

**实验报告**

**实 验（六）**

题 目 Cachelab

高速缓冲器模拟

专 业 计算机科学与技术

学　　 号 1170300520

班　　 级 1703005

学 生 郭子阳

指 导 教 师 吴 锐

实 验 地 点 G712

实 验 日 期

**计算机科学与技术学院**

**目 录**

[第1章 实验基本信息 - 3 -](#_Toc500230541)

[1.1 实验目的 - 3 -](#_Toc500230542)

[1.2 实验环境与工具 - 3 -](#_Toc500230543)

[1.2.1 硬件环境 - 3 -](#_Toc500230544)

[1.2.2 软件环境 - 3 -](#_Toc500230545)

[1.2.3 开发工具 - 3 -](#_Toc500230546)

[1.3 实验预习 - 3 -](#_Toc500230547)

[第2章 实验预习 - 4 -](#_Toc500230548)

[2.1 画出存储器层级结构，标识容量价格速度等指标变化（5分） - 4 -](#_Toc500230549)

[2.2用CPUZ等查看你的计算机Cache各参数，写出各级Cache的C S E B s e b（5分） - 4 -](#_Toc500230550)

[2.3写出各类Cache的读策略与写策略（5分） - 4 -](#_Toc500230551)

[2.4 写出用gprof进行性能分析的方法（5分） - 4 -](#_Toc500230552)

[2.5写出用Valgrind进行性能分析的方法（（5分） - 4 -](#_Toc500230553)

[第3章 Cache模拟与测试 - 5 -](#_Toc500230554)

[3.1 Cache模拟器设计 - 5 -](#_Toc500230555)

[3.2 矩阵转置设计 - 5 -](#_Toc500230556)

[第4章 总结 - 6 -](#_Toc500230557)

[4.1 请总结本次实验的收获 - 6 -](#_Toc500230558)

[4.2 请给出对本次实验内容的建议 - 6 -](#_Toc500230559)

[参考文献 - 7 -](#_Toc500230560)

# 第1章 实验基本信息

## 1.1 实验目的

理解现代计算机系统存储器层级结构

掌握Cache的功能结构与访问控制策略

培养Linux下的性能测试方法与技巧

深入理解Cache组成结构对C程序性能的影响

## 1.2 实验环境与工具

### 1.2.1 硬件环境

Intel Core i7 6700HQ，8GB RAM，128GB SSD

### 1.2.2 软件环境

Windows 10专业版 64位，Ubuntu 18.04.1 64位

### 1.2.3 开发工具

Codeblocks 17.12，gcc，cgdb，vscode

## 1.3 实验预习

上实验课前，必须认真预习实验指导书（PPT或PDF）

了解实验的目的、实验环境与软硬件工具、实验操作步骤，复习与实验有关的理论知识。

画出存储器的层级结构，标识其容量价格速度等指标变化

用CPUZ等查看你的计算机Cache各参数，写出C S E B s e b

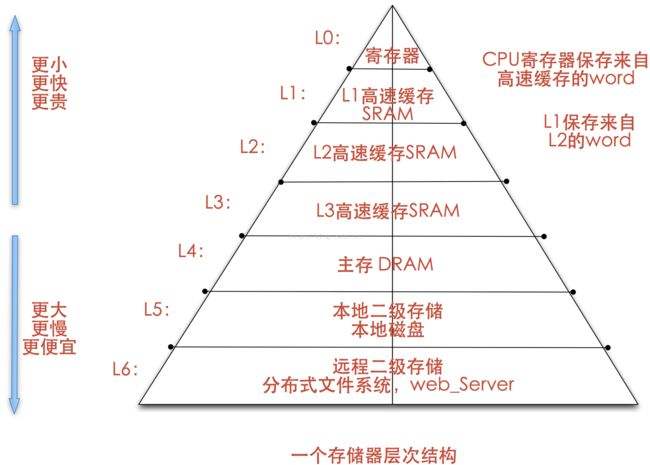
写出Cache的基本结构与参数

写出各类Cache的读策略与写策略

掌握Valgrind与Gprof的使用方法

# 第2章 实验预习

## 2.1 画出存储器层级结构，标识容量价格速度等指标变化（5分）



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | C | S | E | B | s | e | b |
| 一级数据缓存 | 32KB\*2 | 64 | 8 | 64B | 6 | 3 | 6 |
| 一级指令缓存 | 32KB\*2 | 64 | 8 | 64B | 6 | 3 | 6 |
| 二级缓存 | 256KB\*2 | 1024 | 4 | 64B | 10 | 2 | 6 |
| 三级缓存 | 3MB | … | 12 | 64B | … | … | 6 |

