计算机设计与实践

单周期CPU与SoC设计

2024 · 夏

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

目录

课程基本信息

- 开课学期：大二夏季 (2024夏)
- 课程学分：3.5
- 总学时：56学时 = **4**学时理论 + **52**学时实验
- 项目名称：支持miniRV/miniLA指令集的SoC设计

课程目标

1. **综合**运用《数字逻辑设计》、《计算机组成原理》等基础课程知识，系统掌握计算机硬件系统设计与实现的**工程**方法；
2. 锻炼对计算机硬件系统的**分析**、**设计**和**创新**能力；
3. 通过计算机部件、CPU，再到SoC的逐步实现，了解一个工程项目的开发过程、锻炼**解决复杂工程**问题的能力
4. 掌握HDL、EDA工具的使用，锻炼计算机硬件系统的**开发**、**调试**能力

课程安排

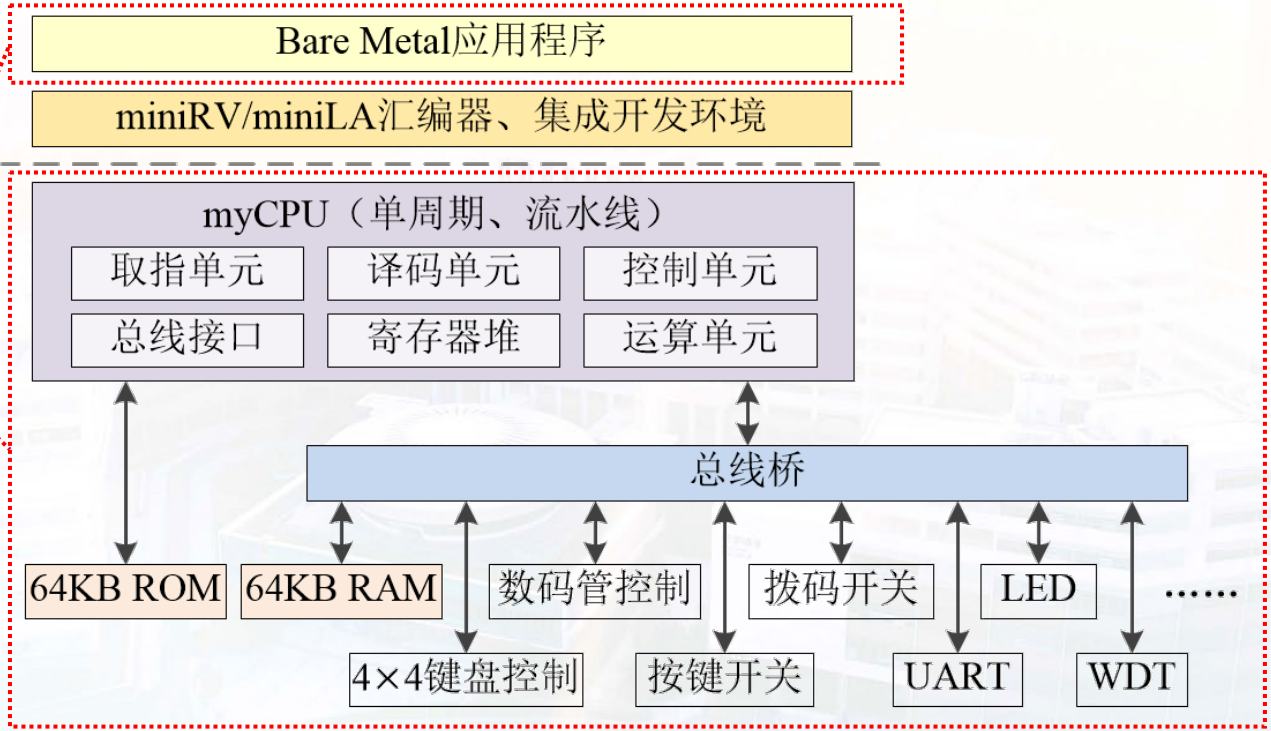
- 总共56学时 = **4**学时理论 + **52**学时实验
- 完成项目：支持miniRV/miniLA指令集的SoC设计

	序号	教学内容	学时
单周期CPU与SoC设计	1-1	单周期CPU与SoC设计要点	2
	1-2	汇编语言程序设计	2
	1-3	单周期CPU与SoC设计与实现	24
流水线CPU设计	2-1	流水线CPU的设计要点	2
	2-2	流水线CPU的设计与实现	22
	2-3	验收、展示&答辩(面向优秀作品，有加分)	4

设计目标

- 设计一个包含CPU、总线与外设的能下板运行和演示的SoC

课程主要设计内容



项目内容

1. 基于miniRV/miniLA指令集，编写汇编程序，在模拟器中运行
2. 使用Verilog HDL，设计**单周期**、**流水线**CPU
3. 基于Trace方法验证CPU功能
4. 为CPU添加总线、外设，形成SoC (System-On-Chip)
5. 将SoC下载到FPGA开发板，并运行1.中的汇编程序

评分：代码&报告(20%) + 项目成绩(含作业)(70%) + 考勤(10%)

一、基础分

及格：完成单周期CPU及SoC的仿真、下板，正确运行要求的汇编程序

中等：达到及格要求的基础上，完成理想流水CPU

良好：达到中等要求的基础上，完成能够处理流水线冲突的流水CPU

优秀：达到良好要求的基础上，将流水线CPU及SoC下板，正确运行要求的汇编程序

扩展或优化内容包括（但不限于）：

1. 自行编写汇编器
2. 优化电路逻辑，在电路设计上优化系统性能（提高主频、降低功耗等）
3. 为其他外设（如键盘、UART等）设计I/O接口，编写测试程序并下板演示
4. 设计实现分支预测、超标量等进阶技术

二、附加分

在每档要求的基础上，完成具有**一定工作量**及**创新性**的额外工作，可获得附加分

实现miniLA指令集，起评分**10**分

平台及参考资料

- 课程实验指导书：comp2012.pages.dev
- 《计算机组成与设计: 硬件/软件接口 (RISC-V版)》
  —— David Patterson, John Hennessy

- riscv/riscv-cores-list: github.com/riscv/riscv-cores-list

RISC-V Cores and SoC Overview

This document captures the status of various cores and SoCs that endeavor to implement the RISC-V specification.
Note that none of these cores/SoCs have passed the in-development RISC-V compliance suite.

Please add to the list and fix inaccuracies - see our CONTRIBUTING file for details.

Cores

Name	Supplier	Links	Capability	Priv. spec	User spec	Pri Lan
Avispado	SemiDynamics	Website	RV64	1.10	RV64GC, 2.2, multicore, V-ready	Syster
Atrevido	SemiDynamics	Website	RV64	1.10	RV64GC, 2.2, multicore, V-ready	Syster
RV32EC_P2	IQonIC Works	Website	RV32	1.11	RV32E[M]C/RV32I[M]C	Syster

计算机设计与实践

单周期CPU与SoC设计

HITSZ 实验与创新实践教育中心

Education Center of Experiments and Innovations, HITSZ

目录

miniRV指令系统

- miniRV是32位整型RISC-V指令集（RV32I）的子集
  - 指令数：37条 (24 + 13) + 3 条例外与中断处理相关
  - 指令类型：R型、I型、S型、B型、U型、J型
  - 指令长度：32位定长
  - 寄存器：32个32位通用寄存器
  - 寻址方式：4种
    - 寄存器寻址、立即数寻址、基址寻址、PC相对寻址

miniRV指令目录

- 算术运算指令 (5) —— add, addi, sub, lui, *auipc (选做)*
- 逻辑运算指令 (6) —— and, andi, or, ori, xor, xori
- 移位运算指令 (6) —— sll, slli, srl, srli, sra, srai
- 加载及存储指令 (8) —— lw, sw, *lb*, *lbu*, *lh*, *lhu*, *sb*, *sh*
- 条件转移指令 (6) —— beq, bne, blt, bge, *bltu*, *bgeu*
- 无条件转移指令 (2) —— jal, jalr
- 比较指令 (4) —— *slt*, *slti*, *sltu*, *sltiu*
- 例外与中断处理 (3) —— *csrrwi*, *mret*, *ecall (选做)*

miniLA指令系统

- miniLA是32位整型LoongArch ISA精简版（LA32R）的子集
  - LoongArch是我们国家自己的指令集，完全自主可控
  - 指令数：38条 (25 + *13*)
  - 指令类型 (按指令格式)：
    - 3R型、2RI5型、2RI12型、1RI20型、2RI16型、I26型
  - 指令长度：32位定长
  - 寄存器：32个32位通用寄存器
  - 寻址方式：4种

miniLA指令目录

- 算术运算指令 (5) —— add.w, addi.w, sub.w, lu12i.w, *pcaddu12i* *(选做)*
- 逻辑运算指令 (6) —— and, andi, or, ori, xor, xori
- 移位运算指令 (6) —— sll.w, slli.w, srl.w, srli.w, sra.w, srai.w
- 加载及存储指令 (8) —— ld.w, st.w, *ld.b*, *ld.bu*, *ld.h*, *ld.hu*, *st.b*, *st.h*
- 条件转移指令 (6) —— beq, bne, blt, bge, *bltu*, *bgeu*
- 无条件转移指令 (3) —— jirl, b, bl
- 比较指令 (4) —— *slt*, *slti*, *sltu*, *sltui*

指令集对比

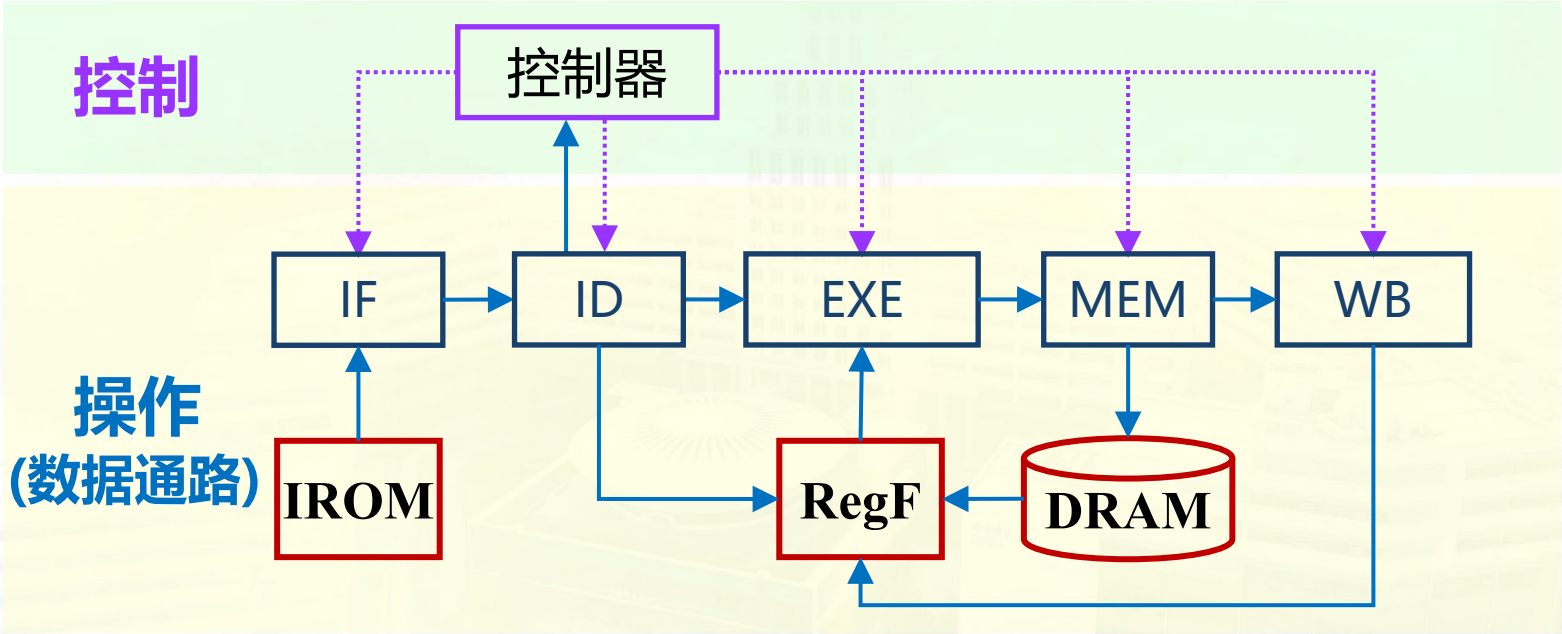
- 两个指令集的指令在功能上一一对应
- 从指令编码角度对比：

对比项	miniRV	miniLA
指令格式	指令格式高度规整 opcode、funct3、funct7位置和长度固定 立即数位序重叠度高 (仅U型指令的位序不同)	指令格式比较规整 opcode位置固定，长度不固定 (4种长度) 立即数位序重叠度一般 (3种位序)
寄存器用途	rs1、rs2始终作为源寄存器 rd始终作为目标寄存器	rj、rk始终作为源寄存器 rd在条件分支指令中充当源寄存器
跳转范围	条件分支：13位偏移量，±1k (32位指令) 直接跳转：21位偏移量，±256k (32位指令)	条件分支：18位偏移量，±32k 直接跳转：28位偏移量，±32M

