# Lab4

## 实验目的

在已有的RISCVcore上实现

1. BTB分支预测
2. 在BTB的基础上实现BHT预测

## 实验环境与工具

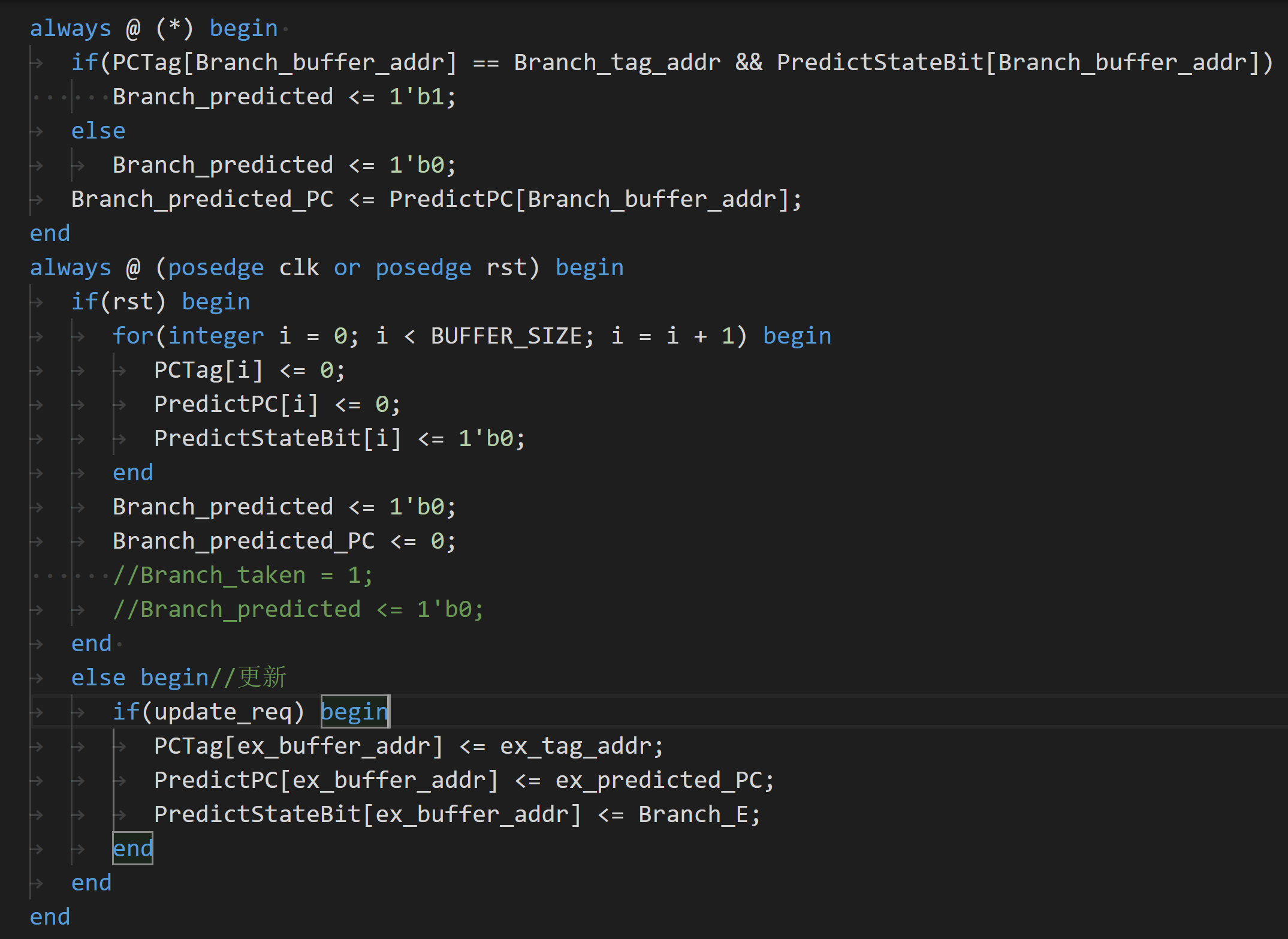
使用windows10下的verilog仿真工具

## 实验内容与过程

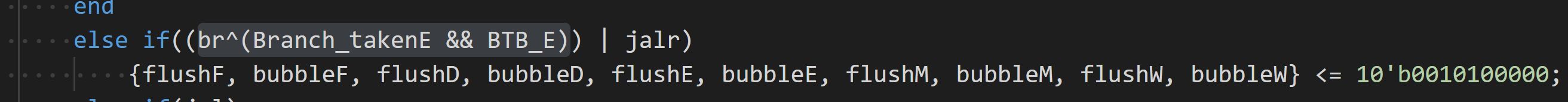
BTB主要代码如下。

1. 当输入的PC在BTB表中且对应的预测位为1，则预测这次为跳转，并输出跳转的PC。
