

计算机组成原理

第二十二讲

刘松波

哈工大计算学部

模式识别与智能系统研究中心

第7章 指令系统

7.1 机器指令

7.2 操作数类型和操作类型

7.3 寻址方式

7.4 指令格式举例

7.5 RISC 技术

7.5 RISC 技术

一、RISC 的产生和发展

RISC (Reduced Instruction Set Computer)

CISC (Complex Instruction Set Computer)

80 — 20 规律 — RISC技术

- 典型程序中 **80%** 的语句仅仅使用处理机中 **20%** 的指令
- 执行频度高的简单指令，因复杂指令的存在，执行速度无法提高
- ？ 能否用 **20%** 的简单指令组合不常用的 **80%** 的指令功能

二、RISC 的主要特征

- 选用使用频度较高的一些 简单指令，复杂指令的功能由简单指令来组合
- 指令 长度固定、指令格式种类少、寻址方式少
- 只有 **LOAD / STORE** 指令访存
- CPU 中有多个 通用 寄存器
- 采用 流水技术 一个时钟周期 内完成一条指令
- 采用 组合逻辑 实现控制器
- 采用 优化 的 编译 程序

三、CISC 的主要特征

- 系统指令 复杂庞大，各种指令使用频度相差大
- 指令 长度不固定、指令格式种类多、寻址方式多
- 访存 指令 不受限制
- CPU 中设有 专用寄存器
- 大多数指令需要 多个时钟周期 执行完毕
- 采用 微程序 控制器
- 难以 用 优化编译 生成高效的代码

四、RISC和CISC 的比较

1. RISC更能 充分利用 VLSI 芯片的面积

2. RISC 更能 提高计算机运算速度

指令数、指令格式、寻址方式少，
通用 寄存器多，采用 组合逻辑，
便于实现 指令流水

3. RISC 便于设计，可 降低成本，提高 可靠性

4. RISC 有利于编译程序代码优化

5. RISC 不易 实现 指令系统兼容

第 8 章 CPU 的结构和功能

8.1 CPU 的结构

8.2 指令周期

8.3 指令流水

8.4 中断系统

8.1 CPU 的结构

一、CPU 的功能

1. 控制器的功能

取指令

指令控制

分析指令

操作控制

执行指令，发出各种操作命令

时间控制

控制程序输入及结果的输出

总线管理

处理中断

处理异常情况和特殊请求

数据加工

2. 运算器的功能

实现算术运算和逻辑运算

二、CPU 结构框图

8.1

1. CPU 与系统总线

指令控制

PC IR

操作控制

CU 时序电路

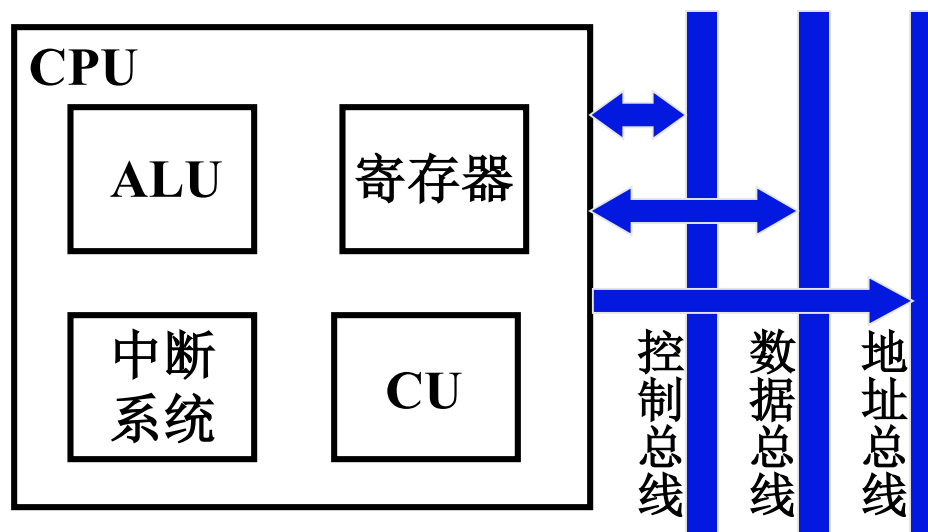
时间控制

数据加工

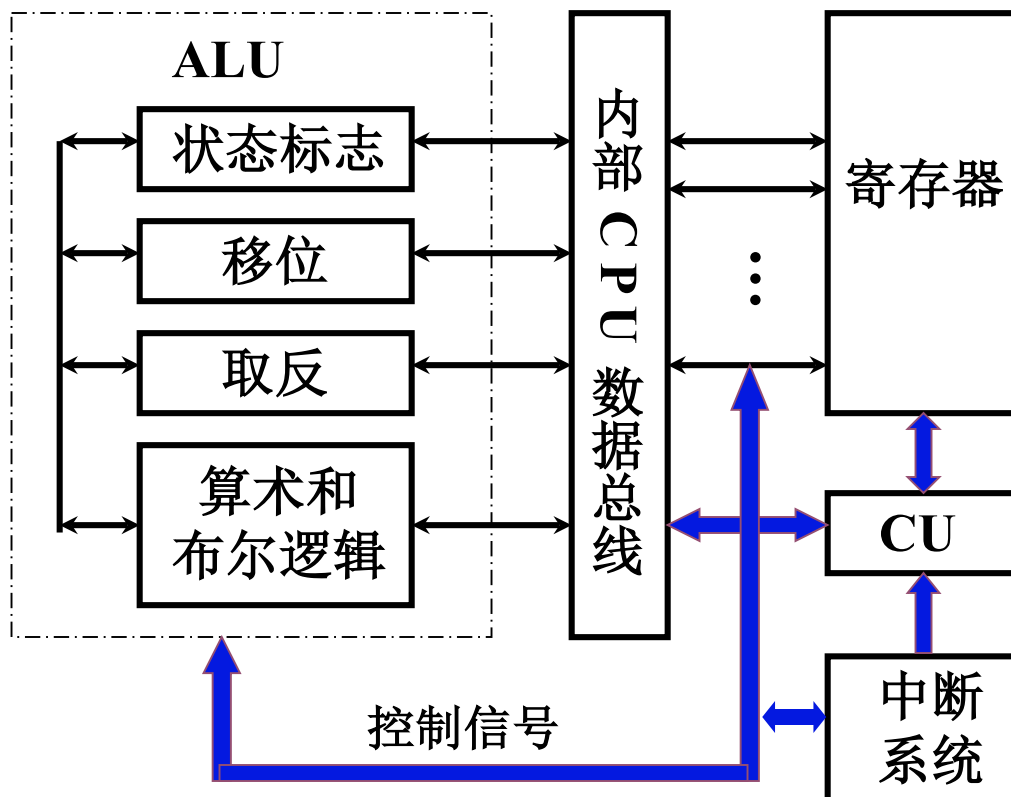
ALU 寄存器

处理中断

中断系统



2. CPU 的内部结构



三、CPU 的寄存器

8.1

1. 用户可见寄存器

- (1) 通用寄存器 存放操作数
可作 某种寻址方式所需的 专用寄存器
- (2) 数据寄存器 存放操作数（满足各种数据类型）
两个寄存器拼接存放双倍字长数据
- (3) 地址寄存器 存放地址，其位数应满足最大的地址范围
用于特殊的寻址方式 段基址 栈指针
- (4) 条件码寄存器 存放条件码，可作程序分支的依据
如 正、负、零、溢出、进位等