目录

CPU执行指令的基本过程

- 指令执行的基本过程：IF-ID-EX-MEM-WB，执行受控制信号控制



CPU = 数据通路 + 控制逻辑

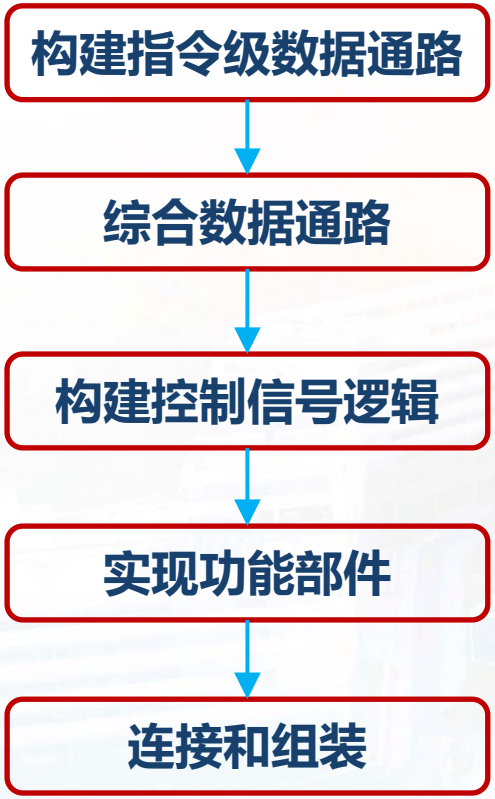
CPU的基本结构

- 从结构上，CPU包含**数据通路**和**控制逻辑**
  - 数据通路
    - 数据流经路径上的所有部件构成的通路，是CPU完成数据处理的物理基础
    - 包括PC、指令存储器、寄存器堆、ALU，etc
  - 控制逻辑
    - 控制运行时数据通路的具体功能，主要指控制器

目录

工程化方法设计CPU

CPU = 数据通路 + 控制逻辑



分析指令功能，以基本部件为起点，构造指令级别的数据通路

当所有指令的数据通路构造完毕后，将其综合为完整的数据通路

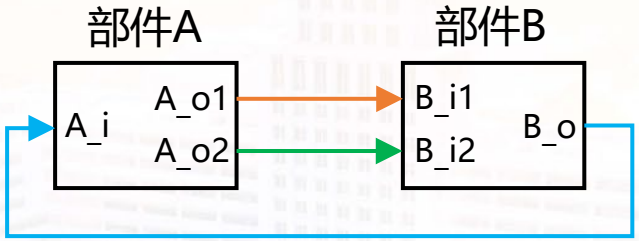
确定控制信号及其取值，建立控制信号表，列出相应的逻辑表达式

提取功能部件，确定功能及接口信号，并使用HDL描述内部具体电路行为或结构

根据数据通路表的连接关系，对各部件进行连接和组装

数据通路的表示

- 数据通路本质上是一个功能部件和连接关系的集合
  - 集合的元素：部件、部件的输入输出信号之间的连接关系



  - 集合表示法：{ A, B, <A.A_o1, B.B_i1>, <A.A_o2, B.B_i2> , <B.B_o, A.A_i>}
  - 表格表示法：

部件A	部件B	
A_i	B_i1	B_i2
B.B_o	A.A_o1	A.A_o2

部件名称

部件的输入信号/引脚

由其他部件产生，并输入到当前引脚的信号

数据通路表模板

- 每条指令对应的部件和连接关系都不同
- 为了方便数据通路的综合，采用表格表示法
  - 建立数据通路表：

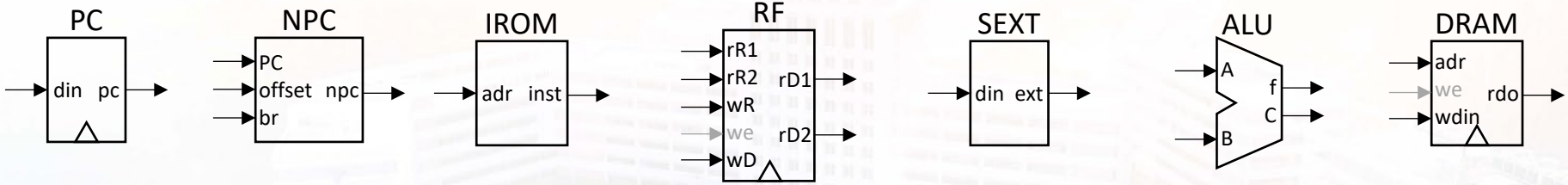
所属单元	取指单元					译码单元					执行单元		存储单元	
功能部件	PC	NPC			IROM	RF				SEXT	ALU		DRAM	
输入信号	din	PC	offset	br	adr	rR1	rR2	wR	wD	din	A	B	adr	wdin

第1-2行：数据通路的功能部件及其所属单元
第 3 行：各功能部件的输入信号/引脚
第 4+行：由其他部件产生，并连接到当前输入引脚的信号
每一行表示一条指令的数据通路

- 第2行所列部件、第3行所列接口信号均仅为参考，应根据实际需要，在设计过程中，进行相应的增删改

基本部件说明

所属单元	取指单元					译码单元					执行单元		存储单元	
功能部件	PC	NPC			IROM	RF				SEXT	ALU		DRAM	
输入信号	din	PC	offset	br	adr	rR1	rR2	wR	wD	din	A	B	adr	wdin



部件	信号名	属性	释义
PC	din	输入	下一条指令地址
	pc	输出	当前指令地址

部件	信号名	属性	释义
NPC	PC	输入	当前指令地址
	offset	输入	跳转的偏移量
	br	输入	条件分支是否跳转
	npc	输出	下一条指令地址

部件	信号名	属性	释义
IROM	adr	输入	指令地址
	inst	输出	指令

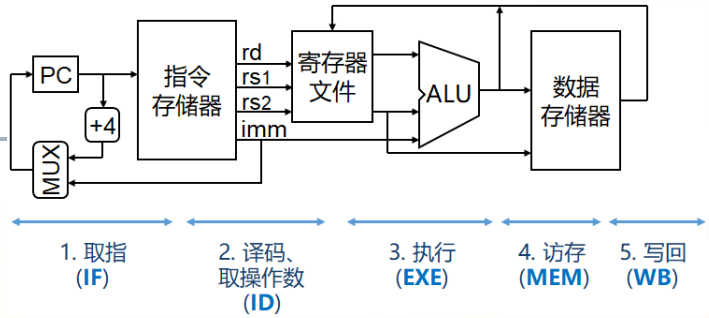
部件	信号名	属性	释义
RF	rR1	输入	读地址1
	rD1	输出	读寄存器数据1
	rR2	输入	读地址2
	rD2	输出	读寄存器数据2
	wR	输入	写地址
	we	输入	写使能
	wD	输入	写寄存器数据

部件	信号名	属性	释义
SEXT	din	输入	指令中的立即数
	ext	输出	扩展后的立即数

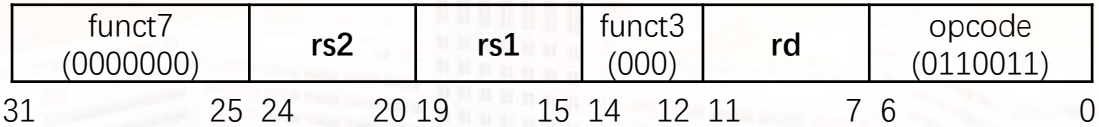
部件	信号名	属性	释义
ALU	A	输入	操作数1
	B	输入	操作数2
	C	输出	运算结果
	f	输出	标志位

部件	信号名	属性	释义
DRAM	adr	输入	访存地址
	rdo	输出	读数据
	we	输入	写使能
	wdin	输入	写数据

指令级数据通路构建

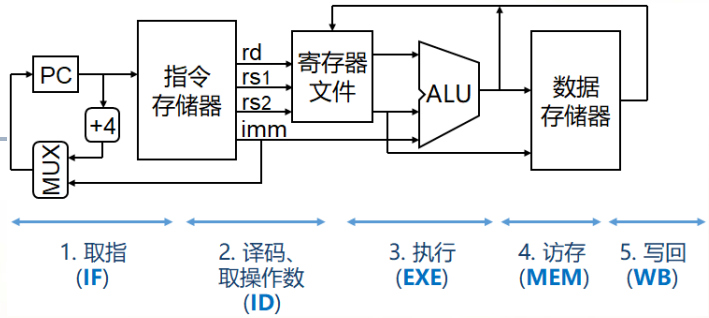


- 示例1：add指令
  - R-类型、加法运算
  - add rd, rs1, rs2　　# (rd) ← (rs1) + (rs2), (PC) ← (PC) + 4
  - 编码格式：

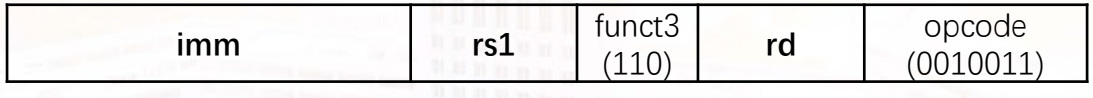


单元	取指单元					译码单元					执行单元		存储单元	
部件	PC	NPC			IROM	RF				SEXT	ALU		DRAM	
输入	din	PC	offset	br	adr	rR1	rR2	wR	wD	din	A	B	adr	wdin
add	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C	-	RF.rD1	RF.rD2	-	-

指令级数据通路构建

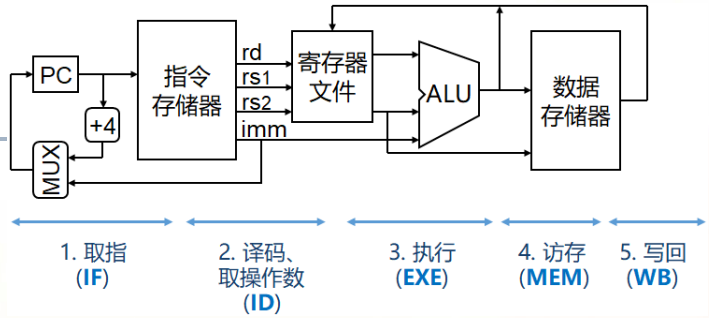


- 示例2：ori指令
  - I-类型、逻辑或运算
  - ori rd, rs1, imm　　# (rd) ← (rs1) | sext(imm), (PC) ← (PC) + 4
  - 编码格式：

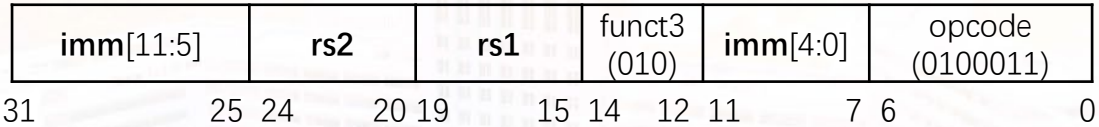


单元	取指单元					译码单元					执行单元		存储单元	
部件	PC	NPC			IROM	RF				SEXT	ALU		DRAM	
输入	din	PC	offset	br	adr	rR1	rR2	wR	wD	din	A	B	adr	wdin
add	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C	-	RF.rD1	RF.rD2	-	-
ori	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	-	IROM.inst[11:7]	ALU.C	IROM.inst[31:20]	RF.rD1	SEXT.ext	-	-

