计算机组成原理

第十一讲

刘松波

哈工大计算学部

模式识别与智能系统研究中心

第 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

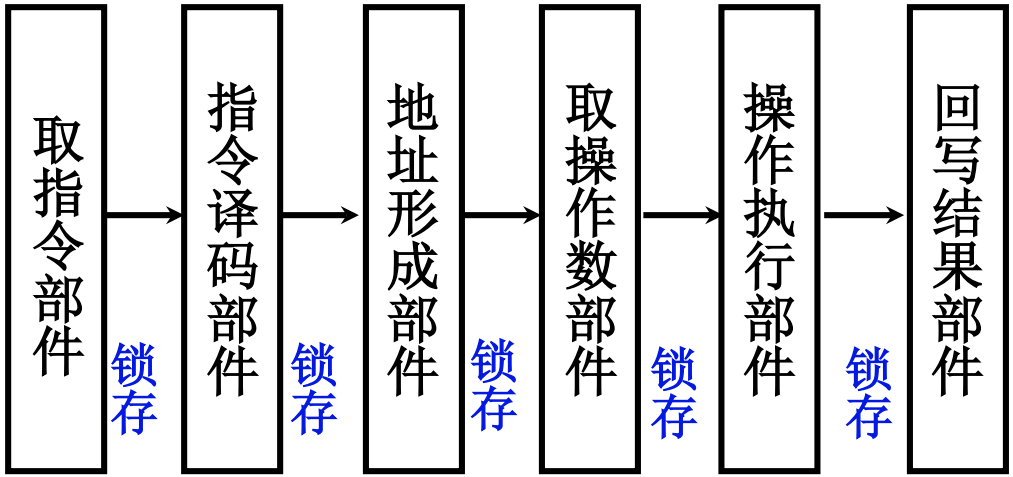
8.4 中断系统

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

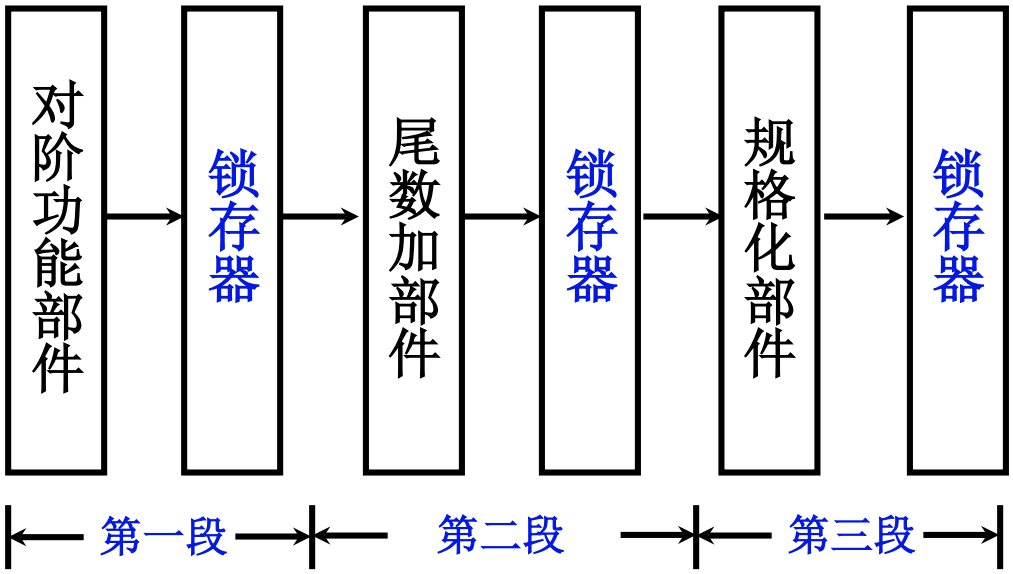
理想情况下，6 级流水 的速度是不采用流水技术的6 倍

2. 运算流水线

8.3

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

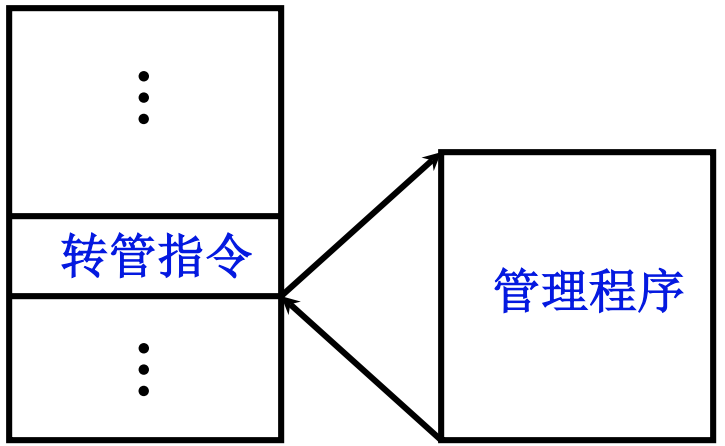
8.4 中断系统

一、概述

1. 引起中断的各种因素

(1) 人为设置的中断

如　转管指令



(2) 程序性事故　溢出、操作码不能识别、除法非法

(3) 硬件故障

(4) I/O 设备

(5) 外部事件　　用 键盘中断 现行程序

2. 中断系统需解决的问题

8.4

