实验报告格式说明：

1. 标题层次建议不超过四级，从第一级开始标号格式依次采用：一、二、三；（一）、（二）、（三）；1、2、3；（1）、（2）、（3）。
   1. 一级标题用小三号字，中文为黑体，英文为Times New Roman，单倍行距，段前段后各0.5行。
   2. 二级标题用四号字，中文为黑体，英文为Times New Roman，单倍行距，段前段后各0.25行。
   3. 三级标题用小四号字，中文为黑体，英文为Times New Roman，单倍行距。
   4. 四级标题用五号字，中文为黑体，英文为Times New Roman，单倍行距。
2. 正文用五号字，中文为宋体，英文为Times New Roman，1.5倍行距。
3. 所有图统一顺序标号，图标题紧挨在图的下方，居中，用小五号字，中文为宋体，英文为Times New Roman，单倍行距，段后0.5行。
4. 所有表统一顺序标号，图标题放在表的上方，居中，用小五号字，中文为宋体，英文为Times New Roman，单倍行距，段前0.5行。
5. **报告形成后删除本模板中所有红色文字！**

**Lab N报告**

学号

姓名

箱子号

一、实验任务（10%）

根据你对实验任务说明的理解，说清楚了这个实验要干什么以及如何检验。

二、实验设计（40%）

**针对Lab3及后续实验**，请参考以下格式完成实验报告。

（一）总体设计思路

阐明总体设计思路，即从系统顶层角度出发，概要性地描述整个系统的工作机制，所需要进行哪些实验设计、完成哪些功能。

在进行本章节描述时，推荐以结构设计图的形式阐述硬件部分，以流程图的形式阐述软件部分。

**如果是自己的新设计，最好给出结构设计图！**

如果实验设计比较复杂，那么最好进行模块划分，挑选重要模块进行描述。

（二）重要模块1设计：TLB模块

1. 工作原理

单个TLB模块的设计根据需要分为读、写和查找三部分，且同时支持取指和访问同时进行，因此有两套查找端口。TLB模块内部是一个二维组织结构的查找表，查找表的每一项分为两部分，一部分存储的信息既参与读写又参与查找比较，另一部分只参与读写。

1. 接口定义

| **名称** | **方向** | **位宽** | **功能描述** |
| --- | --- | --- | --- |
| clk | IN | 1 | 时钟信号 |
| 取指查找端口 | | | |
| s0\_vppn | IN | 19 | 来自访存虚地址的31..13为 |
| s0\_va\_bit12 | IN | 1 | 来自访存虚地址12位 |
| s0\_asid | IN | 10 | 来自CSR.ASID的ASID域 |
| s0\_found | OUT | 1 | 用于判定是否产生TLB重填异常 |
| s0\_index | OUT | 16 | 记录命中在查找表的第几项 |
| s0\_ppn | OUT | 20 | 用于产生最终的物理地址 |
| s0\_ps | OUT | 6 | 用于产生最终的物理地址 |
| s0\_plv | OUT | 2 | 和s\_found一起判定是否产生页特权等级不合规异常 |
| s0\_mat | OUT | 2 | TLB模块输出信息，存储访问类型 |
| s0\_d | OUT | 1 | 和s\_found、s\_d一起判定是否产生页修改异常 |
| s0\_v | OUT | 1 | 和s\_found一起判定是否产生页无效异常 |
| 访存查找端口，端口功能与取指查找端口功能相同 | | | |
| s1\_vppn | IN | 19 | 来自访存虚地址的31..13为 |
| s1\_va\_bit12 | IN | 1 | 来自访存虚地址12位 |
| s1\_asid | IN | 10 | 来自CSR.ASID的ASID域 |
| s1\_found | OUT | 1 | 用于判定是否产生TLB重填异常 |
| s1\_index | OUT | 16 | 记录命中在查找表的第几项 |
| s1\_ppn | OUT | 20 | 用于产生最终的物理地址 |
| s1\_ps | OUT | 6 | 用于产生最终的物理地址 |
| s1\_plv | OUT | 2 | 和s\_found一起判定是否产生页特权等级不合规异常 |
| s1\_mat | OUT | 2 | TLB模块输出信息，存储访问类型 |
| s1\_d | OUT | 1 | 和s\_found、s\_d一起判定是否产生页修改异常 |
| s1\_v | OUT | 1 | 和s\_found一起判定是否产生页无效异常 |
|  |  |  |  |
| invtlb\_valid | IN | 1 | 指令有效 |
| invtlb\_op | IN | 5 | 标识invtlb的具体操作类型 |
|  |  |  |  |
| we | IN | 1 | 写使能输入信号 |
| w\_index | IN | 16 | 写地址 |
| w\_e | IN | 1 | 写入的TLB表项信息 |
| w\_vppn | IN | 19 | 写入的TLB表项信息 |
| w\_ps | IN | 6 | 写入的TLB表项信息 |
| w\_asid | IN | 10 | 写入的TLB表项信息 |
| w\_g | IN | 1 | 写入的TLB表项信息 |
|  |  |  |  |
| w\_ppn0 | IN | 20 | 写入的TLB表项信息 |
| w\_plv0 | IN | 2 | 写入的TLB表项信息 |
| w\_mat0 | IN | 2 | 写入的TLB表项信息 |
| w\_d0 | IN | 1 | 写入的TLB表项信息 |
| w\_v0 | IN | 1 | 写入的TLB表项信息 |
|  |  |  |  |
| w\_ppn1 | IN | 20 | 写入的TLB表项信息 |
| w\_plv1 | IN | 2 | 写入的TLB表项信息 |
| w\_mat1 | IN | 2 | 写入的TLB表项信息 |
| w\_d1 | IN | 1 | 写入的TLB表项信息 |
| w\_v1 | IN | 1 | 写入的TLB表项信息 |
|  |  |  |  |
| r\_index | IN | 16 | 读地址 |
| r\_e | OUT | 1 | 输出读的结果 |
| r\_vppn | OUT | 19 | 输出读的结果 |
| r\_ps | OUT | 6 | 输出读的结果 |
| r\_asid | OUT | 10 | 输出读的结果 |
| r\_g | OUT | 1 | 输出读的结果 |
|  |  |  |  |
| r\_ppn0 | OUT | 20 | 输出读的结果 |
| r\_plv0 | OUT | 2 | 输出读的结果 |
| r\_mat0 | OUT | 2 | 输出读的结果 |
| r\_d0 | OUT | 1 | 输出读的结果 |
| r\_v0 | OUT | 1 | 输出读的结果 |
| r\_ppn1 | OUT | 20 | 输出读的结果 |
| r\_plv1 | OUT | 2 | 输出读的结果 |
| r\_mat1 | OUT | 2 | 输出读的结果 |
| r\_d1 | OUT | 1 | 输出读的结果 |
| r\_v1 | OUT | 1 | 输出读的结果 |

