课程总结

2021年秋

学习目的

- 了解计算机的组成
  - 五大组成部件
- 掌握计算机的运行原理
  - 计算机为什么能执行高级语言程序
- 了解现代计算机中的一些核心技术
  - 流水、Cache、虚拟存储
- 提高编程能力
- 培养设计计算机的技能
- 成为计算机科学家、计算机专家

学习目标

- 掌握单CPU计算机的完整硬件组成
  - 基本工作原理
  - 内部运行机制
  - 建立完整计算机系统概念
- 了解计算机系统的新发展
- 达到能独立设计一台完整计算机的水平
  - 硬件、软件齐全
  - 功能基本完整
- 知识和能力两方面都提高

教学思路

- 程序是如何在硬件上运行的?
- 指令在计算机内的表示?
- 指令的核心功能是什么?
- 如何完成对数据的加工?
- 如何完成指令执行流程的控制?
- 数据存储的需求是什么?
- 如何实现数据存储和高效访问?
- 数据如何入/出计算机?

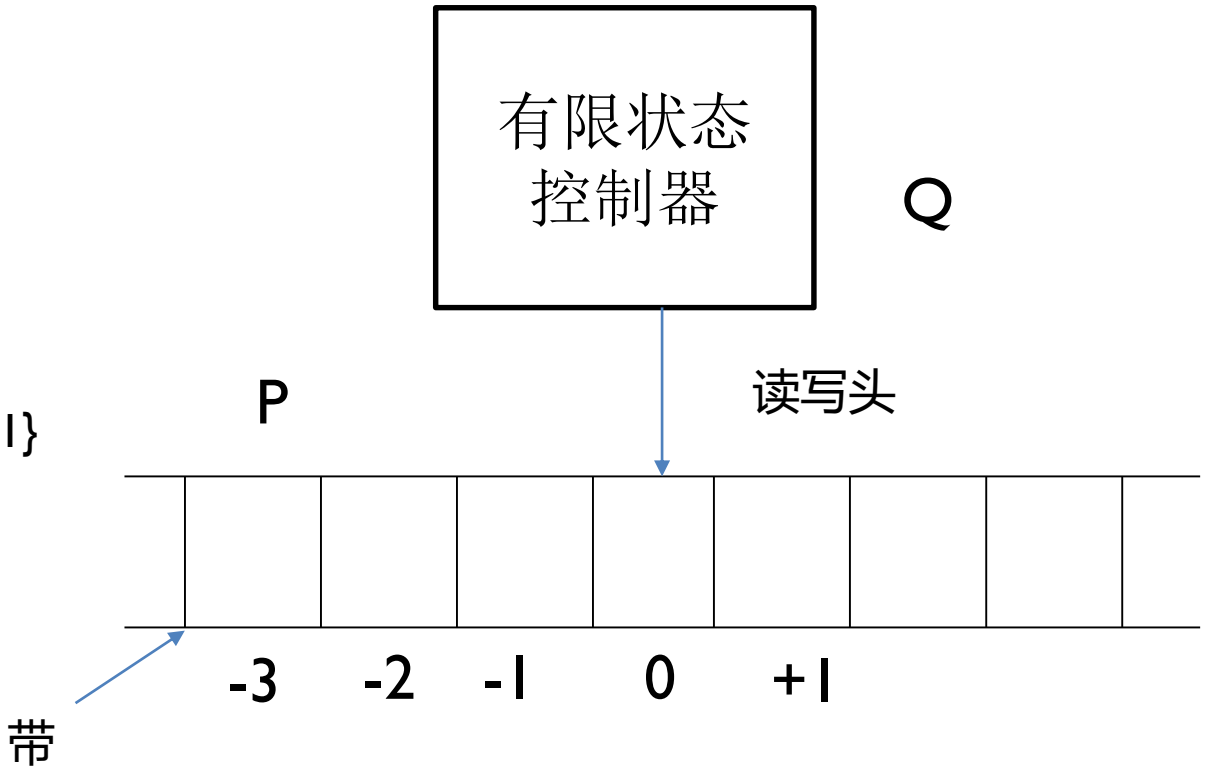
信息时代

- 信息技术(Information Technology)
- * 计算技术(Computation) —计算机
- * 通信技术(Communication) —通信机
- * 网络技术——
- 计算机网络= 计算+ 通信
- (Network= Computation + Communication)

