

第10章 控制单元的设计

10.1 组合逻辑设计

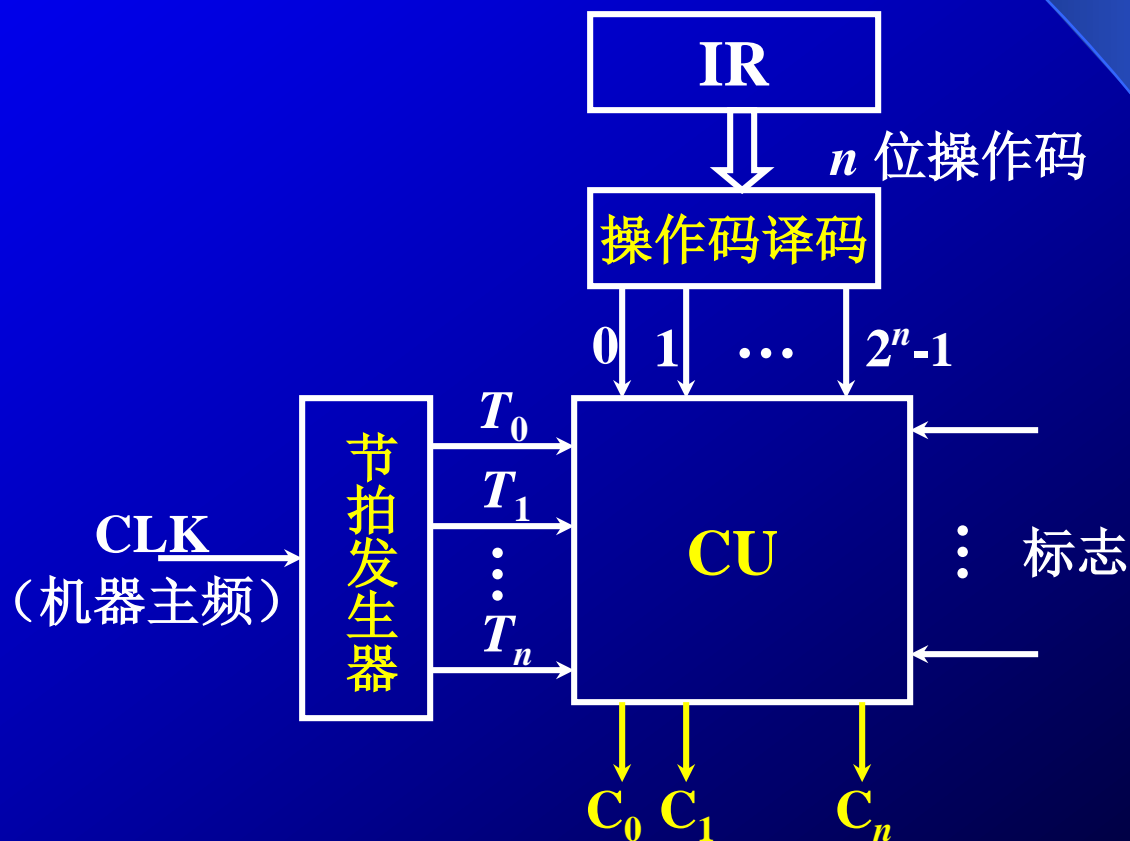
10.2 微程序设计



10.1 组合逻辑设计

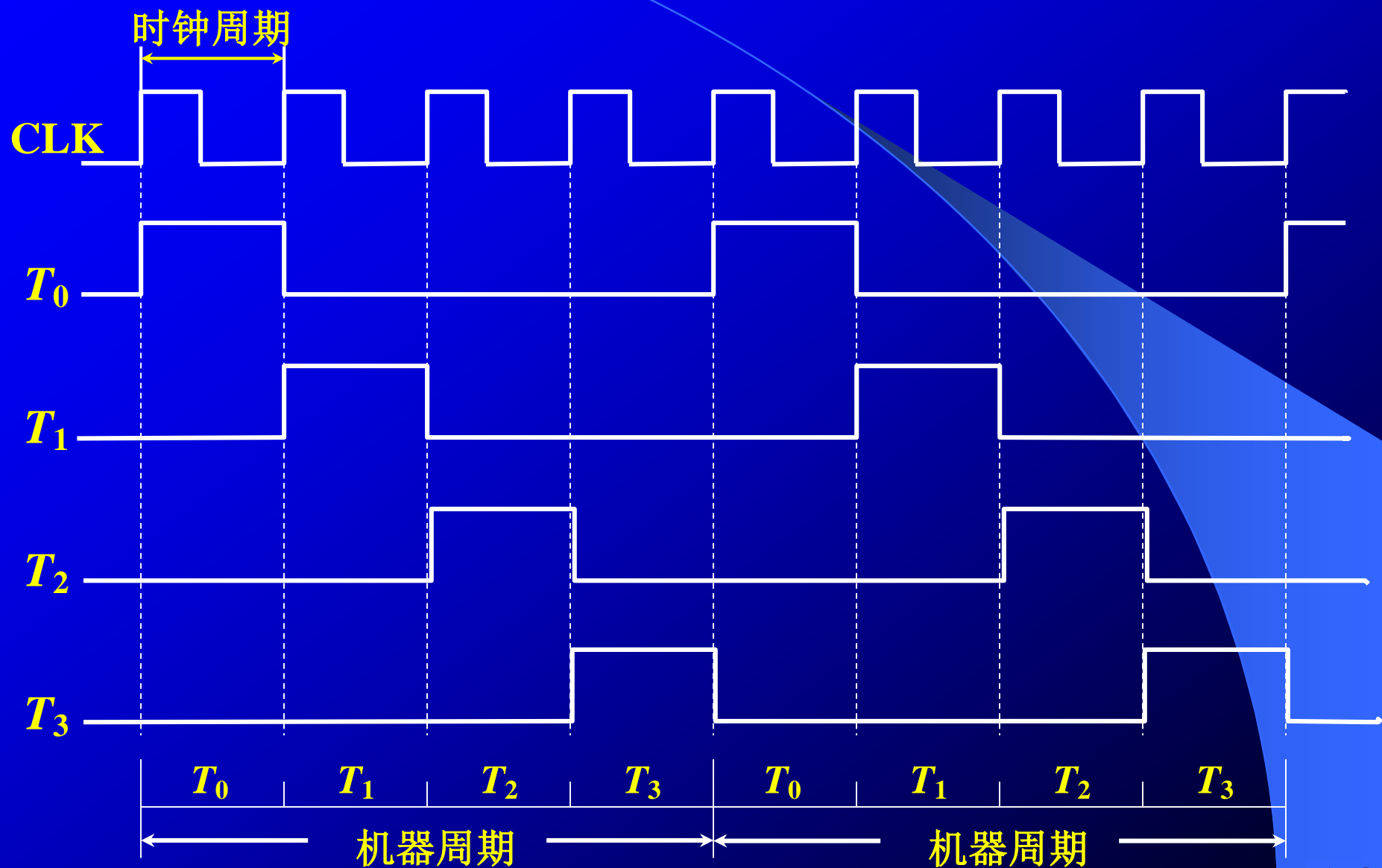
一、组合逻辑控制单元框图

1. CU 外特性



2. 节拍信号

10.1



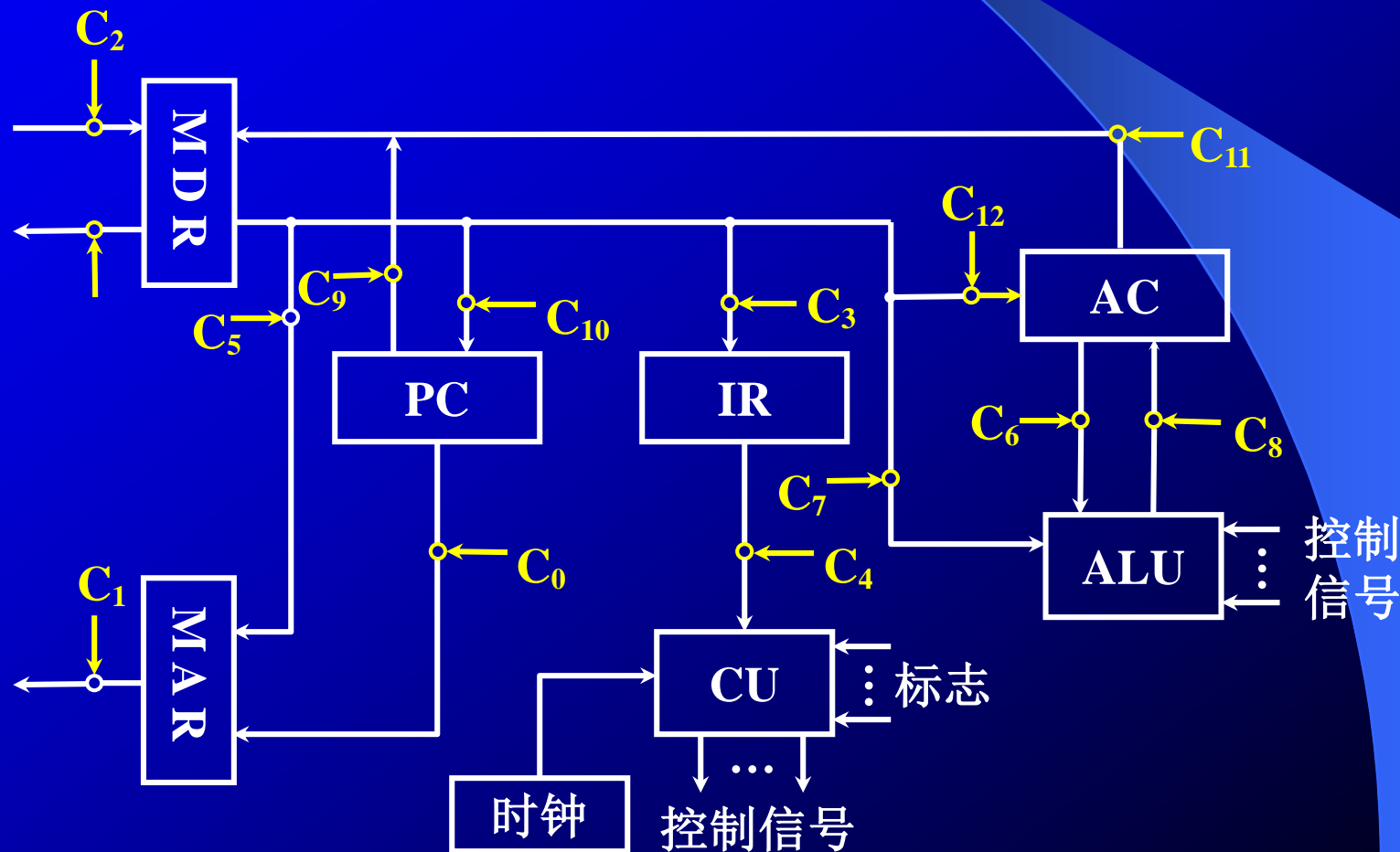
二、微操作的节拍安排

10.1

采用 同步控制方式

一个 机器周期 内有 3个节拍（时钟周期）

CPU 内部结构采用非总线方式



1. 安排微操作时序的原则

原则一 微操作的 先后顺序不得 随意 更改

原则二 被控对象不同 的微操作

尽量安排在 一个节拍 内完成

原则三 占用 时间较短 的微操作

尽量 安排在一个节拍 内完成

并允许有先后顺序

2. 取指周期 微操作的 节拍安排

T_0 PC \longrightarrow MAR

1 \longrightarrow R

原则二

T_1 M (MAR) \longrightarrow MDR

(PC) + 1 \longrightarrow PC

原则二

T_2 MDR \longrightarrow IR

OP (IR) \longrightarrow ID

原则三

3. 间址周期 微操作的 节拍安排

T_0 Ad (IR) \longrightarrow MAR

1 \longrightarrow R

T_1 M (MAR) \longrightarrow MDR

T_2 MDR \longrightarrow Ad (IR)

4. 执行周期 微操作的 节拍安排

10.1

① CLA T_0

T_1

T_2 $0 \longrightarrow AC$

② COM T_0

T_1

T_2 $\overline{AC} \longrightarrow AC$

③ SHR T_0

T_1

T_2 $L(AC) \longrightarrow R(AC)$

$AC_0 \longrightarrow AC_0$



④ CSL T_0
 T_1
 T_2 $R(AC) \longrightarrow L(AC)$ $AC_0 \longrightarrow AC_n$

⑤ STP T_0
 T_1
 T_2 $0 \longrightarrow G$

⑥ ADD X T_0 $Ad(IR) \longrightarrow MAR$ $1 \longrightarrow R$
 T_1 $M(MAR) \longrightarrow MDR$
 T_2 $(AC) + (MDR) \longrightarrow AC$

⑦ STA X T_0 $Ad(IR) \longrightarrow MAR$ $1 \longrightarrow W$
 T_1 $AC \longrightarrow MDR$
 T_2 $MDR \longrightarrow M(MAR)$

⑧ LDA X

T_0 Ad (IR) \longrightarrow MAR 1 \longrightarrow R

10.1

T_1 M (MAR) \longrightarrow MDR

T_2 MDR \longrightarrow AC

⑨ JMP X

T_0

T_1

T_2 Ad (IR) \longrightarrow PC

⑩ BAN X

T_0

T_1

T_2 $A_0 \cdot \text{Ad (IR)} + \bar{A}_0 \cdot \text{PC} \longrightarrow \text{PC}$



三、组合逻辑设计步骤

10.1

1. 列出操作时间表