1. 功能描述
2. 读写功能设计

由于TLB模块的读写方式与寄存器堆的读写方式类似，因此一样是同步写异步读，只需将TLB的相关信息使用非阻塞赋值写入或使用组合逻辑读出即可。其中需要注意由于在LoogArch精简版中只支持4KB和4MB两种页大小，TLB模块内部只用1bit来存放页大小信息，因此需要进行一个简单的转换。

assign r\_ps   = tlb\_ps4MB[r\_index] ? 6'd21 : 6'd12;

1. 查找功能设计

在进行查找时，查找流程并非串行化，而是同时比较所有项，在讲义中已经给出了代码，我们在这里使用generate进行代码的缩减。要判断是否匹配，则对每一个页表项，比较vppn的高十位，而后根据页大小是4MB还是4KB判断是否需要比较vppn的低十位。具体来说，是在LoogArch精简版中，每个页表项存放了相邻的一奇偶相邻页表信息，所以 TLB 页表项中存放虚页号的是系统中虚页号/2 的内容，也就是2MB大小只需比较32-21-1=10位，4KB只需比较32-12-1=19位。虚页号的最低位不需要存放在 TLB 中，查找 TLB 时在根据被查找虚页号的最低位决定是选择奇数号页还是偶数号页的物理转换信息。

assign s1\_odd = tlb\_ps4MB[s1\_index] ? s1\_vppn[8] : s1\_va\_bit12;

最后比较全局标志位g和地址空间表示位asid（区分不同进程中同样的虚地址），判断是否属于本进程。若以上全部满足，则说明命中。

generate for(i = 0; i < TLBNUM; i = i + 1) begin

        assign match0[i] = (s0\_vppn[18:9] == tlb\_vppn[i][18:9])

                        && (tlb\_ps4MB[i] || s0\_vppn[8:0] == tlb\_vppn[i][8:0])

                        && ((s0\_asid == tlb\_asid[i]) || tlb\_g[i]);

        assign match1[i] = (s1\_vppn[18:9] == tlb\_vppn[i][18:9])

                        && (tlb\_ps4MB[i] || s1\_vppn[8:0] == tlb\_vppn[i][8:0])

                        && ((s1\_asid == tlb\_asid[i]) || tlb\_g[i]);

end

endgenerate

最后对于INVTLB指令的支持，在讲义中也已经大致写出，将该指令个操作分成四种“子匹配”判断条件，而后再进行组合。由于op为5位，因此设置的invtlb\_mask为32位，需要将高25为置0。

（三）重要模块2设计：XXX模块

……

三、实验过程（50%）

（一）实验流水账

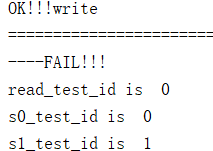
记录哪一天，几点到几点，做了什么事，结果如何。事情不要展开来写。

（二）错误记录

重点记录调试过程和机理分析。请以**图文结合**的方式进行描述，如有波形图应当**分组（Group）分明、分割（Divider）清晰、有标志线（Marker）指示关键时刻**。

1、错误1：页大小错误

（1）错误现象



在新的测试文件中进行测试，发现原代码无法通过。

（2）分析定位过程

根据老师在群里所发，结合指令集手册，可以看出原来对于4MB页大小理解出现问题。通过指令集手册中所说，龙芯架构 32 位精简版只支持 4KB 和 4MB 两种页大小，对应 TLB 表项中的 PS 值分别是 12 和 21。其中4MB页大小对应的是透明大页的页表项，其在填入TLB过程中等分为2个2MB大小相同页表属性的表项。因此需要将4MB页大小相关的代码进行修改。

（3）修正效果

首先是检索匹配项时，4MB页大小需要匹配高十位而不是高九位。而后对于判断奇偶页，是对于虚地址的第22位进行判断，也就是vppn的第九位。

assign s0\_odd = tlb\_ps4MB[s0\_index] ? s0\_vppn[8] : s0\_va\_bit12;

在cond匹配条件时，需要更改cond4的匹配项，与match的更改类似。

assign cond[i][3] = (s1\_vppn[18:9] == tlb\_vppn[i][18:9]) && (tlb\_ps4MB[i] || s1\_vppn[8:0] == tlb\_vppn[i][8:0]);

最后要更改所有ps域的赋值，当是4MB页大小时，ps赋值为21。

2、错误2：错误简介命名

……

四、实验总结（可选）

供同学们吐槽之用。

……