计算机设计与实践

单周期CPU与SoC设计

2022·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 (2022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-1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

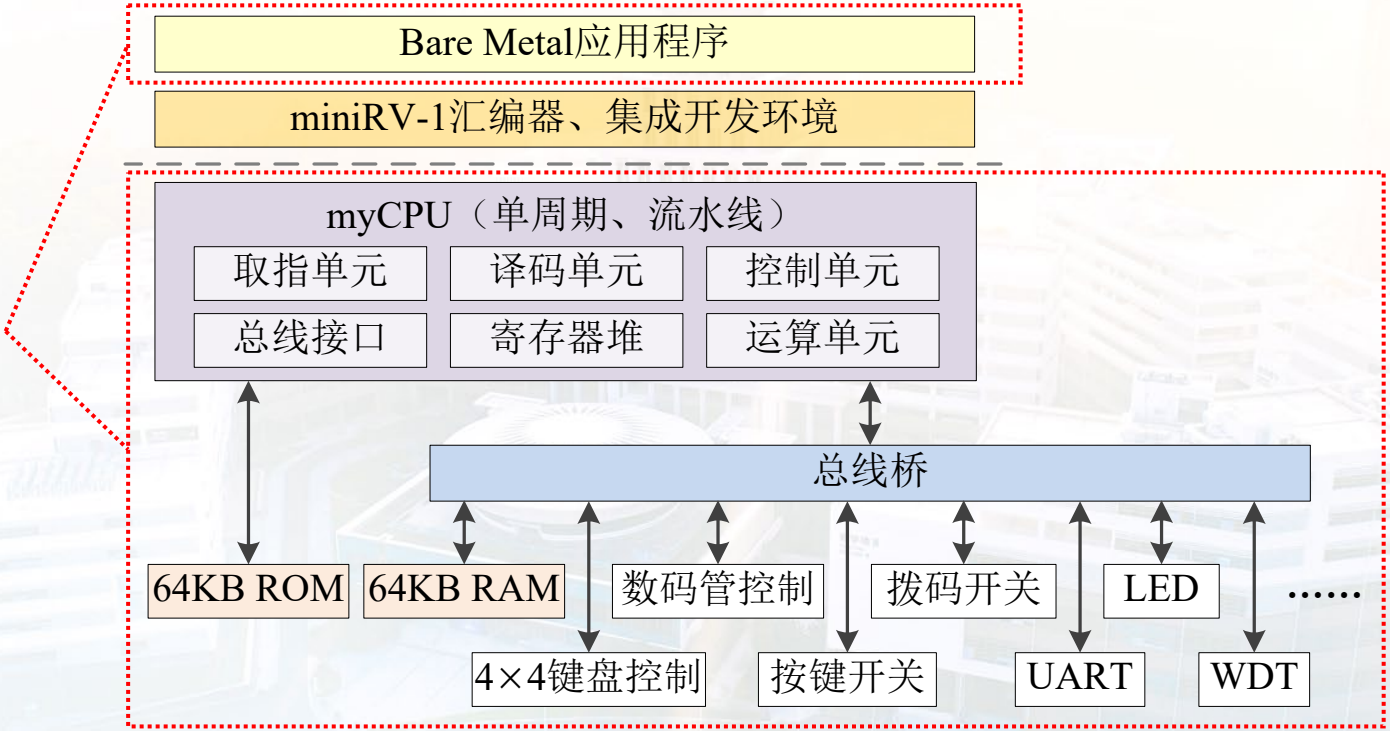
- 总共56学时 = 4学时理论 + 52学时实验
- 完成项目：支持miniRV-1指令集的SoC设计

	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	RISC-V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、答辩(面向优秀作品)	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-1指令集，编写汇编程序，并在Logisim上运行；
2. 使用Verilog HDL，设计**单周期**、**流水线**RISC-V 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(60%) + 作业(10%) + 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

平台及参考资料

- 课程实验指导书：https://hitsz-cslab.gitee.io/cpu/
- 答疑平台：piazza.com/hitsz/summer2022/comp2012, Access Code: comp2012
- 《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:
  https://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-1指令系统

- miniRV-1是32位整型RISC-V指令集（RV32I）的子集
  - 指令数：37条 (24 + *13*)
  - 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

