

第二届系统能力培养大赛 参赛指南

2018.5.13



比赛阶段任务解析

初阶任务指导

实验箱使用说明

单周期SRAM接口说明

中阶任务指导

辅助调试手段说明

高阶任务指导

比赛阶段划分

初阶

- 有望获奖
- 对无基础的：极具挑战性



中阶

- 可以冲刺前十
- 供有基础的向上冲刺



高阶

- 冲冠
- 能力有余的，各展所长



任务：基本5级流水CPU实现，仅CPU

- MIPS I基准指令集：57条指令，部分中断例外
- 无复杂总线接口，单周期SRAM接口，无仲裁
- 运行功能测试通过



任务：支持SoC搭建的CPU

- MIPS I基准指令集：57条指令，部分中断例外
- 支持复杂总线接口：类SRAM接口，或AXI接口，或其他
- 运行随机延迟功能测试通过
- 运行性能测试通过



任务：基于嵌入式CPU，向不同方向延伸

- 系统方向：
- 专研方向：
- 综合方向
- 异构方向：

比赛阶段划分

初阶

- 有望获奖
- 对无基础的：极具挑战性

任务：基本5级流水CPU实现，仅CPU

- MIPS I基准指令集：57条指令，部分中断例外
- 无复杂总线接口，单周期SRAM接口，无仲裁
- 运行功能测试通过

中阶

- 可以冲刺前十
- 供有基础的向上冲刺

任务：支持SoC搭建的CPU

- MIPS I基准指令集：57条指令，部分中断例外
- 支持复杂总线接口：类SRAM接口，或AXI接口，或其他
- 运行随机延迟功能测试通过
- 运行性能测试通过

高阶

- 冲冠
- 能力有余的，各展所长

任务：基于嵌入式CPU，向不同方向延伸

- 系统方向：
- 专研方向：
- 综合方向
- 异构方向：

预赛

决赛

比赛阶段任务解析

初阶任务指导

实验箱使用说明

单周期SRAM接口说明

中阶任务指导

辅助调试手段说明

高阶任务指导

初阶任务指导

初阶是基础，极具挑战性，勿好高骛远

- **任务：基本5级流水CPU实现**
 - **MIPS I基准指令集：57条指令，部分中断例外，**
 - **五级流水，无复杂总线接口，无仲裁**
- **考验**
 - **课程知识点**
 - **Verilog编写**
 - **硬件调试能力**
 - **FPGA开发能力**
 - **学习能力**
- **特点**
 - **对无训练基础的，最后真正小组合作完成的：**
极具挑战性，极具成就感，收获极大

难者不会，会者不难；写出来的，才是自己的

1. 环境初步准备：只要一个Vivado

- Vivado2018.1
 - 安装：大赛发布包：doc/*vivado安装说明.pdf*
 - 使用：大赛发布包：doc/*vivado使用说明.pdf*
- 目前，无实验箱
 - 设计+仿真

设计(Verilog)->仿真

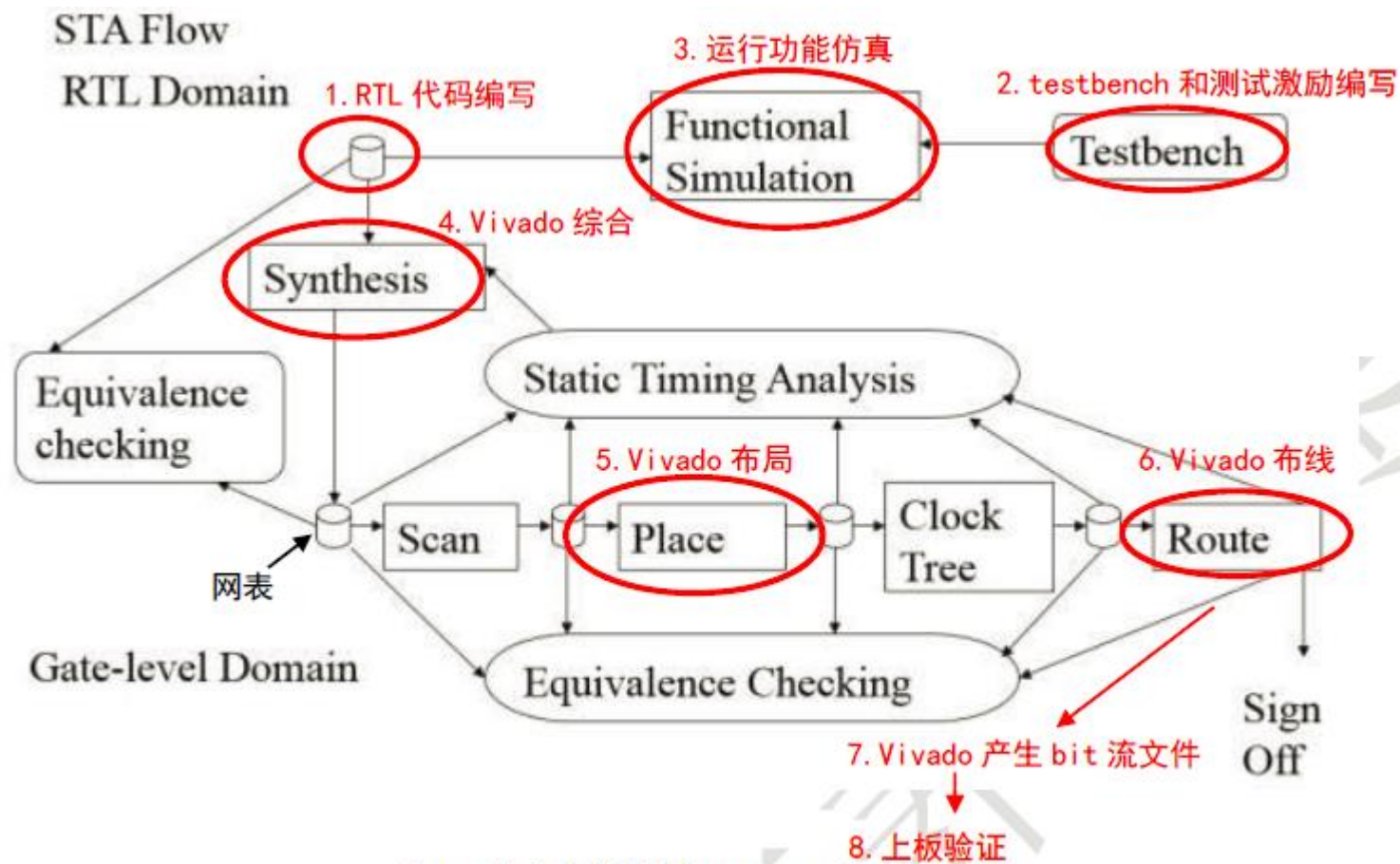


图 1-1 数字电路设计与 FPGA 开发对照图

2. 设计：知识准备——Verilog

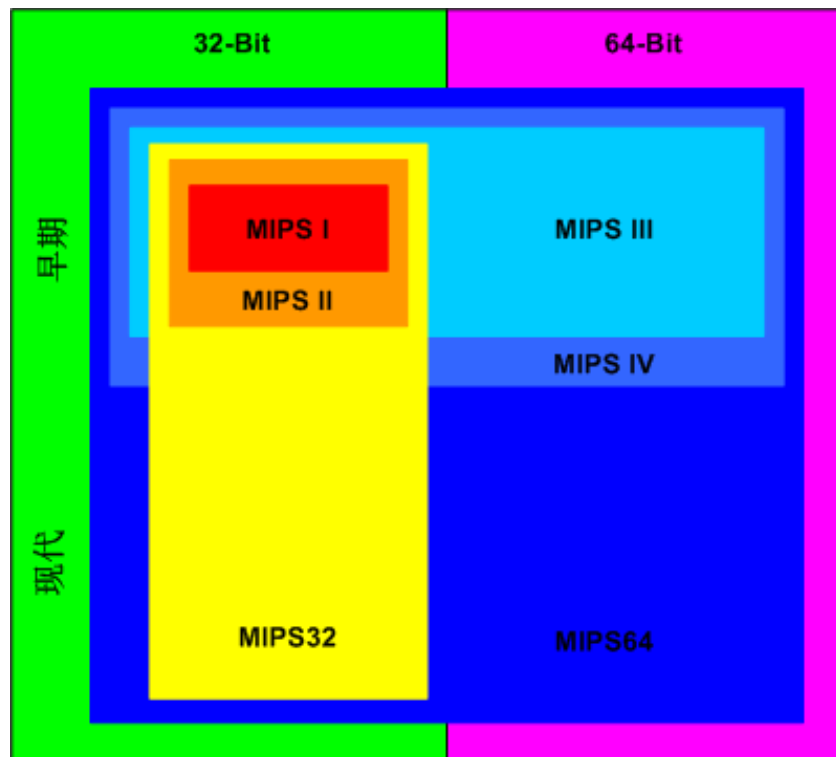
➤ Verilog: **写Verilog是在画电路**

- 可综合性，不可综合性
- 综合性语法：assign，always@(posedge clock)，module，状态机

提问	不可综合	可综合
电路有初始值概念吗？	reg [3:0] cnt=0;	...if (reset) cnt <= 4'd0;...
电路有等待5ns的概念吗？	# 5;	计数时钟拍数。
电路各模块是串行跑的吗？		

2. 设计：知识准备——CPU架构

- MIPS I指令集
 - 发布包：doc/ **系统能力设计大赛MIPS基准指令集手册.pdf**
 - 发布包：doc/ **MIPS32手册.rar**
- **57条指令：MIPS I（无非对齐）+ ERET**
- **4个CP0寄存器**
- **6种例外（含中断）**
- **采用直接地址映射**
- **不区分核心态和用户态**



3. 仿真：自己进行简单仿真

- 自己设置输入激励：灌指令
- 自己设置输出比较：可以testbench比较，也可以灌入的指令程序比较

➤ 自己写testbench：

实例化myCPU，设置输入激励的灌入，设置输出比较（如果需要）

建议：自设计+仿真的循环迭代

3. 仿真：运行官方提供的功能测试

- 原理：MyCPU上运行软件程序
- 功能测试，发布包：**fun_test/**
 - **func_test/soft**：软件程序，MIPS汇编程序
 - **func_test/soc/soc_lite_fun_test**：用于功能测试的SoC，无延时版本

➤ 编译 功能测试程序

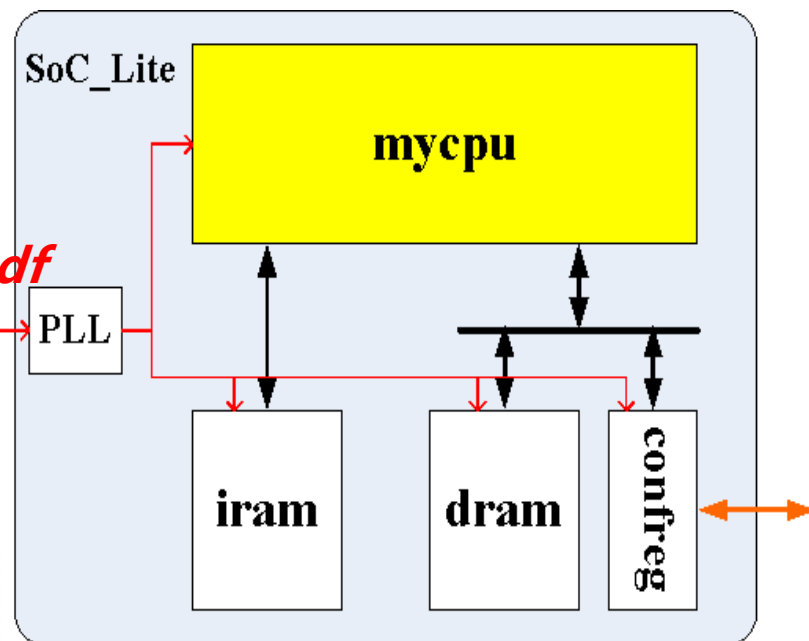
环境：发布包 doc/**交叉编译工具链安装.pdf**

Makefile：进入soft/src/，进行make

➤ 设置指令/数据 ram的数据

➤ 运行仿真

再次建议：自设计+仿真的循环迭代



4. 上板：需使用实验箱

- 仿真通过后，再上板（综合实现）
- 注意使用Verilog的可综合语句。
- 上板：对设计进行 **综合、布局布线、生成bit流文件**
下载bit流文件，实验箱（FPGA板）上观察现象。

