体系结构 Lab4

PB19000015 贾欣宇

2022 年 4 月 27 日

目录

1 实验目的

- 实现 BTB（Branch Target Buffer）和 BHT（Branch History Table）两种动态分支预测器；
- 体会动态分支预测对流水线性能的影响.

2 实验内容

2.1 分支预测实现

根据分支预测方式的选择，分支预测信号 pr 更新如下：

```
1 if(Predict_en == `BTB)
2     pr = pr_en & pr_btb;
3 else if(Predict_en == `BHT)
4     pr = pr_en & pr_bht;
5 else
6     pr = 0;
```

Listing 1: 更新分支预测信号 pr

其中 pr_en 为 BTB 命中信号，pr_btb 为 BTB 预测结果，pr_bht 为 BHT 预测结果. 当使用 BHT 进行分支预测时，BTB 的预测结果将不被采纳.

分支预测信号 pr 在生成 NPC 和处理 Hazard 中的作用如下：

```
if (br && ~prE) NPC = br_target;
else if (~br && prE) NPC = PCE_4;
else if (jalr) NPC = {jalr_target[31:1], 1'b0};
else if (jal) NPC = jal_target;
else if (prF) NPC = pr_target;
else NPC = PC;
```

Listing 2: 生成 NPC

```
if ((br && ~prE) || (~br && prE) || jalr)
begin
    bubbleF = 0; flushF = 0;
    bubbleD = 0; flushD = 1;
    bubbleE = 0; flushE = 1;
    bubbleW = 0; flushW = 0;
end
```

Listing 3: 处理 Hazard

分支预测的具体实现如下.

2.1.1 BTB

- BTB 是一个 1 bit 预测器，如果上次这条分支指令跳转，那么这次它也跳转；如果上次不跳，那么这次也不跳；
- BTB 保存当前地址、目标地址、有效位和跳转状态，类似于直接映射的 Cache；
- BTB 内容读取在 IF 阶段进行，根据 BTB 是否命中及 BTB 跳转状态信息决定下一条指令的地址；
- BTB 根据实际是否跳转和 IF 阶段是否命中的信息，在 EX 阶段修改保存的数据.

```
pr_target = 0;
pr_en = 0;
pr_btb = 0;
for(integer i = 0; i < 64; i++) begin
    if(PC_IF == prTag[i] && prValid[i]) begin
        pr_target = prTarget[i];
        pr_en = 1;
        pr_btb = prBtb[i];
        break;
    end
```

```
11 end
```

Listing 4: 生成 BTB 预测结果

以上是 BTB 预测结果的生成，该过程在 IF 阶段一周期内完成.

```
1  if(opE == `B_TYPE) begin
2      integer i;
3      for(i = 0; i < 64; i++) begin
4          if(PC_EX == prTag[i]) begin
5              prBtb[i] <= br;
6              break;
7          end
8      end
9      if(i == 64 && br) begin
10         prTag[pointer] <= PC_EX;
11         prTarget[pointer] <= br_target;
12         prValid[pointer] <= 1;
13         prBtb[pointer] <= 1;
14         pointer <= pointer + 1;
15     end
16 end
```

Listing 5: 更新 BTB 保存数据

以上是 BTB 保存数据的更新，该过程在 EX 阶段一周期内完成. 由于 BTB 存储空间有限，仅在未记录的跳转指令发生跳转时在 BTB 中加入该指令的记录，如此并不会影响分支预测的结果.

2.1.2 BHT

- BHT 存储 2 bit 预测数据，在 IF 阶段进行读取；
- BHT 实现一个状态机，在 EX 阶段根据状态机更新预测数据；
- BHT 每项存储的数据量很小，因此其存储项目数可以远大于 BTB.

```
1  assign pr_bht = prBht[tagIF][1];
```

Listing 6: 生成 BHT 预测结果

BHT 使用指令 [9:2] 位对应存储地址的预测数据的高位作为分支预测结果.

```
1  if(opE == `B_TYPE) begin
2      if(br) begin
3          prBht[tagEX] <= (prBht[tagEX] == 2'b11) ? 2'b11 : prBht[tagEX] + 2'b01;
4      end
5      else begin
```

```
6           prBht[tagEX] <= (prBht[tagEX] == 2'b00) ? 2'b00 : prBht[tagEX] - 2'b01;
7       end
8   end
```

Listing 7: 更新 BHT 保存数据

BHT 实现了如图 1 的状态机来更新预测数据（自 01 状态即 weakly not taken 状态启动）.