2. 控制和状态寄存器

(1) 控制寄存器

PC → MAR → M → MDR → IR

控制 CPU 操作

其中 **MAR、MDR、IR**

用户不可见

PC

用户可见

(2) 状态寄存器

状态寄存器

存放条件码

PSW 寄存器

存放程序状态字

3. 举例

Z8000

8086

MC 68000

四、控制单元 CU 和中断系统

1. CU 产生全部指令的微操作命令序列

组合逻辑设计

硬连线逻辑

微程序设计

存储逻辑

参见 第 4 篇

2. 中断系统

参见 8.4 节

五、ALU

参见 第 6 章

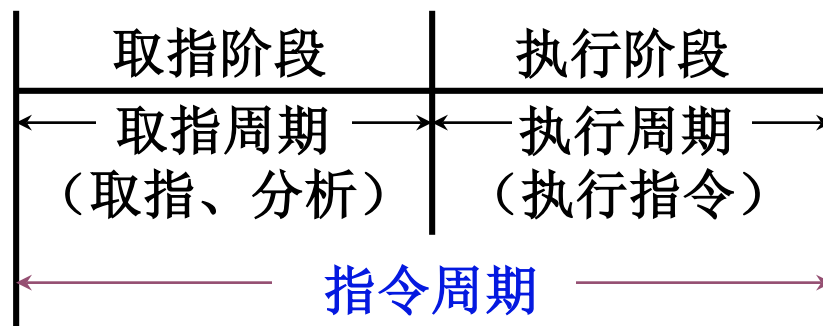
8.2 指令周期

一、指令周期的基本概念

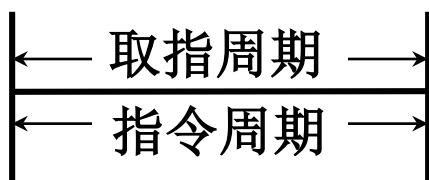
1. 指令周期

取出并执行一条指令所需的全部时间

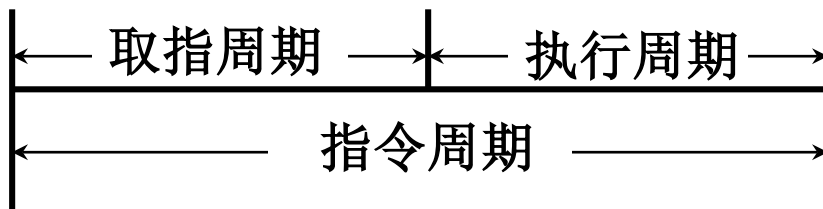
完成一条指令 { 取指、分析 取指周期
 执行 执行周期



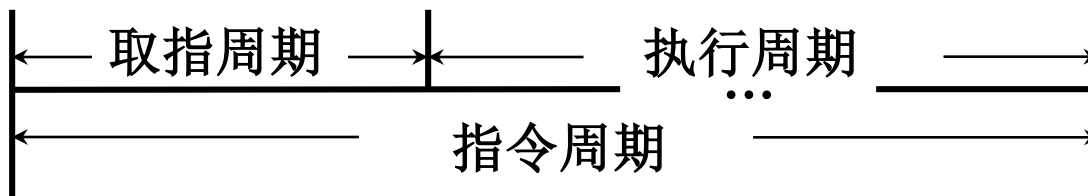
2. 每条指令的指令周期不同



NOP

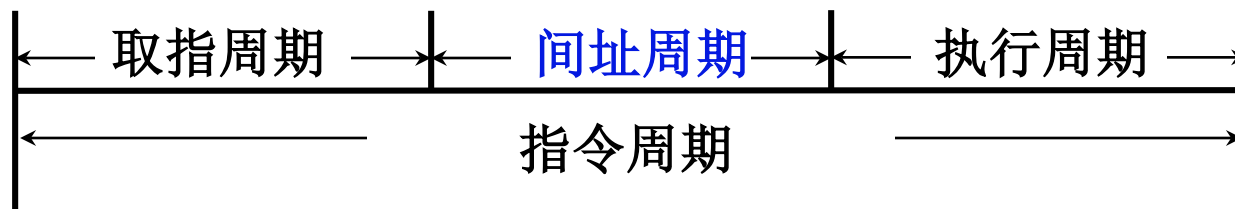


