计算机组成原理

翁睿

哈尔滨工业大学

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

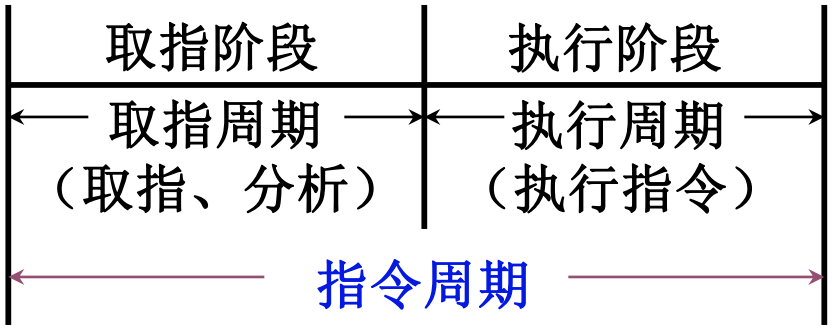
8.2 指 令 周 期

一、 指令周期的基本概念

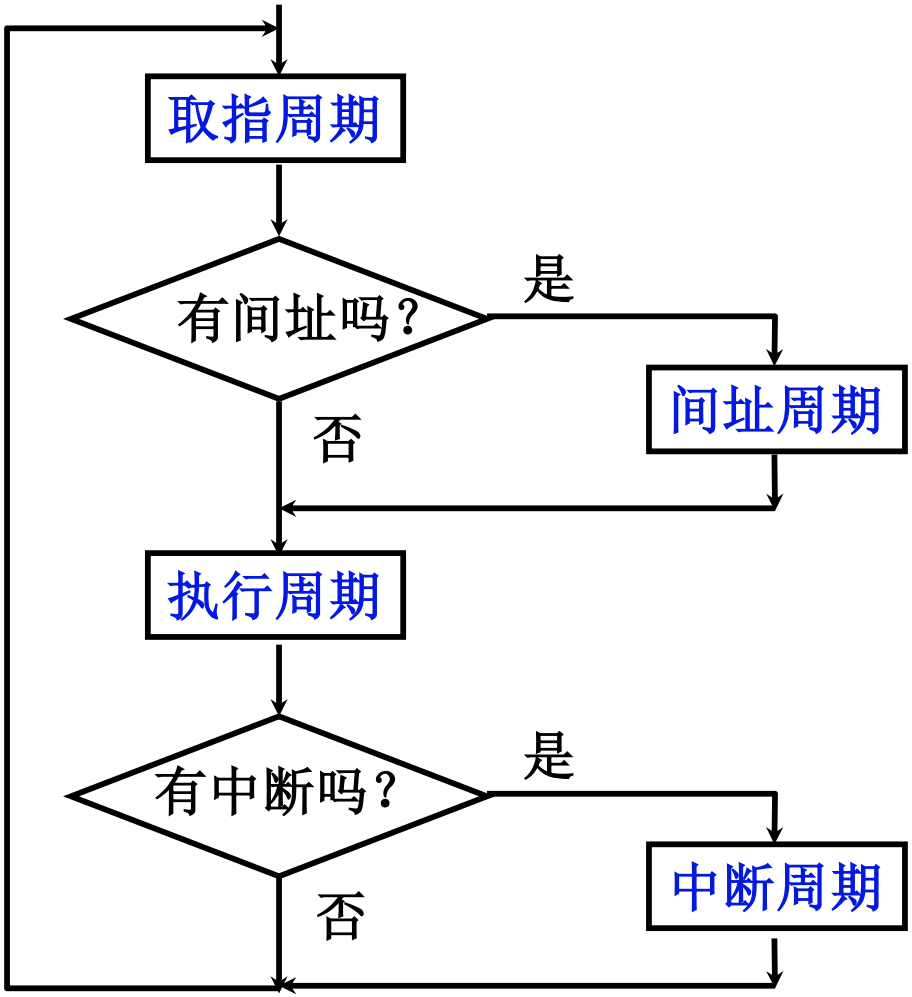
1．指令周期

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



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

基于此：

划分CPU 的

4个工作周期

每个工作周期的数据流向，需要记住！

8.3 指令流水

一、如何提高机器速度

高速器件　改进系统结构，开发系统的并行性

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

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

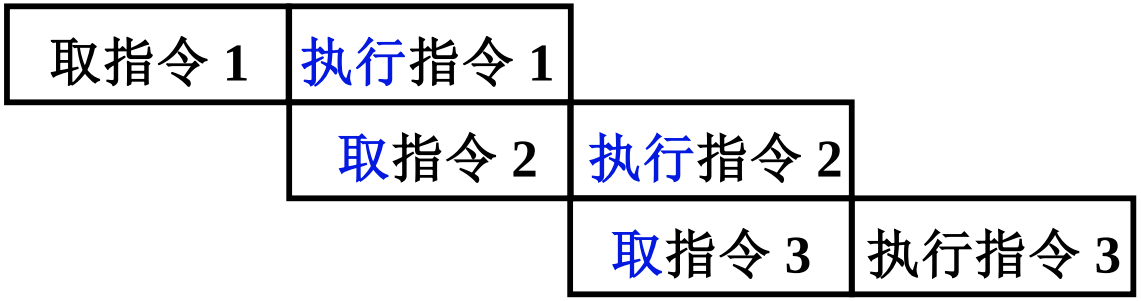
8.3

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

(1) 执行时间 > 取指时间

