CS220
Introduction to Computer Organisation
Lab 8

Abhinav Agarwal[*]

1 Introduction

In this lab, you will experiment with the multiple branch predictor implementation of the 6-stage pipelined processor. A 6-stage pipelined implementation will be provided, and it will contain skeleton code for the branch predictor, and the mispredict related functionality such as multiple epochs.

Compiling and running the code can be done identically to the previous labs, using the `smips` run script.

2 A 6-stage pipeline with branch predictors

The provided 6-stage pipelined SMIPS processor is comprised of three source files, `Proc.bsv`, `AddrPredict.bsv` and `DirPredict.bsv`. `AddrPredict.bsv` implements the next instruction address predictor used at the `Fetch` stage, and `DirPredict.bsv` implements the branch target predictor used at the `Decode` stage.

The usage and update of the two predictors are encapsulated in `handleFetchRedirect` and `handleDecodeRedirect`, respectively.

The mispredict related issues such as multiple epochs have already been implemented.

Exercise (30 points): In the code given, both predictors' update methods do nothing, and the predictors return static values. Your job is to complete the implementation of the modules `mkBtb` and `mkCounterPred2Bit` according to the material covered during class. Also implement a 1 bit direction predictor `mkCounterPred1Bit` and a 3 bit direction predictor `mkCounterPred3Bit`. Experiment with different implementations, and try to get the cycle count to decrease. Submit the design that gives best possible cycle count for benchmarks, and describe the configuration in `discussion.txt`.

[*] Adapted by Amey Karkare for CS220

Discussion (10 points): How much difference in cycle counts does having no branch predictors, one branch predictor, and two branch predictors have? Do you feel it's significant?

Discussion (5 points): For the branch target predictor, does the two bit implementation perform much better than the 1 bit implementation? How about a 3 bit implementation?