工作周期标记	节拍	状态条件	微操作命令信号	CLA	COM	ADD	STA	LDA	JMP
FE 取指	T_0		$PC \rightarrow MAR$						
			$1 \rightarrow R$						
	T_1		$M(MAR) \rightarrow MDR$						
			$(PC) + 1 \rightarrow PC$						
	T_2		$MDR \rightarrow IR$						
			$OP(IR) \rightarrow ID$						
		I	$1 \rightarrow IND$						
		\bar{I}	$1 \rightarrow EX$						

间址特征



三、组合逻辑设计步骤

10.1

1. 列出操作时间表

工作周期标记	节拍	状态条件	微操作命令信号	CLA	COM	ADD	STA	LDA	JMP
IND 间址	T_0		Ad (IR) \rightarrow MAR						
			1 \rightarrow R						
	T_1		M(MAR) \rightarrow MDR						
	T_2		MDR \rightarrow Ad (IR)						
		$\overline{\text{IND}}$	1 \rightarrow EX						

间址周期标志



三、组合逻辑设计步骤

10.1

1. 列出操作时间表

工作周期标记	节拍	状态条件	微操作命令信号	CLA	COM	ADD	STA	LDA	JMP
EX 执行	T_0		$Ad(IR) \rightarrow MAR$						
			$1 \rightarrow R$						
			$1 \rightarrow W$						
	T_1		$M(MAR) \rightarrow MDR$						
			$AC \rightarrow MDR$						
	T_2		$(AC) + (MDR) \rightarrow AC$						
			$MDR \rightarrow M(MAR)$						
			$MDR \rightarrow AC$						
			$0 \rightarrow AC$						



三、组合逻辑设计步骤

10.1

1. 列出操作时间表

工作周期标记	节拍	状态条件	微操作命令信号	CLA	COM	ADD	STA	LDA	JMP
FE 取指	T_0		$PC \rightarrow MAR$	1	1	1	1	1	1
			$1 \rightarrow R$	1	1	1	1	1	1
	T_1		$M(MAR) \rightarrow MDR$	1	1	1	1	1	1
			$(PC) + 1 \rightarrow PC$	1	1	1	1	1	1
	T_2		$MDR \rightarrow IR$	1	1	1	1	1	1
			$OP(IR) \rightarrow ID$	1	1	1	1	1	1
		I	$1 \rightarrow IND$			1	1	1	1
		\bar{I}	$1 \rightarrow EX$	1	1	1	1	1	1



三、组合逻辑设计步骤

10.1

1. 列出操作时间表

工作周期标记	节拍	状态条件	微操作命令信号	CLA	COM	ADD	STA	LDA	JMP
IND 间址	T_0		Ad (IR) \rightarrow MAR			1	1	1	1
			1 \rightarrow R			1	1	1	1
	T_1		M(MAR) \rightarrow MDR			1	1	1	1
	T_2		MDR \rightarrow Ad (IR)			1	1	1	1
		$\overline{\text{IND}}$	1 \rightarrow EX			1	1	1	1