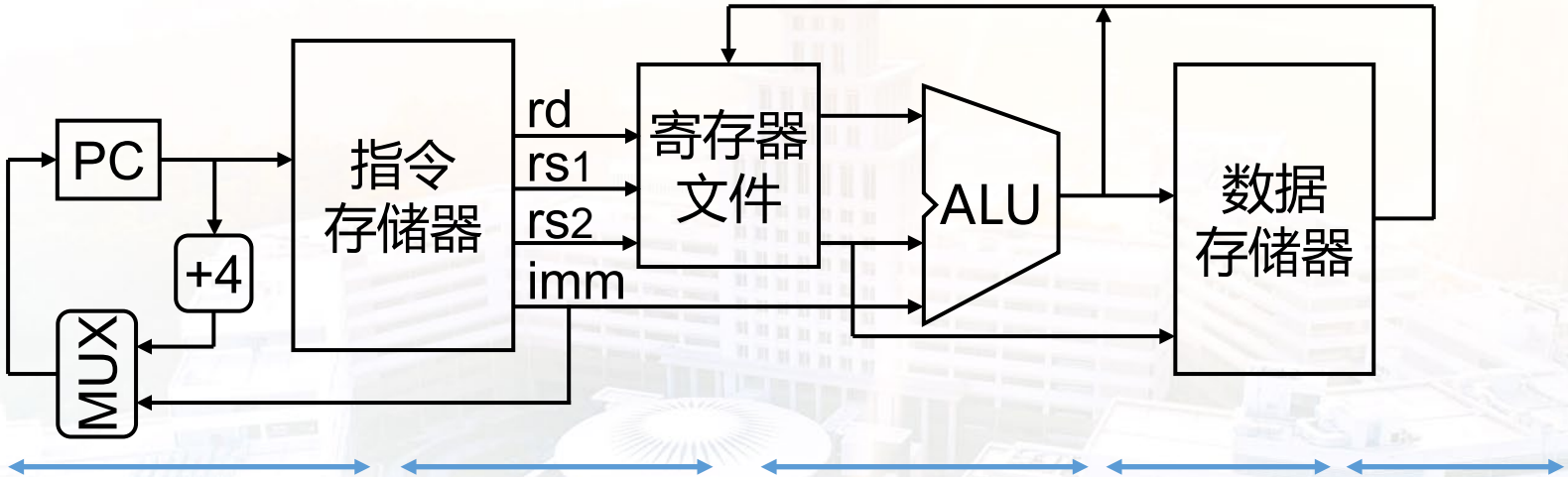
miniRV-1指令目录

- 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*

目录

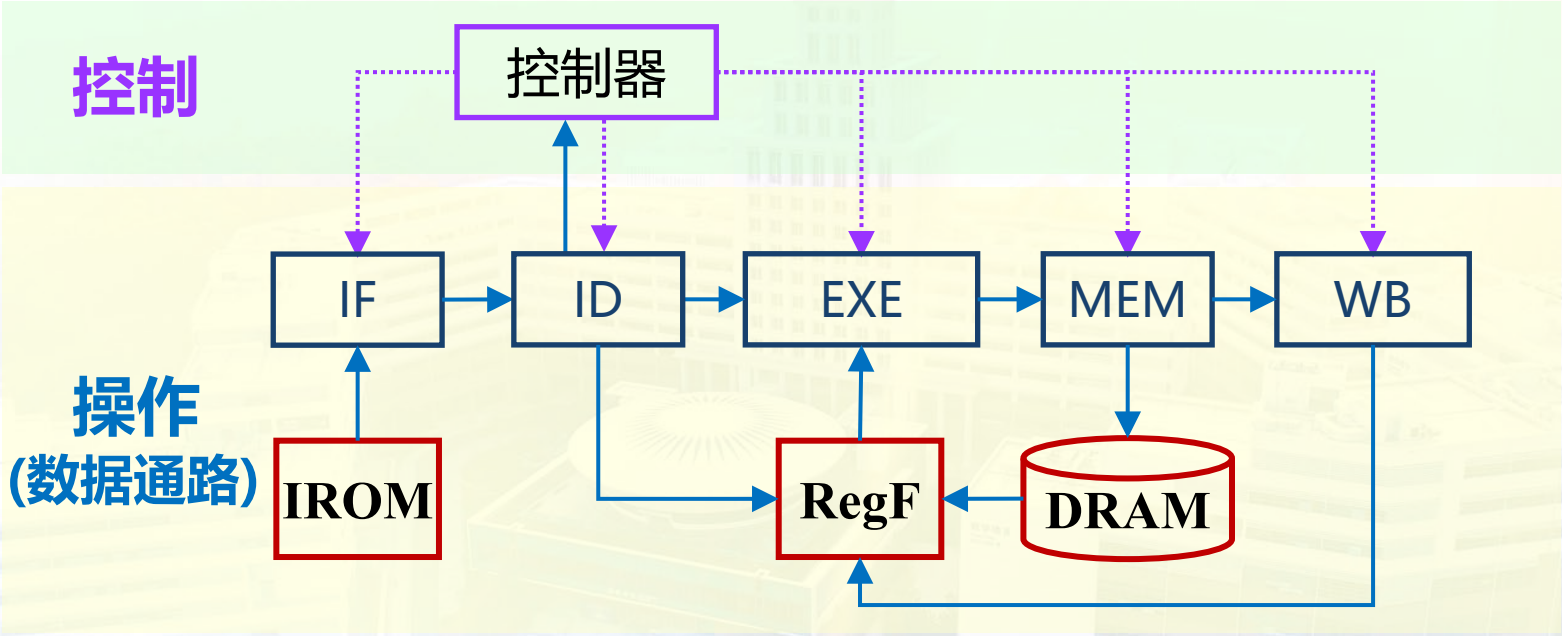
CPU执行指令的基本过程

- 从信息流的角度看指令执行过程



CPU执行指令的基本过程

- 指令执行过程需在合适的**控制**下才能有序进行



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

构造功能部件

构造一组必要的数据通路功能部件：对于每个功能部件，都需要定义其功能及接口信号，并用HDL描述其内部具体的电路行为或结构。

构造指令级别的数据通路与控制信号取值

根据指令操作语义，构造指令级别的数据通路与相关功能部件的控制信号取值。

综合数据通路

当所有指令级别的数据通路构造完毕后，将其综合为完整数据通路。

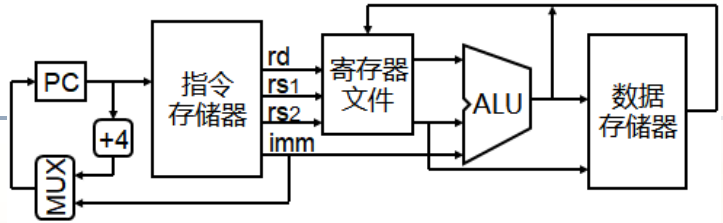
综合控制器

将全部指令对应的控制信号取值均建立后，合并所有的控制信号取值，形成控制信号矩阵，并生成相应的控制信号表达式。

生成工程项目

将第3、第4步的结果翻译为HDL语言，并在顶层工程文件中组装数据通路与控制器。

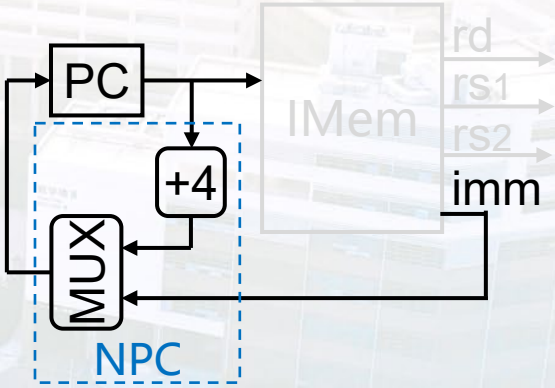
数据通路的主要功能部件



- ✓ **PC**（Program Counter）
- ✓ 指令存储器（Instruction Memory，**IROM**）
  - 只读：输入地址，输出指令
- ✓ 数据存储器（Data Memory，**DRAM**）
  - 读操作：输入地址，输出数据
  - 写操作：输入地址、数据
- ✓ 寄存器文件（Register File，**RF**）
  - 读操作：输入2个寄存器号，输出2个32bit的寄存器值
  - 写操作：输入1个寄存器号、待写入的32bit数据
- ✓ **ALU**（Arithmetic & Logic Unit）
  - 算术、逻辑、移位、比较运算：输入2个操作数，输出运算结果

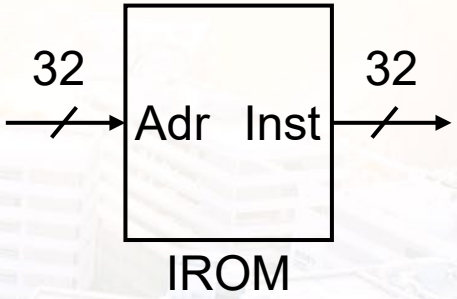
构造功能部件 —— PC设计

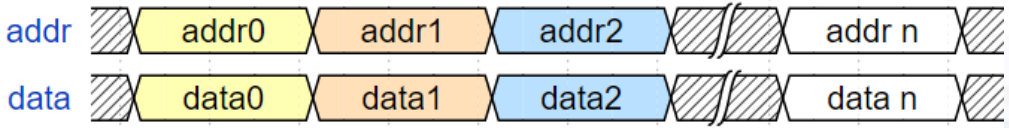
- PC是一个32bit寄存器，存储着当前指令的地址
  - 对于32位RISC CPU，指令均为32bit定长，即每条指令4Byte
  - 因此PC的BIT1和BIT0恒为0，故也可使用30bit寄存器
- CPU复位后PC的初始值即为首条指令的地址，如0x0000_0000H
- 执行时，需不停地更新PC的值
  - 默认情况下：PC <- PC + 4
  - 若遇到分支指令：PC <- imm



