

计算机组成原理

实验4 AXI-Lite总线接口设计



HITSZ 实验与创新实践教育中心
Education Center of Experiments and Innovations, HITSZ

实验目的

- 了解CPU与SoC的关系，熟悉SoC的基本架构
- 了解基于ready-valid握手协议的总线工作原理
- 熟悉AXI4-Lite总线协议，掌握其总线接口的实现方法



实验内容

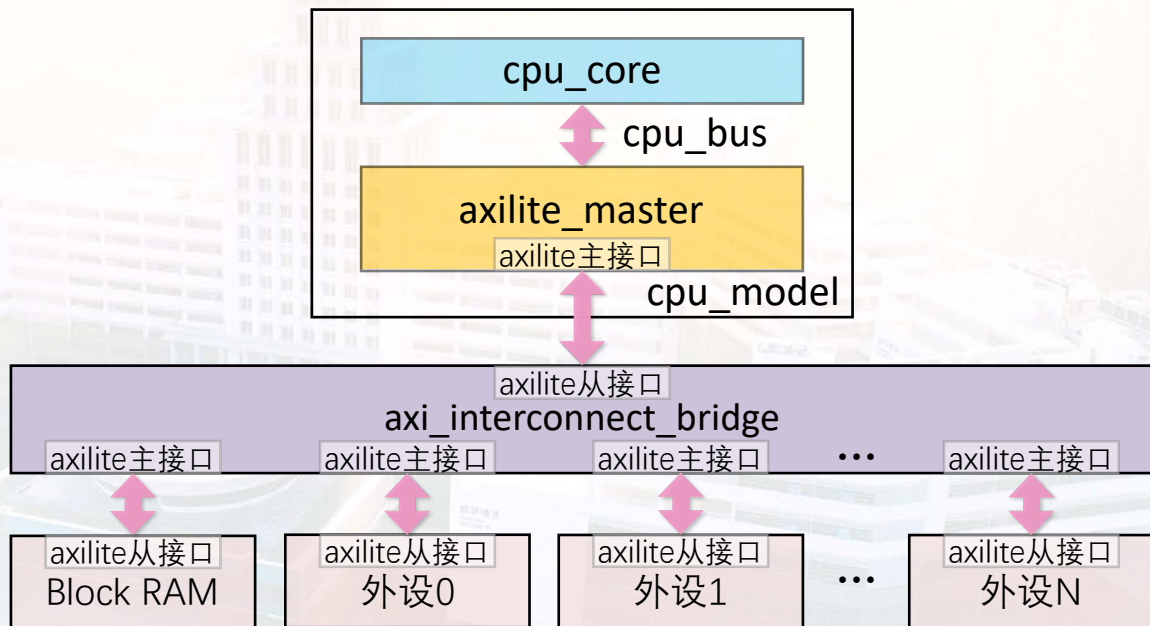
- 在SoC模板工程上，使用Verilog实现**axilite_master模块**，使得CPU能够通过AXI4-Lite协议读写存储器和I/O设备
 - 实现AW、W、AR、R四个通道
 - 为简化设计，axilite_master模块总是先处理完一个请求，再处理下一个请求，即不考虑连续数据访问、读写请求重叠的情况
- 运行功能仿真，通过所有测试用例，不需下板

实验原理

◆ SoC是集成在芯片上的相对完整的计算机系统

• 模板工程的SoC架构

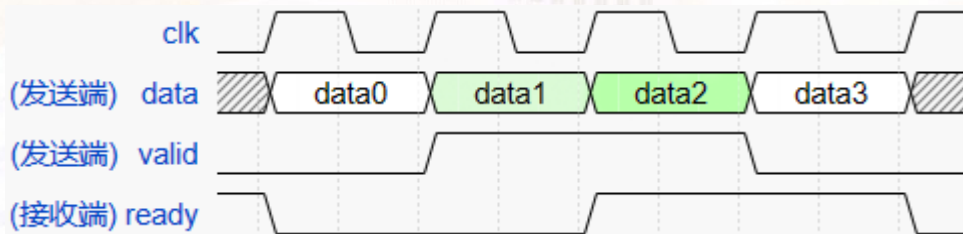
- cpu_core: 32位模型
- cpu_bus: 处理器核
对外的简易总线
- 核心通过专门的模块
实现AXI-Lite总线
- 总线桥互联多个设备



