Review Books 3.9 & 3.10

3.9用于指令传送和推测的高级技术

3.9.1提高指令提取带宽

本节内容:两种处理分支的方法;处理器将指令预测和预取功能结合在一起的方式

1.分支目标缓存/分支目标缓冲区

概念：存储着一条分支之后下一条指令的预测地址的分支预测缓存

下图给出了一个分支目标缓冲区

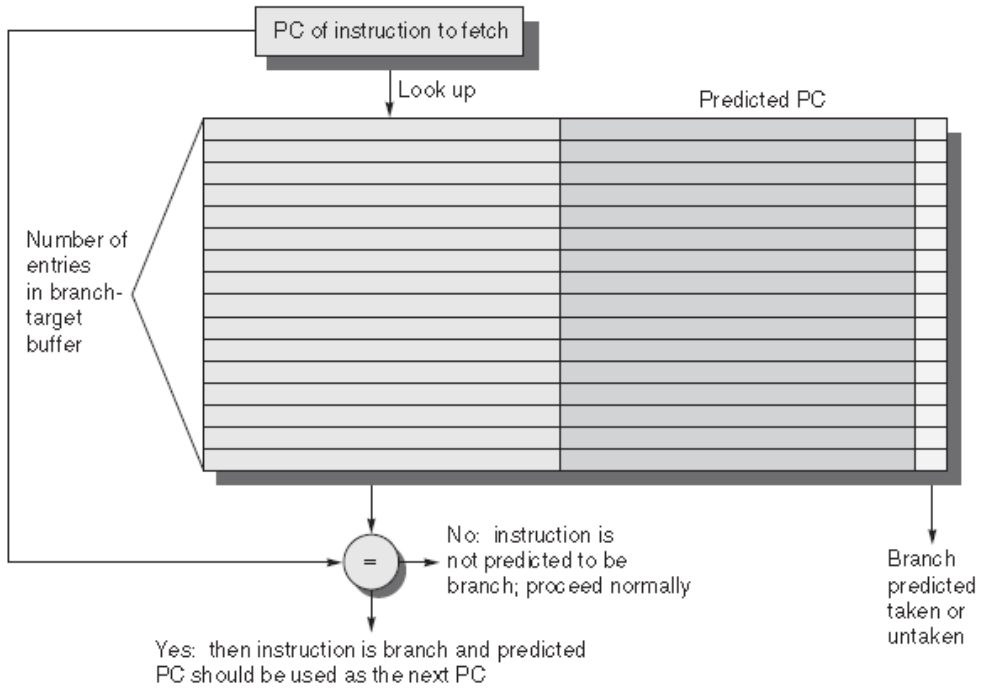

Figure 3.21 A branch-target buffer. The PC of the instruction being fetched is matched against a set of instruction addresses stored in the first column; these represent the addresses of known branches. If the PC matches one of these entries, then the instruction being fetched is a taken branch, and the second field, predicted PC, contains the prediction for the next PC after the branch. Fetching begins immediately at that address. The third field, which is optional, may be used for extra prediction state bits.