图灵机（Turing Machine）

- 确定型图灵机(1932)
- * 有穷符号集P
- * 有穷状态集Q
- * 转移函数

F: Q ∘ P ---- Q ∘ P ∘ {-1, +1}



通用机(Universal Machine)概念

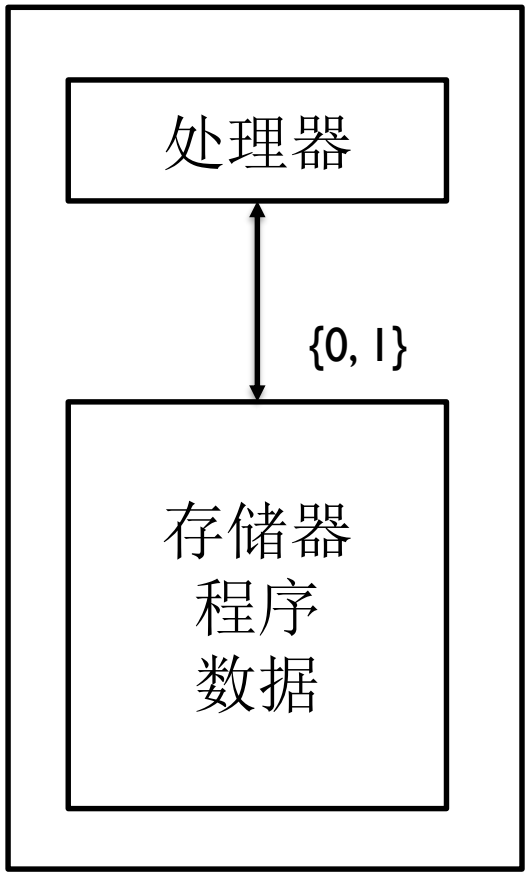
图灵可计算

- •“computable numbers”
- 图灵可计算—
- 在有限机械步中可完成的计算
- •“computational complexity”
-

- 计算复杂性：指数（与问题规模的关系）
- 多项式

计算机的诞生

- 1943 电子器件的应用（电子管）
- 1946 ENIAC 电子计算机
- 1949 EDSAC 存储程序计算机
- 1953 IBM 701（电子管）
- 1960 晶体管计算机
- 1965 IBM 360（集成电路）
- * 冯诺曼(von Neumann)结构
- 按地址存储数据和程序的串行计算结构



计算机（理论上）能做什么?

- 对{0，1} 进行以下基本运算:
- +：1+1=0*，1+0=1（0+1=1），0+0=0
- -：1-1=0，1-0=1，0-1=1*，0-0=0
- &：1 &1=1，1 & 0=0（0 & 1=0），0 & 0=0
- |：1 |1=1，1 | 0=1（0 | 1=1），0 | 0=0
- -：-1=0，-0=1
- 如此简单的运算能解决复杂的问题?

计算机能做什么

- 文字处理、科学计算、通讯，社会生活的方方面面
- 只能执行指令系统中的指令，完成规定的功能
- 只能完成二进制算术加法运算和逻辑运算
- 只能完成逻辑运算

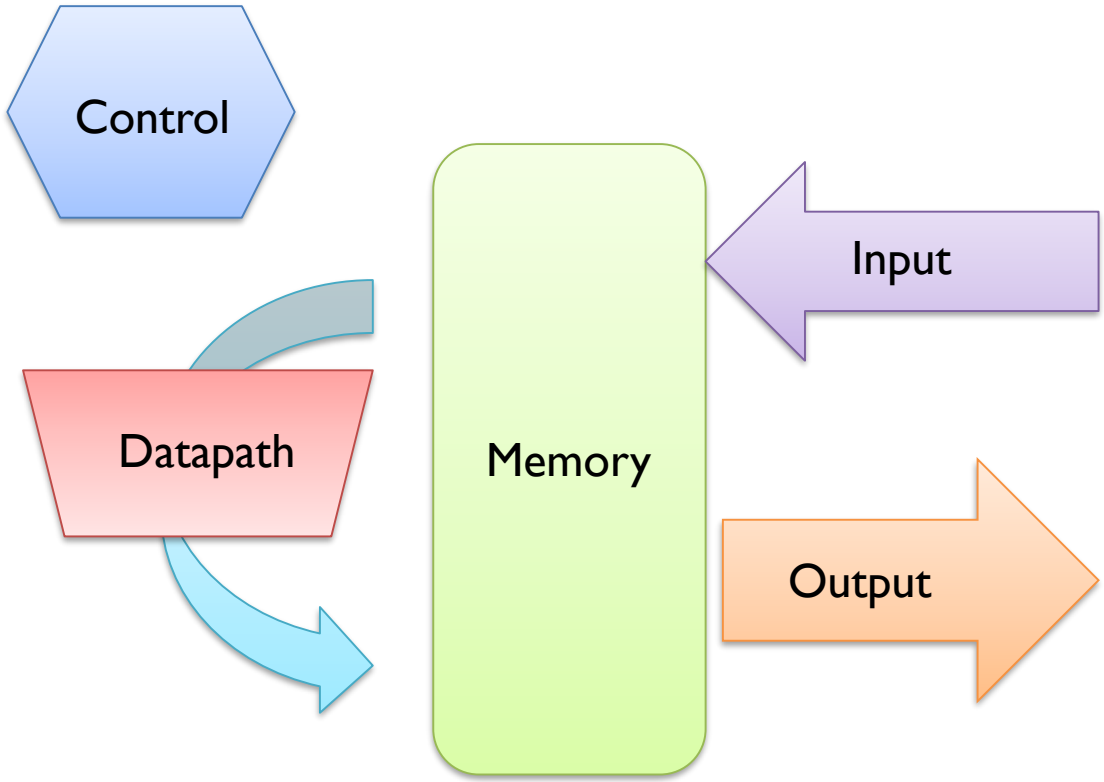
计算机为什么能完成这些工作

- 将任务分解成算逻运算的组合
  - 程序设计和算法
  - 好处：固化操作，能快速复制
  - 缺点：并不是所有任务都能分解成算逻运算的组合
- 自动快速执行算逻运算
  - 比人脑运算要快
  - 不足：受时间、空间限制
- 接收指令，并输出结果
  - 存储程序
  - 输入输出系统和设备

怎么完成

- ALU
  - 完成算术逻辑运算
- 存储器
  - 存储程序和数据
- 输入设备
  - 人机交互
- 输出设备
  - 人机交互
- 总线
  - 各部件之间连接和数据交换
- 控制器
  - 自动、连续完成

什么是计算机?