(1) 各中断源 如何 向 CPU 提出请求？

(2) 各中断源 同时 提出 请求 怎么办？

(3) CPU 什么 条件、什么 时间、以什么 方式 响应中断？

(4) 如何 保护现场？

(5) 如何 寻找入口地址？

(6) 如何 恢复现场，如何 返回？

(7) 处理中断的过程中又 出现新的中断 怎么办？

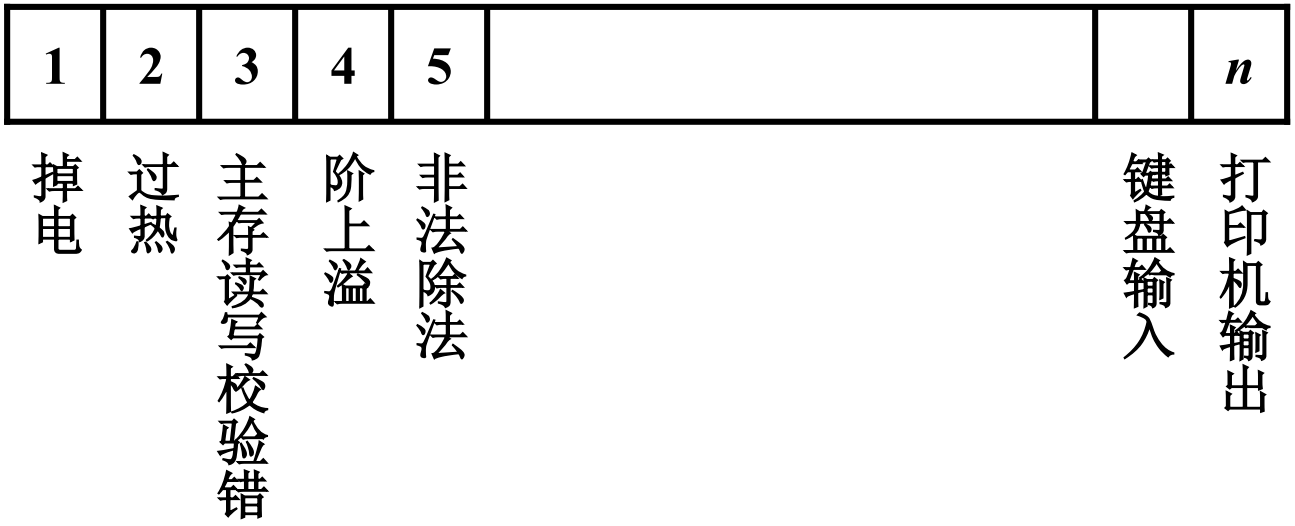
硬件 + 软件

二、中断请求标记和中断判优逻辑 8.4

1. 中断请求标记 INTR

一个请求源 一个 INTR 中断请求标记触发器

多个INTR 组成 中断请求标记寄存器



INTR 分散 在各个中断源的 接口电路中

INTR 集中 在 CPU 的中断系统 内

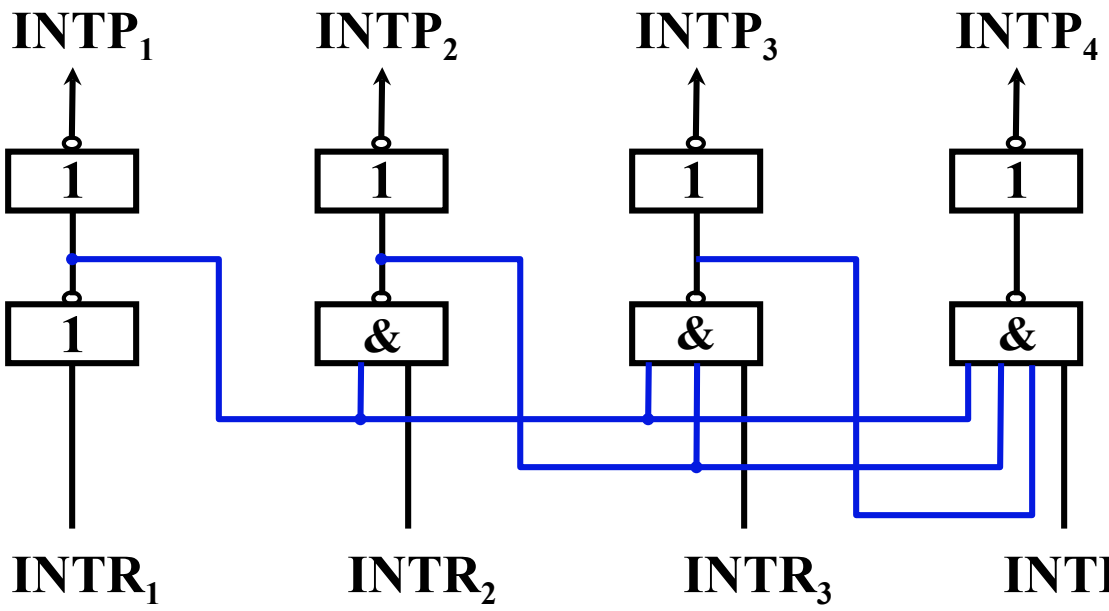
2. 中断判优逻辑

(1) 硬件实现（排队器）

① 分散 在各个中断源的 接口电路中　链式排队器

参见 第五章

② 集中 在 CPU 内

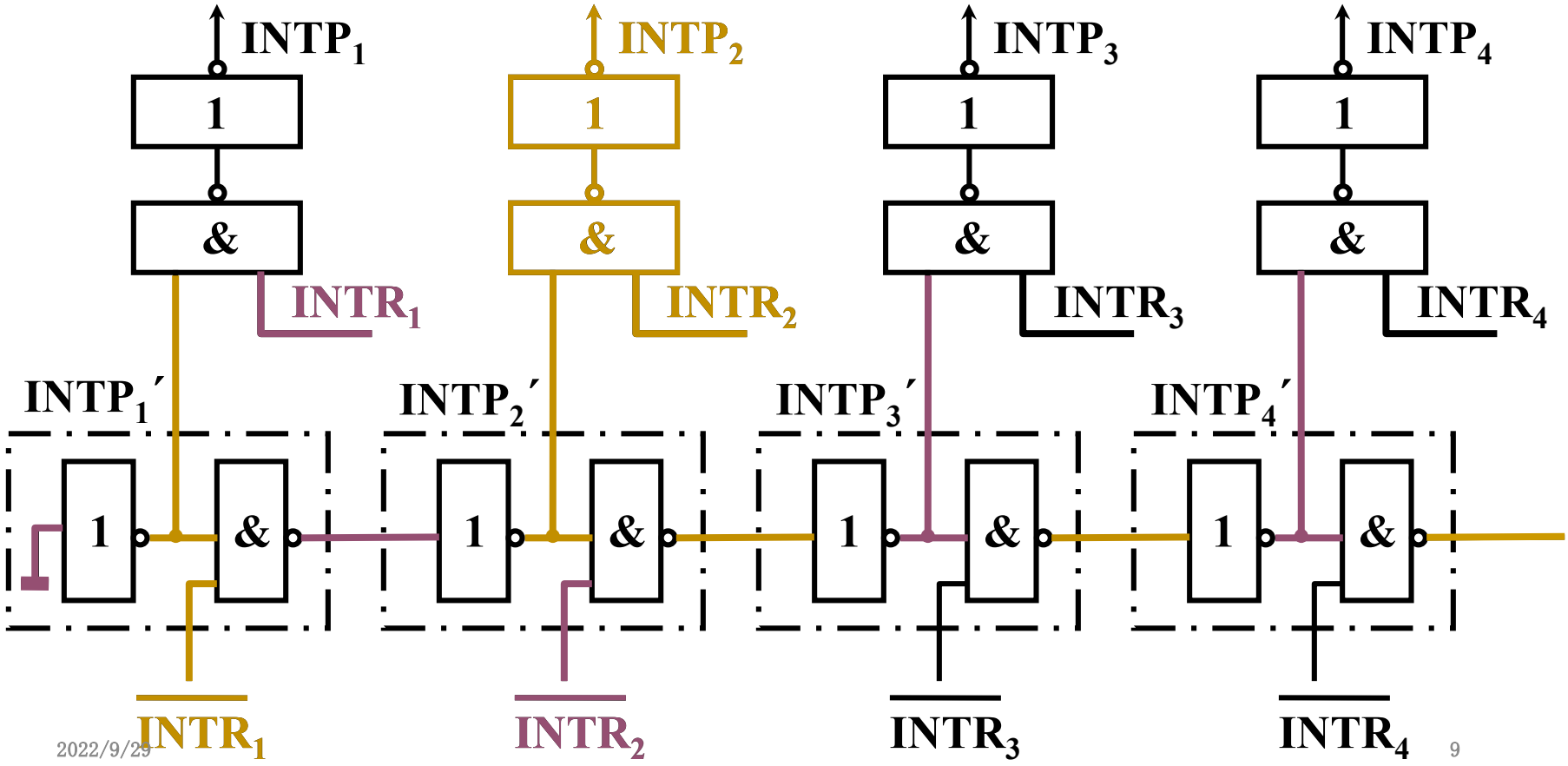


$INTR_1$、$INTR_2$、$INTR_3$、$INTR_4$ 优先级 按 降序 排列

排队器