下图显示了在简单的五级流水线中使用分支目标缓冲区的步骤

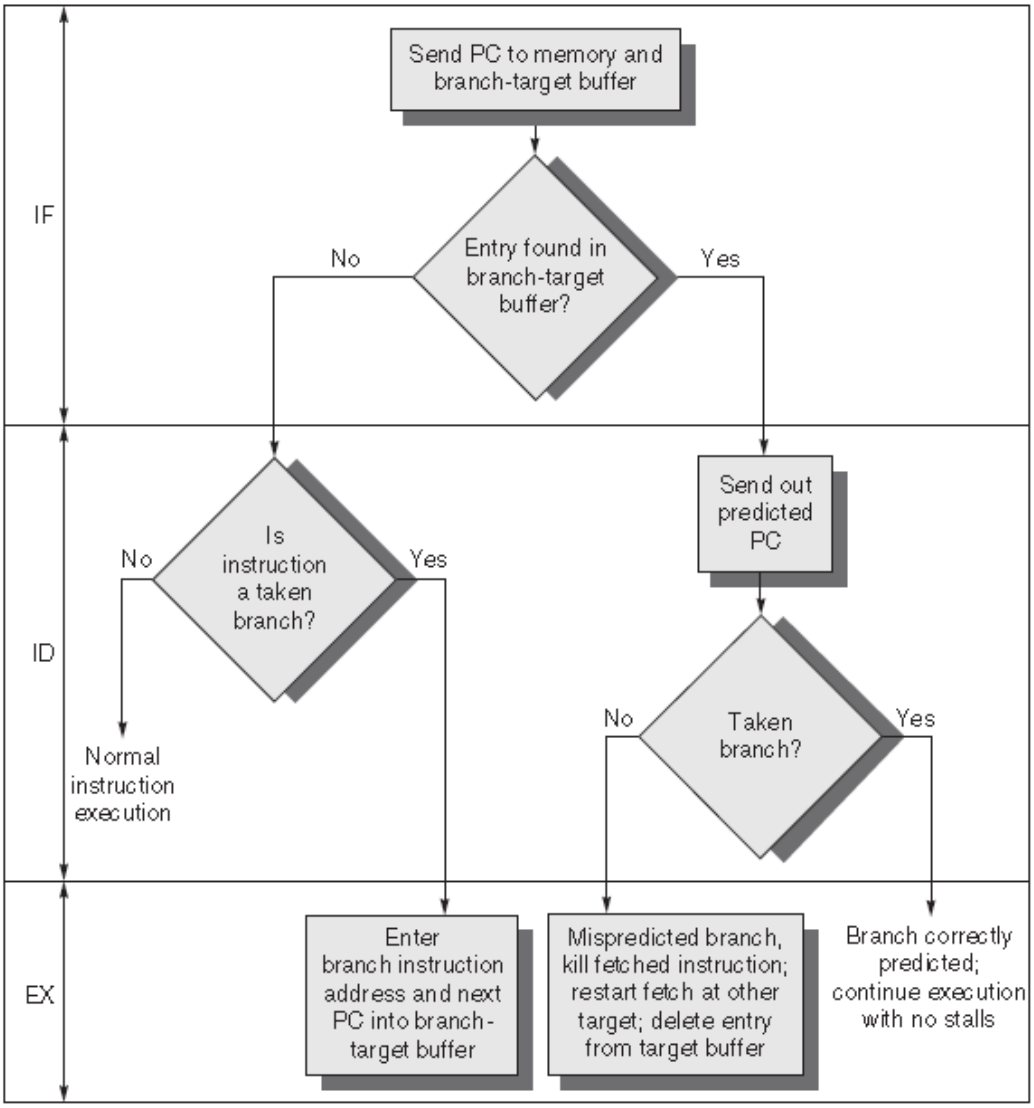

Figure 3.22 The steps involved in handling an instruction with a branch-target buffer.