指令级数据通路构建

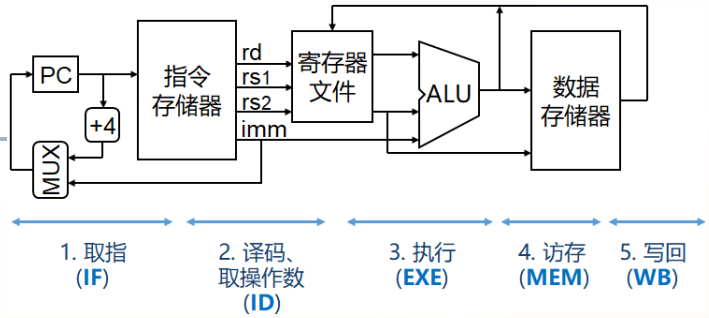


- 示例3：sw指令
  - S-类型、存储操作
  - sw rs2, offset(rs1)　# MEM[(rs1) + sext(offset)] ← (rs2), (PC) ← (PC) + 4
  - 编码格式：

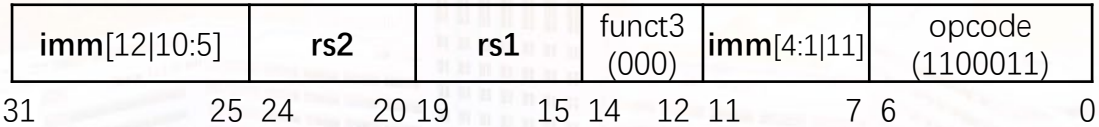


单元	取指单元					译码单元					执行单元		存储单元	
部件	PC	NPC			IROM	RF				SEXT	ALU		DRAM	
输入	din	PC	offset	br	adr	rR1	rR2	wR	wD	din	A	B	adr	wdin
add	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C	-	RF.rD1	RF.rD2	-	-
ori	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	-	IROM.inst[11:7]	ALU.C	IROM.inst[31:20]	RF.rD1	SEXT.ext	-	-
sw	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	-	-	IROM.inst[31:25|11:7]	RF.rD1	SEXT.ext	ALU.C	RF.rD2

指令级数据通路构建

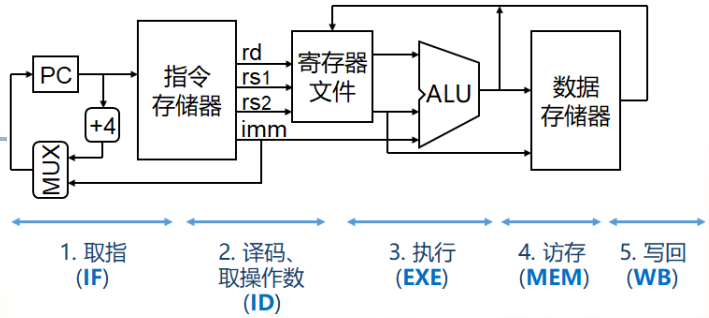


- 示例4：beq指令
  - B-类型、分支跳转
  - beq rs1, rs2, offset　　# (PC) ← (rs1) == (rs2) ? (PC) + sext(offset) : (PC) + 4
  - 编码格式：



单元	取指单元					译码单元					执行单元		存储单元	
部件	PC	NPC			IROM	RF				SEXT	ALU		DRAM	
输入	din	PC	offset	br	adr	rR1	rR2	wR	wD	din	A	B	adr	wdin
add	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C	-	RF.rD1	RF.rD2	-	-
ori	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	-	IROM.inst[11:7]	ALU.C	IROM.inst[31:20]	RF.rD1	SEXT.ext	-	-
sw	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	-	-	IROM.inst[31:25|11:7]	RF.rD1	SEXT.ext	ALU.C	RF.rD2
beq	NPC.npc	PC.pc	SEXT.ext	ALU.f	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	-	-	IROM.inst[31|7|30:25|11:8]	RF.rD1	RF.rD2	-	-

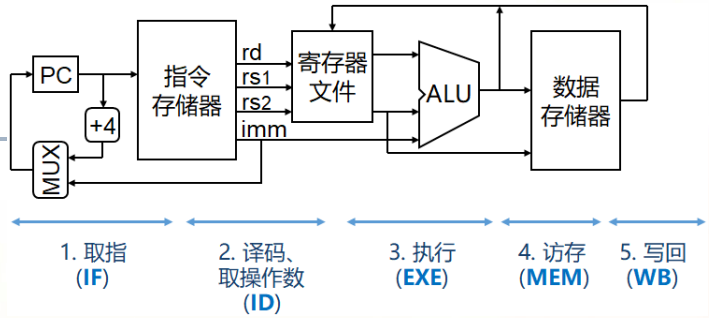
指令级数据通路构建



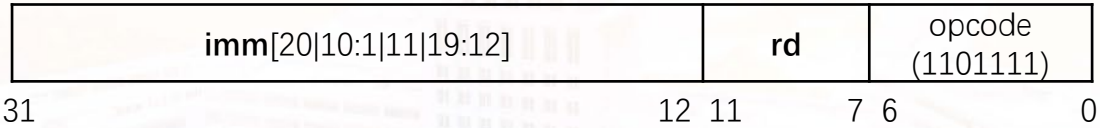
- 示例5：lui指令
  - U-类型、立即数高位赋值操作
  - lui rd, imm　　# (rd) ← sext(imm << 12), (PC) ← (PC) + 4
  - 编码格式：

单元	取指单元					译码单元					执行单元		存储单元	
部件	PC	NPC			IROM	RF				SEXT	ALU		DRAM	
输入	din	PC	offset	br	adr	rR1	rR2	wR	wD	din	A	B	adr	wdin
add	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C	-	RF.rD1	RF.rD2	-	-
ori	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	-	IROM.inst[11:7]	ALU.C	IROM.inst[31:20]	RF.rD1	SEXT.ext	-	-
sw	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	-	-	IROM.inst[31:25|11:7]	RF.rD1	SEXT.ext	ALU.C	RF.rD2
beq	NPC.npc	PC.pc	SEXT.ext	ALU.f	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	-	-	IROM.inst[31|7|30:25|11:8]	RF.rD1	RF.rD2	-	-
lui	NPC.npc	PC.pc	-	-	PC.pc	-	-	IROM.inst[11:7]	SEXT.ext	IROM.inst[31:12]	-	-	-	-

指令级数据通路构建

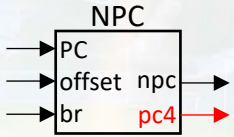


- 示例6：jal指令
  - J-类型、无条件跳转
  - jal rd, offset　　# (rd) ← (PC) + 4, (PC) ← (PC) + sext(offset)
  - 编码格式：



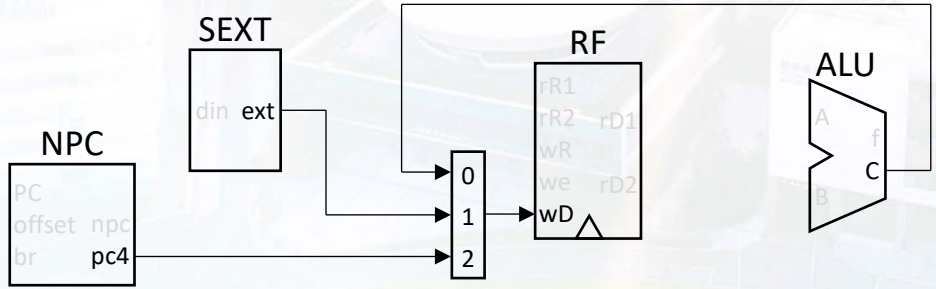
单元	取指单元					译码单元					执行单元		存储单元	
部件	PC	NPC			IROM	RF				SEXT	ALU		DRAM	
输入	din	PC	offset	br	adr	rR1	rR2	wR	wD	din	A	B	adr	wdin
add	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C	-	RF.rD1	RF.rD2	-	-
ori	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	-	IROM.inst[11:7]	ALU.C	IROM.inst[31:20]	RF.rD1	SEXT.ext	-	-
sw	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	-	-	IROM.inst[31:25|11:7]	RF.rD1	SEXT.ext	ALU.C	RF.rD2
beq	NPC.npc	PC.pc	SEXT.ext	ALU.f	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	-	-	IROM.inst[31|7|30:25|11:8]	RF.rD1	RF.rD2	-	-
lui	NPC.npc	PC.pc	-	-	PC.pc	-	-	IROM.inst[11:7]	SEXT.ext	IROM.inst[31:12]	-	-	-	-
jal	NPC.npc	PC.pc	SEXT.ext	-	PC.pc	-	-	IROM.inst[11:7]	NPC.pc4	IROM.inst[31|19:12|20|30:21]	-	-	-	-

- 修改NPC部件以实现jal写回：



综合数据通路

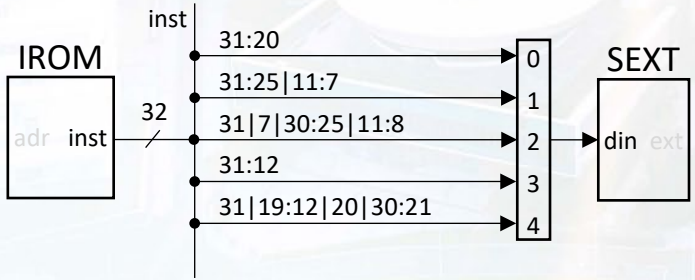
单元	取指单元					译码单元					执行单元		存储单元	
部件	PC	NPC			IROM	RF				SEXT	ALU		DRAM	
输入	din	PC	offset	br	adr	rR1	rR2	wR	wD	din	A	B	adr	wdin
add	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C	-	RF.rD1	RF.rD2	-	-
ori	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	-	IROM.inst[11:7]	ALU.C	IROM.inst[31:20]	RF.rD1	SEXT.ext	-	-
sw	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	-	-	IROM.inst[31:25|11:7]	RF.rD1	SEXT.ext	ALU.C	RF.rD2
beq	NPC.npc	PC.pc	SEXT.ext	ALU.f	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	-	-	IROM.inst[31|7|30:25|11:8]	RF.rD1	RF.rD2	-	-
lui	NPC.npc	PC.pc	-	-	PC.pc	-	-	IROM.inst[11:7]	SEXT.ext	IROM.inst[31:12]	-	-	-	-
jal	NPC.npc	PC.pc	SEXT.ext	-	PC.pc	-	-	IROM.inst[11:7]	NPC.pc4	IROM.inst[31|19:12|20|30:21]	-	-	-	-
综合	NPC.npc	PC.pc	SEXT.ext	ALU.f	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C SEXT.ext NPC.pc4	IROM.inst[31:20] IROM.inst[31:25|11:7] IROM.inst[31|7|30:25|11:8] IROM.inst[31:12] IROM.inst[31|19:12|20|30:21]	RF.rD1	RF.rD2 SEXT.ext	ALU.C	RF.rD2



方法1：多路选择器

综合数据通路

单元	取指单元					译码单元					执行单元		存储单元	
部件	PC	NPC			IROM	RF				SEXT	ALU		DRAM	
输入	din	PC	offset	br	adr	rR1	rR2	wR	wD	din	A	B	adr	wdin
add	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C	-	RF.rD1	RF.rD2	-	-
ori	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	-	IROM.inst[11:7]	ALU.C	IROM.inst[31:20]	RF.rD1	SEXT.ext	-	-
sw	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	-	-	IROM.inst[31:25|11:7]	RF.rD1	SEXT.ext	ALU.C	RF.rD2
beq	NPC.npc	PC.pc	SEXT.ext	ALU.f	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	-	-	IROM.inst[31|7|30:25|11:8]	RF.rD1	RF.rD2	-	-
lui	NPC.npc	PC.pc	-	-	PC.pc	-	-	IROM.inst[11:7]	SEXT.ext	IROM.inst[31:12]	-	-	-	-
jal	NPC.npc	PC.pc	SEXT.ext	-	PC.pc	-	-	IROM.inst[11:7]	NPC.pc4	IROM.inst[31|19:12|20|30:21]	-	-	-	-
综合	NPC.npc	PC.pc	SEXT.ext	ALU.f	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C SEXT.ext NPC.pc4	IROM.inst[31:20] IROM.inst[31:25|11:7] IROM.inst[31|7|30:25|11:8] IROM.inst[31:12] IROM.inst[31|19:12|20|30:21]	RF.rD1	RF.rD2 SEXT.ext	ALU.C	RF.rD2

