Cpu cache模拟器实验报告

报告中至少包含以下内容：

(a) 模拟器设计思想；

(b) 测试数据以及详细的测试报告；

(c) 作业感悟

1.模拟器设计思想

本cache模拟器按照作业要求，实现了保证双核cpu的 cache一致性四状态协议，同时还有自行发挥部分。

1.1 模拟器结构设计

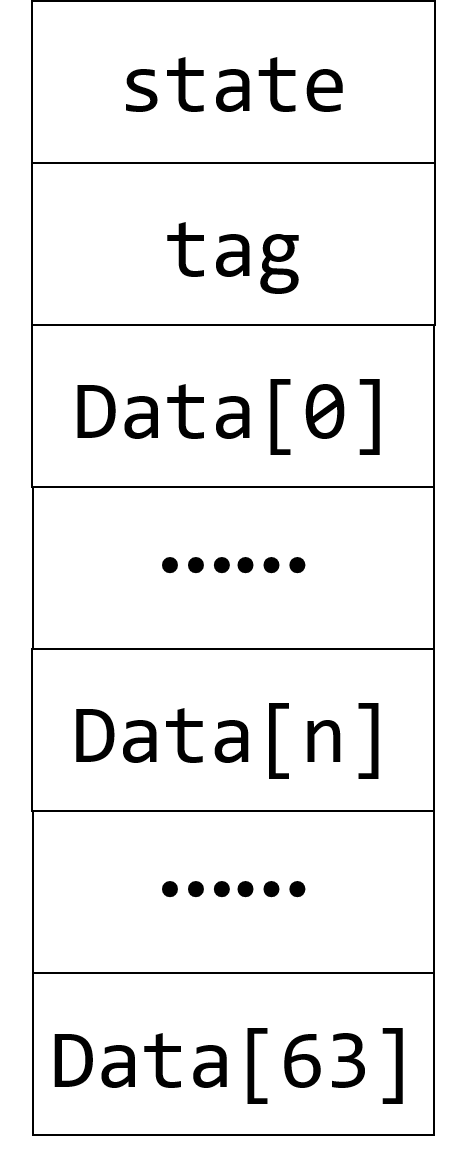
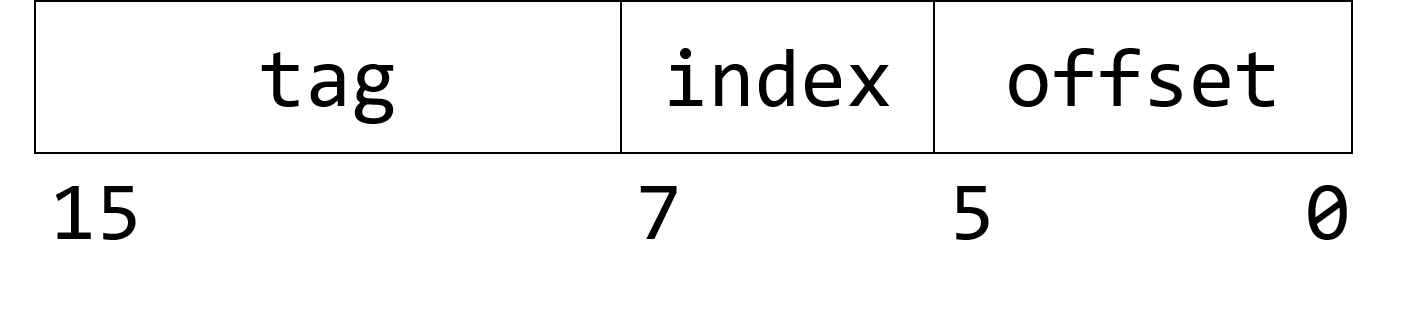
本模拟器有两个cache，单个Cache中有八个cache line，cache line示意图如图1.1.1所示，cache总容量为512Byte。