初阶任务指导-总结

1. 环境初步准备：只要一个Vivado

2. 设计：知识准备——CPU架构

3. 仿真：自己进行简单仿真

3. 仿真：运行官方提供的功能测试

新增交叉编译环境！涉及SoC搭建！

4. 上板：需使用实验箱

涉及实验箱使用！

初阶任务指导-总结

- 两个环境
 - Vivado
 - 交叉编译环境：需要运行功能测试时再使用
- 设计
- 仿真
 - SoC_lite的理解：单周期RAM，涉及实验箱使用
- 上板（实际运行）
 - 涉及实验箱使用

比赛阶段任务解析

初阶任务指导

实验箱使用说明

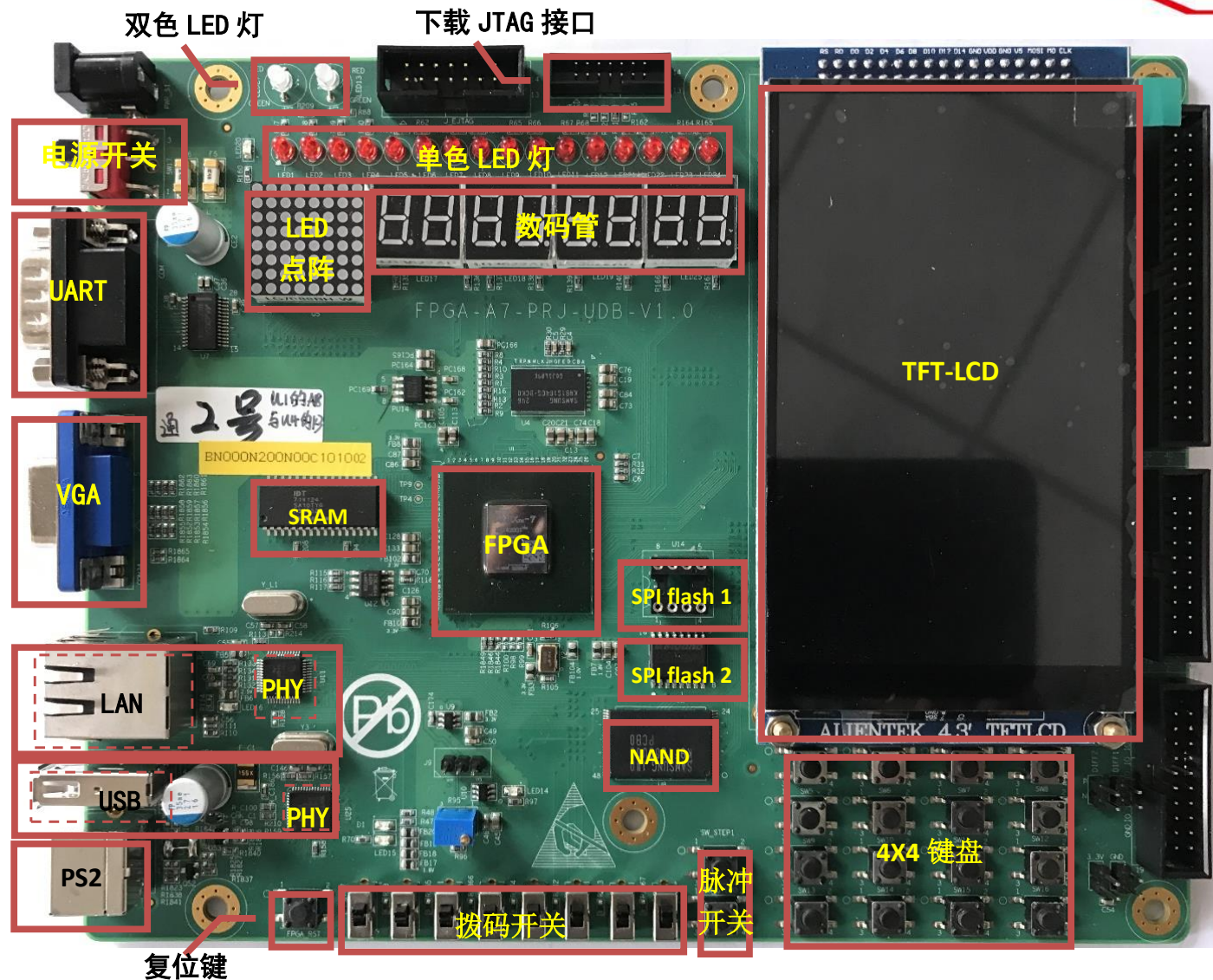
单周期SRAM接口说明

中阶任务指导

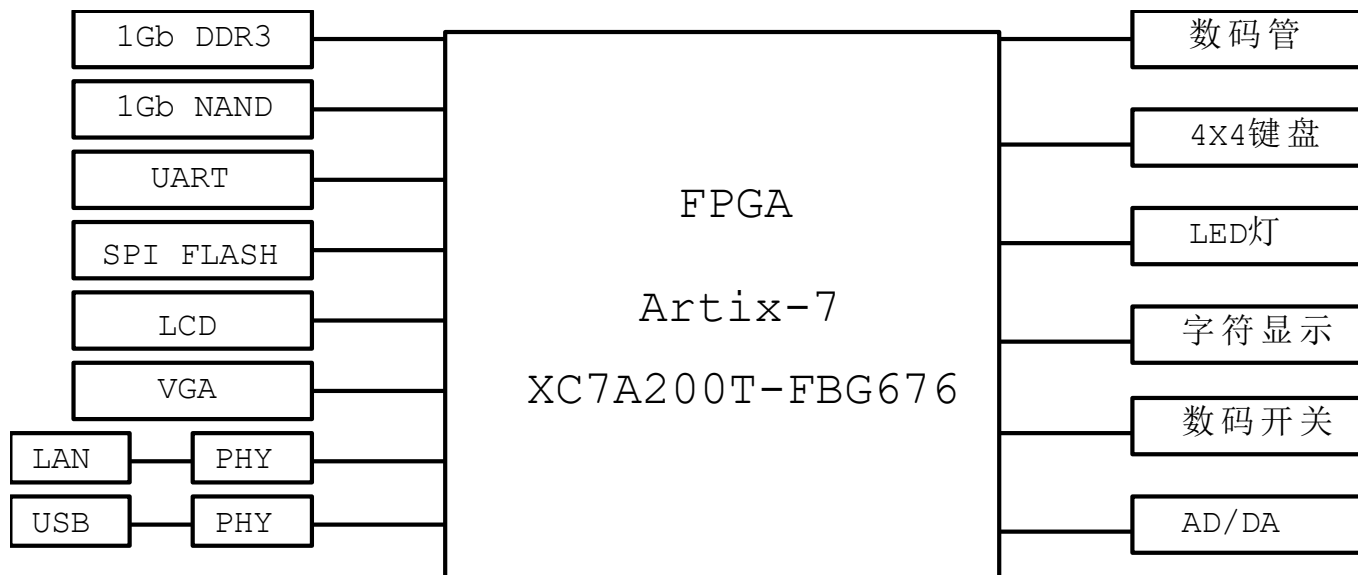
辅助调试手段说明

高阶任务指导

实验箱使用说明——实物图



实验箱使用说明——示意图



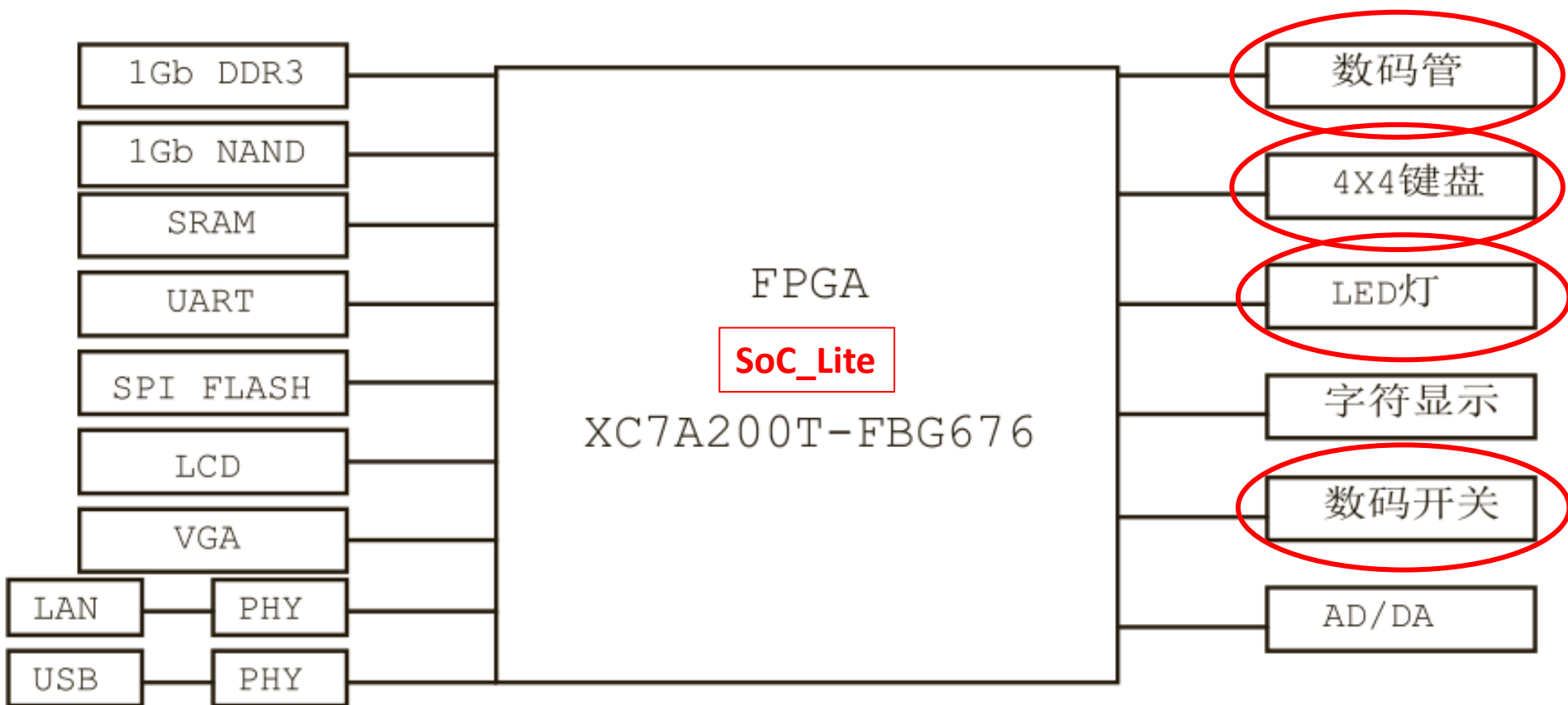
➤ 相关介绍资料

- doc/[龙芯体系结构教学实验箱 \(Artix-7 \) 介绍.pdf](#)
- doc/[龙芯体系结构实验箱 \(Artix-7 \) -原理图与引脚列表.pdf](#)

➤ FPGA开发：

- doc/[vivado使用说明.pdf](#)
- 仿真，约束文件，综合等

实验箱使用说明——SoC_Lite



实验箱使用说明——实验箱检测

- 拿到实验箱后，可以对实验进行检测
- GPIO类外设测试
 - 单色LED灯、双色LED灯、数码管、点阵、LCD触摸屏、
 - 复位键、拨码开关、按钮开关、矩阵键盘
 - 发布包 *fpga_test/FPGA_gpio_test*
- 复杂外设测试
 - 网口、DDR3颗粒、串口、SPI flash。
 - 发布包 *fpga_test/FPGA_soc_test*