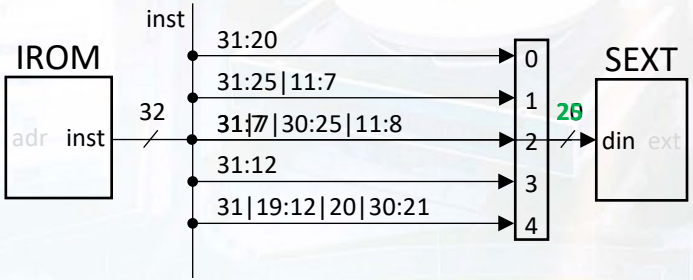


① 存在冗余连接，浪费线网资源

② 不够简洁美观

综合数据通路

单元	取指单元					译码单元					执行单元		存储单元	
部件	PC	NPC			IROM	RF				SEXT	ALU		DRAM	
输入	din	PC	offset	br	adr	rR1	rR2	wR	wD	din	A	B	adr	wdin
add	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C	-	RF.rD1	RF.rD2	-	-
ori	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	-	IROM.inst[11:7]	ALU.C	IROM.inst[31:20]	RF.rD1	SEXT.ext	-	-
sw	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	-	-	IROM.inst[31:25|11:7]	RF.rD1	SEXT.ext	ALU.C	RF.rD2
beq	NPC.npc	PC.pc	SEXT.ext	ALU.f	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	-	-	IROM.inst[31|7|30:25|11:8]	RF.rD1	RF.rD2	-	-
lui	NPC.npc	PC.pc	-	-	PC.pc	-	-	IROM.inst[11:7]	SEXT.ext	IROM.inst[31:12]	-	-	-	-
jal	NPC.npc	PC.pc	SEXT.ext	-	PC.pc	-	-	IROM.inst[11:7]	NPC.pc4	IROM.inst[31|19:12|20|30:21]	-	-	-	-
综合	NPC.npc	PC.pc	SEXT.ext	ALU.f	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C SEXT.ext NPC.pc4	IROM.inst[31:7]	RF.rD1	RF.rD2 SEXT.ext	ALU.C	RF.rD2



方法2：拓宽输入

构造控制逻辑

- 控制逻辑：根据指令的操作码 (opcode)和功能码 (funct3/funct7)，产生执行指令所需的**控制信号**

- 控制信号
  - 操作选择信号　　指令执行时，选择部件要完成的具体操作
    - 多功能部件 (NPC、SEXT、ALU、DRAM等)
  - 多路选择信号　　对多输入源进行选择
    - 多路选择器

添加控制信号

- 在数据通路表中新增控制信号：

单元	取指单元					译码单元					执行单元		存储单元	
部件	PC	NPC			IROM	RF				SEXT	ALU		DRAM	
输入	din	PC	offset	br	adr	rR1	rR2	wR	wD	din	A	B	adr	wdin
add	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C	-	RF.rD1	RF.rD2	-	-
ori	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	-	IROM.inst[11:7]	ALU.C	IROM.inst[31:20]	RF.rD1	SEXT.ext	-	-
sw	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	-	-	IROM.inst[31:25|11:7]	RF.rD1	SEXT.ext	ALU.C	RF.rD2
beq	NPC.npc	PC.pc	SEXT.ext	ALU.f	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	-	-	IROM.inst[31|7|30:25|11:8]	RF.rD1	RF.rD2	-	-
lui	NPC.npc	PC.pc	-	-	PC.pc	-	-	IROM.inst[11:7]	SEXT.ext	IROM.inst[31:12]	-	-	-	-
jal	NPC.npc	PC.pc	SEXT.ext	-	PC.pc	-	-	IROM.inst[11:7]	NPC.pc4	IROM.inst[31|19:12|20|30:21]	-	-	-	-
综合	NPC.npc	PC.pc	SEXT.ext	ALU.f	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C SEXT.ext NPC.pc4	IROM.inst[31:7]	RF.rD1	RF.rD2 SEXT.ext	ALU.C	RF.rD2
操作选择	-	npc_op			-	-	-	rf_we		sext_op	alu_op		ram_we	
多路选择	-	-	-	-	-	-	-	-	rf_wsel	-	-	alub_sel	-	-

确定控制信号取值

- 建立控制信号取值表：
  - 结合数据通路表分析每条指令，确定控制信号取值：

指令	opcode	funct3	funct7	npc_op	rf_we	rf_wsel	sext_op	alu_op	alub_sel	ram_we
add	0110011	000	0000000	00	1	00	-	00	00	0
ori	0010011	110	-	00	1	00	000	01	01	0
sw	0100011	010	-	00	0	-	001	00	01	1
beq	1100011	000	-	01	0	-	010	00	00	0
lui	0110111	-	-	00	1	01	011	-	-	0
jal	1101111	-	-	10	1	10	100	-	-	0

部件	NPC		
输入	PC	offset	br
add	PC.pc	-	-
ori	PC.pc	-	-
sw	PC.pc	-	-
beq	PC.pc	SEXT.ext	ALU.f
lui	PC.pc	-	-
jal	PC.pc	SEXT.ext	-
操作选择	npc_op		

确定控制信号取值

- 建立控制信号取值表：
  - 结合数据通路表分析每条指令，确定控制信号取值：

指令	opcode	funct3	funct7	npc_op	rf_we	rf_wsel	sext_op	alu_op	alub_sel	ram_we
add	0110011	000	0000000	NPC_PC4	1	WB_ALU	-	ALU_ADD	ALUB_RS2	0
ori	0010011	110	-	NPC_PC4	1	WB_ALU	EXT_I	ALU_OR	ALUB_EXT	0
sw	0100011	010	-	NPC_PC4	0	-	EXT_S	ALU_ADD	ALUB_EXT	1
beq	1100011	000	-	NPC_BEQ	0	-	EXT_B	ALU_ADD	ALUB_RS2	0
lui	0110111	-	-	NPC_PC4	1	WB_EXT	EXT_U	-	-	0
jal	1101111	-	-	NPC_JMP	1	WB_PC4	EXT_J	-	-	0

```
`define NPC_PC4 2'b00
`define NPC_BEQ 2'b01
`define NPC_JMP 2'b10
```

NPC_PC4: NPC部件输出PC+4作为下一条指令地址

NPC_BEQ: NPC部件根据ALU标志位确定下一条指令地址

NPC_JMP: NPC部件输出PC+offset作为下一条指令地址

提取功能部件

- 数据通路表的起点：7个基本功能部件

单元	取指单元					译码单元					执行单元		存储单元	
部件	PC	NPC			IROM	RF				SEXT	ALU		DRAM	
输入	din	PC	offset	br	adr	rR1	rR2	wR	wD	din	A	B	adr	wdin
add	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C	-	RF.rD1	RF.rD2	-	-
ori	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	-	IROM.inst[11:7]	ALU.C	IROM.inst[31:20]	RF.rD1	SEXT.ext	-	-
sw	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	-	-	IROM.inst[31:25|11:7]	RF.rD1	SEXT.ext	ALU.C	RF.rD2
beq	NPC.npc	PC.pc	SEXT.ext	ALU.f	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	-	-	IROM.inst[31|7|30:25|11:8]	RF.rD1	RF.rD2	-	-
lui	NPC.npc	PC.pc	-	-	PC.pc	-	-	IROM.inst[11:7]	SEXT.ext	IROM.inst[31:12]	-	-	-	-
jal	NPC.npc	PC.pc	SEXT.ext	-	PC.pc	-	-	IROM.inst[11:7]	NPC.pc4	IROM.inst[31|19:12|20|30:21]	-	-	-	-
综合	NPC.npc	PC.pc	SEXT.ext	ALU.f	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C SEXT.ext NPC.pc4	IROM.inst[31:7]	RF.rD1	RF.rD2 SEXT.ext	ALU.C	RF.rD2
操作选择	-	npc_op			-	-	-	rf_we		sext_op	alu_op		ram_we	
多路选择	-	-	-	-	-	-	-	-	rf_wsel	-	-	alub_sel	-	-

- 数据通路已设计好，功能部件的数量、接口等也相继确定

功能部件清单

◆ 取指单元

① **PC**

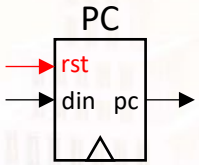
- ✓ 增加复位、时钟

② **NPC**

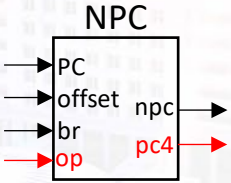
- ✓ 增加PC+4输出 (jal写回)
- ✓ 增加操作选择信号

③ **IROM**

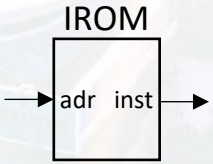
- ✓ 无变化



部件	信号名	属性	释义
PC	din	输入	下一条指令地址
	pc	输出	当前指令地址
	clk	输入	时钟信号
	rst	输入	复位信号



NPC	PC	输入	当前指令地址
	offset	输入	跳转的偏移量
	br	输入	条件分支是否跳转
	npc	输出	下一条指令地址
	pc4	输出	PC+4的值
	op	输入	控制npc的产生



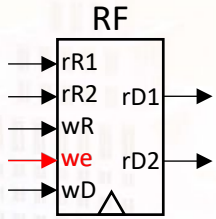
IROM	adr	输入	指令地址
	inst	输出	指令

功能部件清单

◆ 译码单元

① RF

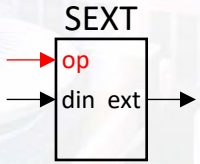
- ✓ 增加时钟
- ✓ 增加操作选择信号(写使能)



部件	信号名	属性	释义
RF	rR1	输入	读地址1
	rD1	输出	读寄存器数据1
	rR2	输入	读地址2
	rD2	输出	读寄存器数据2
	wR	输入	写地址
	we	输入	写使能
	wD	输入	写寄存器数据
	clk	输入	时钟信号

② SEXT

- ✓ 增加操作选择信号



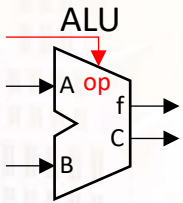
SEXT	din	输入	指令中的立即数
	ext	输出	扩展后的立即数
	op	输入	控制扩展方式

功能部件清单

- 执行、访存单元

① **ALU**

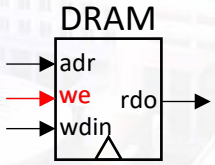
- ✓ 增加操作选择信号



部件	信号名	属性	释义
ALU	A	输入	操作数1
	B	输入	操作数2
	C	输出	运算结果
	f	输出	标志位
	op	输入	控制运算

② **DRAM**

- ✓ 增加时钟
- ✓ 增加操作选择信号(写使能)



部件	信号名	属性	释义
DRAM	adr	输入	访存地址
	rdo	输出	读数据
	we	输入	写使能
	wdin	输入	写数据
	clk	输入	时钟信号

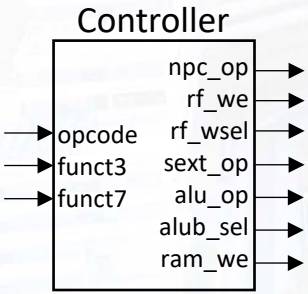
功能部件清单

控制器

- 输出：数据通路表的操作选择信号、多路选择信号

操作选择	-	npc_op			-	-	-	rf_we		sext_op		alu_op		ram_we		
多路选择	-	-	-	-	-	-	-	-	rf_wsel	-		-	alub_sel	-	-	

- 输入：指令的操作码、功能码



部件	信号名	属性	释义
Controller	opcode	输入	指令的操作码
	funct3	输入	指令的3位功能码
	funct7	输入	指令的7位功能码
	npc_op	输出	控制NPC部件产生指令地址
	rf_we	输出	控制寄存器堆的读写操作
	rf_wsel	输出	控制寄存器堆写回数据的来源
	sext_op	输出	控制SEXT部件的符号扩展方式
	alu_op	输出	控制ALU的运算类型
	alub_sel	输出	控制ALU输入口B的数据来源
	ram_we	输出	控制DRAM的读写操作

