计算机组成原理

翁睿

哈尔滨工业大学

第 5 章　输入输出系统

5.1　概述

5.2　外部设备

5.3　I/O接口

5.4　程序查询方式

5.5　程序中断方式

5.6　DMA方式

三、I/O 设备与主机的联系方式 5.1

1. I/O 设备编址方式

(1) 统一编址　　用取数、存数指令

(2) 不统一编址　　有专门的 I/O 指令

2. 设备选址

用设备选择电路识别是否被选中

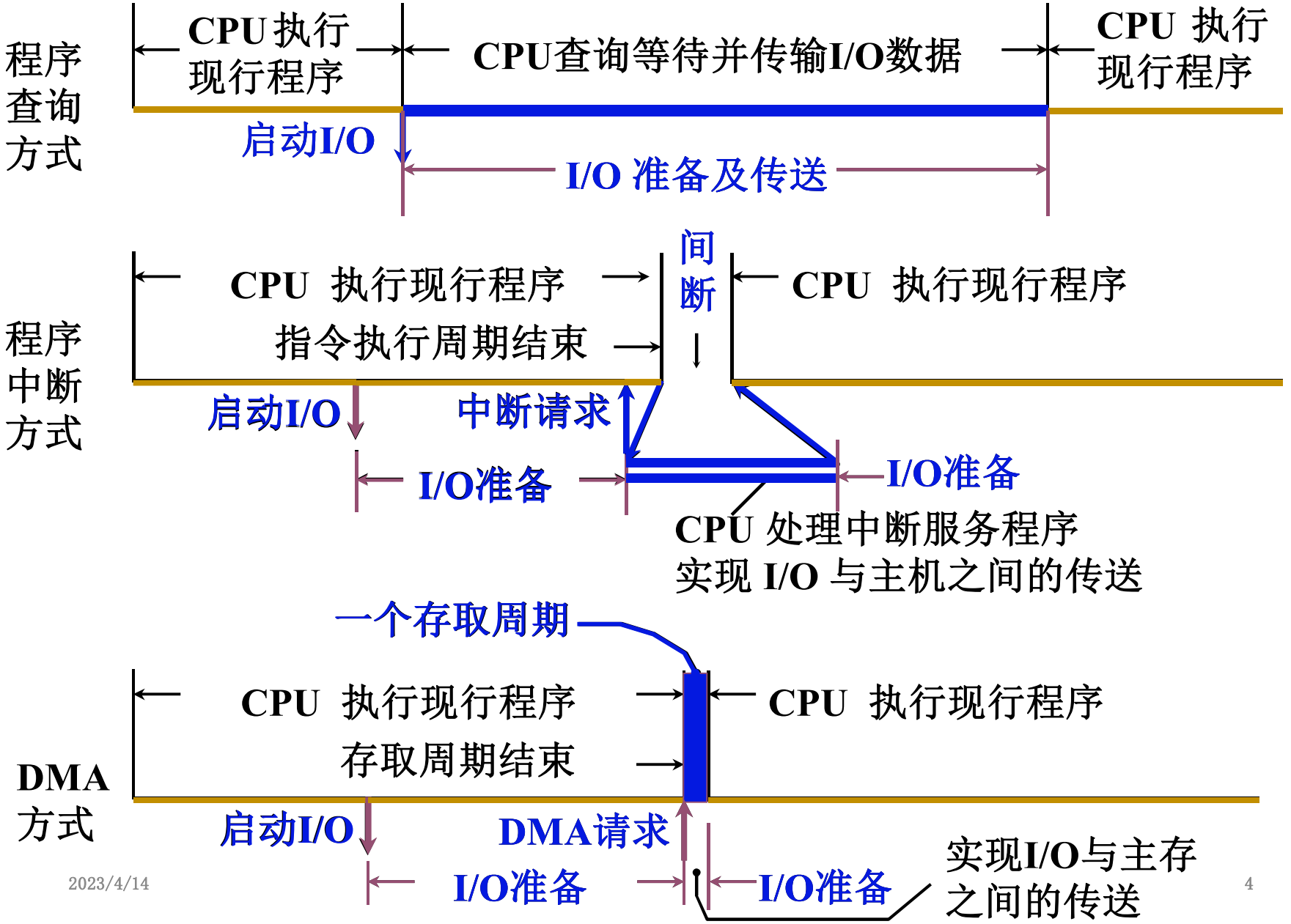
3. 传送方式

(1) 串行

(2) 并行

三种方式的 CPU 工作效率比较

5.1



5.3 I/O 接 口

一、概述

为什么要设置接口？

1. 实现设备的选择

2. 实现数据缓冲达到速度匹配

3. 实现数据串—并格式转换

4. 实现电平转换

5. 传送控制命令

6. 反映设备的状态（“忙”、“就绪”、“中断请求”）

二、接口的功能和组成

5.3

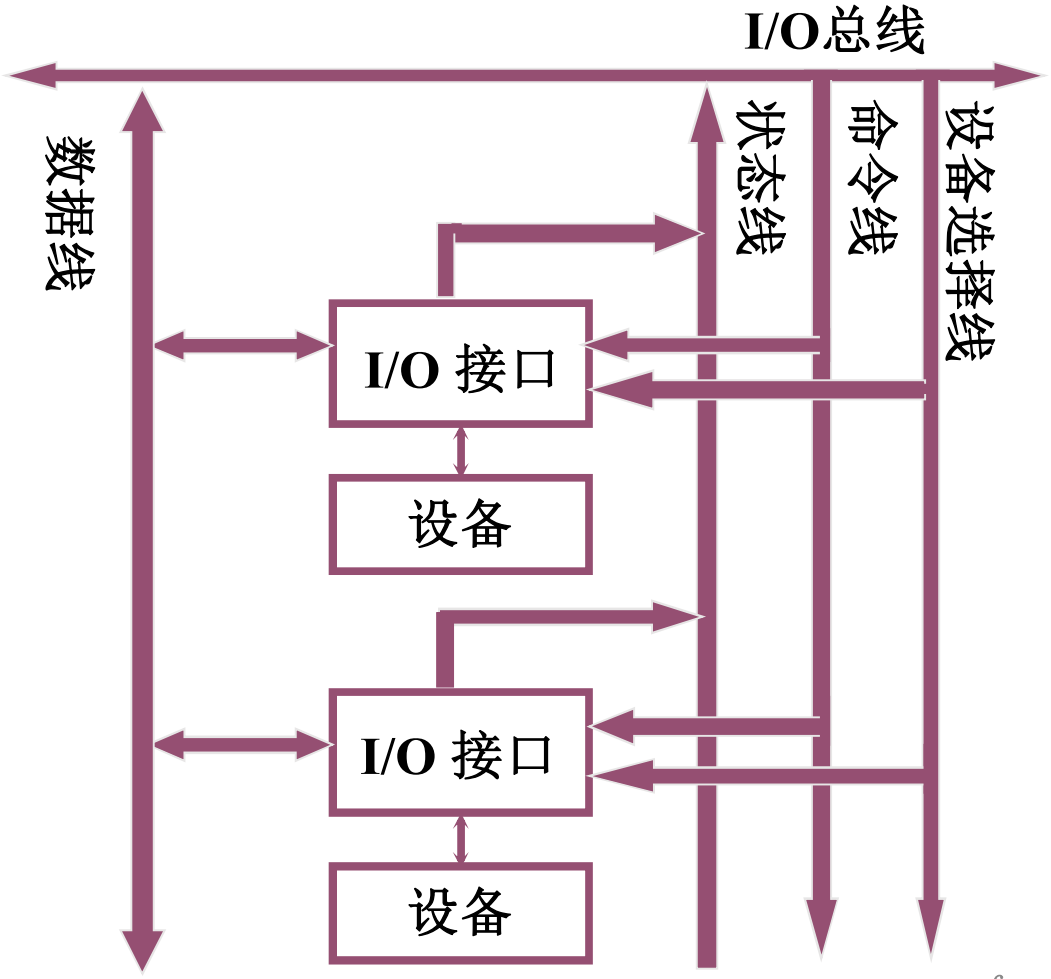
1. 总线连接方式的 I/O 接口电路

(1) 设备选择线

(2) 数据线

(3) 命令线

(4) 状态线



2. 接口的功能和组成

功能	组成
选址功能	设备选择电路
传送命令的功能	命令寄存器、命令译码器
传送数据的功能	数据缓冲寄存器
反映设备状态的功能	设备状态标记

将大多数IO设备共用的接口电路做在一个芯片中

- 完成触发器 D
- 工作触发器 B
- 中断请求触发器 INTR
- 屏蔽触发器 MASK

其余设备相关的电路做在设备控制器中

3. I/O 接口的基本组成

5.3

三、接口类型

5.3

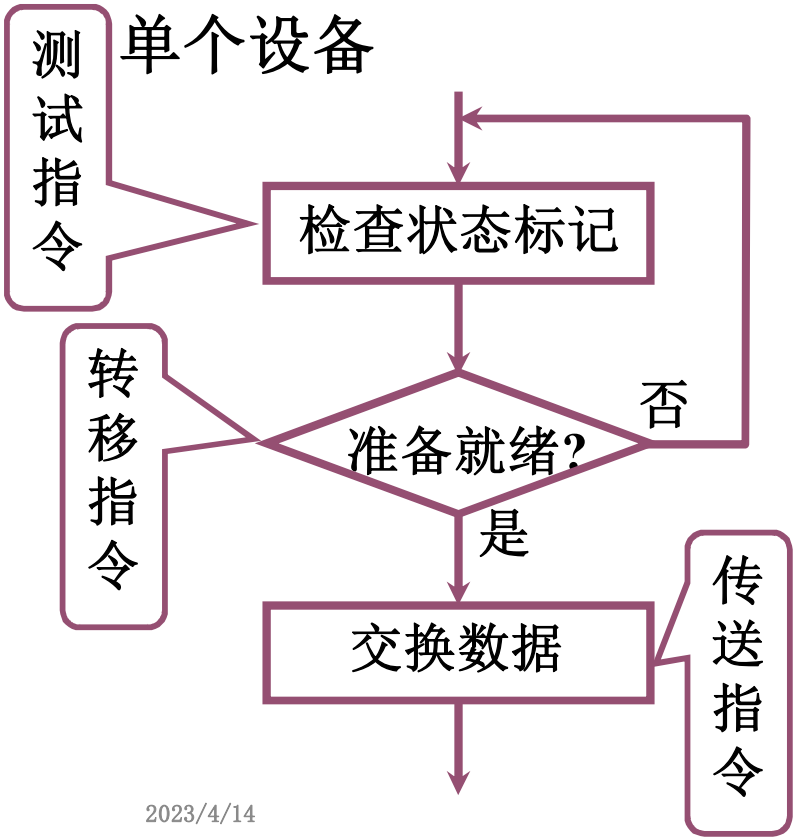
1. 按数据 传送方式 分类
   - 并行接口
   - 串行接口
2. 按功能 选择的灵活性 分类
   - 可编程接口
   - 不可编程接口
3. 按 通用性 分类
   - 通用接口
   - 专用接口
4. 按数据传送的 控制方式 分类
   - 中断接口
   - DMA 接口

