实验 7-2 报告

第 27 小组
姚子薇、陈乐滢

一、实验任务

在 CPU 中增加 TLBR、TLBWI、TLBP 指令，增加 Index、EntryHi、EntryLo0、EntryLo1 、PageMask 这五个 CP0 寄存器，增加 32 项 TLB 结构。验证方法即运行专用功能测试 tlb_func，仿真及上板通过前 7 项测试。

二、实验设计

（一）MMU 模块设计及 TLB 相关指令实现

接口名称	输入/输出	接口说明
clk	input	时钟。
resetn	input	复位信号，低电平有效。
vaddr_inst[31:0]	input	指令的虚拟地址。
vaddr_data[31:0]	input	数据的虚拟地址。
paddr_inst[31:0]	output	（经过 TLB 转换的）指令的物理地址。
paddr_data[31:0]	output	（经过 TLB 转换的）数据的物理地址。
index_min[4:0]	input	来自 CP0 index 寄存器的输入，只保留低五位。
entryhi_min[31:0]	input	来自 CP0 entryhi 寄存器的输入，格式与 CP0 中的定义相同。
entrylo0_min[31:0]	input	来自 CP0 entrylo0 寄存器的输入，格式与 CP0 中的定义相同。
entrylo1_min[31:0]	input	来自 CP0 entrylo1 寄存器的输入，格式与 CP0 中的定义相同。
pagemask_min[31:0]	input	来自 CP0 pagemask 寄存器的输入，格式与 CP0 中的定义相同。
index_mout[31:0]	output	写入 CP0 index 寄存器的输出，格式与 CP0 中的定义相同。
entryhi_mout[31:0]	output	写入 CP0 entryhi 寄存器的输出，格式与 CP0 中的定义相同。
entrylo0_mout[31:0]	output	写入 CP0 entrylo0 寄存器的输出，格式与 CP0 中的定义相同。
entrylo1_mout[31:0]	output	写入 CP0 entrylo1 寄存器的输出，格式与 CP0 中的定义相同。
pagemask_mout[31:0]	output	写入 CP0 pagemask 寄存器的输出，格式与 CP0 中的定义相同。
de_is_tlbr	input	来自 CPU 译码级的信号，表示当前处在译码级的是 tlbr 指令。
de_is_tlbp	input	来自 CPU 译码级的信号，表示当前处在译码级的是 tlbp 指令。
de_is_tlbwi	input	来自 CPU 译码级的信号，表示当前处在译码级的是 tlbwi 指令。

表 1. MMU 模块接口说明

尽管我们的 axi 总线的 CPU 已经完成，但是因为加了总线之后的 cpu 效率很低，结构也有点乱，所以我们还是使用了 sram 接口完成这个实验。

这次主要实现的是添加 MMU 模块，用寄存器堆实现 32 项 TLB，在 CP0 中添加 TLB 的 entryhi，entrylo0/1，pagemask，index 寄存器。添加三条 TLB 指令：tlbp, tlbr, tlbwi。

tlbp 指令是根据 entryhi 寄存器查找 32 项 TLB 寻找匹配项，根据查找结果更新 index 寄存器，如果找到了更新 index[4:0]为项号，没找到写 index 的最高位为 1。我们将 32 项 TLB 对应为一个 32 位的变量，找到哪一项对应就将该位置 1，得到一个独热的数，然后用一个 32-5 的编码器将其对应为项号，写入 CP0 模块中的 Index 寄存器。

tlbr 指令是根据 index 寄存器中的[4:0]位读取 TLB 中对应项，将读出来的值对应赋值给 CP0 中的 entryhi，entrylo0/1，pagemask 寄存器。

tlbwi 指令和 tlbr 指令相反，是将 CP0 中的 entryhi，entrylo0/1，pagemask 寄存器中的值，对应赋值给 TLB 表项中第 index[4:0]项。

