**NC State University**

**Department of Electrical and Computer Engineering**

**ECE 463/521: Fall 2014**

**Project #2: Branch Prediction**

**by**

**Harshvardhan Aggarwal**

NCSU Honor Pledge: "I have neither given nor received unauthorized aid on this test or assignment."

Student’s electronic signature: \_\_Harshvardhan Aggarwal\_\_

(sign by typing your name)

Course number: \_\_\_\_\_\_\_521\_\_\_\_\_\_\_

(463 or 521 ?)

**Analysis of Branch Predictor**

1. **Bimodal Predictor**
2. Gcc\_trace.txt
3. Perl\_trace.txt
4. Jpeg\_trace.txt

Analysis of Bimodal Branch Predictor

From the above 3 graphs of gcc\_trace, perl\_trace, and jpeg\_trace, it can be concluded that the misprediction rate decreases as the number of bits used to index into bimodal table is increased. Also, in gcc\_trace and perl\_trace cases, the misprediction rate shows diminishing returns as the number of bits become large. However, in jpeg\_trace, the misprediction rate starts to level-off from start only. The rate does not reduce considerably for increasing number of bits. Thus, for all three cases, the additional hardware required to support more index bits does not increase performance.

Design:

For gcc\_trace, m = 11 gives a bimodal predictor design that reduces misprediction rate (13.65) and also minimizes predictor cost in bits. For greater values of m, the misprediction rate reduces in steps of 1%, which doesn’t offset costs required to support more bits for predictor.

For perl\_trace, m = 10 gives a bimodal predictor design that reduces misprediction rate (11.95) and also minimizes predictor cost in bits. For greater values of m, the misprediction rate reduces very slowly, and hence doesn’t provide increase performance while supporting more bits for predictor.

For jpeg\_trace, m = 7 gives a bimodal predictor design that reduces misprediction rate (13.65) and also minimizes predictor cost in bits. For increasing values of m, the misprediction rate reduces very slowly by less than 0.2% which doesn’t offset costs required to support more bits for predictor.

1. **Gshare Branch Predictor**
2. Gcc\_trace.txt
3. Jpeg\_trace.txt
4. Perl\_trace.txt

Analysis of Gshare Branch Predictor

From the above 3 graphs of gcc\_trace, perl\_trace, and jpeg\_trace, it can be concluded that the misprediction rate decreases as the number of bits used to index into bimodal table is increased and simultaneously the number of bits used for Global Branch history register is also increased. This means that as the BHR takes into account history of branches takes, therefore, more bits means more history and hence it performs better. Also, in gcc\_trace and perl\_trace cases, the misprediction rate shows diminishing returns as the number of bits become large. However, in jpeg\_trace, the misprediction rate starts to level-off from start only. The rate does not reduce considerably for increasing number of bits. Thus, for all three cases, the additional hardware required to support more index bits does not increase performance. Another interesting rend to note is when m is kept constant, the misprediction rate increases as we increase the bits for BHR. This can be attributed to that very old history doesn’t contribute to predict correctly and thus decreases performance.

Design:

For gcc\_trace, M = 11 and N = 2 gives a gshare predictor design that reduces misprediction rate (13.71) and also minimizes predictor cost in bits. For greater values of m, the misprediction rate reduces in steps of 1% and similarly for larger values of N the misprediction rate also increases, which doesn’t offset costs required to support more bits for predictor.

For perl\_trace, M = 11 and N = 6 gives a gshare predictor design that reduces misprediction rate (8.6) and also minimizes predictor cost in bits. For greater values of m, the misprediction rate reduces very slowly, and hence doesn’t provide increase performance while supporting more bits for predictor. Similarly, for greater values of N, the misprediction rate increases and hence does not provide improvement in performance.

For jpeg\_trace, M = 7 and N = 2 gives a gshare predictor design that reduces misprediction rate (8.0) and also minimizes predictor cost in bits. For increasing values of m, the misprediction rate reduces very slowly by less than 0.2% and similarly, for greater values of N, the misprediction rate increases on average by more than 1%, which doesn’t offset costs required to support more bits for predictor.