三、组合逻辑设计步骤

10.1

1. 列出操作时间表

工作周期标记	节拍	状态条件	微操作命令信号	CLA	COM	ADD	STA	LDA	JMP
EX 执行	T_0		Ad (IR) \rightarrow MAR			1	1	1	
			1 \rightarrow R			1		1	
			1 \rightarrow W				1		
	T_1		M(MAR) \rightarrow MDR			1		1	
			AC \rightarrow MDR				1		
	T_2		(AC)+(MDR) \rightarrow AC			1			
			MDR \rightarrow M(MAR)				1		
			MDR \rightarrow AC					1	
			0 \rightarrow AC	1					



2. 写出微操作命令的最简表达式

10.1

$$M(MAR) \longrightarrow MDR$$

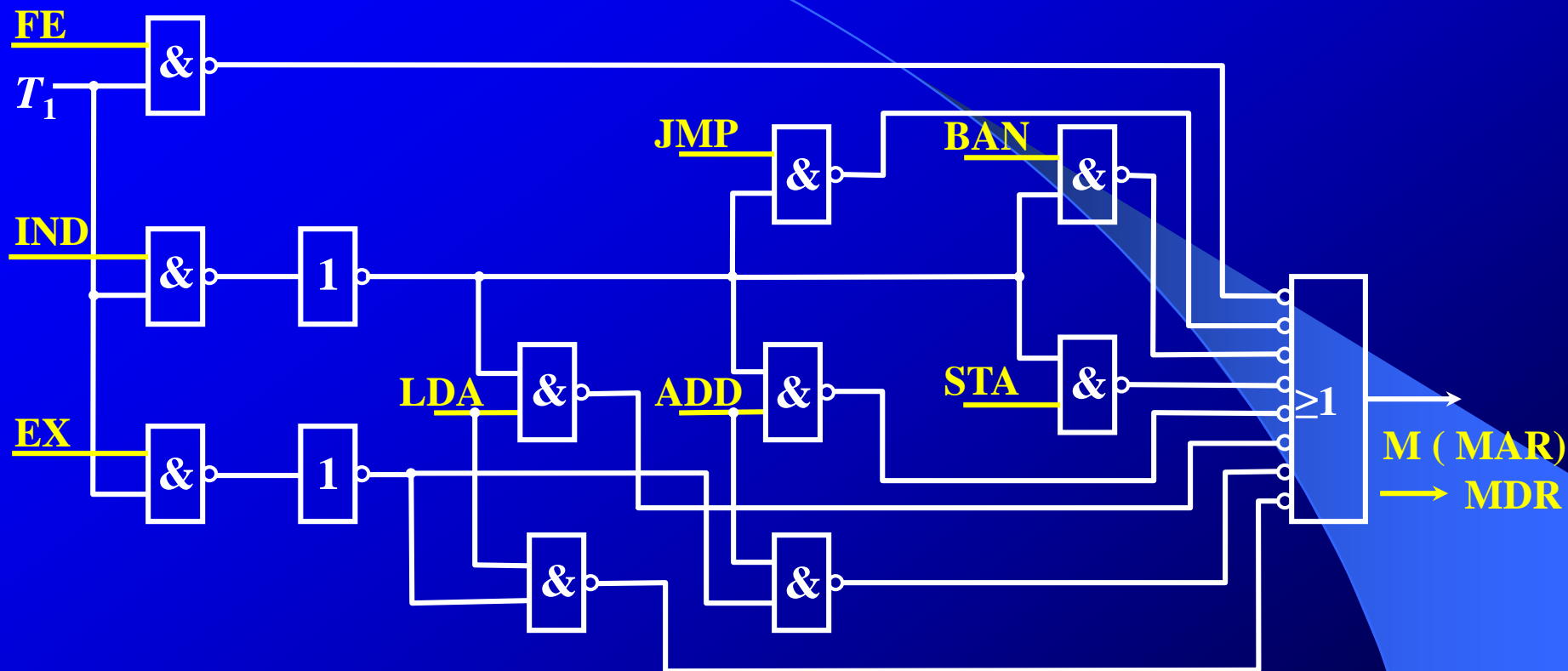
$$= FE \cdot T_1 + IND \cdot T_1 (ADD + STA + LDA + JMP + BAN) \\ + EX \cdot T_1 (ADD + LDA)$$

$$= T_1 \{ FE + IND (ADD + STA + LDA + JMP + BAN) \\ + EX (ADD + LDA) \}$$



3. 画出逻辑图

10.1



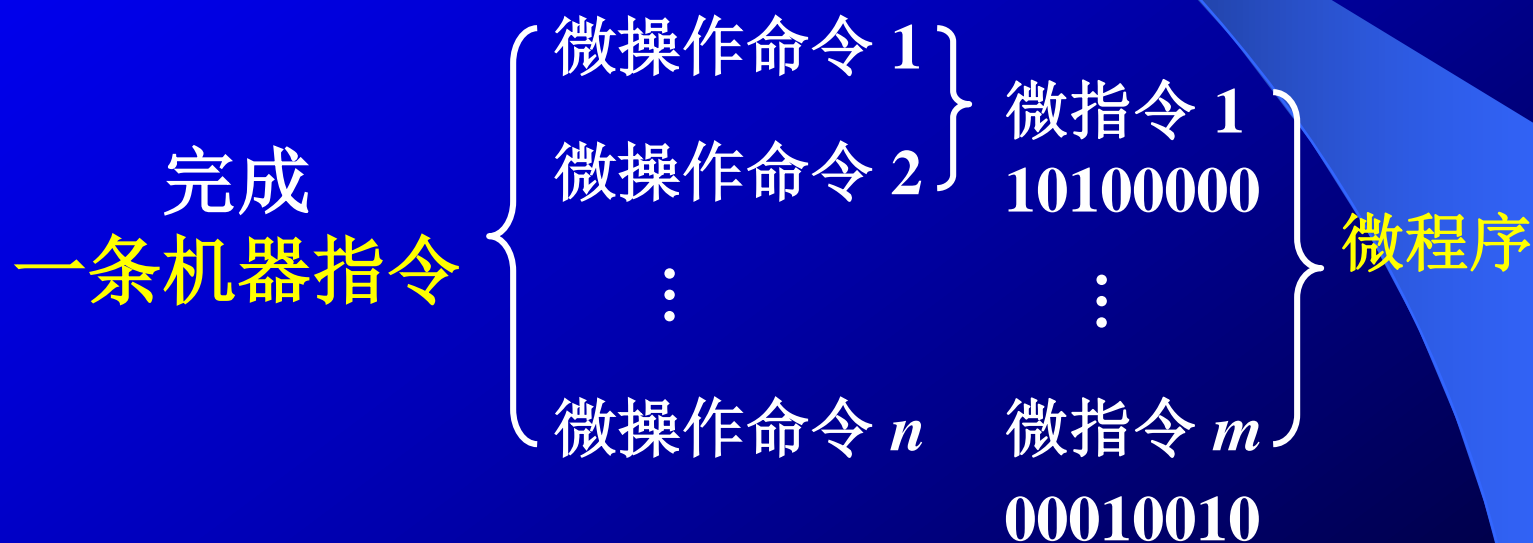
特点

- 思路清晰，简单明了
- 庞杂，调试困难，修改困难
- 速度快 (RISC)