5.4 程序查询方式

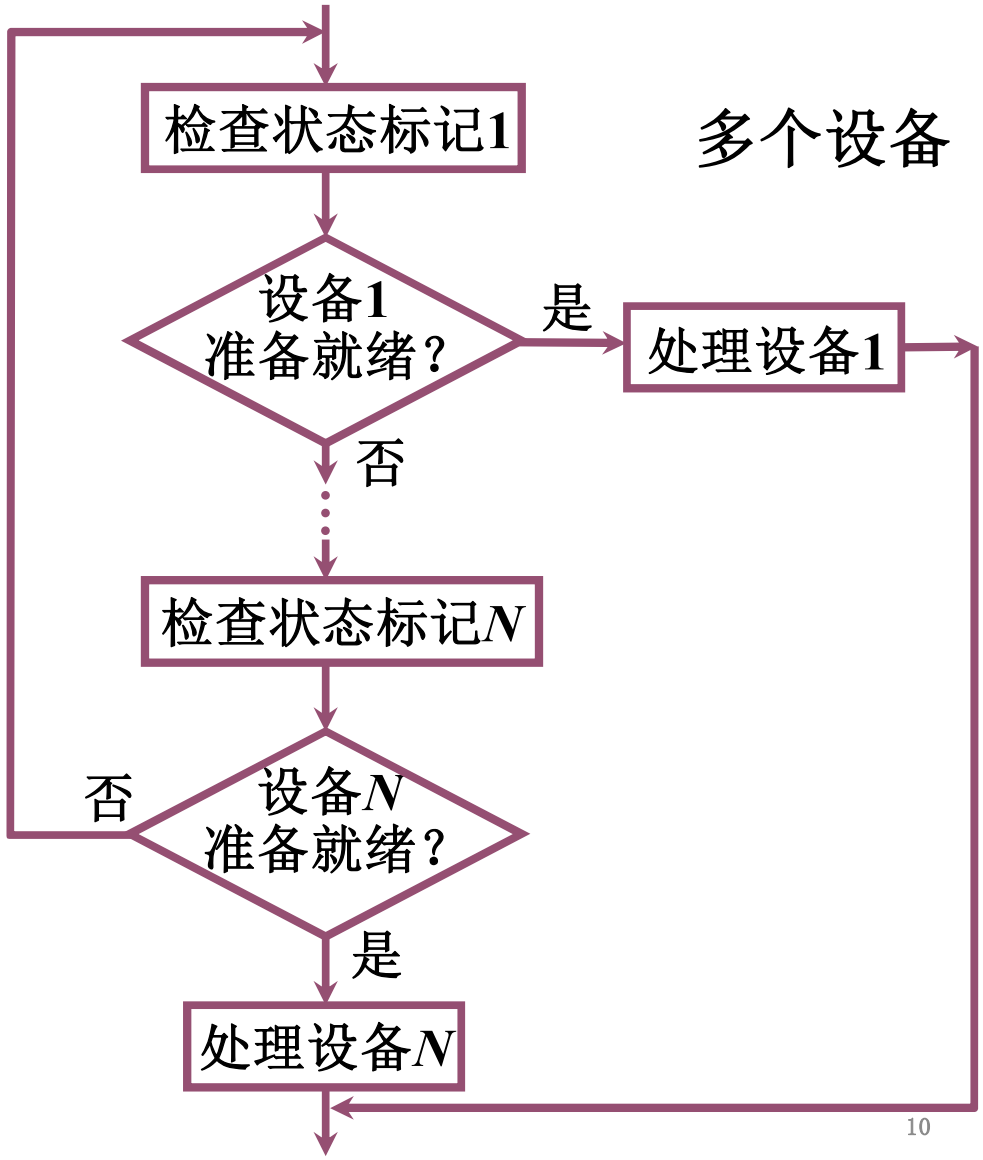
一、程序查询流程

1. 查询流程

单个设备

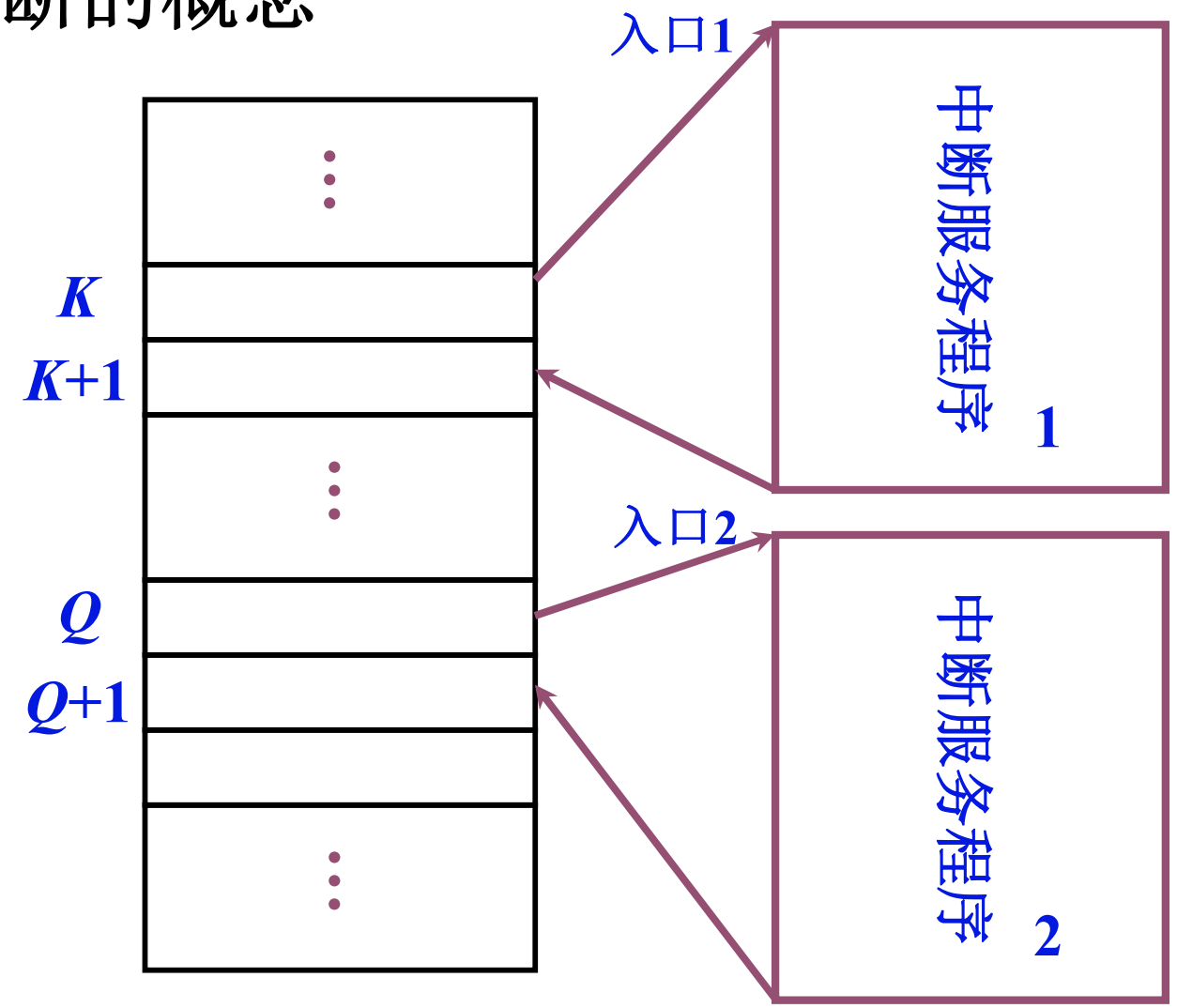


多个设备



5.5 程序中断方式

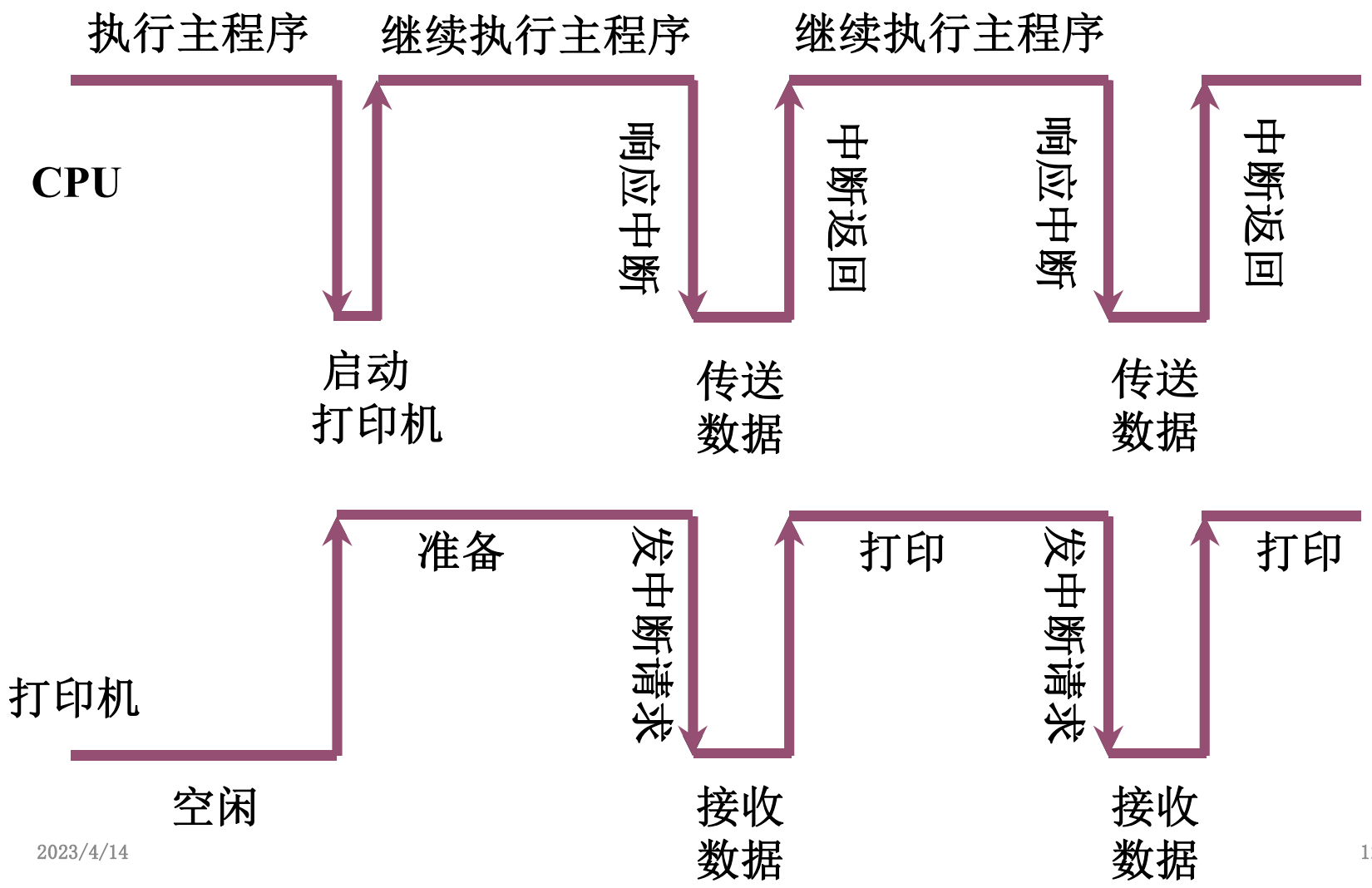
一、中断的概念



二、I/O 中断的产生

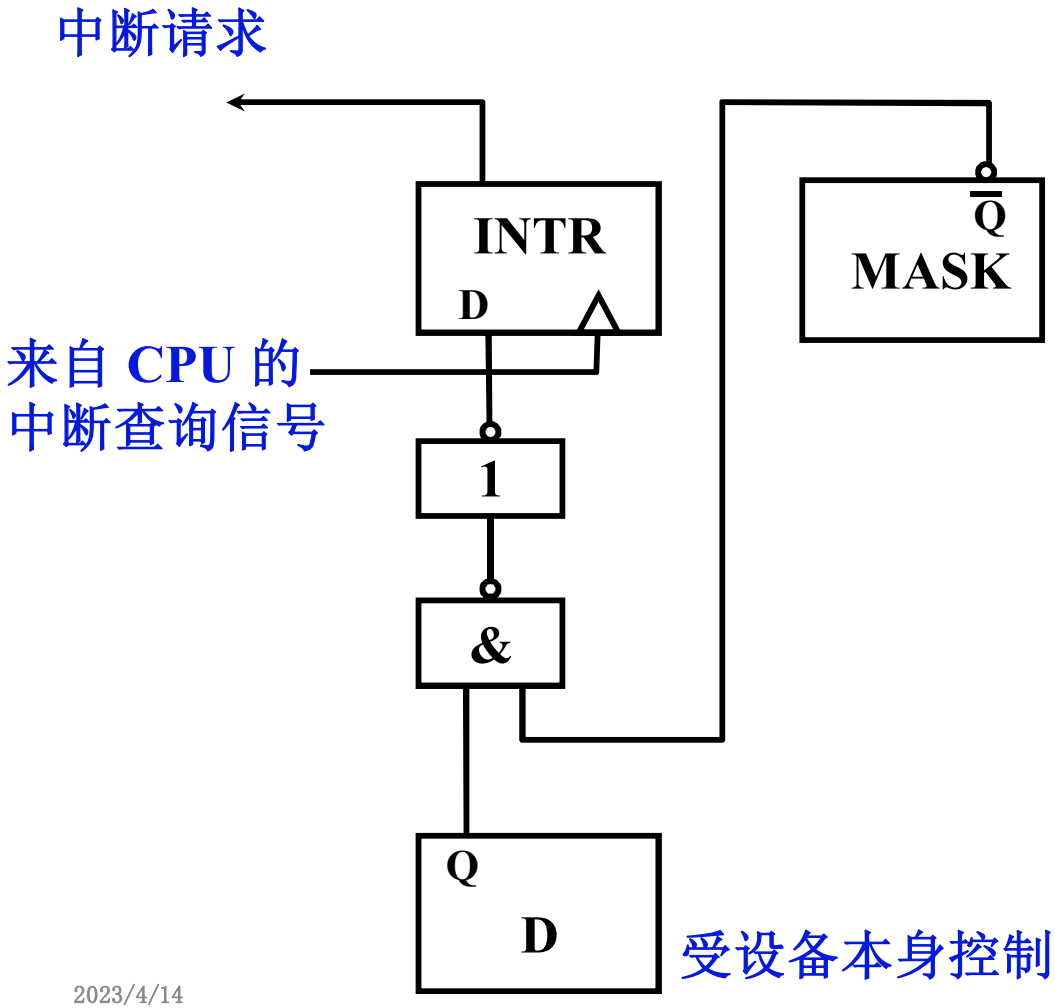
5.5

以打印机为例　CPU 与打印机并行工作



三、程序中断方式的接口电路 5.5

1. 配置中断请求触发器和中断屏蔽触发器



INTR
中断请求触发器

INTR = 1 有请求

MASK
中断屏蔽触发器

MASK = 1 被屏蔽

D 完成触发器

2. 排队器

5.5

排队 { 硬件　在 CPU 内或在接口电路中（链式排队器）
　　 软件　通过中断识别程序判断优先级

3. 中断向量地址形成部件

入口地址 { 由软件产生　按请求来源跳转到对应入口点
　　 硬件向量法　由 硬件 产生 向量地址
　　再由 向量地址 找到 入口地址

四、I/O 中断处理过程

5.5

1. CPU 响应中断的条件和时间

(1) 条件

允许中断触发器 EINT = 1

用 开中断 指令将 EINT 置 “1”