- Datapath: 完成算术和逻辑运算，通常包括其中的寄存器。
- Control: CPU的组成部分，它根据程序指令来指挥datapath，memory以及I/O运行，共同完成程序功能。
- Memory: 存放运行时程序及其所需要的数据的场所。
- Input: 信息进入计算机的设备，如键盘、鼠标等。
- Output: 将计算结果展示给用户的设备，如显示器、磁盘、打印机、喇叭等。

运算器

- 数据表示
  - 数值数据表示
  - 逻辑数据表示
  - 字符数据表示
  - 检错纠错码
- 算术运算
  - 加法运算
  - 减法运算
  - 乘法运算
  - 除法运算
  - 浮点数运算
- 逻辑运算
  - 逻辑与、或、非
- 电路实现

数据表示

- 二进制数据表示
  - 数值数据和逻辑数据统一
  - 字符和数值统一
  - 指令和数据统一
  - 最容易实现
- 补码数据表示
  - 减法和加法统一
  - 乘法和加法统一（加法、移位）
  - 除法和加法统一（加、减和移位）
- 浮点数表示
  - IEEE754标准
  - 规格化数、非规格化数、表示范围
- 检错纠错码
  - 奇偶校验
  - 海明校验

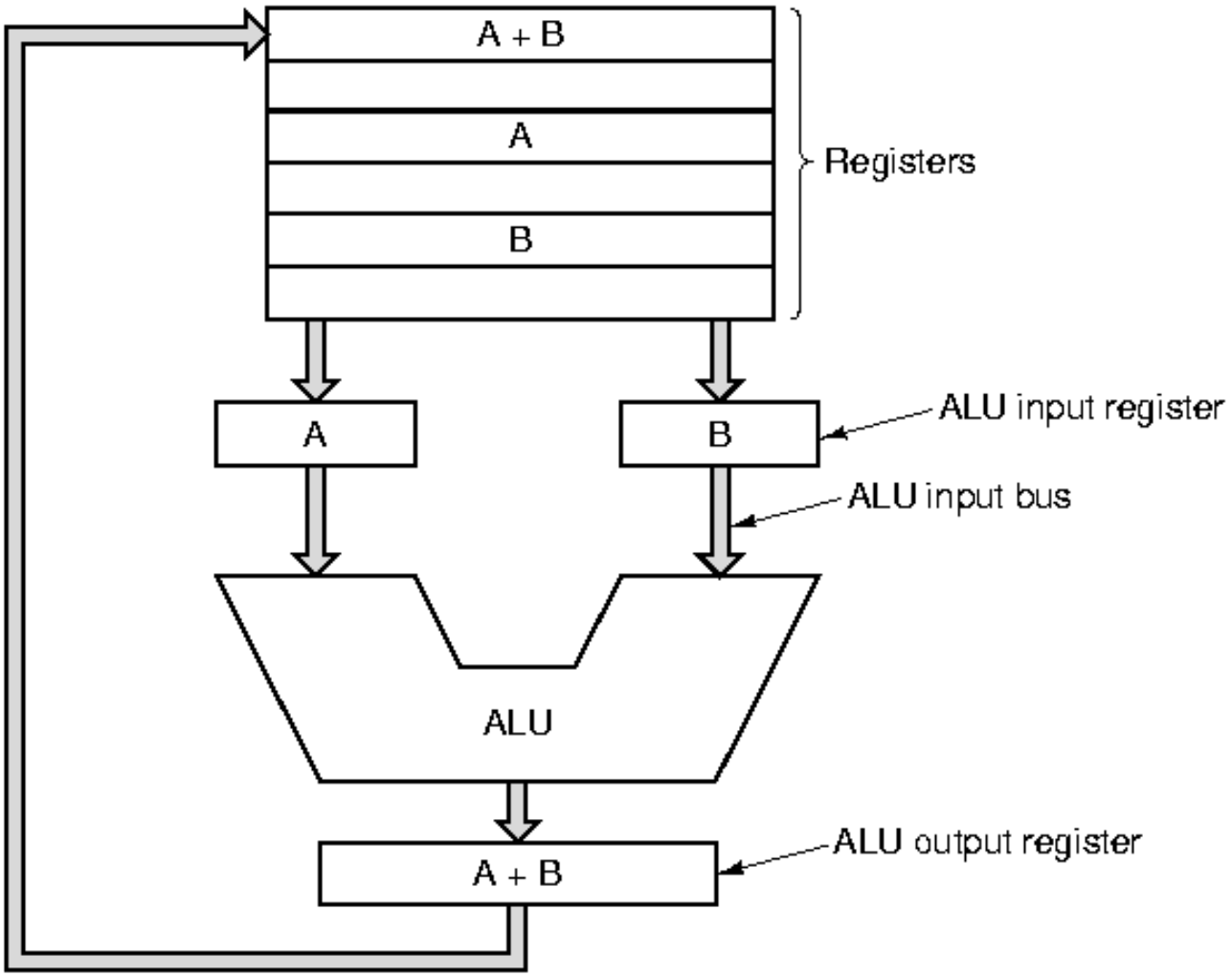
算数运算和逻辑运算

- 完成算术运算
  - 加、减、乘、除
- 给出运算结果
- 给出结果状态
  - C、Z、V、S
- 浮点数据的算术运算
- 根据标志位进行逻辑判断
- 指令中的逻辑判断

电路实现

- ALU
- 移位器
- 寄存器组
- Q寄存器
- 多路选通电路
- 译码器

Datapath


A + B
A
B
Registers
A
B
ALU input register
ALU input bus
ALU
A + B
ALU output register