实验箱使用说明——注意事项

➤ 板上电源

- 实验板为5V电源供电，请使用配套的电源进线供电。
- 禁止使用其他电源进行供电。

➤ 电源开关

- 为前后方向推动的开关，禁止上下拉扯

➤ 左侧电源线、串口、VGA、网口、USB等接口

- 为左后方向的插拔，禁止上下拉扯

➤ 右侧多排针接口

- 为FPGA外扩接口，禁止连接电源或短路

保护实验设备，人人有责

比赛阶段任务解析

初阶任务指导

实验箱使用说明

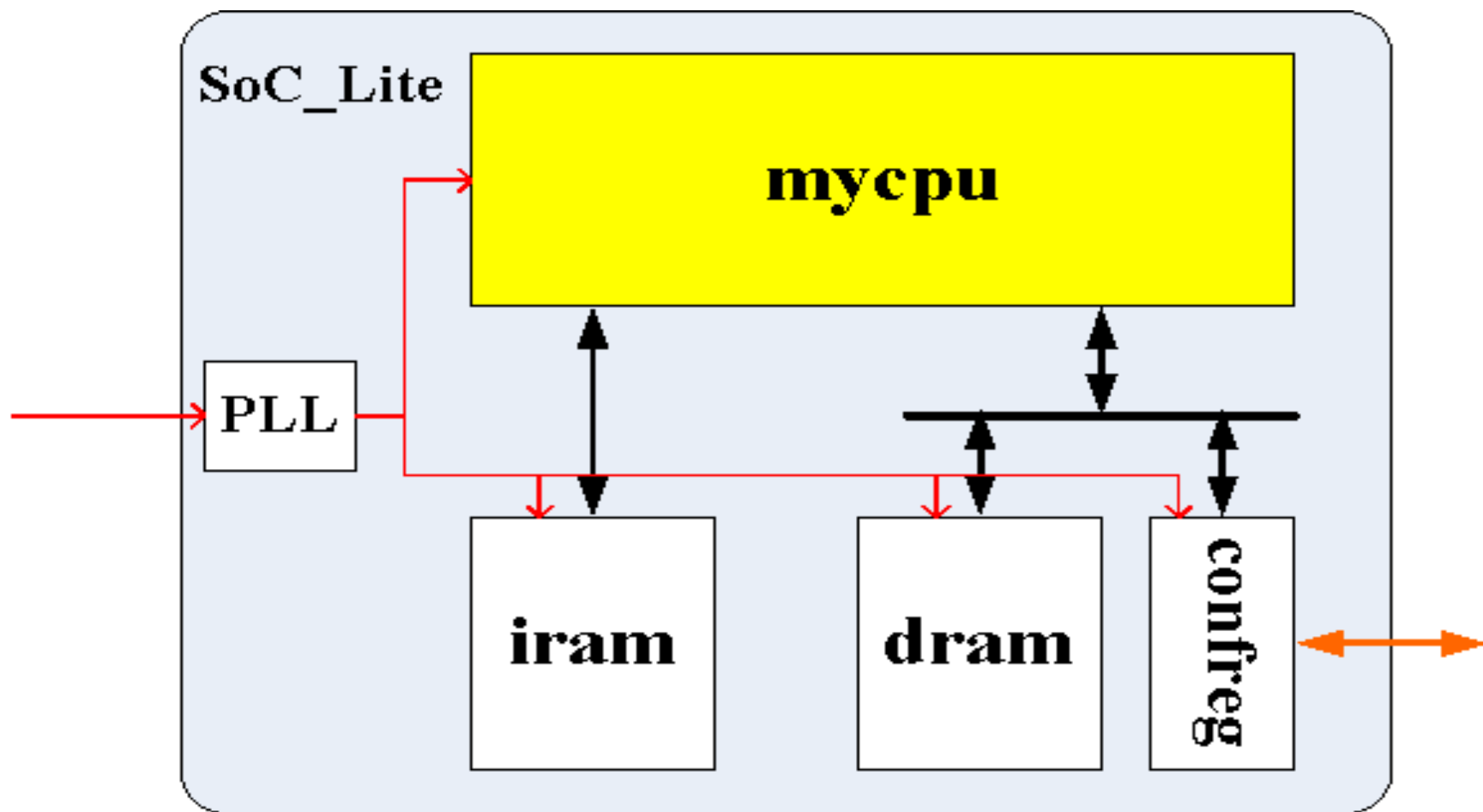
单周期SRAM接口说明

中阶任务指导

辅助调试手段说明

高阶任务指导

单周期SRAM接口说明



单周期SRAM接口说明

➤ 简单

➤ 信号少、无握手信号，一次传输一个数据

➤ 举例：单端口、32位地址、32位数据、字节写使能

名称	宽度	方向	描述
clk	1	—>mster/slave	时钟
en	1	—>slave	使能信号，片选作用
addr	32	—slave	地址，读写均使用同一地址线
wen	4	—>slave	字节写使能，指示写字节的位置
din	32	—>slave	写数据
dout	32	slave—>	读数据

单周期SRAM接口说明

- 写字节使能，支持任意的写字节组合
 - 写1字节：4'b0001, 4'b0010, 4'b0100, 4'b1000
 - 写2字节：4'b0011, 4'b0101, 4'b1001, 4'b0110, 4'b1010, 4'b1100
 - 写3字节：4'b0111, 4'b1011, 4'b1101, 4'b1110
 - 写4字节：4'b1111
- 地址是一个**数据宽度 (4bytes) 寻址 !!!**
 - 数据宽度32
 - 数据深度 2^{32}
 - 容量：32 x 2^{32} bits = 4 x 4GBytes
 - 举例：32 x 4的sram，地址需要2bit

但共指示了16字节

比如0~3字节，均使用地址00表示

计算机存储中，字节寻址

指示16字节，需要4 bit的地址，

0~3字节，对应地址0~3

地址00

32 bits(0~3字节)

地址01

32 bits(4~7字节)

地址10

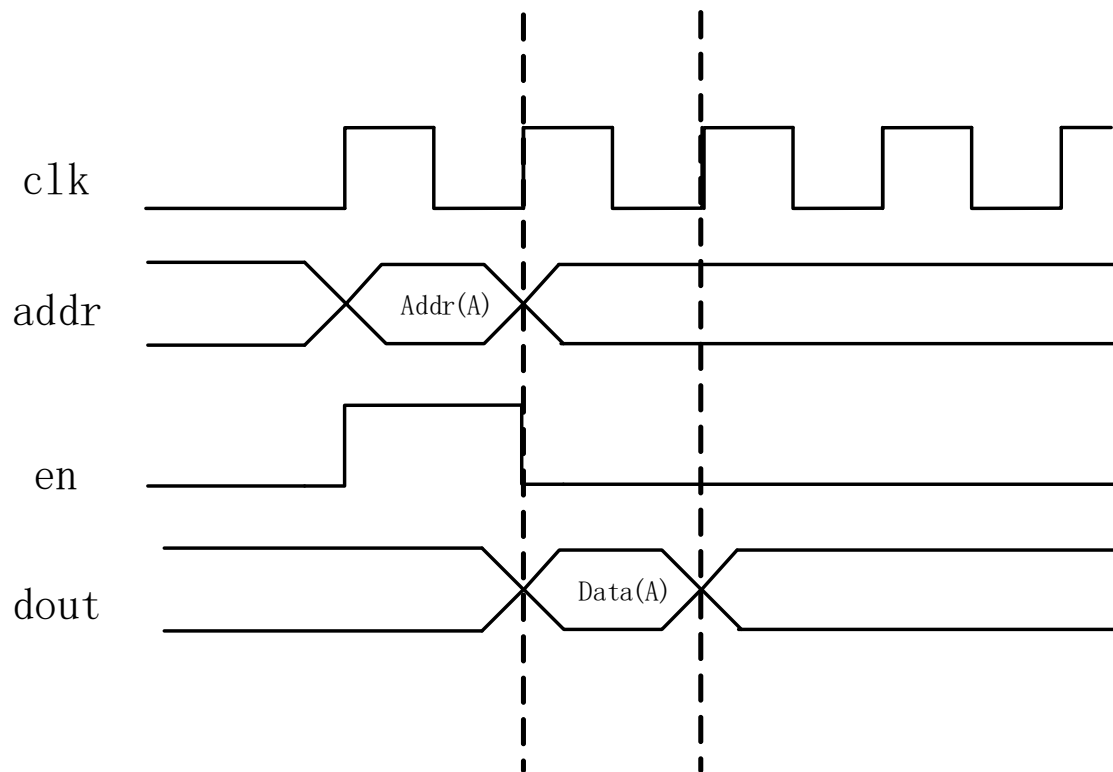
32 bits(8~11字节)

地址11

32 bits(12~15字节)

单周期SRAM接口说明

- 读数据 时序图
 - 当拍传地址，下拍回数据
 - 单周期返回，无延迟访存模型
 - CPU中的寄存器堆（Regfiles）与之不同。



```
always @(posedge clock)
begin
    if(en)
        begin
            rdata <= ram[addr];
        end
    end
end
```

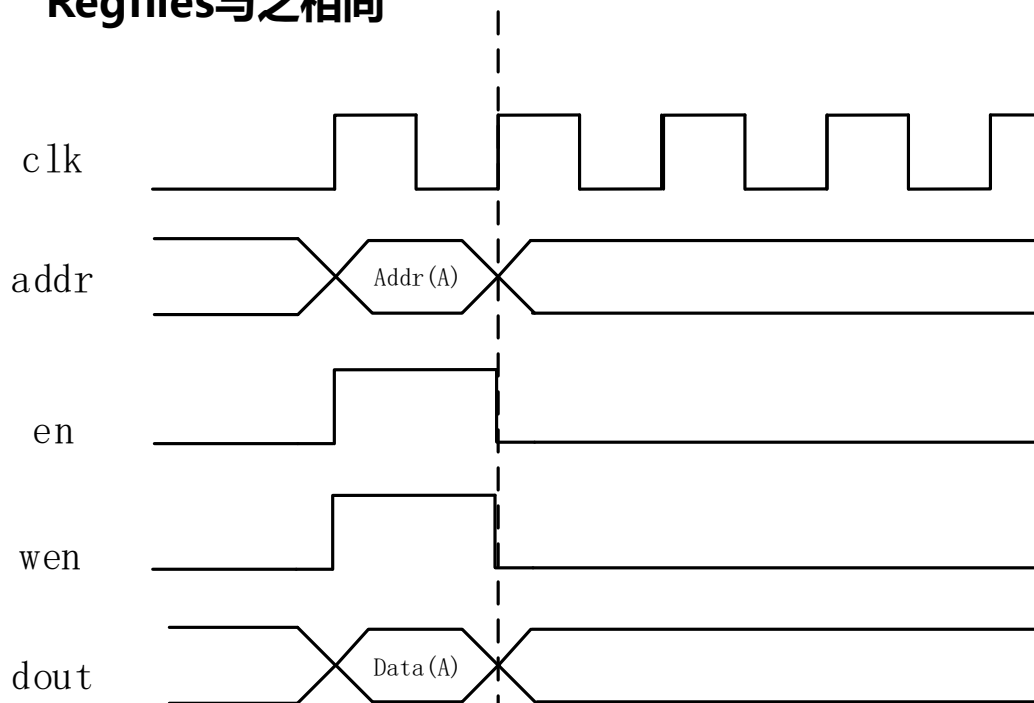
```
assign rdata = regfile[addr];
```

...