执行过程的后半部分，取址部件空闲

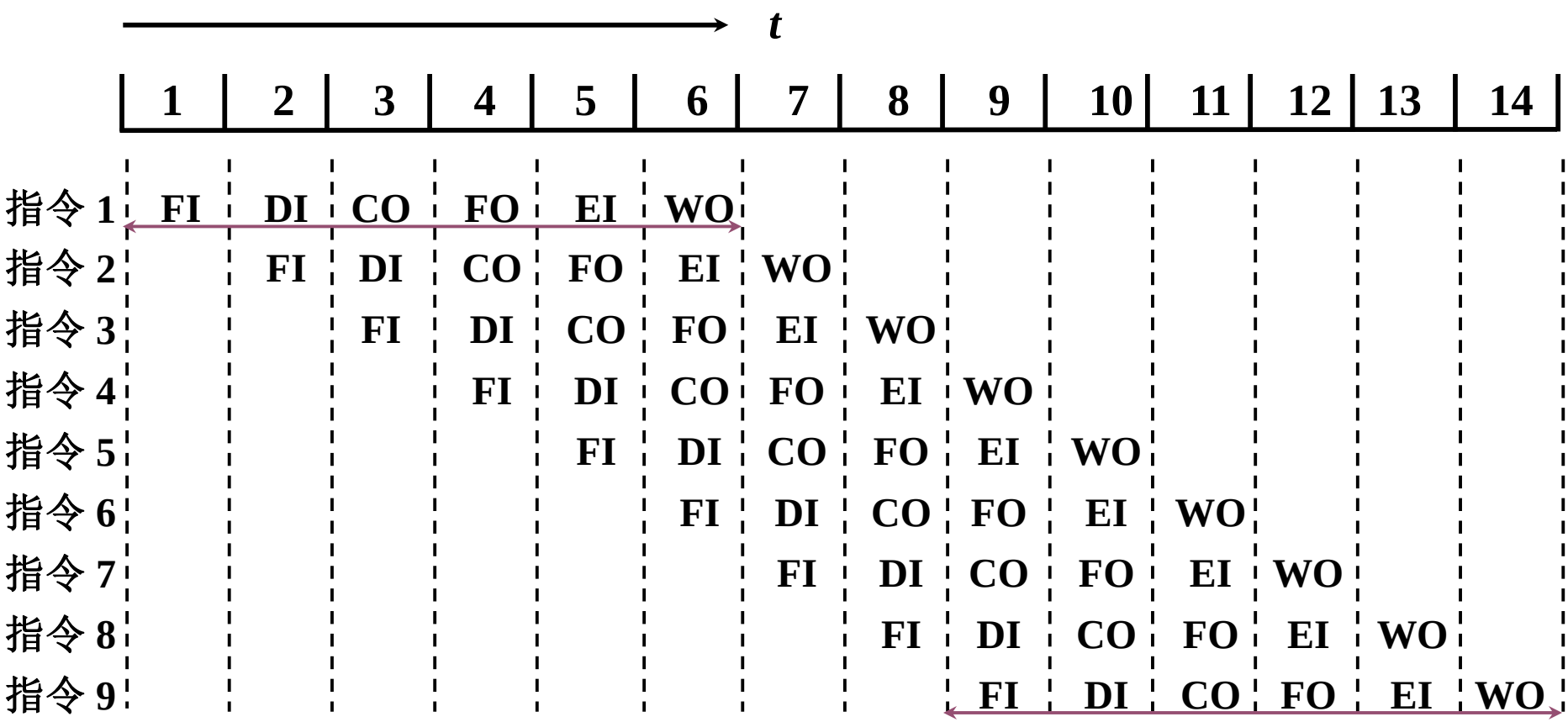
(2) 条件转移指令 对指令流水的影响

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

解决办法 ？ 猜测法

4. 指令的六级流水

8.3



完成 一条指令 6 个时间单位

串行执行 6 × 9 = 54 个时间单位

六级流水 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
- 停顿等待
- 指令存储器和数据存储器分开
- 指令预取技术 （适用于访存周期短的情况）

2. 数据相关　不同指令因重叠操作，可能改变操作数的实际 读/写 访问顺序

解决办法
- 后推法
- 采用 旁路技术（定向技术）

3. 控制相关　由转移指令引起

解决办法
- 猜测法，未猜中则产生转移损失

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

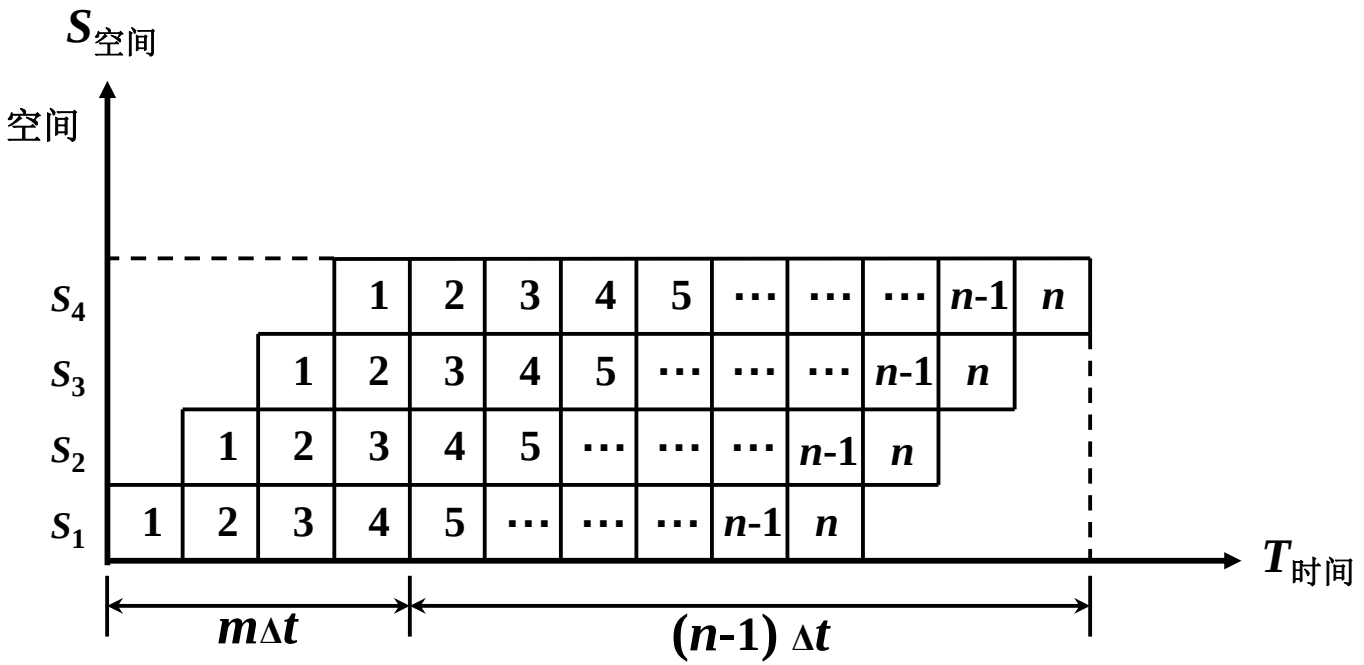
$$S_p = \frac{nm \cdot \Delta t}{m \cdot \Delta t + (n-1) \cdot \Delta t} = \frac{nm}{m + n - 1}$$