构造功能部件 —— IROM设计

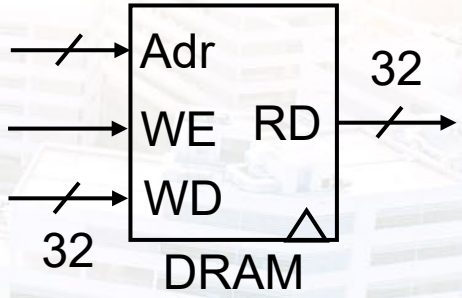
- IROM是一个单端口的只读存储器
  - 输入地址：PC
  - 输出数据：指令
- IROM的读取操作是组合逻辑
  - 读时序：





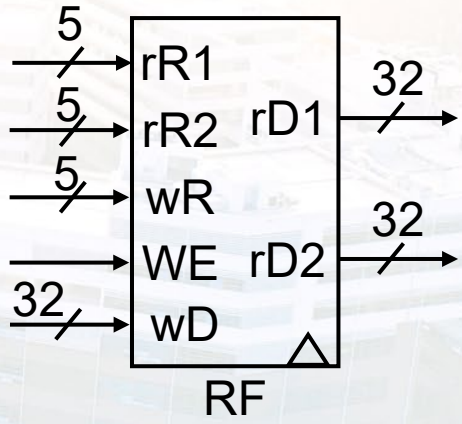
构造功能部件 —— DRAM设计

- DRAM是一个按地址访问的存储器，可读可写
  - 读操作：输入地址，输出数据
  - 写操作：输入地址、数据，受写使能信号WE控制
- DRAM的读写逻辑：
  - WE=0：RD <- DRAM[Adr]
  - WE=1：DRAM[Adr] <- WD



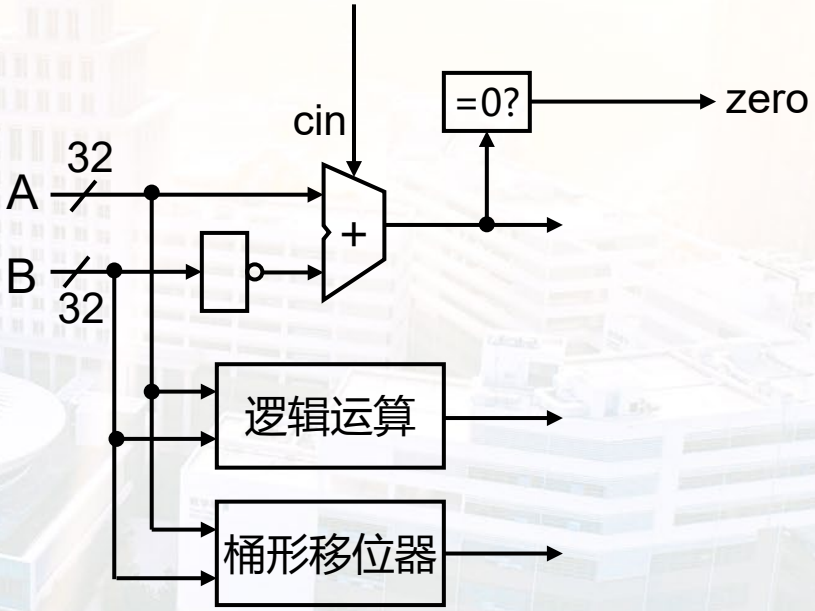
构造功能部件 —— RF设计

- RF包含32个32bit寄存器
  - 1条指令最多有2个源寄存器(rs1、rs2)、1个目标寄存器(rd)
  - 因此，RF需要3个“地址”端口、3个“数据”端口
  - 读操作：输入寄存器号，输出寄存器存储的值
  - 写操作：输入寄存器号、数据，受WE控制
- RF的读写逻辑：
  - WE=0: rD1<-REG[rR1]， rD2<-REG[rR2]
  - WE=1: REG[wR] <- wD
  - **e.g.** **add** *x2, x9, x5* *# rR1=9, rR2=5; wR=2*



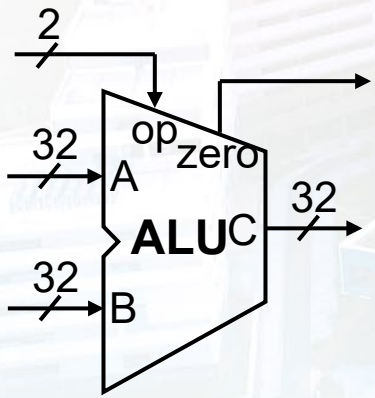
构造功能部件 —— ALU设计

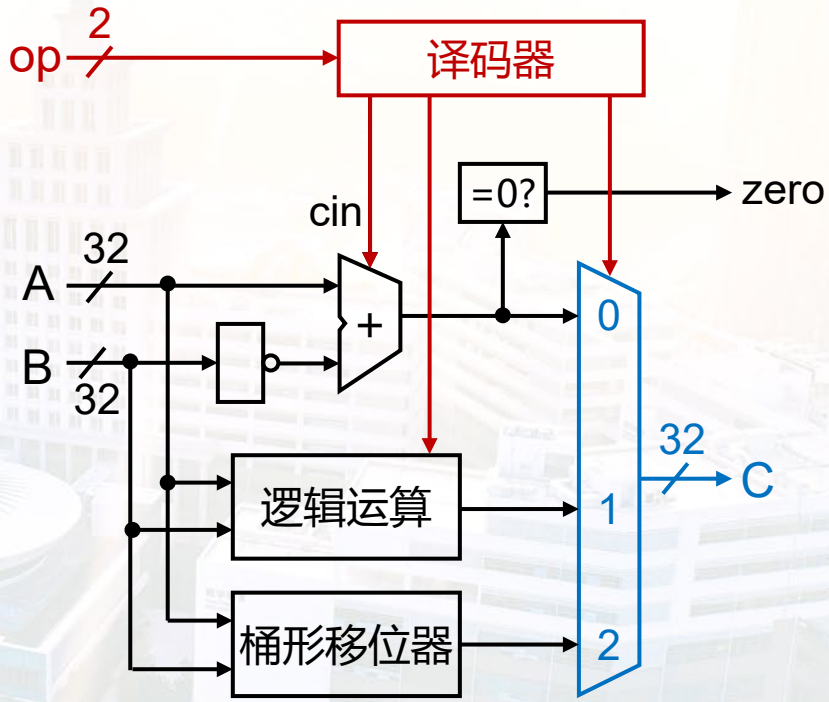
- ALU的功能：
  - 算术运算（add、sub）
    - ✓ 加法器、补码逻辑
  - 逻辑运算（and、or, etc）
    - ✓ 按位操作
  - 移位运算（sll、srl, etc）
    - ✓ 桶形移位器
  - 比较运算（slt、sltu, etc）
    - ✓ 复用sub，判断标志位