单周期SRAM接口说明

➤ 写数据 时序图

- 地址和数据 同拍给
- Regfiles与之相同

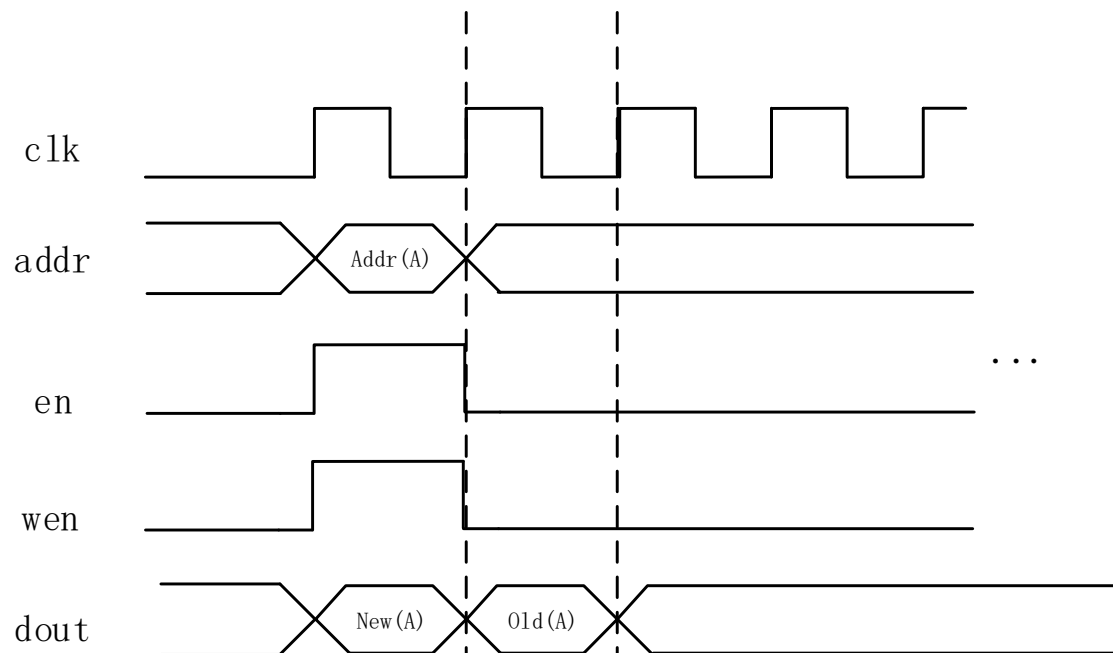


```
always @(posedge clock)
begin
    if(wen)
    begin
        ram[addr] <= wdata;
    end
end
```

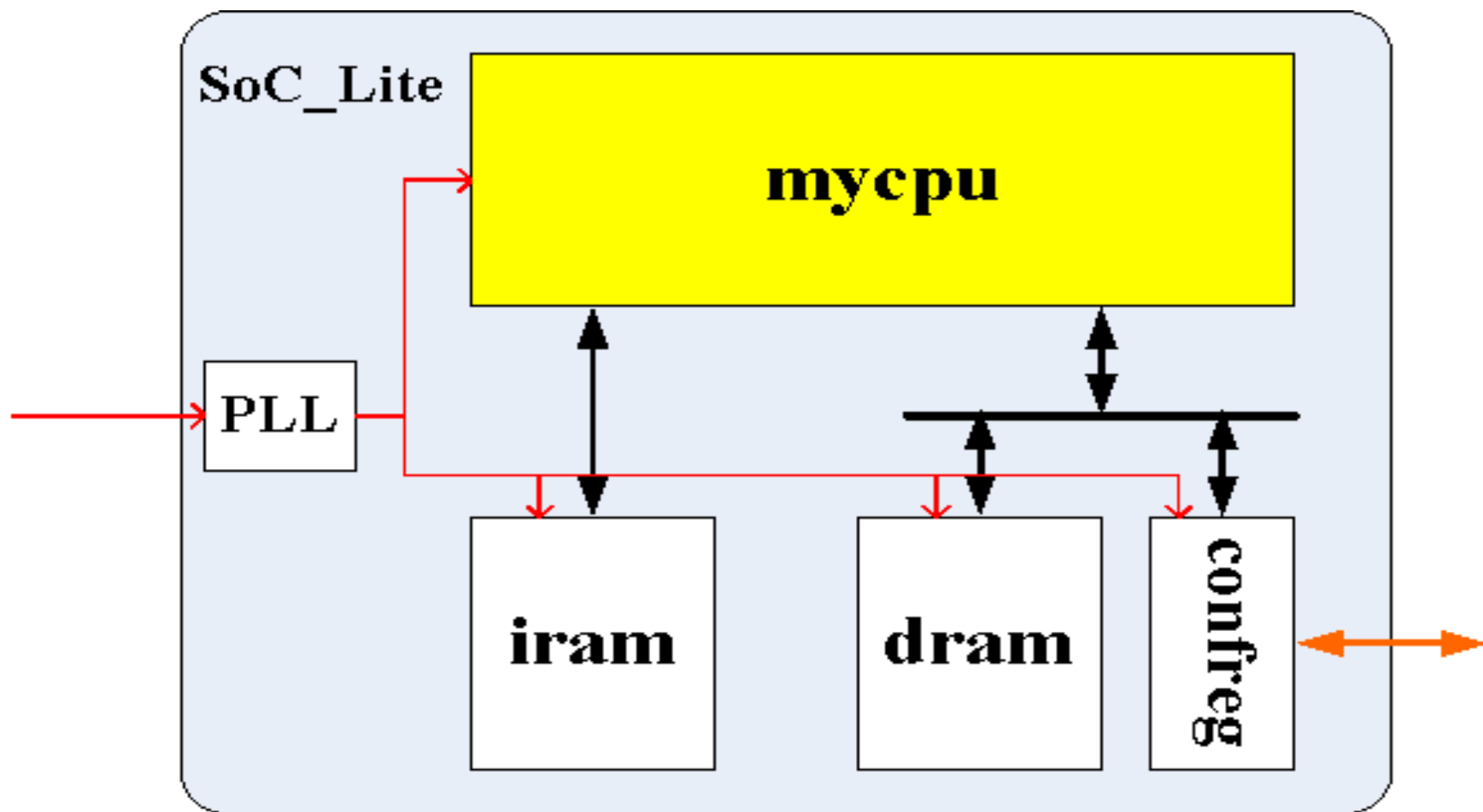
```
always @(posedge clock)
begin
    if(wen)
    begin
        regfile[addr] <= wdata;
    end
end
```

单周期SRAM接口说明

➤ 同时读写：



单周期SRAM接口说明



比赛阶段任务解析

初阶任务指导

实验箱使用说明

单周期SRAM接口说明

中阶任务指导

辅助调试手段说明

高阶任务指导

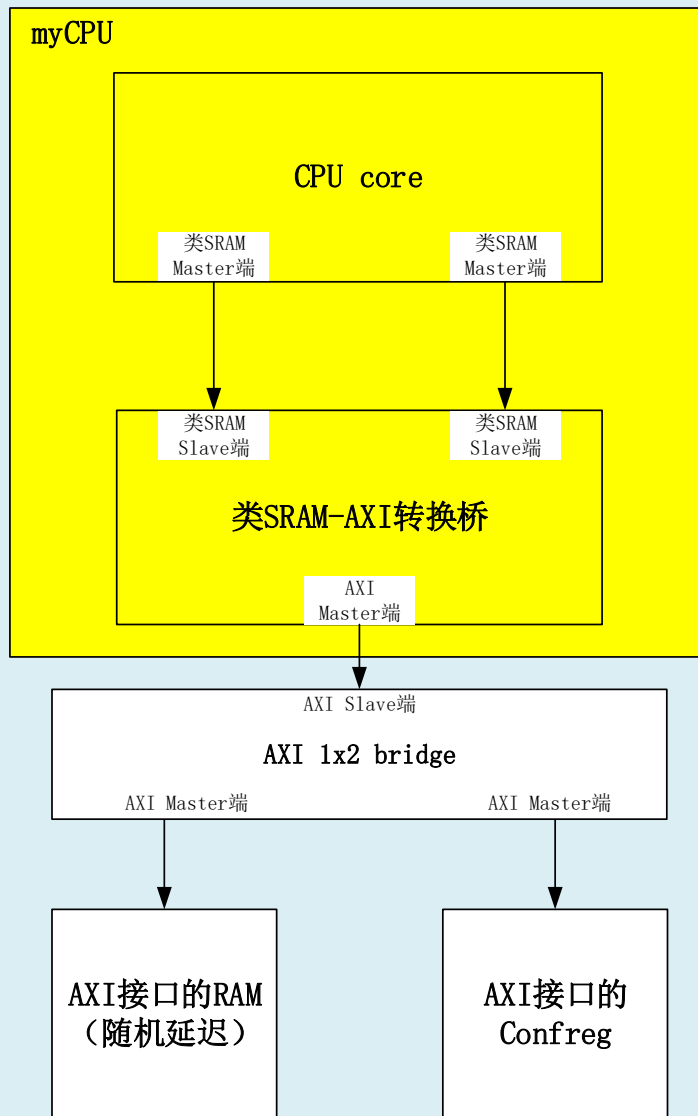
中阶任务指导

增加带握手的总线接口

- 增加复杂总线接口
 - 方案一：类SRAM接口
 - 方案二：AXI接口
 - 其他方案：如AHB-Lite
- 取指/数据 仲裁
- 随机功能测试
- 性能测试

完成初阶，再考虑中阶

SoC_AXI_Lite



中阶任务指导——总线握手的必要性

- **读写单周期返回的不足：**
 - 限制CPU频率
- **传输握手：**
 - 可协调不同速度的设备间交换数据
- **Master和slave区分：**
 - Master：发起请求
 - Slave：响应请求，返回数据
 - SoC_Lite，CPU和inst ram，哪个是master，哪个是slave？

中阶任务指导——总线握手的必要性

➤ 握手分类：

- 约定握手（不需要旗帜）：
- 单向握手（一方摇旗帜）：
- 双向握手（双方摇旗帜）：

握手一旦成功，只会持续一拍！多个周期看到握手成功，那是针对不同传输的握手！

举个例子：

目标：两个山头被包围，同时约定突围

方式：

- 约定握手：天黑即突围
- 单向握手：山头A对山头B说：我随时可以，你摇旗帜，我们就突围。
- 双向握手：山头A和山头B同时摇旗帜，即突围

中阶任务指导——总线握手的必要性

➤ 所有传输都是有握手的：

➤ 单周期返回的SRAM接口：

地址传输是单向握手，slave随时接受地址请求，片选en置上，就认为握手成功
数据传输时约定握手，约定地址传输下拍数据就返回了。

➤ 握手分类：

➤ 约定握手：

➤ 单向握手：类SRAM数据传输，AHB协议数据传输(Hready)

➤ 双向握手：AXI协议是双向握手，valid & ready

从上到下，握手越来越复杂，所实现的功能却越来越完善。

双向握手可实现约定握手、单向握手的功能。但反之却不然。

中阶任务指导——类SRAM接口

- 为SRAM接口增加地址传输握手信号addr_ok和数据传输握手信号data_ok
- 地址传输：双向握手，req & addr_ok同时有效
- 数据传输：单向握手，data_ok有效，master随时可以接受
- 握手成功只持续一拍。

信号	位宽	方向	功能
clk	1	input	时钟
req	1	master→slave	请求信号，为1时有读写请求，为0时无读写请求
wr	1	master→slave	该次请求是写
size	[1:0]	master→slave	该次请求传输的字节数，0: 1byte； 1: 2bytes； 2: 4bytes。
addr	[31:0]	master→slave	该次请求的地址
wdata	[31:0]	master→slave	该次请求的写数据
addr_ok	1	slave→master	该次请求的地址传输OK，读：地址被接收；写：地址和数据被接收
data_ok	1	slave→master	该次请求的数据传输OK，读：数据返回；写：数据写入完成。
rdata	[31:0]	slave→master	该次请求返回的读数据。