用 关中断 指令将 EINT 置 “ 0”

（或由硬件 自动复位）

(2) 时间

当 D = 1（随机发生）且 MASK = 0 时

在每条指令执行阶段的结束前

CPU 发 中断查询信号（将 INTR 置“1”）

五、中断服务程序流程

5.5

1. 中断服务程序的流程

(1) 保护现场

- 程序断点的保护　　中断隐指令完成
- 寄存器内容的保护　进栈指令

(2) 中断服务

对不同的 I/O 设备具有不同内容的设备服务

(3) 恢复现场　　出栈指令

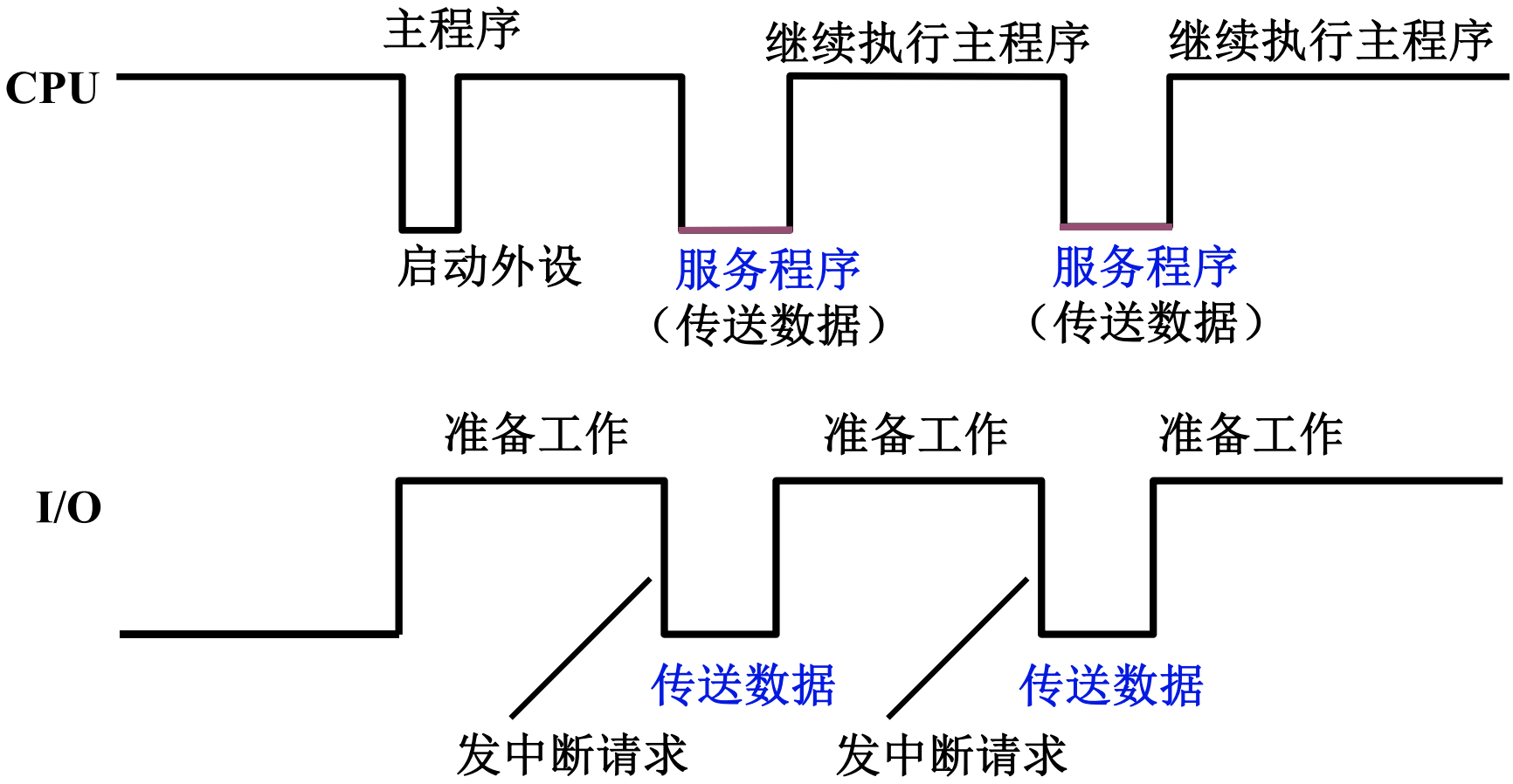
(4) 中断返回　　中断返回指令

2. 单重中断和多重中断

单重 中断　不允许中断 现行的 中断服务程序

多重 中断　允许级别更高 的中断源
中断 现行的 中断服务程序

主程序和服务程序抢占 CPU 示意图 5.5



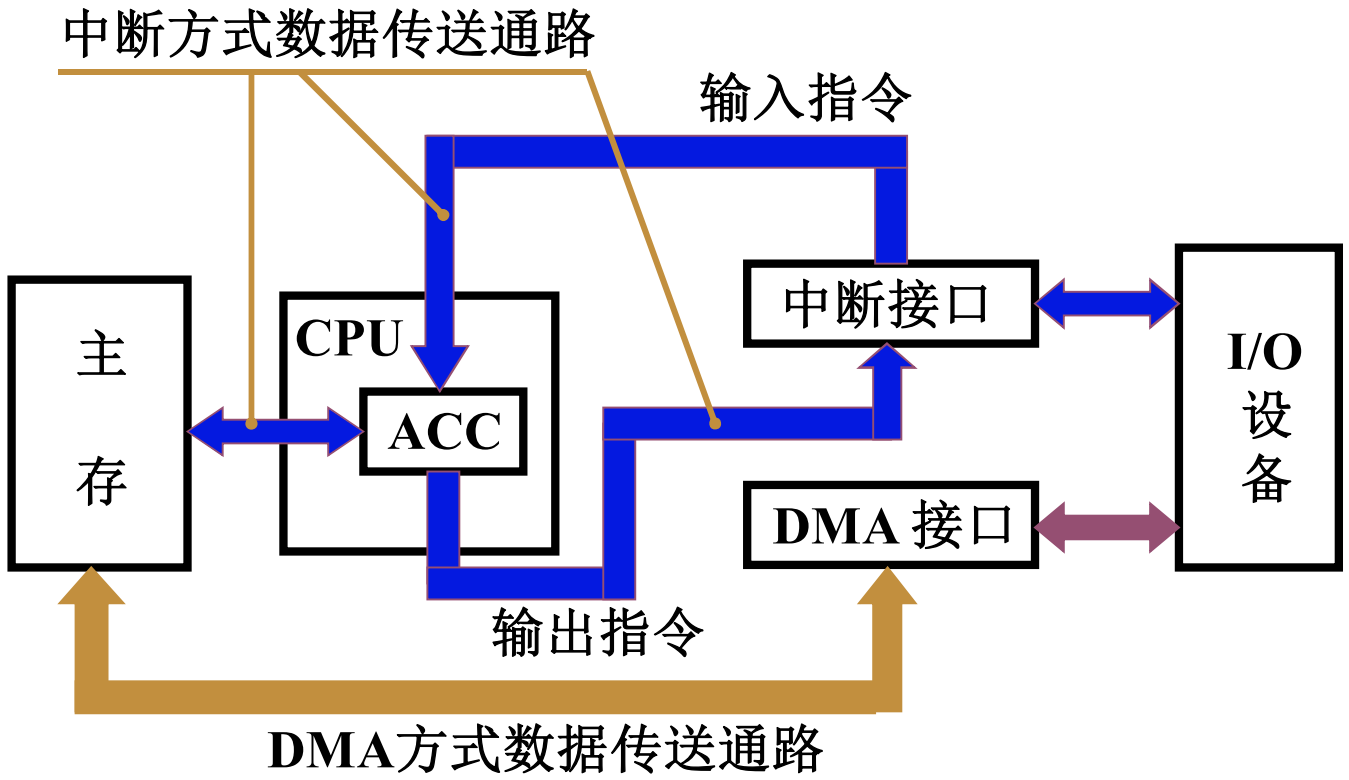
宏观 上 CPU 和 I/O 并行 工作

微观 上 CPU 中断现行程序 为 I/O 服务

5.6 DMA 方式

一、DMA 方式的特点

1. DMA 和程序中断两种方式的数据通路



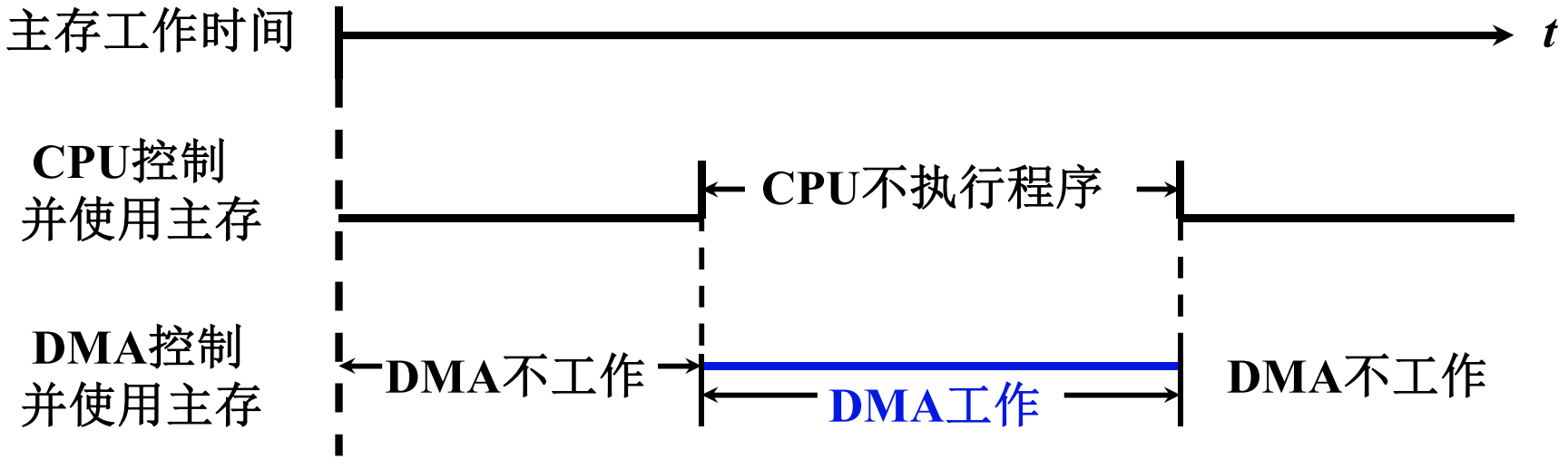
2. DMA 与主存交换数据的三种方式 5.6

(1) 停止 CPU 访问主存

控制简单

CPU 处于不工作状态或保持状态

未充分发挥 CPU 对主存的利用率



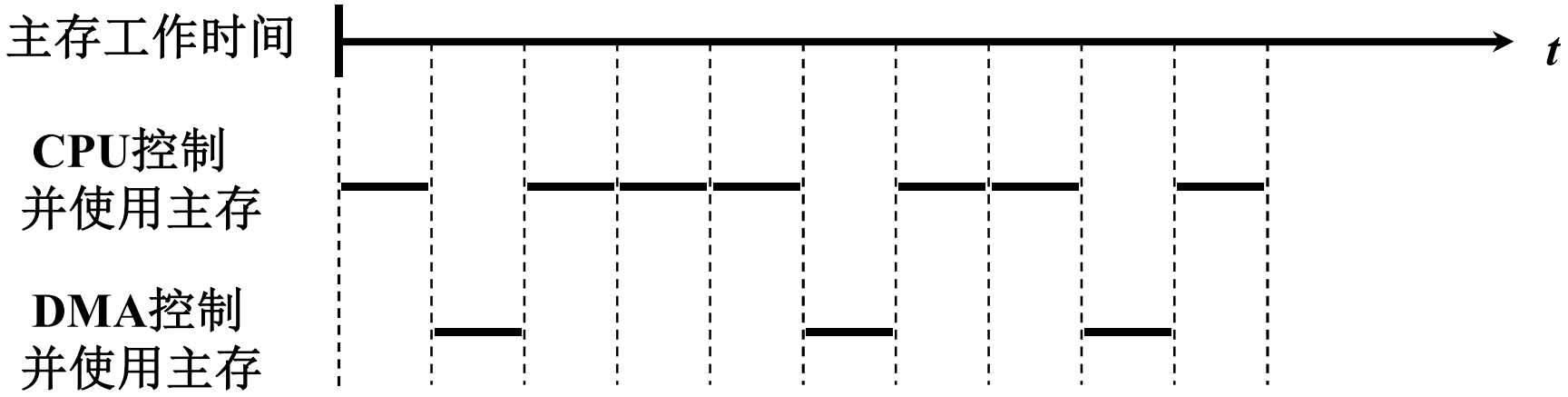
(2) 周期挪用（或周期窃取）

DMA 访问主存有三种可能

- CPU 此时不访存
- CPU 正在访存
- CPU 与 DMA 同时请求访存