构造功能部件 —— ALU设计

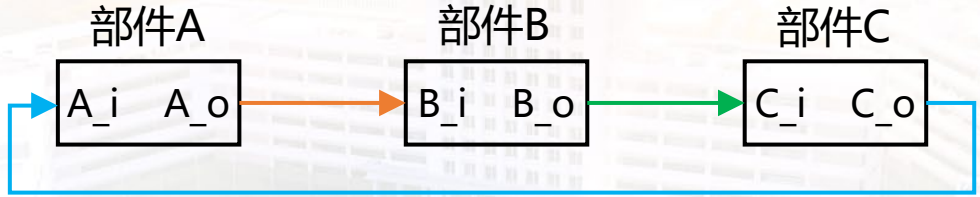
- 集成各项功能：
  - 多路选择器选择ALU输出
  - 译码器选择具体运算





构造数据通路

- 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
  - E.g. 3个部件的连接关系



  - 连接关系集合：{<C.C_o, A.A_i>, <A.A_o, B.B_i>, <B.B_o, C.C_i>}
  - 表格表示法：

部件A	部件B	部件C
A_i	B_i	C_i
C.C_o	A.A_o	B.B_o

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

构造数据通路

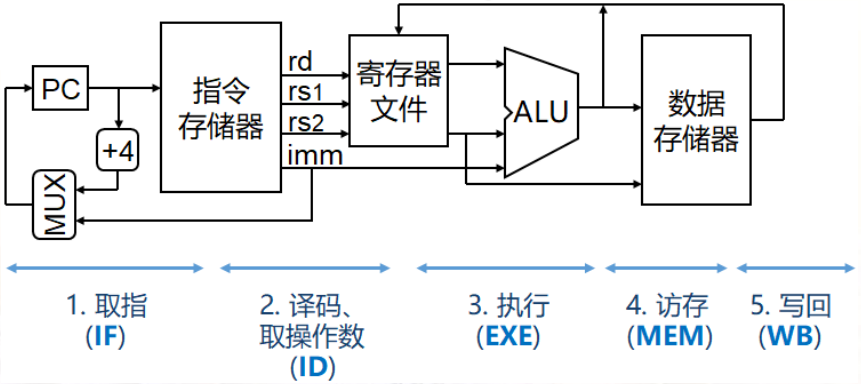
- 每条指令对应的部件和连接关系都不同
- 为了方便数据通路的综合，采用表格驱动设计的方法
  - E.g. 某数据通路包含PC、NPC、IROM、RF、ALU、DRAM的部件
  - 建立数据通路表：

所属单元	取指单元					译码单元					执行单元		存储单元	
部件	PC	NPC			IROM	RF				S_EXT	ALU		DRAM	
输入信号	din	PC	Imm	RA	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

构造数据通路

必要的功能部件

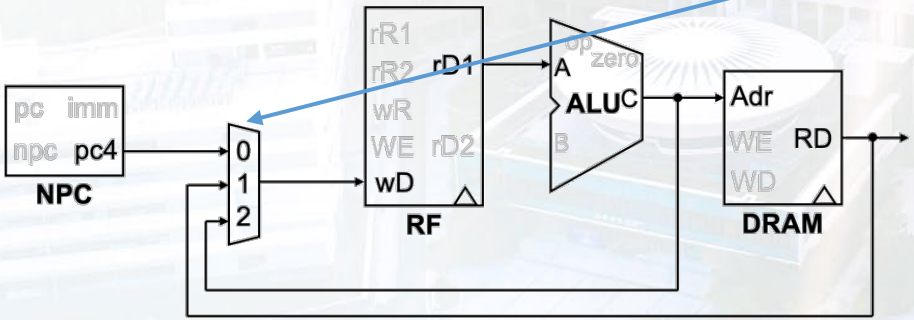


构造数据通路

所属单元	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	
输入信号	din	PC	imm	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		
sub	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		
lw	NPC.npc	PC.pc		PC.pc	IROM.inst[19:15]		IROM.inst[11:7]	DRAM.rd	IROM.inst[31:20]	RF.rD1	SEXT.ext	ALU.C	
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	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]			IROM.inst[31|7|30:25|11:8]	RF.rD1	RF.rD2		
jal	NPC.npc	PC.pc	SEXT.ext	PC.pc			IROM.inst[11:7]	NPC.pc4	IROM.inst[31|19:12|20|30:21]				

综合数据通路

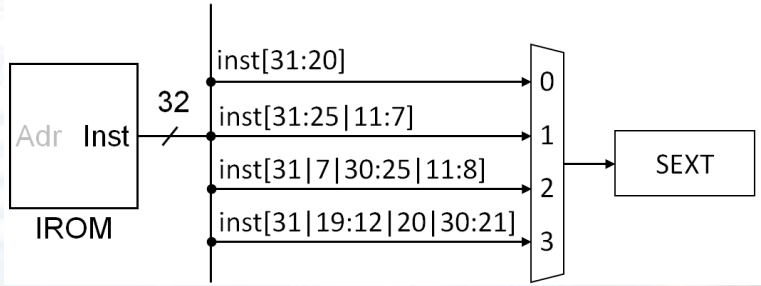
所属单元	取指单元				译码单元					执行单元		存储单元	
部件	PC	NPC		IROM	RF				SEXT	ALU		DRAM	
输入信号	din	PC	imm	adr	rR1	rR2	wR	wD	din	A	B	adr	wdin
add	NPC.npc	PC.pc		PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C		RF.rD1	RF.rD2		
sub	NPC.npc	PC.pc		PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C		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		
lw	NPC.npc	PC.pc		PC.pc	IROM.inst[19:15]		IROM.inst[11:7]	DRAM.rd	IROM.inst[31:20]	RF.rD1	SEXT.ext	ALU.C	
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	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]			IROM.inst[31|7|30:25|11:8]	RF.rD1	RF.rD2		
jal	NPC.npc	PC.pc	SEXT.ext	PC.pc			IROM.inst[11:7]	NPC.pc4	IROM.inst[31|19:12|20|30:21]				
完整	NPC.npc	PC.pc	SEXT.ext	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C DRAM.rd NPC.pc4	IROM.inst[31:7]	RF.rD1	RF.rD2 SEXT.ext	ALU.C	RF.rD2



