

**实验报告**

**实 验（六）**

题 目 Cachelab

高速缓冲器模拟

专 业 计算机

学　　 号 1180300303

班　　 级 1836101

学 生 宿梓航

指 导 教 师 刘宏伟

实 验 地 点 G712

实 验 日 期 11.28.2019

**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 i7-8750H, 2.2GHz

16G RAM

Samsung SSD 970 PRO 1TB

### 1.2.2 软件环境

Windows 10 1903 x64

Ubuntu 19.04

Vmware Workstation 15.1

### 1.2.3 开发工具

Visual Studio 2019

GCC 8.3.0，8.1.0

GNU nano 3.2

## 1.3 实验预习

填写

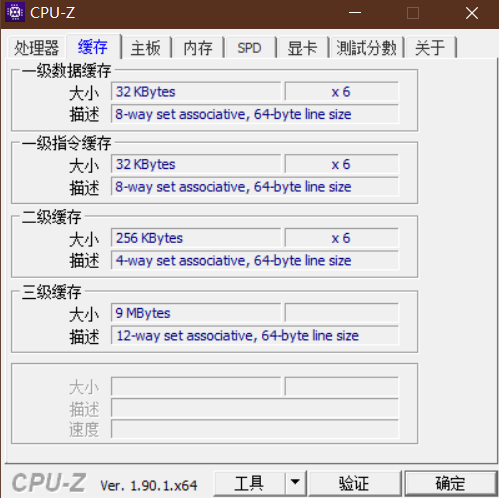
# 第2章 实验预习

## 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）

|  |  |  |  |
| --- | --- | --- | --- |
| 层级 | 容量(升序) | 速度(降序) | 价格(降序) |
| 寄存器 | 1 | 1 | 1 |
| L1高速缓存 | 2 | 2 | 2 |
| L2高速缓存 | 3 | 3 | 3 |
| L3高速缓存 | 4 | 4 | 4 |
| 主存 | 5 | 5 | 5 |
| 本地二级储存 | 6 | 6 | 6 |
| 远程二级储存 | 7 | 7 | 7 |

## 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）



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | C | S | E | B | s | b |
| L1数据 | 32K | 64 | 8 | 64 | 6 | 6 |
| L1指令 | 32K | 64 | 8 | 64 | 6 | 6 |
| L2 | 256K | 1024 | 4 | 64 | 10 | 6 |
| L3 | 9M | 12288 | 12 | 64 | 40 | 6 |

## 2.3写出各类Cache的读策略与写策略（5分）

读策略：

如果缓存命中，则从cache中读相应数据到CPU或上一级cache中；

如果缓存未命中，则从主存或下一级cache中读取数据，这可能替换出一行数据。

写策略:

