Tomasulo’s Alg. and Loop Unrolling

Consider a loop

LOOP:  
L.D F0, 0(R1)  
MUL.D F4, F0, F2  
S.D F4, 0(R1)  
DADDUI R1, R1,#-8  
BNE R1, R2, LOOP

We will assume that branch is always predicted as taken and issue instructions from two loop iterations

- Assume none of the load/store or FP operations have completed

Issue     Execute         Write result
L.D F0, 0(R1)  
MUL.D F4, F0, F2  
S.D F4, 0(R1)  

Dynamic Memory Disambiguation

Order of loads and stores must be preserved
Since they access memory locations we can examine order only after we calculate effective address
Effective address calculation is performed in order
- Address of a load is examined against A fields of all store buffers
- Address of a store is examined against A fields of all load and store buffers

Dynamic Hardware Branch Prediction

Predict the outcome of a branch
- Change the prediction after observing a few iterations
To achieve good effectiveness we must
- Have accurate prediction technique
- Have a low cost for misprediction

Local Prediction: Branch Prediction Buffer

A table indexed by low bits of branch instruction address
- It contains a bit indicating whether the branch was recently taken or not
- If it turns out we have been wrong the bit is inverted

1-bit Branch Prediction Buffer

Problem – even simplest branches are mispredicted twice
First time: prediction = 0  
but the branch is taken  
(change prediction to 1 miss)
Time 2, 3, 4: prediction = 1  
and the branch is taken  
Time 5: prediction = 1  
but the branch is not taken  
(change prediction to 0 miss)
2-bit Branch Prediction Buffer

To amend this we will use 2 bits, one if the previous branch was not taken, the other if it was taken.

! Assign two prediction bits, one if the previous branch was not taken, the other if it was taken. This is (1,1) predictor and it uses outcome of 1 previous branch to do prediction with 1-bit predictor.

n-bit Branch Prediction Buffer

We can generalize this technique to n-bit prediction buffers.

• When the counter is \(2^{n-1}\), branch is predicted as taken.

• Those predictors are not much more accurate than 2-bit.

Correlating (Global) Branch Predictors

Assign two prediction bits, one if the previous branch was not taken, the other if it was taken.

b1: BNEZ R1, L1
DADDDUI R1, R0, #1
L1: DSUBUI R3, R1, #1
b2: BNEZ R3, L2

Time 1: prediction = 00, not taken
Time 2: prediction = 01, not taken
Time 3: prediction = 10, not taken
Time 4: prediction = 11, taken
Time 5: prediction = 11, taken

Correlating Branch Predictors

Assign two prediction bits, one if the previous branch was not taken, the other if it was taken.

b1: BNEZ R1, L1
DADDDUI R1, R0, #1
b2: BNEZ R3, L2

Correlating Branch Predictors (m,n)

Observe behavior of m previous branches, use n-bit predictor.

Correlating Branch Predictors (m,n)
Correlating Branch Predictors (m,n)

- $2^m$ combinations, n-bits each

Branch address

- n bits
- n bits
- n bits

m bits indicating outcome of m previous branches

---

Tournament Predictors

- Combine one global and one local predictor with a selector

First selector was right
Second selector was wrong