TLB Cache AXI 总线的五级流水 CPU

姓名：梁朝阳 2311561　**专业**：密码科学与技术

目录

摘要

关键词： MIPS、五级流水线、CP0、异常处理、中断、协处理器

1 实验要求

在现有的五级流水线 CPU 的基础上，增加 TLB 模块和 cache 模块。

1. 本次实验是学期末综合实验，主要考察大家对 TLB 和 cache 的理解和应用，大家根据自己的能力和时间往后做即可，不强制要去全部功能都做出来。
2. TLB 部分推荐完成 TLB 模块设计和 TLB 相关指令和 CP0 寄存器部分，例外支持部分大家自己把握。
3. Cache 部分至少设计出 ICache 和 DCache，并尝试在 CPU 中集成测试，其他部分大家自己把握。

2 CPU 全局设计图

3 实验原理与设计

3.1 TLB 模块

3.1.1 TLB 参数设计

设计采用全相联 TLB，表项数为 4。页大小固定为 4KB（VPN=addr[31:12]，offset=addr[11:0]），每条表项包含 VPN、PPN、valid、dirty 位。采用固定映射（PPN=VPN）并在复位/首次时钟初始化 valid/dirty 为 1，保证基本访问稳定可用。异常编码支持 TLBL/TLBS/Modify，优先级为 miss/invalid 优先于 modify。

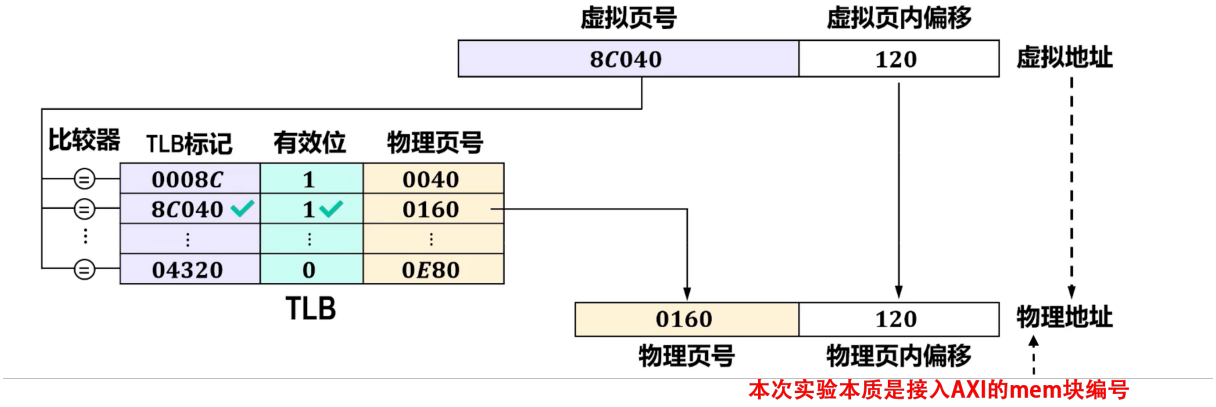

图 3.1: TLB 地址转换 (设计图)

TLB（Translation Lookaside Buffer）用于加速虚拟地址到物理地址的转换。流水线访存时先用虚拟页号（VPN）在 TLB 中查找，如果命中则得到物理页号（PPN）并拼接页内偏移形成物理地址；若未命中或表项无效则触发 TLB 相关异常。

加了 TLB 后的一大改变 之前的 AXI 接到的 mem 我只写了为 256 个 word。实际访问时仅使用地址低位作为索引，高位地址会发生回绕（alias），因此逻辑地址范围大于物理存储容量时会映射到同一片存储区域。（简单来说就是直接把高位扔了，现在 TLB 可以在某种意义上实现一个简单的地址转换）。

3.2 Cache 模块

3.3 I/D-Cache 参数设计

两个 cache 虽然公能有区别，但是结构类似（设计的参数）：

共同点是直映结构、4 行、16B 行大小（byte offset = addr[1:0]，word offset = addr[3:2]，index = addr[5:4]，tag = addr[31:6]）