2. 当预测结果与跳转结果不一致的时候更新BTB表



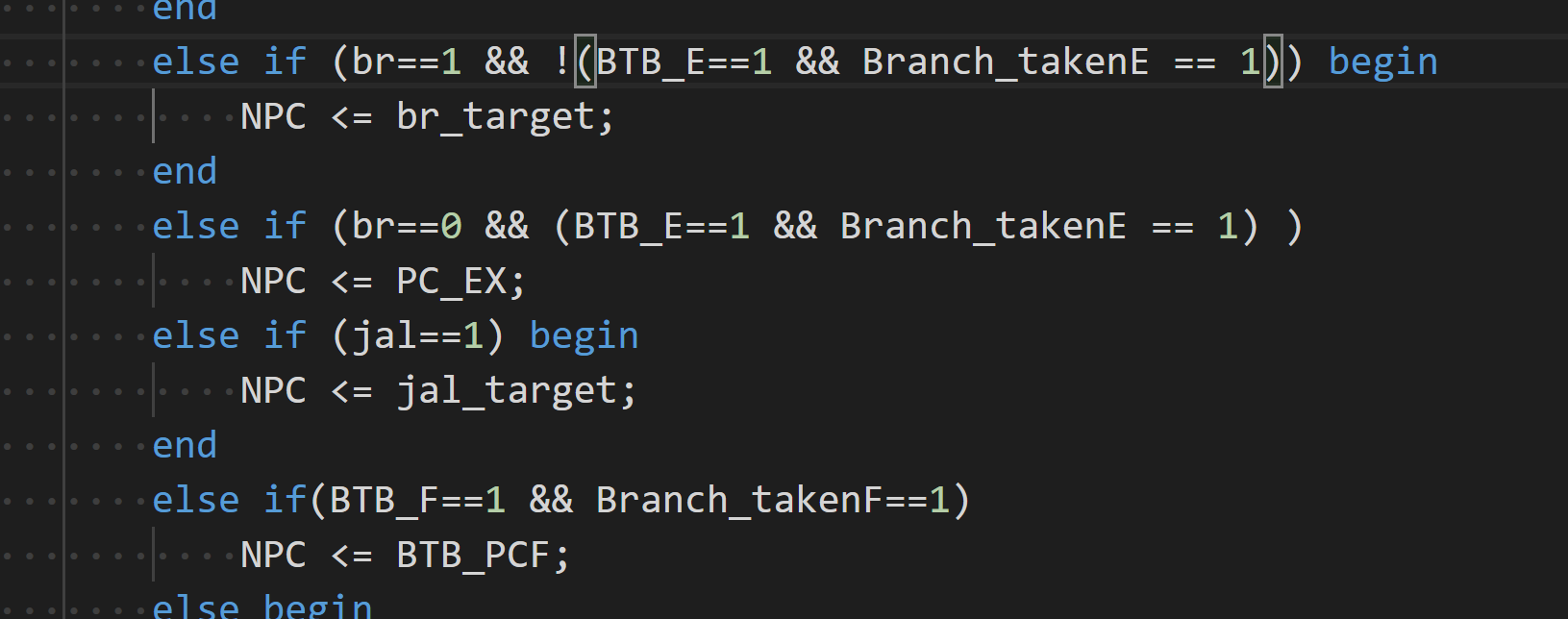


Hazard中，若是预测结果与实际跳转结果不一致则清空ID与EX



NPC中，

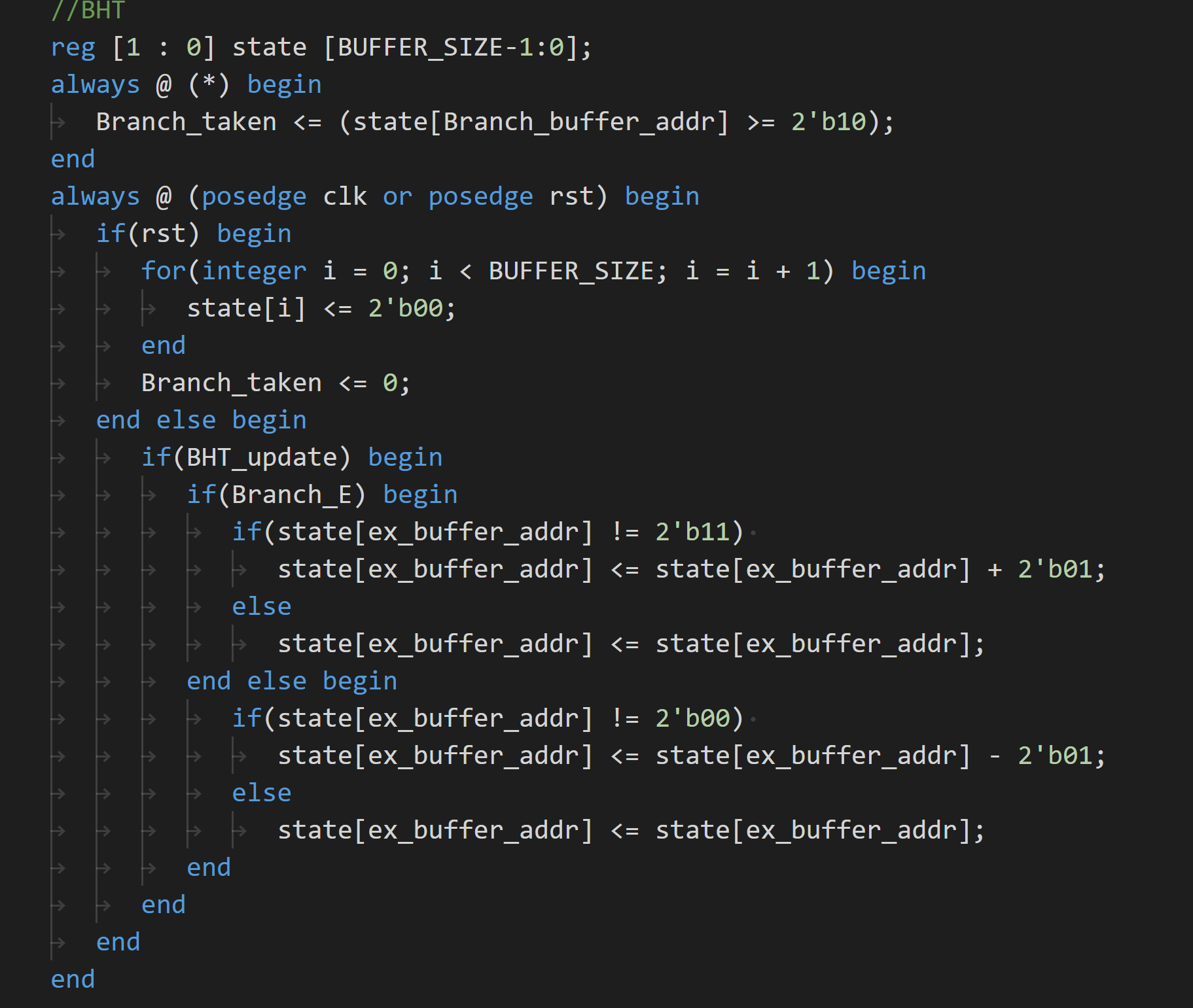
1. 若是实际跳转但之前预测不跳转，则选择实际跳转的PC。
2. 若是实际不跳转但是预测跳转，则从EX段对应指令的下一条开始执行。
3. 若是BTB预测跳转则选择预测的PC



