Rojin Atefimehr

Project Final Report

The objective of this project was to develop a program capable of simulating a 1024-entry BTB using program traces that provide a sequence of program counter (PC) addresses. By extracting specific bits from these addresses, the program determines the BTB entry and predicts the behavior of branch instructions, whether they will be taken or not.

Based on the last digit of my id which is an odd number

1. A) Espresso_int with Class State Machine:

   Total Instructions: 809370
   Hit: 130256
   Miss: 452
   Right: 123405
   Wrong: 6851
   Taken_Branch: 94774
   Collision: 452
   Hit Rate: 99.65%
   Accuracy: 94.74%
   Wrong Percentage: 9.79%

   B) Espresso_int with State Machine B:

   Total Instructions: 809370
   Hit: 130256
   Miss: 452
   Right: 123821
   Wrong: 4957
   Taken_Branch: 94774
   Collision: 452
   wrong Prediction: 671
   Hit Rate: 99.65%
   Accuracy: 95.06%
   Wrong Percentage: 13.54%

   C) Spice_FP with Class State Machine:

   Total Instructions: 799451
   Hit: 141760
   Miss: 10085

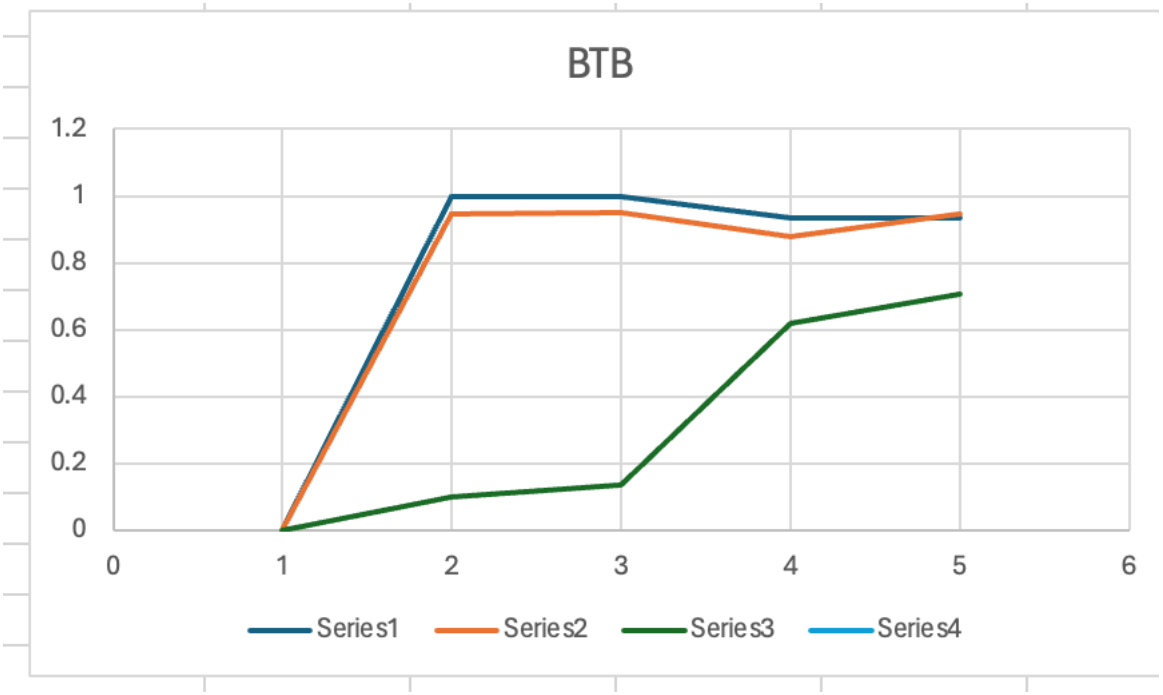
Right: 124247
Wrong: 17513
Taken_Branch: 132207
Collision: 10085
Hit Rate: 93.36%
Accuracy: 87.65%
Wrong Percentage: 61.94%

D) Spice_FP with State Machine B:

Total Instructions: 799451
Hit: 141760
Miss: 10085
Right: 134320
Wrong: 15390
Taken_Branch: 132207
Collision: 10085
wrong Prediction: 10848
Hit Rate: 93.36%
Accuracy: 94.75%
Wrong Percentage: 70.49%

2.

	A	B	C	D
	States	Hit Rate	Accuracy	Address_Wrong
	Espresso_int with Class State Machine	99.65%	94.74%	9.79%
	Espresso_int with State Machine B	99.65%	95.06%	13.54%
	Spice_FP with Class State Machine	93.36%	87.65%	61.94%
	Spice_FP with State Machine B	93.36%	94.75%	70.49%



Blue: Hit Rate
Orange: Accuracy
Green: Wrong Prediction Address

3.

According to my results, the BTB's performance varies between the Espresso_int and Spice_FP benchmarks. For Espresso_int, both state machines offer high hit rates, but State Machine B has better accuracy despite a higher chance of incorrect addresses. With Spice_FP, State Machine B improves accuracy but also increases the chance of wrong addresses more than Class State Machine. In essence, State Machine B seems to offer a slight edge in predictive accuracy across both benchmarks.

When considering the overall efficacy between the Class State Machine and State Machine B, it's evident that State Machine B tends to outperform in accuracy. However, this comes with the trade-off of higher Address Wrong rates. This trade-off suggests that while State Machine B might be the better choice for scenarios where the most accurate branch prediction is necessary, the Class State Machine could be preferable in situations where minimizing the incidence of incorrect branch predictions is critical, particularly for applications sensitive to misdirection, such as Spice_FP.

Therefore, if the aim is to prioritize precision in branch prediction, State Machine B may be the better performer, especially for integer operations like those in Espresso_int. However, when considering the importance of correct branch address predictions, especially in complex floating-point operations as seen with Spice_FP, the Class State Machine holds its ground by keeping the rate of wrong addresses lower.

4. The usefulness of prediction schemes like BTBs in pipelined processors is clear: they enhance efficiency by reducing stalls due to branch instructions. A well-tuned BTB allows the processor to maintain steady instruction flow, essential for high performance. However, the accuracy of predictions is critical. While high accuracy minimizes stalls, incorrect predictions can lead to costly pipeline flushes. Therefore, while BTBs are beneficial, their design must align with the workload and processor architecture to truly improve performance. In essence, the effectiveness of BTBs is measured by their ability to strike a balance between accurate predictions and minimal misdirection, ensuring a smooth and efficient pipeline operation.