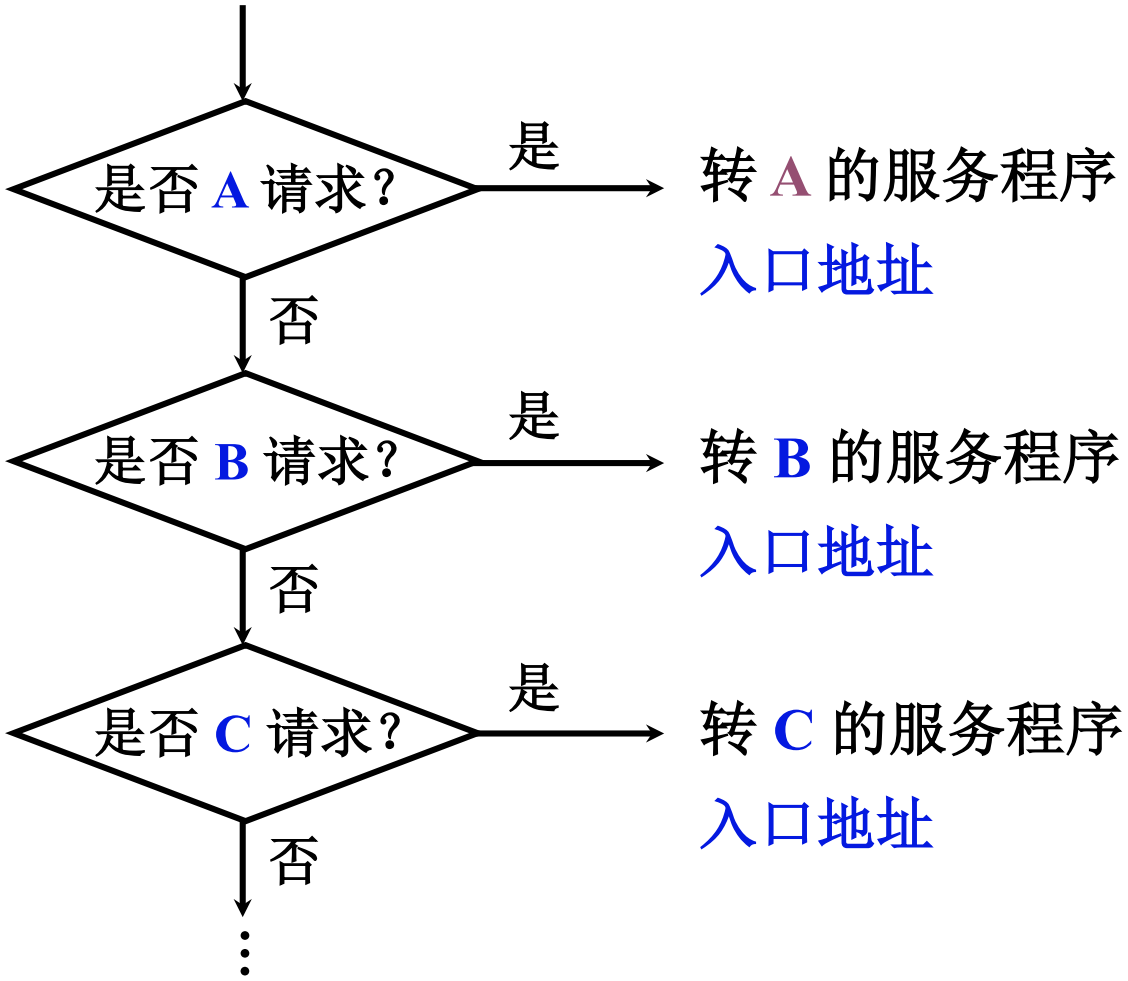
5.5

硬件 在 CPU 内或在接口电路中（链式排队器）



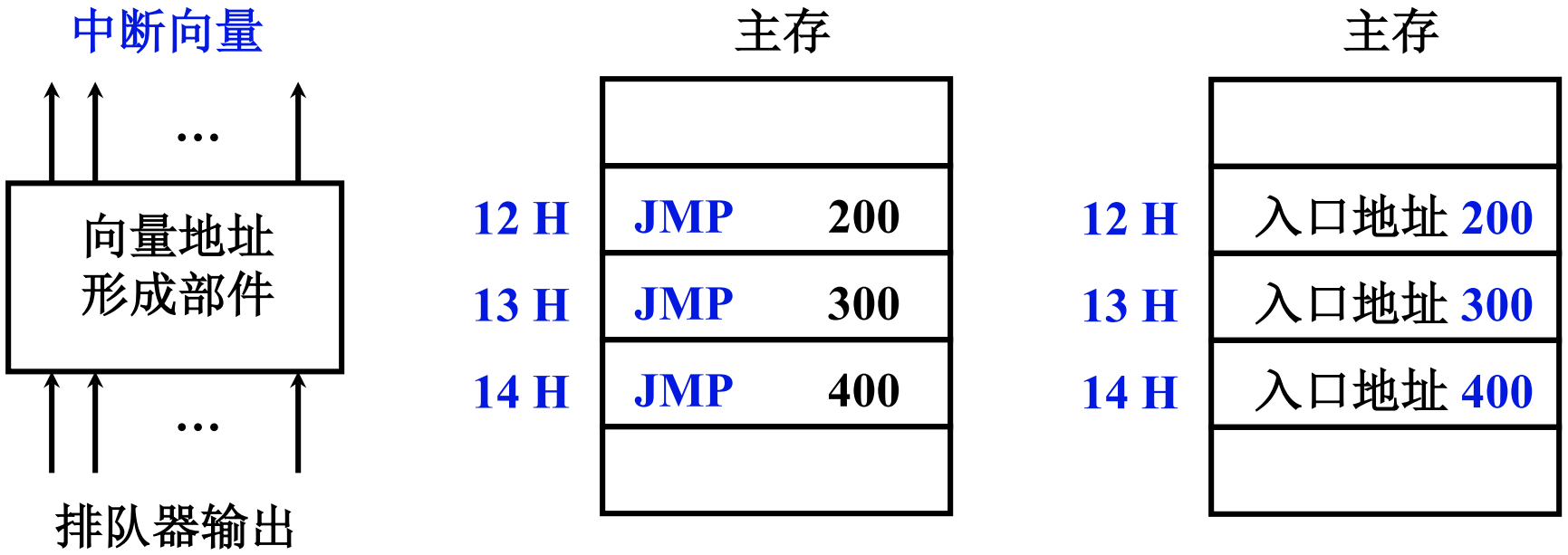
(2) 软件实现（程序查询）

A、B、C 优先级按 降序 排列



三、中断服务程序入口地址的寻找 8.4

1. 硬件向量法



向量地址　12H、13H、14H

入口地址　200、 300、 400

2. 软件查询法

8.4

八个中断源 1，2，…　8 按 降序 排列

中断识别程序（入口地址 M）

地 址	指 令	说 明
M	SKP　DZ $1^{\#}$	$1^{\#}$　D = 0 跳（D为完成触发器）
	JMP　$1^{\#}$ SR	$1^{\#}$　D = 1 转$1^{\#}$ 服务程序
	SKP　DZ $2^{\#}$	$2^{\#}$　D = 0 跳
	JMP　$2^{\#}$ SR	$2^{\#}$　D = 1 转$2^{\#}$ 服务程序
	⋮	
	SKP　DZ $8^{\#}$	$8^{\#}$　D = 0 跳
	JMP　$8^{\#}$ SR	$8^{\#}$　D = 1 转$8^{\#}$ 服务程序