BHT的实现参照下表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| BTB | BHT | REAL | NPC\_PRED | Flush | NPC\_REAL | BTB\_update |
| Y | Y | Y | BUF | N | BUF | N |
| Y | Y | N | BUF | Y | PC\_EX+4 | Y |
| Y | N | Y | PC\_IF+4 | Y | Br\_target | N |
| Y | N | N | PC\_IF+4 | N | PC\_EX+4 | Y |
| N | Y | Y | PC\_IF+4 | Y | Br\_target | Y |
| N | Y | N | PC\_IF+4 | N | PC\_EX+4 | N |
| N | N | Y | PC\_IF+4 | Y | Br\_target | Y |
| N | N | N | PC\_IF+4 | N | PC\_EX+4 | N |

代码如下，状态0.1表示不跳转，状态2，3表示跳转



## 实验结果

BHT的实现参照下表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| BTB | BHT | REAL | NPC\_PRED | Flush | NPC\_REAL | BTB\_update |
| Y | Y | Y | BUF | N | BUF | N |
| Y | Y | N | BUF | Y | PC\_EX+4 | Y |
| Y | N | Y | PC\_IF+4 | Y | Br\_target | N |
| Y | N | N | PC\_IF+4 | N | PC\_EX+4 | Y |
| N | Y | Y | PC\_IF+4 | Y | Br\_target | Y |
| N | Y | N | PC\_IF+4 | N | PC\_EX+4 | N |
| N | N | Y | PC\_IF+4 | Y | Br\_target | Y |
| N | N | N | PC\_IF+4 | N | PC\_EX+4 | N |

我所提交的代码为BHT，要实现BTB只需注释BHT部分并设Branch\_taken永远为1。

不预测即为预测不跳转。只需设Branch\_predict永远为0即可

实验截图在最后

BTB结果统计如下

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 策略 | 跳转指令数 | 预测成功次数 | 预测失败次数 | 用时/ns | 总周期数 | 周期数与静态的差值 |
| BTB | 101 | 99 | 2 | 1256 | 313 | 196 |
| BHT | 101 | 98 | 3 | 1264 | 315 | 198 |
| 不预测 | 101 | 1 | 100 | 2040 | 509 | -- |

BHT结果统计如下

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 策略 | 跳转指令数 | 预测成功次数 | 预测失败次数 | 用时/ns | 总周期数 | 周期数与静态的差值 |
| BTB | 110 | 88 | 22 | 1528 | 381 | 154 |
| BHT | 110 | 95 | 15 | 1476 | 367 | 168 |
| 不预测 | 110 | 11 | 99 | 2144 | 535 | -- |

矩阵乘法结果，开始时间为12ns，忽略不计

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 策略 | 跳转指令数 | 预测成功次数 | 预测失败次数 | 用时/ns | 总周期数 | 周期数与静态的差值 |
| BTB | 4623 | 4072 | 545 | 1176264 | 294065 | 7546 |
| BHT | 4623 | 4076 | 547 | 1176232 | 294057 | 7554 |
| 不预测 | 4623 | 273 | 4350 | 1206448 | 301611 | -- |

快排结果，开始时间为12ns，忽略不计

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 策略 | 跳转指令数 | 预测成功次数 | 预测失败次数 | 用时/ns | 总周期数 | 周期数与静态的差值 |
| BTB | 6614 | 4728 | 1886 | 161338 | 40333 | -956 |
| BHT | 6614 | 5495 | 1119 | 155202 | 38799 | 578 |
| 不预测 | 6614 | 5206 | 1409 | 157514 | 39377 | -- |