10.2 微程序设计

一、微程序设计思想的产生

1951 英国剑桥大学教授 Wilkes



一条机器指令对应一个微程序

存入 **ROM**

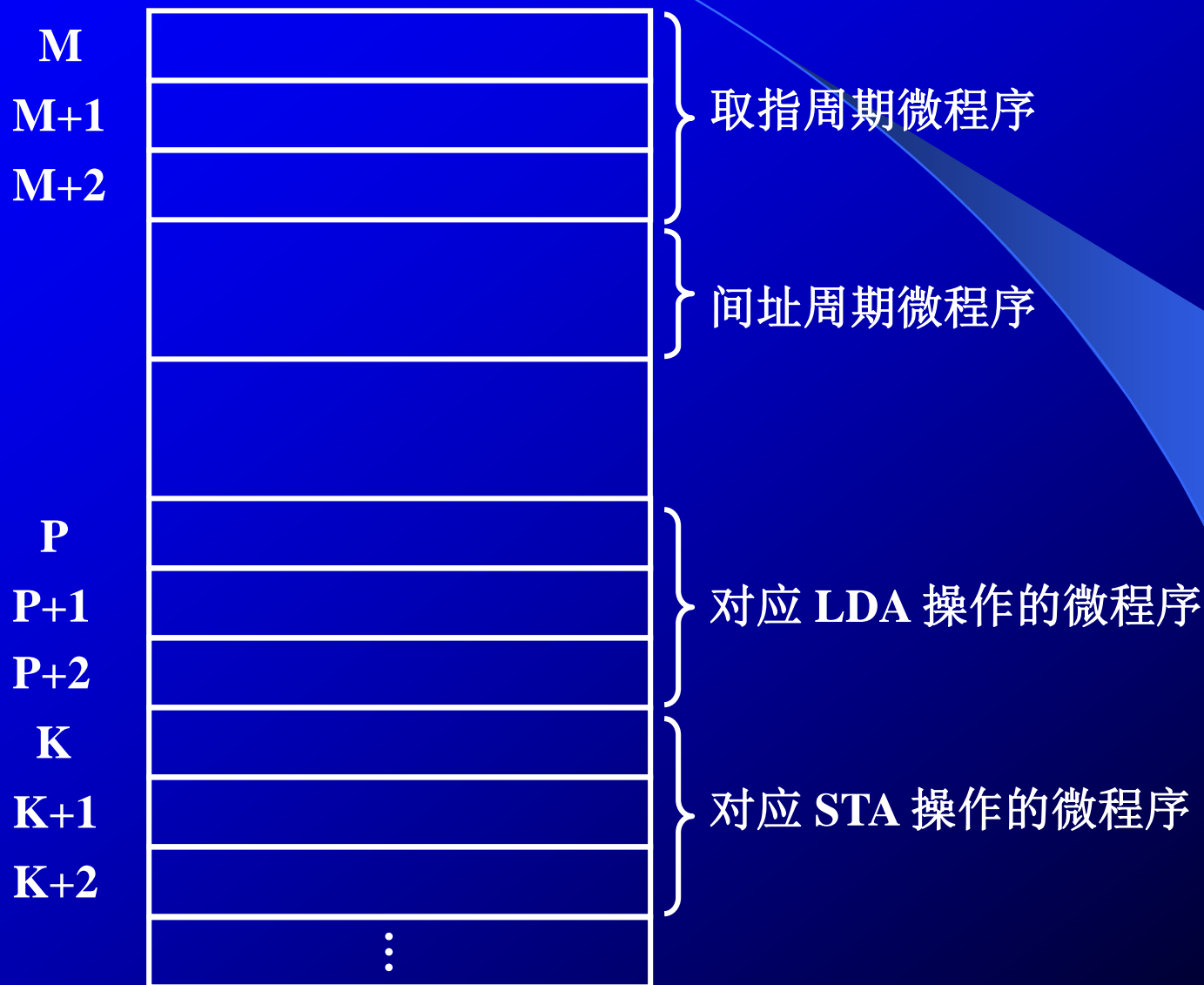
存储逻辑



二、微程序控制单元框图及工作原理

10.2

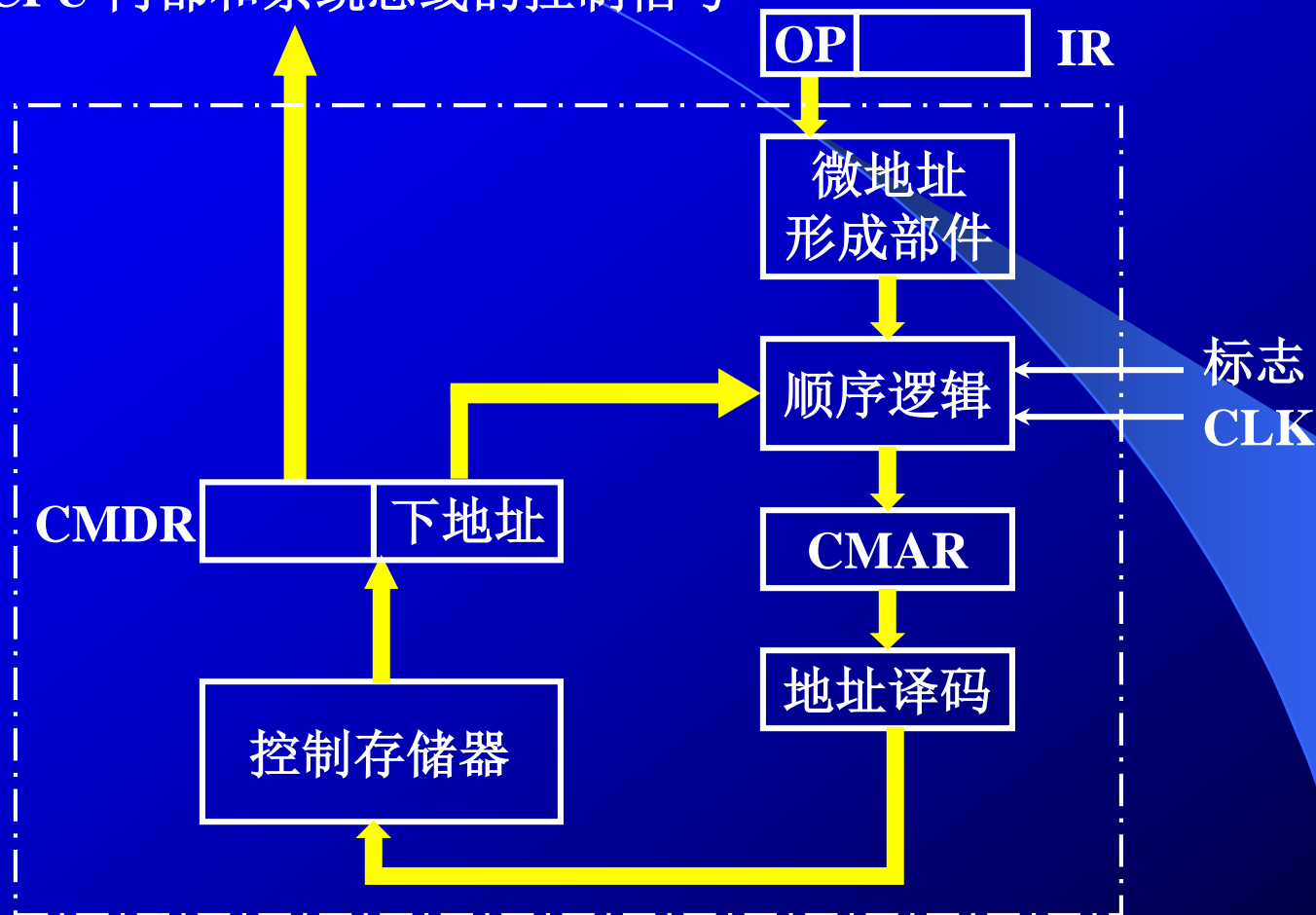
1. 机器指令对应的微程序



2. 微程序控制单元的基本框图

10.2

至 CPU 内部和系统总线的控制信号

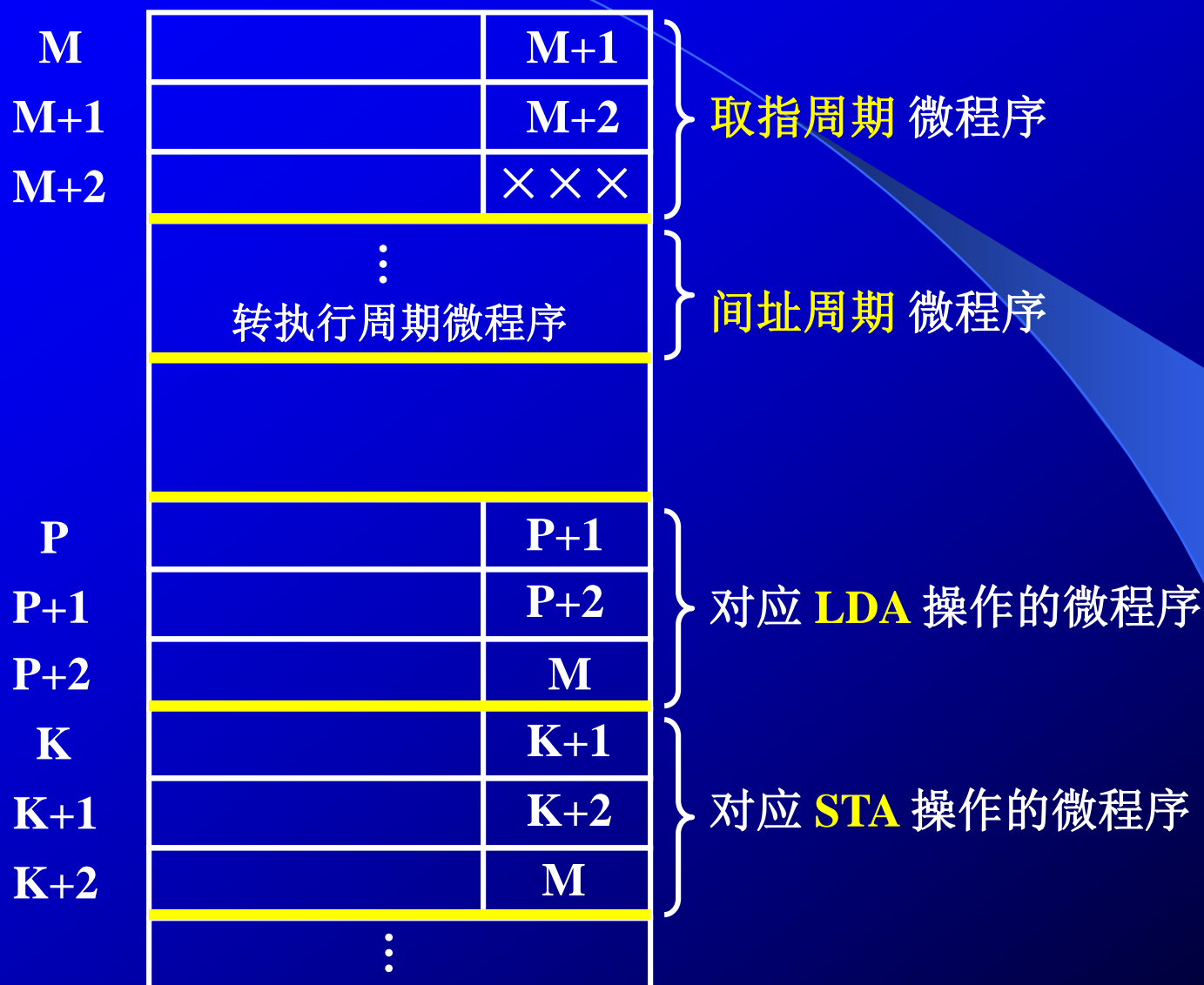


微指令基本格式



二、微程序控制单元框图及工作原理

10.2



3. 工作原理

10.2



3. 工作原理

10.2

(1) 取指阶段 执行取指微程序

$M \rightarrow CMAR$

$CM(CMAR) \rightarrow CMDR$

由 CMDR 发命令

形成下条微指令地址 $M + 1$

$Ad(CMDR) \rightarrow CMAR$

$CM(CMAR) \rightarrow CMDR$

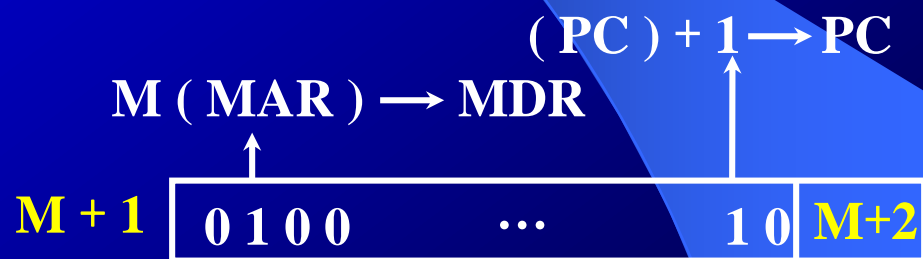
由 CMDR 发命令

形成下条微指令地址 $M + 2$

$Ad(CMDR) \rightarrow CMAR$

$CM(CMAR) \rightarrow CMDR$

由 CMDR 发命令



(2) 执行阶段 执行 LDA 微程序

10.2

OP (IR) \longrightarrow 微地址形成部件 \longrightarrow **CMAR** (**P** \longrightarrow **CMAR**)

CM (CMAR) \longrightarrow **CMDR**

由 **CMDR** 发命令

形成下条微指令地址 **Ad (CMDR)** \longrightarrow **CMAR**

CM (CMAR) \longrightarrow **CMDR**

由 **CMDR** 发命令

形成下条微指令地址 **Ad (CMDR)** \longrightarrow **CMAR**

CM (CMAR) \longrightarrow **CMDR**

由 **CMDR** 发命令

形成下条微指令地址 **Ad (CMDR)** \longrightarrow **CMAR**

Ad (IR) \longrightarrow **MAR**

1 \longrightarrow **R**



M (MAR) \longrightarrow **MDR**



MDR \longrightarrow **AC**



(**M** \longrightarrow **CMAR**)