四、中断响应

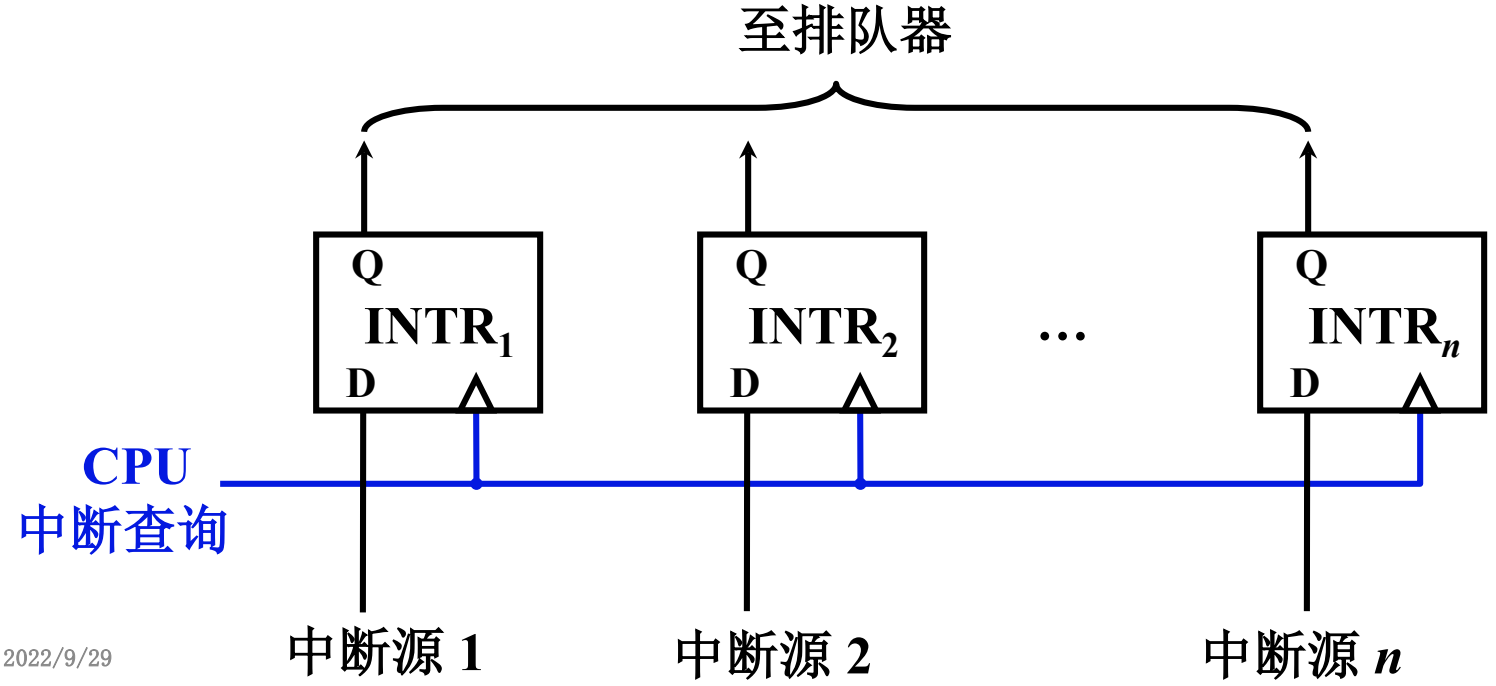
8.4

1. 响应中断的 条件

允许中断触发器　EINT = 1

2. 响应中断的 时间

指令执行周期结束时刻由CPU 发查询信号



3. 中断隐指令

8.4

(1) 保护程序断点

断点存于 特定地址（ 0 号地址） 内　　断点 进栈

(2) 寻找服务程序入口地址

向量地址 ⟶ PC　（硬件向量法）

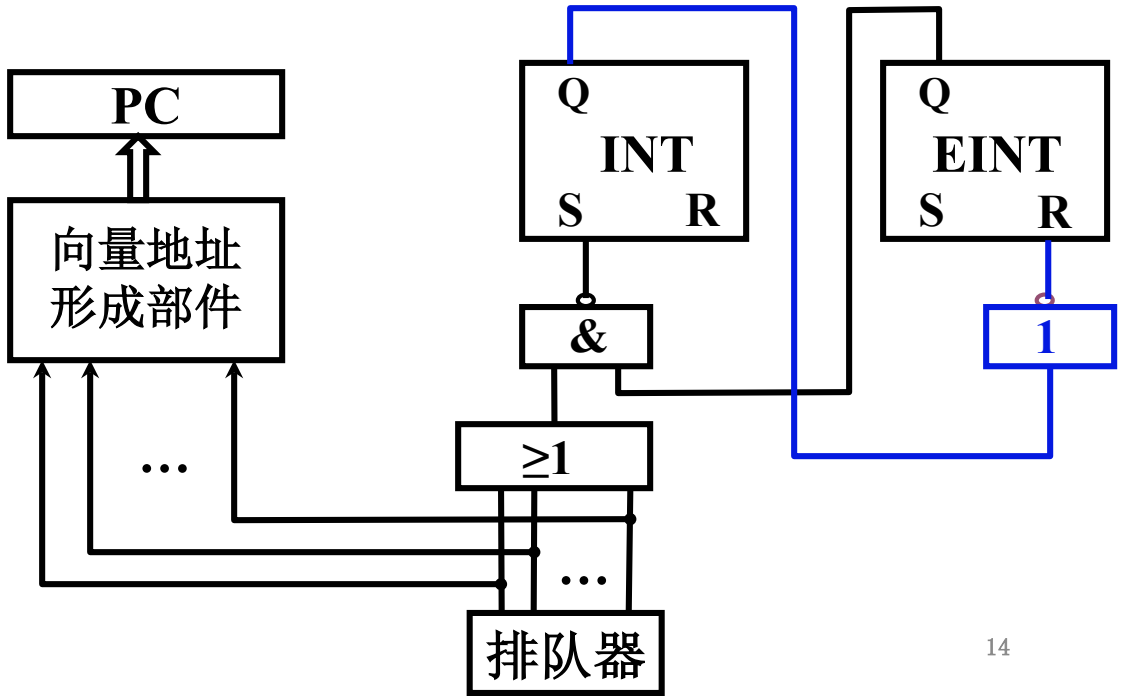
中断识别程序 入口地址 M ⟶ PC　（软件查询法）

(3) 硬件 关中断

INT 中断标记

EINT 允许中断

R – S 触发器



五、保护现场和恢复现场

8.4

1. 保护现场
   - 断点 —— 中断隐指令 完成
   - 寄存器 内容 —— 中断服务程序 完成
2. 恢复现场 —— 中断服务程序 完成

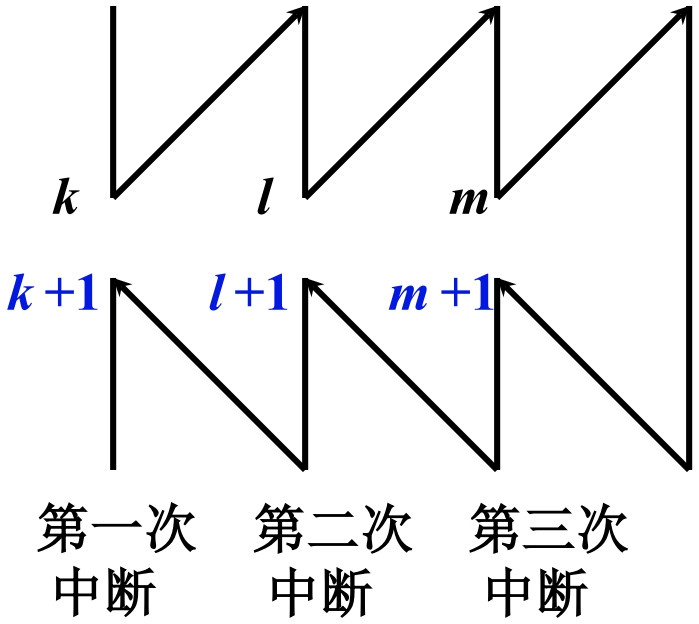