## 实验分析

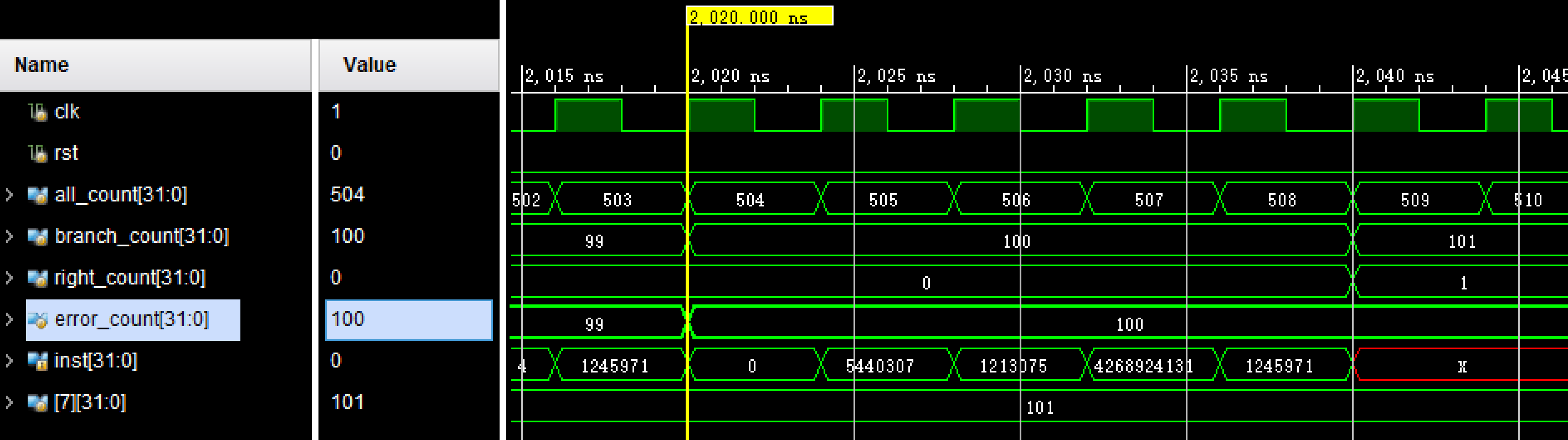
1. 若预测跳转且实际跳转，则比不预测要少两个周期，因为不需要清空ID，EX，分支收益为2cycle
2. 若预测跳转且实际未跳转，则比不预测要多两个周期，因为需要清空ID，EX，开销为2cycle
3. 从整体上看，三种策略的效率为BHT>BTB>不预测
4. btb.S中BHT策略不如BTB策略是因为其中只有一个循环
5. bht.S中BHT策略优于BTB策略是因为其中有两层循环
6. 快排中效率 BHT>不预测>BTB 是因为快排的跳转收实际数据的影响很大，没有明确规律，预测很容易失败。
7. 矩阵乘法中效率 BHT>BTB>不预测 是因为矩阵乘法有循环，较有规律，预测效果较好

## 实验小结

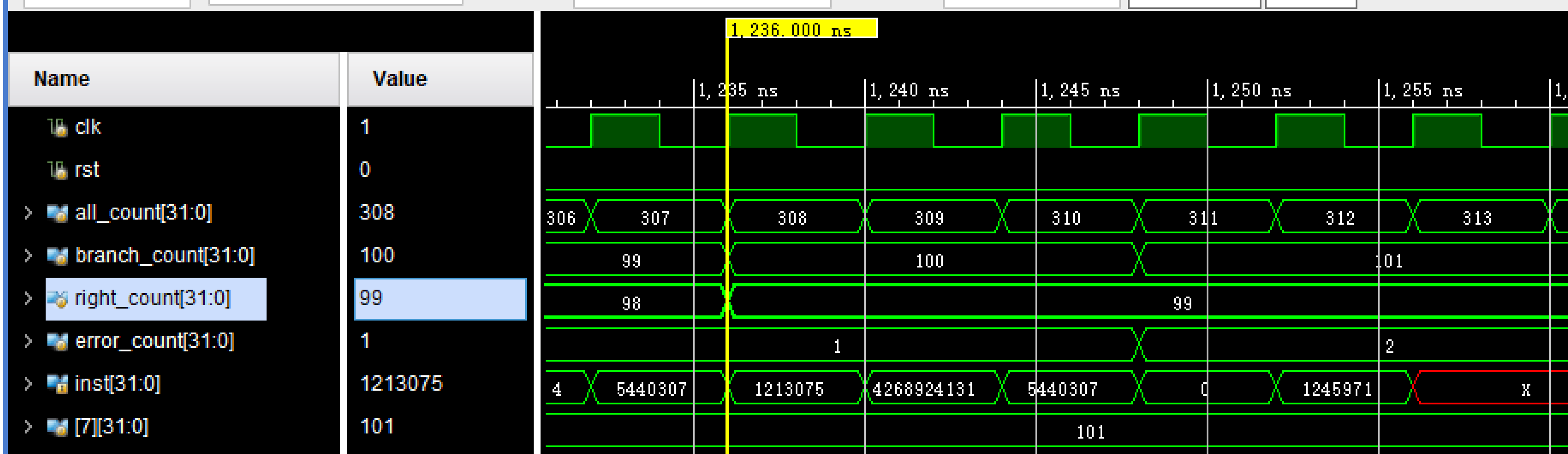
通过本次实验我初步掌握了BTB与BHT预测策略的原理，并了解了这两个策略在实际运用中的效率情况。