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



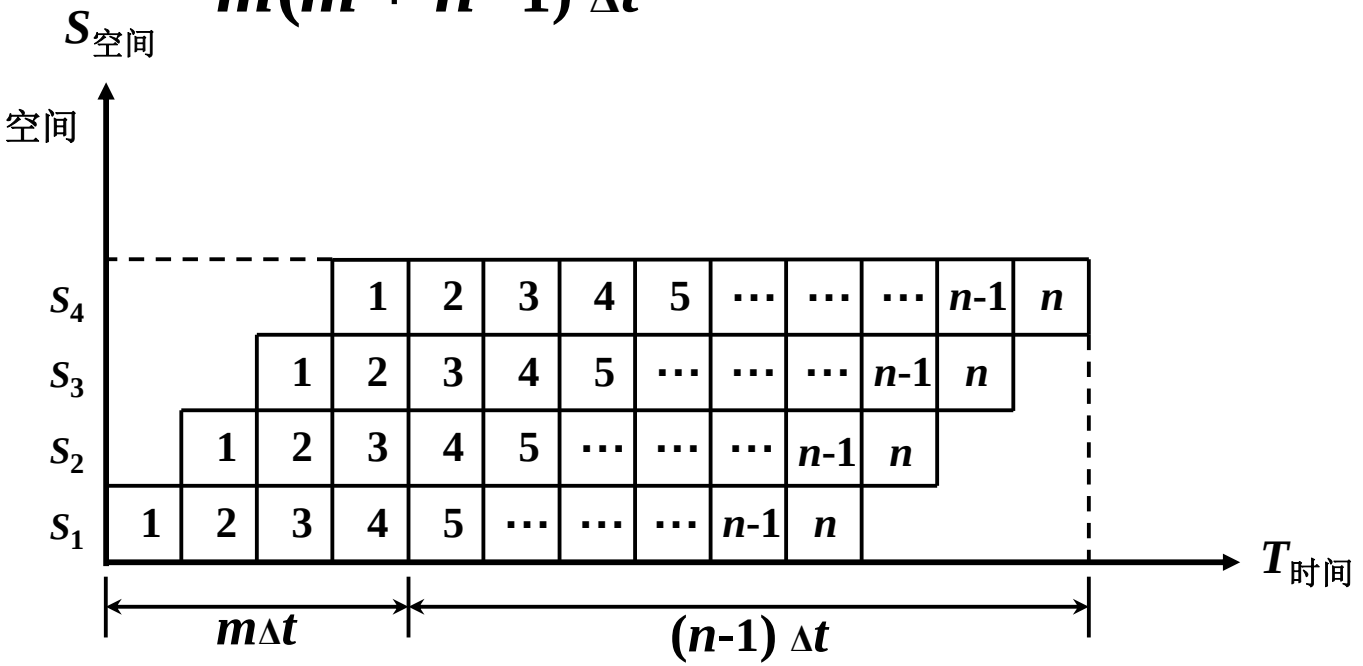
3. 效率

8.3

流水线中各功能段的 利用率

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

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



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
     需配置多个功能部件

2. 超流水线技术
   - 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）

3. 超长指令字技术
   - 编译程序将 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

2. 运算流水线

共性特点：按执行流程进行分段处理

在段与段之间插入锁存器

分段原则：每段 操作时间 尽量 一致

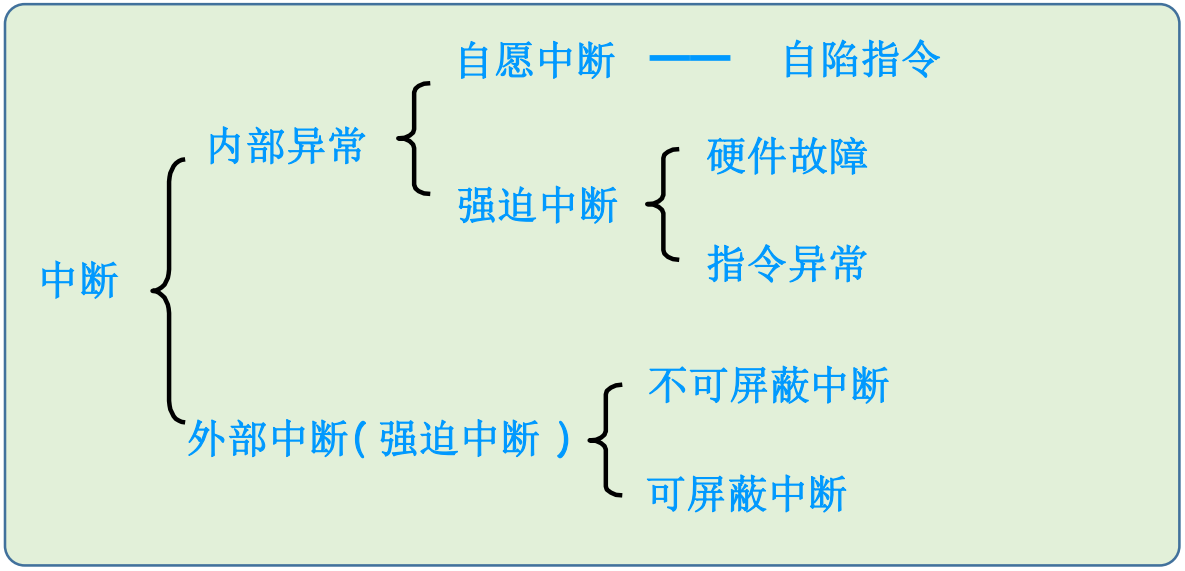
8.4 中断系统

8.4

一、概述

中断的分类与作用

- 中断技术赋于计算机应变能力，将有序的运行和无序的事件统一起来，大大增强了系统的处理能力
  - 主机外设并行工作
  - 程序调试
  - 故障处理
  - 实时处理
  - 人机交互