分支目标缓冲区的一种变体是存储一个或多个目标指令，用于作为预测目标地址的补充或替代。

2.返回地址预测器

原因：尽管过程返回操作可以用分支目标缓存区预测，但若从多个地方调用这个进程，而且来自一个地方的多个调用在时间方面比较松散，那这种预测方法的准确性会很低。

工作方式相当于一个栈。这种结构缓存最近的返回地址：在调用时将返回地址压入栈中，在返回时弹出一个地址。

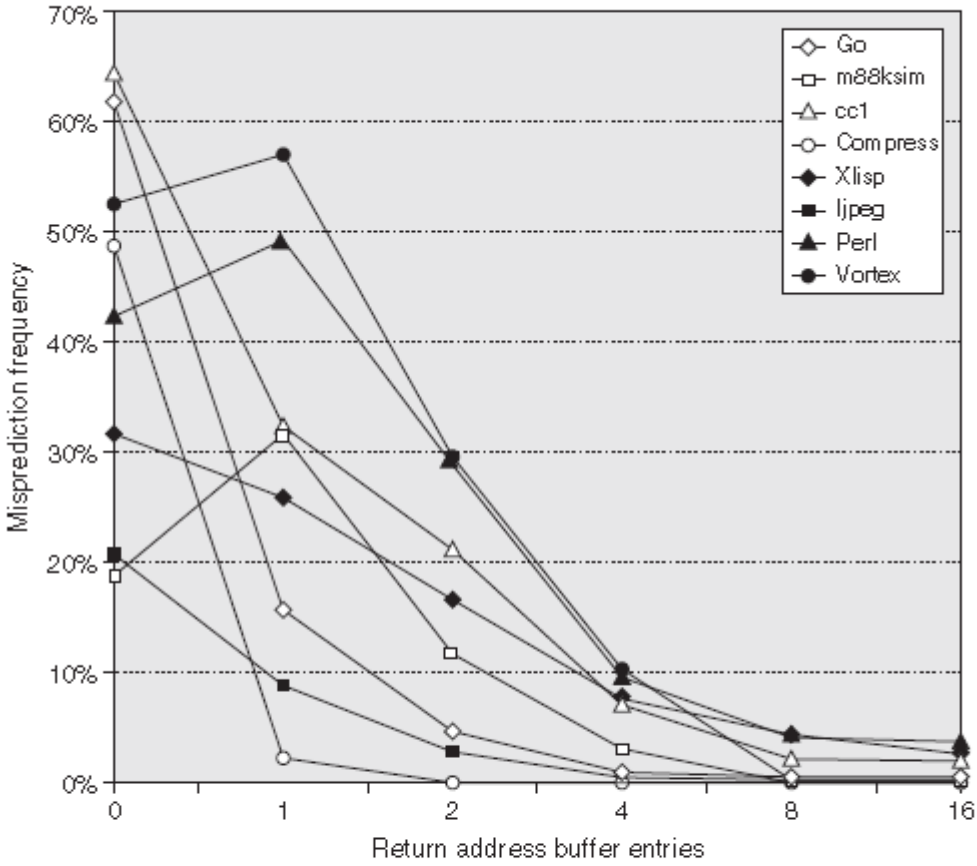

Figure 3.24 Prediction accuracy for a return address buffer operated as a stack on a number of SPEC CPU95 benchmarks. The accuracy is the fraction of return addresses predicted correctly. A buffer of 0 entries implies that the standard branch prediction is used. Since call depths are typically not large, with some exceptions, a modest buffer works well. These data come from Skadron et al. [1999] and use a fix-up mechanism to prevent corruption of the cached return addresses.