## 实验截图

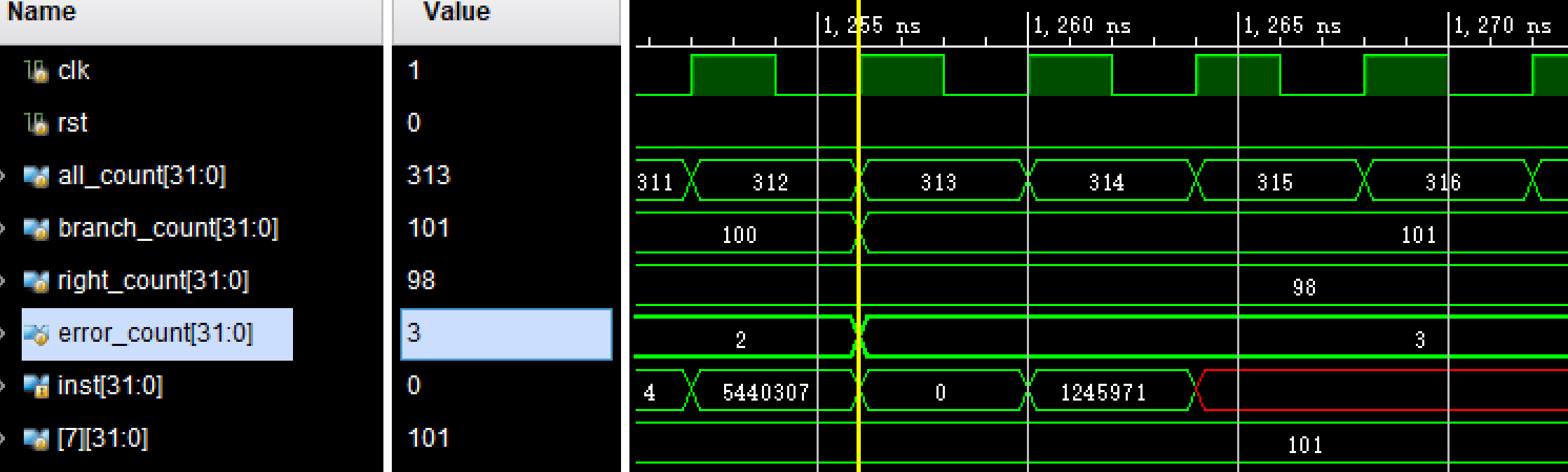
btb.S 不预测



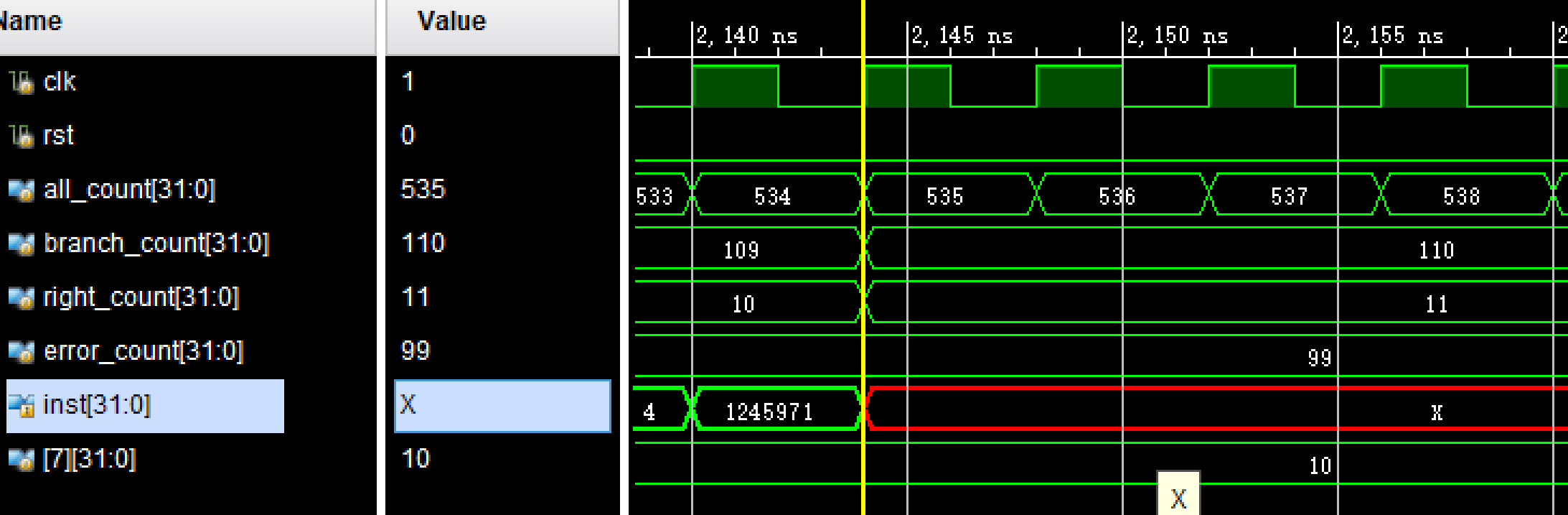
btb.S BTB策略



