**Computer Architecture 2013**

**Appendix D**

**Branch Prediction**

Since a branch instruction is evaluated at the ID stage, there is at most one cycle penalty for flushing the IF stage instruction. Therefore, we must predict the branch target in the IF stage and get the target address at the same cycle if we want gain some benefits.

We show a data path example with the BHT and the Branch Unit in the IF stage. Please refer to “Appendix E” to build the IF stage simulation. There are two outputs from the Branch Unit, one is a one-bit signal indicating whether the instruction is a branch instruction, and the other one is the target address calculated by PC-relative addressing as discussed in the textbook. The output of BHT is a one-bit signal indicating whether the branch prediction is taken (signal =1 for taken and vise versa.)

Note that at the ID stage we ~~also~~ need to verify the correctness of prediction. In case of wrong prediction, the wrong instruction is flushed and the correct instruction is fetched in. Since the IF.Flush from the ID stage is set to 1 if the branch is taken and vice versa, we can compare the IF.Flush with the result of prediction to check if the prediction is correct or wrong. Then the result is used to set the PC and determine whether the IF/ID register should be flushed. Additionally, the BHT should not be update when a branch instruction stall in ID stage, because it can not update BHT correctly.

Address(PC)

Memory

Taken/Not Taken

BHT

0x200824

0x 00A1325C

Is a branch instruction?

Branch Unit

Target Address

Fig. 1: An illustration of Branch Prediction

|  |  |
| --- | --- |
| Index | Prediction bit |
| 0000 | 00 |
| 0001 | 10 |
| 0002 | 10 |
| …… | …… |
| 1111 | 10(Taken) |

Fig. 2: An BHT example.

Taken

Not Taken

Taken

Taken

Not Taken

Taken

Not Taken

Not Taken

Fig. 3: A state diagram for a 2-bit branch prediction scheme.