综合方法1：使用多路选择器

综合数据通路

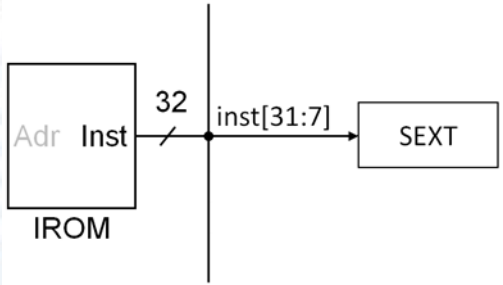
所属单元	取指单元				译码单元					执行单元		存储单元	
部件	PC	NPC		IROM	RF				SEXT	ALU		DRAM	
输入信号	din	PC	imm	adr	rR1	rR2	wR	wD	din	A	B	adr	wdin
add	NPC.npc	PC.pc		PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C		RF.rD1	RF.rD2		
sub	NPC.npc	PC.pc		PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C		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		
lw	NPC.npc	PC.pc		PC.pc	IROM.inst[19:15]		IROM.inst[11:7]	DRAM.rd	IROM.inst[31:20]	RF.rD1	SEXT.ext	ALU.C	
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	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]			IROM.inst[31|7|30:25|11:8]	RF.rD1	RF.rD2		
jal	NPC.npc	PC.pc	SEXT.ext	PC.pc			IROM.inst[11:7]	NPC.pc4	IROM.inst[31|19:12|20|30:21]				
完整	NPC.npc	PC.pc	SEXT.ext	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C DRAM.rd NPC.pc4	IROM.inst[31:7]	RF.rD1	RF.rD2 SEXT.ext	ALU.C	RF.rD2



综合方法2：增加输入信号

综合数据通路

所属单元	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	
输入信号	din	PC	imm	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		
sub	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		
lw	NPC.npc	PC.pc		PC.pc	IROM.inst[19:15]		IROM.inst[11:7]	DRAM.rd	IROM.inst[31:20]	RF.rD1	SEXT.ext	ALU.C	
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	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]			IROM.inst[31|7|30:25|11:8]	RF.rD1	RF.rD2		
jal	NPC.npc	PC.pc	SEXT.ext	PC.pc			IROM.inst[11:7]	NPC.pc4	IROM.inst[31|19:12|20|30:21]				
完整	NPC.npc	PC.pc	SEXT.ext	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C DRAM.rd NPC.pc4	IROM.inst[31:7]	RF.rD1	RF.rD2 SEXT.ext	ALU.C	RF.rD2



综合方法2：增加输入信号

构造控制单元

- 控制单元：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、存储器等)
  - 多路选择信号　对多输入源进行选择
    - 多路选择器
  - 分支控制信号　控制分支指令是否跳转
    - ALU

添加控制信号

- 在数据通路表中新增控制信号：

所属单元	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	
输入信号	din	PC	imm	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		
sub	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		
lw	NPC.npc	PC.pc		PC.pc	IROM.inst[19:15]		IROM.inst[11:7]	DRAM.rd	IROM.inst[31:20]	RF.rD1	SEXT.ext	ALU.C	
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	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]			IROM.inst[31|7|30:25|11:8]	RF.rD1	RF.rD2		
jal	NPC.npc	PC.pc	SEXT.ext	PC.pc			IROM.inst[11:7]	NPC.pc4	IROM.inst[31|19:12|20|30:21]				
完整	NPC.npc	PC.pc	SEXT.ext	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C DRAM.rd NPC.pc4	IROM.inst[31:7]	RF.rD1	RF.rD2 SEXT.ext	ALU.C	RF.rD2
操作选择信号	-	npc_op		-	-	-	rf_we		sext_op	alu_op		-	dram_we
多路选择信号	-	-		-	-	-	-	wd_sel	-	-	alub_sel	-	-
分支控制信号	-	-		-	-				-	branch		-	

确定控制信号取值

- 建立控制信号取值表：

指令	npc_op	rf_we	wd_sel	sext_op	alu_op	alub_sel	branch	dram_we

- 逐条指令分析，确定其控制信号取值：

指令	npc_op	rf_we	wd_sel	sext_op	alu_op	alub_sel	branch	dram_we
add	pc+4	1	'b00	X	ADD	0	0	0
sub	pc+4	1	'b00	X	SUB	0	0	0
ori	pc+4	1	'b00	'b00	OR	1	0	0
lw	pc+4	1	'b01	'b00	ADD	1	0	0
sw	pc+4	0	'bXX	'b01	ADD	1	0	1
beq	'b01	0	'bXX	'b10	SUB	0	1	0
jal	'b10	1	'b10	'b11	X	X	0	0

综合控制信号

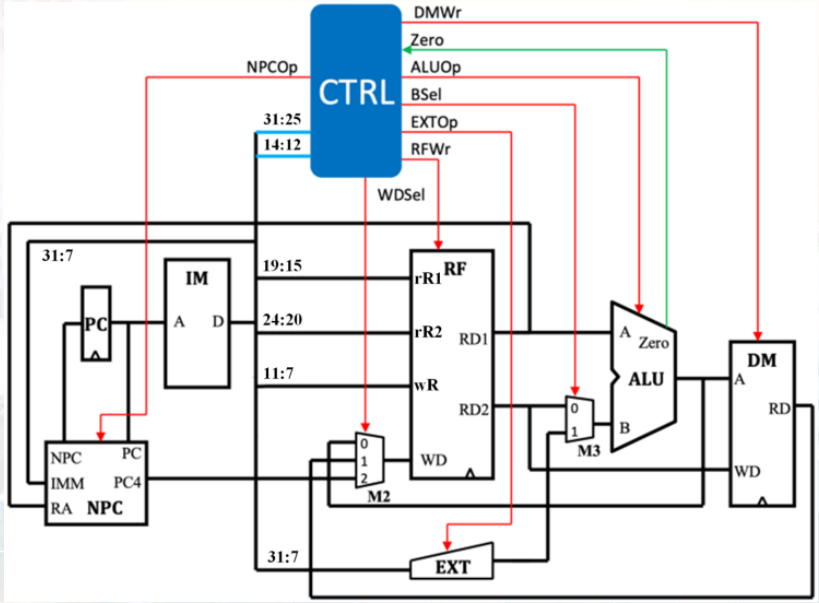
- 根据控制信号取值表，化简控制信号的逻辑表达式
- 根据表达式，设计控制单元
- 在数据通路图中添加控制单元

