








Assignment			
Project name	SA1		
Document ref			
Version			
Release date			
Author	Wu Fei		
Classification	[Document classification]		
Distribution List	[Distribution list]		
Approved by	Name	Signature	Date



Ming Hsieh Department of Electrical Engineering
University of Southern California, Los Angeles, CA 90089
Fall 2022
EE 557

Simulation assignment 1

Submit this report and the coding files as following:

 AllTaken	9/26/2022 11:45 AM	C Header Sourc...
 bimodal	9/26/2022 11:38 AM	C Header Sourc...
 bpred	9/26/2022 11:49 AM	C++ Source File
 Correlated	9/26/2022 1:27 PM	C Header Sourc...
 makefile	9/16/2021 11:52 PM	文件
 makefile.rules	9/16/2021 11:52 PM	RULES 文件
 TwoGlobal	9/26/2022 11:38 AM	C Header Sourc...

1.1 Ways for prediction

To access the requirement, I add some .H files to implement the other three ways of prediction. The main changes are about the buffer size change as the requirement, pointer renaming, Instruction function change base on the prediction methods, HistAddress function to change the return of the array of the history addresses and condbranch function to claim the SBP implementation and the conditions to take the prediction.

In the .cpp file, I call all the four methods and generate the results like the format provided. Some of them show blow while some show in the submitted coding.

1.1.1 Always taken

CondBranch function

```
inline VOID ALLTAKEN::CondBranch(ALLTAKEN *at, INT8 * branchHistory,
VOID * ip, BOOL taken)
{
    INT8 history;

    at->_references++;

    history = *branchHistory & 0x3;
    //at->_predicts += (taken && history >= 2);
    //at->_predicts += (!taken && history <= 1);
    at->_predicts += (taken); // instead of the above conditional counting,
always taken takes all the predictions

    INT8 delta = 0;
    if (taken && history < 3)
    {
        delta = 1;
    }
    if (!taken && history > 0)
    {
        delta = -1;
    }
}
```

```

    }

    *branchHistory += delta;
}

```

1.1.2 2-bit Global

CondBranch function

```

inline VOID TWOGLOBAL::CondBranch(TWOGLOBAL *tg, INT8 * branchHistory,
VOID * ip, BOOL taken)
{
    INT8 history;

    tg->_references++;
    //2bit saturating branch predictor
    history = *branchHistory & 0x3; //taking the last two significant bits
    tg->_predicts += (taken && history >= 2); // 10 11
    tg->_predicts += (!taken && history <= 1); // 00 01

    INT8 delta = 0;
    if (taken && history < 3)
    {
        delta = 1;
    }
    if (!taken && history > 0)
    {
        delta = -1;
    }

    *branchHistory += delta;
}

```

1.1.3 2-bit Bimodal

Using the same SBP as the 2-bit global predictor while change the value of the buffer size as:

```

enum
{
    //TABLESIZE = 4096,
    TABLESIZE = 1, // do not consider pridicted history as the index of
the branch history buffer
    BUNDLESIZE = 32, //For bimodal predictor, we use the last 5 bits of
branch PC to index the branch prediction buffer (BPB).
}

```

```
MAXTHREADS = 100  
};
```

CondBranch function

```
inline VOID BIMODAL::CondBranch(BIMODAL *bm, INT8 * branchHistory,  
VOID * ip, BOOL taken)  
{  
    INT8 history;  
  
    bm->_references++;  
    //2bit saturating branch predictor  
    history = *branchHistory & 0x3; //taking the last two significant bits  
    bm->_predicts += (taken && history >= 2);  
    bm->_predicts += (!taken && history <= 1);  
  
    INT8 delta = 0;  
    if (taken && history < 3)  
    {  
        delta = 1;  
    }  
    if (!taken && history > 0)  
    {  
        delta = -1;  
    }  
  
    *branchHistory += delta;  
}
```