此时 CPU 将总线控制权让给 DMA

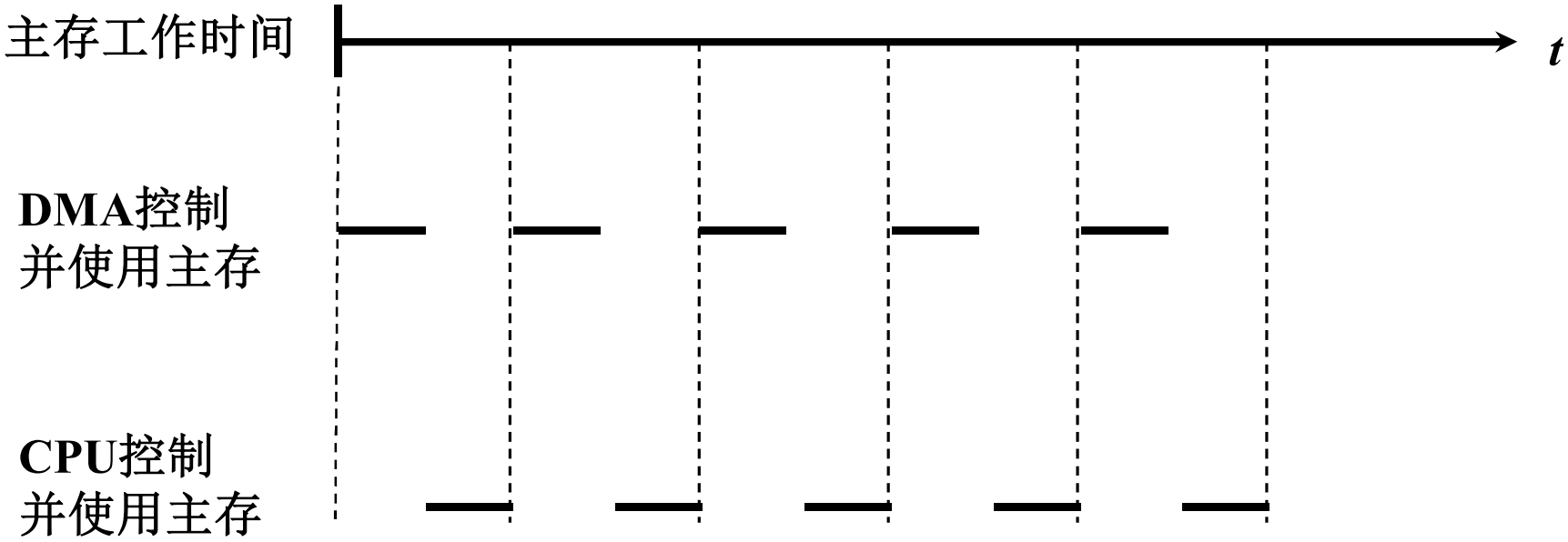
DMA优先：因为I/O不立即访问可能导致数据丢失(被新数据覆盖)



(3) DMA 与 CPU 交替访问

CPU 工作周期 { C_1 专供 DMA 访存
C_2 专供 CPU 访存

所有指令执行过程中的一个基准时间



不需要 申请建立和归还 总线的使用权

二、DMA 接口的功能和组成

5.6

1. DMA 接口功能

(1) 向 CPU 申请 DMA 传送

(2) 处理总线 控制权的转交

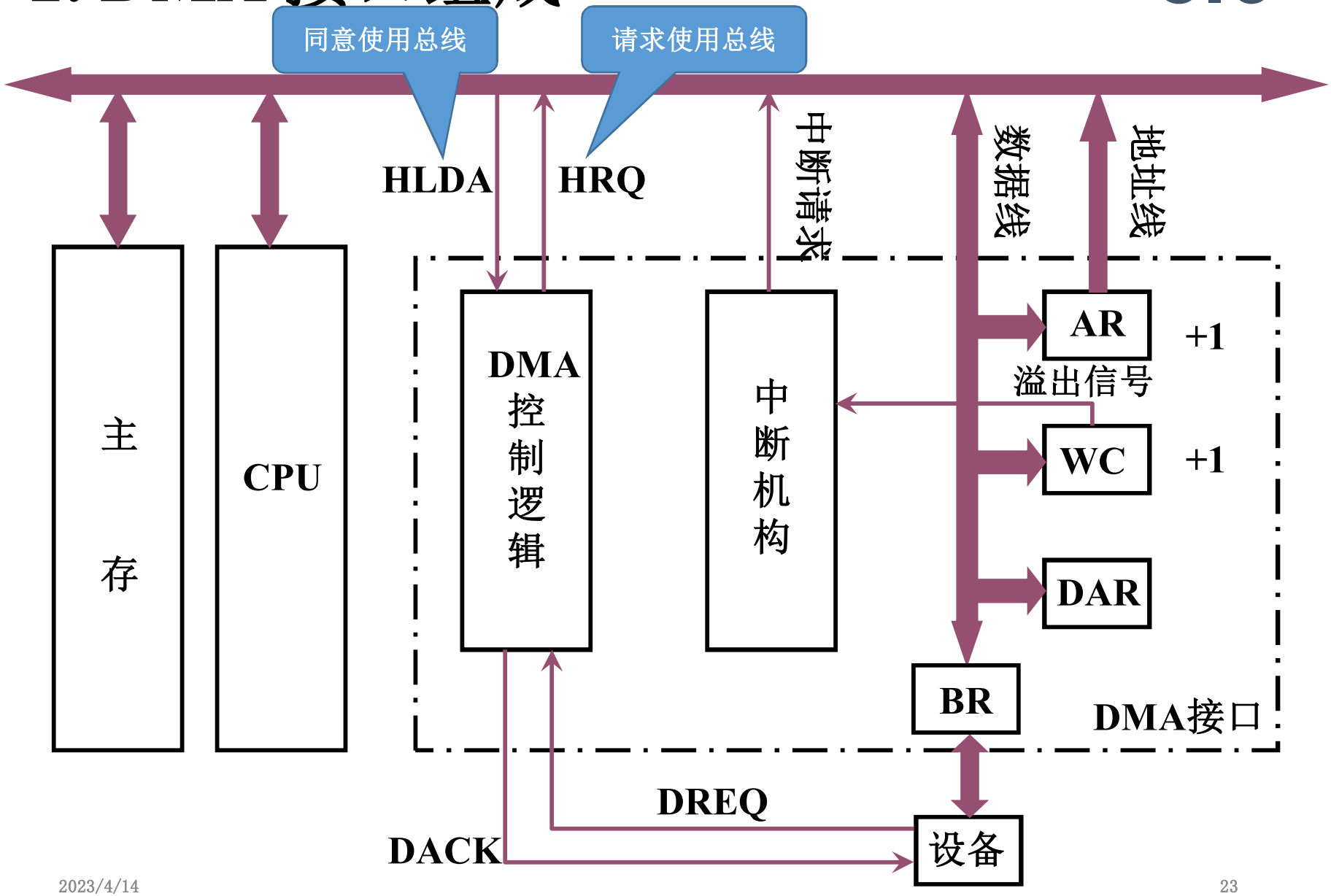
(3) 管理 系统总线、控制 数据传送

(4) 确定 数据传送的 首地址和长度

修正 传送过程中的数据 地址 和 长度

(5) DMA 传送结束时，给出操作完成信号

2. DMA 接口组成


同意使用总线
请求使用总线
HLDA
HRQ
中断请求
数据线
地址线
主存
CPU
DMA控制逻辑
中断机构
AR
+1
溢出信号
WC
+1
DAR
BR
DMA接口
DREQ
DACK
设备

三、DMA 的工作过程 5.6

1. DMA 传送过程

预处理 ➔ 数据传送 ➔ 后处理

CPU完成　CPU不参与　CPU完成

(1) 预处理

通过几条输入输出指令预置如下信息

- 通知 DMA 控制逻辑传送方向（入/出）
- 设备地址——→ DMA 的 DAR
- 主存地址——→ DMA 的 AR
- 传送字数——→ DMA 的 WC

(2) DMA 传送过程示意

CPU

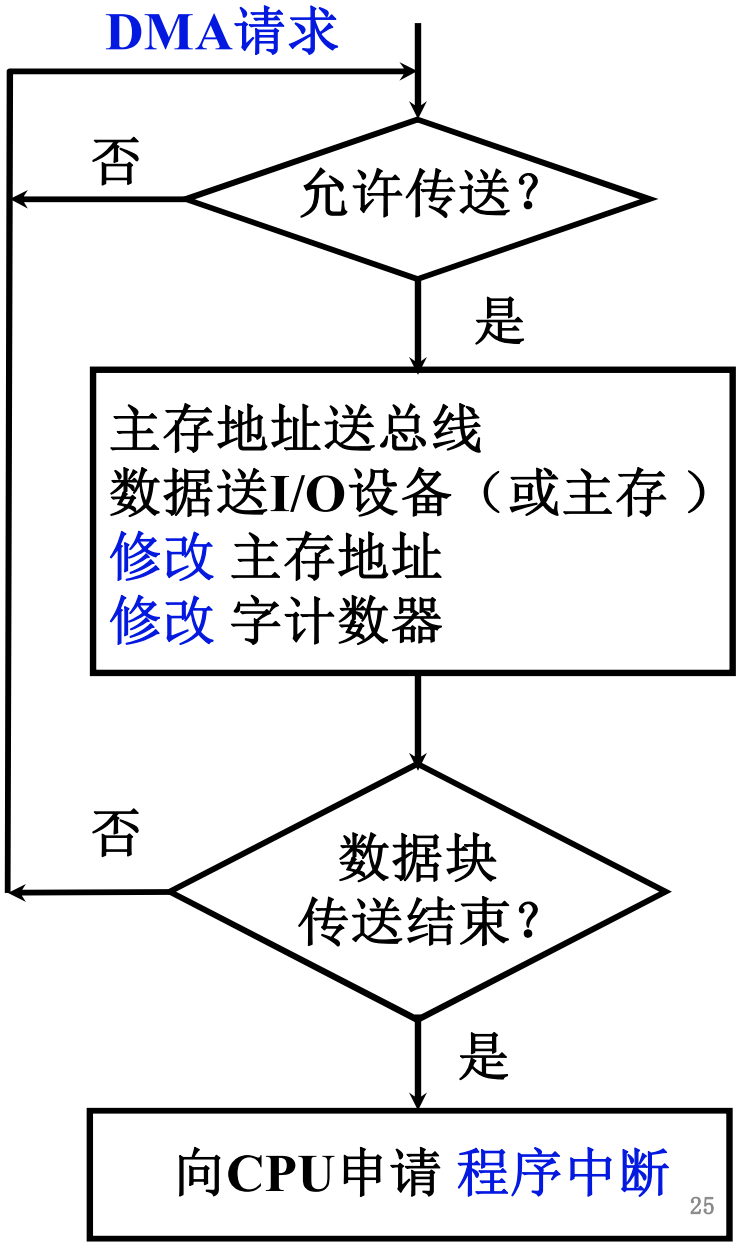
预处理：
主存起始地址 ⟶ DMA
设备地址 ⟶ DMA
传送数据个数 ⟶ DMA
启动设备

