实验 5-1 报告

学号：2016K8009929029
姓名：张丽玮

一、实验任务（10%）

第一阶段实现环境的变化，在虚拟 cpu 上完成一个类 SRAM 接口到 AXI 接口的转换桥，方便后续实验。

(1) 完全带握手的类 SRAM 接口到 AXI 接口的转换桥 RTL 代码编写。

(2) 通过简单的读写测试。

二、实验设计（30%）

设计了类 SRAM 接口到 AXI 接口的转换桥。

类 SRAM 接口为 inst\data 双通道，AXI 接口为 r/w 双通道；转换桥内部采用 r/w 双通道独立状态机.

信号	位宽	方向	功能	备注
AXI 时钟与复位信号				
aclk	1	input	AXI 时钟	
aresetn	1	input	AXI 复位，低电平有效	
读请求地址通道，（以 ar 开头）				
arid	[3:0]	master—>slave	读请求的 ID 号	取指为 0 取数为 1
araddr	[31:0]	master—>slave	读请求的地址	
arlen	[7:0]	master—>slave	读请求控制信号，请求传输的长度(数据传输拍数)	固定为 0
arsize	[2:0]	master—>slave	读请求控制信号，请求传输的大小(数据传输每拍的字节数)	
arburst	[1:0]	master—>slave	读请求控制信号，传输类型	固定为 2'b01
arlock	[1:0]	master—>slave	读请求控制信号，原子锁	固定为 0
arcache	[3:0]	master—>slave	读请求控制信号，CACHE 属性	固定为 0
arprot	[2:0]	master—>slave	读请求控制信号，保护属性	固定为 0
arvalid	1	master—>slave	读请求地址握手信号，读请求地址有效	
arready	1	slave—>master	读请求地址握手信号，slave 端准备好接受地址传输	
读请求数据通道，（以 r 开头）				
rid	[3:0]	slave—>master	读请求的 ID 号，同一请求的 rid 应和 arid 一致	指令回来为 0
				数据回来为 1
rdata	[31:0]	slave—>master	读请求的读回数据	
rresp	[1:0]	slave—>master	读请求控制信号，本次读请求是否成功完成	可忽略
rlast	1	slave—>master	读请求控制信号，本次读请求的最后一拍数据的指示信号	可忽略
rvalid	1	slave—>master	读请求数据握手信号，读请求数据有效	
rready	1	master—>slave	读请求数据握手信号，master 端准备好接受数据传输	
写请求地址通道，（以 aw 开头）				
awid	[3:0]	master—>slave	写请求的 ID 号	固定为 1
awaddr	[31:0]	master—>slave	写请求的地址	
awlen	[7:0]	master—>slave	写请求控制信号，请求传输的长度(数据传输拍数)	固定为 0
awsize	[2:0]	master—>slave	写请求控制信号，请求传输的大小(数据传输每拍的字节数)	
awburst	[1:0]	master—>slave	写请求控制信号，传输类型	固定为 2'b01
awlock	[1:0]	master—>slave	写请求控制信号，原子锁	固定为 0
awcache	[3:0]	master—>slave	写请求控制信号，CACHE 属性	固定为 0
awprot	[2:0]	master—>slave	写请求控制信号，保护属性	固定为 0
awvalid	1	master—>slave	写请求地址握手信号，写请求地址有效	
awready	1	slave—>master	写请求地址握手信号，slave 端准备好接受地址传输	
写请求数据通道，（以 w 开头）				
wid	[3:0]	master—>slave	写请求的 ID 号	固定为 1
wdata	[31:0]	master—>slave	写请求的写数据	
wstrb	[3:0]	master—>slave	写请求控制信号，字节选通位	
wlast	1	master—>slave	写请求控制信号，本次写请求的最后一拍数据的指示信号	固定为 1
wvalid	1	master—>slave	写请求数据握手信号，写请求数据有效	
wready	1	slave—>master	写请求数据握手信号，slave 端准备好接受数据传输	
写请求响应通道，（以 b 开头）				
bid	[3:0]	slave—>master	写请求的 ID 号，同一请求的 bid、wid 和 awid 应一致	可忽略
bresp	[1:0]	slave—>master	写请求控制信号，本次写请求是否成功完成	可忽略
bvalid	1	slave—>master	写请求响应握手信号，写请求响应有效	
bready	1	master—>slave	写请求响应握手信号，master 端准备好接受写响应	

以上为这个转换桥的接口设置。

独立状态机一个 always 块控制读，一个 always 块控制写，这样实现读写独立，就可以同时执行。但是因为读写有相关——读后写相关、写后读相关，因此增加了一个阻塞延迟信号。

```
reg         wr_haz;
reg         rw_haz;
```

```
  wr_haz    <= ^w_status && data_addr[31:2] == w_addr[31:2];
```

写后读的阻塞延迟，根据 w_status 判断。

```
 rw_haz    <= ^r_status && data_addr[31:2] == r_addr[31:2];
```

读后写的阻塞延迟，根据 r_status 判断。

三、实验过程（60%）

（一）实验流水账

11 月 28 日，下午 4 点到晚上 10 点，阅读任务书，弄清楚新环境的框架

11 月 29 日，下午 4 点到第二天 2 点，写转换桥代码，测试 xxx

11 月 30 日，下午 6 点到晚上 12 点，debug，通过一部分测试，继续 debug

12 月 1 日，下午 6 点到晚上 10 点，通过测试

（二）错误记录

具体描述实验过程中的错误，环境问题、仿真阶段、上板阶段的都可以记录。

1、错误 1

（1）错误现象

测试点 0pass 之后读取数据一直 xxx

（2）分析定位过程

有 Z 调 Z，有 X 调 X，发现 inst 和 data 的 ok 信号一直是 0

（3）错误原因

逻辑出了问题，握手了也没有接收数据

（4）修正效果

可以读入数据

（5）归纳总结（可选）

看波形真的很头秃。

2、错误 2

（1）错误现象

读 0x00008002 和 0x00008000 错误

（2）分析定位过程

0x00008000 是第一条请求，第一条请求未成功

（3）错误原因

valid 信号未初始化，

（4）修正效果

```
w_data      <= 32'd0;
awvalid_en  <= 1'b0;
wvalid_en   <= 1'b0;
```

增加了初始化。

（5）归纳总结（可选）

无。

3、错误 3

（1）错误现象

读 0x00008040 有误。

（2）分析定位过程

跟踪波形发现前一条的写指令没有执行，因此读到的是上一条指令的值。

（3）错误原因

只进行了读写独立处理，但是没有考虑读后写、写后读相关

（4）修正效果

增加了延迟阻塞信号，判断冲突，如果冲突进行阻塞。

（5）归纳总结（可选）

调试波形的时候不仅要考虑当前错误值，可能还要追溯前面几条找到根源。

四、实验总结（可选）

很长一段时间没有理解实验到底要我们做什么，可能因为换了环境，不是在原 cpu 基础上改。

一开始以为这个握手信号和上学期计组实现的差异不大，写一个简单的状态机判断读写就行，然后发现差得很大……听课的时候草草听过去了但是没实际概念，实际进行的时候就知道这个东西调起来很烦有很多坑……

现在对于接下来两周实验充满了畏惧。