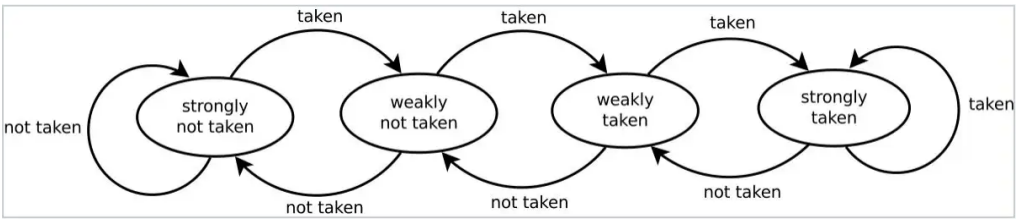

图 1: BHT 状态机

2.2 实验分析

分别对四组样例（BTB、BHT、快速排序、矩阵乘法）使用 BTB 和 BHT 分支预测方案进行测试，并测试一组无预测情况作为对比，结果如表 1.

样例	BTB	BHT	快速排序	矩阵乘法
分支收益（周期）	2	2	2	2
分支代价（周期）	2	2	2	2
总周期数（无预测）	511	537	35,616	71,504
总周期数（BTB）	315	383	36,116	63,900
总周期数（BHT）	315	365	34,586	63,360
周期数差值（BTB）	196	154	-500	7,604
周期数差值（BHT）	196	172	1,030	8,144
分支指令数	101	110	6,633	4,624
分支指令执行数	100	99	1,698	4,350
预测正确数（BTB）	99	88	4,685	4,076
预测正确数（BHT）	99	97	5,450	4,346
预测错误数（BTB）	2	22	1,948	548
预测错误数（BHT）	2	13	1,183	278

表 1: 实验结果统计

由于测试次数很多，选取 BTB 样例的无预测测试和矩阵乘法样例的 BHT 测试结果如图 2、图 3 所示. 由测试结果可知：

- 分支收益和分支代价与具体样例无关，只与 CPU 设计有关；

- 若预测错误次数过多，使用分支预测时的性能可能反而劣于未使用分支预测时；
- 对每种分支预测方案，周期数差值 = (分支指令执行数 − 预测错误数) ∗ 分支代价；
- 对 BTB 方案，预测错误数 = 循环个数 ∗ 2，对 BHT 方案，预测错误数 = 循环个数 + 循环种数；
- BHT 方案预测错误数较少、运行总周期数较少、性能较优.

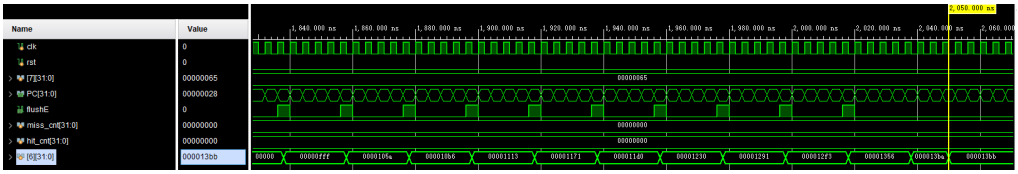

图 2: BHT 状态机

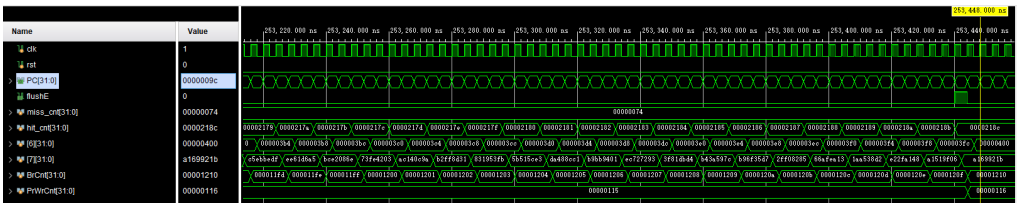

图 3: BHT 状态机

3 实验总结

本次实验中完成了 BTB 和 BHT 分支预测器的设计，并通过测试体会了二者对流水线性能的影响，加深了对分支预测的理解.