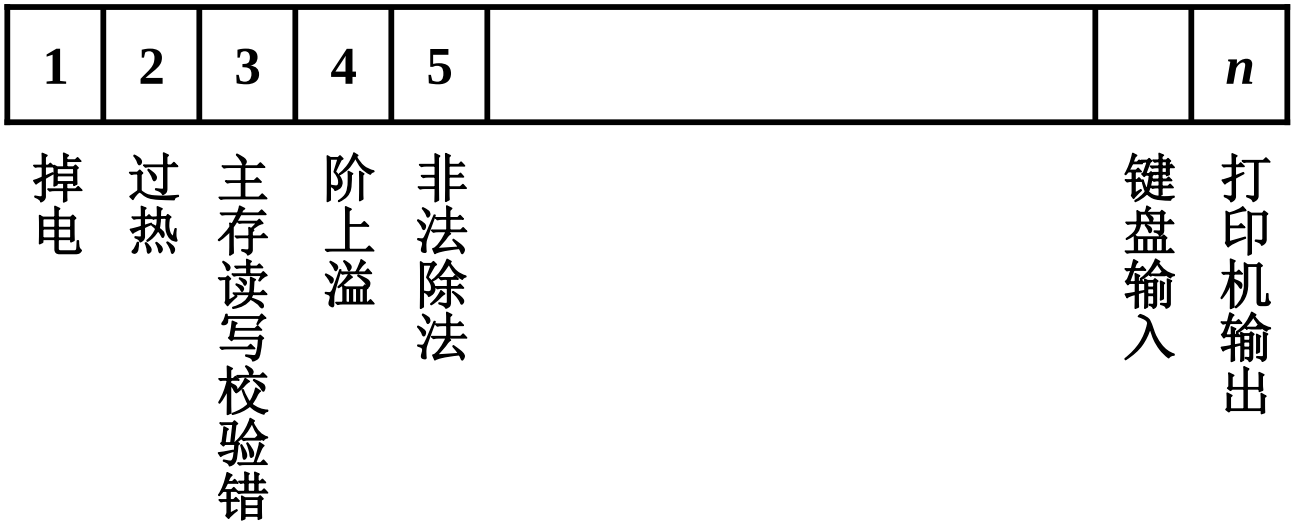


二、中断请求标记和中断判优逻辑 8.4

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

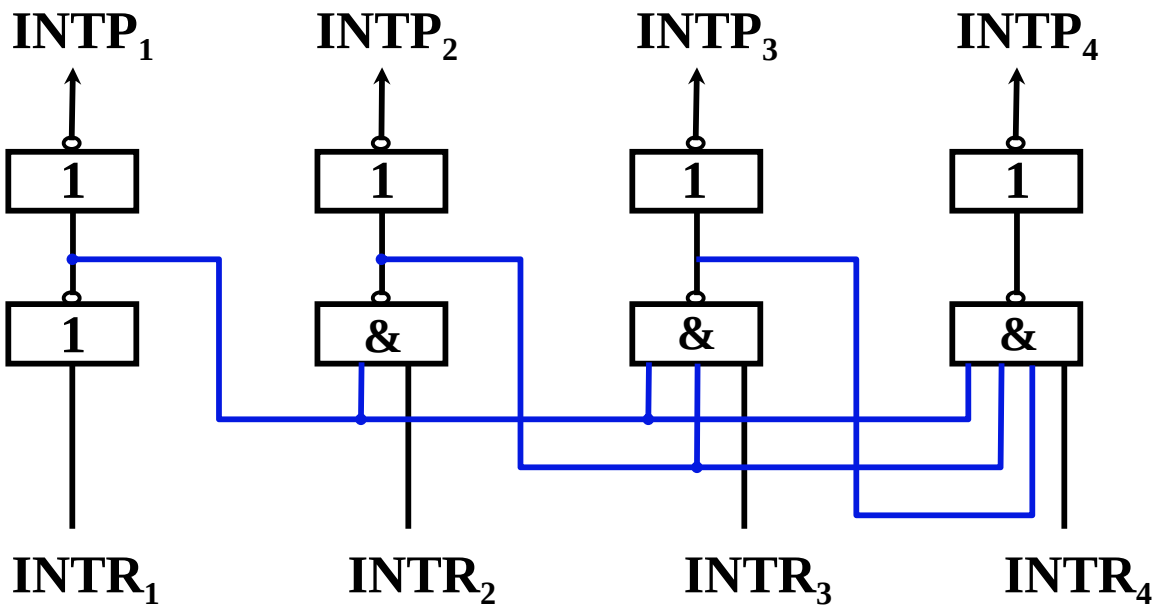
2. 中断判优逻辑

(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

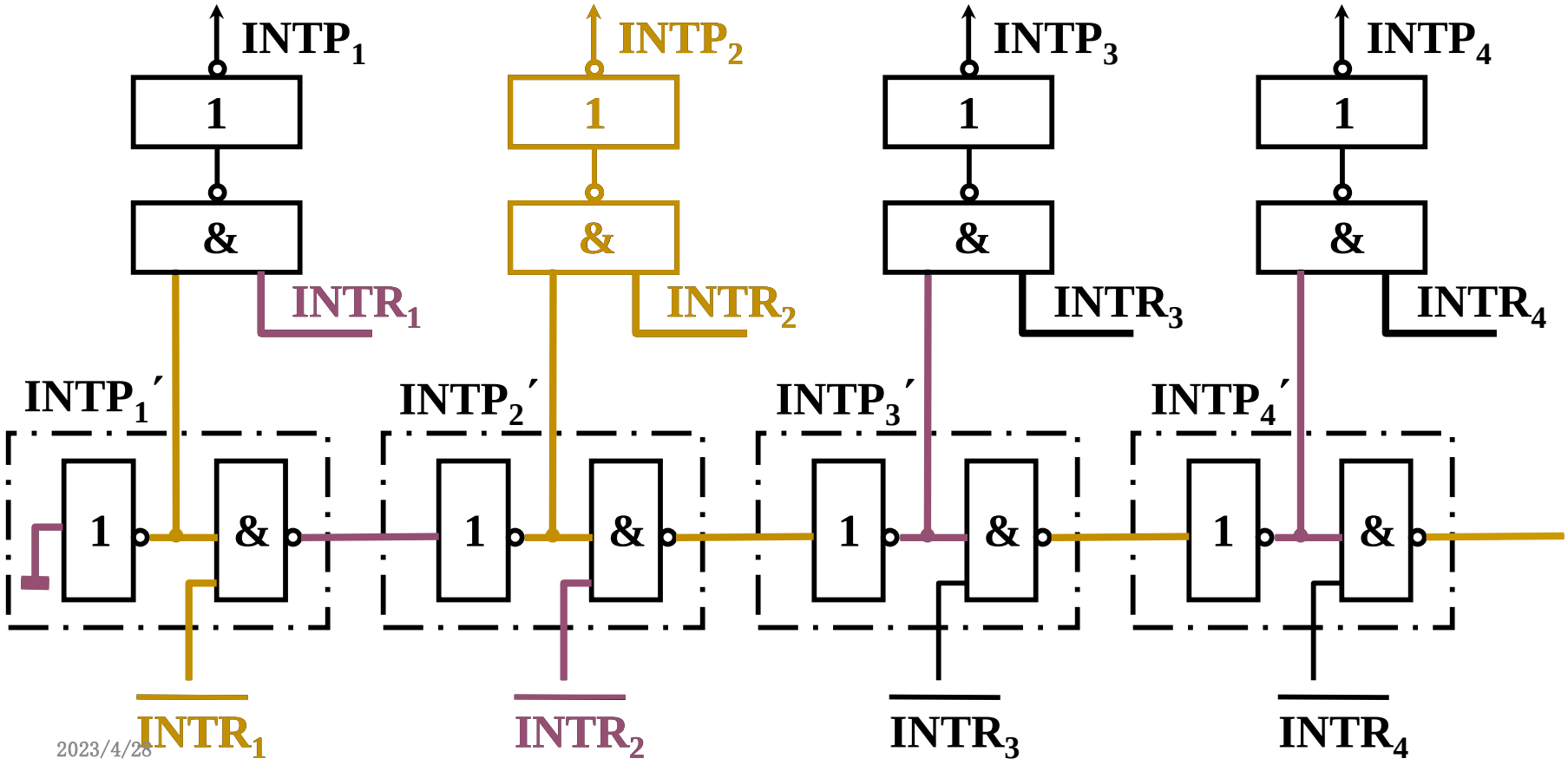


$INTR_1$、$INTR_2$、$INTR_3$、$INTR_4$ 优先级 按 降序 排列

排队器

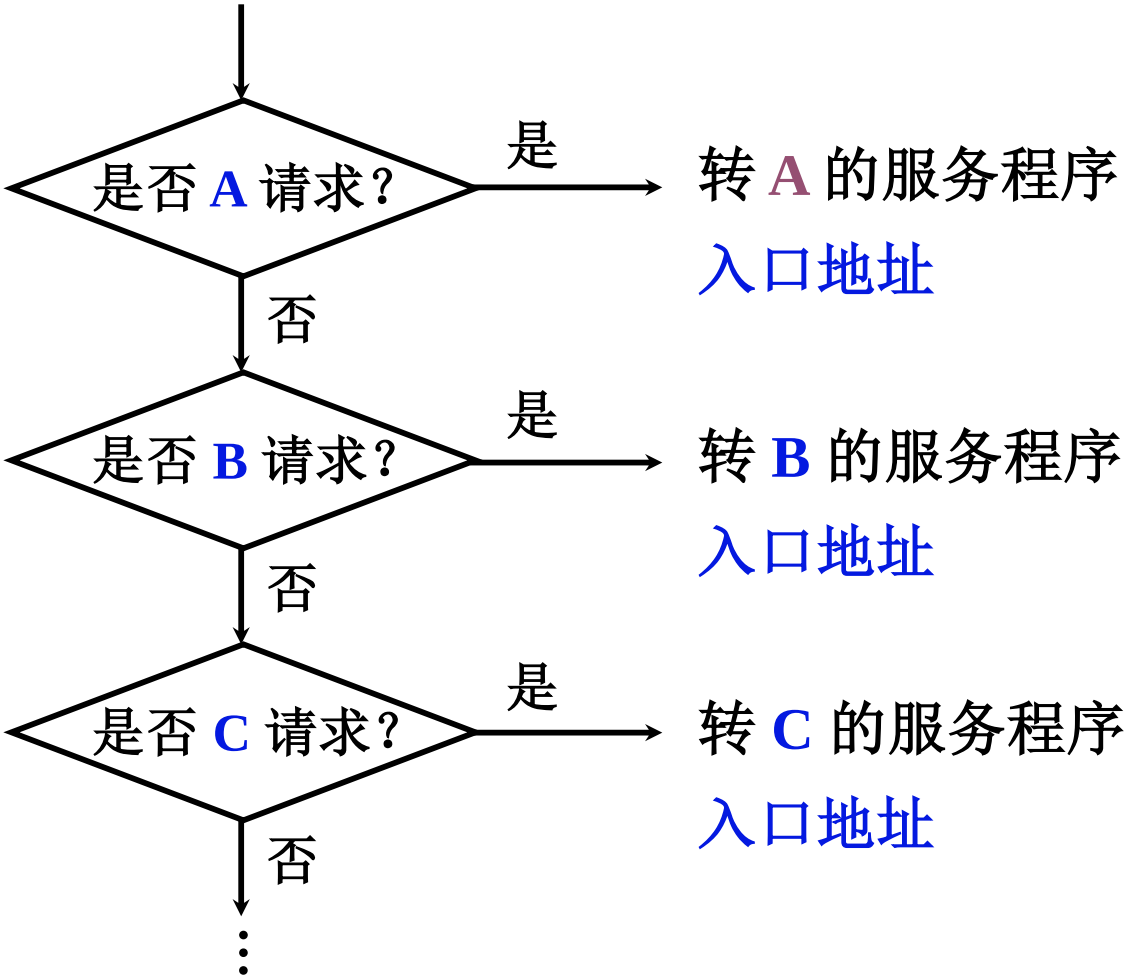
5.5

硬件 在 CPU 内或在接口电路中（链式排队器）



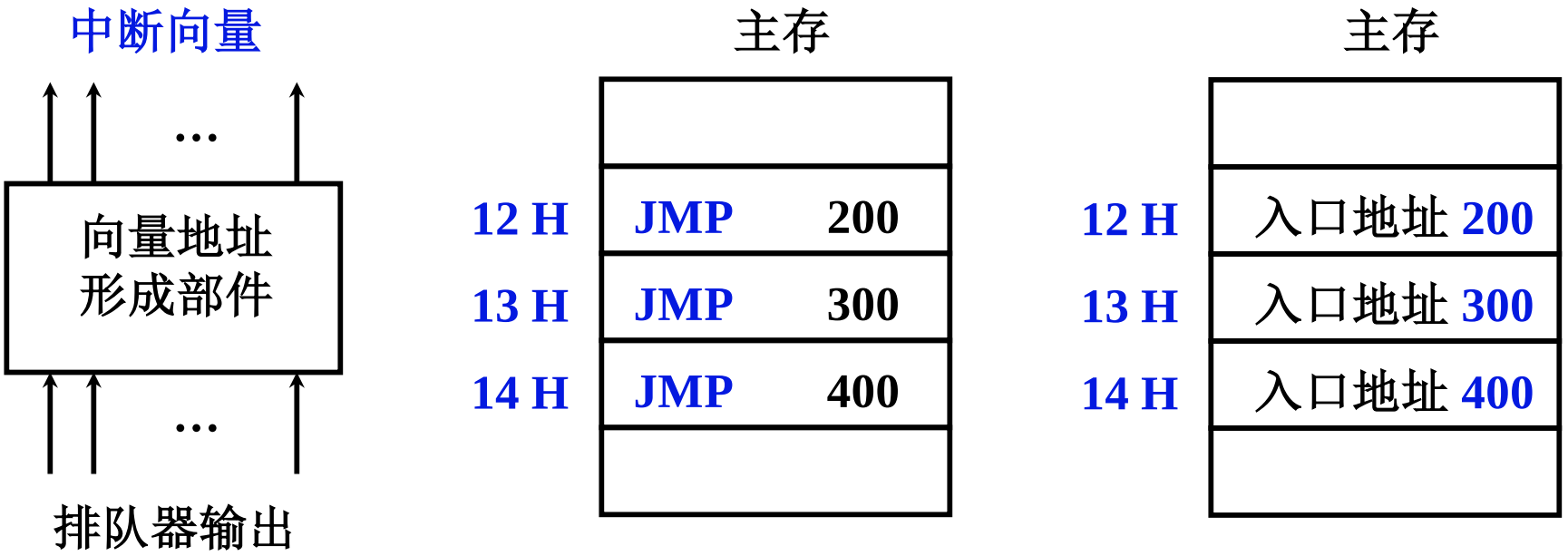
