实验 8 报告

学号：2017K8009922027 2017K8009929011
姓名：张磊、郭豪
箱子号：39

一、实验任务（10%）

任务：a) 按照课程讲义规定的接口设计 TLB 模块；

b) 通过简单的读写和查找测试；

二、实验设计（40%）

（一）总体设计思路

1. 根据 TLB 所需的两个查询端口，一个读端口和一个写端口，设计对应的模块接口；

2. TLB 结构采用寄存器型的向量数组存储，共 16 个 TLB 项；

3. 读和查询操作的输出结果都可以直接采用组合逻辑得出；

4. 写操作在时钟上升沿时，根据写使能判断时候需要写入，如果写使能为 1，则将对应数据写入对应的 TLB 项中；

（三）重要模块 1 设计：MATCH 模块；

1、工作原理

对所有项同时进行比较，如果匹配成功则将对应位置 1；

```
assign match0[ 0] = (s0_vpn2==tlb_vpn2[ 0]) && ((s0_asid==tlb_asid[ 0]) || tlb_g[ 0]);
assign match0[ 1] = (s0_vpn2==tlb_vpn2[ 1]) && ((s0_asid==tlb_asid[ 1]) || tlb_g[ 1]);
assign match0[ 2] = (s0_vpn2==tlb_vpn2[ 2]) && ((s0_asid==tlb_asid[ 2]) || tlb_g[ 2]);
assign match0[ 3] = (s0_vpn2==tlb_vpn2[ 3]) && ((s0_asid==tlb_asid[ 3]) || tlb_g[ 3]);
assign match0[ 4] = (s0_vpn2==tlb_vpn2[ 4]) && ((s0_asid==tlb_asid[ 4]) || tlb_g[ 4]);
assign match0[ 5] = (s0_vpn2==tlb_vpn2[ 5]) && ((s0_asid==tlb_asid[ 5]) || tlb_g[ 5]);
assign match0[ 6] = (s0_vpn2==tlb_vpn2[ 6]) && ((s0_asid==tlb_asid[ 6]) || tlb_g[ 6]);
assign match0[ 7] = (s0_vpn2==tlb_vpn2[ 7]) && ((s0_asid==tlb_asid[ 7]) || tlb_g[ 7]);
assign match0[ 8] = (s0_vpn2==tlb_vpn2[ 8]) && ((s0_asid==tlb_asid[ 8]) || tlb_g[ 8]);
assign match0[ 9] = (s0_vpn2==tlb_vpn2[ 9]) && ((s0_asid==tlb_asid[ 9]) || tlb_g[ 9]);
assign match0[10] = (s0_vpn2==tlb_vpn2[10]) && ((s0_asid==tlb_asid[10]) || tlb_g[10]);
assign match0[11] = (s0_vpn2==tlb_vpn2[11]) && ((s0_asid==tlb_asid[11]) || tlb_g[11]);
assign match0[12] = (s0_vpn2==tlb_vpn2[12]) && ((s0_asid==tlb_asid[12]) || tlb_g[12]);
assign match0[13] = (s0_vpn2==tlb_vpn2[13]) && ((s0_asid==tlb_asid[13]) || tlb_g[13]);
assign match0[14] = (s0_vpn2==tlb_vpn2[14]) && ((s0_asid==tlb_asid[14]) || tlb_g[14]);
assign match0[15] = (s0_vpn2==tlb_vpn2[15]) && ((s0_asid==tlb_asid[15]) || tlb_g[15]);
```

2、功能描述

快速判断是否存在匹配的 TLB 项；

（四）重要模块 2 设计：S0_index 模块；

1、工作原理

不采用多路选择器，而是采用 1 级组合逻辑直接得出对应的 index；

```
assign s0_index = ({4{match0 == 16'b0000_0000_0000_0001}} & 4'd0)
                | ({4{match0 == 16'b0000_0000_0000_0010}} & 4'd1)
                | ({4{match0 == 16'b0000_0000_0000_0100}} & 4'd2)
                | ({4{match0 == 16'b0000_0000_0000_1000}} & 4'd3)
                | ({4{match0 == 16'b0000_0000_0001_0000}} & 4'd4)
                | ({4{match0 == 16'b0000_0000_0010_0000}} & 4'd5)
                | ({4{match0 == 16'b0000_0000_0100_0000}} & 4'd6)
                | ({4{match0 == 16'b0000_0000_1000_0000}} & 4'd7)
                | ({4{match0 == 16'b0000_0001_0000_0000}} & 4'd8)
                | ({4{match0 == 16'b0000_0010_0000_0000}} & 4'd9)
                | ({4{match0 == 16'b0000_0100_0000_0000}} & 4'd10)
                | ({4{match0 == 16'b0000_1000_0000_0000}} & 4'd11)
                | ({4{match0 == 16'b0001_0000_0000_0000}} & 4'd12)
                | ({4{match0 == 16'b0010_0000_0000_0000}} & 4'd13)
                | ({4{match0 == 16'b0100_0000_0000_0000}} & 4'd14)
                | ({4{match0 == 16'b1000_0000_0000_0000}} & 4'd15);
```