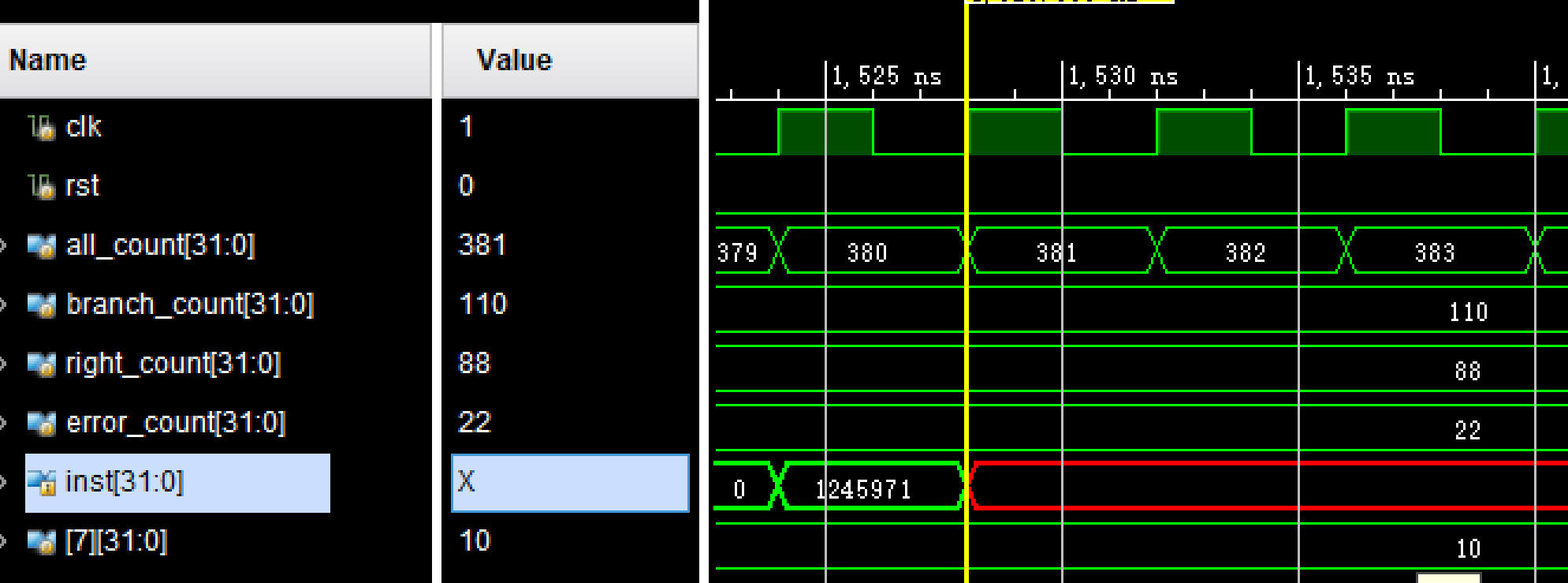
btb.S BHT策略



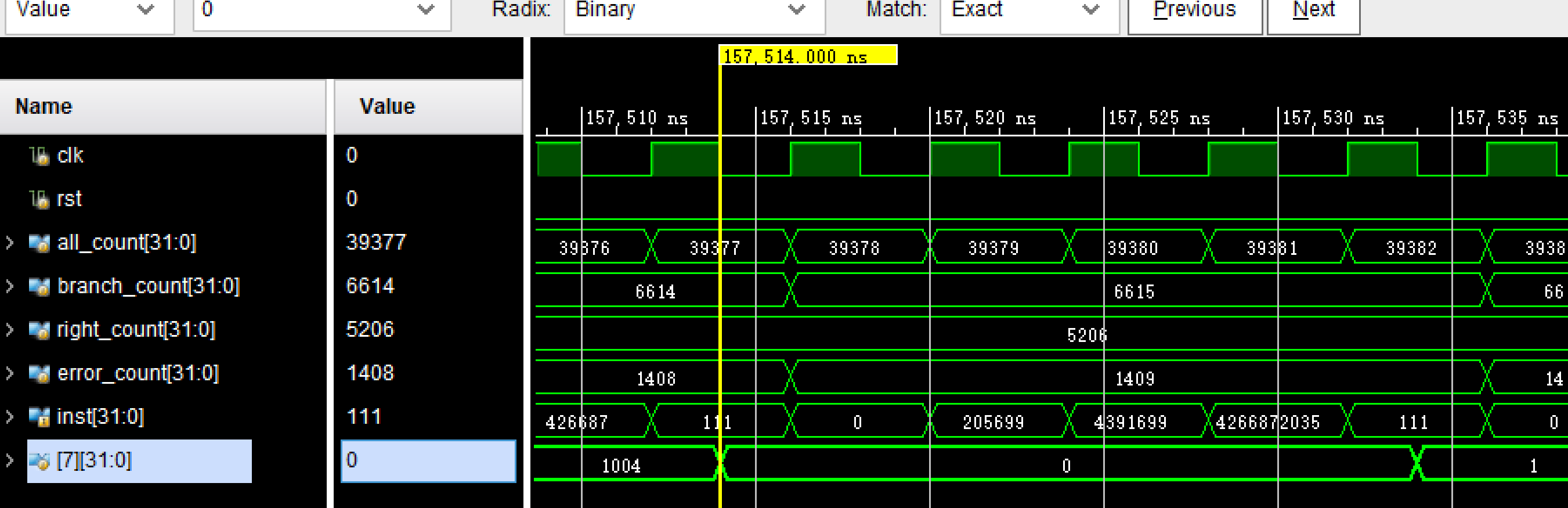
bht.S 不预测