数据传送：
继续执行主程序
同时完成一批数据传送

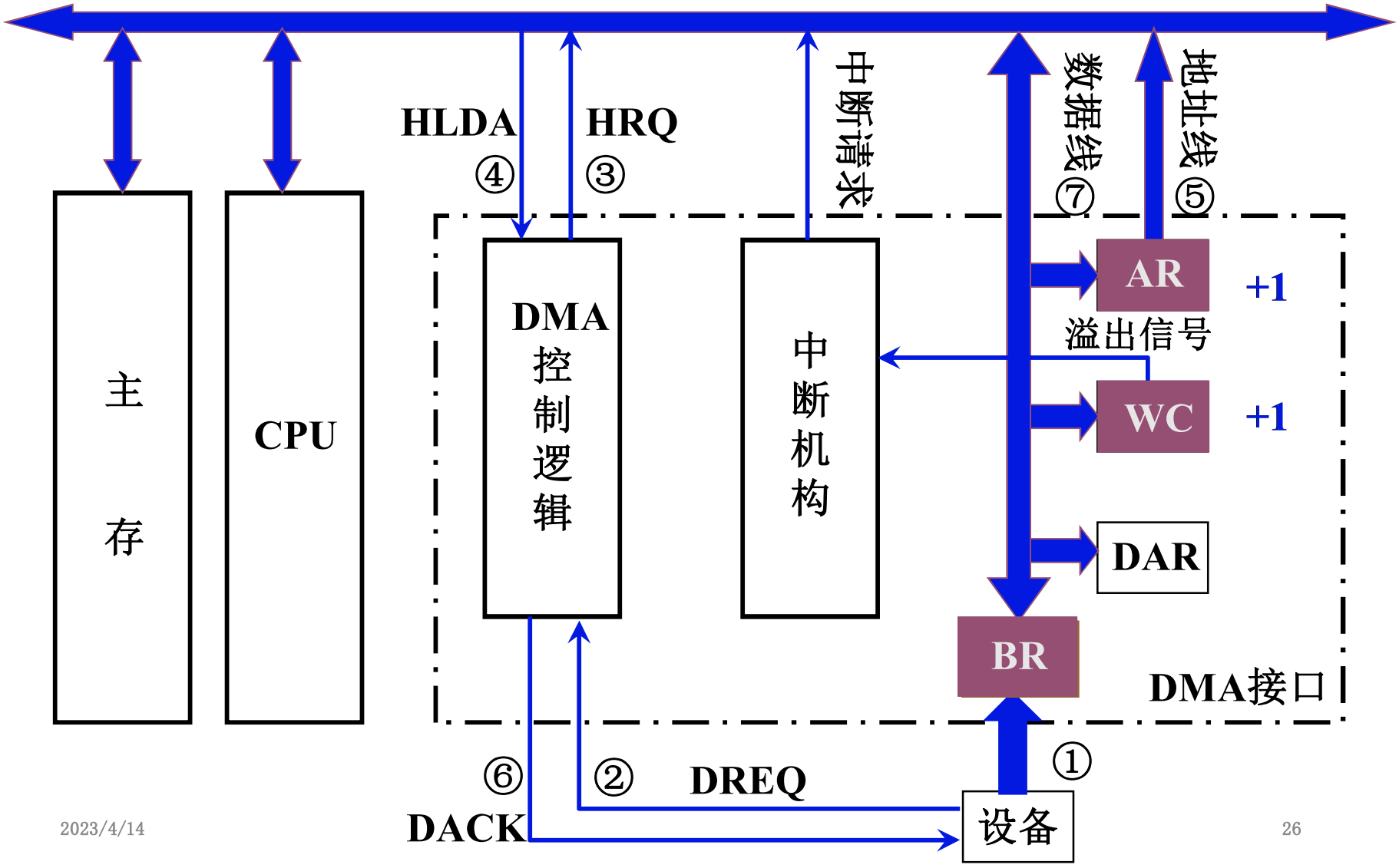
后处理：
中断服务程序
做 DMA 结束处理

继续执行主程序

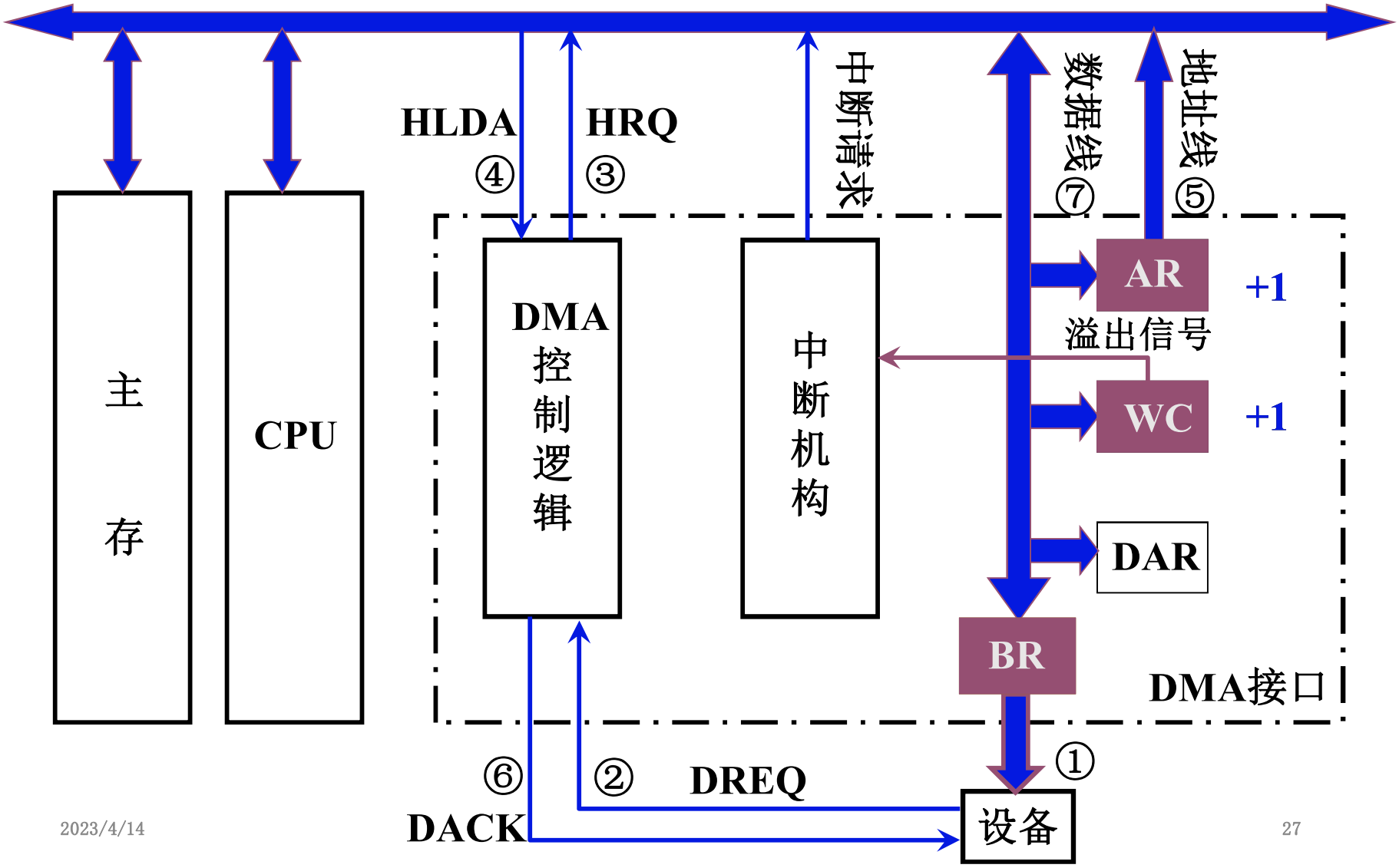
数据传送



(3) 数据传送过程（输入）


主存
CPU
HLDA
④
HRQ
③
DMA控制逻辑
中断请求
中断机构
数据线⑦
地址线⑤
AR
+1
溢出信号
WC
+1
DAR
BR
DMA接口
⑥
DACK
②
DREQ
①
设备

(4) 数据传送过程（输出）


主存
CPU
DMA控制逻辑
中断机构
HLDA
④
HRQ
③
中断请求
数据线⑦
地址线⑤
AR
+1
溢出信号
WC
+1
DAR
BR
DMA接口
⑥
DACK
②
DREQ
①
设备