(3) 取指阶段 执行取指微程序

 $M \rightarrow \text{CMAR}$ $\text{CM}(\text{CMAR}) \rightarrow \text{CMDR}$

由 CMDR 发命令

⋮



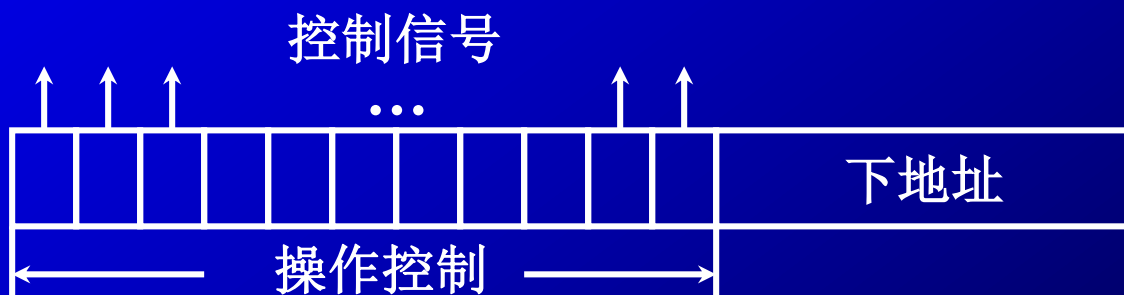
全部微指令存在 CM 中，程序执行过程中 只需读出

- 关键
- 微指令的 操作控制字段如何形成微操作命令
 - 微指令的 后续地址如何形成

三、微指令的编码方式（控制方式）

1. 直接编码（直接控制）方式

在微指令的操作控制字段中，
每一位代表一个微操作命令

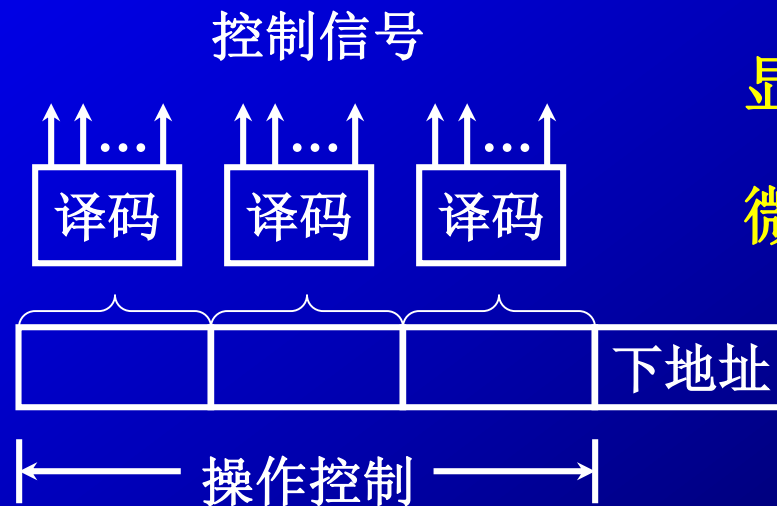


速度最快

某位为 “1” 表示该控制信号有效

2. 字段直接编码方式

将微指令的控制字段分成若干“段”，
每段经译码后发出控制信号



显式编码

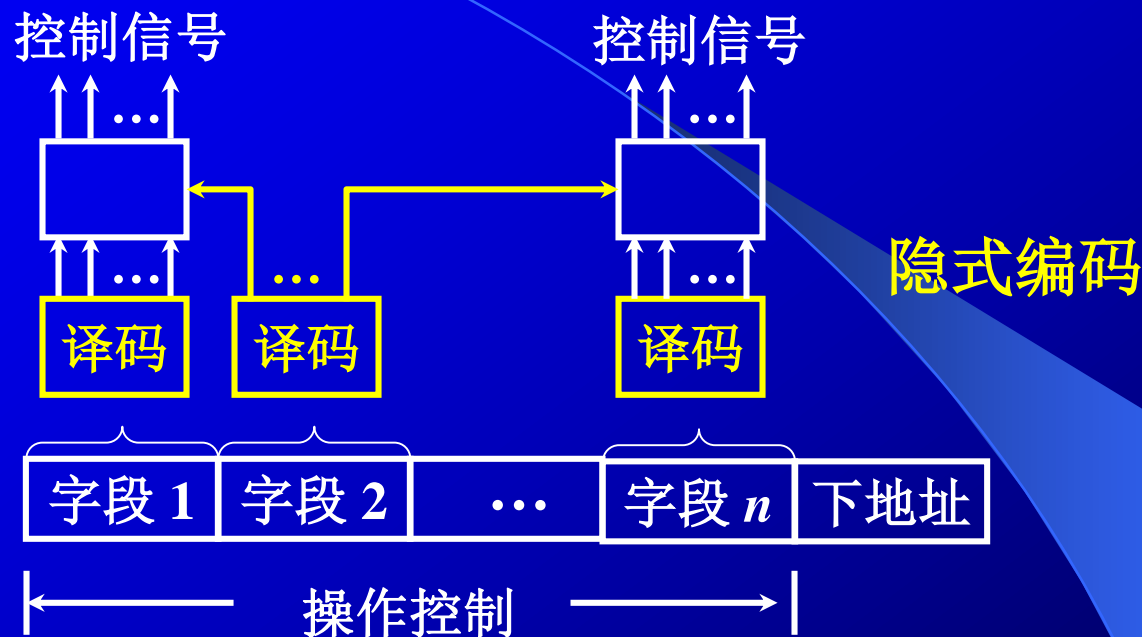
微程序执行速度较慢

每个字段中的命令是 **互斥** 的

缩短 了微指令 **字长**，**增加** 了译码 **时间**

3. 字段间接编码方式

10.2



4. 混合编码

直接编码和字段编码（直接和间接）混合使用

5. 其他

四、微指令序列地址的形成

10.2

1. 微指令的 **下地址字段** 指出
2. 根据机器指令的 **操作码** 形成
3. **增量计数器**

$$(\text{CMAR}) + 1 \rightarrow \text{CMAR}$$

4. 分支转移

操作控制字段	转移方式	转移地址
--------	------	------

转移方式

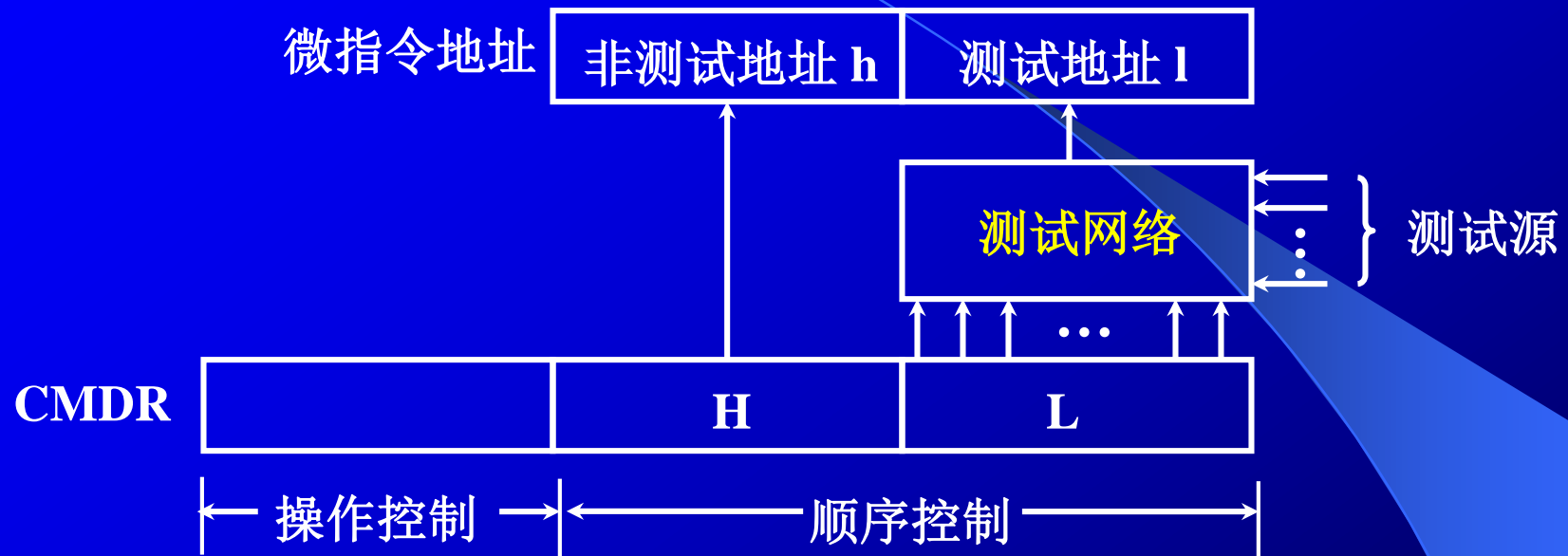
指明判别条件

转移地址

指明转移成功后的去向



5. 通过测试网络



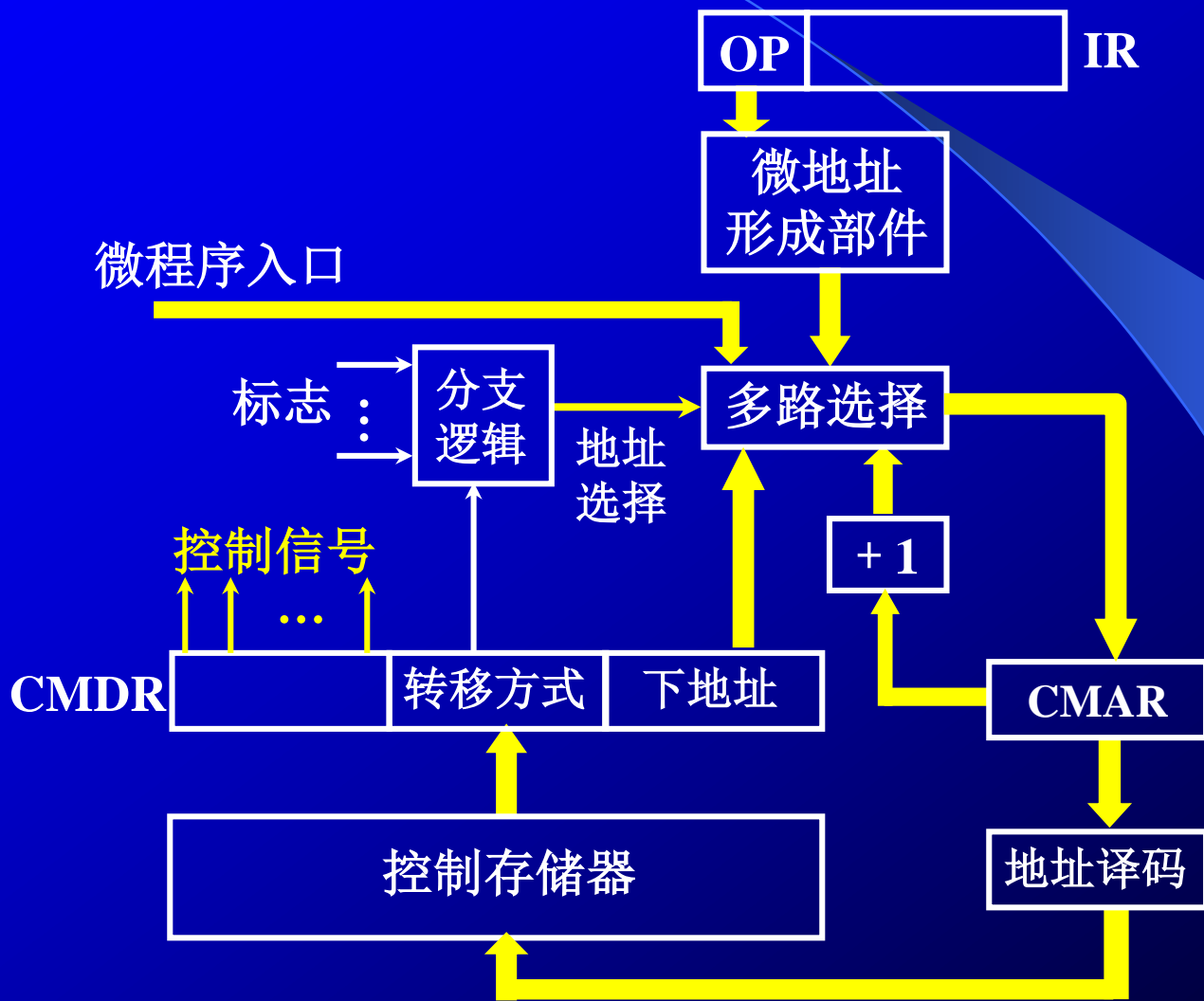
6. 由硬件产生微程序入口地址

第一条微指令地址 由专门 **硬件** 产生

中断周期 由 **硬件** 产生 **中断周期微程序首地址**

7. 后续微指令地址形成方式原理图

10.2



五、微指令格式

1. 水平型微指令

一次能定义并执行多个并行操作

如 直接编码、字段直接编码、字段间接编码、
直接和字段混合编码

2. 垂直型微指令

类似机器指令操作码 的方式

由微操作码字段规定微指令的功能

3. 两种微指令格式的比较

- (1) 水平型微指令比垂直型微指令 并行操作能力强，
灵活性强
- (2) 水平型微指令执行一条机器指令所要的
微指令 数目少，速度快
- (3) 水平型微指令 用较短的微程序结构换取较长的
微指令结构
- (4) 水平型微指令与机器指令 差别大

