实验报告

实验名称（利用 GEM5 评测系统性能）

智能 1501 201508010528 耿昊

实验目标

利用 GEM5 仿真 x86-64 系统，测量（多线程）FFT 程序运行时间，与实际 x86-64 系统的测试结果进行比较。

实验要求

- 采用 C/C++编写程序，选择合适的运行时间测量方法
- 根据自己的机器配置选择合适的输入数据大小 n，保证足够长度的运行时间
- 对于不同的线程数目，建议至少选择 1 个，2 个，4 个，8 个，16 个线程进行测试
- 回答思考题，答案加入到实验报告叙述中合适位置

思考题

1. GEM5 是什么？怎么使用？
2. 利用仿真器评测系统性能的优点是什么？缺点是什么？
3. GEM5 仿真系统测得的程序性能与实际系统测得的程序性能差别大不大？可能的原因是什么？

实验内容

GEM5 的安装和使用

GEM5 的官方文档在 http://gem5.org/Documentation， 包括了安装和使用说明。

GEM5 网站也提供了 x86-64 系统 Linux 镜像下载，可以在 GEM5 的下载页面找到。

大家也可以参考 https://blog.csdn.net/u012822903/article/details/62444286 及其相关文章，了解如何利用 GEM5 进行 Linux 全系统模拟和运行自己的测试程序。

利用 GEM5 测试 FFT 程序在 x86-64 系统上的性能

在安装好 GEM5，并掌握如何使用 GEM5 运行自己的测试程序后，可以在 GEM5 上运行 FFT 程序，测试其性能。使用实验一中的单线程 FFT 程序，记录测试数据。

测试

测试平台

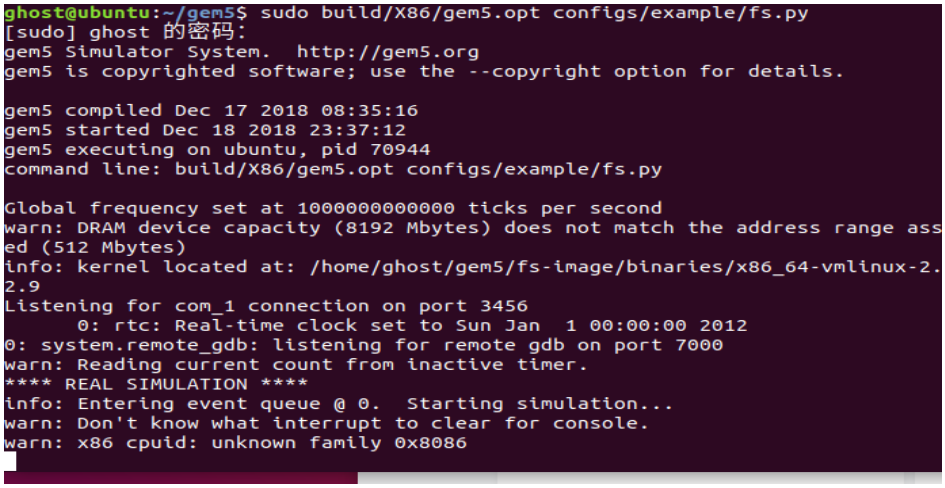
在如下机器上进行了测试：

部件	配置	备注
CPU	core i5-8300H	

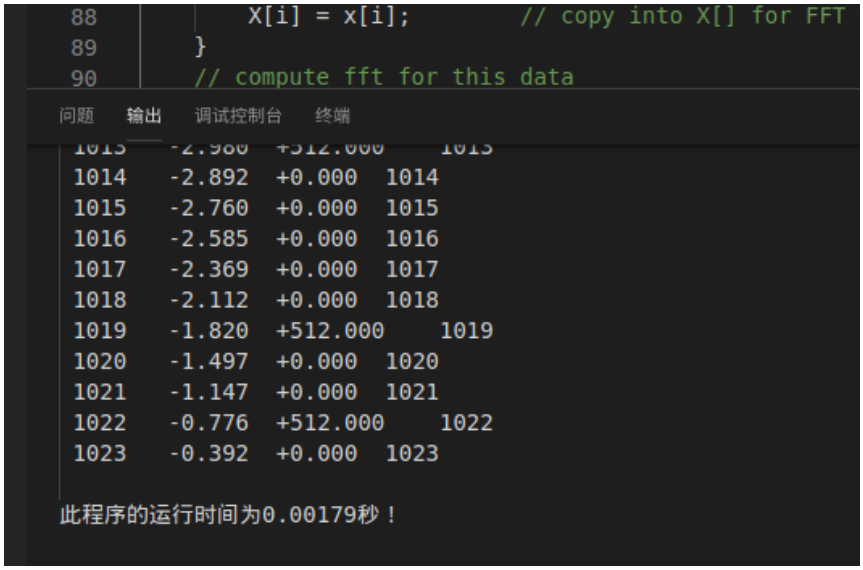
部件	配置	备注
内存	DDR4 4GB	
操作系统	Ubuntu 18.04 LTS（虚拟机）	中文版

