrical & Computer Engineering
Carnegie Mellon University

Slides developed by Profs. Hill, Wood, Sohi, and Smith of University of Wisconsin — Madison
Instruction Fetching

Multiple instructions must be fetched per cycle
IBM POWER solution e.g., RS6000 (simplified)
A branch can initiate fetching anywhere in a line
Contiguous lines are in an odd/even pair

RS6000 I-fetch

address

set

+1

mux

mux

Even Sets

Odd Sets

8 instructions

align/switch mux

4 instructions
Dynamic Branch Prediction

Why?  target comp (indep) vs. outcome comp (data dep)
  • have hardware guess whether and where a branch will go
  e.g.,  address  instruction
      0x64   bnez r1, +4
      0x74   add r3, r2, r1
start with branch PC (0x64) and produce
  • prediction (T-taken or NT- not taken
  • prediction + target PC (0x74)
  • prediction + target PC + target instruction (add r3, r2,r1)

Two-Bit Counters

With single prediction bit, two mispredictions on loops
  e.g.
  TTTNNNTTNTTNTNTNN
Solution: use saturating counter to implement “hysteresis”

What about special loop predictor?
  • may be useful as part of hybrid predictor
Two-Bit Counters

Branch History Table

Branch PC => prediction
- prediction (T, NT)

2-bit global branch history

Chapter 4: Instruction Level Parallelism
Branch History Table

Read prediction with LSBs of branch PC
Change bit on misprediction
May use bit from wrong PC (aliasing)
Better predictors:
Two-bit state machine (invented by Jim Smith)
Correlating predictors (invented by Yale Patt)

Correlating Predictors

Different branches may be correlated:
If (aa == 2) aa = 0;
If (bb == 2) bb = 0;
If (aa != bb) { . . .
If the first two are true, third is false
Save recent branch outcomes (approximation to path followed)
  ※ use branch history register (BHR) to hold recent outcomes
  ※ use both PC and BHR to access table
  ※ recent proposals keep more complete path information
Two-Level Predictors (Yeh & Patt)

PC and BHR can be combined as:
○ concatenated
○ completely overlapped
○ partially overlapped

PAp predictor
PAg predictor
GAg predictor

PAp

Per-address history:
☑ Branches correlate with their own history
☑ Also, referred to as “local”

Per-address patterns:
☑ no conflicts among branches

What branch behavior do they predict?
**PAg**

**Per-address history:**
- Another local predictor

**Global patterns:**
- Take advantage of similar loop patterns
- Save space

**GAg**

**Global history:**
- Branch history correlation among all branches
- Also, referred to as "g"

**Global patterns:**
- Conflicts from all over the program
Gshare (McFarling)

Reduces the conflicts among program phases
Distributes patterns across tables
Uses hashing of PC