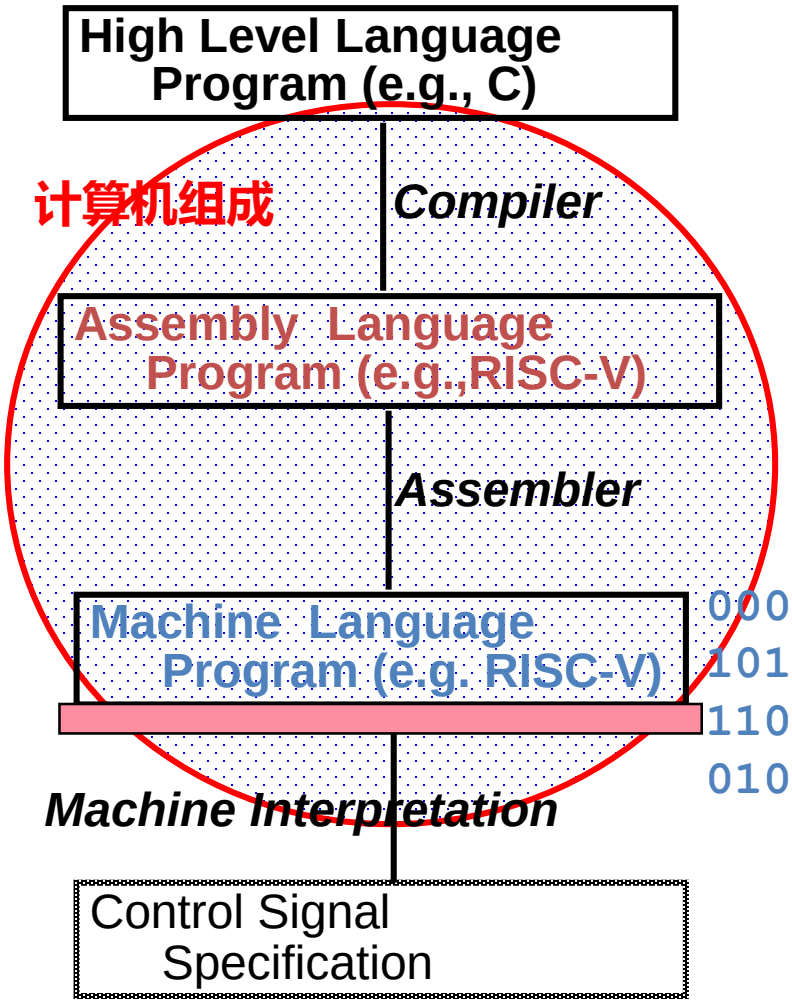
运算器

- 算术逻辑运算
- 数据表示
  - 原、反、补码
  - 检错纠错码
  - 浮点数据表示IEEE754
- 数据运算
  - 补码加、减运算
  - 原码一位乘除运算
  - 浮点数算术运算
- 电路实现

控制器

- 自动执行指令
  - 将指令系统的指令转换为完成指令功能对应的控制信号
  - 分步骤执行指令
  - 得到下一条指令的地址
- 连续执行指令
  - 下地址
  - 节拍
  - 段间寄存器
- 提高指令执行速度
  - 指令流水
  - 多流水线设置
  - 多核
  - 并行计算机

不同层次的程序




```
temp = v[k];
v[k] = v[k+1];
v[k+1] = temp;
```

```
lw t0, t2, 0
lw t1, t2, 4
sw t1, t2, 0
sw t0, t2, 4
```

```
0000 1001 1100 0110 1010 1111 0101 1000
1010 1111 0101 1000 0000 1001 1100 0110
1100 0110 1010 1111 0101 1000 0000 1001
0101 1000 0000 1001 1100 0110 1010 1111
```

指令系统和指令格式

- 指令和指令系统
  - 指令是指挥计算机各部件完成规定功能的命令
  - 计算机系统的全部指令的集合称为指令系统
- 操作码
  - 指明指令需要完成的功能
  - 对指令进行译码的输入
- 操作数地址
  - 指明指令处理的对象
- 指令格式
  - 如何在指令字中安排操作码和操作数地址