3.集成指令提取单元

满足多发射处理器的要求。

包括功能：

（1）集成分支预测

（2）指令预取

（3）指令存储器访问与缓存

3.9.2推测：实现问题与扩展

本节内容：涉及推测设计权衡的四个问题；值推测

1.推测支持：存储器重命名与重排序缓冲区

与ROB（重排序缓冲区）相比，重命名方法的一个优点是简化了指令提交过程。

2.推测的代价

推测能够尽早发现那些本来会使流水线停顿的事件，比如缓存缺失。但推测不是毫无代价的，它需要时间和能量，错误预测的恢复过程会进一步降低性能。

3.多分支预测

三种情况：（1）分支出现频率非常高；（2）分支高度汇集；（3）功能单元中的延迟很长

对多个分支进行推测会使推测恢复过程稍微变得复杂，但在其他方面比较简单。

4.推测与能耗效率的挑战

推测消耗更多性能的情况：（1）对某些指令进行了推测，但不需要他们的结果；（2）撤销推测，恢复处理器的状态，以便在适当的地址处继续执行。

对于整数应用程序来说，推测的能耗效率不会很高。

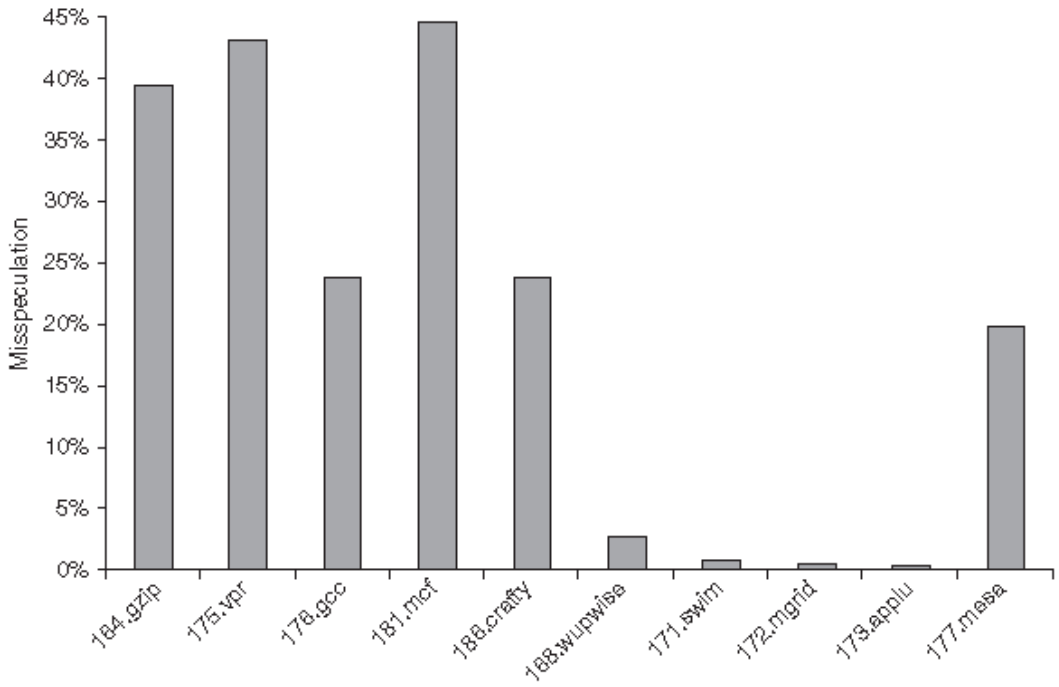

Figure 3.25 The fraction of instructions that are executed as a result of misspeculation is typically much higher for integer programs (the first five) versus FP programs (the last five).

5.值预测（成果缺乏吸引力，未在实际处理器中应用）

一种提高程序中可用ILP数目的技术，它尝试预测一条指令可能生成的值。

相关的较早思想（**地址别名预测**）得到应用，这种技术用来预测两个存储指令或一个载入指令与一个存储指令是否引用同一存储器地址。

3.10ILP局限性的研究

3.10.1硬件模型

完美处理器的假设：

（1）无线寄存器重命名

(2) 完美分支预测

(3) 完美跳转预测

(4) 完美存储器地址别名分析

(5) 完美缓存

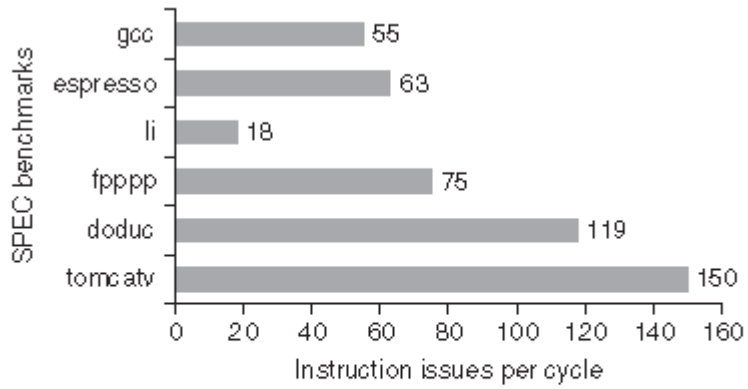

Figure 3.26 ILP available in a perfect processor for six of the SPEC92 benchmarks. The first three programs are integer programs, and the last three are floating-point programs. The floating-point programs are loop intensive and have large amounts of loop-level parallelism.