图 1.1.2

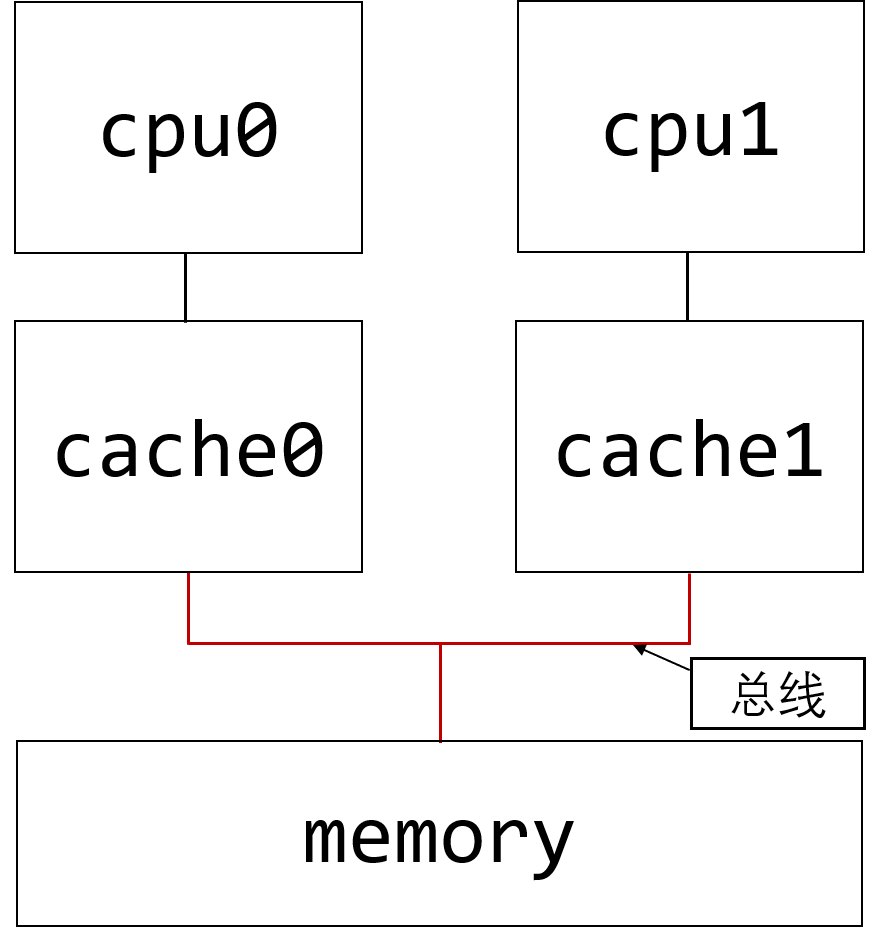


图 1.1.3

图 1.1.1

模拟器memory的地址从0x0000-0x7FFF，每个地址可以存储1Byte的数据，容量为32KB。cache与memory采用两路组相联的映射策略，cache被分为四组。主存地址共十五位，其中的tag，index，offset位如图1.1.2所示。

memory和cache使用总线连接，如图1.1.3所示。总线的设计如图1.1.4所示。

1.2 模拟器功能设计

模拟器采用Write Once策略，即第一次写采用WT策略，之后的写采用WB策略。

cpu读取指令文件之后，将地址解码为tag部分，index部分和offset部分。接着cpu使用cache的目录表判断自己是否命中。

1.2.1 cache命中所需内存数据时，模拟器的运行情况

执行读操作，则直接读取数据即可，状态保持不变；

执行写操作，则需要根据命中的cache line的状态执行不同的操作。

若其状态为E或者M，则直接修改其中对应offset的数据，并使其状态进入或保持M。无需修改另一个cache中可能命中的cache line的状态，因为当前cache line 进入E状态时，已经广播过了，另一个cache已经执行过远程写的响应了。因为不是第一次写，所以无需修改memory中的数据；

若其状态为S，则首先修改自己的状态为E，并且更新数据，接着向总线广播写操作、写的数据、以及数据有效信号和地址，同时将总线状态信号拉高，表示已占用。如上的总线信号对另一个cache来说是远程写操作，它首先判断自己是否命中对应地址。若命中，则置对应cache line的状态为I；若不命中，则无操作。memory监听到总线上的数据有效，则主动更新对应地址的数据，与发起总线占用的cache更改后的数据保持一致。最后释放总线，所有信号都置无效。

1.2.2 cache不命中所需内存数据时，模拟器的运行情况

第一步需要在cache中开辟一个cache line，用于保存数据。第二步从总线上加载数据。第三步才是cache的读写操作。

第一步首先判断第index组有无状态为I的cache line。若有则直接将其tag修改为本次操作地址的tag；若没有，则使用随机替换策略，随机选择其中一个cache line来占用，若要被占用的cache line 处于E或者S状态，则直接修改其tag为本次操作tag，因为memory中保存有相同的数据，所以不用担心数据丢失。若被占用的cache line处于M状态，则需要先将其数据写回到memory中，不能直接替换，当前cache发起总线写操作，总线地址、数据都是要被替换的cache line的，数据有效信号拉高，总线的目标器件地址为memory。memory监听到上述信号之后，从总线上取下数据，并将总线释放，所有信号置无效。最后将被替换的cache line的tag替换为本次操作的tag。

第二步需要从总线上读取数据，所以发起总线读请求，总线地址设为当前要操作的主存地址，总线目标器件地址设为另一个cache的器件地址。另一个cache监听到如上信号之后，首先判断自己是否命中总线上的主存地址，若不命中则置总线的response信号为-1，这意味着交由主存向总线上写数据。若命中，则response信号置为cache line在组中的位置（在本设计中，一个cache组有两个cache line，response为0或1），总线的state\_response信号为cache line的状态，的，则它要将数据写到总线上，以供memory或发起请求的cache读取（M状态时，memory也要从主存上读数据）。发起读请求的cache从总线上拿走数据，并判断response信号是否为-1，若是，则表示该数据与内存共享，置cache line的状态为E；若不是-1，则表示该数据与其他cache以及memory共享，置cache line状态为S。

第三步，cpu对cache进行读写操作。若读，则直接从cache中读走数据即可。若写，则将自己的cache line的状态置为E，并将数据写入总线，让主存读取，另一个cache执行远程写的响应操作。

上述步骤的流程如图1.2.1所示。

若两个核心同时对同一内存地址分别进行读或写，则执行写命令的核心优先执行，以保持读到的数据是最新的。

2模拟器测试