ADD mem

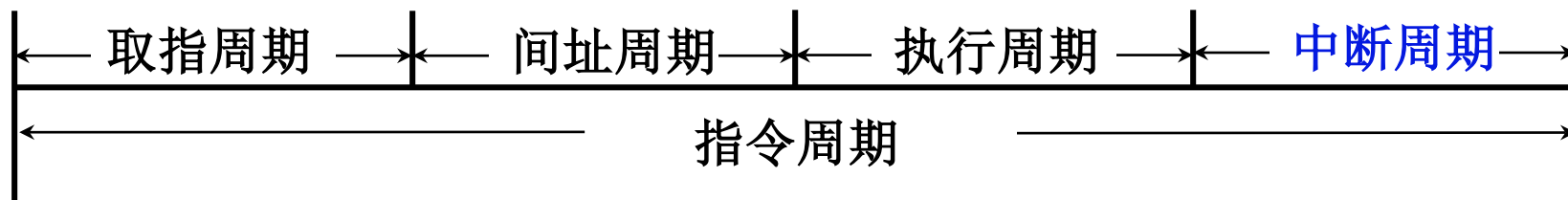


MUL mem

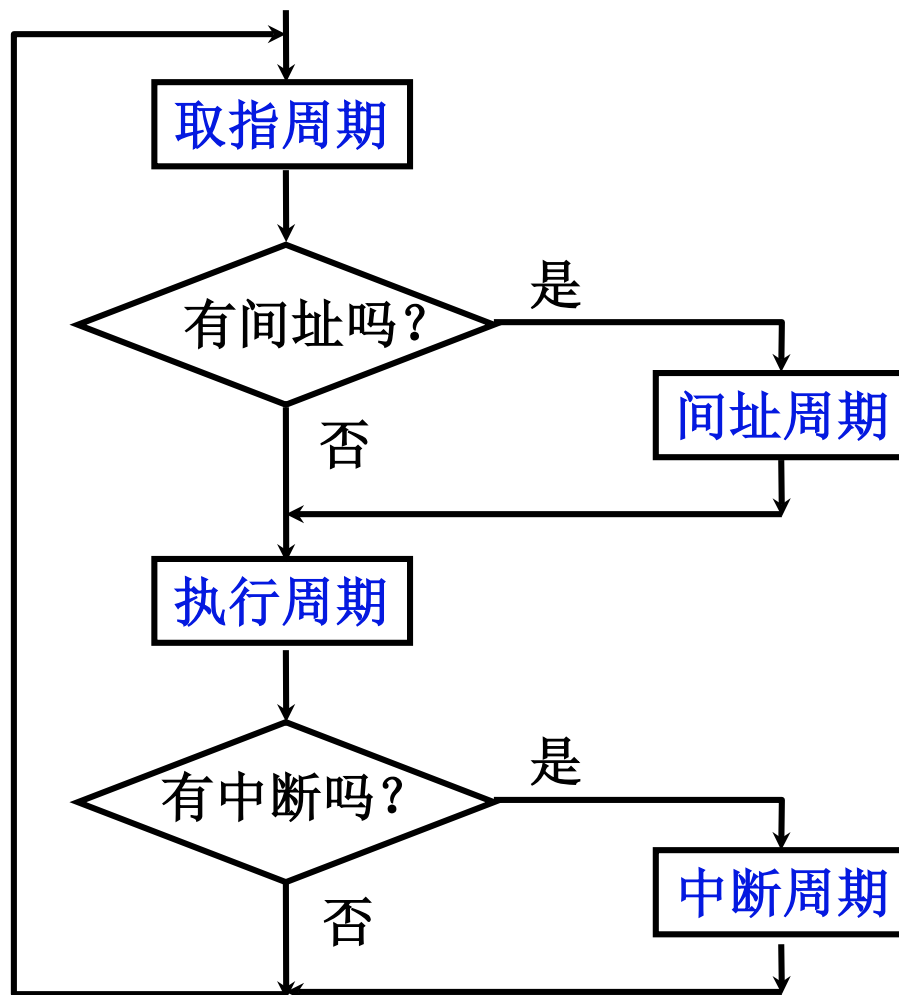
3. 具有间接寻址的指令周期



4. 带有中断周期的指令周期



5. 指令周期流程



6. CPU 工作周期的标志

CPU 访存有四种性质

取 指令

取指周期

取 地址

间址周期

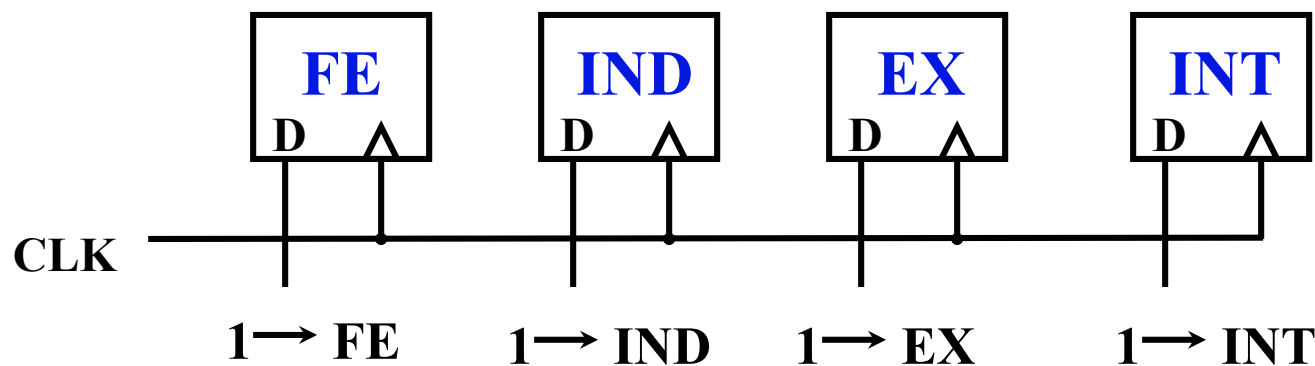
取 操作数

执行周期

存 程序断点

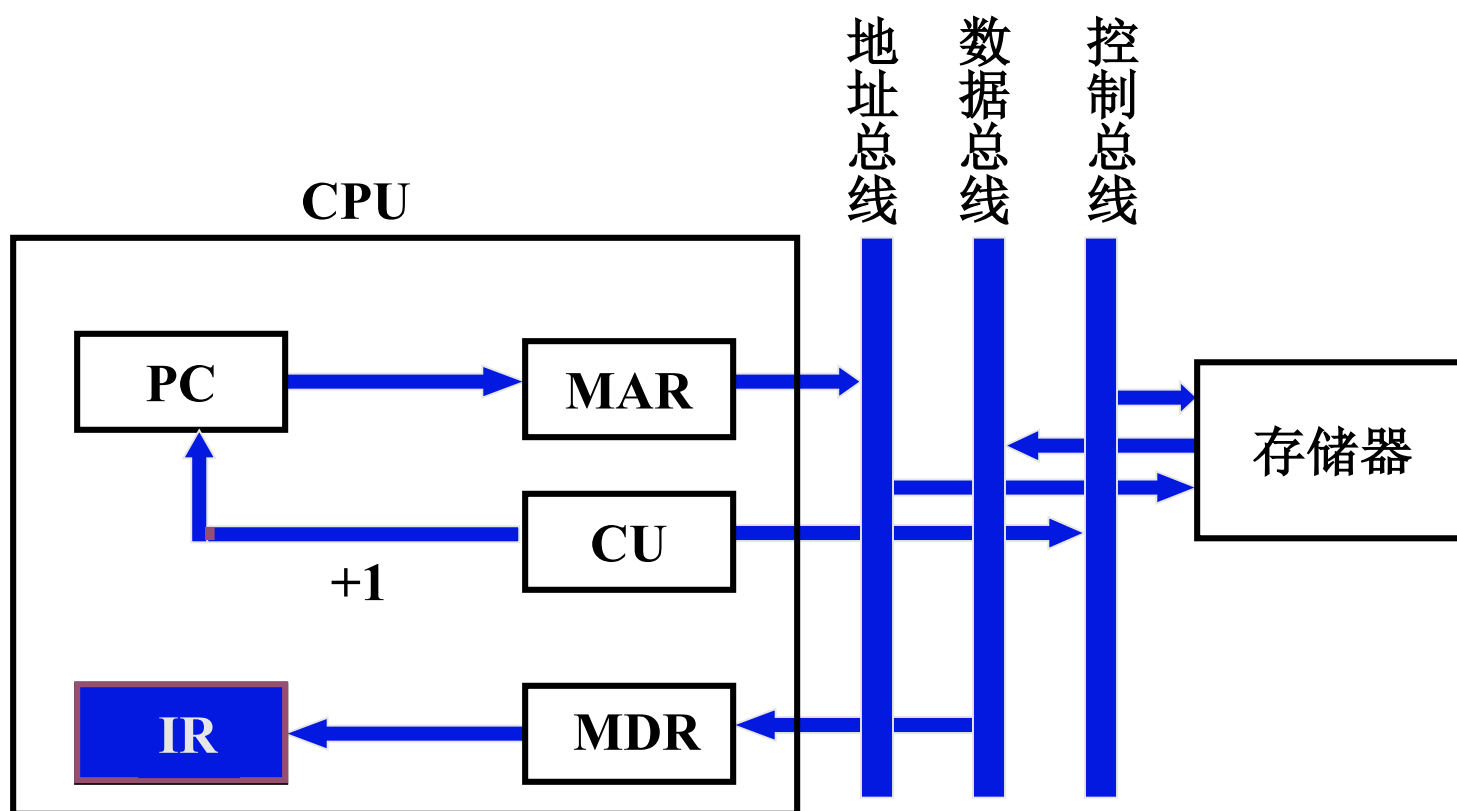
中断周期

CPU 的
4个工作周期

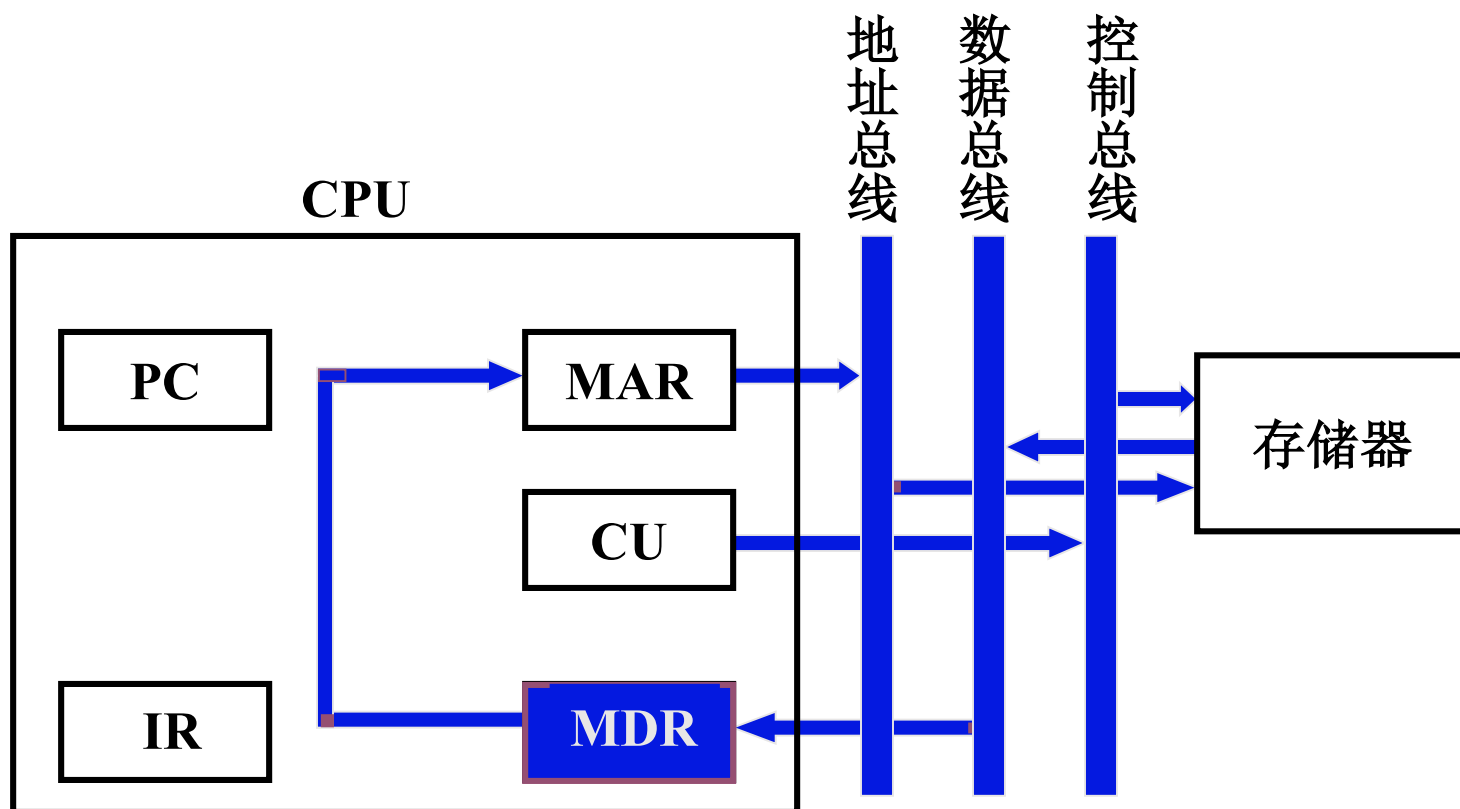


二、指令周期的数据流

1. 取指周期数据流



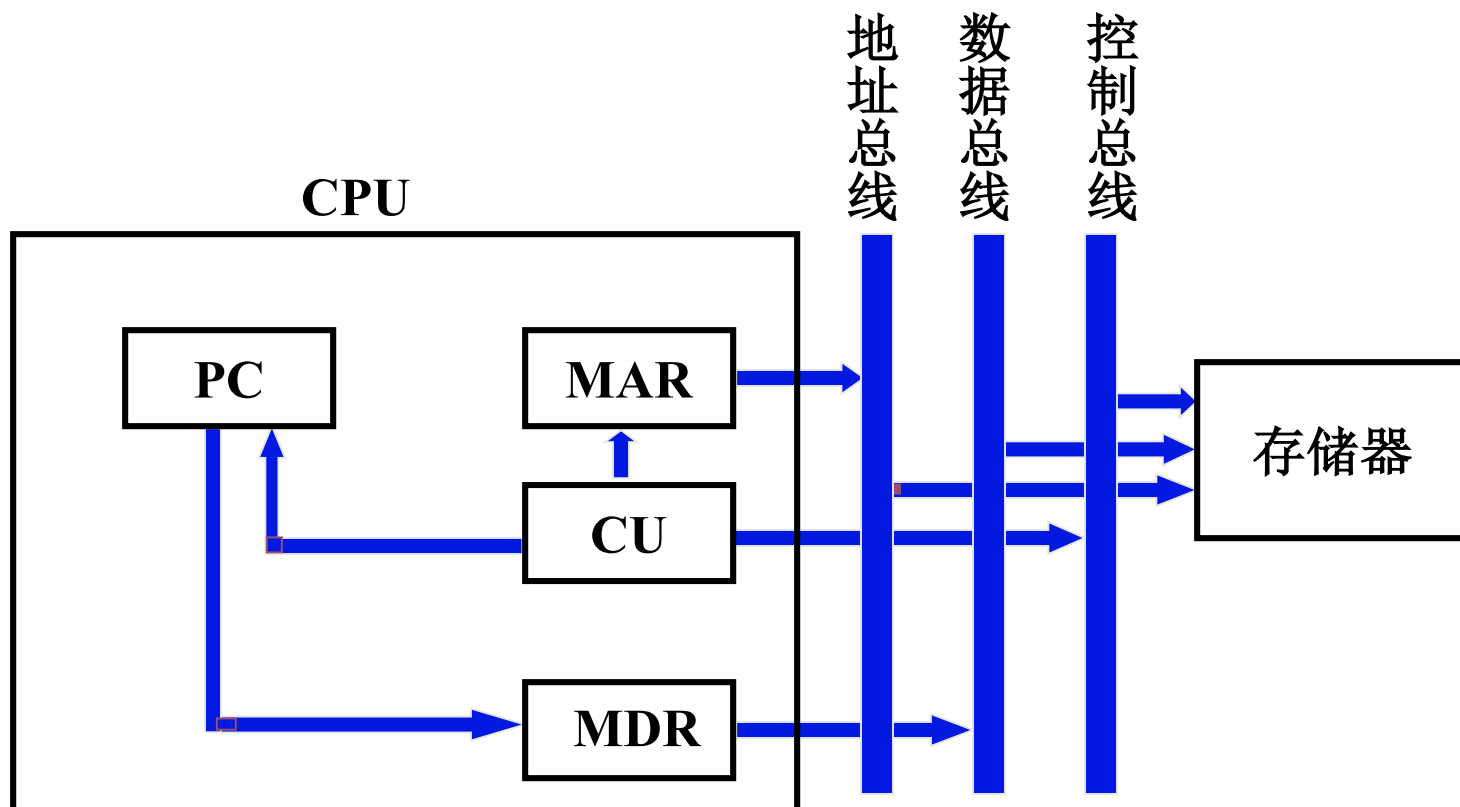
2. 间址周期数据流



3. 执行周期数据流

不同指令的执行周期数据流不同

4. 中断周期数据流



8.3 指令流水

一、如何提高机器速度

1. 提高访存速度

高速芯片

Cache

多体并行

2. 提高 I/O 和主机之间的传送速度

中断

DMA

通道

