**现代计算机体系架构第二次课程实验**

**——超标量处理器性能和Cache架构仿真**

# 目标：

本次实验的目的是针对Out of order（O3）处理器和Cache技术进行仿真。实验采用整数向量运算程序作为测试样例，探究在不同参数下的处理器、cache性能变化，并与Minor CPU模型进行比较。

# 实验准备：

复习超标量处理器和Cache架构相关部分课件。

参考：阅读附件gem5\_tuturial 41-43页教程了解gem5 checkpoint的使用方法

# 实验内容：

## 1. 搭建实验环境

**1.1. 测试程序编译**

下面的这段C++代码(iaxpy.cpp)实现了随机向量的产生及运算

|  |
| --- |
| … …  // Start of iaxpy loop m5\_dump\_reset\_stats(0, 0);  for (int i = 0; i < N ; i++)  {  Y[i] = X[i] / alpha + Y[i];  }  m5\_dump\_reset\_stats(0, 0);  // End of iaxpy loop  … … |

程序中加入m5\_dump\_reset\_stats(0,0)语句以观察感兴趣代码区域的执行情况。

将iaxpy.cpp置于gem5目录下，使用以下命令将其编译为可执行文件：

|  |
| --- |
| g++ iaxpy.cpp util/m5/src/abi/x86/m5op.S -o daxpy-gem5 --static --std=gnu++0x -O2 -I ./include/ |

**1.2. 仿真环境搭建**

编译X86指令集下的处理器仿真程序build/X86/gem5.opt。

注意：请先删除build/目录，打开build\_opts/X86文件，在CPU\_MODELS后添加MinorCPU选项，后再编译生成X86架构处理器仿真程序。此外，编译X86架构的内存要求较高，最好配置4GB以上。

采用configs/example/se.py文件测试1.1中得到的daxpy-gem5文件，你会在 m5out/stats.txt文件中看到仿真报告输出，其中包含了感兴趣部分代码的执行信息。

## 2. 关于处理器性能的问题

**2.1. 设置仿真脚本**

将附件中的python仿真脚本文件夹hw2放入到configs/目录下，通过阅读hw2/和common/文件夹下的代码，回答问题并指出是在哪里找到了答案。

注意：configs/目录下提供的是硬件架构的配置信息，如果想详细了解cache， CPU的配置参数和方法，请阅读gem5/src/目录下的cpu/minor, cpu/o3, mem/cache等目录中的文件。

1. 默认O3 CPU的commit width是多少？
2. 默认ROB (reorder buffer)的entries size是多少？
3. 默认的L1 Cache的大小是多少？
4. 默认的L1 Cache映射和替换策略是什么？
5. 默认的除法单元的操作的延时是多少？
6. 默认的整数ALU单元的延时是多少？
7. 默认的ALU (add/sub)和整数乘除(mul/div)的运算单元数量是多少？

**2.2. （\*选做）采用 objdump -S 指令导出 daxpy-gem5的汇编代码**

1. 在感兴趣区域里有多少条汇编指令？给出开始/结束行序号+代码。
2. 哪些指令是除计算之外的额外的指令？
3. 采用默认的参数，估计O3处理器下，感兴趣部分代码的IPC是多少？

**2.3. 采用 提供的run.py（而非之前使用的se.py）分别用 Minor 和 O3 CPU 类型仿真 daxpy-gem5**

1. Minor CPU和O3 CPU上运行的IPC分别是多少？

（\*选做）如果你做了2.2部分，那么感兴趣区域在O3 CPU的实际IPC与你估计的有多少差距？结合stats 文件，给出限制了 ILP 的因素。

1. 如果除法单元的操作延时变更为10 cycles，Minor 和 O3类型的 CPU，哪一种性能上受到的影响比较大，为什么？
2. 在默认的延时设定下，每一种 CPU 类型下branch prediction error数量分别是多少？优化branch predictor是否必要？

**2.4. Cache 设计的性能分析**

1. 分析L1 Cache大小对性能的影响。根据例程中的代码，调整L1 Cache的大小，分析其对性能的影响。
2. 分析不同替换策略对代码执行性能的影响。在全相联映射和组相联映射策略中，当Cache发生Miss时，需要获取新的数据并替换原有数据。常见的替换策略有随机法（RAND），先入先出法（FIFO），最近最少使用法（LRU）等。请任选两种或两种以上不同替换策略，**描述其原理并分析其对 iaxpy 程序性能的影响**。

（gem5提供了多种常见的替换策略，详见 src/mem/cache/replacement\_policies/ ReplacementPolicies.py中提供的策略名称）

## 3. 设计方法综合探究

假设你得到一个设计X86 CPU的机会。有两种处理器架构可以选择：一款是O3 CPU，1GHz。一款是高速的In order CPU, 3GHz；有两种除法运算单元可以选择：一款是高速除法器，可以每1个周期执行一个除法运算，有10周期的issue延时，既可以用于O3 CPU也可以用于In order CPU；第二种是流水线型除法器，可以每10个周期执行一次除法操作，issue延时为1个周期，但只能用于In order CPU。如果奇葩的CTO只关心执行iaxpy程序的性能，你应该如何设计CPU？请简述理由，向CTO推销你想采取的**CPU模型和除法器选择**。

# 提交要求：

1. **PDF或DOCX格式**的实验报告，回答所有问题。
2. 不方便翻译的名词可以直接用英文，需要包括典型结果输出，可以截图或以文本形式粘贴，例如：

|  |
| --- |
| \*\*\*\* REAL SIMULATION \*\*\*\* info: Entering event queue @ 0. Starting simulation...  info: Increasing stack size by one page.  Hello world!  hack: be nice to actually delete the event here  Exiting @ tick 13410500 because target called exit() |

报告中也可以说明你在实验中遇到的问题以及解决方法。

1. 如果有自行编写的脚本文件请附上并在报告中说明。
2. **将实验报告和其他材料（代码、仿真结果等）打包，命名为“HW2\_姓名\_学**

**号.zip”**，在网络学堂上提交，请注意截止时间。