寻址方式

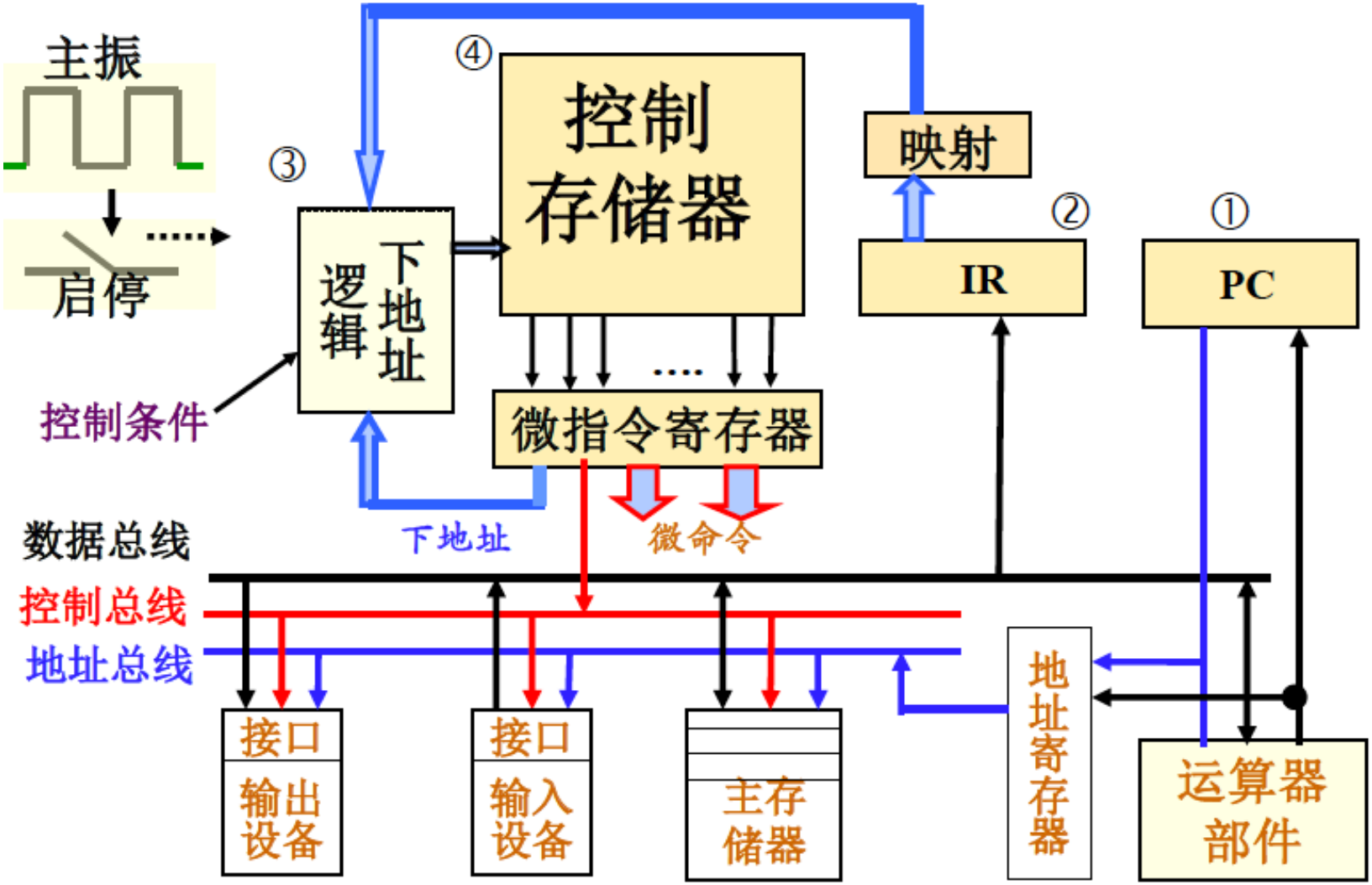
- 立即数寻址
  - 常量
- 寄存器寻址
- 直接寻址
- 间接寻址
- 变址寻址
- 堆栈寻址
- 基地址寻址
  - 段表

多周期控制器组成

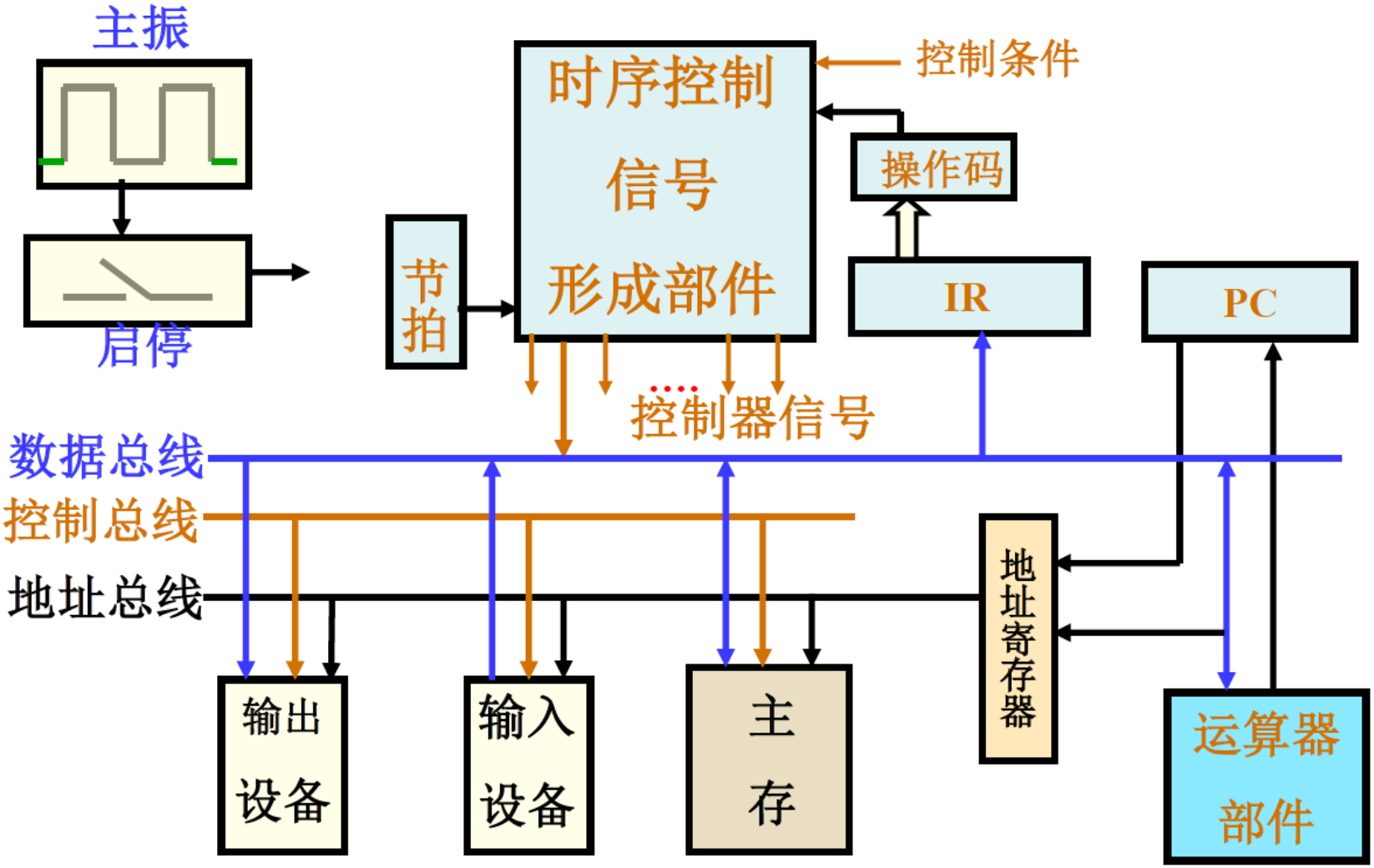
- ①程序计数器**PC**
  - 存放指令地址，有增量或接收新值功能
- ②指令寄存器**IR**
  - 存放指令内容：操作码与操作数地址
- ③指令执行步骤标记线路
  - 指明每条指令的执行步骤和相对次序关系
- ④控制信号记忆或产生线路
  - 给出计算机各功能部件协同运行所需要的控制信号