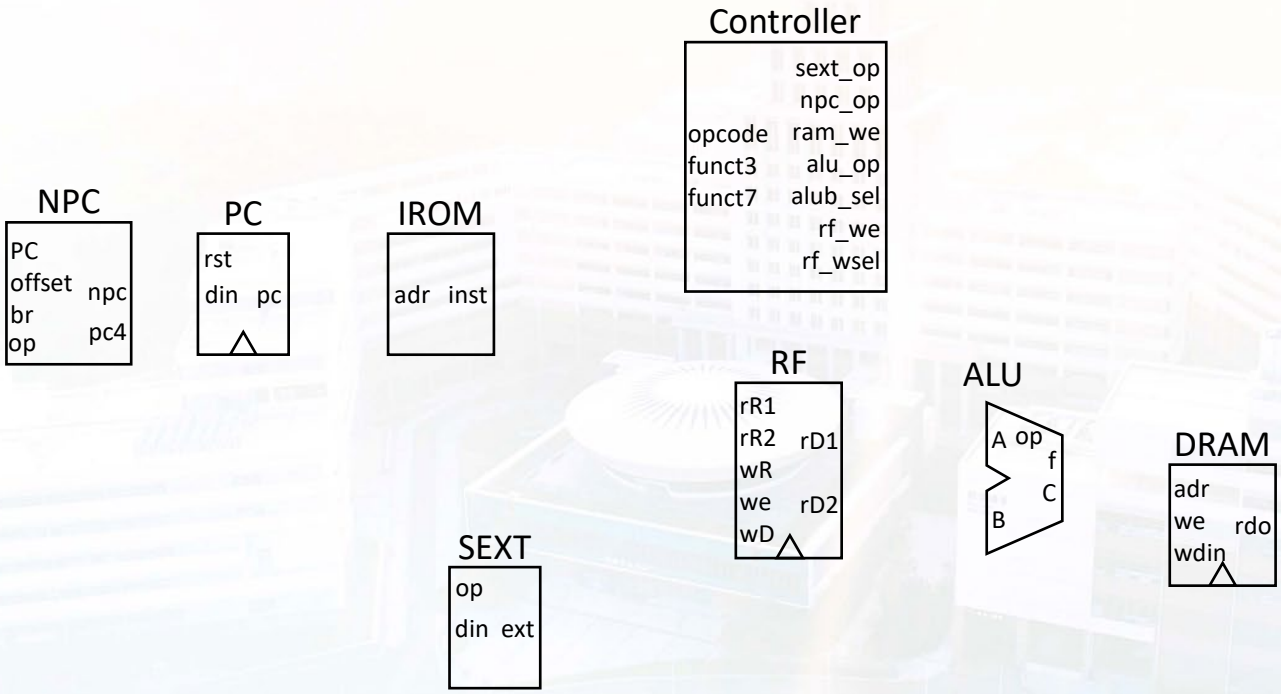
数据通路绘制

- CPU = 数据通路 + 控制器
  - 数据通路 = 功能部件 + 功能部件之间的连接关系
  - 数据通路表描述了功能部件及其连接关系、控制信号
  - 控制信号取值表描述了控制信号的数字逻辑

单元	取指单元					译码单元					执行单元		存储单元	
部件	PC	NPC			IROM	RF				SEXT	ALU		DRAM	
输入	din	PC	offset	br	adr	rR1	rR2	wR	wD	din	A	B	adr	wdin
综合	NPC.npc	PC.pc	SEXT.ext	ALU.f	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C SEXT.ext NPC.pc4	IROM.inst[31:7]	RF.rD1	RF.rD2 SEXT.ext	ALU.C	RF.rD2
操作选择	-	npc_op			-	-	-	rf_we		sext_op	alu_op		ram_we	
多路选择	-	-	-	-	-	-	-	-	rf_wsel	-	-	alub_sel	-	-

数据通路绘制

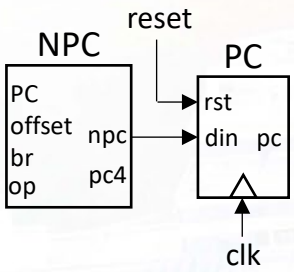
- 排列所有功能部件：

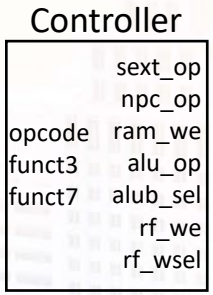


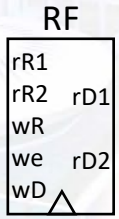
数据通路绘制

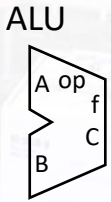
- 连接取指单元：

单元	取指单元				
部件	PC	NPC			IROM
输入	din	PC	offset	br	adr
综合	NPC.npc	PC.pc	SEXT.ext	ALU.f	PC.pc
操作选择	-	npc_op			-
多路选择	-	-	-	-	-





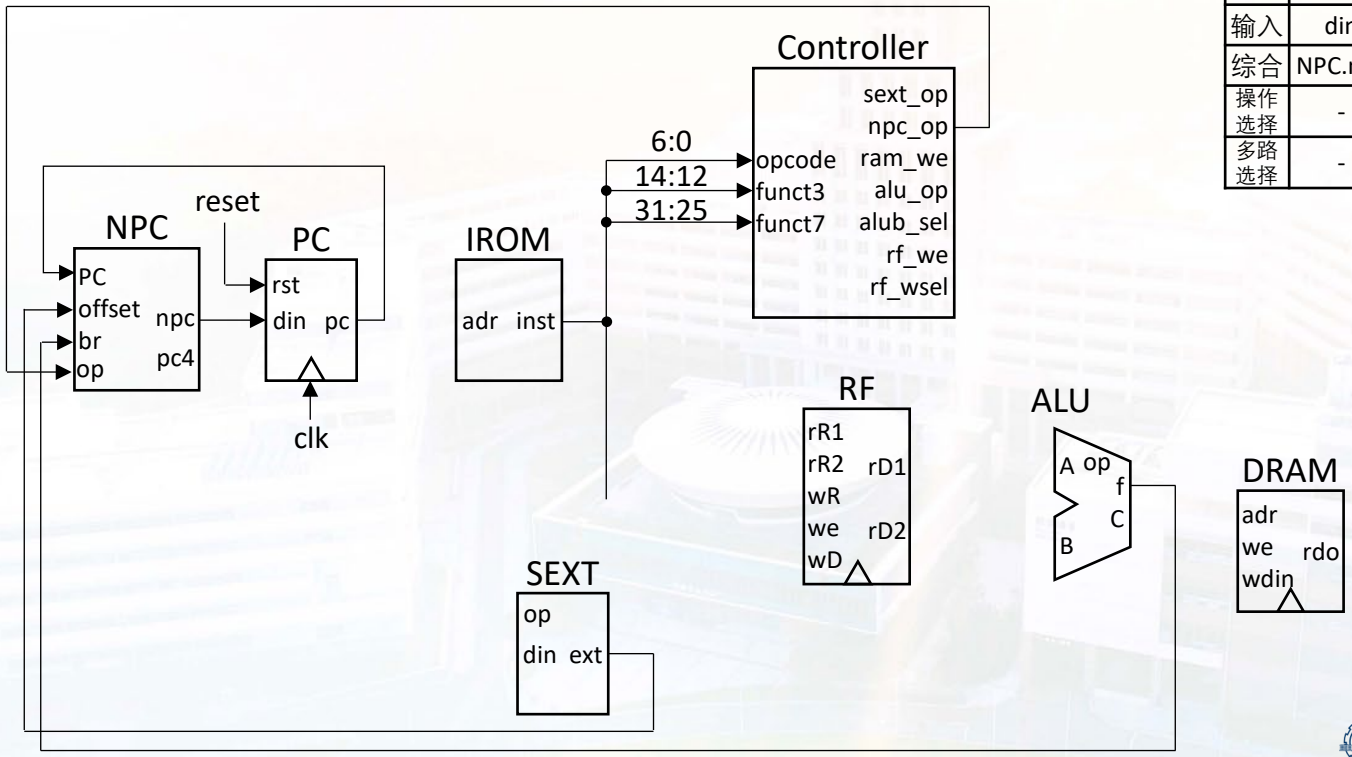




数据通路绘制

- 连接取指单元：

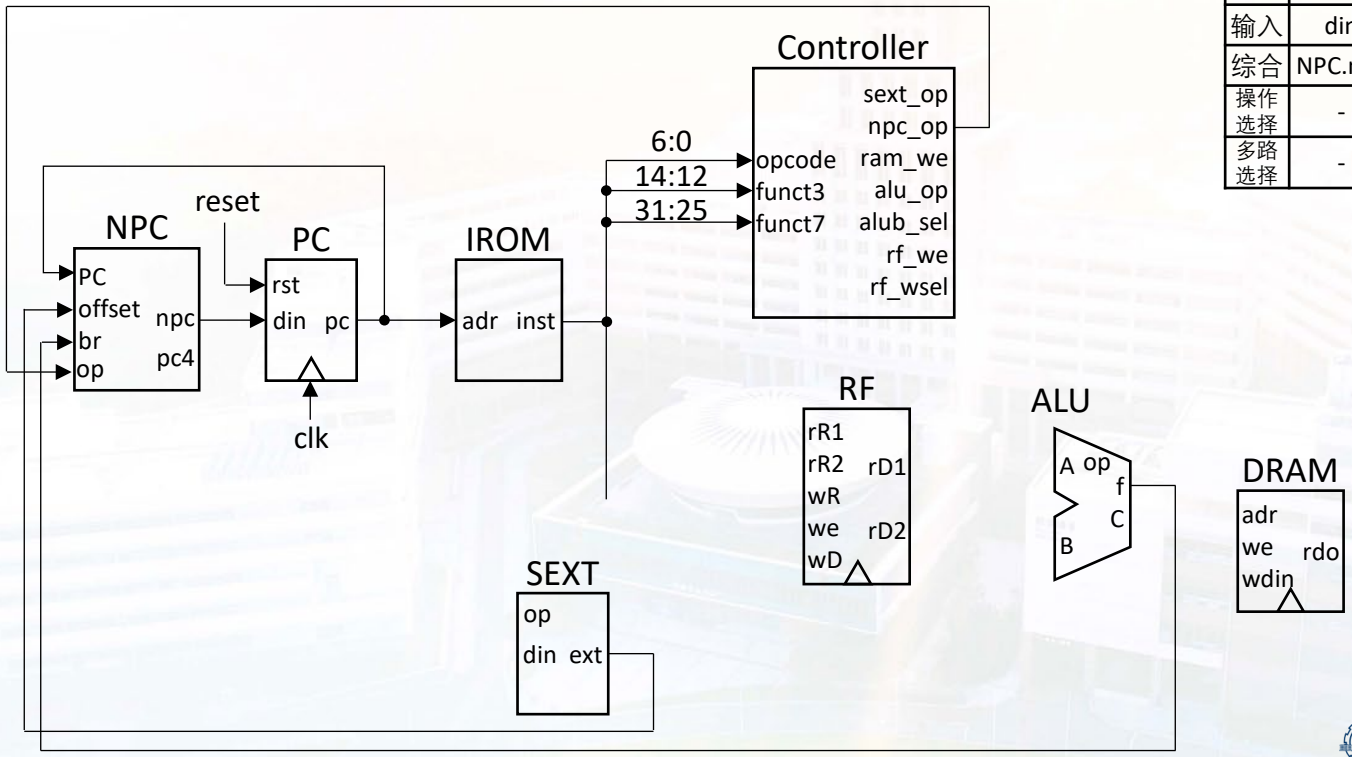
单元	取指单元				
部件	PC	NPC			IROM
输入	din	PC	offset	br	adr
综合	NPC.npc	PC.pc	SEXT.ext	ALU.f	PC.pc
操作选择	-	npc_op			-
多路选择	-	-	-	-	-



