**实验三、四 Cache基本性能&结构性能分析**

模拟器A的设计思想、特色：

本模拟器参考借鉴了各个开源的C#项目代码，并在此基础上自主编写完成。

程序主要分为Cache.cs和Form1.cs两部分，Cache.cs为后端Cache算法程序，Form1.cs为前端交互式界面。

特色：本软件以.NET Framework 4.5为基础框架，整体架构清爽简洁，对于Cache配置操作有对应的帮助文本框和报错提示，便于初学者学习操作。本软件占用空间大小仅为155KB，易于存储、拷贝携带，相比于模拟器C（MyCache），小了近5倍 。

模拟器内4种trace测试：

1. 022.li.din：





Cache大小与不命中率的关系图

在022.li.din文件测试中，选择不同的Cache大小（从8KB到64KB）。可见，随着Cache大小增大，不命中率整体呈现下降趋势，且读数据不命中率变化最明显。

1. 047.tomcatv.din：





相联度与不命中率的关系图

在047.tomcatv.din文件测试中，选择不同的相联度（从1到8）。可见，在相联度为1、2时，不命中率整体呈现下降趋势；而随着相联度增大，不命中次数均变化不明显，数据呈现上下波动。

1. 078.swm256.din：





块大小与不命中率的关系图

在047.tomcatv.din文件测试中，选择不同的相联度（从1到8）。可见，在相联度为1、2时，不命中率整体呈现下降趋势；而随着相联度增大，不命中次数均变化不明显，数据呈现上下波动。

1. 085.gcc.din：



调块策略与不命中率的关系图

在085.gcc.din文件测试中，选择是否按写分配。可见，按写分配的不命中率均比不按写分配的低，且在写数据不命中率中差异明显。

四个文件各个不命中率对比表

实验感悟

在最初着手做这个实验，搜寻网上资料时，发现大部分模拟器是用C语言、CLI的形式实现，并没有交互式界面，无法直观展示操作；且操作十分麻烦复杂，与实验目标不符。而MyCache在网上的报告教程多如牛毛，并不能较好地锻炼学生的编程能力与创新思维。

通过本次实验，引用开源项目，自主编写Cache模拟器程序，来学习理解Cache的相关知识，极大地锻炼了我们计算机体系结构初学者的动手编程能力，也让我们更好地学习理解了Cache的原理、实现过程以及性能关系，受益匪浅！

十分感谢任课老师对本次实验的精心设计和安排！