3.10.2可实现处理器上ILP的局限性

本节将研究一些处理器的性能。

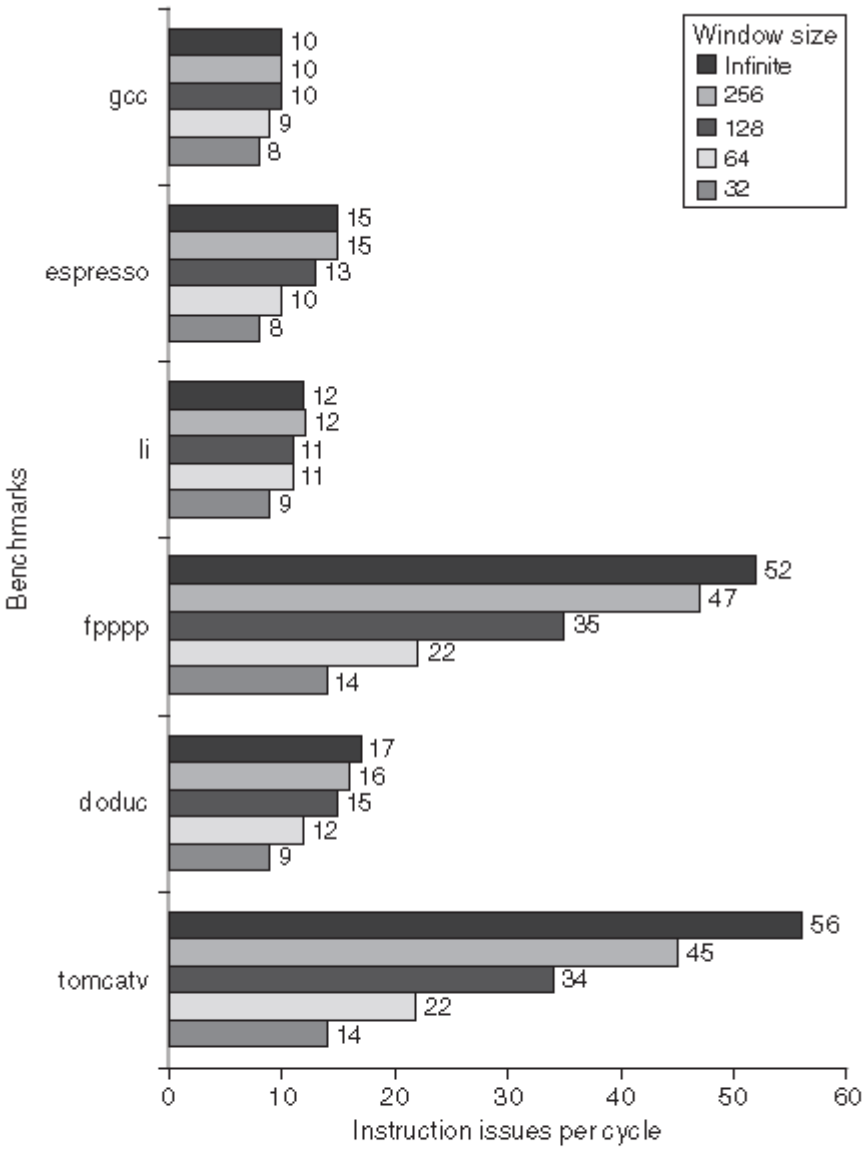

Figure 3.27 The amount of parallelism available versus the window size for a variety of integer and floating-point programs with up to 64 arbitrary instruction issues per clock. Although there are fewer renaming registers than the window size, the fact that all operations have one-cycle latency and the number of renaming registers equals the issue width allows the processor to exploit parallelism within the entire window. In a real implementation, the window size and the number of renaming registers must be balanced to prevent one of these factors from overly constraining the issue rate.

对于各种整数与浮点程序，每个时钟周期发射64个任意指令时，可用并行数随窗口大小的变化情况。

挑战：如何更好地利用集成电路上有限的可用资源

3.10.3超越本研究的局限

两种局限：在完美推测处理器中也会存在的局限性；在一种或多种现实模型中存在的局限性

适用于完美模型的最重要局限性：

（1）访问存储器的WAW和WAR冒险

（2）不必要的相关

（3）克服数据流限制

对于不够完美的处理器，已经提出了几种可以发现更多ILP的思想。其中一个例子为沿多条路径进行推测，这样可以降低错误恢复的成本、发现更多的并行。

此节介绍的所有限制都不是根本性的限制，因为克服它们并不需要改变什么物理定律。主要是实践中的限制，如窗口大小、别名检测、分支预测，这都是设计人员和研究人员需要应对的挑战。