I/O 处理机

多总线

3. 提高运算器速度

高速芯片

改进算法

快速进位链

• 提高整机处理能力

高速器件

改进系统结构，开发系统的并行性

二、系统的并行性

8.3

1. 并行的概念

并行 { **并发** 两个或两个以上事件在 **同一时间段** 发生
同时 两个或两个以上事件在 **同一时刻** 发生

时间上互相重叠

2. 并行性的等级

过程级（程序、进程）	粗粒度	软件实现
指令级（指令之间） （指令内部）	细粒度	硬件实现

三、指令流水原理

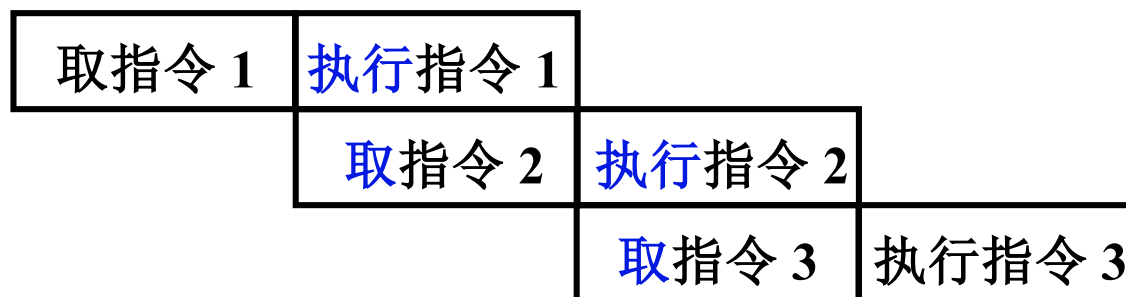
8.3

1. 指令的串行执行



取指令 取指令部件 完成 总有一个部件 空闲
执行指令 执行指令部件 完成

2. 指令的二级流水

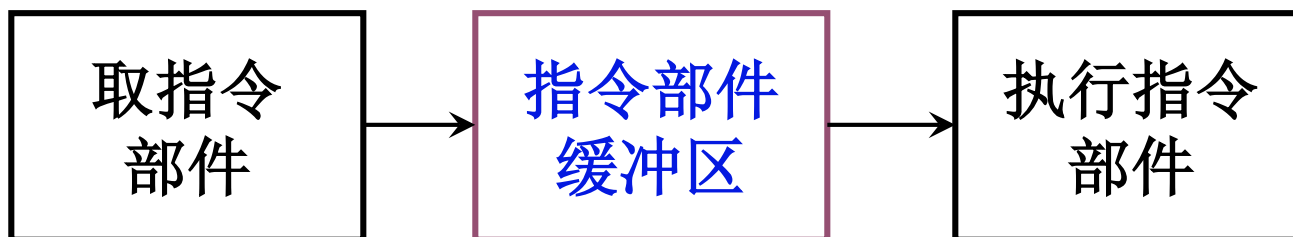


若 取指 和 执行 阶段时间上 完全重叠
指令周期 减半 速度提高 1 倍

3. 影响指令流水效率加倍的因素

8.3

(1) 执行时间 > 取指时间



(2) 条件转移指令 对指令流水的影响

必须等 上条 指令执行结束，才能确定 下条 指令的地址，

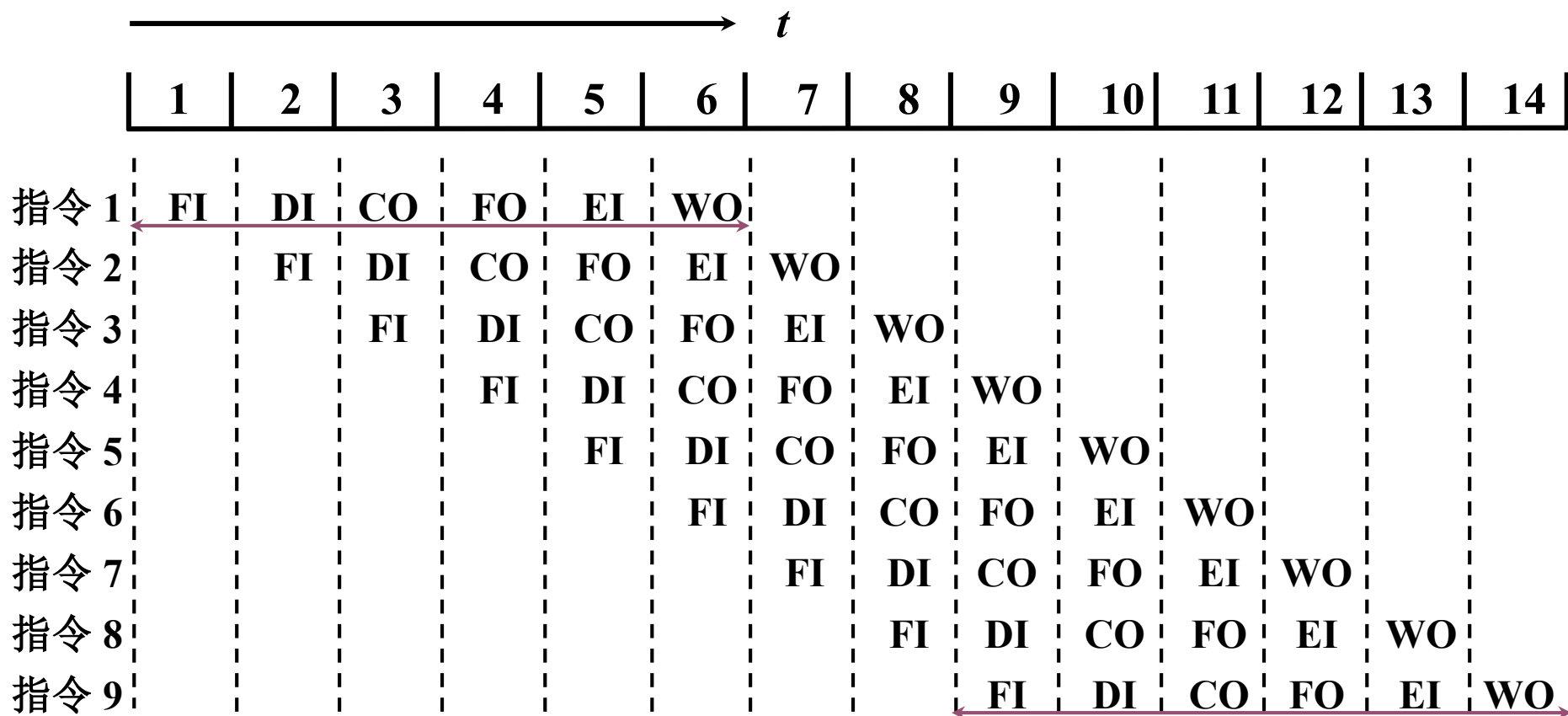
造成时间损失

猜测法