数据通路绘制

- 连接取指单元：

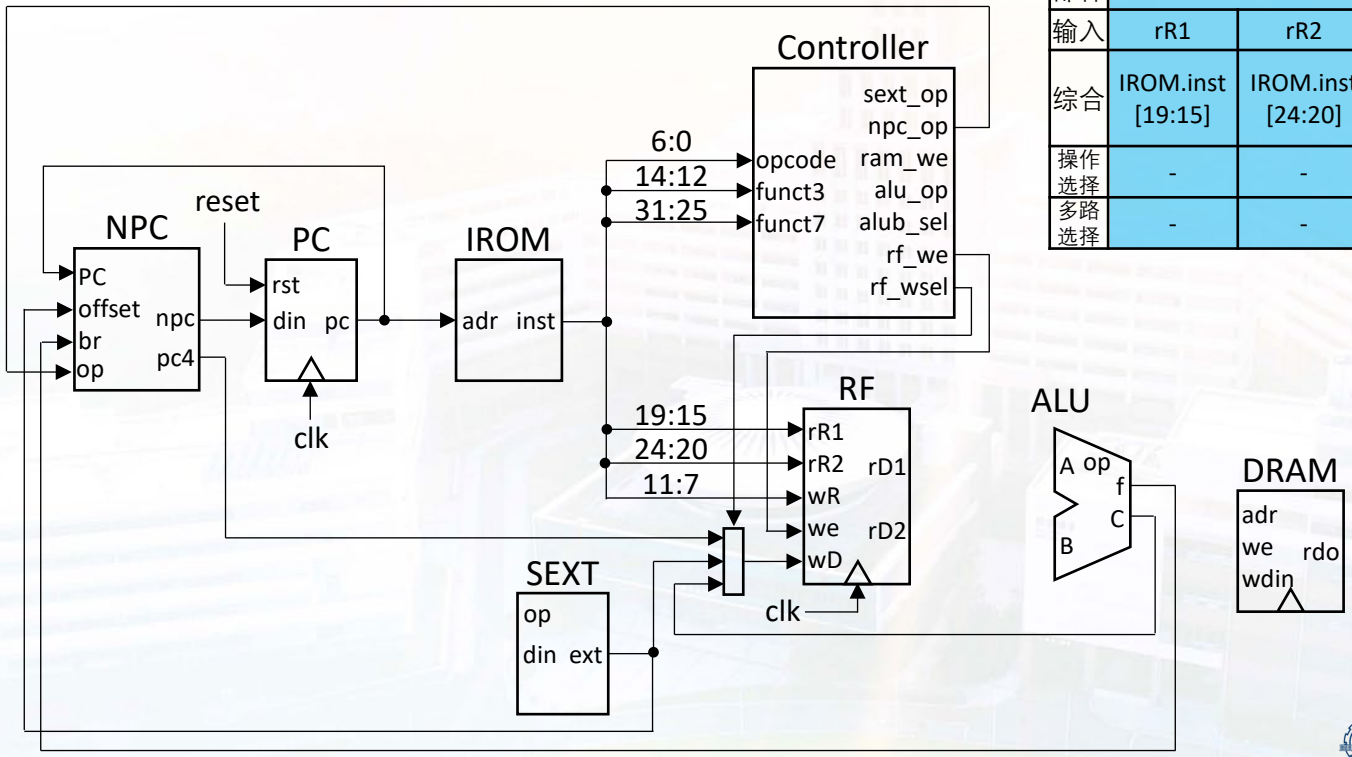
单元	取指单元				
部件	PC	NPC			IROM
输入	din	PC	offset	br	adr
综合	NPC.npc	PC.pc	SEXT.ext	ALU.f	PC.pc
操作选择	-	npc_op			-
多路选择	-	-	-	-	-



数据通路绘制

- 连接译码单元：

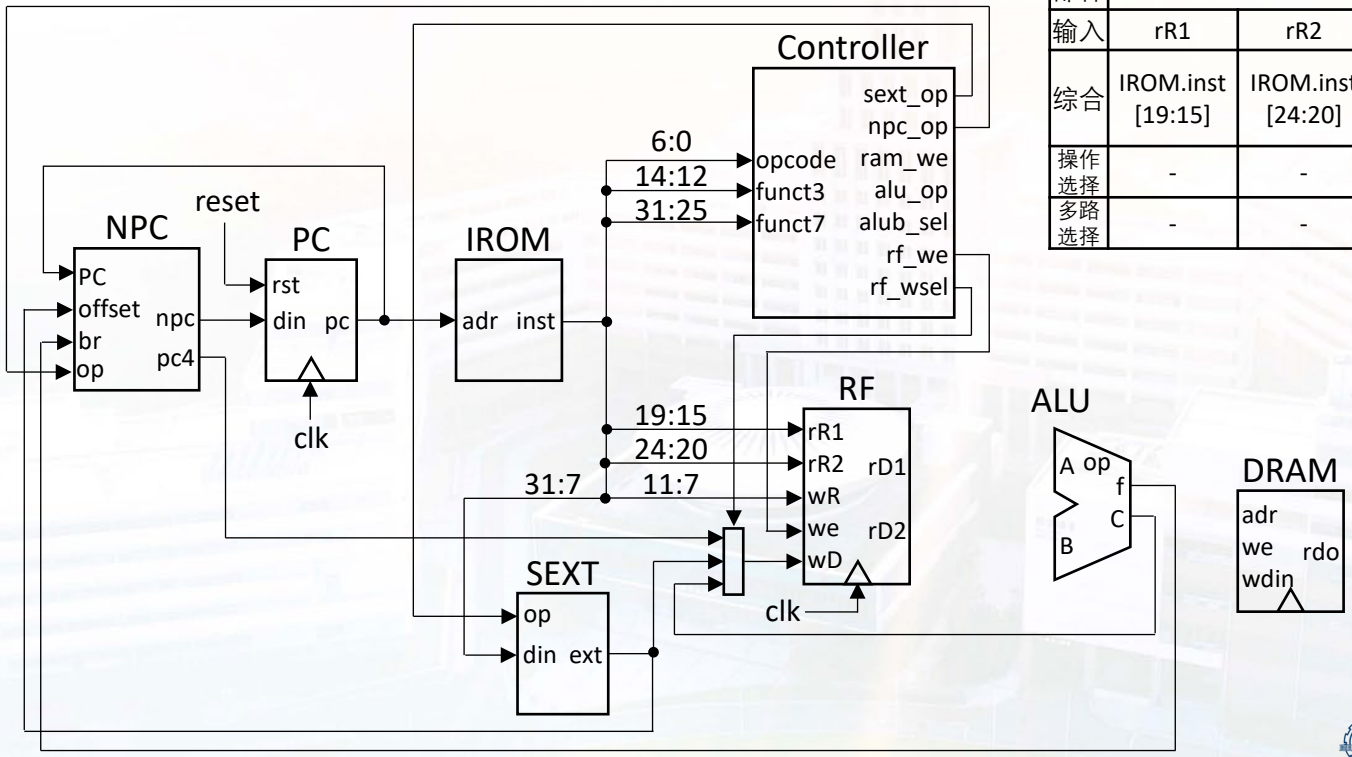
单元	译码单元				
部件	RF				SEXT
输入	rR1	rR2	wR	wD	din
综合	IROM.inst [19:15]	IROM.inst [24:20]	IROM.inst [11:7]	ALU.C SEXT.ext NPC.pc4	IROM.inst [31:7]
操作选择	-	-	rf_we		sext_op
多路选择	-	-	-	rf_wsel	-



数据通路绘制

- 连接译码单元：

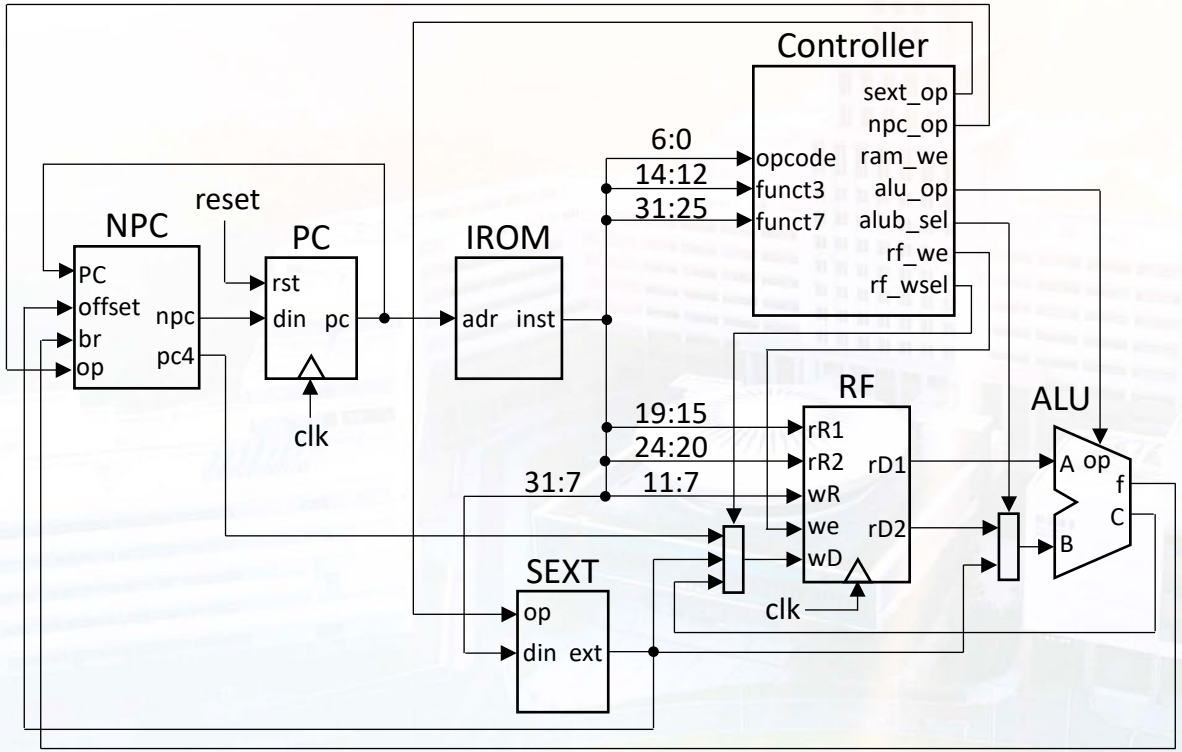
单元	译码单元				
部件	RF				SEXT
输入	rR1	rR2	wR	wD	din
综合	IROM.inst [19:15]	IROM.inst [24:20]	IROM.inst [11:7]	ALU.C SEXT.ext NPC.pc4	IROM.inst [31:7]
操作选择	-	-	rf_we		sext_op
多路选择	-	-	-	rf_wsel	-



数据通路绘制

- 连接执行单元、存储单元：

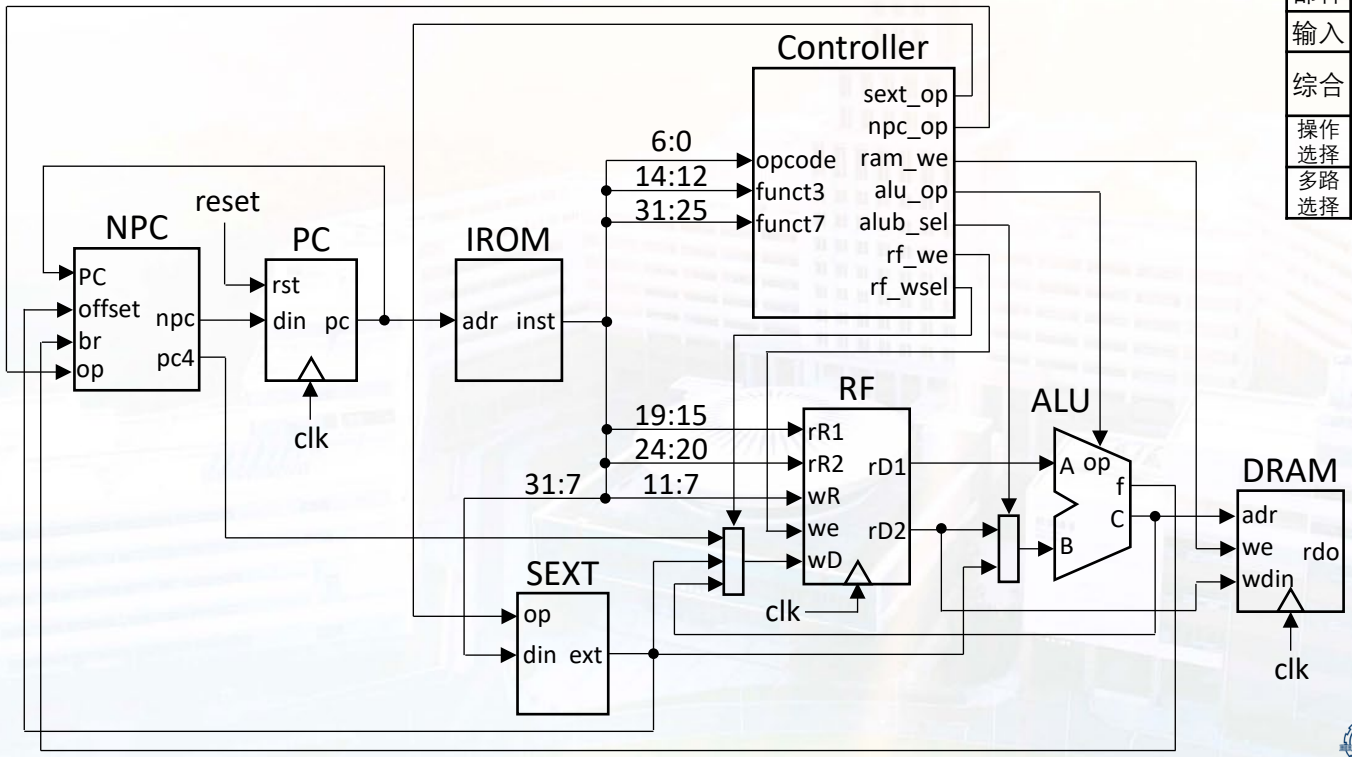
单元	执行单元		存储单元	
部件	ALU		DRAM	
输入	A	B	adr	wdin
综合	RF.rD1	RF.rD2 SEXT.ext	ALU.C	RF.rD2
操作选择	alu_op		ram_we	
多路选择	-	alub_sel	-	-