六、静态微程序设计和动态微程序设计

10.2

静态 微程序无须改变, 采用 **ROM**

动态 通过 **改变微指令** 和 **微程序** 改变机器指令,
有利于仿真, 采用 **EPROM**

七、毫微程序设计

1. 毫微程序设计的基本概念

微程序设计 用 **微程序** 解释机器指令

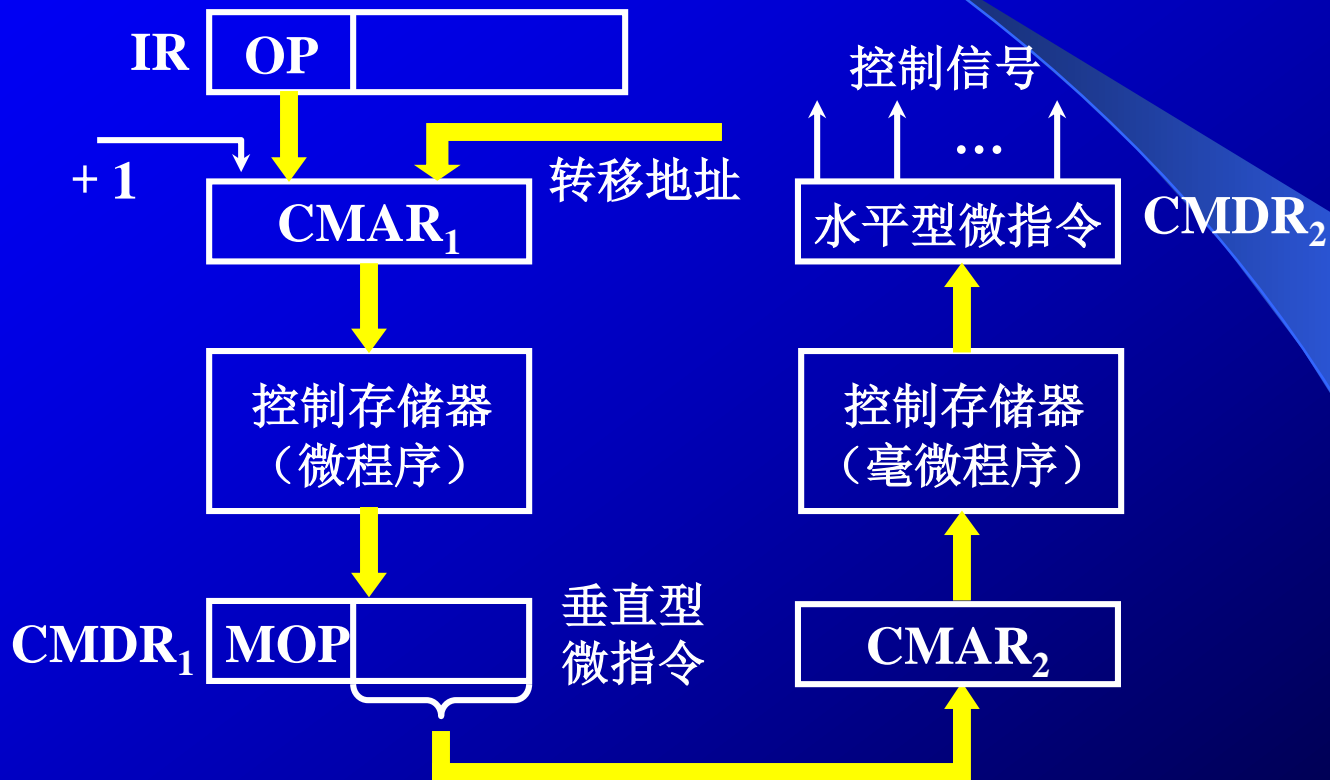
毫微程序设计 用 **毫微程序** 解释微程序

毫微指令与微指令 的关系好比 **微指令与机器指令** 的关系



2. 毫微程序控制存储器的基本组成

10.2



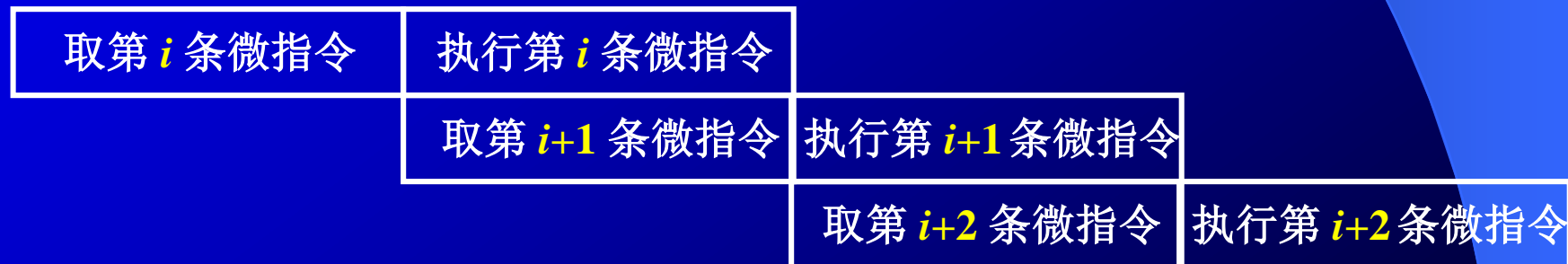
八、串行微程序控制和并行微程序控制

10.2

串行 微程序控制



并行 微程序控制



九、微程序设计举例

10.2

1. 写出对应机器指令的微操作及节拍安排

假设 CPU 结构与组合逻辑相同

(1) 取指阶段微操作分析

3 条微指令

T_0 $PC \rightarrow MAR$ $1 \rightarrow R$

T_1 $M(MAR) \rightarrow MDR$ $(PC) + 1 \rightarrow PC$

T_2 $MDR \rightarrow IR$ $OP(IR) \rightarrow$ 微地址形成部件

若需考虑如何安排这条微指令？

则取指操作需 3 条微指令

$Ad(CMDR) \rightarrow CMAR$

$OP(IR) \rightarrow$ 微地址形成部件 $\rightarrow CMAR$

(2) 取指阶段的微操作及节拍安排

考虑到需要 形成后续微指令的地址

T_0 $PC \longrightarrow MAR$ $1 \longrightarrow R$

T_1 $Ad (CMDR) \longrightarrow CMAR$

T_2 $M (MAR) \longrightarrow MDR$ $(PC) + 1 \longrightarrow PC$

T_3 $Ad (CMDR) \longrightarrow CMAR$

T_4 $MDR \longrightarrow IR$ $OP (IR) \longrightarrow$ 微地址形成部件

T_5 $OP (IR) \longrightarrow$ 微地址形成部件 $\longrightarrow CMAR$

(3) 执行阶段的微操作及节拍安排

10.2

考虑到需形成后续微指令的地址

取指微程序的入口地址 M
由微指令下地址字段指出

- 非访存指令

- ① CLA 指令

$T_0 \quad 0 \longrightarrow AC$

$T_1 \quad \mathbf{Ad (CMDR) \longrightarrow CMAR}$

- ② COM 指令

$T_0 \quad \overline{AC} \longrightarrow AC$

$T_1 \quad \mathbf{Ad (CMDR) \longrightarrow CMAR}$

③ SHR 指令

$$T_0 \quad L(AC) \longrightarrow R(AC) \quad AC_0 \longrightarrow AC_0$$
$$T_1 \quad \text{Ad}(\text{CMDR}) \longrightarrow \text{CMAR}$$