(5) 后处理

校验送入主存的数是否正确

是否继续用 DMA

测试传送过程是否正确，错则转诊断程序

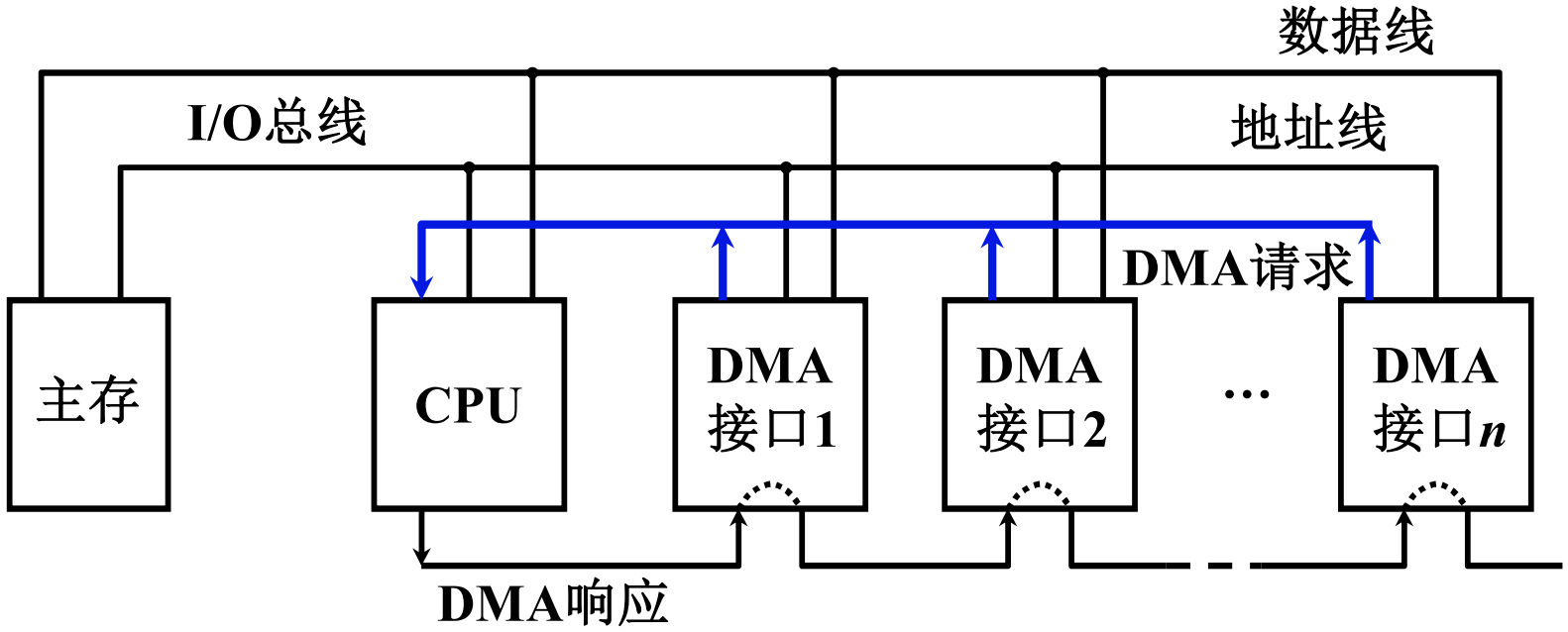
由中断服务程序完成

DMA传输伴随着中断过程

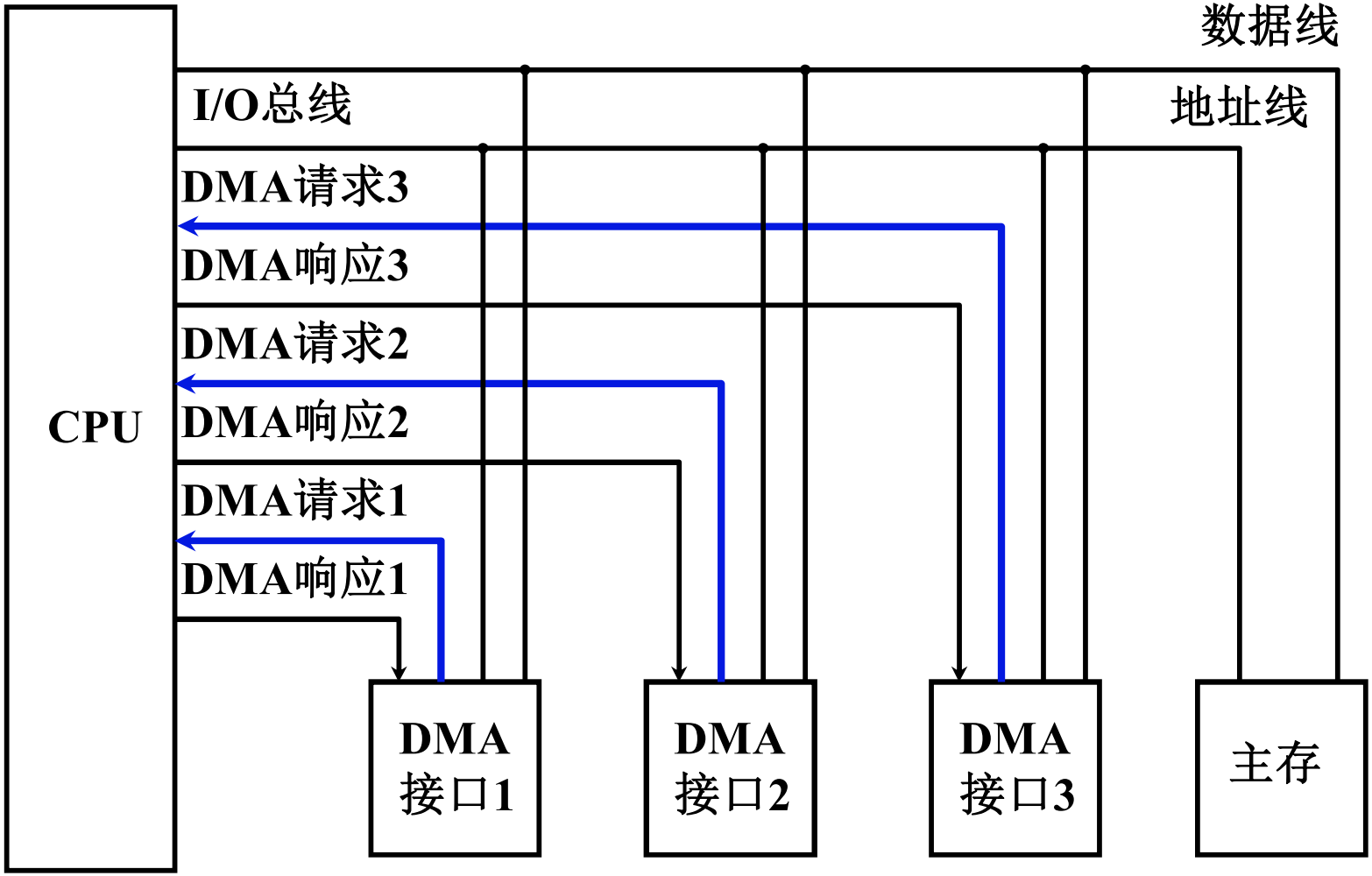
但其中断服务程序完成的功能不同

2. DMA 接口与系统的连接方式

(1) 具有公共请求线的 DMA 请求



(2) 独立的 DMA 请求


数据线
I/O总线
地址线
DMA请求3
DMA响应3
DMA请求2
DMA响应2
DMA请求1
DMA响应1
CPU
DMA
接口1
DMA
接口2
DMA
接口3
主存

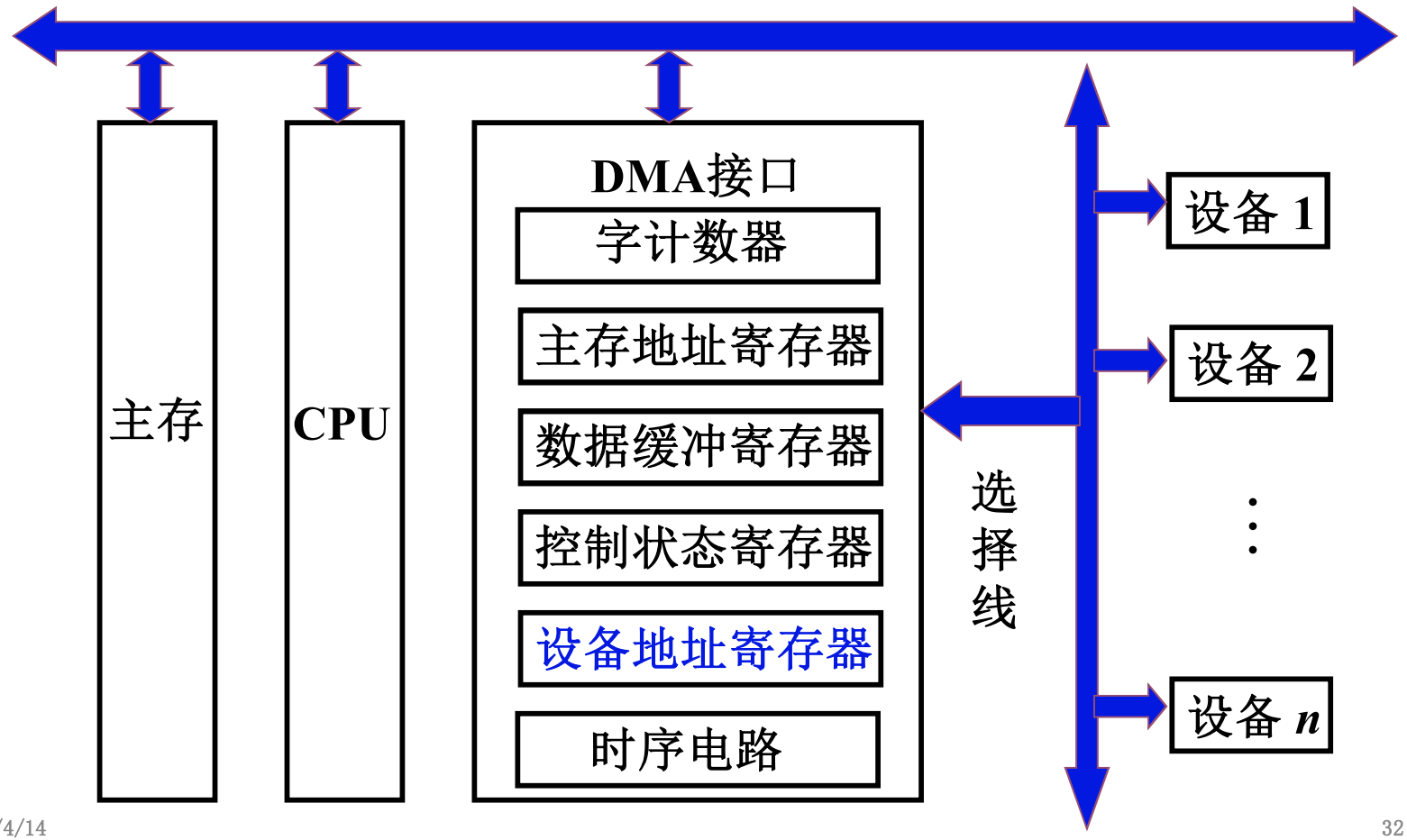
3. DMA 方式与程序中断方式的比较 5.6

	中断方式	DMA 方式
(1) 数据传送	程序	硬件
(2) 响应时间	指令执行结束	存取周期结束
(3) 处理异常情况	能	不能
(4) 中断请求	传送数据	后处理
(5) 优先级	低	高

