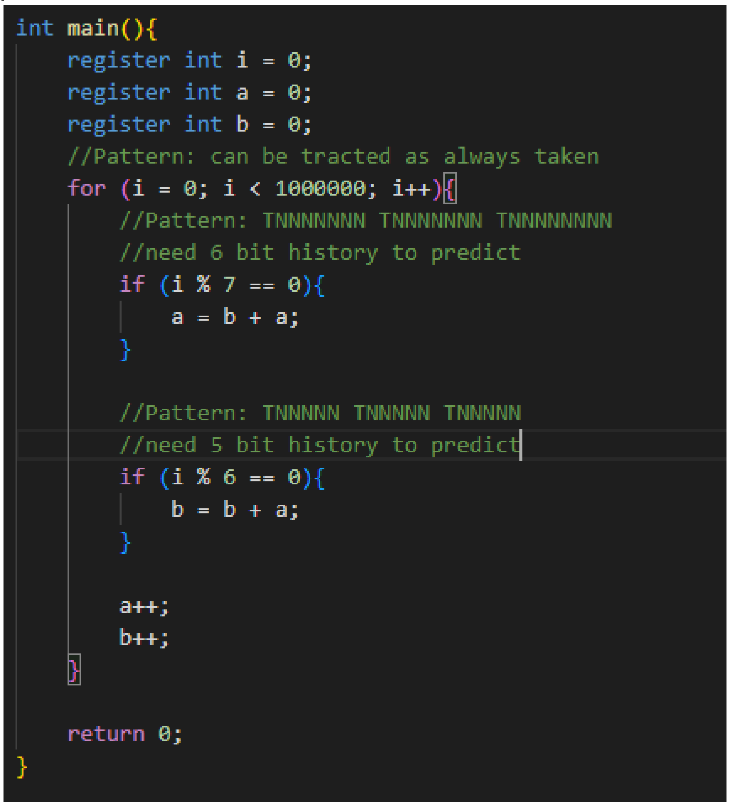
ECE 552 Lab 2 Report

Francis Guo 1005931397 Rongbo Zhang 1005273393

For our micro benchmark, we have designed a 3 branch instruction of the C code as shown. It has a always taken branch from the loop and two if statements create the pattern of 6 and pattern of 8. The code below shows the details of the microbenchmark.



```c
int main(){
    register int i = 0;
    register int a = 0;
    register int b = 0;
    //Pattern: can be tracted as always taken
    for (i = 0; i < 1000000; i++){
        //Pattern: TNNNNNNN TNNNNNNN TNNNNNNN
        //need 6 bit history to predict
        if (i % 7 == 0){
            a = b + a;
        }

        //Pattern: TNNNNN TNNNNN TNNNNN
        //need 5 bit history to predict
        if (i % 6 == 0){
            b = b + a;
        }

        a++;
        b++;
    }

    return 0;
}
```

For the compile flag, we use -O0 for the compile flag to avoid the compiler optimizing the unused variables and operations away.
For the performance of each predictor(MPKI):

	2bit-sat	2-level	open-end
astar	24.64	11.903	5.213
bwaves	7.879	7.146	1.787
bzip2	8.167	8.651	8.041
gcc	21.067	14.824	3.241
gromacs	9.074	7.484	5.657
hmmer	13.567	14.872	11.894
mcf	24.502	13.494	10.467
soplex	7.118	6.819	4.904
Average	14.501	10.694	6.400
microbench mark	8.840	4.095	4.104

For the open-end predictor, we have implemented a TAGE branch predictor, which has 7 banks of 1K entries. Each entry consists of a 11 bits tag, 3 bit prediction counter and a 2 bit usable counter. All banks will use the PC and different length of branch history for indexing the entry of the bank and calculate tags. So, the total size need for the open end predictor is:
Banks = 2 byte x 1024 x7 = 14K bytes.
We have used a 8 bit saturation branch predictor to get the base prediction for the TAGE prediction. And for the banks in the TAGE predictor, they will get predictions based on different length of branch history and PC to get index and tag for different banks. And we will take the result as the prediction from the longest history bank.
For the 8 bit base predictor, we used 256 entries for the base predictor. The size need for the 8 bit saturation base predictor is: 1 byte x 512 = 256 byte.
To recorde the history, we used a global history table for storage of the branch history with 128 bits of history, this will need 16 bytes of space.
Total space needed is : 14,608 bytes.（116,864 bits, still under the limit）
We have referenced the design from https://jilp.org/vol7/v7paper10.pdf.

In terms of the size, power and latency of the predictor.
For the Two-Level predictor:

- BHT
  The block size is 1 byte，each entry is 6 bits of history. Total space: 512byte.
  Type ram.

Area	Access Latency	Leakage Power
0.0391 x 0.0269 mm	0.1635 ns	0.195006 mw

- PHT
  The block size is 1 byte. 8 entry indexed by pc, and 64 entry indexed by history. Each entry is 2 bytes. Total 512 byte.
  Type ram..

Area	Access Latency	Leakage Power
0.0391 x 0.0269mm	0.163585ns	0.195006mw

For the open-end predictor:

- 8 bit base predictor:
  The block size is 1 byte. 256 entries indexed by c, and each entry is 1 byte of history.Total size 256 byte.
  type ram.

Area	Access Latency	Leakage Power
0.0421 x 0.0142mm	0.147232ns	0.0993983mw

- Banks:
  7 parallel banks. Each bank has 1024 entries. Each entry is 2 bytes. total 14K bytes.
  type cash.

Area	Access Latency	Leakage Power
0.1173 x 0.2818mm	0.543406ns	7.47617 mw