数据通路绘制

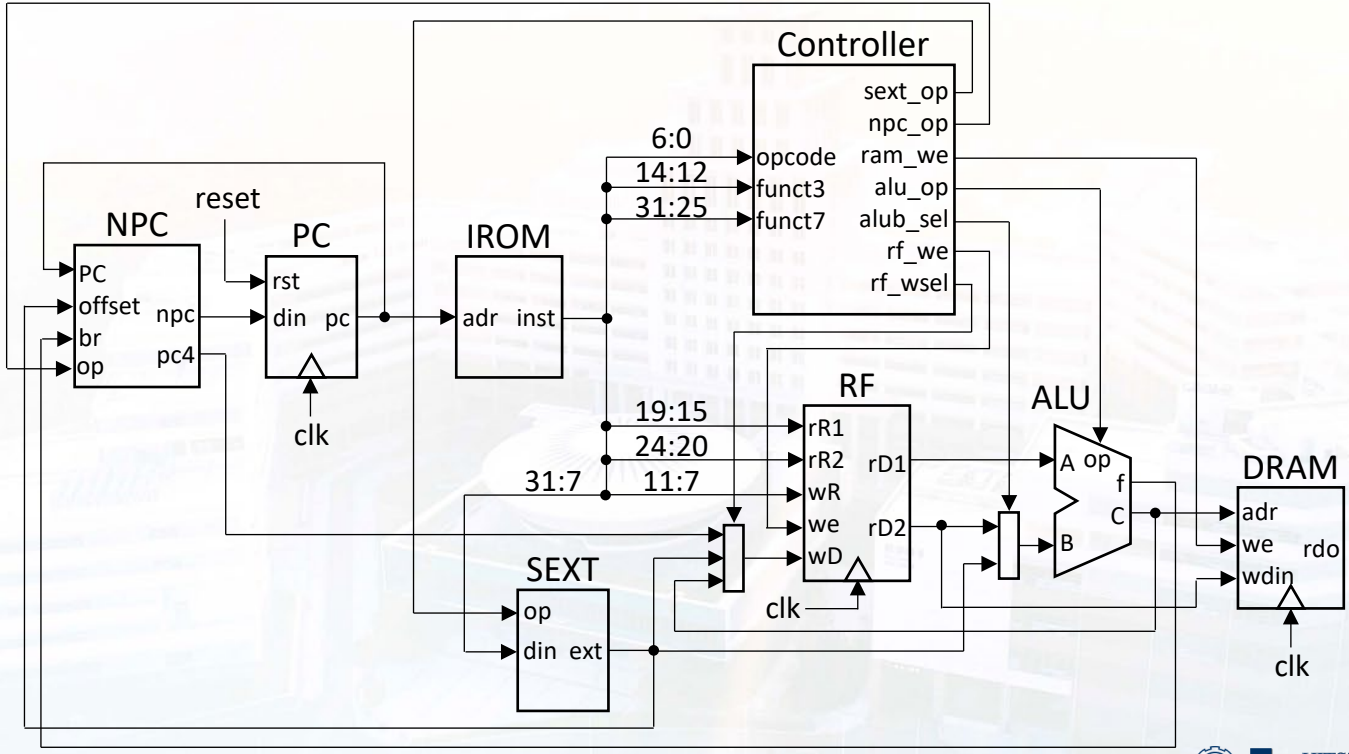
- 连接执行单元、存储单元：

单元	执行单元		存储单元	
部件	ALU		DRAM	
输入	A	B	adr	wdin
综合	RF.rD1	RF.rD2 SEXT.ext	ALU.C	RF.rD2
操作选择	alu_op		ram_we	
多路选择	-	alub_sel	-	-



数据通路绘制

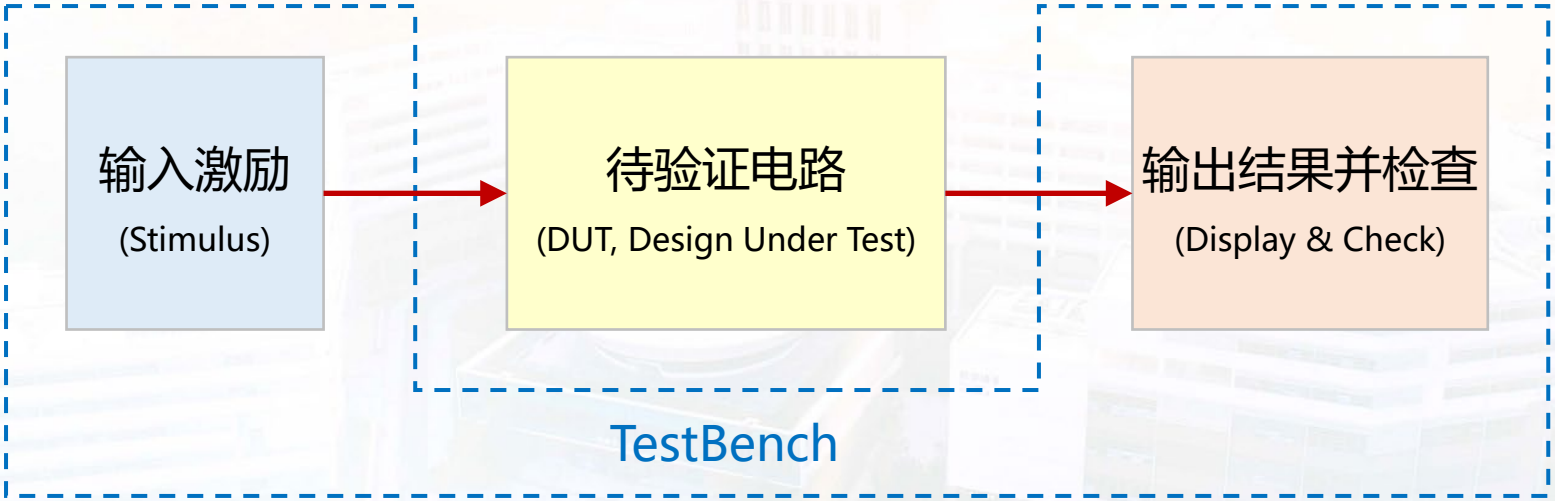
- 完整数据通路示例（6条指令）：



目录

数字电路的功能验证

- 如何对数字电路的功能进行验证？ —— 时序仿真
- 仿真 —— 基于软件模拟 (而非电路实测) 来验证电路功能的方法

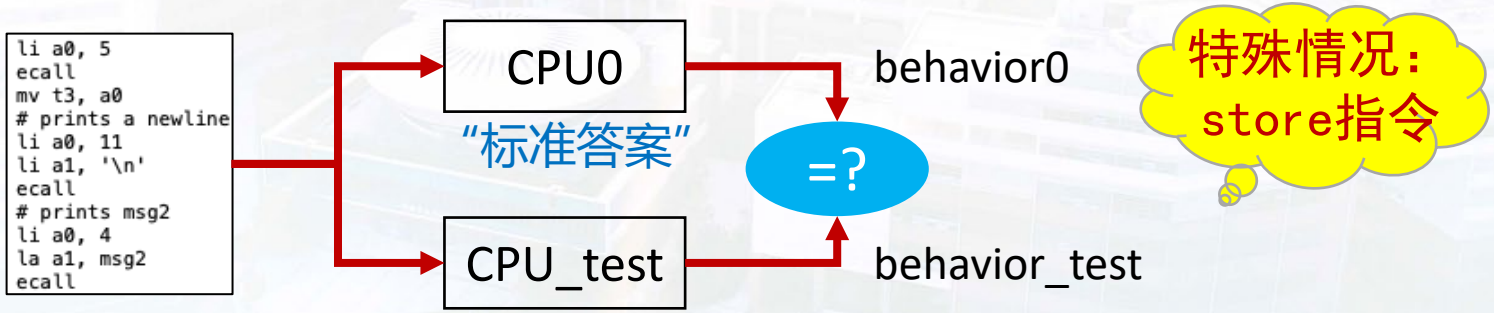


CPU的功能验证

- CPU也是数字电路，也可采用相同的验证方法，但效率太低
  - 只能分模块验证，或借助外设输出
  - 出错点通过测试程序的逻辑路径传递很远之后才能被发现
- 改进：
  - 用**一段指令序列**作为输入激励，通过**观察程序执行结果**来验证CPU功能
  - 验证效率大幅提高，但难以定位错误
- 更好的方法：基于Trace比对的功能验证

CPU的功能验证 —— 基于Trace比对

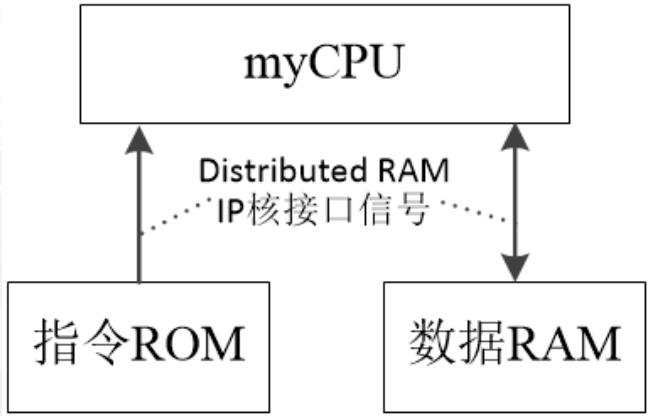
- Trace：CPU执行指令序列时产生的信息 (PC、写寄存器的信息，etc)
- 基于Trace比对的验证方法：
  - ① 用已知功能正确的CPU运行测试程序，记录Trace0 (Golden Trace)
  - ② 用待验证CPU运行测试程序，产生Trace1
  - ③ 将Trace1与Trace0进行实时比对，如果出现不同，立即报错并停止



目录

SoC设计

- SoC (System-on-Chip，片上系统) 在单芯片上集成数字信号处理器、微处理器、数据转换器、接口电路等模块，可直接实现信号的采集、转换、处理、存储、通信等功能
- 没有总线和外设的CPU是“光杆司令”，没有实用价值
- 如何改造?
  - 提取RAM IP核接口信号总线
    作为系统总线 —— DRAM总线



SoC设计 —— 总线

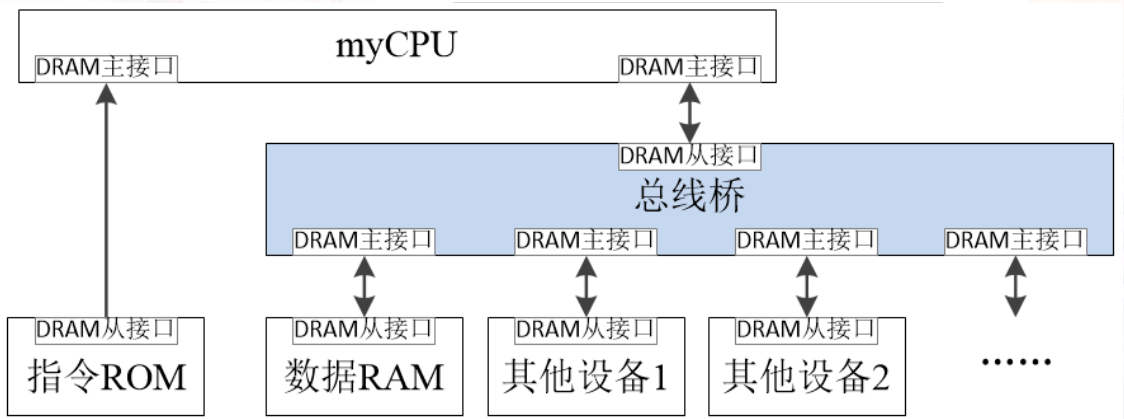
- CPU与存储器通过DRAM总线接口一对一连接
  - 如何连接其他设备?
  - 增加**总线桥**

