

（深圳）

实验作业

开课学期： 2022春季

课程名称：计算机组成原理（实验）

实验名称： 直接映射Cache设计

实验性质： 综合设计型

实验学时： 4 地点： T2612

学生班级： 11

学生学号： 200111132

学生姓名： 吴桐

作业成绩：

实验与创新实践教育中心制

2022年5月

|  |
| --- |
| 1. Cache模块设计 |
| （画出读、写的状态转移图，并描述状态之间的转移关系和转移条件、以及每个状态需要完成什么操作。） |
| 1. **调试报告** |
| （仿真截图及时序分析，要求包含读命中、读缺失、写命中及写缺失共四种情况的分析，且每种情况需列举2个测试用例进行分析。）  1.仿真通过  （这里由于写测试每次都要刷新一下，无法检测到cache内部是否及时更新，所以这里我暂时在下一个测试更改一下driver文件，不进入缓冲状态，以便可以测出是否更新了cache内部）  可以从输出中看出cache也被及时的修改了  二.波形分析  （后面波形中的current\_state 为自己调出波形cache中的状态，而不是测试driver的状态）  ①读命中  1.  读命中从ready态接受信号后，变成2 check态，随后hit=1检测到命中，cache读出数据11与检验数据一致，然后直接跳回ready态准备下一次读写  2.  重新找了另一个读缺失后的，读命中从ready态接受信号后，变成2 check态，随后hit=1检测到命中，cache读出数据33与检验数据一致，然后直接跳回ready态准备下一次读写  ②读缺失  1.  Cur状态0为ready 进入2（check）后，进入了1（refill）可以看出读缺失，需要重新进入主存中获取数据。  在一定周期后，rvalid\_mem2cache高电位，同时wea写使能有效，即下一个周期能够cache写入新数据。  下一个周期重新回答2（check），输出数据cache\_rdata为26和trace\_rdata验证数据相同。并且hit = 1,为读命中，随后返回ready准备下一次读写  2.  之前读一直命中，当新的读指令来了之后，Cur状态0为ready 进入2（check）后，进入了1（refill）可以看出读缺失，需要重新进入主存中获取数据。  在一定周期后，rvalid\_mem2cache高电位，同时wea写使能有效，即下一个周期能够cache写入新数据。  （之前那个图结果放大一下）  下一个周期重新回答2（check），输出数据cache\_rdata为128和trace\_rdata验证数据相同。并且hit = 1,为读命中，随后返回ready准备下一次读写  ③写命中  1.  写请求发出后，进入tagcheck状态，发现hit，写命中  Wea生效，写入cache，并且写入主存，拉高cache\_whit用于后续判断写命中  下一个状态进入flush状态，进行一个一定不会读命中的读，看到cache中data变化，当这个读结束（命中后）重新进入正常读状态  读取之前写的地址后，重新读命中后数据为写入的255，检测成功  2.  写请求发出后，进入tagcheck状态，发现hit，写命中，此时cache数据为内部之前存在的45  Wea生效，写入cach和写入主存，并且拉高cache\_whit用于后续判断写命中  下一个状态进入flush状态，进行一个一定不会读命中的读，看到cache中data变化，当这个读结束（命中后）重新进入正常读状态  读取之前写的地址后，重新读命中后数据为写入的255，检测成功  ④ 写缺失  1.  C:\Users\Wufisher\AppData\Roaming\Tencent\Users\1105889424\QQ\WinTemp\RichOle\~E%VPL~_~L2~X7FC496Y1FJ.png  这里特别拿这个有毛刺的来说明一下原因（由于test文件特殊的地址路径以及输入周期延迟造成，并非程序原因，不会影响正确结果）  这个毛刺是由于  上一次的写缺失后，tag高位多为1，下一次再准备写缺失地址的时候，会在进入tagcheck后才更换新的tag地址（cache更新新的tag要延迟一个周期），（如图）由于之前写缺失地址cachetag的巧合，所以这里会有一瞬间hit为1，但是组合逻辑下后面新的输入地址后瞬间改变，不会影响到下一次状态的正确转移和操作判断  正常分析，当hbit判断为0的时候，下个周期转回ready态，然后进入flush状态刷入新的一行（高位tag多为0），随后再刷入原地址，读出数据，与trace\_data检测相同，即读出该地址原来未修改的主存数据。验证未对读缺失处理  2.  C:\Users\Wufisher\AppData\Roaming\Tencent\Users\1105889424\QQ\WinTemp\RichOle\GX6~)UMH)]O~G%45)EJ2XEQ.png  这个就是没有毛刺的情况  从下图tag变化也可以看出和之前有毛刺的不一样，但是都不影响结果  C:\Users\Wufisher\AppData\Roaming\Tencent\Users\1105889424\QQ\WinTemp\RichOle\SVYL_XRO@6[0GU)HIJ(MCZH.png  当hit判断为0的时候，下个周期转回ready态，然后进入flush状态刷入新的一行（高位tag多为0），  随后再刷入原地址，读出数据，与trace\_data检测相同，即读出该地址原来未修改的主存数据。验证成功未对读缺失处理 |