如果缓存命中，则可以选择写回，或者直写。

直写，即立即更新主存或下一级cache；

写回，即仅当替换算法驱逐某一个块时，才更新主存或下一级cache。

如果缓存不命中，可采用写分配或者非写分配。

写分配，即加载低一层中的块，然后更新它的值。

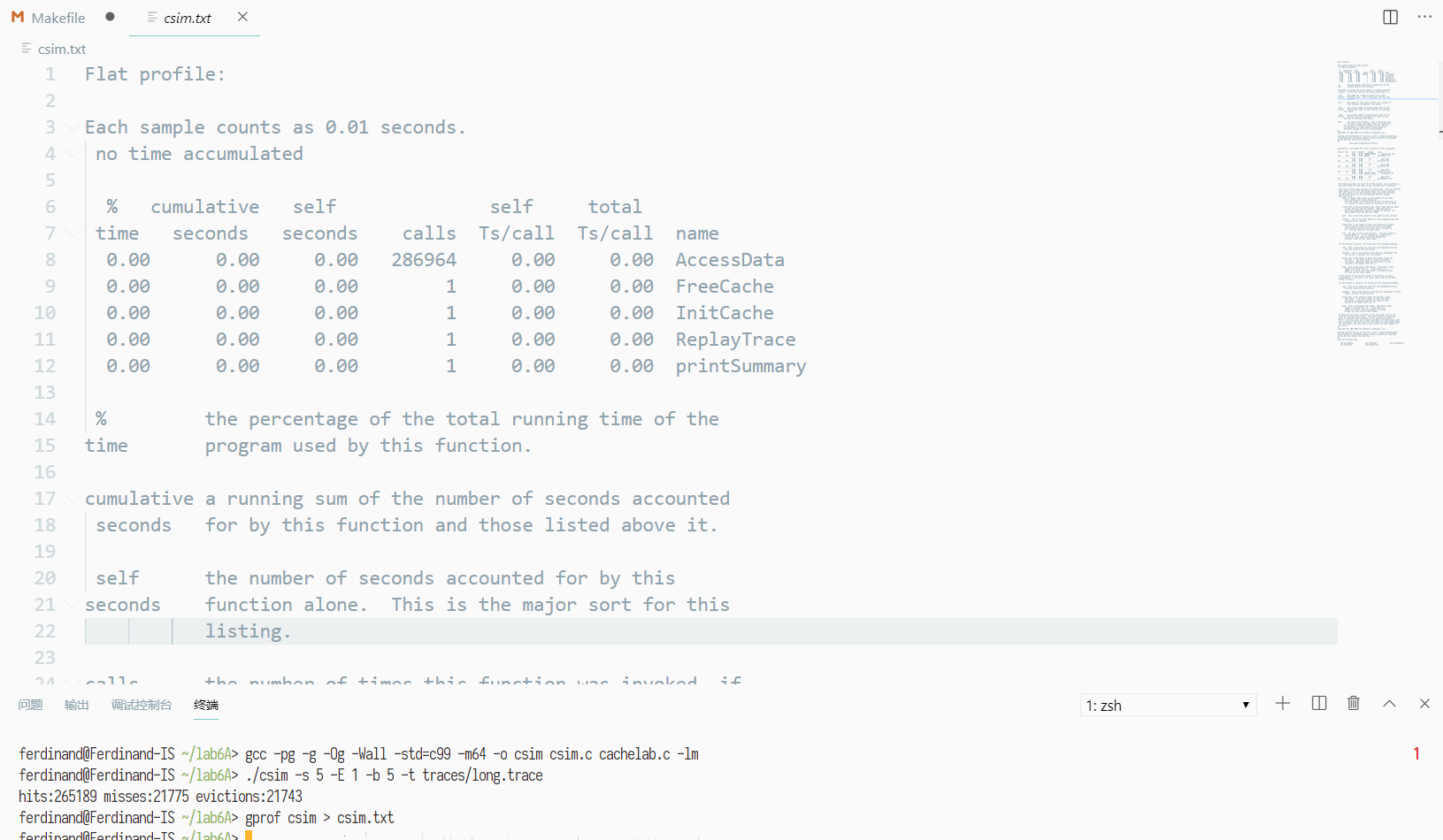
非写分配，则是直接更新低一层中的块。

## 2.4 写出用gprof进行性能分析的方法（5分）

1.使用-pg开关和-0g运行gcc，进行编译

2. 正常运行程序

3.执行gprof+可执行文件来进行分析



第一部分为各个函数花费的时间，第二部分为函数调用历史。

## 2.5写出用Valgrind进行性能分析的方法（（5分）

执行valgrind --tool=callgrind+通常运行程序的命令行，即可生成程序性能相关数据callgrind.out.\*这是一个纯文本文件。

# 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：

根据所给程序框架，

我们首先需要编写三个函数，并补全main函数。

首先是

void InitCache()

初始化Cache的函数，那么我们只需要malloc 整个Cache为组数\*8，

然后再用calloc给Cache的每一个组初始化即可。

至于SetIndexMask变量，应该为组数-1

void FreeCache()

挨个free即可

void AccessData(ulong address)

是本题核心，按照提示使用LRU算法。

    ulong tag = address >> ((IndexBitsCount + BlockOffsetBitsCount) & 0xFF);

    TCacheSet cacheSet = Cache[(address >> BlockOffsetBitsCount) & SetIndexMask];

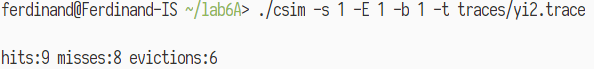
先获取对应的标签和组，然后在组中顺序遍历查找标签，如果找到则缓存命中。更新命中者的LRU全局LRU之后返回即可。

