ECE552 Lab Assignment 2: Dynamic Branch Prediction

Justin Sabatini & Viet Minh Nguyen

1. Microbenchmark Statistical Validation *("gcc mb.c -o mb" - no optimization)*

The benchmark has an outer loop (1000000), and an inner loop (10). Inside the inner loop, we utilize the branch condition (i+j) % 3 == 0. As i changes with every iteration, it produces a varying branch pattern. Overall branch outcome: 1000000*(T(part1)(part2)(part3)) + (part 1) +N

We have 6 history bits that should recognize the entire branching pattern (e.g TTTNTN); all branch instruction PCs are only either (outer loop), (inner loop), or (if), so a PAp should predict very well (matching very low **MPKI = 0.076**). Also, there should be at least 3,000,000 conditional branches (correct - the rest may come from printf statement or main() function).

```c
int main() {
    int result = 0;

    // Loop a significant number of times to stress-test the predictor.
    // The expected branch outcome would be:
    // T - (then inner outcomes) * (1000000 / 3) - N
    for (int i = 0; i < 1000000; i++) {
        // Nested loops can create more complex branch patterns.
        // The expected branch outcome is (inner outcomes)(for-if outcome pairs):
        // Part 1) i % 3 == 0: TT TN TN TT TN TN TT TN TN TT N
        // Part 2) i % 3 == 1: TN TN TT TN TN TT TN TN TT TN N
        // Part 3) i % 3 == 2: TN TT TN TN TT TN TN TT TN TN N
        for (int j = 0; j < 10; j++) {
            if ((i+j) % 3 == 0) {
                result += j;
            }
            result -= j;
        }
    }
    printf("Result: %d\n", result); // To avoid optimization out.
    return 0;
}
```

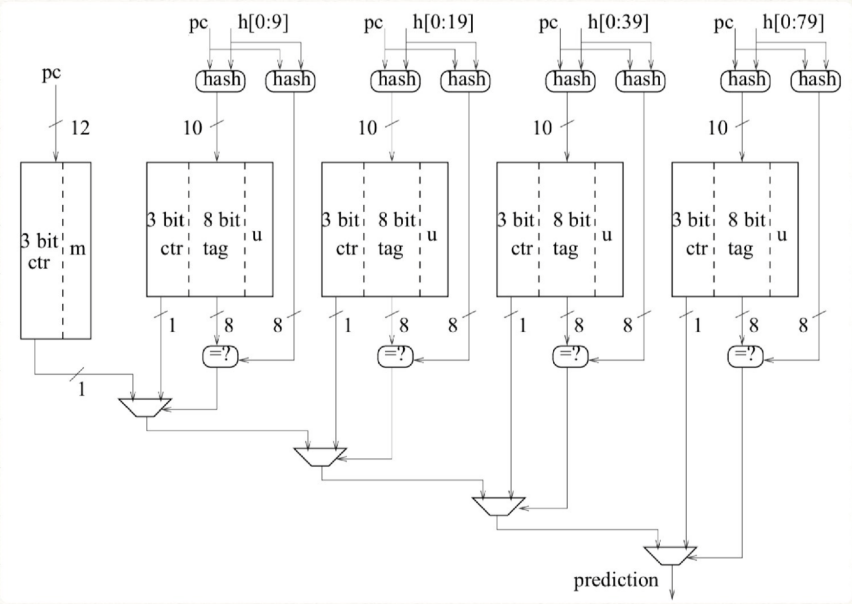
```
ug169:~/ece552/cbp4-assign2% predictor branchtrace.gz
.........................

NUM_INSTRUCTIONS         :   25724851
NUM_CONDITIONAL_BR       :    3022232

2bitsat: NUM_MISPREDICTIONS      :     394994
2bitsat: MISPRED_PER_1K_INST     :     15.355
2level:  NUM_MISPREDICTIONS      :       1949
2level:  MISPRED_PER_1K_INST     :      0.076
openend: NUM_MISPREDICTIONS      :       1643
openend: MISPRED_PER_1K_INST     :      0.064
```

2. Mispredictions Table

	2-bit sat counter	PAp with 2-bit sat counter	open-ended predictor
astar	Mispredicted: 3695923 MPKI: 24.639	Mispredicted: 1785582 MPKI: 11.904	Mispredicted: 604539 MPKI: **4.030**
bwaves	Mispredicted: 1181950 MPKI: 7.880	Mispredicted: 1072046 MPKI: 7.147	Mispredicted: 560515 MPKI: **3.737**
bzip2	Mispredicted: 1224989 MPKI: 8.167	Mispredicted: 1297738 MPKI: 8.652	Mispredicted: 1264983 MPKI: **8.433**
gcc	Mispredicted: 3161205 MPKI: 21.075	Mispredicted: 2223775 MPKI: 14.825	Mispredicted: 447941 MPKI: **2.986**
gromacs	Mispredicted: 1361054 MPKI: 9.074	Mispredicted: 1122653 MPKI: 7.484	Mispredicted: 1048521 MPKI: **6.990**
hmmer	Mispredicted: 2035059 MPKI: 13.567	Mispredicted: 2230922 MPKI: 14.873	Mispredicted: 2118689 MPKI: **14.125**
mcf	Mispredicted: 3657995 MPKI: 24.387	Mispredicted: 2024355 MPKI: 13.496	Mispredicted: 1755410 MPKI: **11.703**
soplex	Mispredicted: 1066132 MPKI: 7.108	Mispredicted: 1023018 MPKI: 6.820	Mispredicted: 710770 MPKI: **4.738**

3. Open-ended Branch Predictor: Based on A PPM-like, tag-based branch predictor

- Global history register: 40 bits
- Base predictor (index = **PC & 0xFFF**): 4096 * 4 bits
- 4 Bank predictors (index = **hash(PC, varying # global history bits)**): 4 * 1024 * 12 bits
  - Tag is calculated separately from index.
  - The chosen prediction is the highest table with a tag match
  - If no tag match, base predictor is used
- Total: 40 + 16k + 4 × 12k = **64K bits** << 128K



4. Predictor Characteristics using CACTI:

Predictor	Two-Level	Open-Ended

Parameter changed	2 tables -> 2 files based on cache.cfg
	- **2level-bpred-1** (BHT)
	  - size = 512, block size = 1
	  - RW port = 1
	- **2level-bpred-2** (PHT)
	  - size = 512, block size = 1
	  - RW port = 2
		- **open-ended-bpred-1** (Base)
		  - Based on cache.cfg (no tags)
		  - size = 4096, block size = 1
		- **open-ended-bpred-2** (Banks 1-4)
		  - Based on pureRAM.cfg (NOTE)
		  - size = 1024, block size = 1
		  - Tag size = 8

Area	0.001 mm^2+ 0.00236 mm^2 = **0.00336mm^2**	0.0068 mm^2 + 0.0038mm^2 * 4 = **0.022mm^2**
Access Latency	0.163585 ns + 0.175571 ns = **0.339ns**	max(0.21062 ns, 0.334774 ns) = **0.334774 ns** (all banks are accessed in parallel)
Leakage Power	0.195 mW + 0.245847 mW = **0.4408mW**	1.523 mW + 0.7747 mW * 4 = **4.622 mW**

NOTE: chose pureRAM.cfg to take into consideration the tag matching comparators that are used in our design. However, the bank entries should be accessed in this order: index - get tag

5. Work Distribution:

Justin Sabatini: Designed the microbenchmark, 2-bit saturating counter predictor, report.pdf
Viet Minh Nguyen: 2-level PAp, open-ended, CACTI, report.pdf