➤ 类SRAM接口与SRAM接口的不同

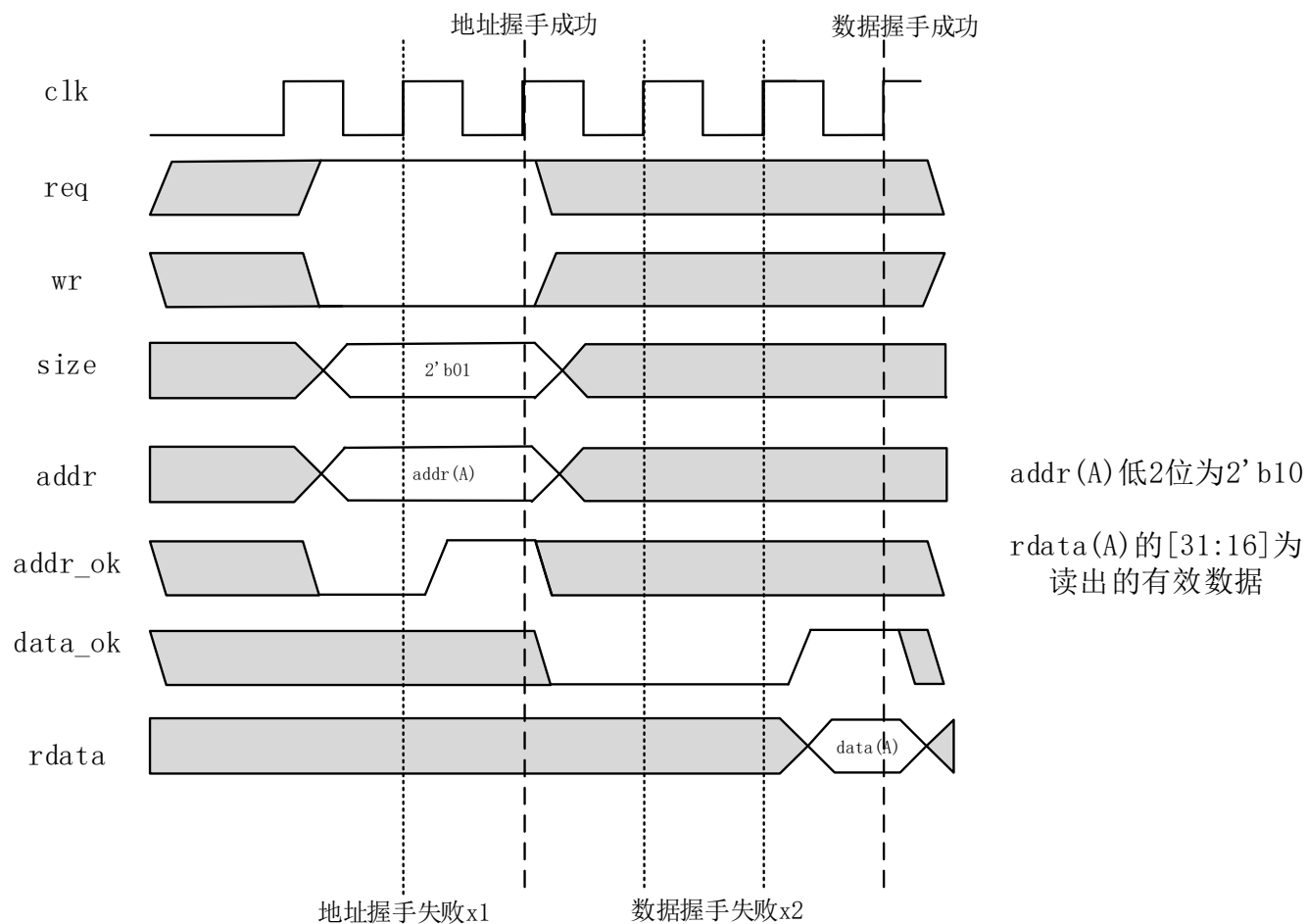
类SRAM地址为字节寻址

1. addr[1:0]=2'b00时，可能的组合：
 size=2'b00, size=2'b01, size=4'b10,
1. addr[1:0]=2'b01时，可能的组合：
 size=2'b00
1. addr[1:0]=2'b10时，可能的组合：
 size=2'b00, size=2'b01
1. addr[1:0]=2'b11时，可能的组合：
 size=2'b00

	data[31:24]	data[23:16]	data[15:8]	data[7:0]
size=2'b00,addr=2'b00	-	-	-	valid
size=2'b00,addr=2'b01	-	-	valid	-
size=2'b00,addr=2'b10	-	valid	-	-
size=2'b00,addr=2'b11	valid	-	-	-
size=2'b01,addr=2'b00	-	-	valid	valid
size=2'b01,addr=2'b10	valid	valid	-	-
size=2'b10,addr=2'b00	valid	valid	valid	valid