如果缓存未命中，则利用LRU策略进行替换。考虑到Valid字段为false的项总有Lru=0<所有其他的Lru，那么我们只要便利整个组，查找Lru最小的项替换掉即可。

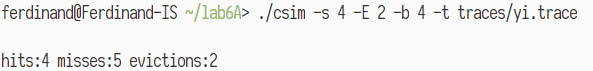
补全main函数其实就是用b和s位数求组数和块大小，移位即可。

为了增强代码可读性，编写的时候对一些变量作了重命名。

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）：



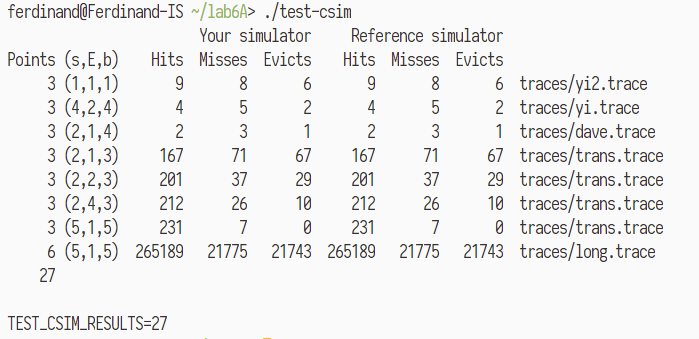
测试用例7的输出截图（5分）：



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：

对于32\*32的矩阵：

考虑到每8个整数(8\*8=64B)可占用Cache的一个组，每次读取A的一行到临时变量，仅占用了一组Cache。那么如果我们再 访问B时，因为冲突不命中的只会有首元素和A占用的那一行。

因此对于32\*32的矩阵只需分为16个8\*8的块，然后对每个块进行操作，即可大大提高Cache命中率。

对于64\*64的矩阵：

如果沿用8\*8的分块，因为B的load导致的冲突不命中会显著增加。

所有我们考虑减少B的load的次数，那么由于矩阵每行64个int，我们最多可以同时写B的4行。所以我们在8\*8的分块下进行细分，每个8\*8分块分成4个4\*4分块，这样可以使得Cache足够装下待处理的B，分别进行处理。

现对4个4\*4分块进行编号按照1-4象限顺序。

考虑到A和B的分块奇数象限为直接对应，而偶数象限是互反的，那么我们可分别按如下顺序进行转置。A12->B12,A34->B34,调换B13

然而这样的效果并不好，原因是调换B13的过程中进行了过多的额外内存访问，然而这并不必要，我们只需要从A中读取即可。

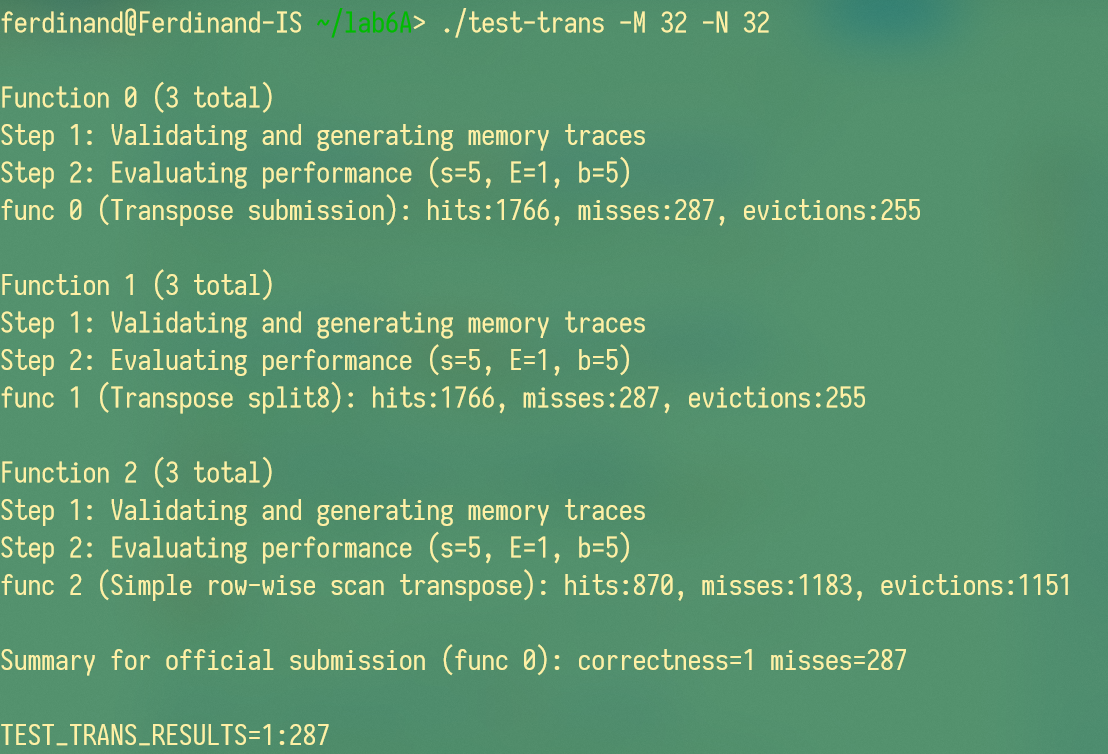
那么我们更改顺序为A12->B12; A4->B4;A3->B1,B1->B3;