(2) 软件实现（由中断识别程序查询）

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

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

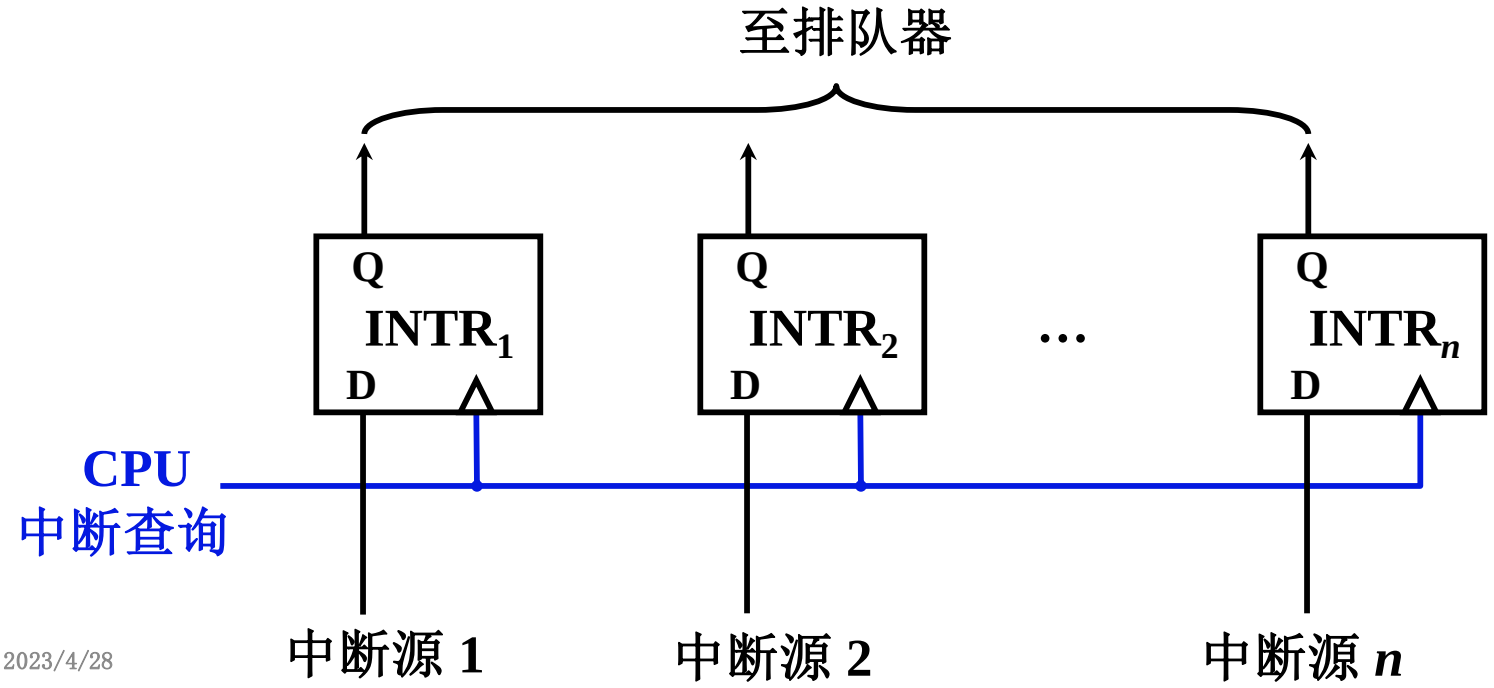
四、中断响应

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

(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）

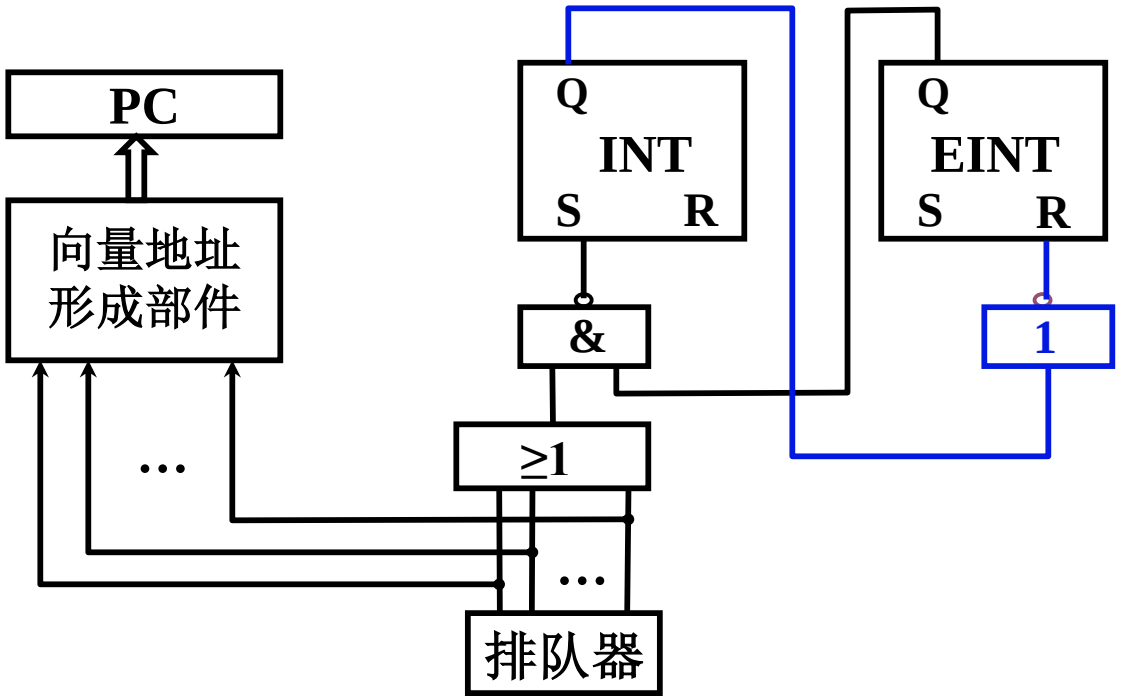
中断识别程序 入口地址 M ⟶ PC　（软件查询法）

(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

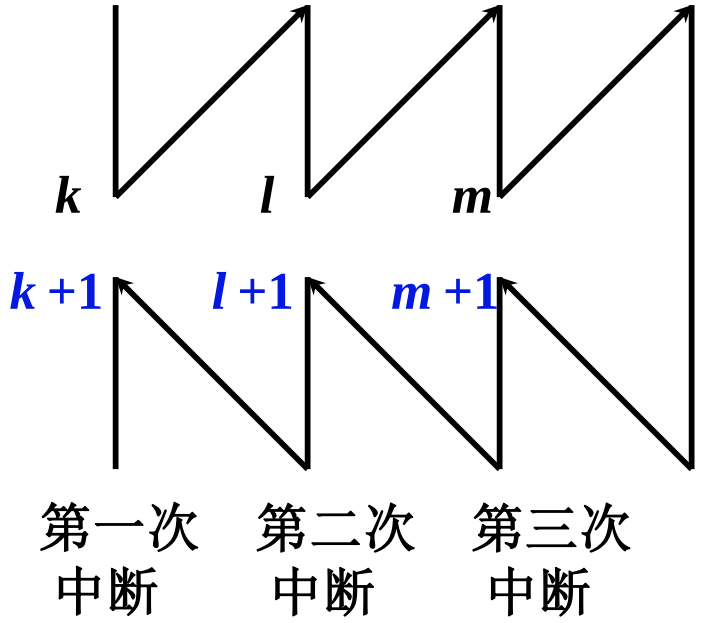