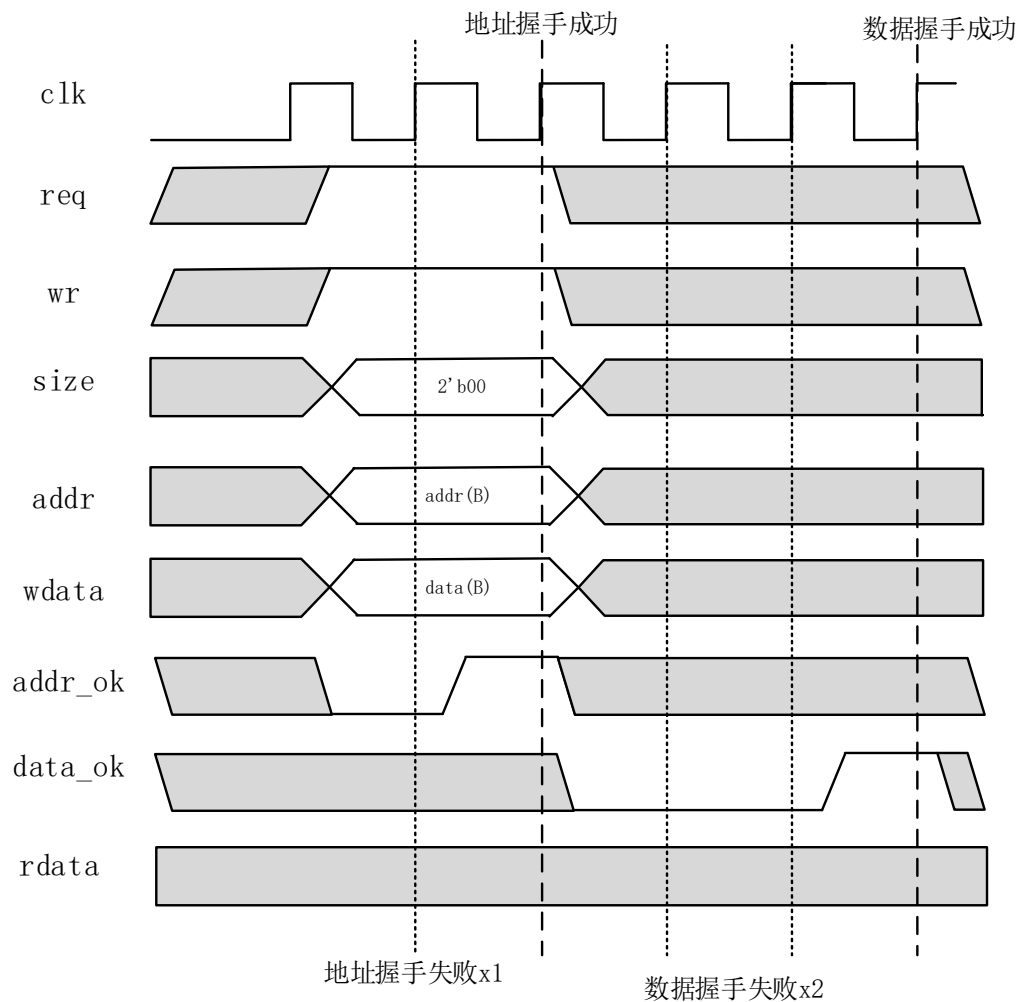
中阶任务指导——类SRAM接口

➤ 一次读时序图



中阶任务指导——类SRAM接口

➤ 一次写时序图



$addr(B)$ 低2位为 $2'b01$

$wdata(B)$ 的 $[15:8]$ 为待写的有效数据

中阶任务指导——类SRAM接口

连续写读时序图

slave返回的data_ok必须顺序返回的。

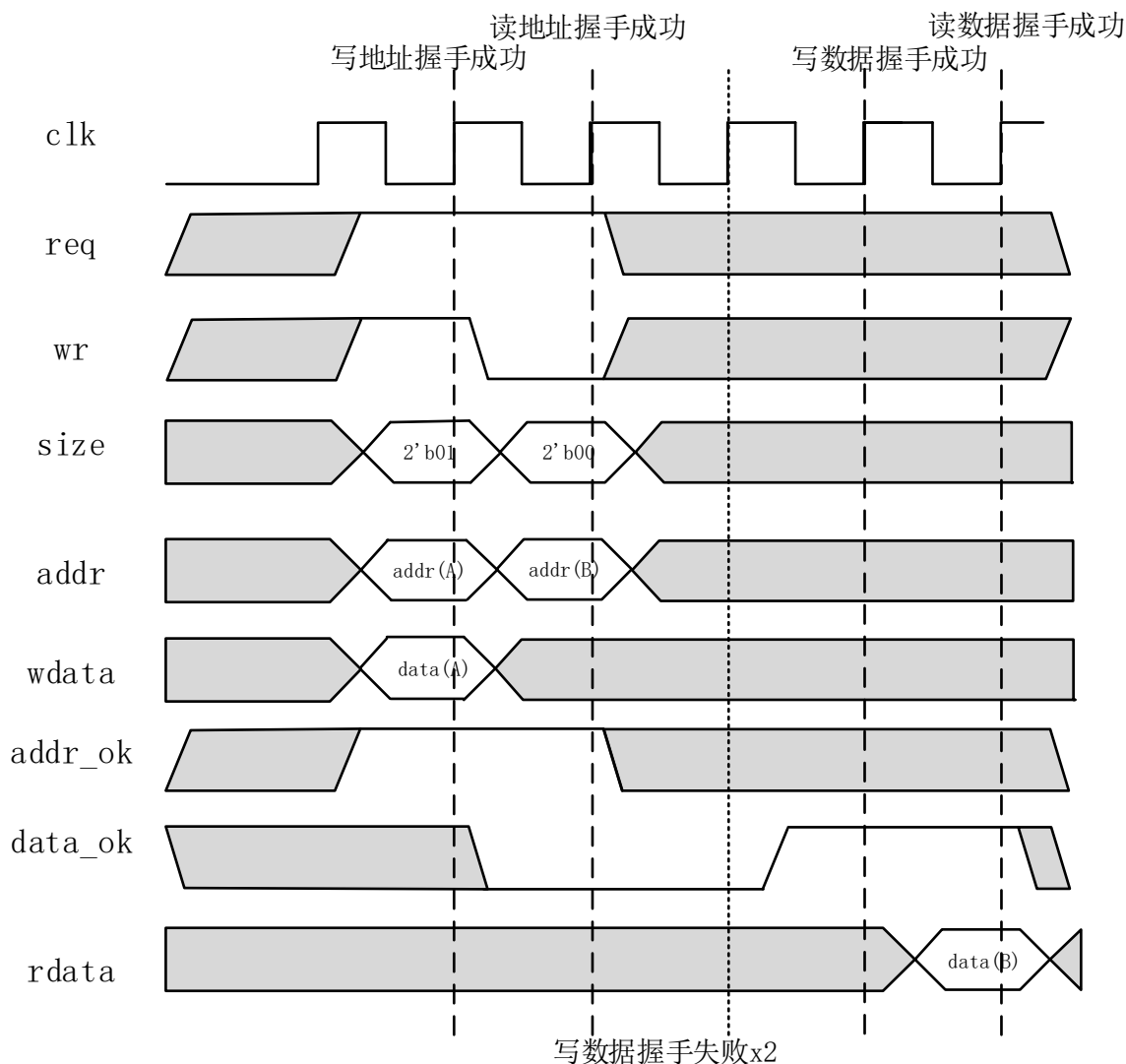
可能出现多次地址握手后，才会出线数据握手。

addr_ok->addr_ok->addr_ok->addr_ok->...->data_ok。

此时数据握手是对应第一次的传输。

master端避免这一情况的出现可以通过拉低req信号。

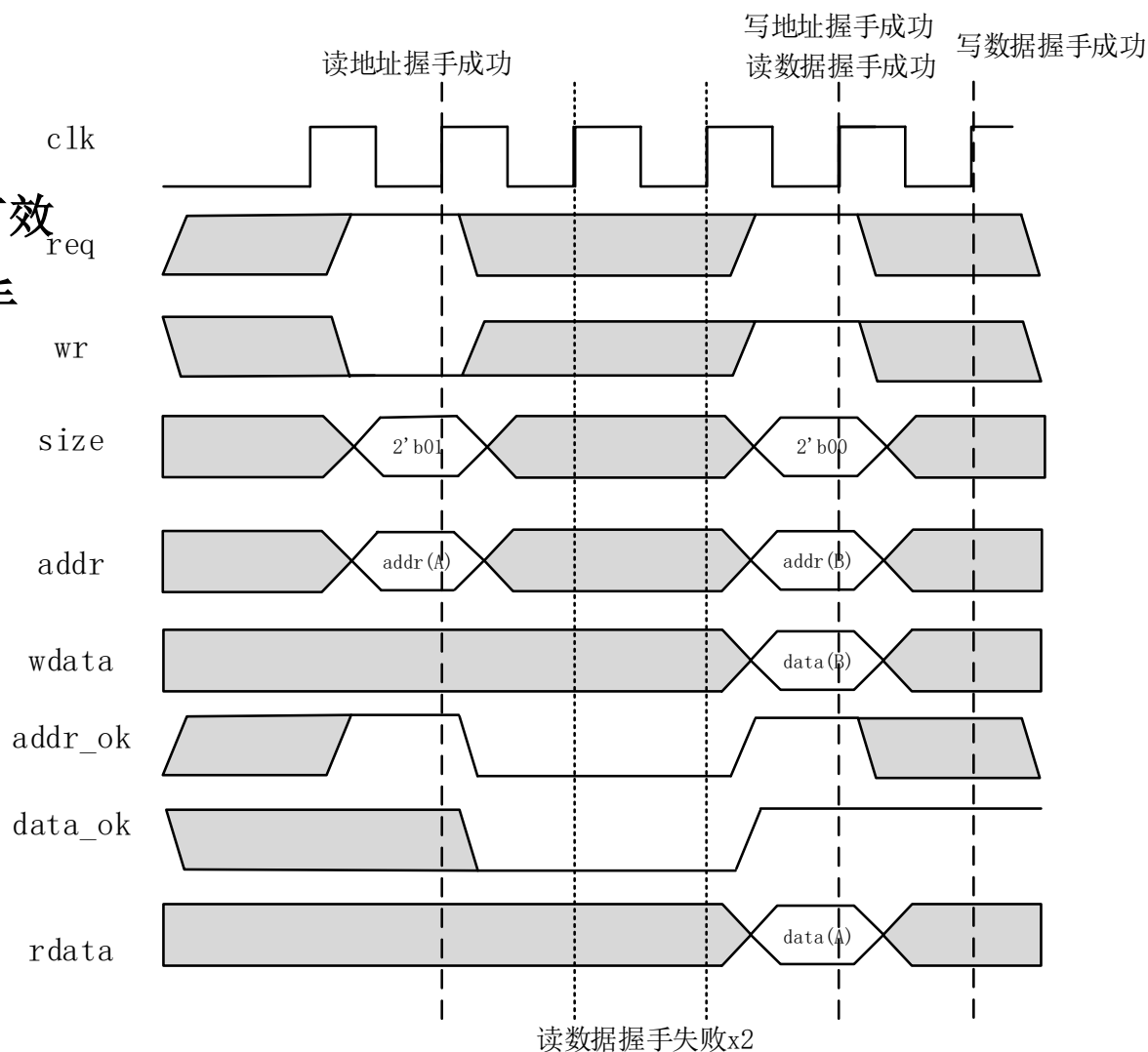
slave避免这一情况的出现可以通过拉低addr_ok



中阶任务指导——类SRAM接口

➤ 连续读写时序图

➤ 当addr_ok和data_ok同时有效时，是针对不同请求的握手



中阶任务指导——总线接口

推荐直接实现AXI接口

发布包：doc/*AMBA总线协议.zip*

- 只使用AXI协议的握手特性
- AXI访问读写依次访问，无需关心 交错访问特性
- 无需关心 burst传输。
- 一旦置起请求，在握手成功前，不能撤下

中阶任务指导—— AXI接口

- **AMBA总线中最高性能的总线接口**
- **AXI-3和AXI-4协议差别不大**
- **AXI主要特性**
 - **读操作，分为：地址传输和数据传输**
 - **写操作，分为：地址传输、数据传输和响应传输**
 - **读写操作完全分开**
 - **因而AXI分为5个通道**
 - 读操作占用两个，写操作占用另外三个通道**
 - **每个通道有自己的握手信号**

中阶任务指导——AXI接口

➤ 两个读通道

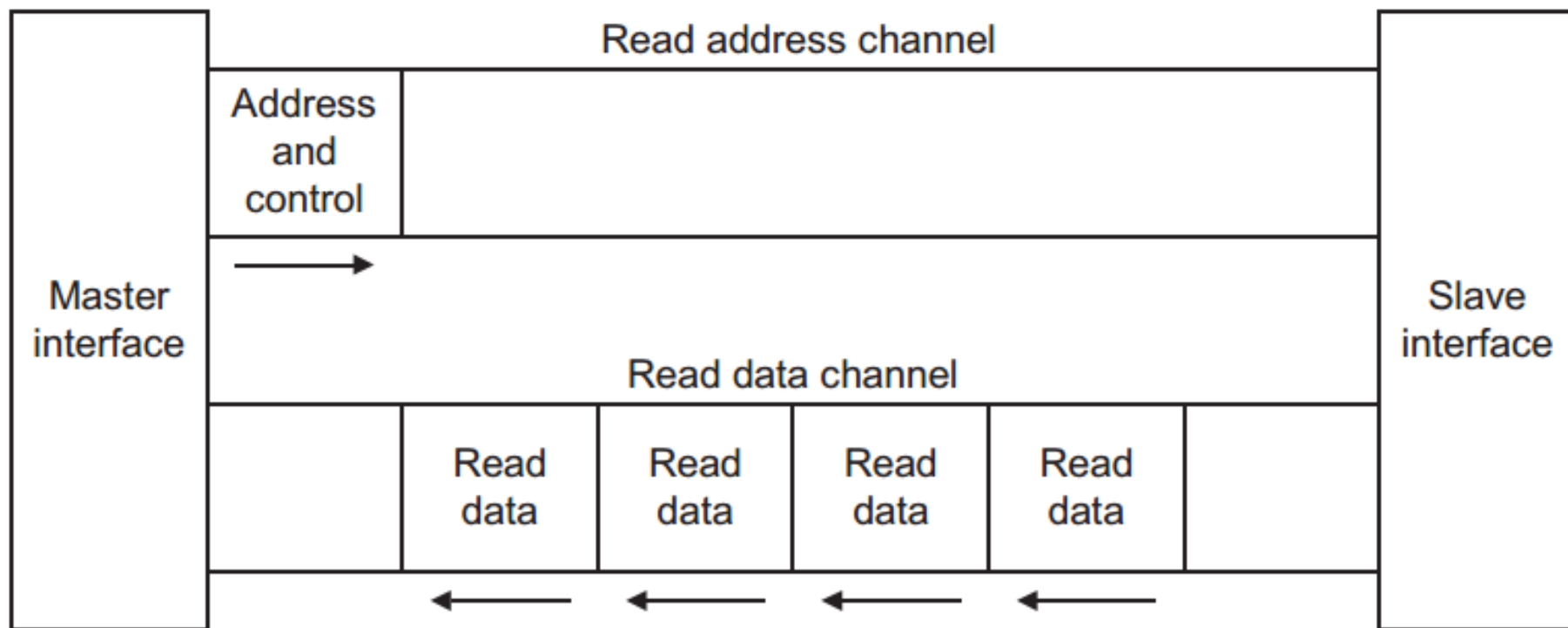


Figure 1-1 Channel architecture of reads

中阶任务指导—— AXI接口

➤ 三个写通道

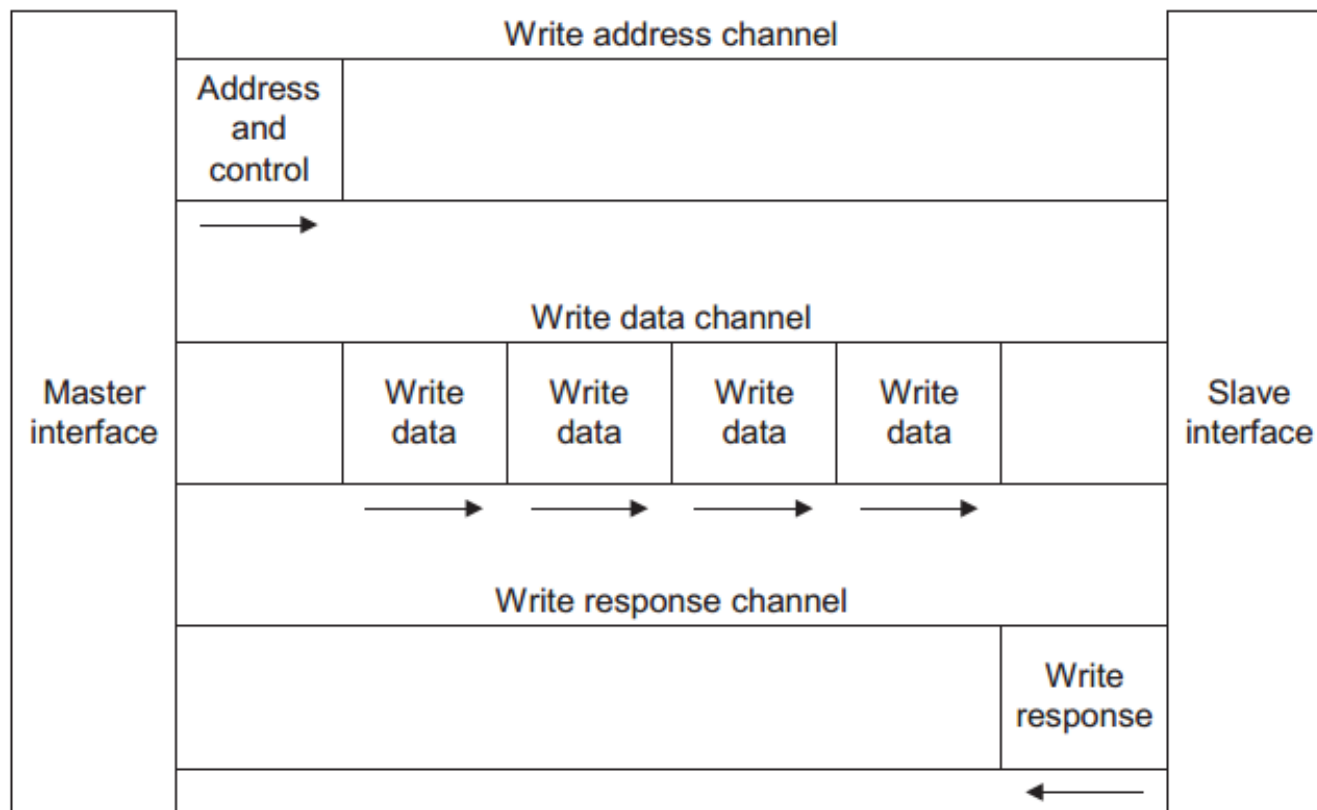


Figure 1-2 Channel architecture of writes

中阶任务指导—— AXI接口

- **AXI接口信号：写地址以aw开头，写数据以w开头，写响应以b开头
读地址以ar开头，读数据以r开头**
- **时钟复位 与 握手信号**

名称	宽度	方向	描述
aclk	1	—>mster/slave	时钟
hresetn	1	—>mster/slave	复位信号，低电平有效
读地址握手			
arvalid	1	—>slave	读请求地址有效
arready	1	—>master	从设备准备好，已接受读地址
读数据握手			
rvalid	1	—>mater	从设备返回数据，读数据有效
rready	1	—>slave	主设备准备好，已接受返回的读数据