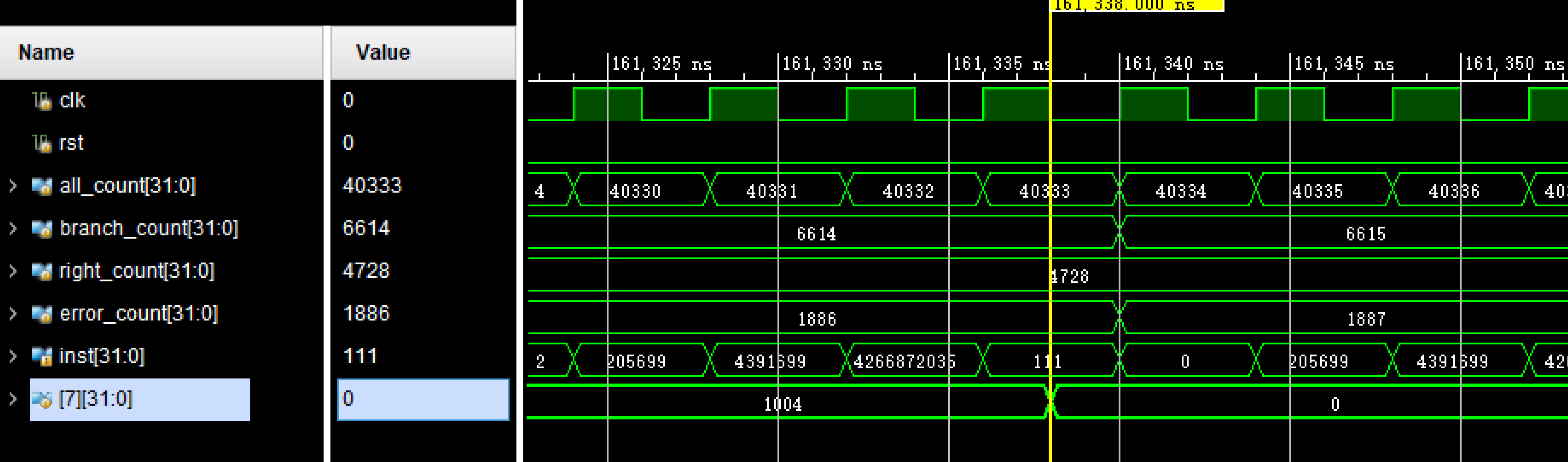
Bht.S BTB策略



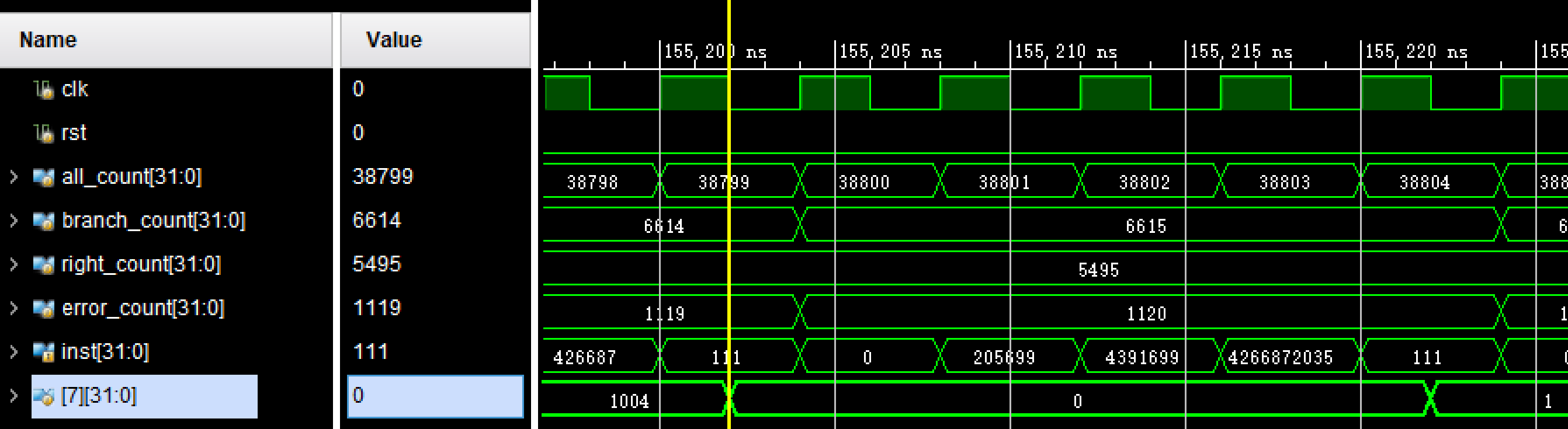
快排 不预测