## 2.2用CPUZ等查看你的计算机Cache各参数，写出各级Cache的C S E B s e b（5分）

## 2.3写出各类Cache的读策略与写策略（5分）

读策略：

1：命中，则从cache中读相应数据到CPU或上一级cache中。

2：失败，则从主存或下一级cache中读取数据，并替换出一行数据，通常采用LRU算法。

写策略：

1：命中，又分两种策略

（1）写回法：只写本级cache，暂时不写数据到主存或下一级cache，等到该行被替换出去时，才将数据写回到主存或下一级cache。

（2）写直达：写本级cache，同时写数据到主存或下一级cache，等到该行被替换出去时，就不用写回数据了。

2：失败，又分两种策略

（1）按写分配，又分两种：[1]先写数据到主存或下一级cache，并从主存或下一级cache读取刚才修改过的数据，即：先写数据，再为所写数据分配cache line；[2]先分配cache line给所写数据，即：从主存中读取一行数据到cache，然后直接对cache进行修改，并不把数据到写到主存或下一级cache，一直等到该行被替换出去，才写数据到主存或下一级cache。

（2）写不分配：直接写数据到主存或下一级cache，并且不从主存或下一级cache中读取被改写的数据，即：不分配cache line给被修改的数据。

## 2.4 写出用gprof进行性能分析的方法（5分）

在编译和链接时，加上-pg选项。

执行编译的二进制程序

程序正常退出后，在运行目录下 生成gmon.out文件。如果原来有gmon.out 文件，将会被覆盖。

用gprof工具分析gmon.out文件。

使用gprof Binary-file gmon.out >report.txt将报告输出在report.txt中。

## 2.5写出用Valgrind进行性能分析的方法（（5分）

valgrind命令的格式如下：

valgrind [valgrind-options] your-prog [your-prog options]

编译（加-g选项）得到a.out，再执行valgrind ./a.out

# 第3章 Cache模拟与测试

## 3.1 Cache模拟器设计

提交csim.c

程序设计思想：

initCache()和freeCache()较简单，主要是accessData()函数，分割请求地址可以得到组索引、标记和块偏移，通过组索引可以找到缓存中对应的组，遍历组中的每一个行，查看是否有相同标记的行，如果有则命中，直接修改缓存行的数据。

如果没有，则是不命中的情况。区分冷不命中和冲突不命中的情况。如果是冷不命中，直接将数据写入无效的空行即可。如果是冲突不命中，则根据LRU算法，查找最早被使用的行，每个缓存行中的数据是第几次写入，数据越小使用的事件越早。此时找出行中的LRU最小的数据并替换。

测试用例1的输出截图（5分）：



测试用例2的输出截图（5分）：



测试用例3的输出截图（5分）：



测试用例4的输出截图（5分）：



测试用例5的输出截图（5分）：



测试用例6的输出截图（5分）：



测试用例7的输出截图（5分）：



测试用例8的输出截图（10分）：



**注：每个用例的每一指标5分（最后一个用例10）**——与参考csim-ref模拟器输出指标相同则判为正确

## 3.2 矩阵转置设计

提交trans.c

程序设计思想：

将矩阵分块，对每一块分别转置，一次取出一个块的所有数据，存放在一个局部变量里，这样每个块访问一次即可，防止转置时抖动的发生。

**32×32（10分）：运行结果截图**

****

**64×64（10分）：运行结果截图**

****

**61×67（20分）：运行结果截图**

# 第4章 总结

## 4.1 请总结本次实验的收获

深刻理解了缓存系统以及其运行机制，理解了缓存的读策略、写策略和替换策略。

## 4.2 请给出对本次实验内容的建议

转置矩阵的优化程度不够