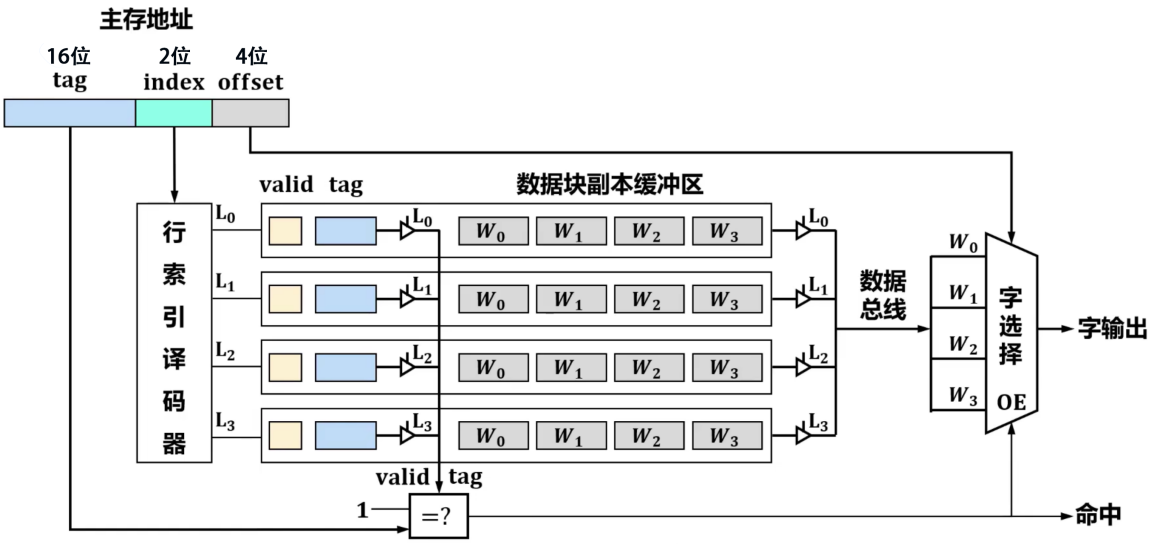

图 3.2: 直接映射 Cache 结构设计图

具体到每个 Cache 的具体实现区别：

1. I-Cache 采用直映结构，4 行，行大小 16B（4 个 word）。索引为 addr[5:4]，标记为 addr[31:6]，字偏移为 addr[3:2]。miss 时发起 burst 长度 4 的 AXI 读事务，填充后更新 tag 与 valid。接口采用 req/resp 握手，支持流水暂停等待。

2. D-Cache 采用直映结构，4 行，行大小 16B（4 个 word），索引与标记划分同 I-Cache。写策略为写直达 + 写分配：store 未命中先 refill，再更新 cache 行并发起单次 AXI 写；load 未命中则 refill 后返回数据。支持字/字节访问，字节写通过读-改-写合并实现，LB/LBU 按符号/无符号扩展返回。

3.3.1 I/D-Cache 区别

I-Cache 仅服务取指路径，属于只读缓存。取指地址先在 I-Cache 中查找，命中则直接返回指令；未命中则触发一次行填充（refill），从指令存储器按 cache 行大小进行 burst 读入，再返回所需指令。由于取指不涉及写操作，I-Cache 不需要脏位与写回策略，实现更简单、时序更稳定。

D-Cache 用于数据访存，支持读写操作。对 load 命中直接返回数据；对 store 命中先更新 cache 行，再根据写策略写到主存。未命中时需要行填充，之后再完成当前读/写。为了简化教学实现，本设计采用直映结构与写直达策略，避免复杂的替换与写回。总结可以为：

1. I-Cache 只读，只做取指；不需要写策略、字节合并、写直达

2. D-Cache 读写都要处理，包含写直达、写分配、LB/LBU/SB 的字节处理

4 实验结果

5 实验总结