1.1.4 2-bit Correlated

Implement the required buffer size for the Correlated predictor:

```
private:  
    enum  
    {  
        //TABLESIZE = 4096,  
        TABLESIZE = 16, //is concatenated to the right side of the global  
branch history register (4 bits) to form total 8 bits and then index the BPB  
        BUNDLESIZE = 16, //For correlated predictor, the least significant 4  
bits of branch PC  
        MAXTHREADS = 100  
    };
```

```

inline INT8 * CORRELATED::HistAddress(INS ins, CORRELATED *cp)
{
    ADDRINT ip = INS_Address(ins);
    //return &(_branchHistory[(ip / BUNDLE_SIZE) % TABLE_SIZE][ip % BUNDLE_SIZE]);
    return &(_branchHistory[cp->_GlobalBranchHistory & 0xf]/*using as the last
four recording of previous branch prediction*/[ip % BUNDLE_SIZE]);
}

```

CondBranch function

```

inline VOID CORRELATED::CondBranch(CORRELATED *cp, INT8 * branchHistory,
VOID * ip, BOOL taken)
{
    INT8 history;

    cp->_references++;
    // For correlated predictor, the least significant 4 bits of branch PC
    if(taken){
        cp->_GlobalBranchHistory = (((cp->_GlobalBranchHistory) & 0xf) * 2) +
1;
    }
    else{
        cp->_GlobalBranchHistory = (((cp->_GlobalBranchHistory) & 0xf) * 2);
    }
    //the same SBP as the 2bit global prediction
    history = *branchHistory & 0x3;
    cp->_predicts += (taken && history >= 2);
    cp->_predicts += (!taken && history <= 1);

    INT8 delta = 0;
    if (taken && history < 3)
    {
        delta = 1;
    }
    if (!taken && history > 0)
    {
        delta = -1;
    }

    *branchHistory += delta;
}

```

1.2 Result

1.2.1 Execution

```
ee557@ee557:~/Desktop/p1$ ls
AllTaken.H bimodal.H bpred.cpp bpred.out Correlated.H makefile makefile.rules obj-intel64 TwoGlobal.H
ee557@ee557:~/Desktop/p1$ cd obj-intel64/
ee557@ee557:~/Desktop/p1/obj-intel64$ ls
bpred.o bpred.so
ee557@ee557:~/Desktop/p1/obj-intel64$ pin -t bpred.so -- tar -czpf /home/ee557/Desktop/p0/pin.tar.gz /home/ee5
57/Desktop/p0/pin-3.17-
tar: Removing leading '/' from member names
tar: /home/ee557/Desktop/p0/pin-3.17-: Cannot stat: No such file or directory
tar: Exiting with failure status due to previous errors
ee557@ee557:~/Desktop/p1/obj-intel64$ pin -t bpred.so -- tar -czpf /home/ee557/Desktop/p0/pin.tar.gz /home/ee5
57/Desktop/p0/pin-3.17-98314-g0c048d619-gcc-linux
tar: Removing leading '/' from member names
ee557@ee557:~/Desktop/p1/obj-intel64$ ls
bpred.o bpred.out bpred.so
ee557@ee557:~/Desktop/p1/obj-intel64$ vi bpred.out
ee557@ee557:~/Desktop/p1/obj-intel64$
```

1.2.2 Bpred.out

```
ee557@ee557: ~/Desktop/p1/obj-intel64
File Edit View Search Terminal Help
===== USC EE557 Fall 2022 - Project 1 =====
Name: Fei Wu
Email: wufei@usc.edu

Always Taken Predictor
Total Prediction: 4.55764e+07
Prediction Rate: 0.464192

2-bit Global Predictor
Total Prediction: 4.55764e+07
Prediction Rate: 0.503463

Bimodal Predictor
Total Prediction: 4.55764e+07
Prediction Rate: 0.776629

Correlated Predictor
Total Prediction: 4.55764e+07
Prediction Rate: 0.899088
=====
```

Showing as the result, it can say that the prediction rate growing up as we using more complicate and delicate methods and more memory to store and manage the prediction history.