保护现场	PUSH
其它服务程序	视不同请求源而定
恢复现场	POP
中断返回	IRET

六、中断屏蔽技术

8.4

1. 多重中断的概念

- 高优先级中断请求能否中断运行中的程序呢？
- 系统硬件、软件开销的权衡
  - ✓ 单重中断
    - 所有中断源均属同一级，离CPU近的优先级高
    - CPU处理某个中断时，不响应其他中断
  - ✓ 多重中断
    - 优先级高的中断可以打断优先级低的中断服务程序
    - 中断嵌套



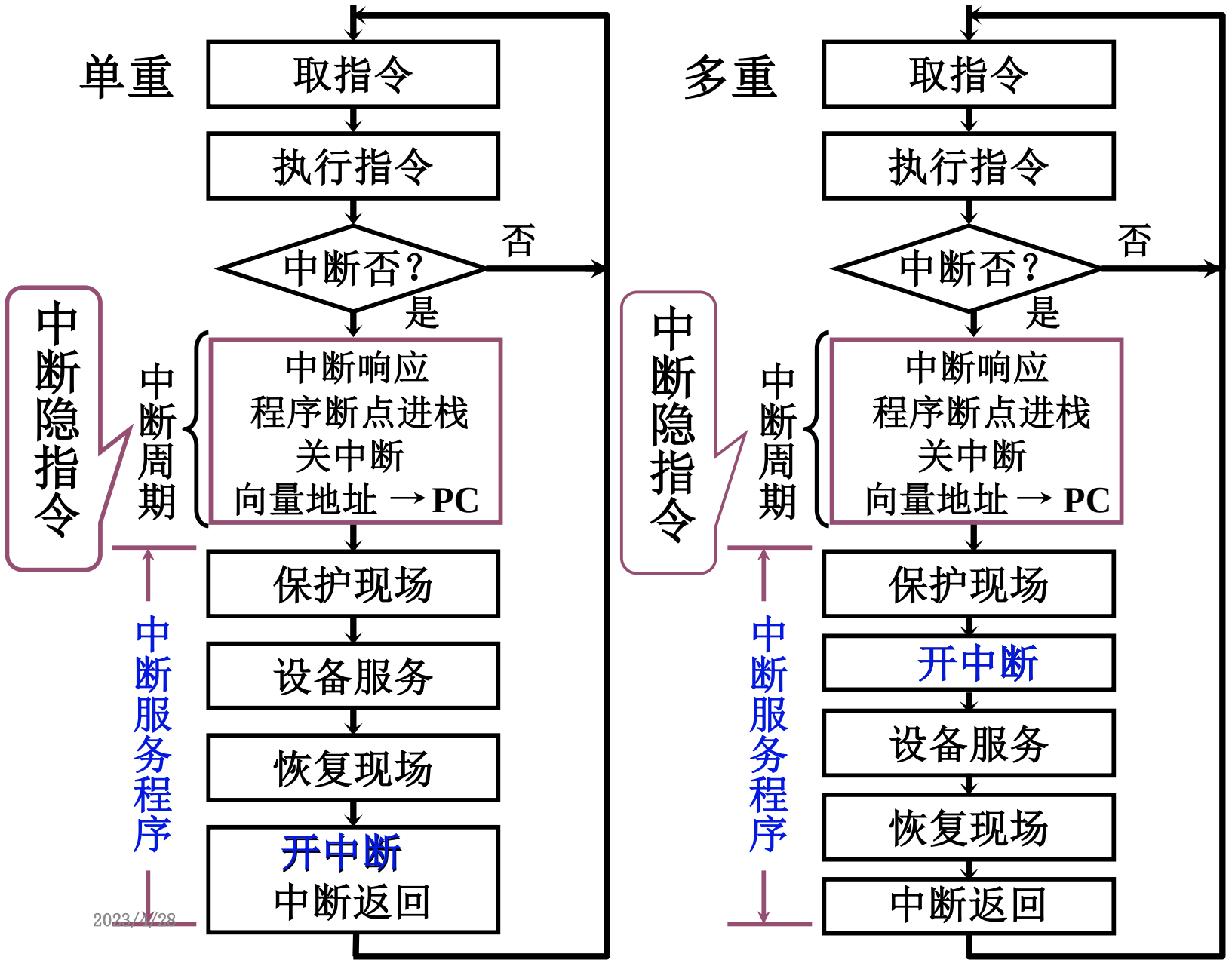
程序断点 $k+1$，$l+1$，$m+1$

划分优先级的一般规律

8.4

- 硬件故障中断属于最高级
- 其次是程序错误中断
- 非屏蔽中断优于可屏蔽中断
- DMA请求优先于I/O设备传送的中断请求
- 高速设备优于低速设备
- 输入设备的中断优于输出设备
- 实时设备优先于普通设备