不过分数只达到了1643，之后就不知道怎么办了，，

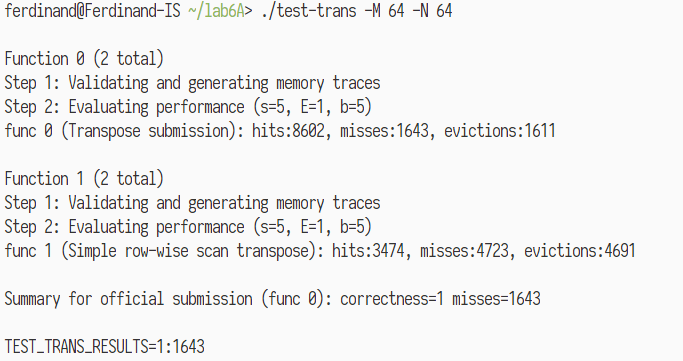
对于61×67矩阵：

先8\*8分块运行一下，发现效果还不错，hhhhh（什么嘛，我打的还蛮准的嘛

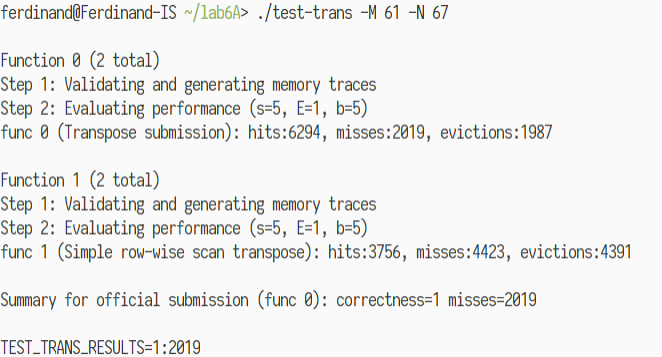
**32×32（10分）：运行结果截图**

****

**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

## 4.2 请给出对本次实验内容的建议

注：本章为酌情加分项。

# 参考文献

**为完成本次实验你翻阅的书籍与网站等**

[1] 林来兴. 空间控制技术[M]. 北京：中国宇航出版社，1992：25-42.

[2] 辛希孟. 信息技术与信息服务国际研讨会论文集：A集[C]. 北京：中国科学出版社，1999.

[3] 赵耀东. 新时代的工业工程师[M/OL]. 台北：天下文化出版社，1998 [1998-09-26]. http://www.ie.nthu.edu.tw/info/ie.newie.htm（Big5）.

[4] 谌颖. 空间交会控制理论与方法研究[D]. 哈尔滨：哈尔滨工业大学，1992：8-13.

[5] KANAMORI H. Shaking Without Quaking[J]. Science，1998，279（5359）：2063-2064.

[6] CHRISTINE M. Plant Physiology: Plant Biology in the Genome Era[J/OL]. Science，1998，281：331-332[1998-09-23]. http://www.sciencemag.org/cgi/ collection/anatmorp.