输入：opcode、funct3/funct7

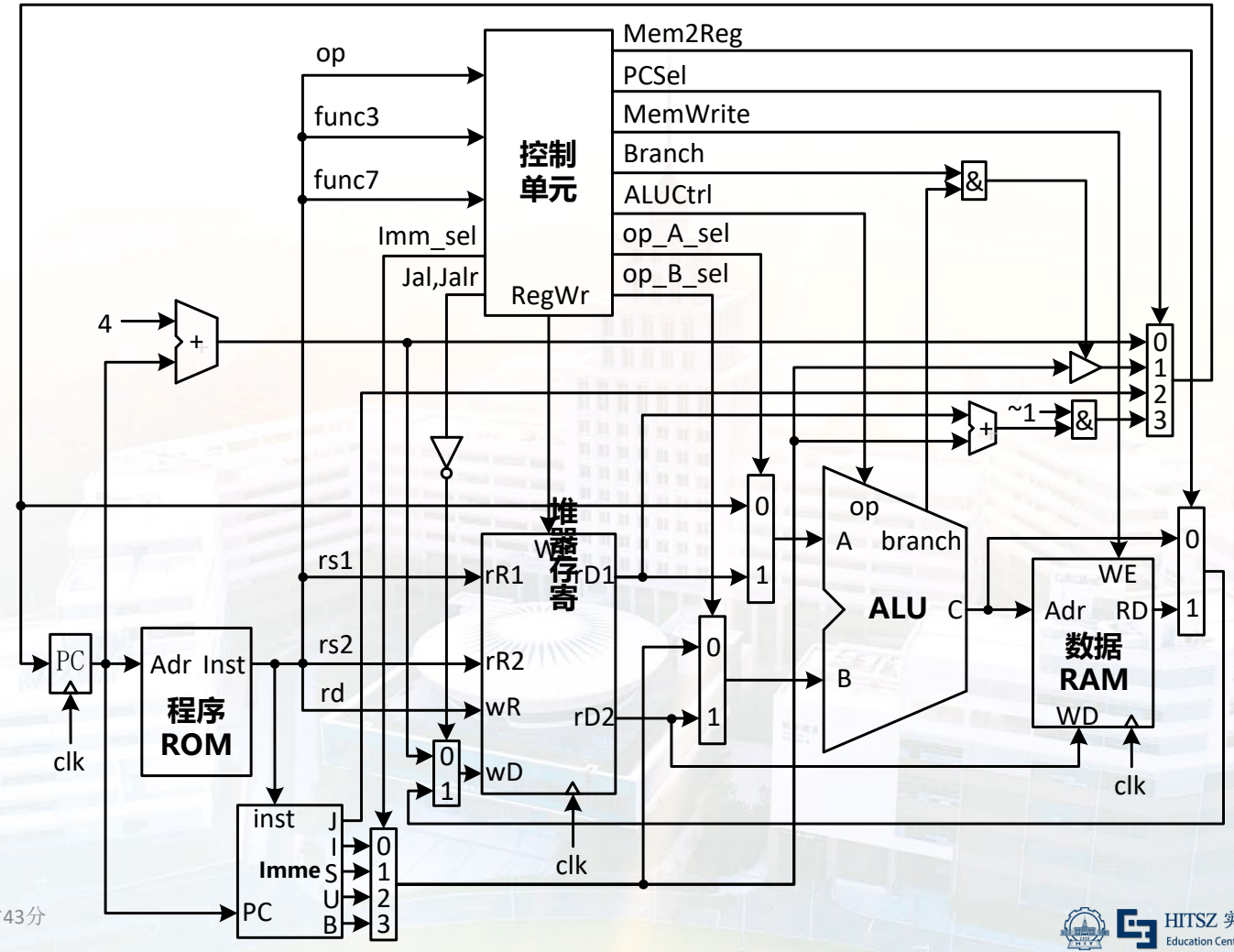
分支控制信号

输出：操作选择信号

多路选择信号



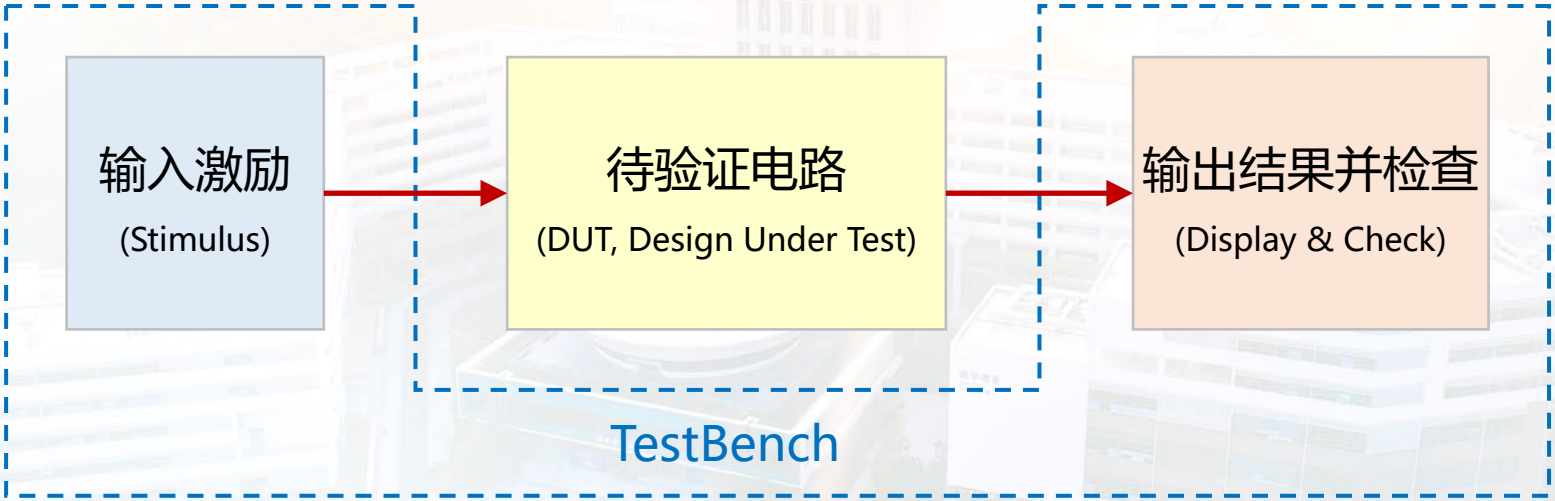
完整数据通路(示例)