中断服务程序

保护现场	PUSH
其它服务程序	视不同请求源而定
恢复现场	POP
中断返回	IRET

六、中断屏蔽技术

8.4

1. 多重中断的概念



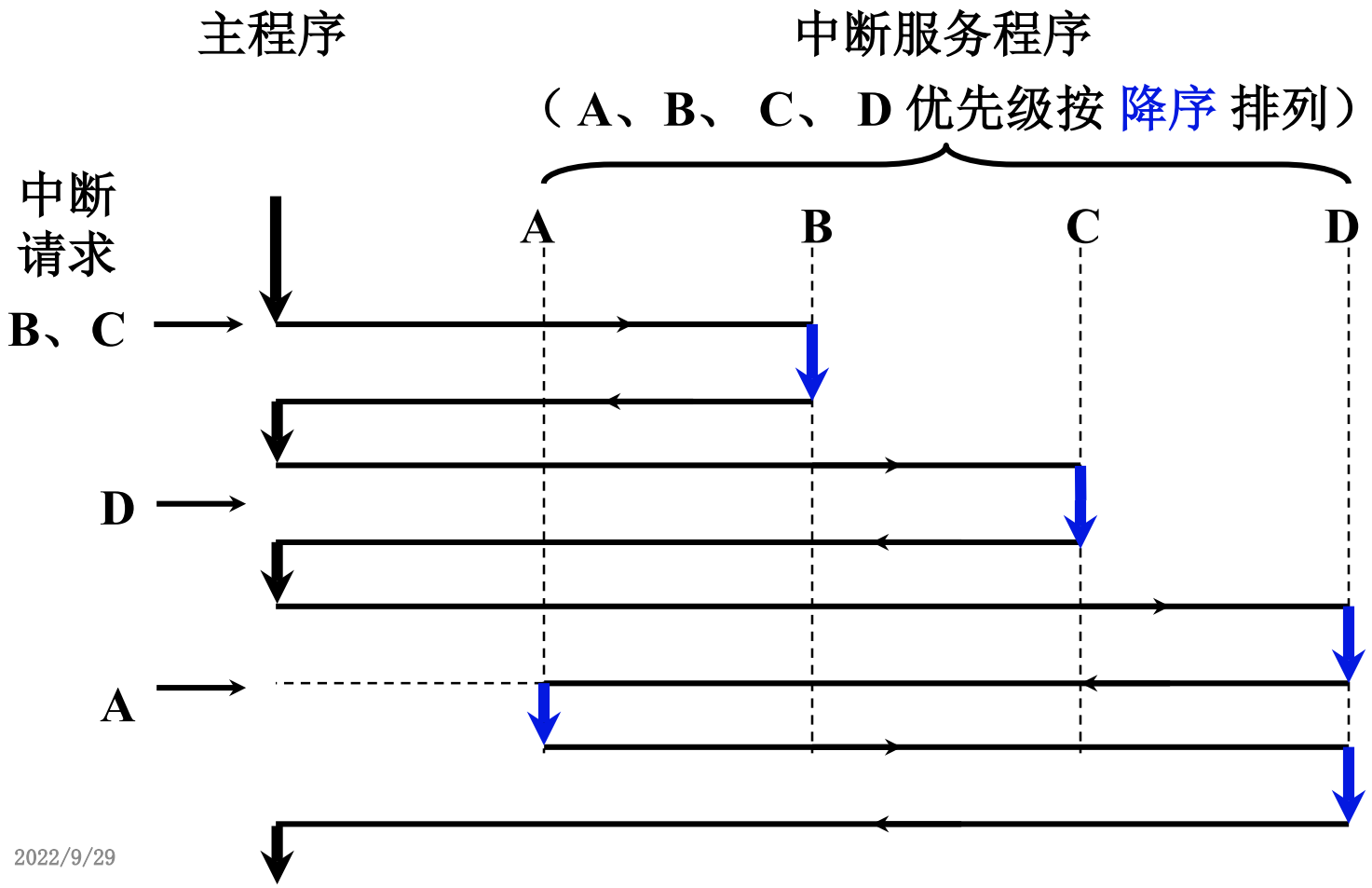
程序断点 $k+1$ ， $l+1$ ， $m+1$

2. 实现多重中断的条件

8.4

(1) 提前 设置 开中断 指令

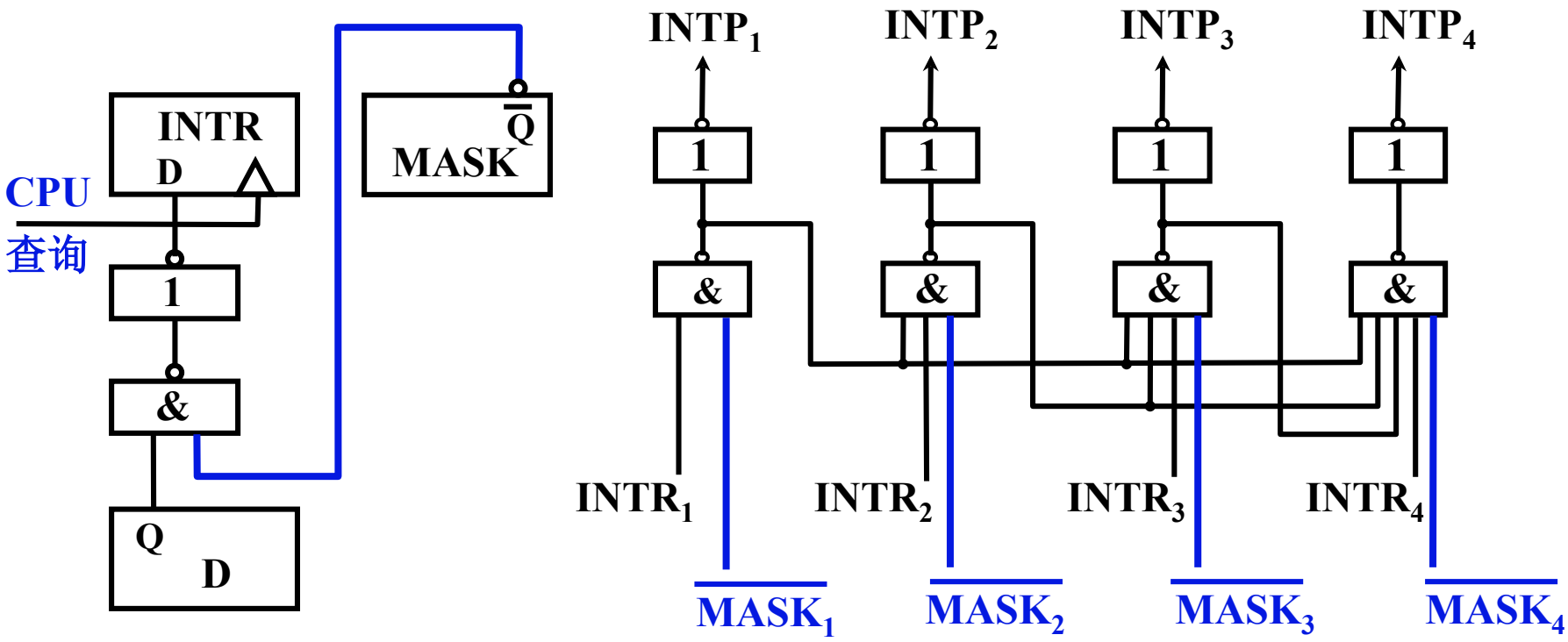
(2) 优先级别高 的中断源 有权中断优先级别低 的中断源



3. 屏蔽技术

8.4

(1) 屏蔽触发器的作用



MASK = 0（未屏蔽）　　$MASK_i$ = 1 （屏蔽）

INTR 能被置 “1”　　$INTP_i$ = 0 （不能被排队选中）

(2) 屏蔽字

16个中断源 1，2，3，… 16 按 降序 排列

优先级	屏 蔽 字
1	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
2	0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
3	0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1
4	0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1
5	0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1
6	0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1
⋮	⋮
15	0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1
16	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1