具体接口在上表中列出。

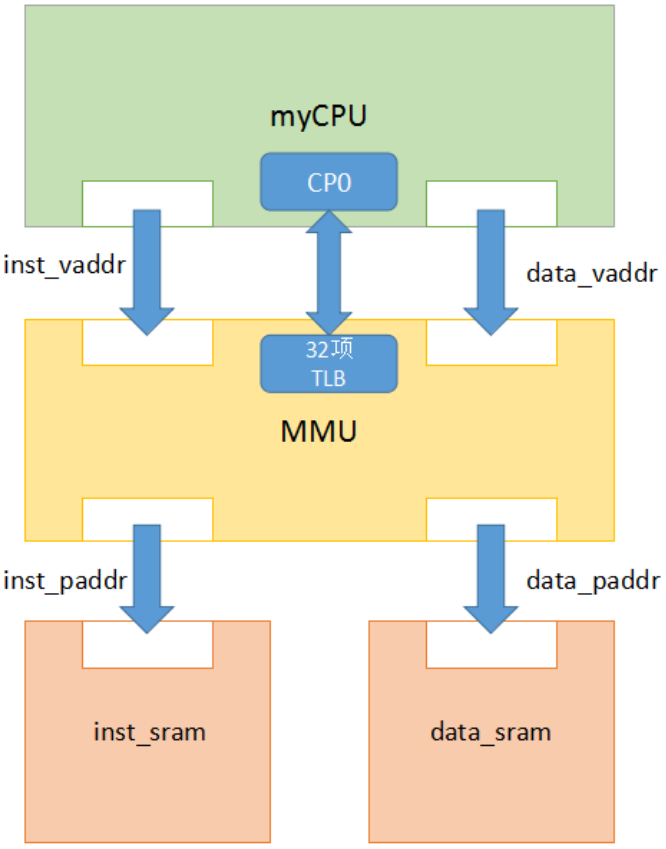

图 1. MMU 与 CP0 及 SRAM 的交互示意图

虚实地址转换在此次实验中并没有被测试到，因为虚地址的范围均在 unmapped 的区域内，并不需要映射。我们暂时的设计如上图，inst 和 data 的查询线路各有一条。

三、实验过程

（一）实验流水账

2017.12.17，10:00 - 11:30，分析任务书，添加 mmu 模块。

2017.12.17，14:00 - 15:30，进一步添加接口，在 cpu 中添加 tlb 相关指令。

2017.12.17，19:00 - 20:30，完成代码，调试至仿真、上板通过。

（二）错误记录

1、错误 1

（1）错误现象

前 5 个测试通过，6、7error。

（2）分析定位过程

bfc00738:　40885000　mtc0t0,c0_entryhi

bfc0073c:　40820000　mtc0v0,c0_index

bfc00740:　42000002　tlbwi

在 tlbwi 的下一拍，entryhi 寄存器又被写了一次，导致写入 TLB 表项的内容有误。

entryhi 寄存器更新的条件应该只有 mtc0 和 tlbwi，检查代码，此时被更新的原因是手滑把 tlbwi 写成了 tlbr。

（3）错误原因

CP0 模块中的 entryhi, entrylo, pagemask 寄存器应该是在 tlbwi 的时候进行更改，原代码里写成了写成了 tlbr。

（4）修正效果

修改后这个问题解决，测试 6 还有别的问题。

2、错误 2

（1）错误现象

前 5 个测试通过，6、7error。

（2）分析定位过程

bfc00758:　42000001　tlbr

观察波形发现这条指令的 exe 级时 CP0 内的寄存器都变成了 xxxx，而且 mmu 里的接口有好多 xxxx，tlbr 的接口是 x，index_min 是 x，基本可以估计是接口定义有误，检查后发现少写了几个接口。

（3）错误原因

手滑，顶层接口定义错误。

（4）修正效果

修改后测试 6 通过。

3、错误 3

（1）错误现象

前 6 个测试通过，7error。

（2）分析定位过程

经检查发现分歧出现在 tlbp 指令。tlbp 指令后 index 寄存器的值变成了 0x80000002,但是本应该为 0x2。

index 的最高位是 1，说明判断为未找到，写了 P 位，但是实际情况是找到了，而且也赋值了 0x2。检查代码发现判断是否找到的赋值语句有误。

（3）错误原因

assign index_mout[31] = ~(|lkup_result_32)这一句要判断 tlb 中是否找到相应项，一开始忘加取反了。

（4）修正效果

修改后这个问题解决，测试 7 还有别的错误。

4、错误 4

（1）错误现象

前 6 个测试通过，7error。

（2）分析定位过程

/media/sf_SHARE/tlb_func_v0.3/tlb_func/inst/n7_tlbp.S:15

bfc009dc:42000008　　tlbp

/media/sf_SHARE/tlb_func_v0.3/tlb_func/inst/n7_tlbp.S:16

bfc009e0:40040000　　mfc0　　a0,c0_index

由波形发现是 mfc0 没写进去，写回地址变成了 0 号寄存器，从 wb 级倒推发现 de_dest 是 1，下一级的 exe_dest 就变成了 0。变成 0 的原因是 exe_eret 的条件没写全，此时被误判为这个情况。

（3）错误原因

exe 级是 eret 的条件写的不全，导致其他指令也被判断成了 eret,写回地址被赋值成了 0 号寄存器。

（4）修正效果

修改后仿真 7 个测试全部通过，上板也通过。

四、实验总结

（一）组员：姚子薇

emmm，一开始知道没有 trace 比对很慌张，写了之后发现 7-2 的确不难，错了之后基本上能很明确的知道错误在哪条指令附近。估计了下 7-3 会费点劲，不过自从被总线实验虐过了之后，我觉得我们的承受能力应该还是有那么一点点的提高吧……

（二）组员：陈乐滢

刚刚加完 CP0 里面的五个寄存器还没有写完 MMU 就过了 5 个测试很开心（虽然也在预料之中）！还好在之前的 trace 比对中也经历过很多次错误定位了，所以（在单个测试很短的情况下）没有 trace 也还是可以比较轻松的完成的！带总线的 CPU 真的不是亲儿子，不是。