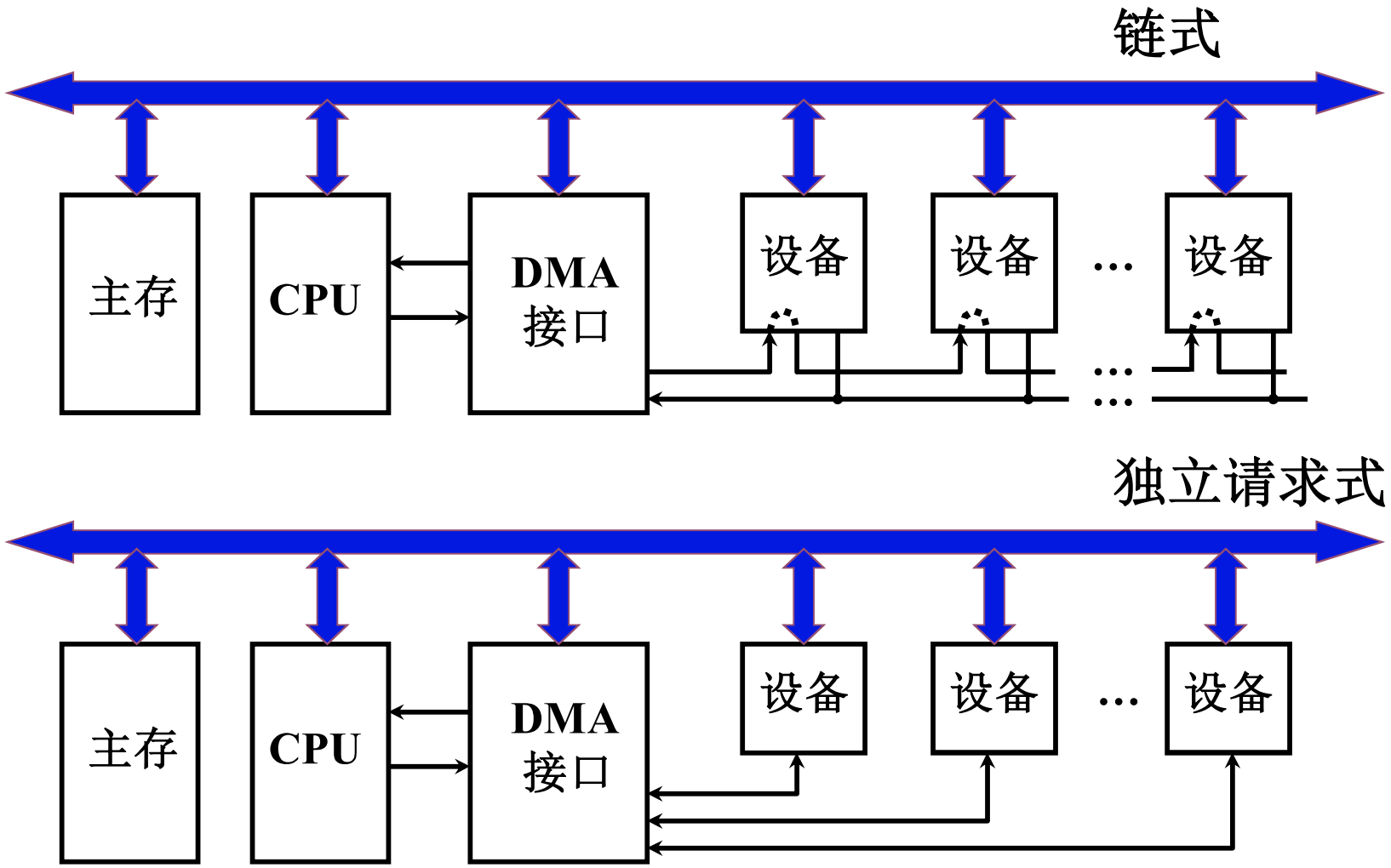
四、DMA 接口的类型

1. 选择型

在 物理上 连接 多个 设备
在 逻辑上 只允许连接 一个 设备

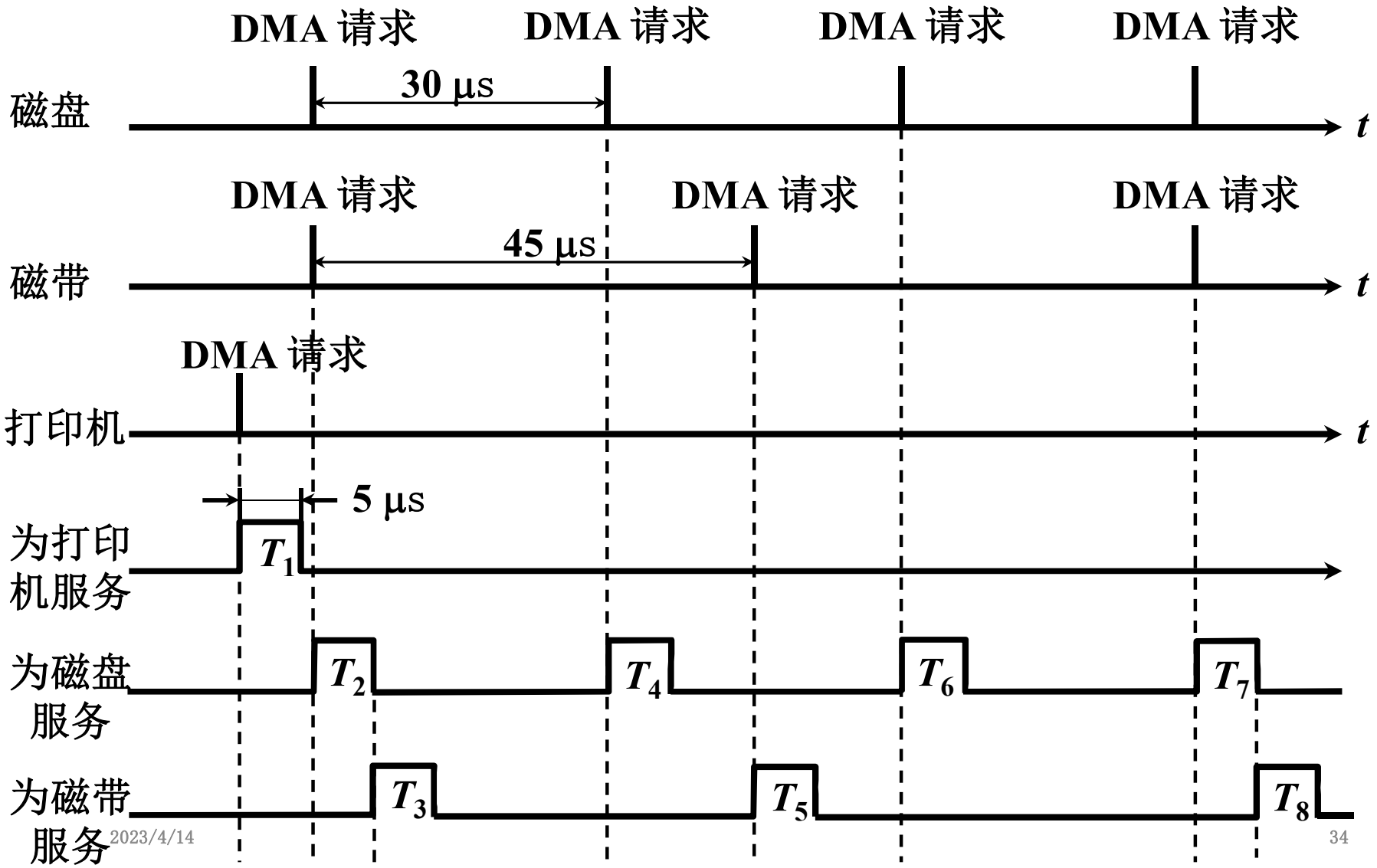


2. 多路型 在 物理上 连接 多个 设备
在 逻辑上 允许连接 多个 设备同时工作



3. 多路型 DMA 接口的工作原理

5.6



第5章 作业

（T5.4, T5.10, T5.31, T5.32, T5.33）

5.4 试比较程序查询方式、程序中断方式和 DMA 方式对 CPU 工作效率的影响。

5.10 什么是 I/O 接口，它与端口有何区别？为什么要设置 I/O 接口？I/O 接口如何分类？

5.31 假设某设备向 CPU 传送信息的最高频率是 40 000 次/秒，而相应的中断处理程序执行时间为 40 μs，试问该外设是否可用程序中断方式与主机交换信息，为什么？

5.32 设磁盘存储器转速为 3 000 r/min，分 8 个扇区，每扇区存储 1 KB，主存与磁盘存储器数据传送的宽度为 16 位（即每次传送 16 位）。假设一条指令最长执行时间是 25 μs，是否可采用一条指令执行结束时响应 DMA 请求的方案，为什么？若不行，应采取什么方案？

