实验 1：Cache 算法的设计、实现与比较

一、 实验内容

编写一个 Cache 模拟程序，通过模拟 trace 文件中的内存访问记录，统计并比较不同 Cache 设计策略下的命中率及性能差异。

要求实现并分析如下实验项目：

（1）直接映射（Direct-Mapped）

- 如果使用直接映射 DM 方法组织 Cache，Cache 命中率为多少？

（2）组关联（Set-Associative）

- 如果使用组关联 SA 方法组织 Cache，当路数为 2-way、4-way、8-way、16-way 时，Cache 命中率分别为多少？并分析关联度对 Cache 命中率的影响。

（3）块大小（Block Size）影响分析

- 如果使用 4-way 组关联 SA 方法组织 Cache 时，当块大小分别为 8Bytes、16Bytes、32Bytes、64Bytes、128Bytes 以及 256Bytes 时，其他参数不变，Cache 命中率分别为多少？并分析块大小对 Cache 命中率的影响。

（4）Victim Cache 扩展

- 在直接映射 DM 的组织方式下，分别添加 4、8、16 和 32 个单元的 Victim Cache，此时 Cache 的命中率（包括 Victim Cache 的命中）各是多少？所有 Cache 命中里，由 Victim Cache 贡献的命中比例为多少？
- 备注：假设 Victim Cache 使用全关联的组织方式与 LRU 替换策略，块大小与原 Cache 块大小一致。原 Cache 的配置保持不变。

（5）MRU（Most Recently Used）路预测

- 如果使用 2-way、4-way、8-way、16-way 组关联 SA 方法组织 Cache，当使用 MRU （Most Recent Use）路预测方法时，Cache 的一次命中率（first hit）、非一次命中率（non-first hit）、总的命中率分别为多少？

（6）Multi-column 路预测

- 如果使用 2-way、4-way、8-way、16-way 组关联 SA 方法组织 Cache，当使用 Multi-column 预测方法时，Cache 的一次命中率（first hit）、非一次命中率（non-first hit）、总的命中率分别为多少？并计算该方法下对 bit vector 的平均搜索长度是多少？

（7）路预测方法比较与分析

- 比较说明 MRU 路预测方法、Multi-column 预测方法的优缺点，并分析背后的原因。

Cache 采取的默认配置如下：

- Cache 大小：256 Kbytes
- Block 大小：32 Bytes
- 内存地址宽度：64 bits
- Cache 替换策略：LRU (Least Recently Used)
- 缺失时写策略：写分配

实验所提供的 trace 文件夹内包含四个内存访问记录文件。每行记录包含访问类型（读/写）和访问地址。

二、 提交说明和评分标准

本实验总分 10 分，提交内容如下：

（1）实验报告（5 分），包括：

a) 简要说明 Cache 模拟器及各类设计策略的实现方法

b) 各实验项目要求的 Cache 命中率结果与思考题分析

（2）代码实现（5 分），包括：

a) 源代码

b) README 文件，说明如何运行模拟程序，并获得实验结果（包括命中率统计、缺失访问的日志文件）

本课程的迟交规则如下：

- 实验时间：共 3 周。
- 若在截止后两周内补交，可获得最高 80%的原始分数。
- 超过两周未提交者，视为未完成该实验。