2、功能描述

快速得出对应项的 index；

（五）重要模块 3 设计：Write 模块；

1、工作原理

时钟上升沿，如果写使能为 1，则将对应的数据写入对应的 TLB 项；

```
always @(posedge clk)
begin
    if (we)
    begin
        tlb_vpn2[w_index] <= w_vpn2;
        tlb_asid[w_index] <= w_asid;
        tlb_g[w_index]    <= w_g;

        tlb_pfn0[w_index] <= w_pfn0;
        tlb_c0[w_index]   <= w_c0;
        tlb_d0[w_index]   <= w_d0;
        tlb_v0[w_index]   <= w_v0;

        tlb_pfn1[w_index] <= w_pfn1;
        tlb_c1[w_index]   <= w_c1;
        tlb_d1[w_index]   <= w_d1;
        tlb_v1[w_index]   <= w_v1;
    end
end
```

2、功能描述

TLB 数据写入；

三、实验过程（50%）

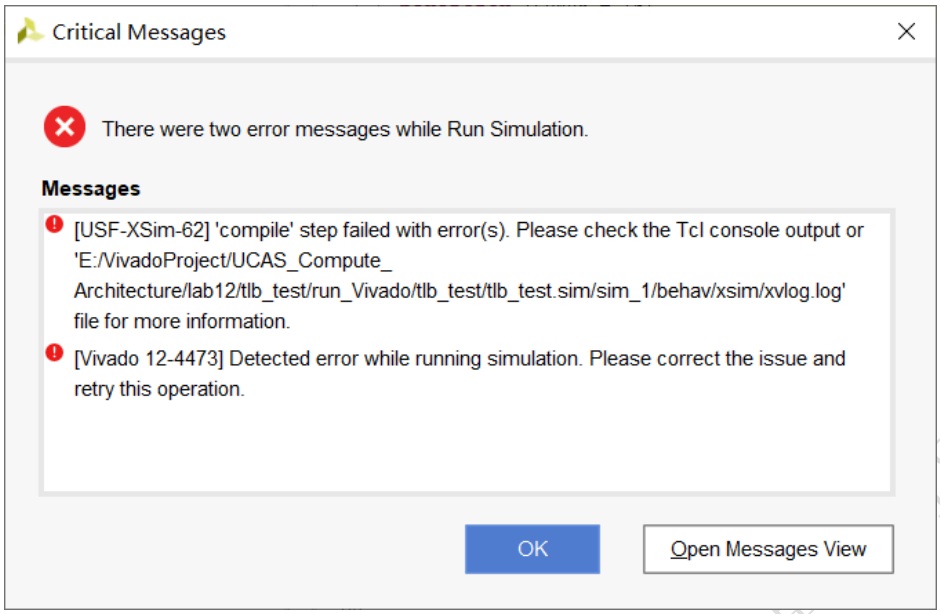
（一）实验流水账

1. 2019 年 11 月 28 日 10:00 - 11:00 完成 rtl 代码编写；
2. 2019 年 11 月 28 日 11:00-2019 年 11 月 28 日 11:00 – 11:30 完成 bug 的修改；

（二）错误记录

1、错误 1：仿真报错

（1）错误现象



（2）分析定位过程

查看 xvlog.log，发现是 rtl 代码的第一行就有问题，所以因该是参数的错误；

```
ERROR: [VRFC 10-1342] root scope declaration is not allowed in verilog 95/2K mode
[E:/VivadoProject/UCAS_Compute_ Architecture/lab12/tlb_test/rtl/tlb/tlb.v:1]
```

（3）错误原因

设置参数的格式错误；

```
1  parameter TLBNUM = 16;
2
3  module tlb
4  //#(
5  //    parameter TLBNUM = 16
6  //)
7  (
8      input clk,
9      // search port 0
10     input  [              18:0] s0_vpn2,
```

（4）修正效果

修正后，顺利开始仿真。

```
1  module tlb
2  #(
3      parameter TLBNUM = 16
4  )
5  (
6      input clk,
```

2、错误 2：查询错误

（1）错误现象

```
OK!!!write
================================================
----FAIL!!!
read_test_id is  5
s0_test_id is 10
s1_test_id is 11
$finish called at time : 4415 ns : File "E:/VivadoP
```

（2）分析定位过程

到 test_bench 里查到出错的原因，发现是 s0_error，然后又对比 s0 的输出结果个对比结果，发现当查询到对应 TLB 项时，输出的 index 值不正确；

（3）错误原因

写代码时疏忽，数据写错，第 4 位本该是 0，结果写成了 1：

```
assign s0_index = ({4{match0 == 16'b0000_0000_0000_0001}} & 4'd0)
                | ({4{match0 == 16'b0000_0000_0000_0010}} & 4'd1)
                | ({4{match0 == 16'b0000_0000_0000_0100}} & 4'd2)
                | ({4{match0 == 16'b0000_0000_0000_1000}} & 4'd3)
                | ({4{match0 == 16'b0000_0000_0001_0000}} & 4'd4)
                | ({4{match0 == 16'b0000_0000_0010_0000}} & 4'd5)
                | ({4{match0 == 16'b0000_0000_0100_1000}} & 4'd6)
                | ({4{match0 == 16'b0000_0000_1000_0000}} & 4'd7)
                | ({4{match0 == 16'b0000_0001_0000_0000}} & 4'd8)
                | ({4{match0 == 16'b0000_0010_0000_0000}} & 4'd9)
```

（4）修正效果

修正后，顺利通过仿真测试。

四、实验总结（可选）

1. 这次实验比较简单，但是写代码的时候还是要细心，尤其是出现大量复制粘贴操作的时候；