中转机构:

访问请求转发

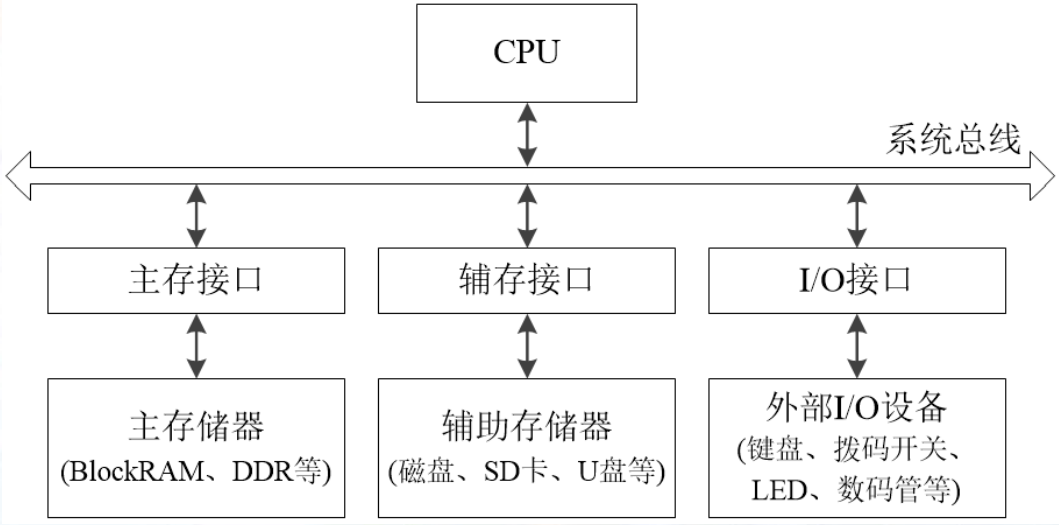
控制机构:

总线仲裁



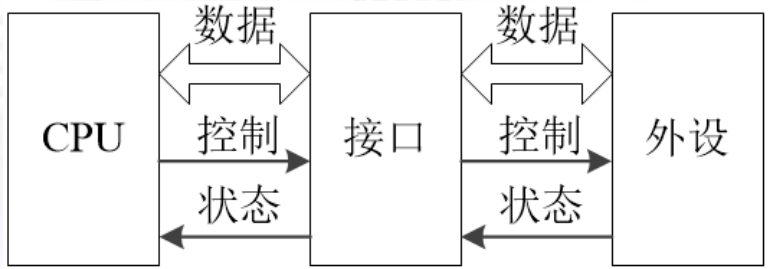
SoC设计 —— 接口

- CPU通过接口连接“外部世界”，接口负责在CPU与“外部世界”之间中转各种信息
  - 接口包括存储器接口、I/O接口



SoC设计 —— 接口

- CPU通过接口连接“外部世界”，接口负责在CPU与“外部世界”之间中转各种信息
  - 接口受CPU控制，外部设备受接口控制

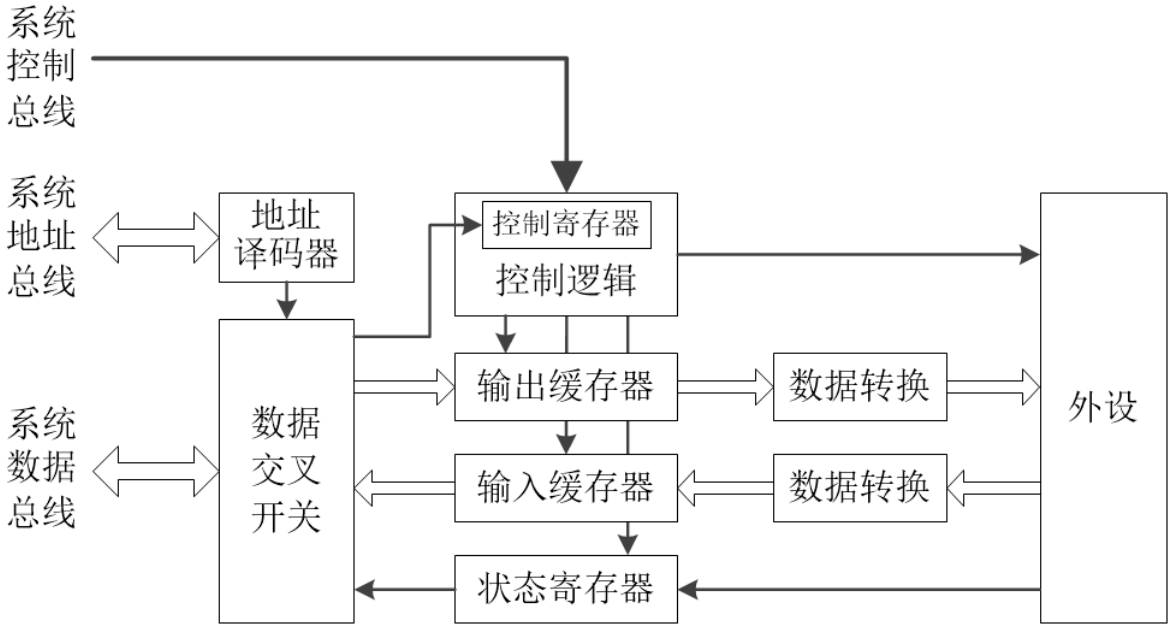


SoC设计 —— 接口的作用与功能

- 信号转换功能/预处理功能
  - 完成总线信号与I/O设备信号之间的转换，如电平转换、串并转换等
- 数据缓存功能
  - 缓存CPU和外设之间的数据，相应的缓存器称为数据口
- 接受和执行CPU命令的功能
  - 使用寄存器存放来自CPU的命令，该寄存器称为命令口
- 控制和监视外设执行的功能
  - 状态寄存器存储外设状态，称为状态口
- 设备选择功能/选址功能
  - 根据访问地址选择相应的I/O接口或接口中的设备

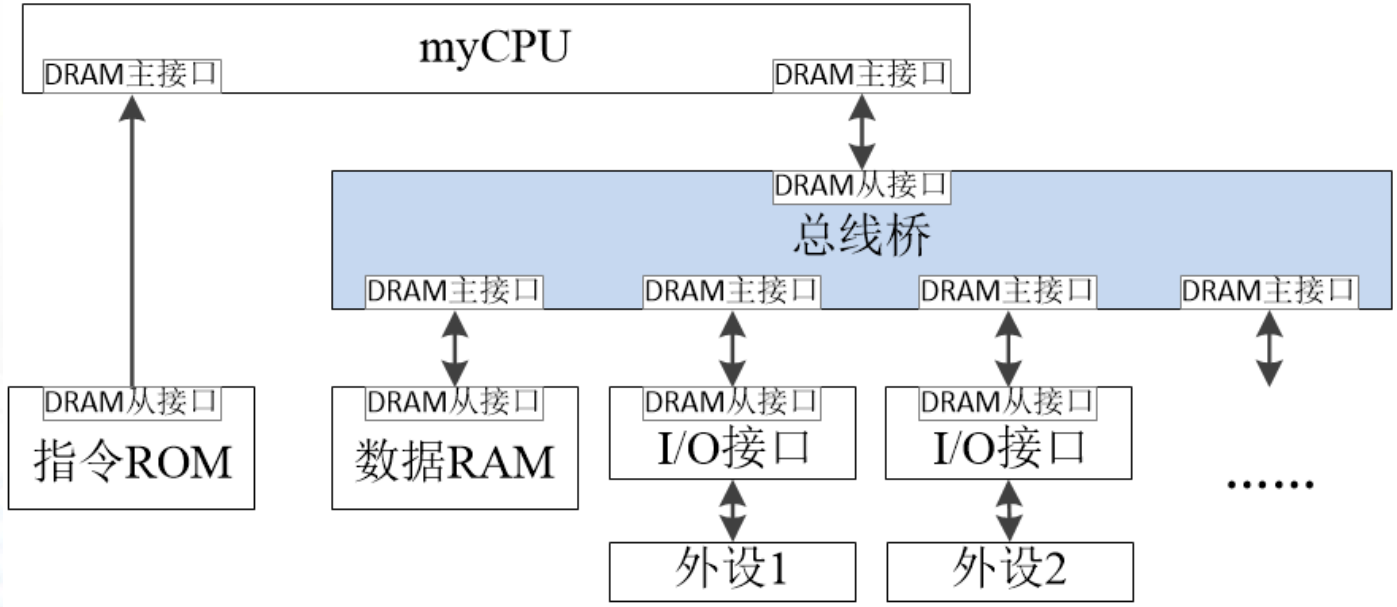
SoC设计 —— 接口的设计

- I/O接口电路的核心：数据口、命令口、状态口、控制逻辑
- CPU通过地址访问外设——基地址确定I/O接口，偏移地址确定具体端口



SoC设计

- 带有总线和外设的SoC架构



作业 (DDL: 7月12日 23:00) 尽早完成!

- 完成数据通路和控制信号的构造、综合

单元	取指单元					译码单元					执行单元		存储单元	
部件	PC	NPC			IROM	RF				SEXT	ALU		DRAM	
输入	din	PC	offset	br	adr	rR1	rR2	wR	wD	din	A	B	adr	wdin
add	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C	-	RF.rD1	RF.rD2	-	-
ori	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	-	IROM.inst[11:7]	ALU.C	IROM.inst[31:20]	RF.rD1	SEXT.ext	-	-
sw	NPC.npc	PC.pc	-	-	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	-	-	IROM.inst[31:25|11:7]	RF.rD1	SEXT.ext	ALU.C	RF.rD2
beq	NPC.npc	PC.pc	SEXT.ext	ALU.f	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	-	-	IROM.inst[31|7|30:25|11:8]	RF.rD1	RF.rD2	-	-
lui	NPC.npc	PC.pc	-	-	PC.pc	-	-	IROM.inst[11:7]	SEXT.ext	IROM.inst[31:12]	-	-	-	-
jal	NPC.npc	PC.pc	SEXT.ext	-	PC.pc	-	-	IROM.inst[11:7]	NPC.pc4	IROM.inst[31|19:12|20|30:21]	-	-	-	-
综合	NPC.npc	PC.pc	SEXT.ext	ALU.f	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C SEXT.ext NPC.pc4	IROM.inst[31:7]	RF.rD1	RF.rD2 SEXT.ext	ALU.C	RF.rD2
操作选择	-	npc_op			-	-	-	rf_we		sext_op	alu_op		ram_we	
多路选择	-	-	-	-	-	-	-	-	rf_wsel	-	-	alub_sel	-	-

- 完成控制信号的取值表

指令	opcode	funct3	funct7	npc_op	rf_we	rf_wsel	sext_op	alu_op	alub_sel	ram_we
add	0110011	000	0000000	NPC_PC4	1	WB_ALU	-	ALU_ADD	ALUB_RS2	0
ori	0010011	110	-	NPC_PC4	1	WB_ALU	EXT_I	ALU_OR	ALUB_EXT	0
sw	0100011	010	-	NPC_PC4	0	-	EXT_S	ALU_ADD	ALUB_EXT	1
beq	1100011	000	-	NPC_BEQ	0	-	EXT_B	ALU_ADD	ALUB_RS2	0
lui	0110111	-	-	NPC_PC4	1	WB_EXT	EXT_U	-	-	0
jal	1101111	-	-	NPC_JMP	1	WB_PC4	EXT_J	-	-	0

谢 谢!

HITSZ 实验与创新实践教育中心

Education Center of Experiments and Innovations, HITSZ