实验原理

◆ ready-valid握手协议：保证信息在发送端、接收端之间准确无误传输

- ready信号：接收端的就绪状态
- valid信号：发送端发送数据的有效状态

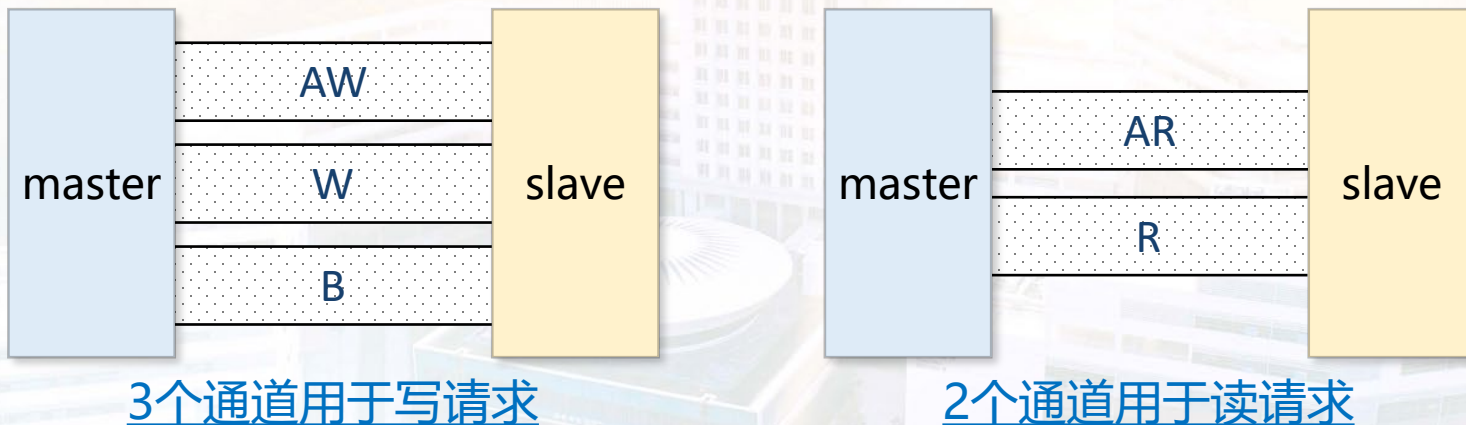


- 对发送端：当且仅当valid有效时，数据才有效 —— data1、data2是有效数据
 - 对接收端：当且仅当ready有效时，才可能接收数据 —— 只有data2被接收
- ◆ 当且仅当**valid、ready同时有效**时，数据才能在clk上升沿时成功传输

实验原理

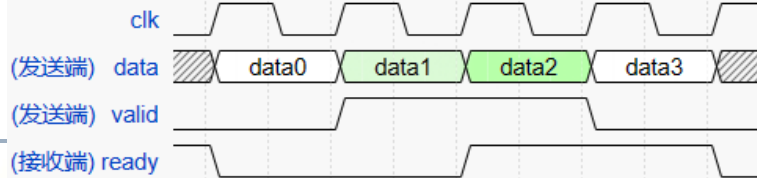
◆ AXI4-Lite总线协议：基于ready-valid握手协议的嵌入式总线

- 包含5个独立通道：写地址(**AW**)、写数据(**W**)、写响应(**B**)、读地址(**AR**)、读数据(**R**)