5.33 试从下面 7 个方面比较程序查询、程序中断和 DMA 三种方式的综合性能。

（1）数据传送依赖软件还是硬件。

（2）传送数据的基本单位。

（3）并行性。

（4）主动性。

（5）传输速度。

（6）经济性。

（7）应用对象。

第 7 章　指 令 系 统

7.1 机器指令

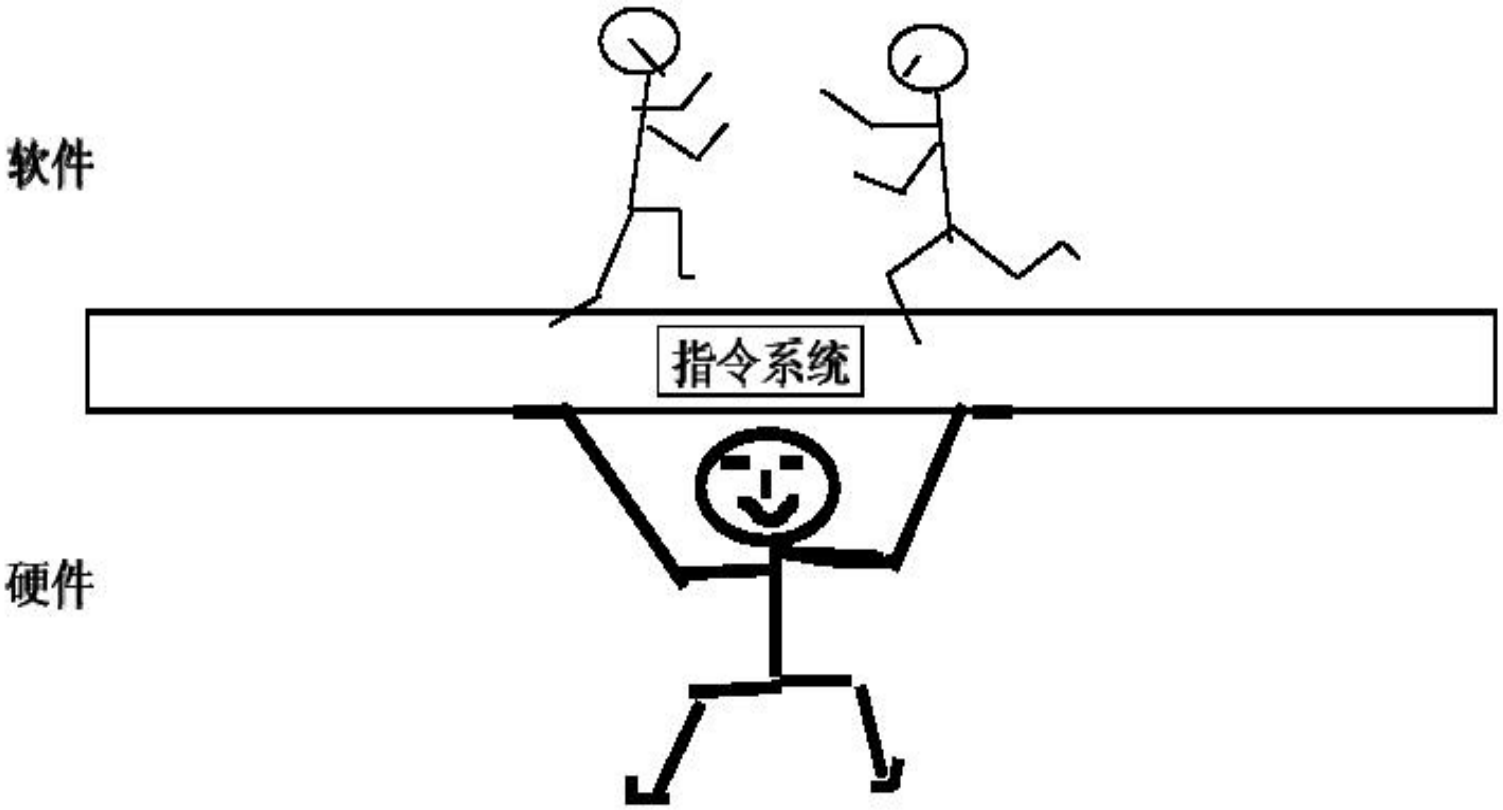
7.2 操作数类型和操作类型

7.3 寻址方式

7.4 指令格式举例

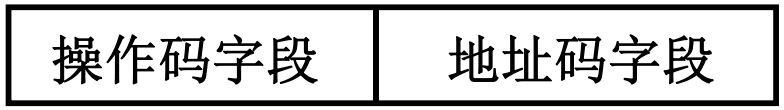
7.5 RISC 技术

指令系统在计算机中的地位



7.1 机 器 指 令

一、指令的一般格式



1. 操作码　反映机器做什么操作

(1) 长度固定

用于指令字长较长的情况 ， **RISC**

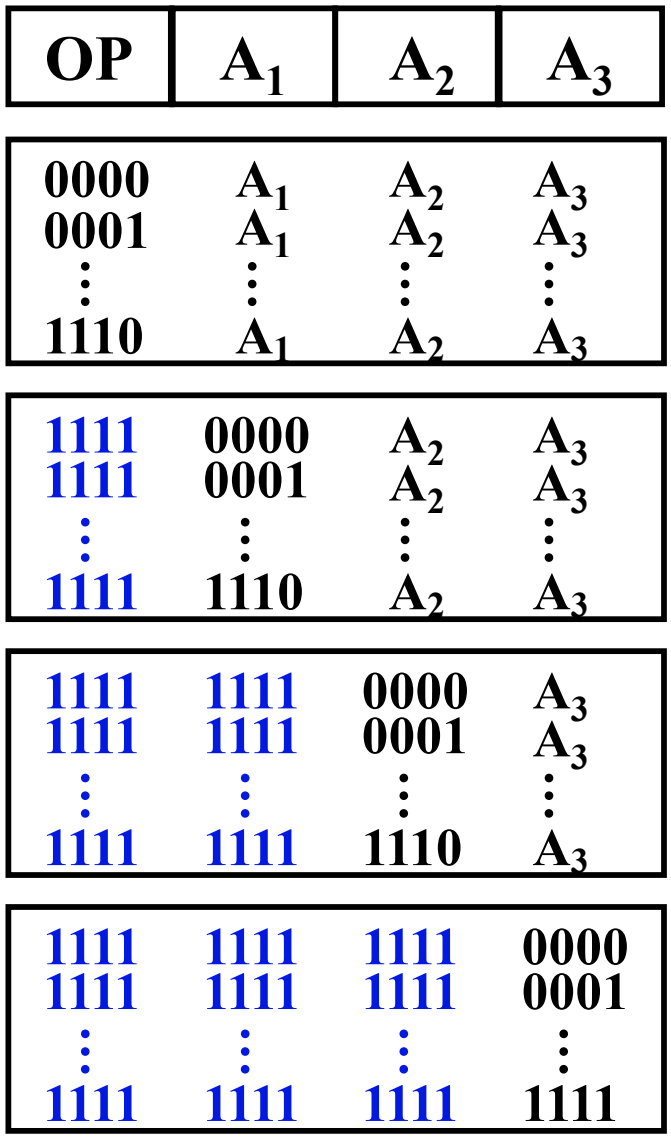
如 **IBM 370** 操作码 **8** 位

(2) 长度可变

操作码分散在指令字的不同字段中

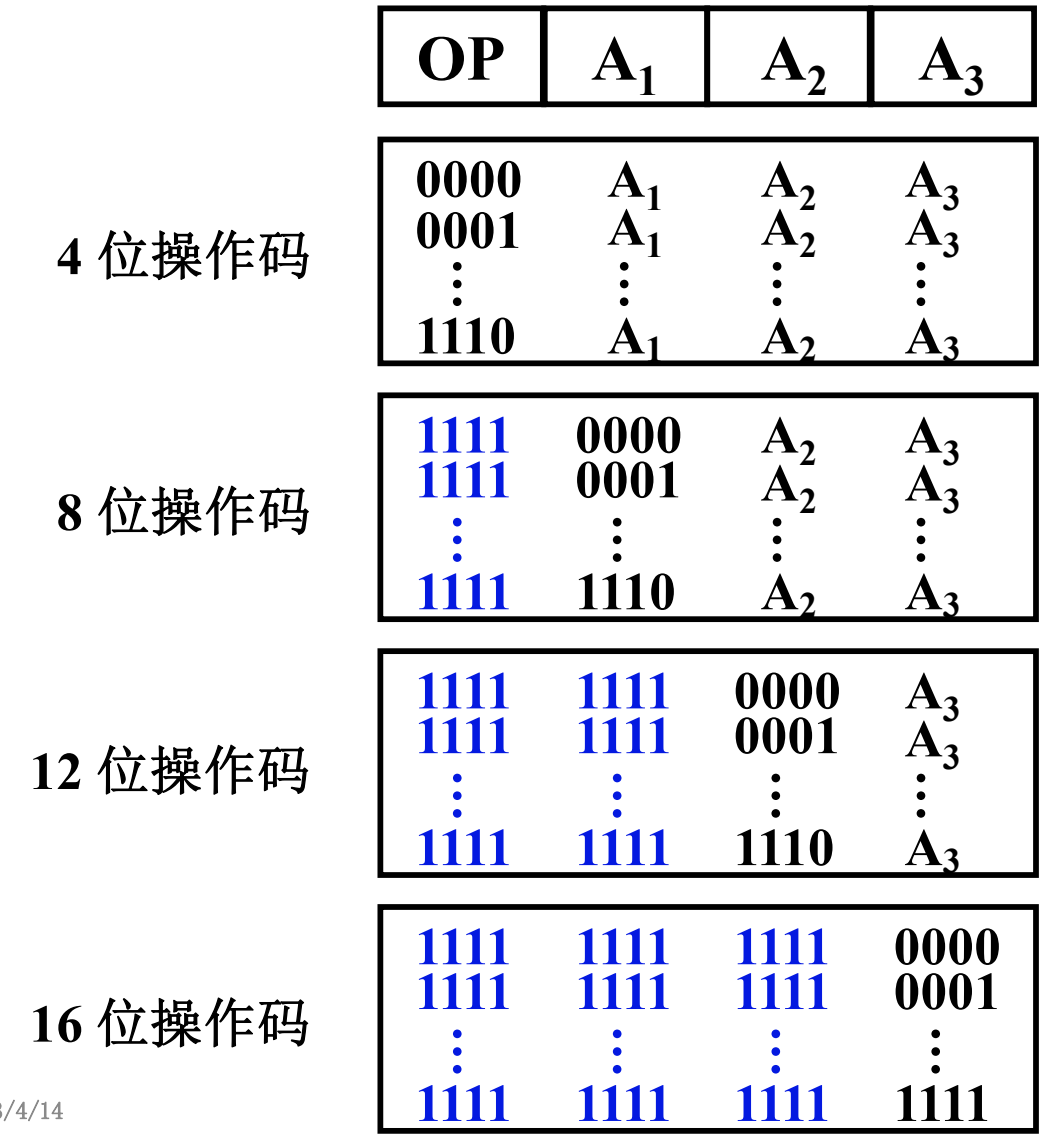
(3) 扩展操作码技术

操作码的位数随地址数的减少而增加



(3) 扩展操作码技术

操作码的位数随地址数的减少而增加

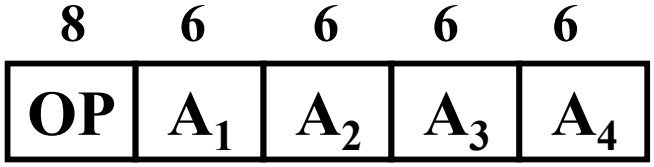


三地址指令操作码
每减少一种最多可多构成
2^4 种二地址指令

二地址指令操作码
每减少一种最多可多
构成2^4 种一地址指令

2. 地址码

(1) 四地址



A_1 第一操作数地址

A_2 第二操作数地址

A_3 结果的地址

A_4 下一条指令地址

(A_1) OP $(A_2) \longrightarrow A_3$

设指令字长为 32 位

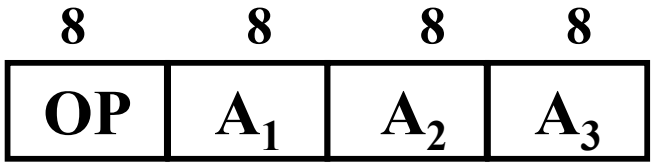
操作码固定为 8 位

4 次访存

寻址范围 $2^6 = 64$

若 PC 代替 A_4

(2) 三地址



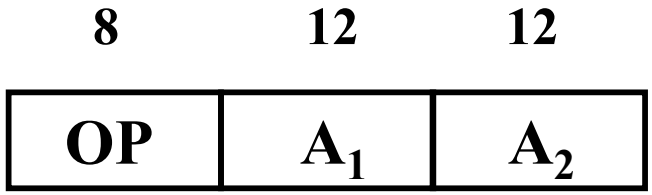
(A_1) OP $(A_2) \longrightarrow A_3$

4 次访存

寻址范围 $2^8 = 256$

若 A_3 用 A_1 或 A_2 代替

(3) 二地址



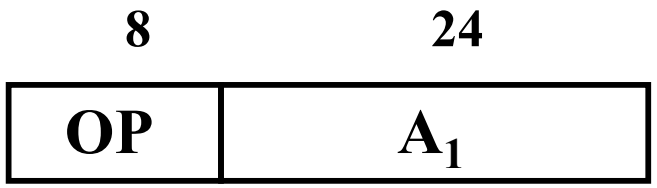
或

(A_1) OP $(A_2) \longrightarrow A_1$　　4 次访存

(A_1) OP $(A_2) \longrightarrow A_2$　　寻址范围 $2^{12} = 4$ K

若结果存于 ACC　3次访存　　若ACC 代替 A_1（或A_2）

(4) 一地址



(ACC) OP $(A_1) \longrightarrow$ ACC　　寻址范围 $2^{24} = 16$ M

(5) 零地址　　无地址码

二、指令字长

指令字长决定于
- 操作码的长度
- 操作数地址的长度
- 操作数地址的个数

1. 指令字长 固定

指令字长 = 存储字长

2. 指令字长 可变

按字节的倍数变化

小结

- 当用一些硬件资源代替指令字中的地址码字段后
  - 可扩大指令的寻址范围
  - 可缩短指令字长
  - 可减少访存次数
- 当指令的地址字段为寄存器时

  三地址　OP　R_1，R_2，R_3

  二地址　OP　R_1，R_2

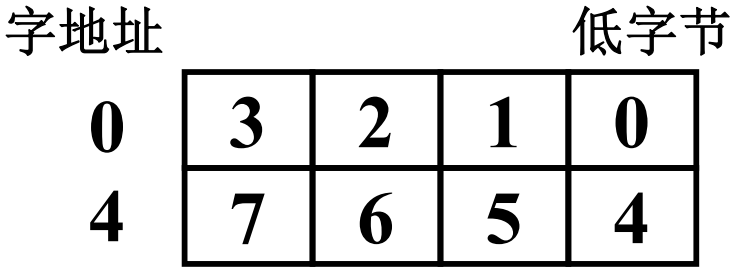
  一地址　OP　R_1
  - 可缩短指令字长
  - 指令执行阶段不访存

7.2 操作数类型和操作种类

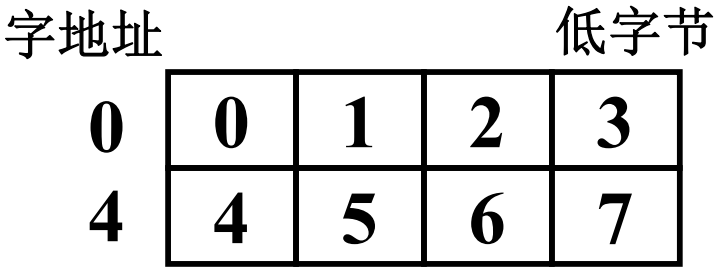
一、操作数类型

地址	无符号整数
数字	定点数、浮点数、十进制数
字符	**ASCII**
逻辑数	逻辑运算

二、数据在存储器中的存放方式



字地址 为 低字节 地址



字地址 为 高字节 地址

存储器中的数据存放（存储字长为32位）

边界对准

				地址（十进制）
字（地址 0）				0
字（地址 4）				4
字节（地址11）	字节（地址10）	字节（地址 9）	字节（地址 8）	8
字节（地址15）	字节（地址14）	字节（地址13）	字节（地址12）	12
半字（地址18）✓		半字（地址16）✓		16
半字（地址22）✓		半字（地址20）✓		20
双字（地址24）▲				24
双字				28
双字（地址32）▲				32
双字				36

边界未对准

			地址（十进制）
字(地址2)		半字(地址0)	0
字节(地址7)	字节(地址6)	字(地址4)	4
半字(地址10)		半字(地址8)	8

三、操作类型

1. 数据传送

源	寄存器	寄存器	存储器	存储器
目的	寄存器	存储器	寄存器	存储器
例如	MOVE	STORE MOVE PUSH	LOAD MOVE POP	MOVE

置“1”，清“0”

2. 算术逻辑操作

加、减、乘、除、增 1、减 1、求补、浮点运算、十进制运算

与、或、非、异或、位操作、位测试、位清除、位求反

如 8086 ADD SUB MUL DIV INC DEC CMP NEG

AAA AAS AAM AAD

AND OR NOT XOR TEST

• ARM数据处理指令——算术运算

助记符	说明	操作	条件码位置
ADD Rd, Rn, operand2	加法运算指令	Rd←Rn+operand2	ADD{cond}{S}
SUB Rd, Rn, operand2	减法运算指令	Rd←Rn-operand2	SUB{cond}{S}
RSB Rd, Rn, operand2	逆向减法指令	Rd←operand2-Rn	RSB{cond}{S}
ADC Rd, Rn, operand2	带进位加法	Rd←Rn+operand2+Carry	ADC{cond}{S}
SBC Rd, Rn, operand2	带进位减法指令	Rd←Rn-operand2-(NOT)Carry	SBC{cond}{S}
RSC Rd, Rn, operand2	带进位逆向减法指令	Rd←operand2-Rn-(NOT)Carry	RSC{cond}{S}

3. 移位操作

算术移位　逻辑移位

循环移位（带进位和不带进位）

4. 转移

(1) 无条件转移　**JMP**

(2) 条件转移

结果为零转　（Z = 1）**JZ**

结果溢出转　（O = 1）**JO**

结果有进位转（C = 1）**JC**

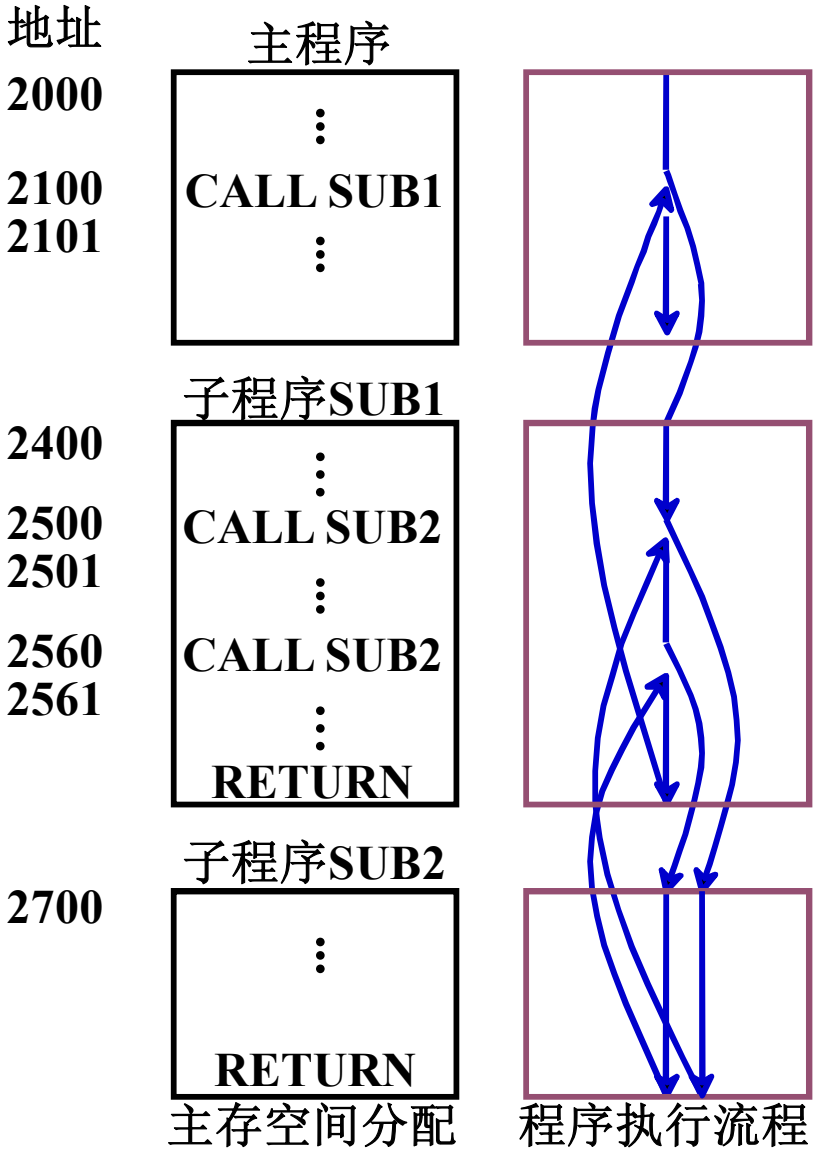
跳过一条指令 **SKP**

如

```
300
 ⋮
305  SKP  DZ;  D = 0 则跳
306
307
```

完成触发器

(3) 调用和返回


地址
主程序
2000
2100
CALL SUB1
2101
子程序SUB1
2400
2500
CALL SUB2
2501
2560
CALL SUB2
2561
RETURN
子程序SUB2
2700
RETURN
主存空间分配
程序执行流程

(4) 陷阱（Trap）与陷阱指令

意外事故的中断

- 一般不提供给用户直接使用

  在出现事故时，由 CPU 自动产生并执行（隐指令）

- 设置供用户使用的陷阱指令

  如 8086　　INT　TYPE　　软中断

  提供给用户使用的陷阱指令，完成系统调用

5. 输入输出

入　　端口地址 ⟶ CPU 的寄存器

如　　IN AX, *m*　　IN AX, DX

出　　CPU 的寄存器 ⟶ 端口地址

如　　OUT *n*, AX　　OUT DX, AX