

**计算机系统结构实验报告**

|  |  |
| --- | --- |
| 姓 名： | 王占成 |
| 学 院： | 计算机科学与技术 |
| 专 业： | 计算机科学与技术 |
| 班 级： | CS1707 |
| 学 号： | U201714786 |
|  |  |

|  |  |
| --- | --- |
| 分数 |  |
| 教师签名 |  |

2020 年.04月.28日

**目 录**

[1. Cache模拟器实验 3](#_Toc23963)

[1.1. 实验目的 3](#_Toc3705)

[1.2. 实验环境 3](#_Toc865)

[1.3. 实验思路 3](#_Toc25045)

[1.4. 实验结果和分析 4](#_Toc21474)

[2. 总结和体会 5](#_Toc18091)

[3. 对实验课程的建议 5](#_Toc32753)

# Cache模拟器实验

* 1. **实验目的**

1. 理解cache工作原理
2. 如何实现一个高效的模拟器
   1. **实验环境**
3. 操作系统：Ubuntu 64 bit；
4. 编译器：GCC 6.6.0；
5. IDE：Visual Studio Code。
   1. **实验思路**
6. 该实验只是模拟cache工作原理中的一小部分(Hit/Miss/Eviction)，csim.c文件已经把最基本的实验框架给我们搭好，我们只需要完成csim.c文件中相应的函数即可。
7. Cache替换采用LRU算法，即最近最不经常使用算法。首先要明白LRU算法的工作流程。检查cache某一组内所有cacheline的tag是否与输入的tag相等，如果相等且valid位为1，则表示cache访问命中。此时，就需要把lru位置零。对于所有的不命中的cacheline，lru计数器加1。如果所有的cacheline都不含有目标tag，则需要把主存块中的内容调入到按照指定规则选取到的cacheline中。规则是，先去找空闲的cacheline，即valid = 0的cacheline；如果没有空闲的cacheline，则此时需要遍历所有的cacheline，寻找lru计数最大的一个cacheline。最后将主存块中的内容存储到选取到的cacheline中。
8. 在本次实验中，用循环遍历cacheset中所有的cacheline，如果命中，增加命中数并且将对应的cacheline的lru清零。如果所有的cacheline都不含有寻找的tag，分成两种情况进行处理：有空闲的cacheline，直接将tag写入空闲cacheline并置valid位为1；否则，将tag写入lru最大的cacheline并置lru为0即可。
9. 对于Load/Store指令，只需要访问主存一次。所以在replayTrace函数中，只需要调用一次accessData。Modify指令需要访问两次主存，因此需要调用两次accessData。
   1. **实验结果和分析**
10. ./csim(-ref) -v -s 4 -E 1 -b 4 -t traces/yi.trace运行结果如下图 1-1：

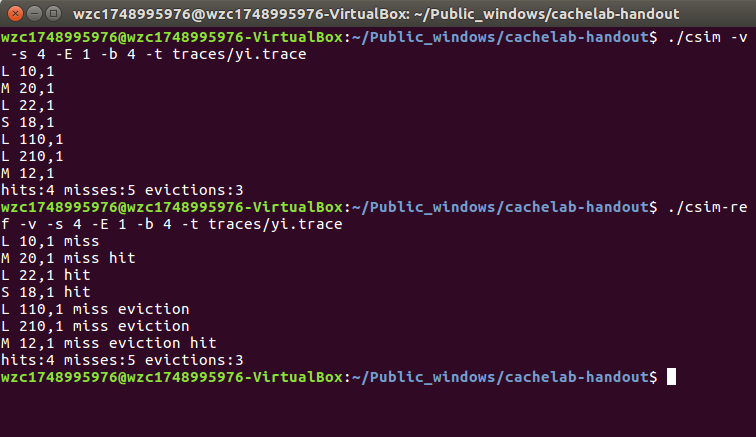


图 1-1

1. ./test-csim运行结果如下图 1-2：

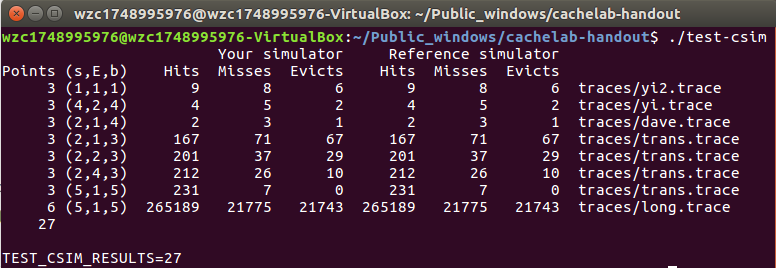


图 1-2

综上进行分析，Hits、Misses、Evicts结果均与标准测试程序的结果相同。

# 总结和体会

1. 这次实验进一步加深了我对cache的理解，之前在组成原理课程中用logisim实现过cache，当时对于cache学习的比较仔细，所以这次实验相对于来说就变得简单了一些。在编写代码的过程中，有些函数一开始编写的比较繁琐，经过优化之后复杂度也能差强人意。
2. 理解底层的LRU算法，对于理解操作系统原理和组成原理有着更好的推进作用。

# 对实验课程的建议

1. 整个cache实验做起来比较抽象，希望能够更完善一些，能够大体通过实验来基本完成LRU算法的所有流程。