CS 378 Research in Computer Architecture — Spring 2019	Programming Assignment #1
Branch Prediction	Due January 31, 2019

In this assignment you will write code for branch predictors, and you will compare their accuracy and performance using a microarchitectural simulator. We will provide the microarchitectural simulator along with the interfaces that you will need to implement your branch predictors. This assignment does not require you to write a lot of code, but it does require you to explore various design choices, and it requires you to carefully evaluate your designs.

1 Your Assignment

The first part of your assignment is to implement several simple branch predictors inside a microarchitectural simulator.

- `Always Taken`: The branch predictor always predicts that the branch is taken.
- `Always Not Taken`: The branch predictor always predicts that the branch is not taken.
- `1-bit Bimodal Branch Predictor`: The branch predictor predicts the last branch outcome for every branch.
- `k-bit Bimodal Branch Predictor`: The branch predictor predicts the common case for the last k branch outcomes for every branch. You are encouraged to try different values of k to find the configuration with the best performance.
- `Two-level Branch Predictor`: The branch predictor uses branch histories and pattern history tables to predict future branch outcomes. The simplest variation of a 2-level predictor uses a global history and a global pattern history table, and you should try varying the history length to see its impact on the predictor's accuracy. We encourage you to further explore the design space by using local histories and local pattern tables.

The second part of your assignment is to run experiments on your branch predictors. We will provide several benchmarks which you will use to compare the accuracy of your branch predictors. You will also compare the impact of your branch predictor's accuracy on the program's execution time.

The third part of your assignment is to describe what you've done, report your results and draw relevant conclusions from your experiments.

2 Details

2.1 Simulator

We will use the ChampSim simulator for this assignment. ChampSim is a trace-based microarchitectural simulator that simulates the effect of executing a program on a software CPU model. ChampSim allows you to observe a variety of useful statistics about the underlying hardware's performance. For this assignment, you will measure the branch predictor's accuracy and its impact on program performance. ChampSim reports the execution time in CPU cycles. A useful measure of the hardware's performance is the number of instructions it executes every cycle (also called the IPC). We have provided scripts to compute both these metrics. The README provides the instructions to compile and run the simulator.

2.2 Benchmarks

The 4 benchmarks for evaluating your designs are linked with the scripts we have provided, and require you to use a UTCS lab machine in order to work properly. For each benchmark, we have provided a trace of 1 billion instructions from a representative region of the program (large programs typically run for billions of instructions, which is too slow to simulate on a microarchitectural simulator). For this assignment, the script will run each benchmark for 50 million instructions, although you can experiment with this if you like. Please note that these simulations will be slow, and each simulation can take 2-10 minutes to finish.

2.3 Report

You should write a report that describes your experiments, the results you got from these experiments and your conclusions from the results. If you've experimented with design points that were not described in the assignment, please include a description of those with an explanation for why you chose to do those experiments.

3 What To Turn In

Please submit your branch predictor files and your report in a .zip file as a Private Note on Piazza by Thursday, Jan 31, 11:59 pm.

4 Extra Credit

If you are feeling adventurous, feel free to try your own branch predictor designs for extra credit. You are welcome to use any ideas that we've discussed in class. These modifications can be turned in for extra credit by February 4, 2019.