- 每个通道都基于ready-valid信号实现握手

实验原理



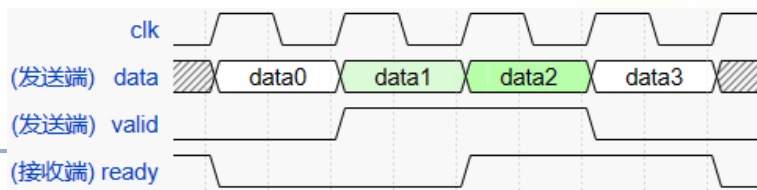
◆ AXI4-Lite写通道：AW通道、W通道、B通道

- 写通道信号：

| 通道 | 信号 | 位宽 | 方向 | 释义 |
|-------------|---------|----|-----|--------------|
| 写地址 (AW) | awaddr | 32 | 主→从 | 写地址 |
| | awvalid | 1 | 主→从 | 写地址的有效信号 |
| | awready | 1 | 从→主 | 从设备AW通道的就绪信号 |
| 写数据 (W) | wdata | 32 | 主→从 | 写数据 |
| | wstrb | 4 | 主→从 | 写使能，每位对应1字节 |
| | wvalid | 1 | 主→从 | 写数据的有效信号 |
| | wready | 1 | 从→主 | 从设备W通道的就绪信号 |
| 写响应 (B) | bready | 1 | 主→从 | 主设备B通道的就绪信号 |
| | bresp | 2 | 从→主 | 写操作结果（可忽略） |
| | bvalid | 1 | 从→主 | 写响应的有效信号 |

- 每个通道都遵循ready-valid握手协议

实验原理



◆ AXI4-Lite读通道：AR通道、R通道

- 读通道信号：

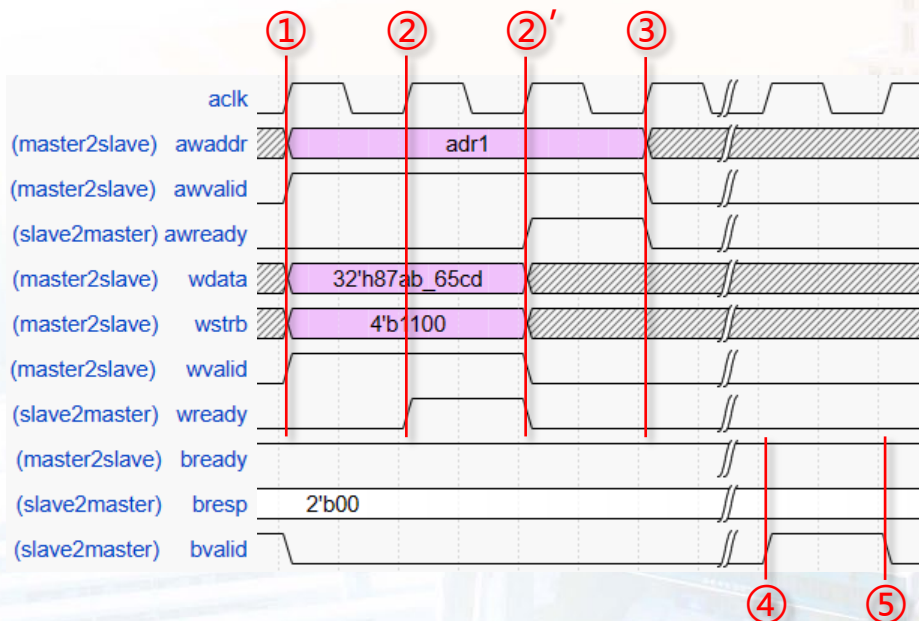
master
发送
slave
接收

slave
发送
master
接收

| 通道 | 信号 | 位宽 | 方向 | 释义 |
|-------------|---------|----|-----|--------------|
| 读地址 (AR) | araddr | 32 | 主→从 | 读地址 |
| | arvalid | 1 | 主→从 | 读地址的有效信号 |
| | arready | 1 | 从→主 | 从设备AR通道的就绪信号 |
| 读数据 (R) | rready | 1 | 主→从 | 主设备R通道的就绪信号 |
| | rdata | 32 | 从→主 | 读数据 |
| | rresp | 2 | 从→主 | 读操作结果（可忽略） |
| | rvalid | 1 | 从→主 | 读响应的有效信号 |

实验原理

◆ AXI4-Lite写时序:



