

CS 3339 – Computer Architecture

Project 4: Branch Predictor Discussion

Instructor
Mr. Lee Hinkle



With deep acknowledgement to Dr. Martin Burtscher
and Ms. Molly O'Neil

Big Picture

There are a total of 6 projects this semester

Project 1 – Disassembler (machine code to assembly)

Project 2 – Emulator (machine code to operation)

Project 3 – Pipelining (simulator for cycle timing)

Project 4 – Branch Prediction (goal = fewer flushes)

Project 5 – Caching (add simulated data cache)

Project 6 – Parallel Processing

The projects are more important than their point values reflect – they reinforce key topics

Project 4

So far, projects assumed branch not taken

IF1 = PC + 8, IF2 = PC + 4, ID working on inst at PC

If branch was taken had to:

- Flush out bad fetches

- Set PC to new target address

Goal – to simulate the benefit of improving branch behavior by making a prediction based on past behavior to eliminate flush cycles if the prediction is correct

Need to store history info

We are storing info about the past behavior of branch instructions (is a given branch likely to be taken and to where)

This table is implemented as two arrays in the skeleton code (one has count, other has target addr)

This table is a type of cache. Caches are fast memory which hold small amounts of data likely to be used again.

Full data in p4 is the
pred and btb value for
every branch instruction

But our table has only
64 entries. We index it
using the pc address of
the branch instruction

Pg 384 and 385 in the textbook may help you consider how to implement the code

The Branch Predictor

addr	instr
1000	bne ...
1004	otherinstr
1008	beq ...
100C	otherinstr
1010	otherinstr
1014	beq ...
1018	otherinstr
101C	bne ...
1020	otherinstr
1024	beq ...
1028	otherinstr

$$\text{index} = (\text{PC}_{\text{branch}} \gg 2) \% \text{BPRED_SIZE}$$

index	pred	btb
0		
1		
2		
3		

2-bit saturating
counter
values = {0,1,2,3}
≥2 pred taken
else pred not taken

holds the target
address – where
you think the
branch will go