- 各部件包括：运算器部件　主存储器部件
- 总线及输入/输出接口（输入/输出设备）
- 也包括：控制器部件
- 设计中的难点，在于解决对运算器、控制器的控制

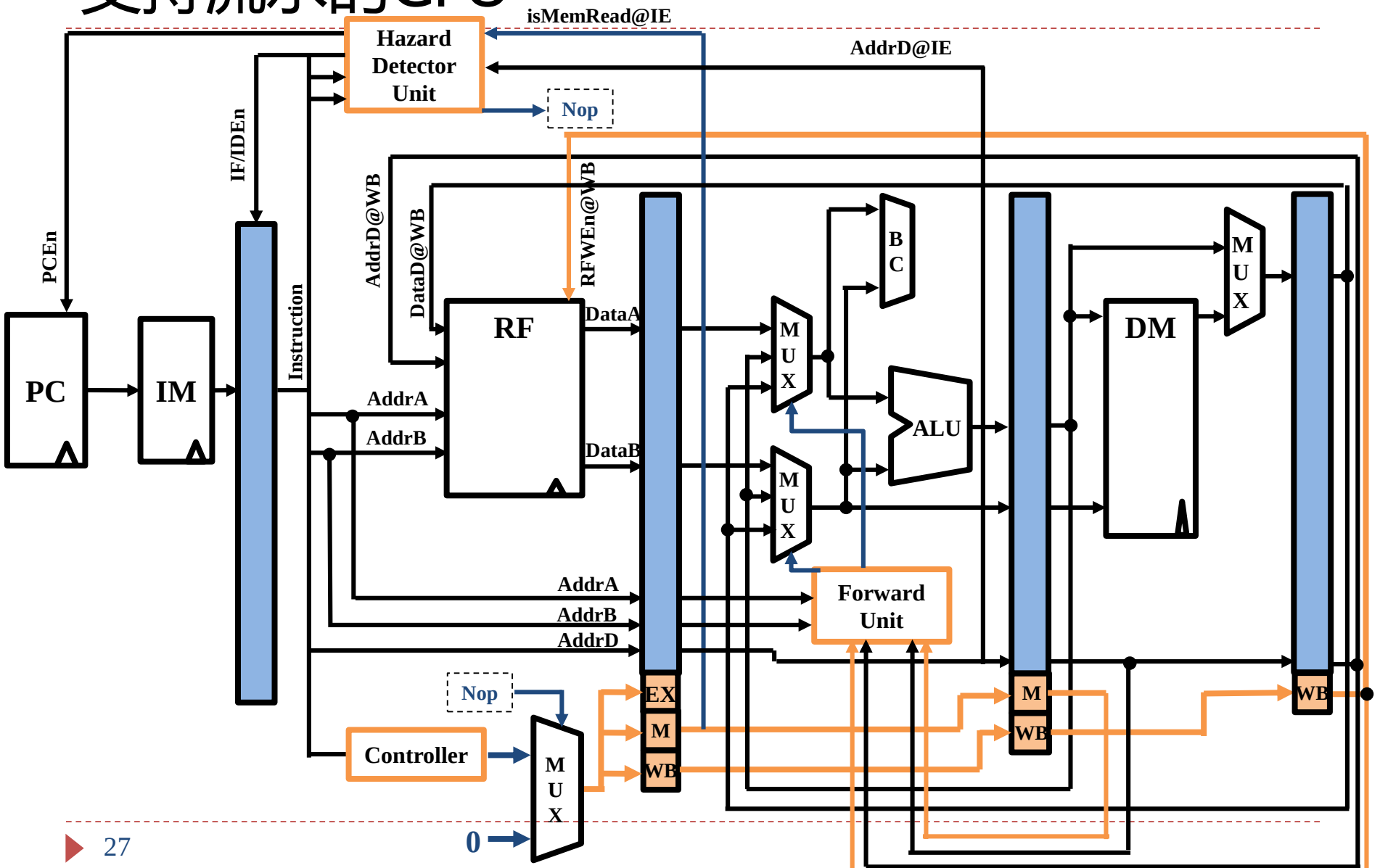
微程序控制器的组成


主振
启停
④
控制
存储器
③
下地址
逻辑
映射
②
IR
①
PC
控制条件
微指令寄存器
下地址
微命令
数据总线
控制总线
地址总线
接口
输出
设备
接口
输入
设备
主存
储器
地址寄存器
运算器
部件