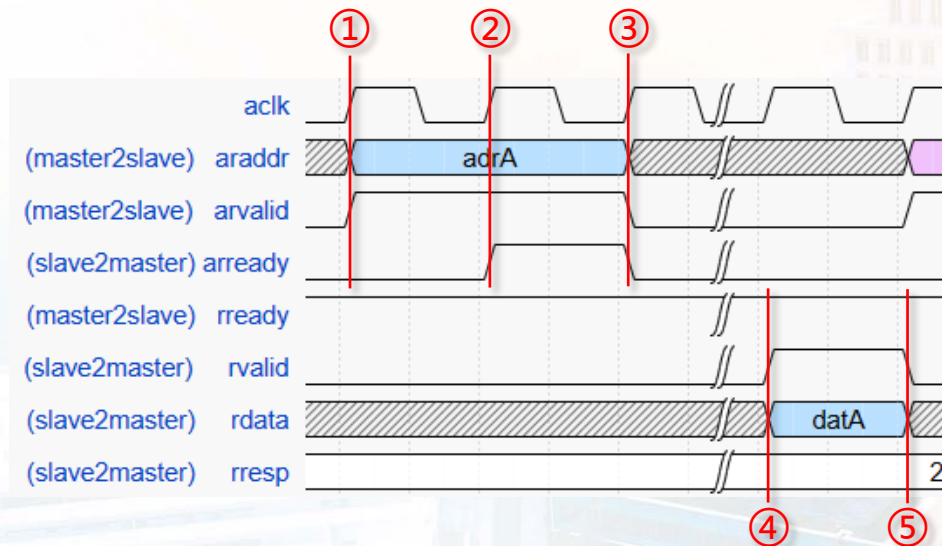
- ① master分别通过AW、W通道发写地址、写数据
- ② slave的W通道就绪(wready有效), 接收写数据
- ②' master撤掉写数据请求;
slave的AW通道就绪(awready有效), 接收写地址
- ③ master撤掉写地址请求; slave执行写操作
- ④ 写操作完成, slave给出写响应
- ⑤ slave撤掉写响应, 一个完整的写请求结束

注意: 各通道**相互独立**

- awready、wready不一定同时有效
- 只要awready有效, slave即可接收写地址请求
- 只要wready有效, slave即可接收写数据请求
- 只有写地址、写数据都被接收, 写请求才被成功接收, 才能开始写操作

实验原理

◆ AXI4-Lite读时序:



◆ cpu_bus总线：W通道、R通道

- 读写均支持字、半字和字节

| 通道 | 信号 | 位宽 | 方向 | 释义 |
|------------|--------|----|-----|----------------|
| 写通道 (W) | wrdy | 1 | 从→主 | 从设备W通道的就绪信号 |
| | wen | 4 | 主→从 | 写使能，支持写字、半字和字节 |
| | waddr | 32 | 主→从 | 写地址 |
| | wdata | 32 | 主→从 | 写数据 |
| 读通道 (R) | rrdy | 1 | 从→主 | 从设备R通道的就绪信号 |
| | ren | 4 | 主→从 | 读使能，支持读字、半字和字节 |
| | raddr | 32 | 主→从 | 读地址 |
| | rvalid | 1 | 从→主 | 读数据的有效信号 |
| | rdata | 32 | 从→主 | 读数据 |

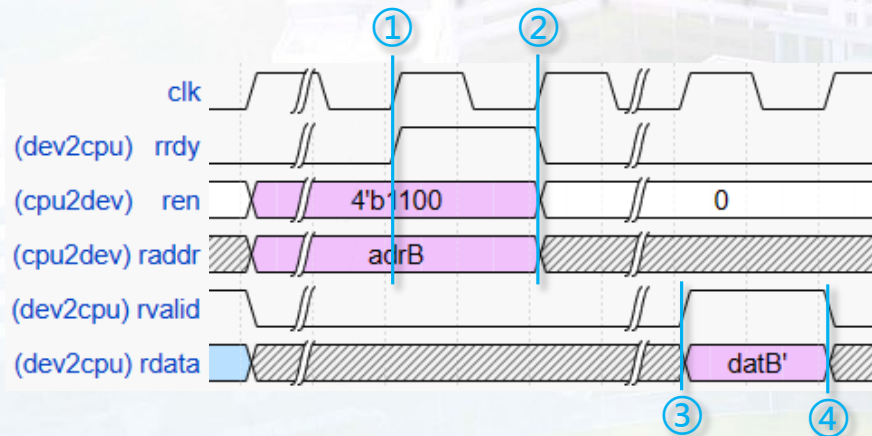
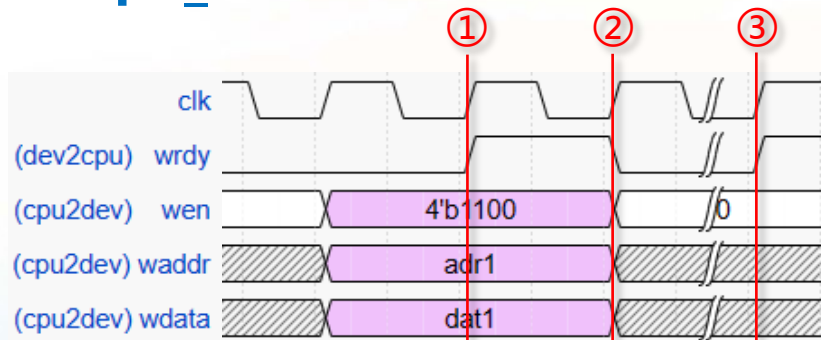
实验原理