(3) 屏蔽技术可改变处理优先等级

响应优先级　　不可改变

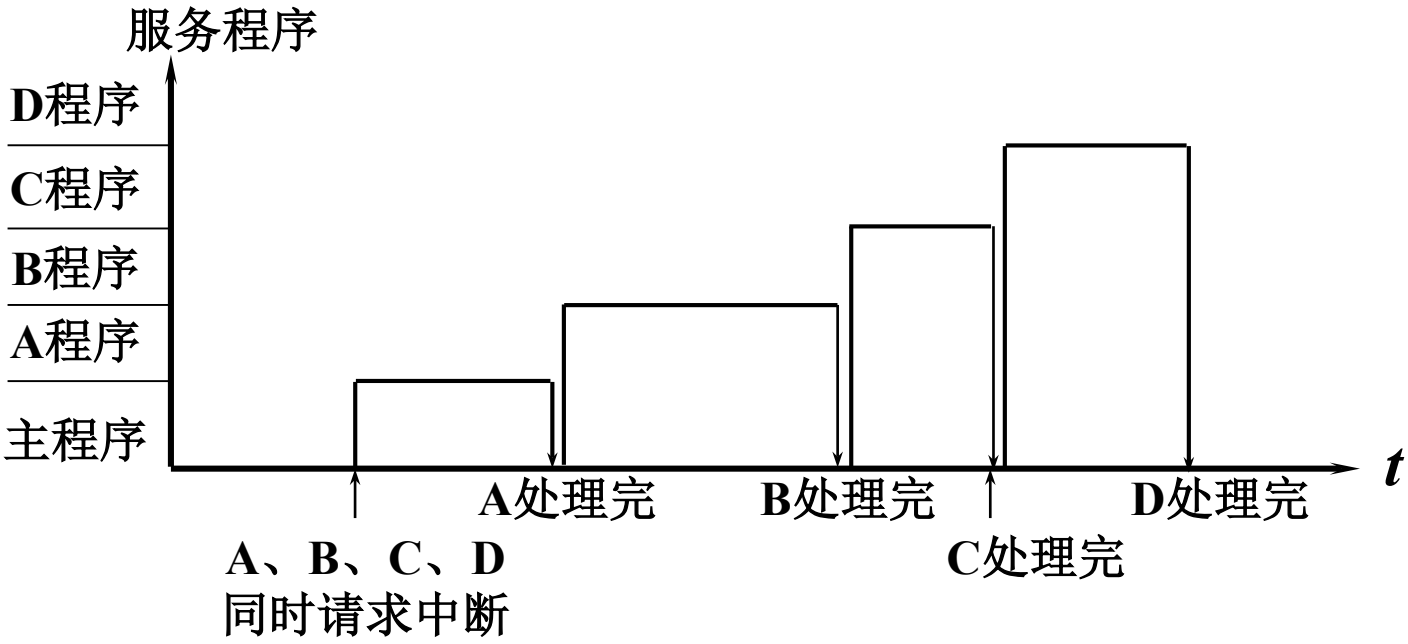
处理优先级　　可改变（通过重新设置屏蔽字）

中断源	原屏蔽字	新屏蔽字
A	1 1 1 1	1 1 1 1
B	0 1 1 1	0 1 0 0
C	0 0 1 1	0 1 1 0
D	0 0 0 1	0 1 1 1