硬连线控制器


主振
启停
时序控制
信号
形成部件
控制条件
操作码
IR
PC
节拍
控制器信号
数据总线
控制总线
地址总线
输出
设备
输入
设备
主
存
地址寄存器
运算器
部件

支持流水的CPU


isMemRead@IE
Hazard Detector Unit
AddrD@IE
Nop
IF/IDEn
PCEn
AddrD@WB
DataD@WB
RFWEn@WB
PC
IM
Instruction
RF
DataA
DataB
AddrA
AddrB
B
C
M
U
X
ALU
DM
Forward Unit
AddrA
AddrB
AddrD
Nop
EX
M
WB
Controller
0

微程序控制器

- 下地址字段
  - 指出下一个微操作
- 微控存
  - 给出全部控制信号
- 为什么能完成?
  - 指令系统是有限的，且指令的微操作也是有限的
  - 存储器技术
- 有什么好处?
  - 扩展容易、实现简单、兼容性好
  - 复杂的指令系统CISC

组合逻辑控制器

- 节拍发生器
  - 标明当前微操作
  - 完成微操作间的转换
- 控制信号生成逻辑
  - 输入：节拍状态和指令操作码
  - 输出：该微操作的全部控制信号
- 为什么能完成?
  - 精简的指令系统、更强大的逻辑实现能力
- 有什么好处?
  - 速度快、适合流水线操作

指令流水的控制

- 段间寄存器
  - 标明当前的流水段
  - 保存上一流水步骤的结果及后续控制信号
- 控制信号生成逻辑
  - 输入：指令操作码
  - 输出：该指令的全部控制信号
- 为什么能完成?
  - 规整的指令系统、更强大的逻辑实现能力
  - 精致的指令执行步骤划分
- 有什么好处?
  - 多条指令并行执行，性能高

多周期CPU的控制器设计

- 确定指令系统
  - 操作码、操作数地址、寻址方式
- 划分指令流程
- 设计每个微操作的控制信号
- 设计节拍或下地址
- 设计时序、启停等其他电路

指令流水

- 指令流水的基本概念
- 指令流水中的冲突
  - 结构冲突
  - 数据冲突
  - 控制冲突
- 指令流水冲突的解决方案
  - 插入等待周期（气泡）
  - 增加资源
  - 旁路技术
  - 分支预测
    - 动态/静态
- 指令流水实现
  - 时空图

存储器

- 处于计算机中心
- 层次存储器结构
  - 高速缓冲存储器（Cache）
  - 主存储器（DRAM）
  - 虚拟存储器
- 半导体存储器
  - SRAM
  - DRAM
  - FLASH
- 磁表面存储器
- 每种存储介质的存储原理、特点

层次存储器系统

- 程序的局部性原理
  - 时间局部性
  - 空间局部性
- 层次间应满足的原则
  - 包含性
  - 一致性

动态存储器的工作特点

- 破坏性读出
  - 读出时被强制清零
  - 预充电延迟
- 需定期刷新
  - 集中刷新
    - 停止读写，逐行刷新
  - 分散刷新
    - 定时周期性刷新
- 快速分页组织

静态存储器（缓存）

- 硬件实现
- 参数
- 地址映射
  - 直接映射
  - 全相联
  - 多路组相联
- 提高命中率
  - 块大小、Cache容量、替换算法
  - 写策略
- 缓存一致性

虚拟存储器

- 逻辑空间到物理空间
- 虚存的管理
  - 段式管理、页式管理、段页式管理
- 段表
  - 段起始地址、段长、控制位
- 页表
  - 实页号、控制位
  - 每一个虚页均在页表中有一个表项进行说明
- 快表（TLB）
  - 页表的Cache，实现虚页号到实页号的转换
  - 转换速度

RISC-V基于页面的虚拟内存

- 分页命名模式：SvX，其中X是以位为单位的虚拟地址长度
- 内存划分为固定大小的页面进行地址转换和对内存内容的保护（页面大小通常为4KB，也有大页面粒度）
- 启用分页的时候，大多数地址（包括load和store的有效地址和PC中的地址）都是虚拟地址
- 要访问物理内存，虚拟地址必须被转换为真正的物理地址
- 通过页表的结构来进行转换
- 权限位指示那些权限模式和通过哪种类型的访问可以操作这个页
- 访问未被映射的页或者访问权限不足会导致页面错误异常（page fault exception）

RISC-V中的TLB

- 如果取指，load，store要访问多次页表，将会大大降低性能
- 所有的现代处理器都使用地址转换缓存（TLB：Translation Lookaside Buffer）来减少这种开销
- 如果操作系统修改了页表，TLB就会变得不可用
- sfence.vma通知处理器，软件可能已经修改了页表，处理器可以刷新TLB
  - rs1指示了那个虚拟地址对应的转换被修改了
  - rs2指示被修改页表的地址空间标识符（一般相当于进程）ASID
  - 如果两者都是x0，整个TLB会被刷新

RISC-V的特权级模式

- 用户模式（user mode）：执行用户应用程序的模式，本学期的大部分指令都执行在用户模式
- 机器模式（machine mode）：运行最可信的代码，直接接触硬件
- 监管者模式（supervisor mode）：为现代操作系统例如Linux等提供支持
- 机器模式和监管者模式都比用户模式有着更高的权限