中阶任务指导——AXI接口

➤ 握手信号

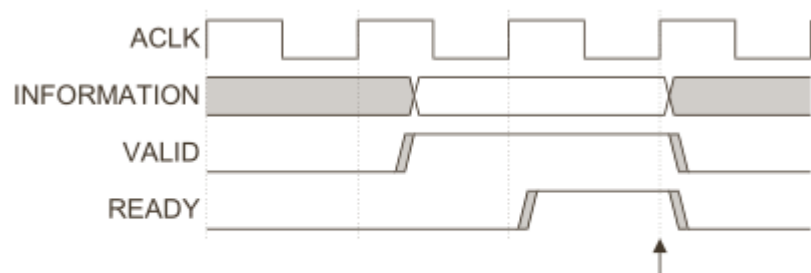


Figure 3-1 VALID before READY handshake

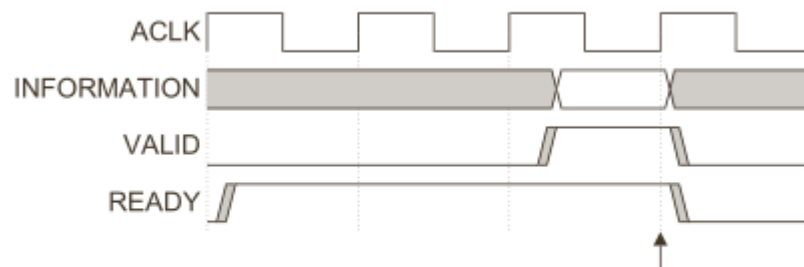


Figure 3-2 READY before VALID handshake

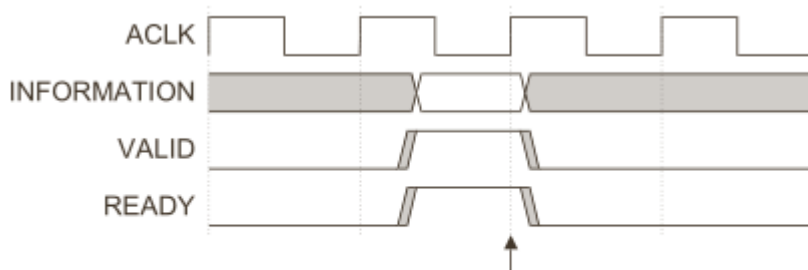


Figure 3-3 VALID with READY handshake

中阶任务指导——AXI接口

➤ 读通道握手依赖关系



Figure 3-4 Read transaction handshake dependencies

➤ 写通道握手依赖关系

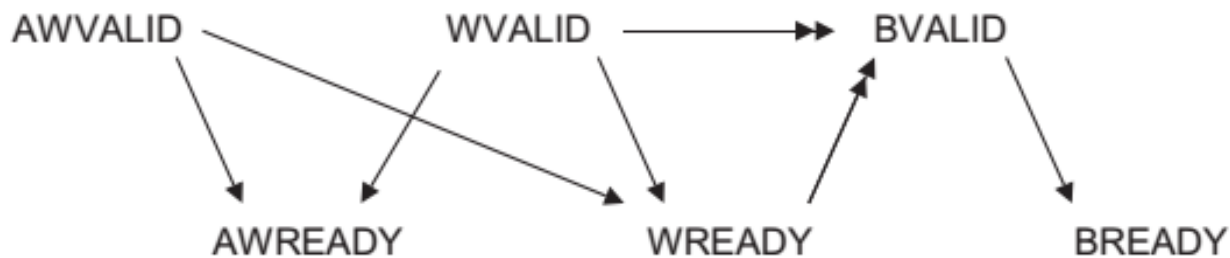


Figure 3-5 Write transaction handshake dependencies

中阶任务指导——AXI接口

➤ 读操作时序

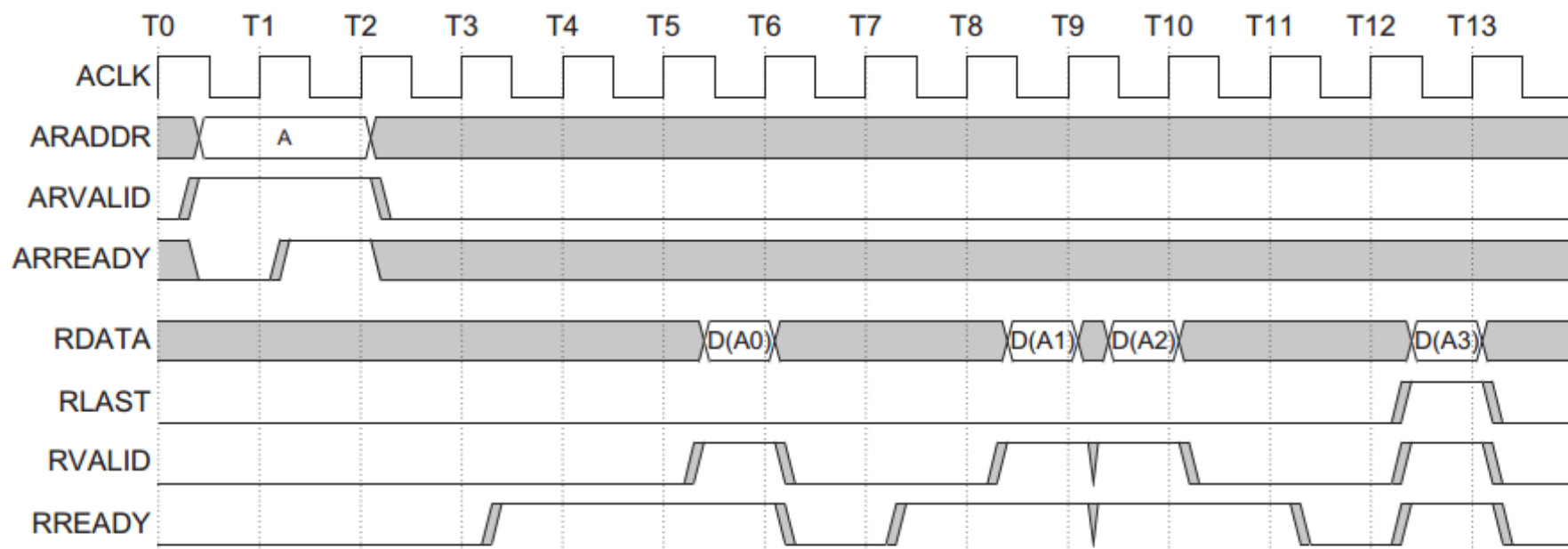


Figure 1-4 Read burst

中阶任务指导——AXI接口

➤ 写操作时序

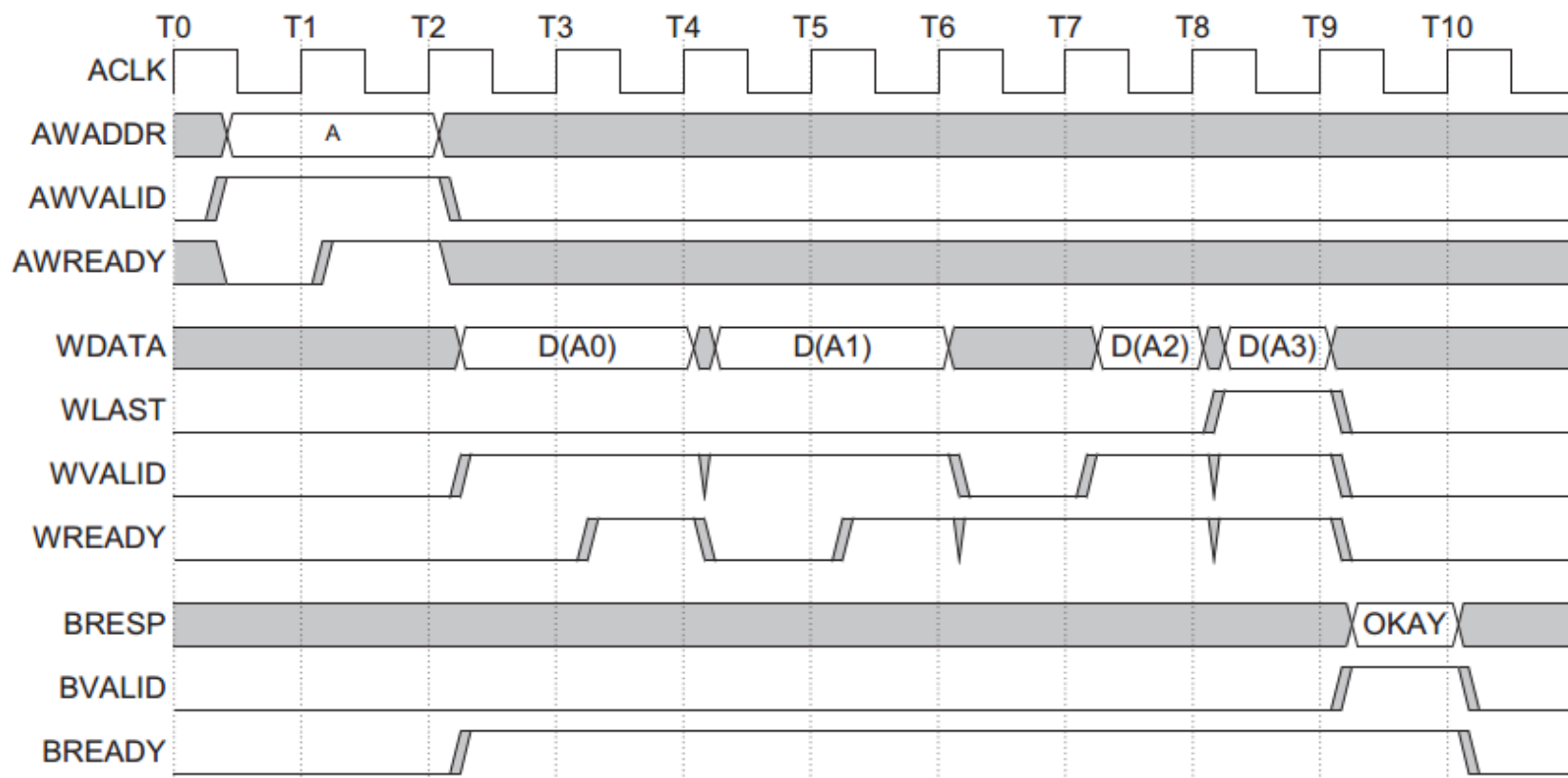


Figure 1-6 Write burst

中阶任务指导——AXI接口

➤ 红色为重点关注，黑色有固定值（参考）

要重点关注的。关于 AXI 接口的时序图，请参考资料 AMBA 协议。

表 1-3 32 位 AXI 接口信号

+

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	固定为 0
araddr	[31:0]	master→slave	读请求的地址	
arlen	[3:0]	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	可忽略
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