响应优先级 **A→B→C→D** 降序排列

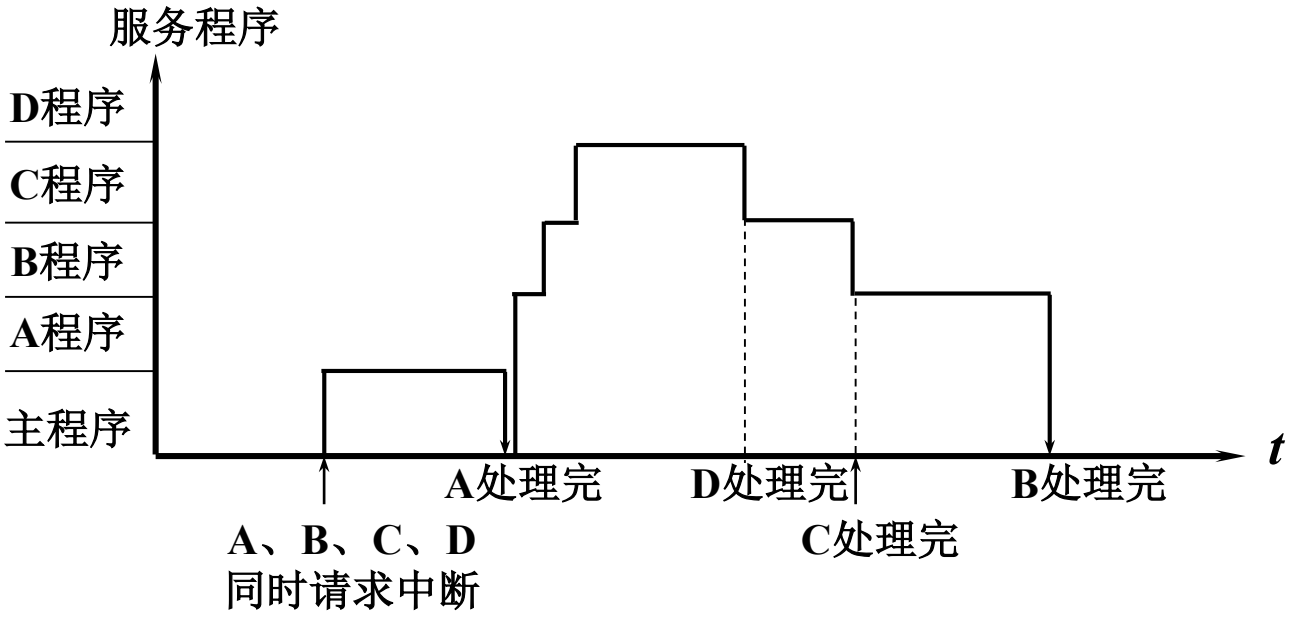
处理优先级 **A→D→C→B** 降序排列

(3) 屏蔽技术可改变处理优先等级



CPU 执行程序轨迹（原屏蔽字）

(3) 屏蔽技术可改变处理优先等级



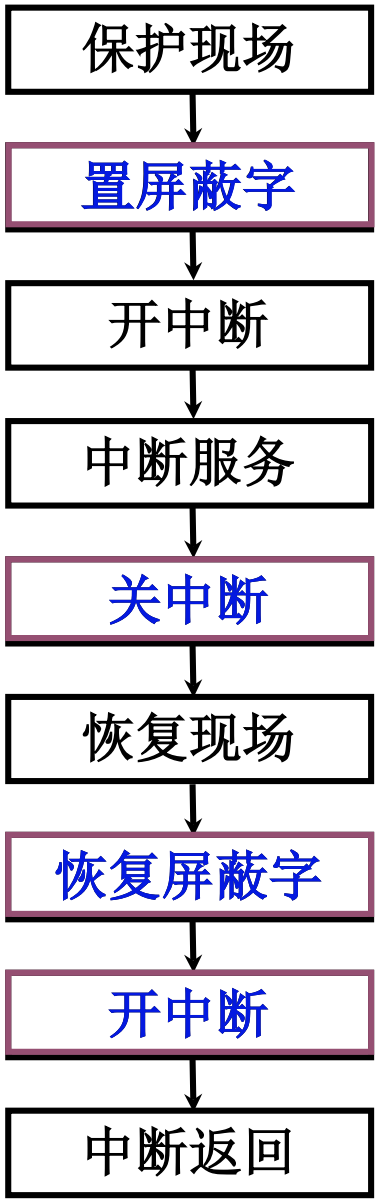
CPU 执行程序轨迹（新屏蔽字）

(4) 屏蔽技术的其他作用

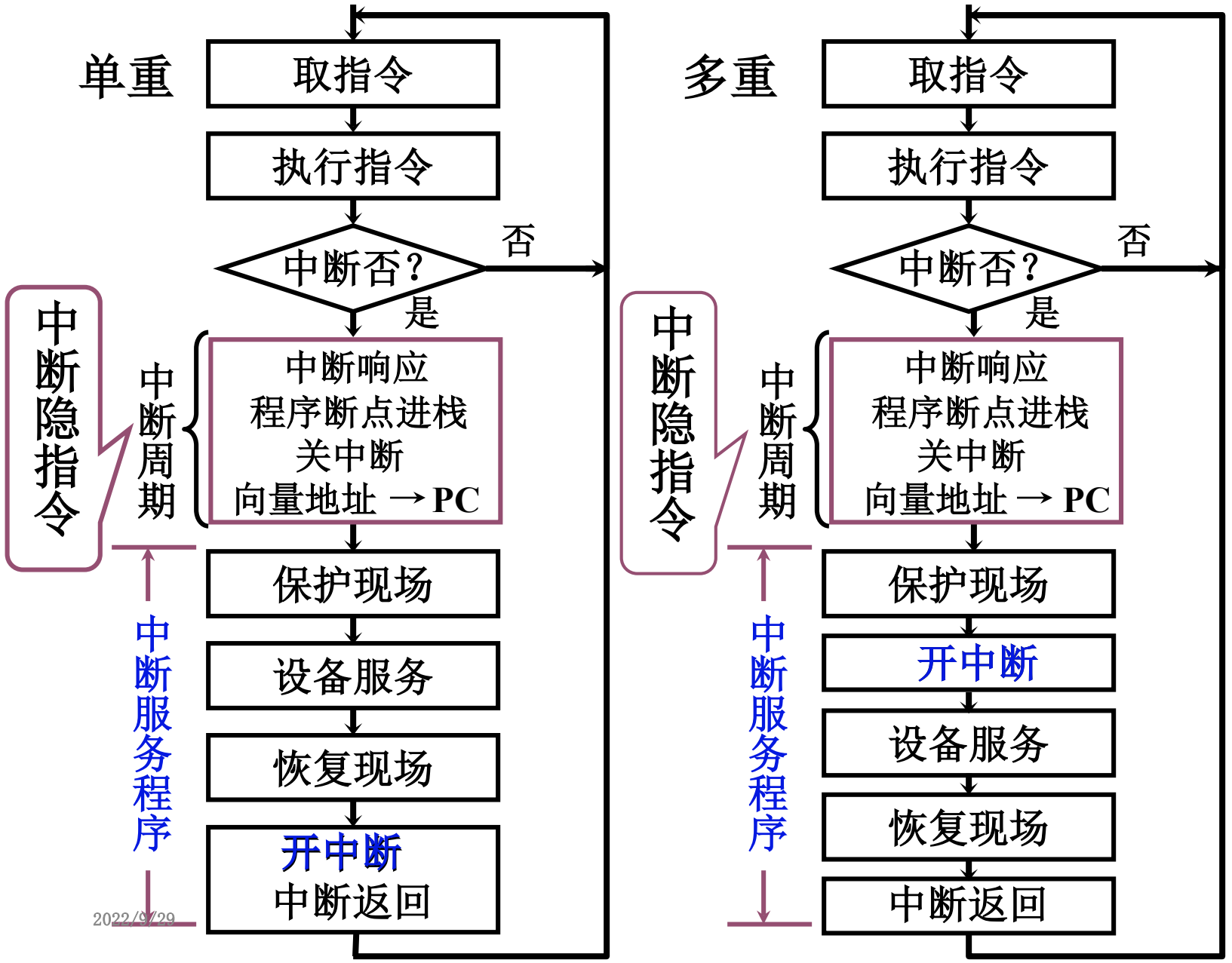
可以 人为地屏蔽 某个中断源的请求

便于程序控制

(5) 新屏蔽字的设置



单重中断和多重中断的服务程序流程



4. 多重中断的断点保护

(1) 断点进栈　　中断隐指令 完成

(2) 断点存入“ 0 ” 地址　　中断隐指令 完成

中断周期　　0 ⟶ MAR

命令存储器写

PC ⟶ MDR　　断点 ⟶ MDR

(MDR) ⟶ 存入存储器

三次中断，三个断点都存入 “ 0 ” 地址

？ 如何保证断点不丢失？

(3) 程序断点存入“0”地址的断点保护 8.4

地　址	内　　容	说　　明
0	××××	存程序断点
5	JMP　SERVE	5 为向量地址
SERVE	STA　SAVE	保护现场
	⋮	
	LDA　0	} 0 地址内容转存
置屏蔽字	STA　RETURN	
	ENI	开中断
	⋮	} 其他服务内容
	LDA　SAVE	恢复现场
	JMP　@　RETURN	间址返回
SAVE	××××	存放 ACC 内容
RETURN	××××	转存 0 地址内容