层级数目	支持的特权级模式	目标软件
1	M	简单的嵌入式系统
2	M, U	安全的嵌入式系统
4	M, S, U	运行操作系统

RISC-V特权指令

- 特权指令以及相应的指令码
- 指令码非常少，很多功能通过控制状态寄存器（CSR：Control State Register）来实现
- mret 机器模式返回，sret 监管者模式返回，wfi 等待中断，sfense.vma 虚拟地址屏障指令，见后面

RV32/64 Privileged Instructions

machine-mode
supervisor-mode } trap return

supervisor-mode fence.virtual memory address

wait for interrupt

31 27	26 25	24 20	19 15	14 12	11 7	6 0	
0001000		00010	00000	000	00000	1110011	R sret
0011000		00010	00000	000	00000	1110011	R mret
0001000		00101	00000	000	00000	1110011	R wfi
0001001		rs2	rs1	000	00000	1110011	R sfence.vma

RISC-V的异常和中断

Interrupt / Exception mcause[XLEN-1]	Exception Code mcause[XLEN-2:0]	Description
1	1	Supervisor software interrupt
1	3	Machine software interrupt
1	5	Supervisor timer interrupt
1	7	Machine timer interrupt
1	9	Supervisor external interrupt
1	11	Machine external interrupt
0	0	Instruction address misaligned
0	1	Instruction access fault
0	2	Illegal instruction
0	3	Breakpoint
0	4	Load address misaligned
0	5	Load access fault
0	6	Store address misaligned
0	7	Store access fault
0	8	Environment call from U-mode
0	9	Environment call from S-mode
0	11	Environment call from M-mode
0	12	Instruction page fault
0	13	Load page fault
0	15	Store page fault

中断时 mcause 的最高有效位置 1，同步异常时置 0，且低有效位
标识了中断或异常的具体原因。只有在实现了监管者模式时才能处理监管者模式中断和页面错误异常

同步异常

- 访问错误异常：物理内存不支持访问类型（写入ROM）时发生
- 断点异常：执行ebreak指令，或者地址与数据与调试触发器匹配时发生
- 环境调用异常：在执行ecall的时候发生，相当于系统调用
- 非法指令异常：在译码中发现无效操作码时发生
- 非对齐地址异常：有效地址不能被访问大小整除时发生

M模式下的异常处理

- 8个控制状态寄存器（CSR）是机器模式下异常处理的必要部分
  - mtvec(Machine Trap Vector)：发生异常时处理器需要跳转到的地址
  - mepc(Machine Exception PC)：指向发生异常的指令
  - mcause(Machine Exception Cause)：发生异常的种类
  - mie(Machine Interrupt Enable)：处理器目前能处理和必须忽略的中断
  - mip(Machine Interrupt Pending)：正准备处理的中断
  - mtval(Machine Trap Value)：trap的附加信息：地址异常中出错的地址，非法指令异常的指令等
  - mscratch(Machine Scratch)：暂时存放一个字大小的数据
  - mstatus(Machine Status)：机器的状态

输入输出系统和设备

- 控制方式
  - CPU如何控制输入/输出？（输入/输出方式）
- 传输方式
  - 使用传输通道、方式、速率等（总线、接口）
- 数据识别和转换
  - 数/模转换、语音识别等，转换为字符、数据等计算机能识别的格式（设备）

控制方式

- 程序直接控制（轮询）
  - CPU直接使用输入/输出指令来控制外部设备
- 程序中断
  - 外部设备请求，CPU响应，CPU与外设并行工作
- 直接存储访问（DMA）
  - 专用输入/输出控制器
  - 独占总线/总线周期窃取
- 通道
  - 字节多路通道
  - 选择通道
  - 数组多路通道
- 外围处理机

总线

- 多设备共享的信息通道
  - 地址
  - 数据
  - 控制
- 多总线系统
- 总线仲裁
  - 主设备和从设备
  - 集中仲裁、分布仲裁
- 总线传输
  - 同步
  - 异步

成组传输

- 对主存储器的要求
  - Cache、DMA等
  - SDRAM
  - PCI总线
- 经过等待时间后，按总线时钟传送数据

接口

- 提供主机识别（指定、找到）使用的I/O设备的支持（为每个设备规定几个地址码或编号）
- 建立主机和设备之间的控制与通信机制
- 提供主机和设备之间信息交换过程中的数据缓冲机构
- 提供主机和设备之间信息交换过程中的其他特别需求支持

外部设备功能

- 完成数据的输入（和/或）输出
  - 信号转换
  - 数据采样
- 与接口进行连接
  - 接口信号，电平标准等
- 与主机进行通信
  - 通过总线进行
  - 速度和方式

考试时间和地点

- 考试时间：
  - 2021年12月27日14：30~16：30
- 考试地点：
  - 一教201（共89人）

关于考试的说明

- 闭卷考试：
- 考试范围：
  - 教学大纲规定的范围
  - 通读教材
- 题型：判断，选择，填空，简答，综合
- 注意事项：
  - 严禁任何形式的作弊
  - 不使用计算器
  - 仔细审题
  - 回答简洁
- 占总评比例：40% 或100%

复习提示

- 仔细阅读PPT，一些基本概念和基本方法需要理解
- 认真领会PPT中的内容
- 考试内容覆盖全面，复习也需要全面
- 数据表示，存储，输入输出等的内容都会考到
- CPU设计这一块流水线是重点
- 12月24日周五课程安排
  - 教学内容：ARM指令系统
  - 授课教师：陈康
  - 授课地点：三教3200或线上

谢谢