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

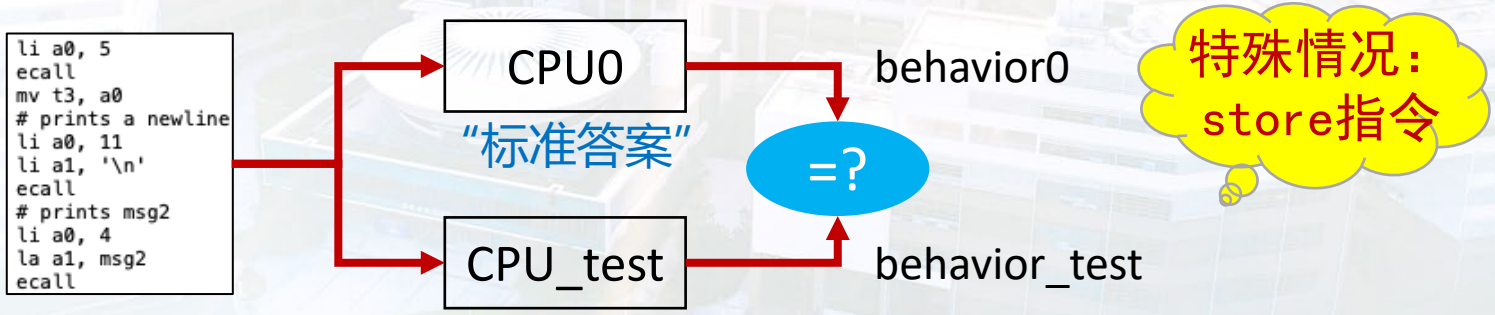


CPU的功能验证

- 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

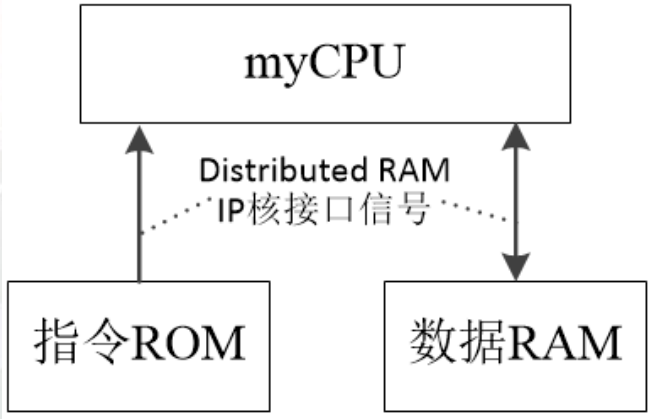
- Trace：CPU执行指令序列时产生的信息 (PC、写寄存器的信息，etc)
- 基于Trace比对的验证方法：
  - ① 用已知功能正确的CPU运行测试程序，记录Trace0 (Golden Trace)
  - ② 用待验证CPU运行测试程序，产生Trace1
  - ③ 将Trace1与Trace0进行实时比对，如果出现不同，立即报错并停止



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

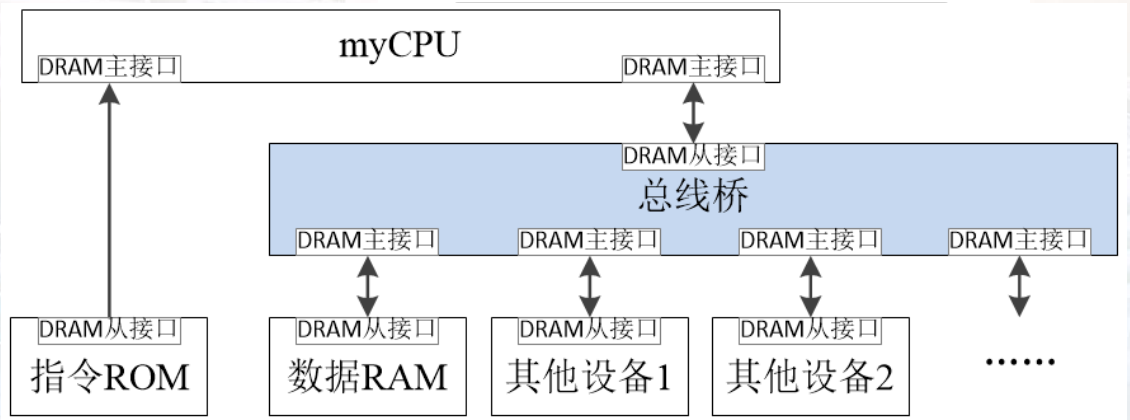
- CPU与存储器通过DRAM总线接口一对一连接
  - 如何连接其他设备?
  - 增加**总线桥**

中转机构:

访问请求转发

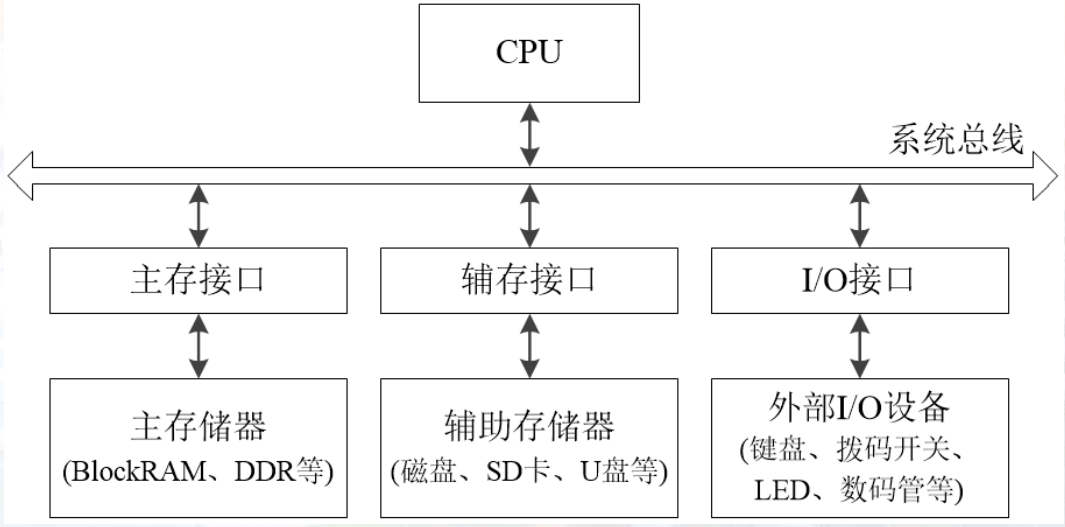
控制机构:

总线仲裁



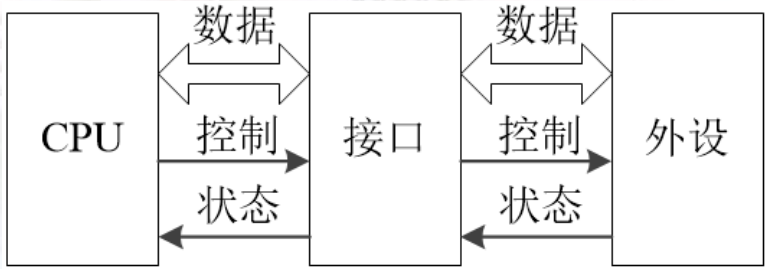
SoC设计 —— 接口

- 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

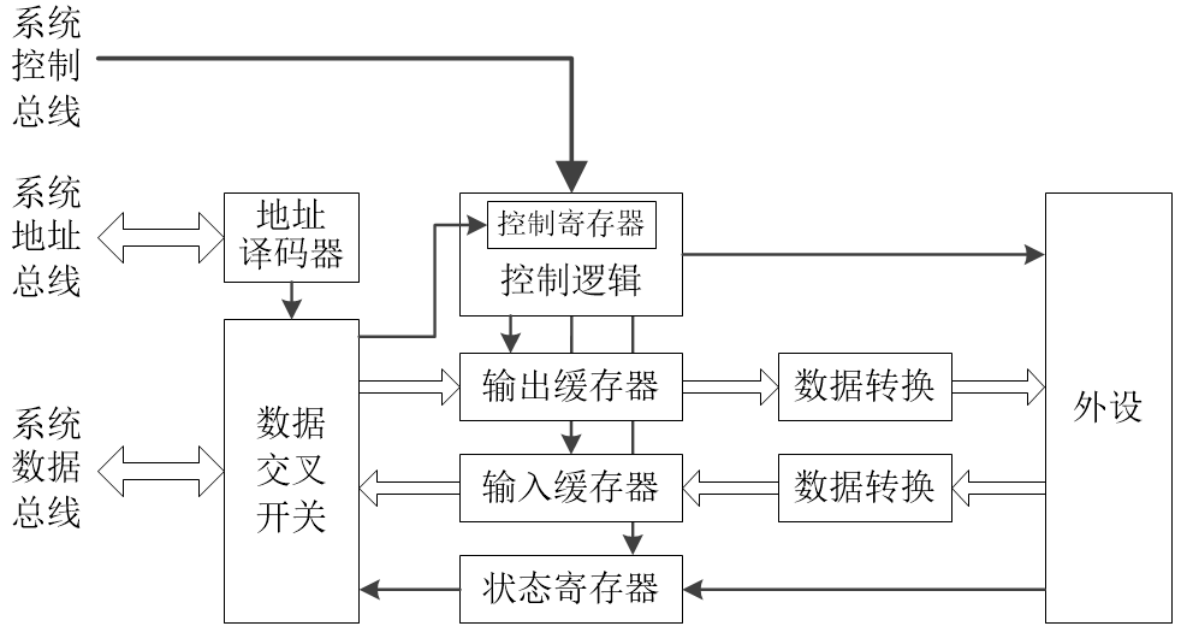


SoC设计 —— 接口的作用与功能

- 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

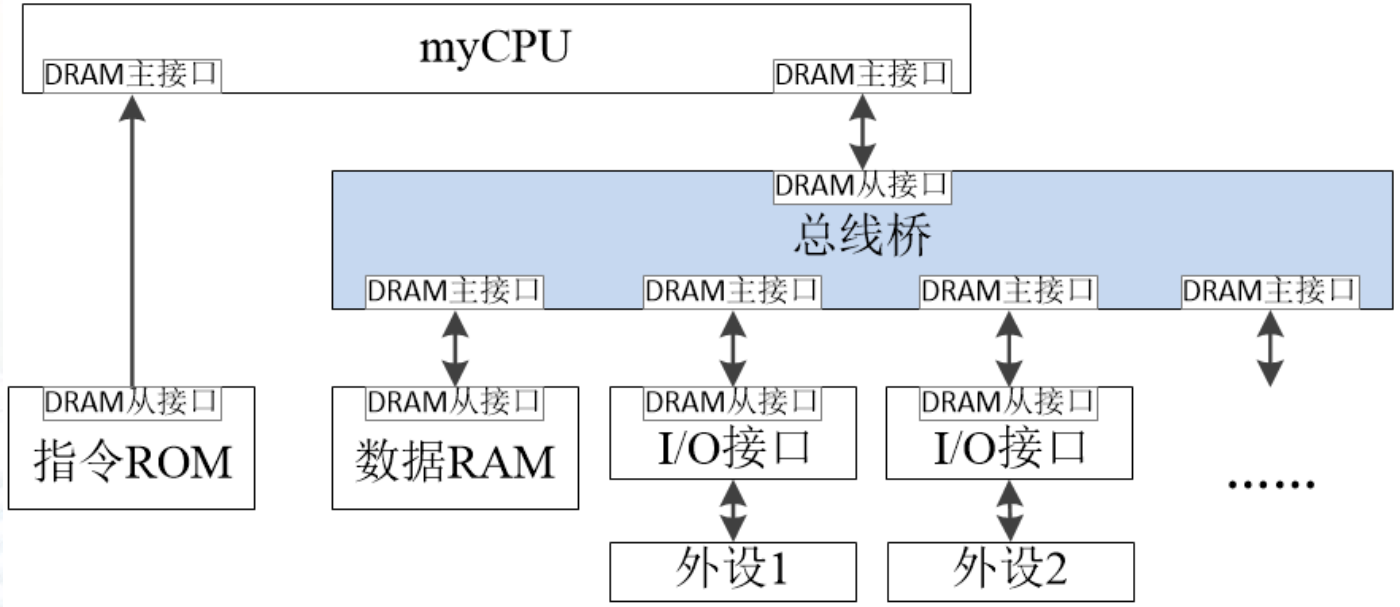
SoC设计 —— 接口的设计

- 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: 6月26日 23:59)

- 完成数据通路和控制信号的构造、综合

所属单元	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	
输入信号	din	PC	imm	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		
sub	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		
lw	NPC.npc	PC.pc		PC.pc	IROM.inst[19:15]		IROM.inst[11:7]	DRAM.rd	IROM.inst[31:20]	RF.rD1	SEXT.ext	ALU.C	
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	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]			IROM.inst[31|7|30:25|11:8]	RF.rD1	RF.rD2		
jal	NPC.npc	PC.pc	SEXT.ext	PC.pc			IROM.inst[11:7]	NPC.pc4	IROM.inst[31|19:12|20|30:21]				
完整	NPC.npc	PC.pc	SEXT.ext	PC.pc	IROM.inst[19:15]	IROM.inst[24:20]	IROM.inst[11:7]	ALU.C DRAM.rd NPC.pc4	IROM.inst[31:7]	RF.rD1	RF.rD2 SEXT.ext	ALU.C	RF.rD2
操作选择信号	-	npc_op		-	-	-	rf_we		sext_op	alu_op		-	dram_we
多路选择信号	-	-		-	-	-	-	wd_sel	-	-	alub_sel	-	-
分支控制信号	-	-		-	-				-	branch		-	

- 完成控制信号的取值表

指令	npc_op	rf_we	wd_sel	sext_op	alu_op	alub_sel	branch	dram_we
add	pc+4	1	'b00	X	ADD	0	0	0
sub	pc+4	1	'b00	X	SUB	0	0	0
ori	pc+4	1	'b00	'b00	OR	1	0	0
lw	pc+4	1	'b01	'b00	ADD	1	0	0
sw	pc+4	0	'bXX	'b01	ADD	1	0	1
beq	'b01	0	'bXX	'b10	SUB	0	1	0
jal	'b10	1	'b10	'b11	X	X	0	0

谢 谢！

HITSZ 实验与创新实践教育中心

Education Center of Experiments and Innovations, HITSZ