解决办法 ？

4. 指令的六级流水

8.3



完成 一条指令

串行执行

六级流水

6 个时间单位

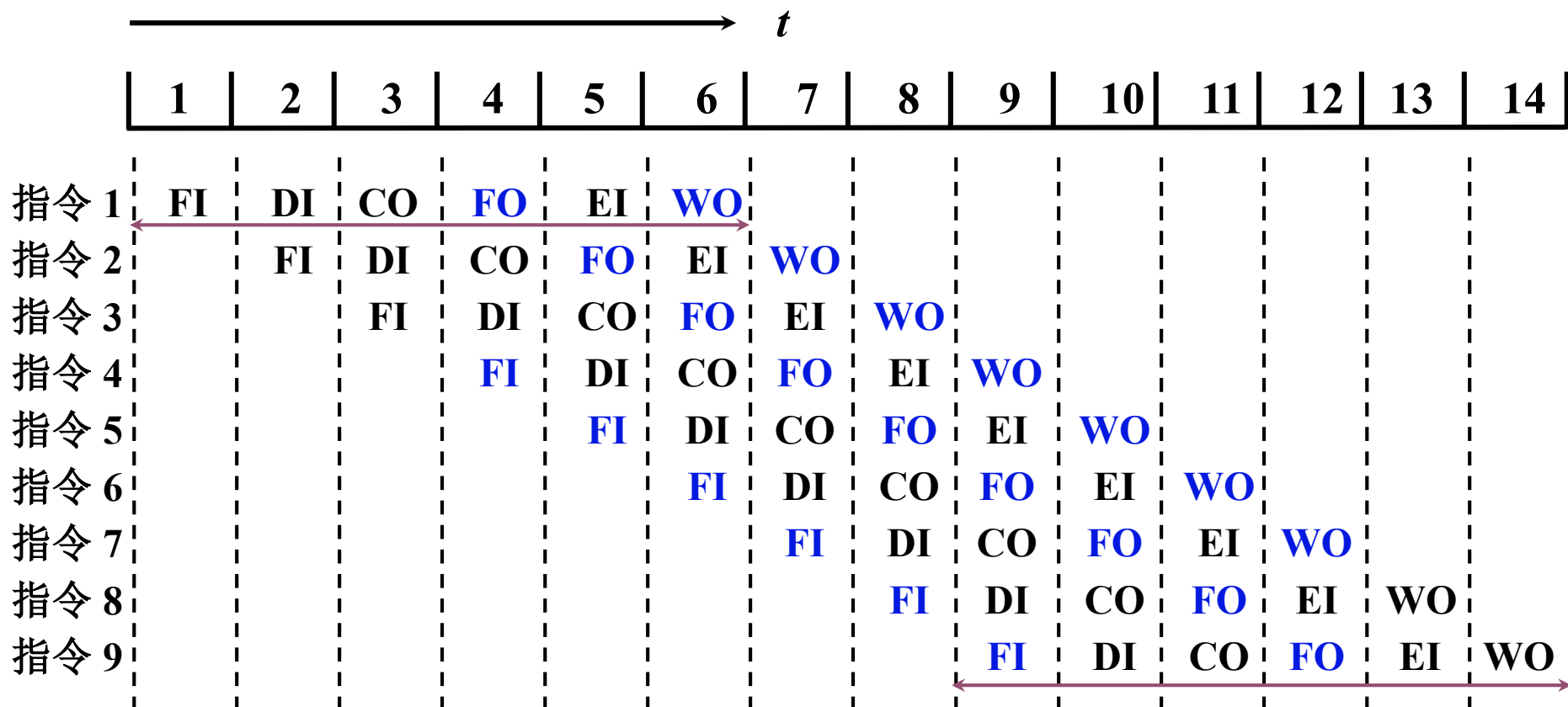
$6 \times 9 = 54$ 个时间单位

14 个时间单位

三、影响指令流水线性能的因素

8.3

1. 结构相关 不同指令争用同一功能部件产生资源冲突



解决办法

- 停顿
- 指令 1 与指令 4 冲突
- 指令 2 与指令 5 冲突
- 指令 1、指令 3、指令 6 冲突
- 指令存储器 and 数据存储器分开
- ...
- 指令预取技术 (适用于访存周期短的情况)

2. 数据相关

8.3

不同指令因重叠操作，可能改变操作数的 读/写 访问顺序

- 写后读相关 (RAW)

SUB R_1 , R_2 , R_3 ; $(R_2) - (R_3) \rightarrow R_1$

ADD R_4 , R_5 , R_1 ; $(R_5) + (R_1) \rightarrow R_4$

- 读后写相关 (WAR)

STA M, R_2 ; $(R_2) \rightarrow M$ 存储单元

ADD R_2 , R_4 , R_5 ; $(R_4) + (R_5) \rightarrow R_2$

- 写后写相关 (WAW)