测试记录

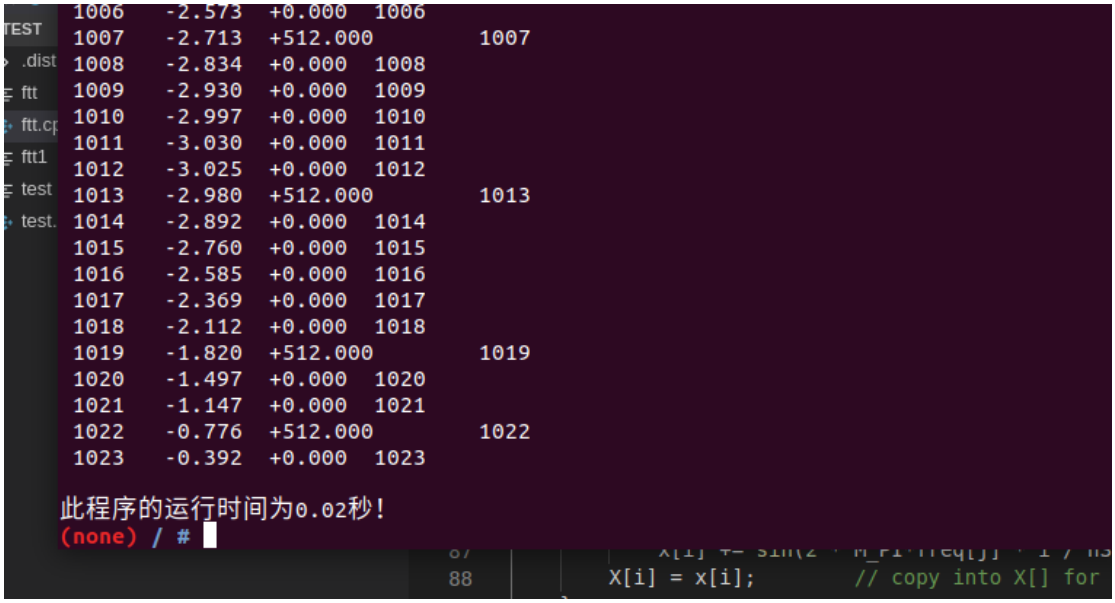
FFT 程序测试结果截图如下：



本地：



Gem5 全系统:



分析和结论

从测试记录来看，FFT 程序在 GEM5 上的执行时间是 0.02s，与实际系统上测得的执行时间相比，GEM5 模拟花费的时间是实际系统的 10 倍。可以得出 GEM5 中一条源代码指令需要许多条指令来解释。

思考题

1.GEM5 是一个 CPU 模拟器。

1) GEM5 模拟器提供了四个不同的 CPU 模型,两个不同的系统模型以及两个不同的内存系统模型,并且支持多种指令集, ARM、ALPHA、MIPS、Power、SPARC 和 x86),其中可以在 ARM、ALPHA 和 x86 三种架构上运行 Linux。在这里我们利用了 linux 上的 x86 框架进行全系统仿真。

2) 全系统仿真的步骤：

①新建文件夹 mountfile，用于存放挂载的文件。

②进行系统挂载，具体命令如下：sudo mount -o,loop,offset=32256 fs-image/disks/linux-x86.img /mnt,

③执行 umount 操作，具体命令如下：sudo umount /mnt

④重新启动 gem5 的 FS，命令如下：sudo build/X86/gem5.opt configs/example/fs.py

⑤打开另一个终端，输入如下命令：sudo ./m5term 127.0.0.1 3456

⑥编译源代码，等待很长时间（20~30min），在这里之所以使用源代码编译是因为 gem5 中编译器版本不匹配，在本机上编译好的静态可执行文件无法正常执行）

⑦执行可执行文件

2.使用仿真模拟器测评系统性能的优缺点：

优点：利用仿真器进行系统测评最大的优点就是在于成本低、风险小。

缺点：

1)仿真器制作难度大，一款强大的仿真器结构十分复杂。因而制作难度大，功能不够完善，这就可能会造成一定的差误。

2)仿真器的仿真时间过长，因为在仿真器中，一条指令的编译可能需要很多条指令来解释，所以大型 CPU 仿真等待时间会比较长。

3.GEM5 仿真系统测得的程序性能与实际系统测得的程序性能差别很大，原因在于模拟器中源代码的一条指令需要多条指令来解释。在虚拟机上的实验结果中可以看出性能相差 10 倍左右。