

lab4实验报告

PB17111623

范睿

补全表格

BTB	BHT	REAL	NPC_PRED	flush	NPC_REAL	BTB update
Y	Y	Y	BUF	N	br_target	N
Y	Y	N	BUF	Y	PC_EX+4	N
Y	N	Y	PC_IF+4	Y	br_target	N
Y	N	N	PC_IF+4	N	PC_EX+4	N
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

结果分析

分析分支收益和分支代价



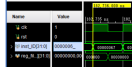



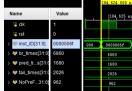
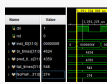


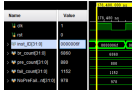
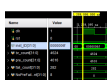
每一个分支在没有预测的情况下有2个时钟周期的代价

对于一个执行N次循环体的循环：

- 无预测的分支代价为 $2(N-1)$
- BTB预测的分支代价为4（第一次和最后一次分支都有代价，其他的分支都没有）
- BTB+BHT的分支代价为6（前两次和最后一次分支都有代价，其他的分支都没有）

对于更复杂的程序，BTB和BTB+BHT的分支代价要小于无预测的分支代价，BTB+BHT的表现要优于BTB。

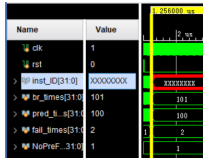
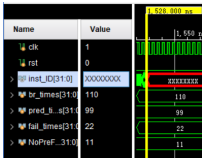
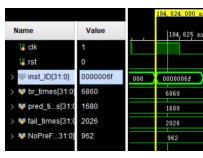
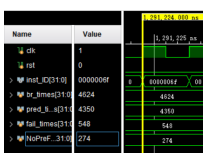
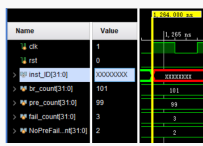
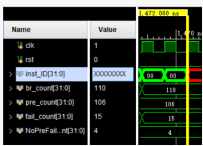
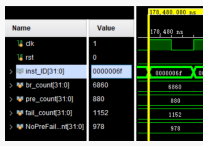
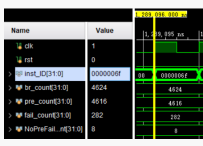
统计未使用分支预测和使用分支预测的总周期数及差值

	btb	btb 周期 数	bht	bht 周期 数	QuickSort	QS周 期数	MatMul	MatMul 周期数
无预测		510		536		45684		330412
BTB		314		382		46156		322806
BTB+BHT		316		368		44620		322274
差值（无 预测-有预 测）		196 194		174 168		-472 1064		7606 8138

- 差值一行，上方是与BTB的差值，下方是与BTB+BHT的差值
- 周期数用时间除以4得到
- btb.S和bht.S时间计算截止值inst_ID变为红色，QuiscSort.S和MalMul.S时间计算截止至第一条jal开始的那个时钟上升沿

统计分支指令数目、动态分支预测正确次数和错误次数

原图

	btb	bht	QuickSort	MatMul
BTB				
BTB+BHT				

- br_count/br_times：分支总数
- pre_count/pred_times：PC跳转至BTB表中结果的次数（预测总数）
- fail_count/fail_times：冲刷次数（包含未预测但分支跳转和预测但未跳转的次数）
- NoPreFail_count/NoPreFail_times：未预测但分支跳转的次数

因此：

- 分指数 = br_count/br_times
- 预测正确次数 = pre_count - (fail_count - NoPreFail_count)（即预测总数-预测错误总数）
- 预测错误次数 = fail_count - NoPreFail_count

数字

	btb	bht	QuickSort	MatMul
BTB	分支数: 101 预测正确次数: 99 预测错误次数: 1	110 88 11	6860 510 1170	4624 4076 274
BTB+BHT	分指数: 101/ 预测正确次数: 98 预测错误次数: 1	110 95 11	6860 706 174	4624 4342 274

对比不同策略并分析以上几点的关系

- 从程序运行周期数来看，带有分支预测的CPU要比不带有分支预测的CPU具有更少的运行周期，且BTB+BHT的程序运行周期比只有BTB的要少。
- 从预测正确和错误次数来看，不论是复杂程序还是简单程序，BTB+BHT的正确次数在整体趋势上多于只有BTB的正确次数，错误次数也都少于BTB的错误次数。BTB+BHT的整体效果优于只有BTB的程序。且在快排这样的复杂程序中，BTB+BHT的预测错误次数也在一个很低的水平。
- 通过对比两个表可以发现，预测正确次数越多，预测错误次数越少，程序用的时钟周期数越少，可见对分支进行预测通过减少分支代价来给系统性能带来优化。

实验结论

对分支进行预测可以通过减少分支代价进而对系统性能进行优化，且2-bit预测器对性能的提升要比1-bit的效果好很多。