④ CSL 指令

$$T_0 \quad R(AC) \longrightarrow L(AC) \quad AC_0 \longrightarrow AC_n$$
$$T_1 \quad \text{Ad}(\text{CMDR}) \longrightarrow \text{CMAR}$$

⑤ STP 指令

$$T_0 \quad 0 \longrightarrow G$$
$$T_1 \quad \text{Ad}(\text{CMDR}) \longrightarrow \text{CMAR}$$

• 访存指令

10.2

⑥ ADD 指令

T_0 $\text{Ad}(\text{IR}) \longrightarrow \text{MAR}$ $1 \longrightarrow \text{R}$

T_1 $\text{Ad}(\text{CMDR}) \longrightarrow \text{CMAR}$

T_2 $\text{M}(\text{MAR}) \longrightarrow \text{MDR}$

T_3 $\text{Ad}(\text{CMDR}) \longrightarrow \text{CMAR}$

T_4 $(\text{AC}) + (\text{MDR}) \longrightarrow \text{AC}$

T_5 $\text{Ad}(\text{CMDR}) \longrightarrow \text{CMAR}$

⑦ STA 指令

T_0 $\text{Ad}(\text{IR}) \longrightarrow \text{MAR}$ $1 \longrightarrow \text{W}$

T_1 $\text{Ad}(\text{CMDR}) \longrightarrow \text{CMAR}$

T_2 $\text{AC} \longrightarrow \text{MDR}$

T_3 $\text{Ad}(\text{CMDR}) \longrightarrow \text{CMAR}$

T_4 $\text{MDR} \longrightarrow \text{M}(\text{MAR})$

T_5 $\text{Ad}(\text{CMDR}) \longrightarrow \text{CMAR}$



⑧ LDA 指令

10.2

T_0 $\text{Ad (IR)} \longrightarrow \text{MAR}$ $1 \longrightarrow \text{R}$

T_1 **$\text{Ad (CMDR)} \longrightarrow \text{CMAR}$**

T_2 $\text{M (MAR)} \longrightarrow \text{MDR}$

T_3 **$\text{Ad (CMDR)} \longrightarrow \text{CMAR}$**

T_4 $\text{MDR} \longrightarrow \text{AC}$

T_5 **$\text{Ad (CMDR)} \longrightarrow \text{CMAR}$**



- 转移类指令

- ⑨ JMP 指令

$T_0 \quad \text{Ad (IR)} \longrightarrow \text{PC}$

$T_1 \quad \text{Ad (CMDR)} \longrightarrow \text{CMAR}$

- ⑩ BAN 指令

$T_0 \quad A_0 \cdot \text{Ad (IR)} + \bar{A}_0 \cdot (\text{PC}) \longrightarrow \text{PC}$

$T_1 \quad \text{Ad (CMDR)} \longrightarrow \text{CMAR}$

全部微操作 20个

微指令 38条



2. 确定微指令格式

(1) 微指令的编码方式

采用直接控制

(2) 后续微指令的地址形成方式

由机器指令的操作码通过微地址形成部件形成

由微指令的下地址字段直接给出

(3) 微指令字长

由 20 个微操作

确定 操作控制字段 最少 20 位

由 38 条微指令

确定微指令的 下地址字段 为 6 位

微指令字长 可取 $20 + 6 = 26$ 位



(4) 微指令字长的确定

10.2

38 条微指令中有 19 条
是关于后续微指令地址 \rightarrow CMAR

其中 $\begin{cases} 1 \text{ 条} & \text{OP}(\text{IR}) \rightarrow \text{微地址形成部件} \rightarrow \text{CMAR} \\ 18 \text{ 条} & \text{Ad}(\text{CMDR}) \rightarrow \text{CMAR} \end{cases}$

若用 $\text{Ad}(\text{CMDR})$ 直接送控存地址线

则省去了输至 CMAR 的时间, 省去了 CMAR

同理 $\text{OP}(\text{IR}) \rightarrow \text{微地址形成部件} \rightarrow \text{控存地址线}$

可省去 19 条微指令, 2 个微操作

$$38 - 19 = 19$$

下地址字段最少取 5 位

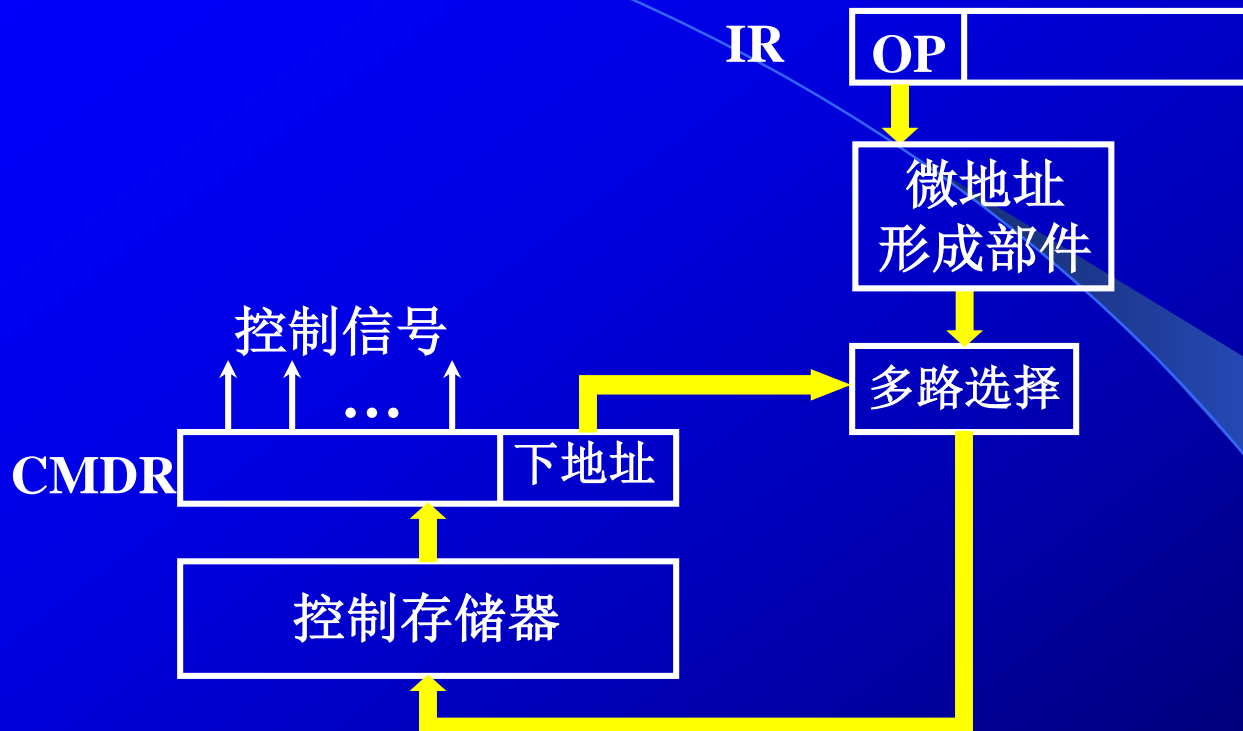
$$20 - 2 = 18$$

操作控制字段最少取 18 位



(5) 省去了 CMAR 的控制存储器

10.2



考虑留有一定的余量

取操作控制字段
下地址字段

18 位 → 24 位
5 位 → 6 位 } 共 30 位

(6) 定义微指令操作控制字段每一位的微操作



3. 编写微指令码点

10.2

微程序 名称	微指令 地址 (八进制)	微指令（二进制代码）														
		操作控制字段									下地址字段					
取指		0	1	2	3	4	...	10	...	23	24	25	26	27	28	29
	00	1	1								0	0	0	0	0	1
	01			1	1						0	0	0	0	1	0
	02					1					×	×	×	×	×	×
CLA	03										0	0	0	0	0	0
COM	04										0	0	0	0	0	0
ADD	10		1					1			0	0	1	0	0	1
	11			1							0	0	1	0	1	0
	12										0	0	0	0	0	0
LDA	16		1					1			0	0	1	1	1	1
	17			1							0	1	0	0	0	0
	20										0	0	0	0	0	0

作业

- 习题： 10.2, 10.7, 10.8, 10.15, 10.21