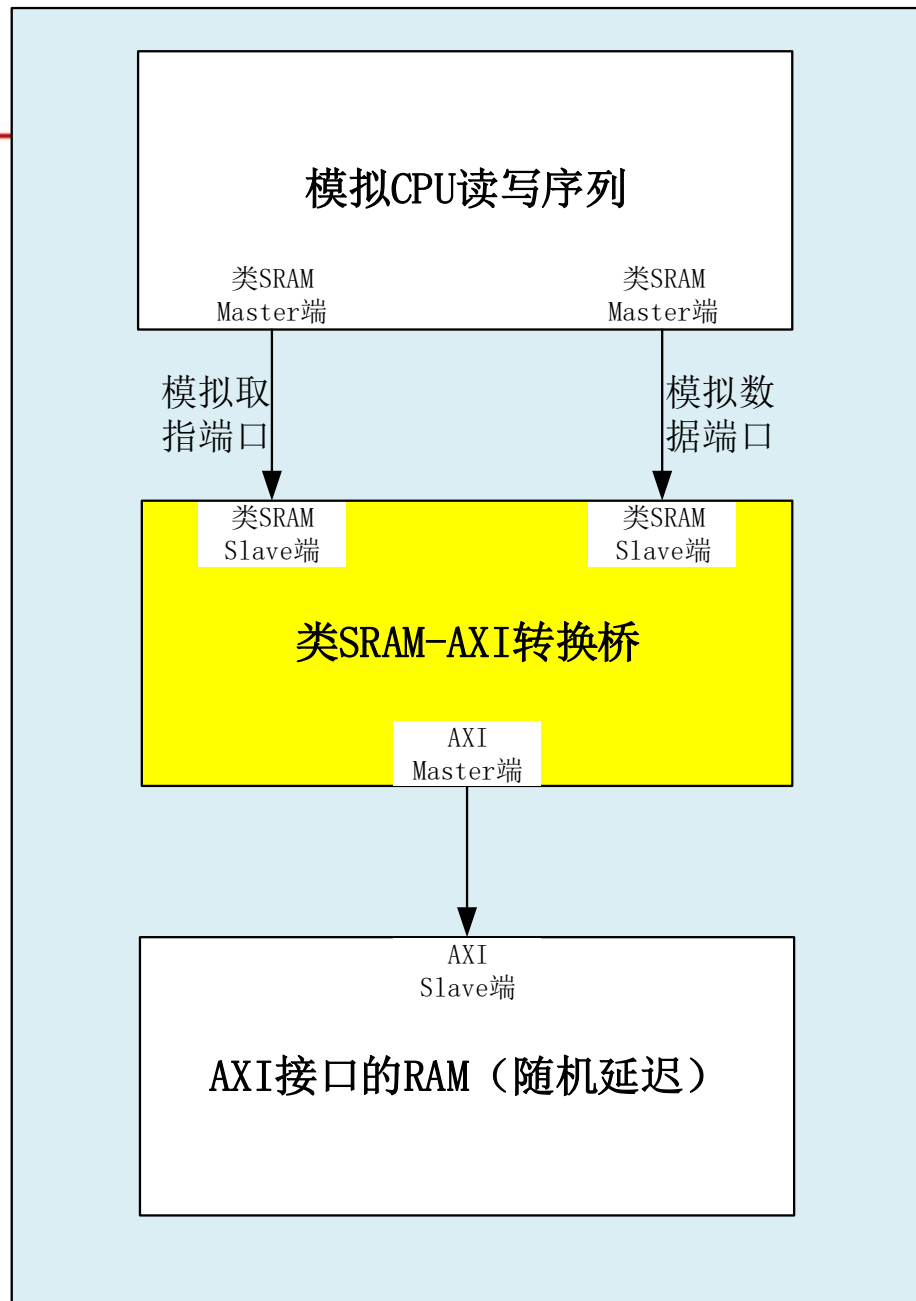
中阶任务指导——AXI接口

➤ 红色为重点关注，黑色有固定值（参考）

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 0
awaddr	[31:0]	master→slave	写请求的地址	
awlen	[3:0]	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 0
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	

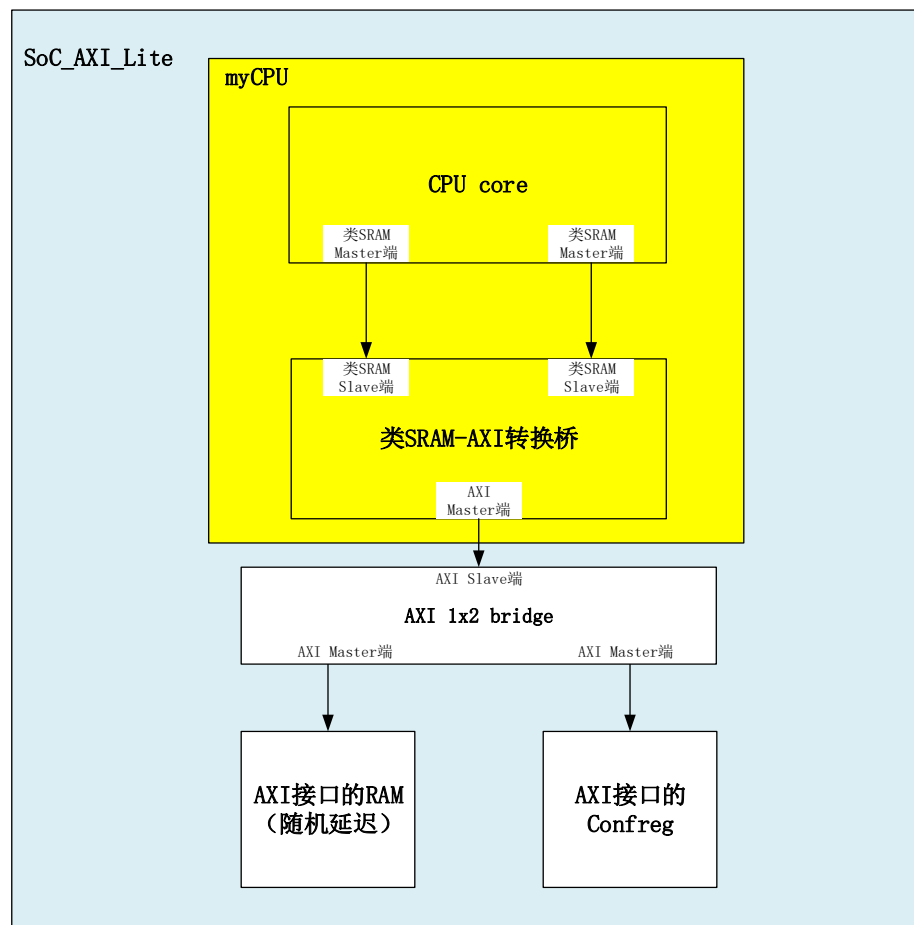
中阶任务指导——仲裁

- 2x1: 两个类SRAM的Slave到一个AXI的Master的桥，
- 仲裁：



中阶任务指导——随机功能测试

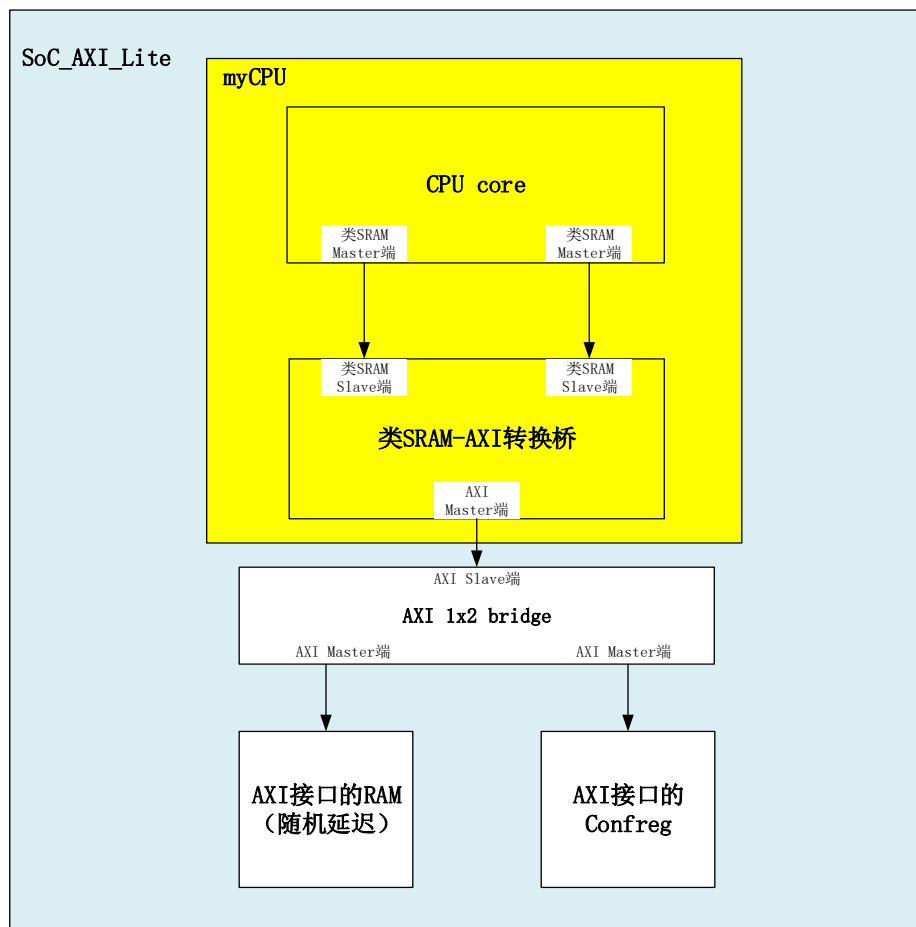
- 功能测试，发布包：**fun_test/**
 - **func_test/soft**：软件程序，MIPS汇编程序
 - **func_test/soc/soc_axi_fun**：用于功能测试的SoC，随机延时版本



中阶任务指导——性能测试

- 性能测试，发布包：**perf_test/**
 - **perf_test/soft**：10个性能测试程序源码
 - **func_test/soc_perf_test**：用于性能测试的SoC，固定延迟
 - 不允许更改编译选项

如果初阶完成了，随机功能测试和性能测试也能顺利进行
难点在于，debug！



中阶任务指导-总结

1. 总线增加握手

类SRAM或AXI接口

2. 取指/数据 仲裁

3. 随机功能测试

SoC_axi_func

4. 性能测试

SoC_axi_perf

预赛内容到此为止

比赛阶段任务解析

初阶任务指导

实验箱使用说明

单周期SRAM接口说明

中阶任务指导

辅助调试手段说明

高阶任务指导

辅助调试手段说明

1. 流程不清晰

参见本ppt

2. 环境不熟悉

Vivado使用说明，交叉编译环境

3. FPGA开发不熟悉

多学多做多试

4. CPU设计知识储备不够

多学多问多思考多讨论

参加本比赛，需要自学很多！

5. 不会调试

仿真工具使用，自学+后续培训

调试指导：仿真 doc/ [仿真调试说明.pdf](#)

上板 doc/ [FPGA在线调试说明.pdf](#)

6. CPU出错不好定位

提升自身调试能力

功能测试Trace比对机制

7. CPU验证不充分

随机测试：仿真时，种子单一，运行速度慢

上板测试：**支持随机种子切换**，运行速度快

一旦发现特点随机种子下有错，仿真运行该随机种子，debug。

比赛阶段任务解析

初阶任务指导

实验箱使用说明

单周期SRAM接口说明

中阶任务指导

辅助调试手段说明

高阶任务指导

决赛各显示神通

- **任务：基于myCPU，向不同方向延伸**
 - **系统方向：myCPU搭建外设丰富的SoC，并启动操作系统，在其上运行应用程序**
 - **专研方向：myCPU提示性能，Cache，双发射，乱序，复杂访存子系统**
 - **综合方向：myCPU既高性能，也可启pmon，但无法启操作系统**
 - **异构方向：myCPU通过AXI接口搭载一个加速核，以加速特定应用的处理**
 - **.....**

- **CPU扩展建议（如要运行pmon或操作系统）：**
 - **实现 非对齐访存指令：LWL/LWR/SWL/SWR**
 - **实现 sync指令：同步访存操作**
 - **实现 Uncache访存：没有Cache也会存在uncache访问的要求，如：LB指令**

总结

预赛

- 初阶：
 - 环境：Vivado+交叉编译
 - 任务：CPU，SRAM接口，无仲裁
 - 能力：Verilog，数字逻辑+组成原理+体系结构，FPGA开发（仿真/综合实现）
 - 运行：功能测试（无延迟版本，SoC_Lite）
- 中阶：
 - 环境：Vivado+交叉编译
 - 任务：增加总线接口，运行复杂测试
 - 能力：+调试能力
 - 运行：随机功能测试（SoC_axi_func），性能测试（SoC_axi_perf）

决赛








- 高阶：
 - 环境：Vivado+交叉编译+很多（串口软件，flash烧写，tftp...）
 - 任务：基于myCPU向各方向延伸
 - 能力：...
 - 运行：自展示

总结——发布包目录

工作 (E:) > nscsc2018 > release



搜索"release"

名称	修改
 doc_v0.02	指导性文档，参考资料 201
 FPGA_test_v1.00	实验箱检测 201
 func_test_v1.01	功能测试，SoC_Lite+SoC_axi_func 201
 perf_test_v1.01	性能测试，SoC_axi_perf 201
 soc_run_os	高阶参考资料，myCPU运行操作系统 201
 Readme_First.txt	201
 大赛指南.docx	201

谢谢！