cpu_core

cpu_bus

axilite_master

◆ cpu_bus写、读时序:



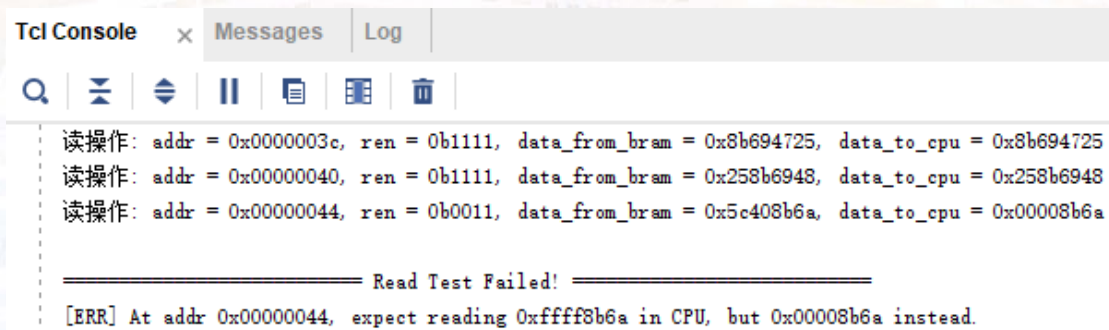
- ① AXI控制器就绪时，拉高wrdy以接收CPU的写请求
- ② 若wen不为0，AXI控制器拉低wrdy，开始处理写请求
- ③ 写请求处理完毕，AXI控制器重新把wrdy拉高

- ① AXI控制器就绪时，拉高rrdy以接收CPU的读请求
- ② 若ren不为0，AXI控制器拉低rrdy，开始处理读请求
- ③ AXI控制器拉高rvalid，并返回读数据 (按需符号扩展)
- ④ AXI控制器撤掉rvalid和读数据
- ⑤ 就绪后重新拉高rrdy



实验步骤

1. 熟悉AXI4-Lite总线协议，熟悉Vivado模板工程
2. 打开模板工程，完成axilite_master.v中的AW、W、AR、R四个通道
3. 运行功能仿真，根据Tcl控制台的调试信息完成调试



```
Tcl Console x Messages Log
[Icons: Search, Collapse, Expand, Run, Stop, Refresh, Copy, Paste]
读操作: addr = 0x0000003c, ren = 0b1111, data_from_bram = 0x8b694725, data_to_cpu = 0x8b694725
读操作: addr = 0x00000040, ren = 0b1111, data_from_bram = 0x258b6948, data_to_cpu = 0x258b6948
读操作: addr = 0x00000044, ren = 0b0011, data_from_bram = 0x5c408b6a, data_to_cpu = 0x00008b6a

===== Read Test Failed! =====
[ERR] At addr 0x00000044, expect reading 0xffff8b6a in CPU, but 0x00008b6a instead.
```

4. 总结及反思，撰写实验报告

验收及提交

- **课堂验收：**
 - 课上检查是否通过Testbench的所有测试用例：2分
- **提交内容**
 - axilite_master.v：1分
 - 实验报告（按模板完成）：4分
- 将上述文件打包成.zip，以“**学号_姓名.zip**”命名提交到作业系统
 - ◆ 注意：**如有雷同，双方均0分！**



HITSZ 实验与创新实践教育中心
Education Center of Experiments and Innovations, HITSZ