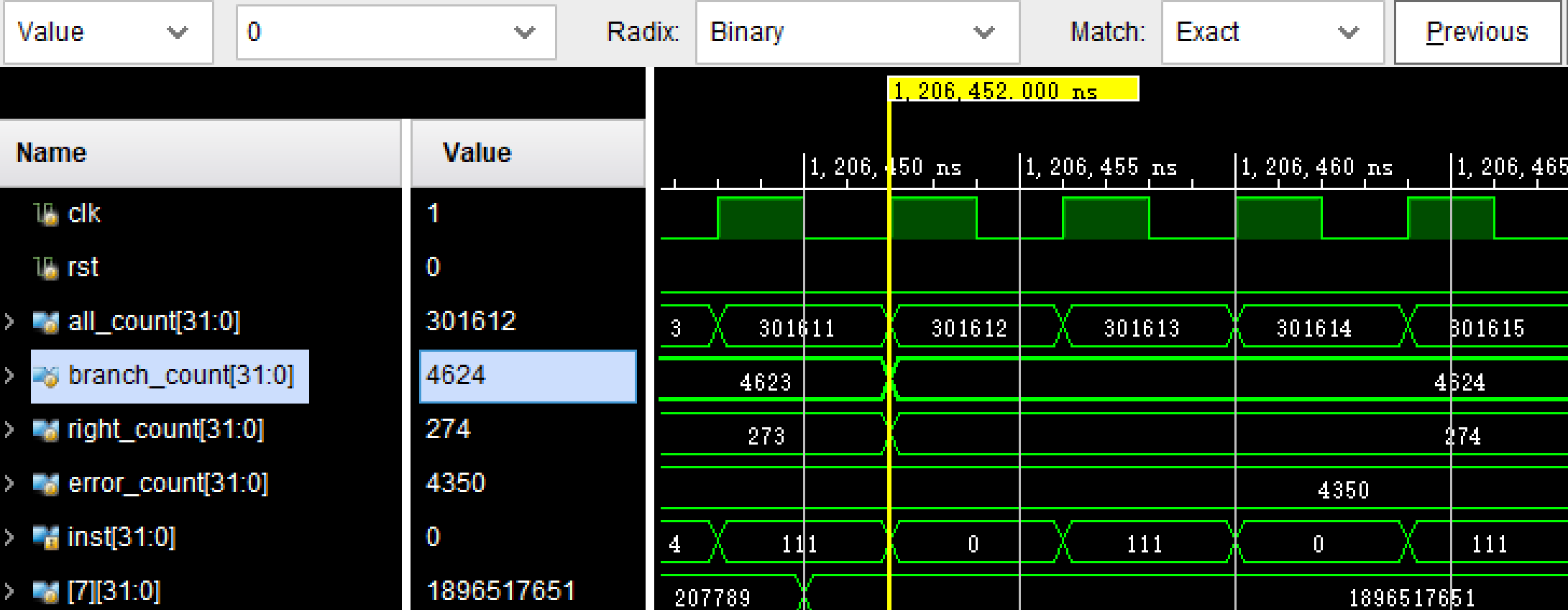
快速排序BTB策略



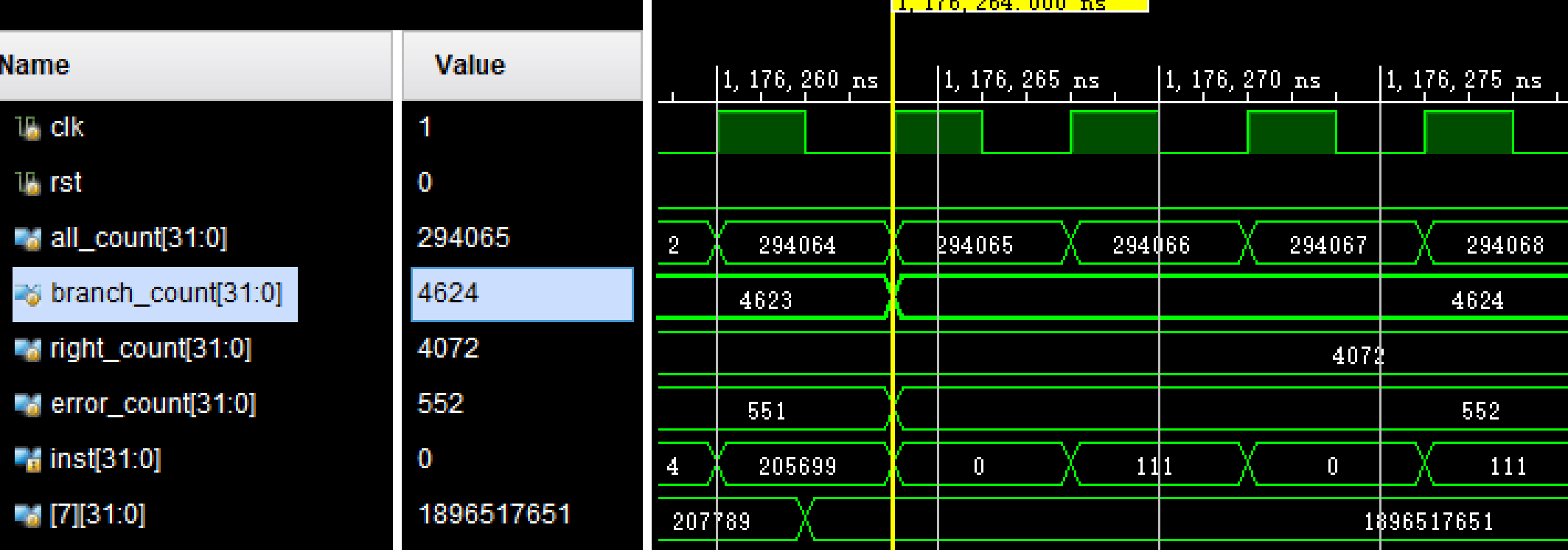
快排BHT策略



矩阵 不预测



矩阵 BTB策略



矩阵 BHT策略

