Justin Leung - 1002949875
Yongrui Zhang - 1002261759

Lab 2 Report

Microbenchmark:

```
int main (void) {
   int a = 0;
   int i;
   for (i = 0; i < 10000; i++){ // for loop branch consistently taken
      if((i % 6) == 0){ // periodic conditional branch, predictable by our 2 - level predictor
         a = 3;
      }
   }
   return 0;
}
```

MPKI: 2 - Bit Sat = 12.598, 2 - Level = 6.340, Open Ended = 5.973

This microbenchmark will test the 2-level Branch Predictor for:

1) Branches that are consistently taken (for loop)
2) Branches that are taken periodically (first if statement)

```
NUM_INSTRUCTIONS          :     294167
NUM_CONDITIONAL_BR        :      41421

2bitsat: NUM_MISPREDICTIONS       :       3706
2bitsat: MISPRED_PER_1K_INST      :     12.598
2level:  NUM_MISPREDICTIONS       :       1865
2level:  MISPRED_PER_1K_INST      :      6.340
openend: NUM_MISPREDICTIONS       :       1757
openend: MISPRED_PER_1K_INST      :      5.973
```

Figure 1 - 10000 loop iterations

```
NUM_INSTRUCTIONS          :    1929167
NUM_CONDITIONAL_BR        :     221421

2bitsat: NUM_MISPREDICTIONS       :      18706
2bitsat: MISPRED_PER_1K_INST      :      9.696
2level:  NUM_MISPREDICTIONS       :       1865
2level:  MISPRED_PER_1K_INST      :      0.967
openend: NUM_MISPREDICTIONS       :       1757
openend: MISPRED_PER_1K_INST      :      0.911
```

Figure 2 - 100000 loop iterations

```
.L4:
    movl    -4(%rbp), %ecx
    movl    $715827883, %edx
    movl    %ecx, %eax
    imull   %edx
    movl    %ecx, %eax
    sarl    $31, %eax
    subl    %eax, %edx
    movl    %edx, %eax
    addl    %eax, %eax
    addl    %edx, %eax
    addl    %eax, %eax
    subl    %eax, %ecx
    movl    %ecx, %edx
    testl   %edx, %edx
    jne .L3
    movl    $3, -8(%rbp)
.L3:
    addl    $1, -4(%rbp)
.L2:
    cmpl    $99999, -4(%rbp)
    jle .L4
```

Figure 3 - Assembly code snippet

To validate our 2-Level Predictor, we expect the periodic conditional branch in our microbenchmark to be predictable since we use 6 history bits. We can verify this by running our benchmark with 100000 loop iteration and see that the number of mispredictions remain the same(see Figure 1 & 2). As seen in Figure 3, there is the *jne .L3* which is the periodic conditional branch where *a=3 / movl $3,-8(%rbp)* is the next line and the *jle .L4* which is the for loops conditional branch.

MPKI:

Benchmark	2-Bit Saturating	2-Level	Open Ended
astar	24.639	11.903	6.287
bwaves	7.886	7.146	6.339
bzip2	8.166	8.651	7.654

gcc	21.079	14.824	5.342
gromacs	9.088	7.484	5.711
hmmer	13.567	14.872	11.696
mcf	24.387	13.494	10.236
solplex	7.107	6.819	5.021
Average	14.49	10.65	7.28

Open Ended Predictor Description:
Our team implemented a Perceptron Branch Predictor. We have a global history register that is 32 bits, and a 2-D private perceptron predictor table array that is 512x32. We use the least significant 9 bits of the PC to access the weights. Basic logic is as follows: set all weights to 0. Make predictions based on learned weights. Update weights based on history and whether the prediction is above a certain threshold.
Size:
Global history register - 32 bits
Perceptron weight table - 512 * 32 * 7
Total = 114720 Bits
CACTI:
Two level Predictor:

	Modified	Cache height x width (mm)	Access time (ns)	Total leakage power (mW)
History Table (2level-bpred-1.cfg)	Total size - 64 bytes Block size - 1 byte	0.0256629 x 0.0149475	0.235609	0.0645814
Predictor Table (2level-bpred-2.cfg)	Total size - 128 bytes Block size - 2 bytes	0.0246389 x 0.0219721	0.260657	0.0882322

Open Ended Predictor

	Modified	Cache height x width (mm)	Access time (ns)	Total leakage power (mW)
Weight Table (open-ended-bpred.cfg)	Total size - 14336 bytes Block size - 28 bytes	0.117394 x 0.213657	0.445554	5.17338

Work Contributions:

Justin Leung - Worked together on 2 bit sat, 2 - level, open ended predictor and report. Created microbenchmark and carried out benchmark verification.
Yongrui Zhang - Worked together on 2 bit sat, 2 - level, open ended predictor and report. Worked on CACTI.