paralllisim

2019年6月1日 10:57

ILP：指令层次的并行
Pipeline 属于ILP的一种

为了提高ILP：deeper pipeline
multiple issue

Multiple Issue：

Multiple issue的两个关键：
怎么放放那些指令进入issue slot
解决data control hazard

- EX data hazard
  - Forwarding avoided stalls with single-issue
  - Now can't use ALU result in load/store in same packet
    - add $t0, $s0, $s1
      load $s2, 0($t0)
    - Split into two packets, effectively a stall
- Load-use hazard
  - Still one cycle use latency, but now two instructions

解决以上两问题：
Static multiple issue： VLIW processor
编译器在编译阶段消除部分或全部hazard：软件保证正确性
静态分支预测 && 代码调度
Dynamic multipe issue：
CPU在执行阶段用高级技术搞 superscalar processor
硬件保证正确性
编译器重新排列代码也会辅助（可以但不必要）

推测：speculation 提高ILP性能　可以由编译器(static)或硬件(dynamic)完成

Eg.　Load 提到store前 猜测不对同一个访问

Eg.　Branch

修复：　静态　软件 例程回复/

动态 硬件 缓存

异常：　静态 软件 添加ISA支持延迟异常

异常不报直到推测确定处理 ‘

静态补丁：

Loop unrolling：循环展开

Static multiple issue为了更好完成代码调度的方式

寄存器重命名：

- 寄存器重命名：由编译器或硬件对寄存器进行重命名以消除
- 反相关：也被称为名字相关，因为寄存器名的重用导致的相
  一个值导致的真正相关。

减少循环控制指令和由于名字相关而产生的stall，代价是使用更多的临时寄存器，并且代码变长

动态：

- Hardware support for reordering the order of instruction execution
- Allow the CPU to execute instructions out of orc avoid stalls
  - But commit result to registers in order

为什么要用动态：

1.不是所有可预测 cache miss

2.大多分支过程是动态的

3.ISA不同产生不同latency 和 hazard

会反相关。
相关，并非由两条指令中使用同

der to