MUL R_3 , R_2 , R_1 ; $(R_2) \times (R_1) \rightarrow R_3$

SUB R_3 , R_4 , R_5 ; $(R_4) - (R_5) \rightarrow R_3$

2022/8/24 解决办法


- 后推法

- 采用 旁路技术

3. 控制相关

8.3

由转移指令引起



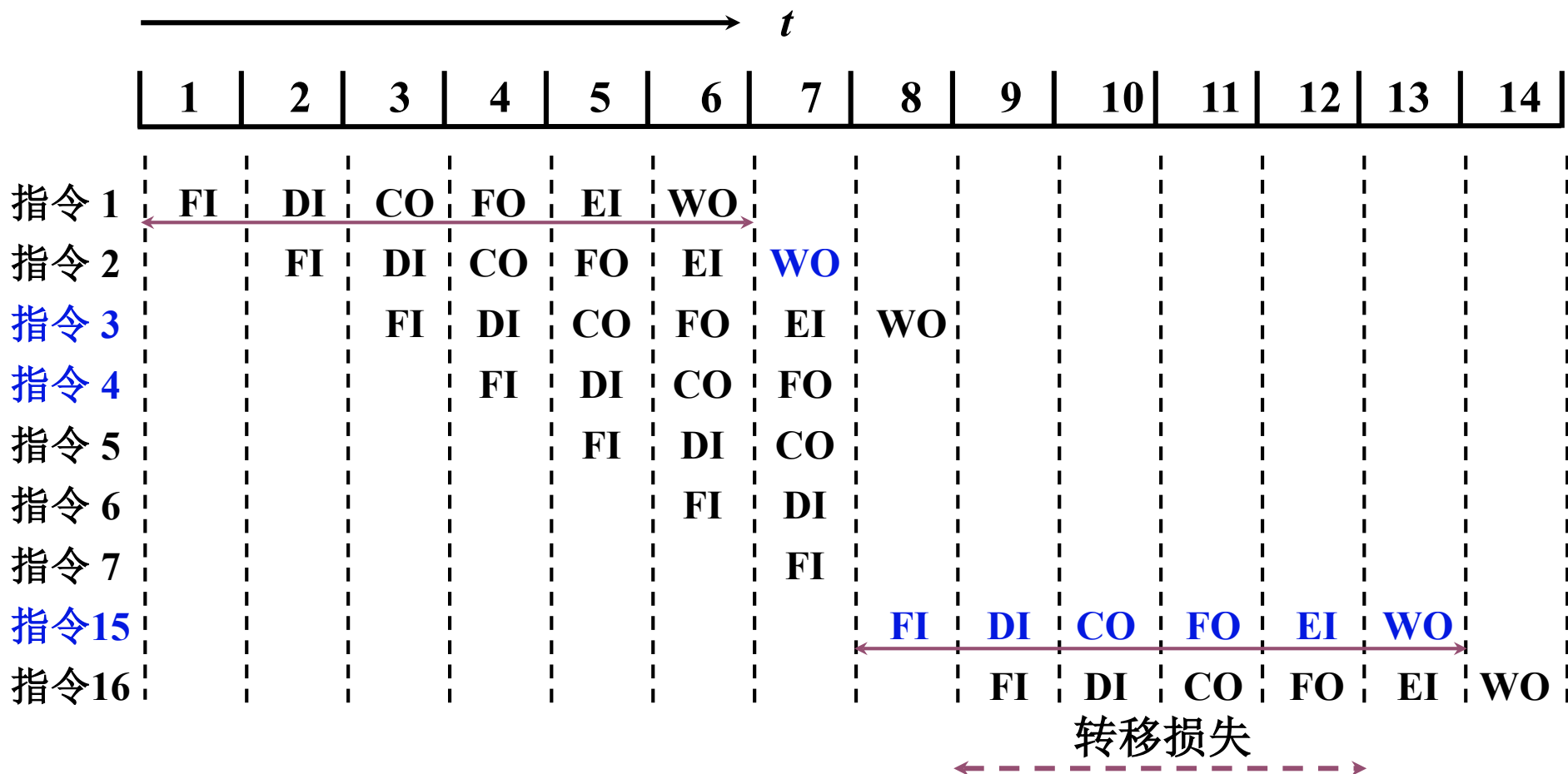
```
LDA    # 0
LDX    # 0
M → ADD  X, D
INX
CPX    # N
BNE    M
DIV    # N
STA    ANS
```

BNE 指令必须等
CPX 指令的结果
才能判断出
是转移
还是顺序执行

3. 控制相关

8.3

设 指令3 是转移指令



四、流水线性能

8.3

1. 吞吐率

单位时间内 流水线所完成指令 或 输出结果 的数量

设 m 段的流水线各段时间为 Δt

- 最大吞吐率

$$T_{pmax} = \frac{1}{\Delta t}$$

- 实际吞吐率

连续处理 n 条指令的吞吐率为

$$T_p = \frac{n}{m \cdot \Delta t + (n-1) \cdot \Delta t}$$

2. 加速比 S_p

m 段的流水线的速度与等功能的非流水线的速度之比

设流水线各段时间为 Δt

完成 n 条指令在 m 段流水线上共需

$$T = m \cdot \Delta t + (n-1) \cdot \Delta t$$

完成 n 条指令在等效的非流水线上共需

$$T' = nm \cdot \Delta t$$

则

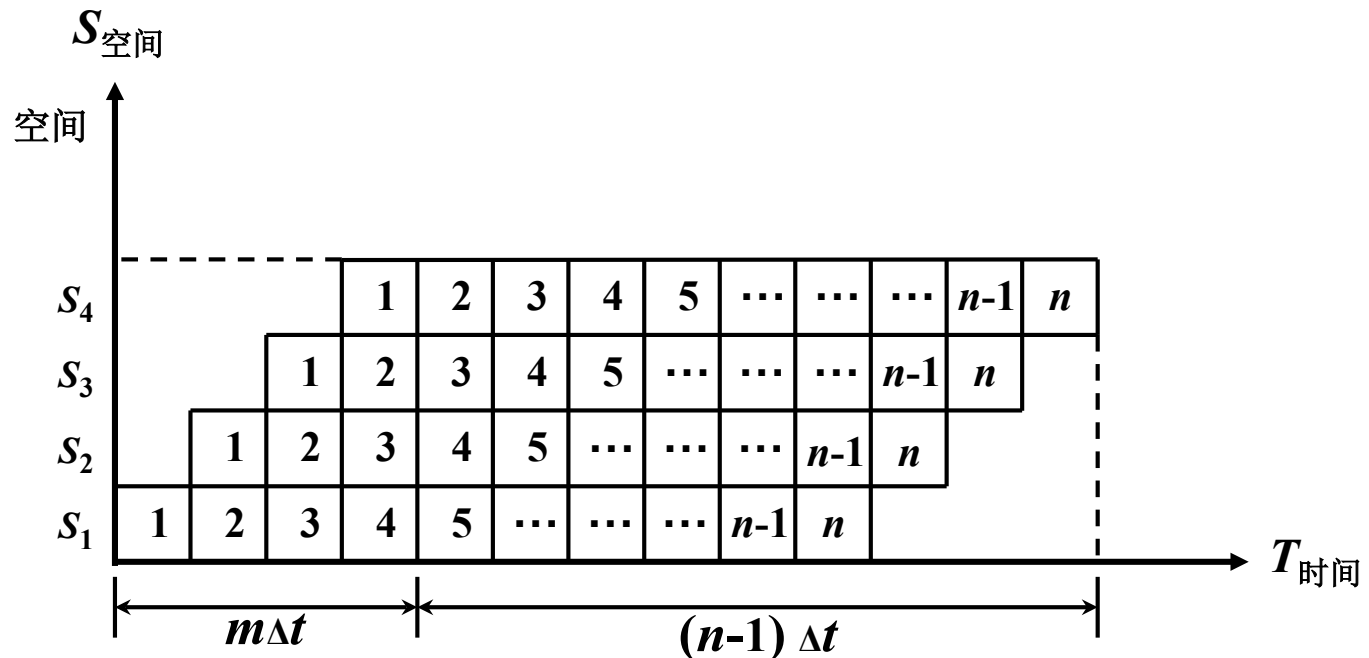
$$S_p = \frac{nm \cdot \Delta t}{m \cdot \Delta t + (n-1) \cdot \Delta t} = \frac{nm}{m + n - 1}$$