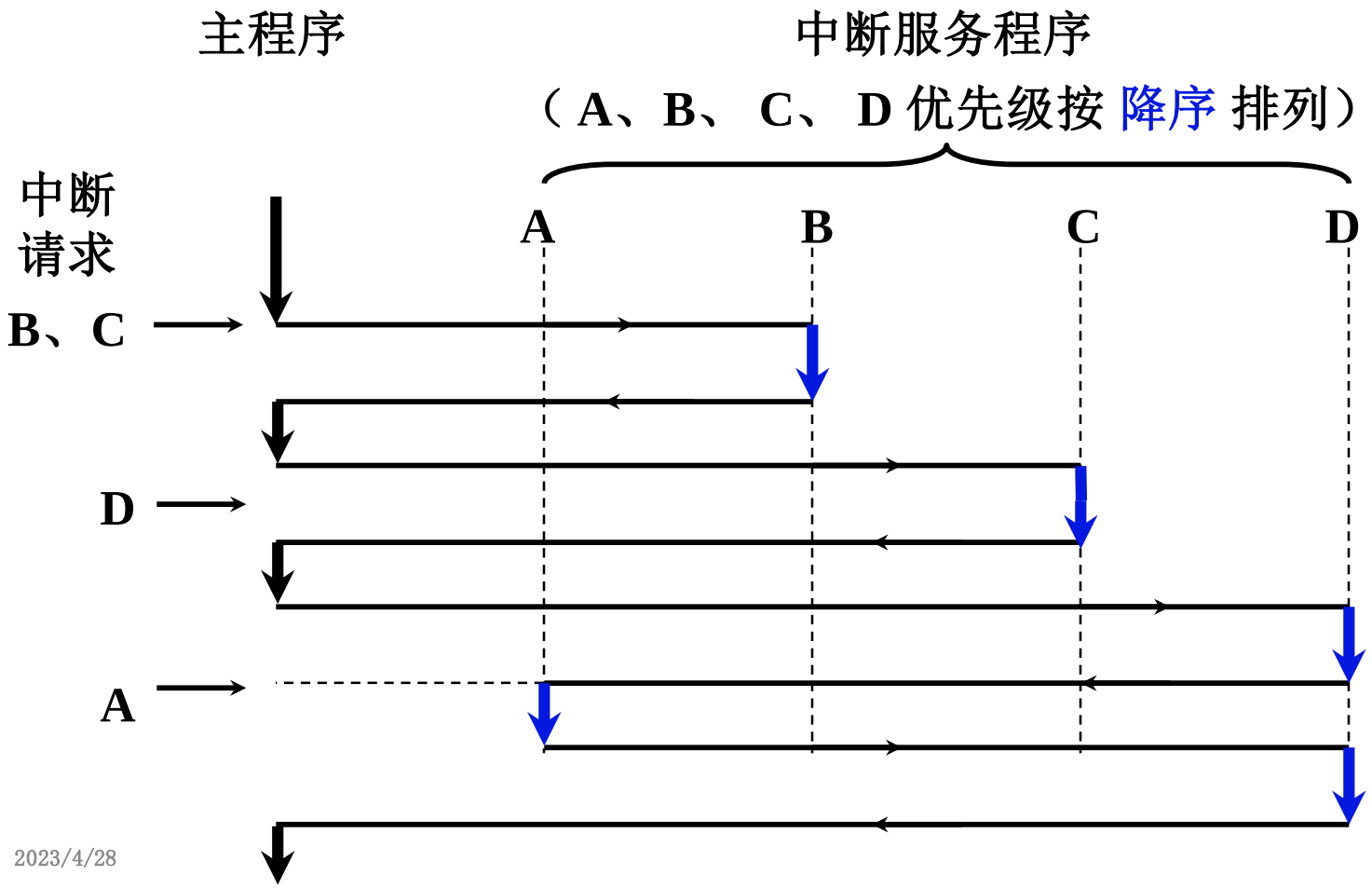
单重中断和多重中断的服务程序流程



2. 实现多重中断的条件

(1) 提前 设置 开中断 指令

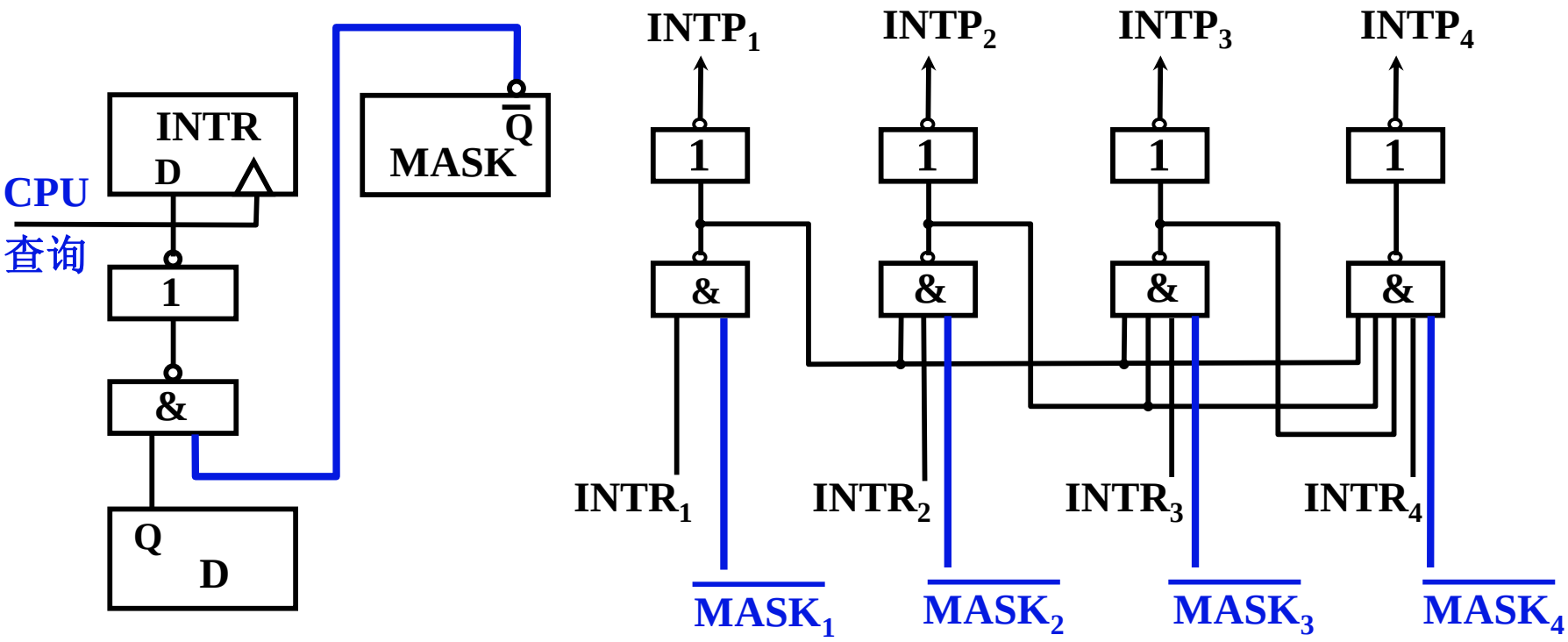
(2) 优先级别高 的中断源 有权中断优先级别低 的中断源



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

假设

响应优先级　　不可改变

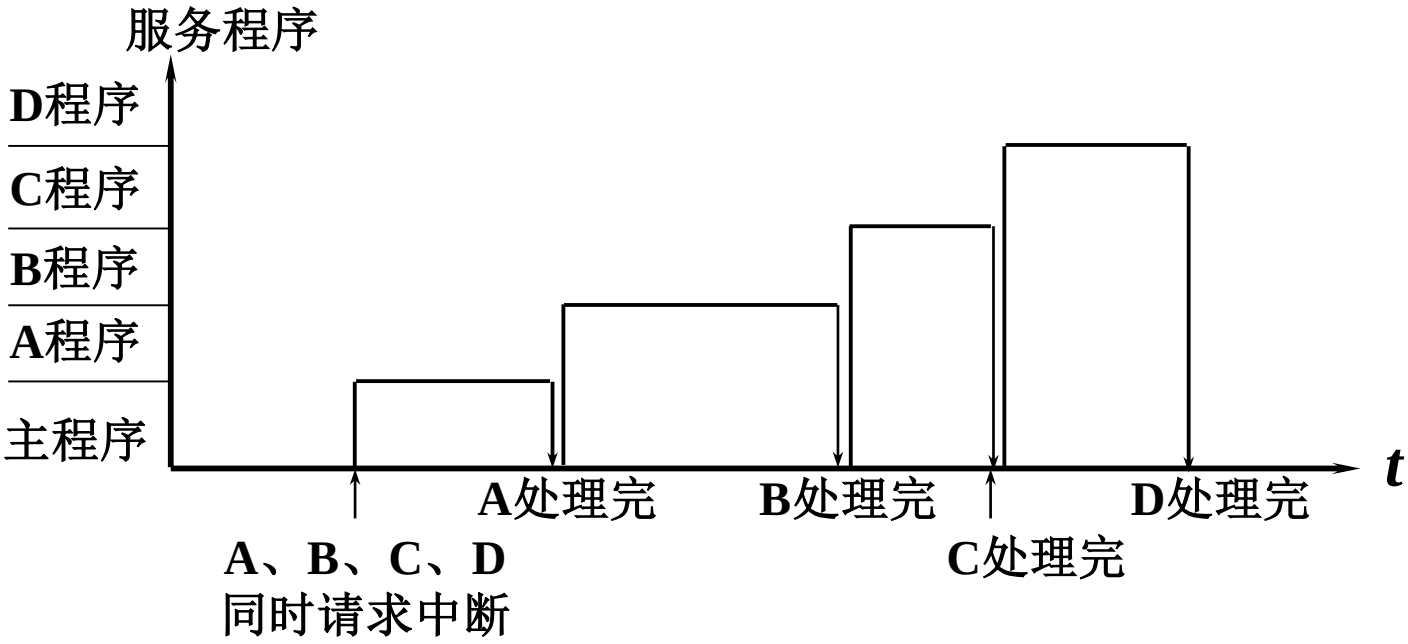
处理优先级　　可改变（通过重新设置屏蔽字）

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

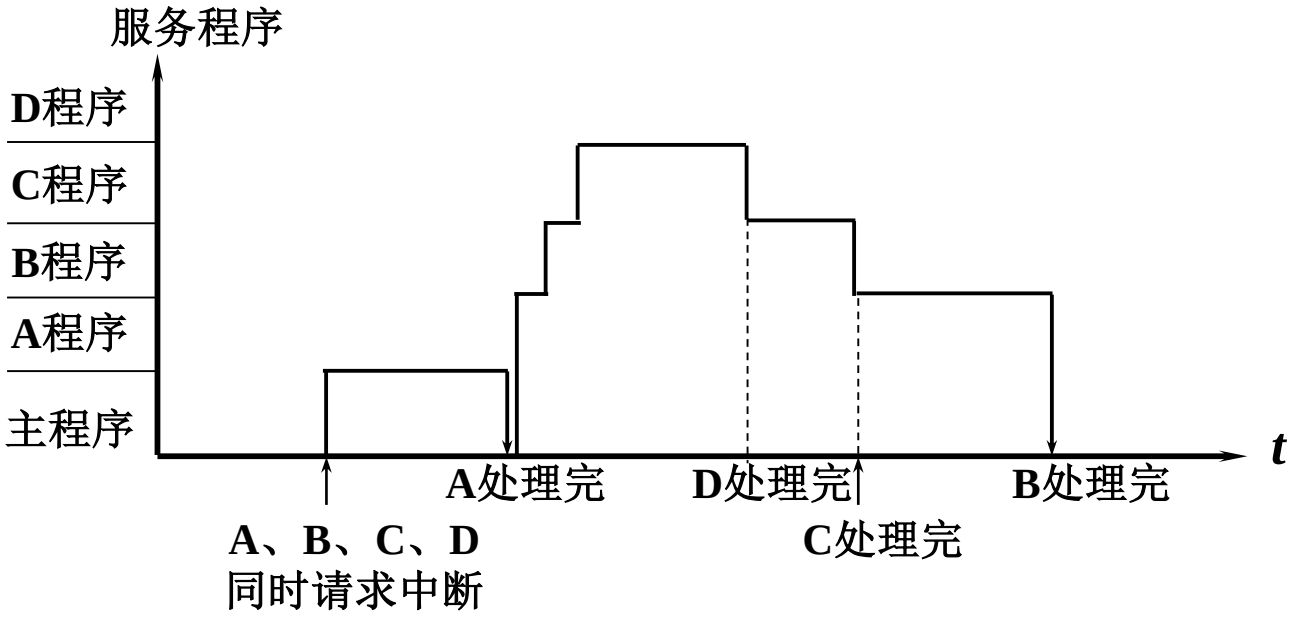
处理优先级 **A→D→C→B** 降序排列

(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



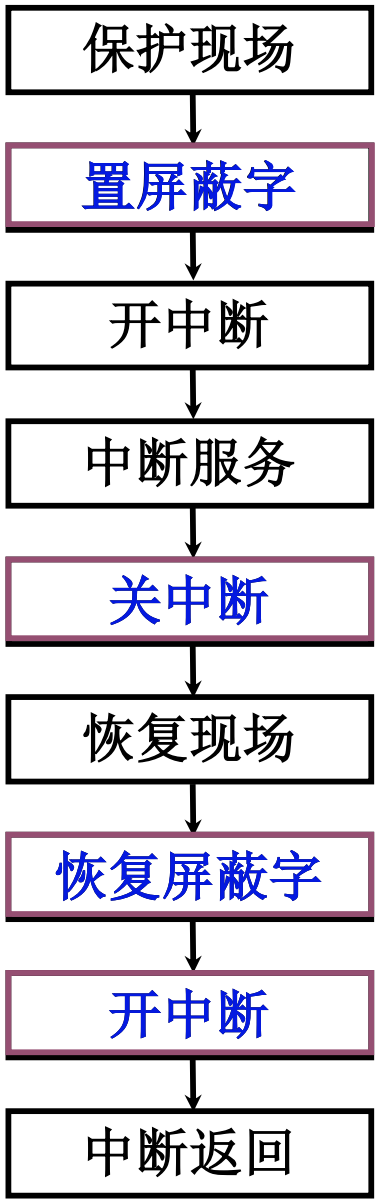
CPU 执行程序轨迹（新屏蔽字）

(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



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

总结：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

友情赠送：

第八章 课后作业

8.1 CPU 有哪些功能？画出其结构框图并简要说明每个部件的作用。

8.4 设 CPU 内有这些部件：PC、IR、SP、AC、MAR、MDR 和 CU。

（1）画出完成间接寻址的取数指令“LDA @X”（将主存某地址单元的内容取至 AC 中）的数据流（从取指令开始）。

（2）画出中断周期的数据流。

8.28 设某机有 4 个中断源 1、2、3、4，其响应优先级按 1→2→3→4 降序排列，现要求将中断处理次序改为 4→1→3→2。根据下图给出的 4 个中断源的请求时刻，画出 CPU 执行程序的轨迹。设每个中断源的中断服务程序时间均为 20 μs。