3. 效率

8.3

流水线中各功能段的 **利用率**

由于流水线有 **建立时间** 和 **排空时间**
因此各功能段的 **设备不可能** 一直 处于 **工作** 状态



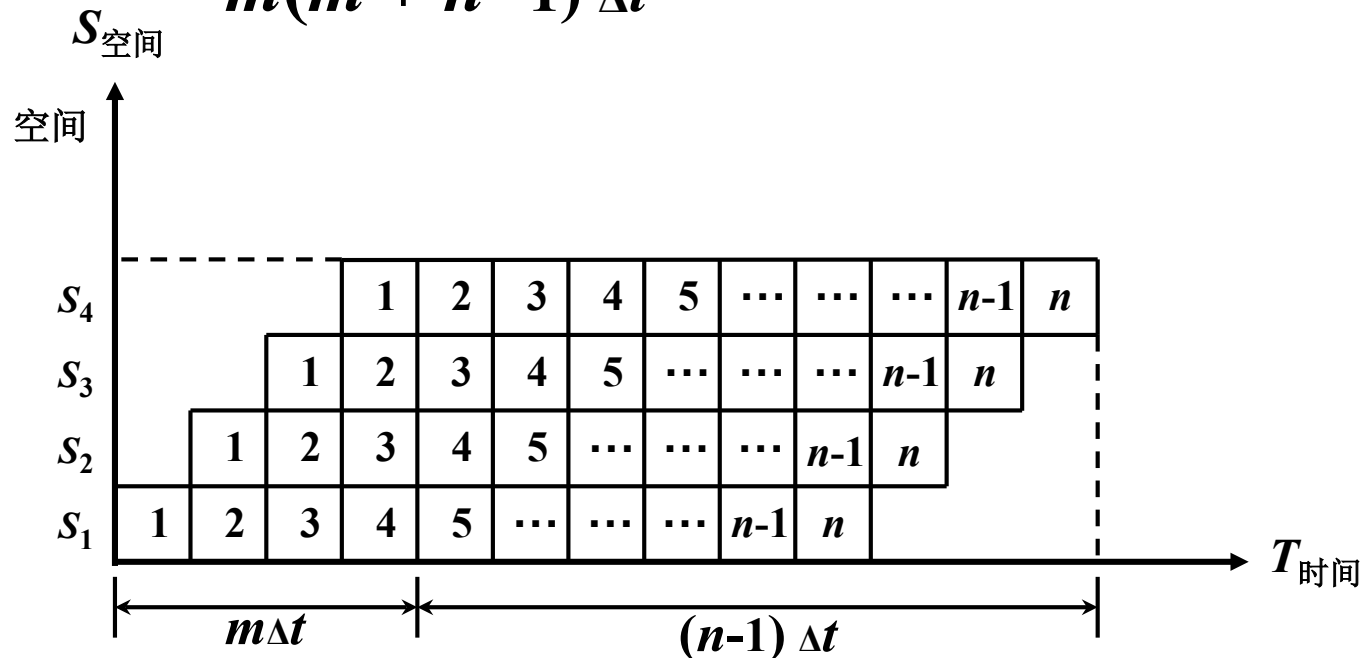
3. 效率

8.3

流水线中各功能段的 **利用率**

效率 = $\frac{\text{流水线各段处于工作时间的时空区}}{\text{流水线中各段总的时空区}}$

$$= \frac{mn\Delta t}{m(m+n-1)\Delta t}$$



五、流水线的多发技术

8.3

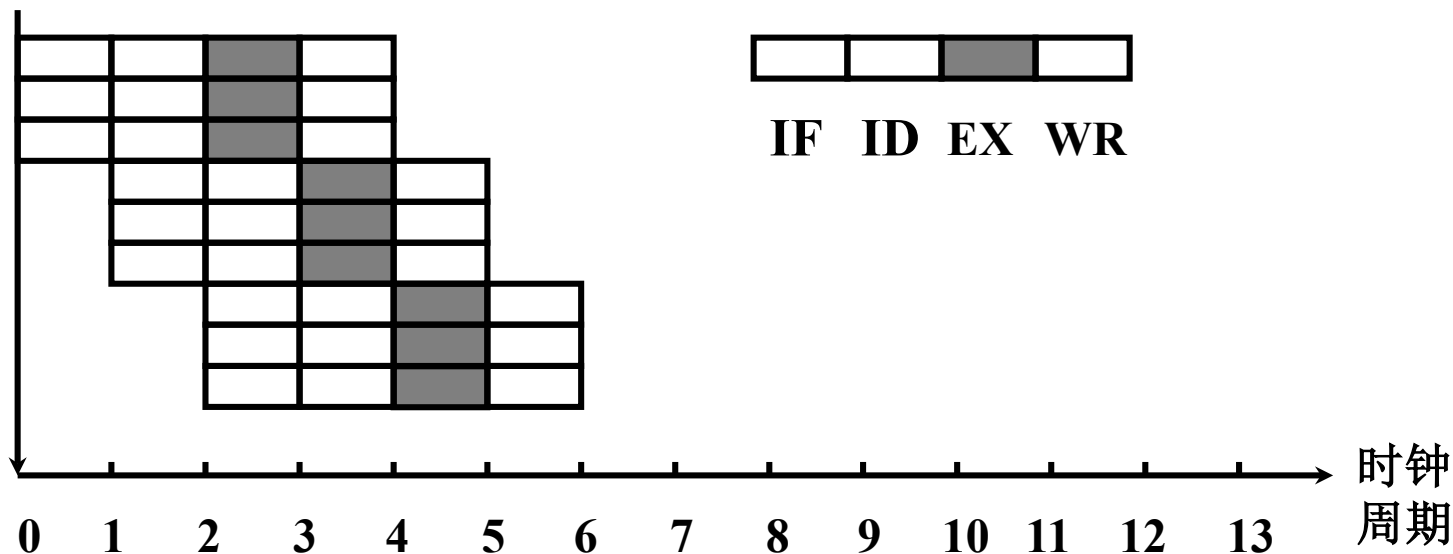
1. 超标量技术

- 每个时钟周期内可 **并发多条独立指令**
配置多个功能部件

- **不能调整** 指令的 **执行顺序**

通过编译优化技术，把可并行执行的指令搭配起来

指令序列



2. 超流水线技术

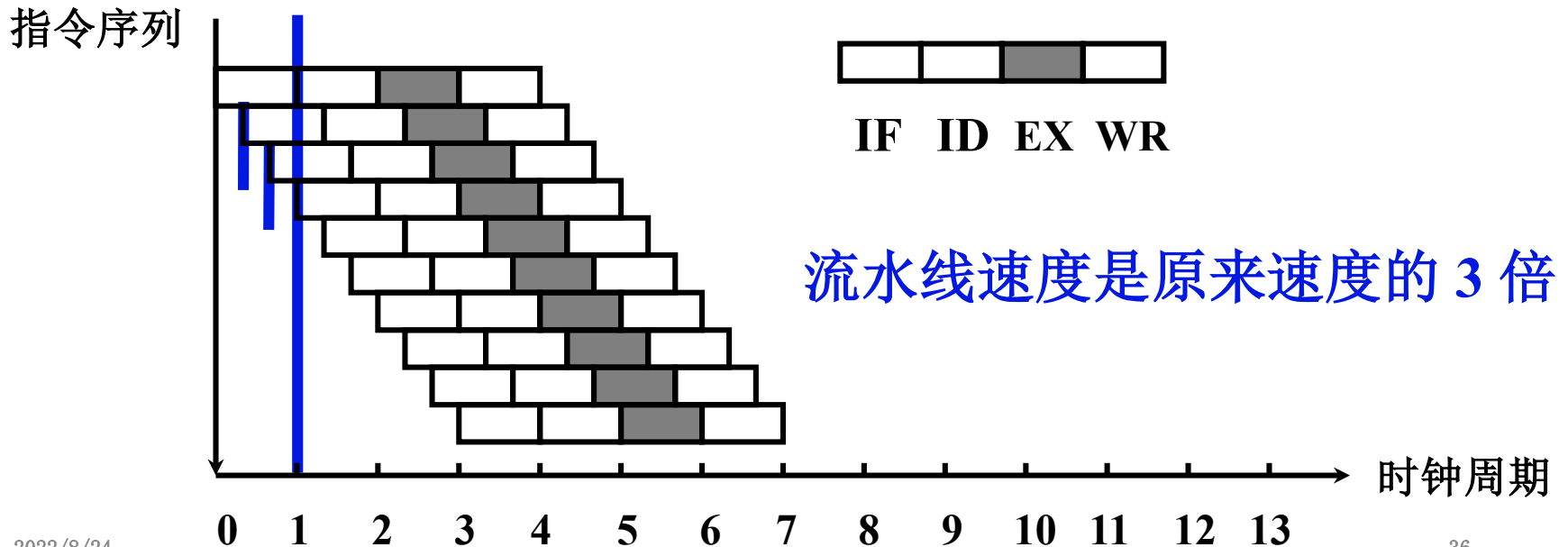
8.3

- 在一个时钟周期内再分段（3段）

在一个时钟周期内 一个功能部件使用多次（3次）

- 不能调整 指令的 执行顺序

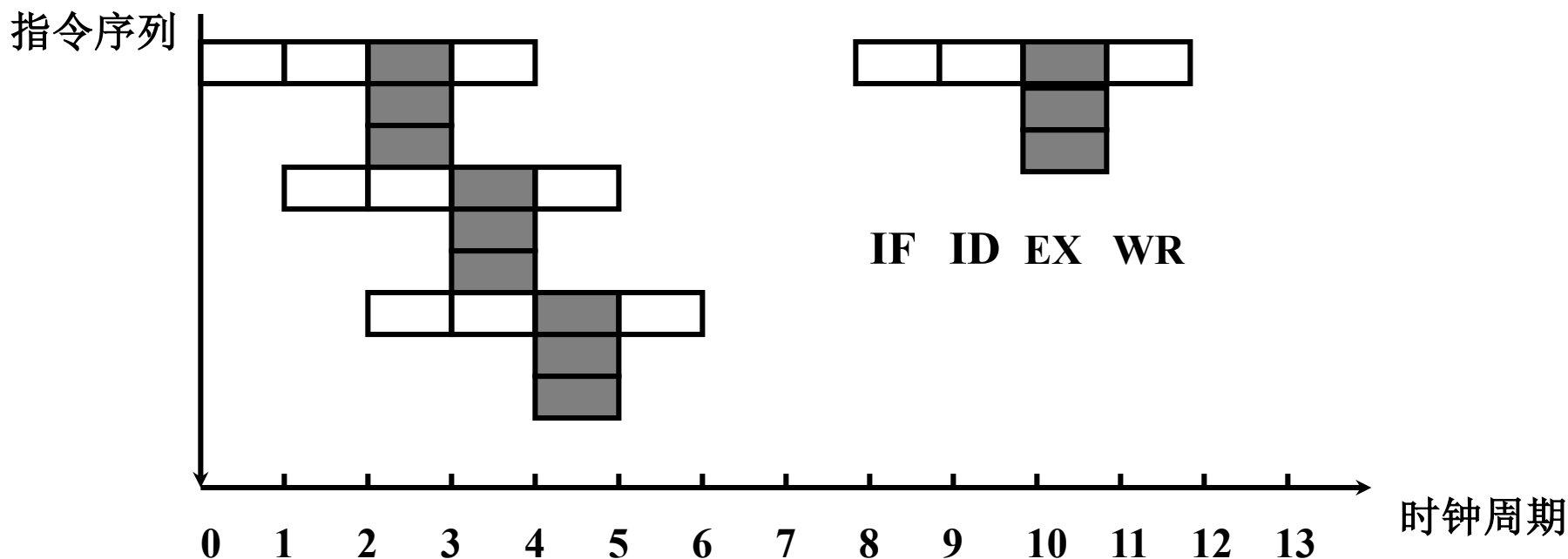
靠编译程序解决优化问题



3. 超长指令字技术

8.3

- 由编译程序 **挖掘** 出指令间 **潜在** 的 **并行性**，
将 **多条** 能 **并行操作** 的指令组合成 **一条**
具有 **多个操作码字段** 的 **超长指令字**（可达几百位）
- 采用 **多个处理部件**

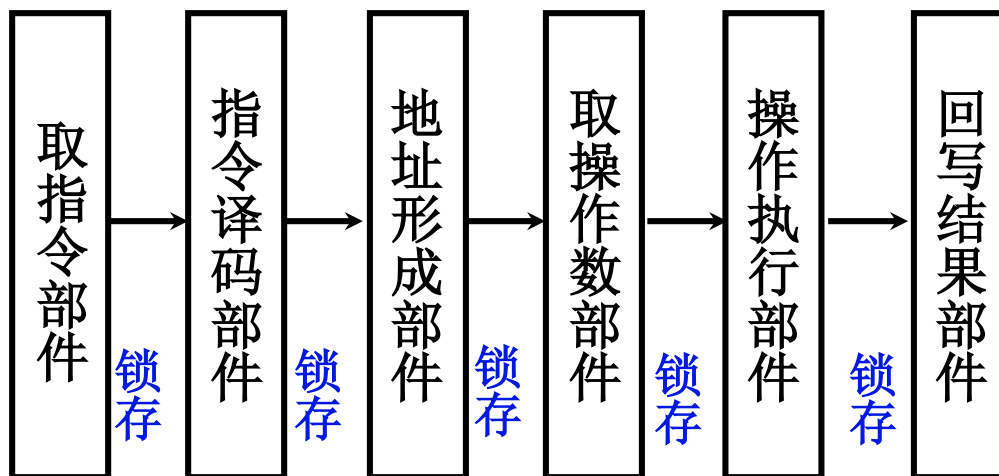


六、流水线结构

8.3

1. 指令流水线结构

完成一条指令分 6 段，每段需一个时钟周期



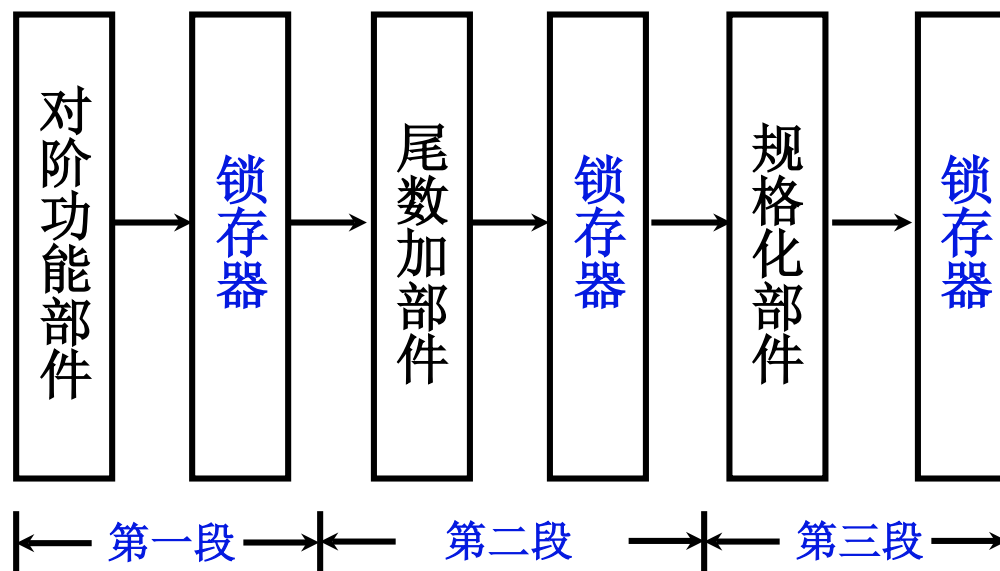
若 流水线不出现断流 1 个时钟周期出 1 结果

不采用流水技术 6 个时钟周期出 1 结果

理想情况下，6 级流水 的速度是不采用流水技术的 6 倍

2. 运算流水线

完成 浮点加减 运算 可分
对阶、尾数求和、规格化 三段



分段原则 每段 操作时间 尽量 一致