

计算机组成原理 (第十讲)



厦门大学信息学院软件工程系 曾文华
2021年6月9日



第4篇 控制单元

第9章 控制单元的功能

第10章 控制单元的设计



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

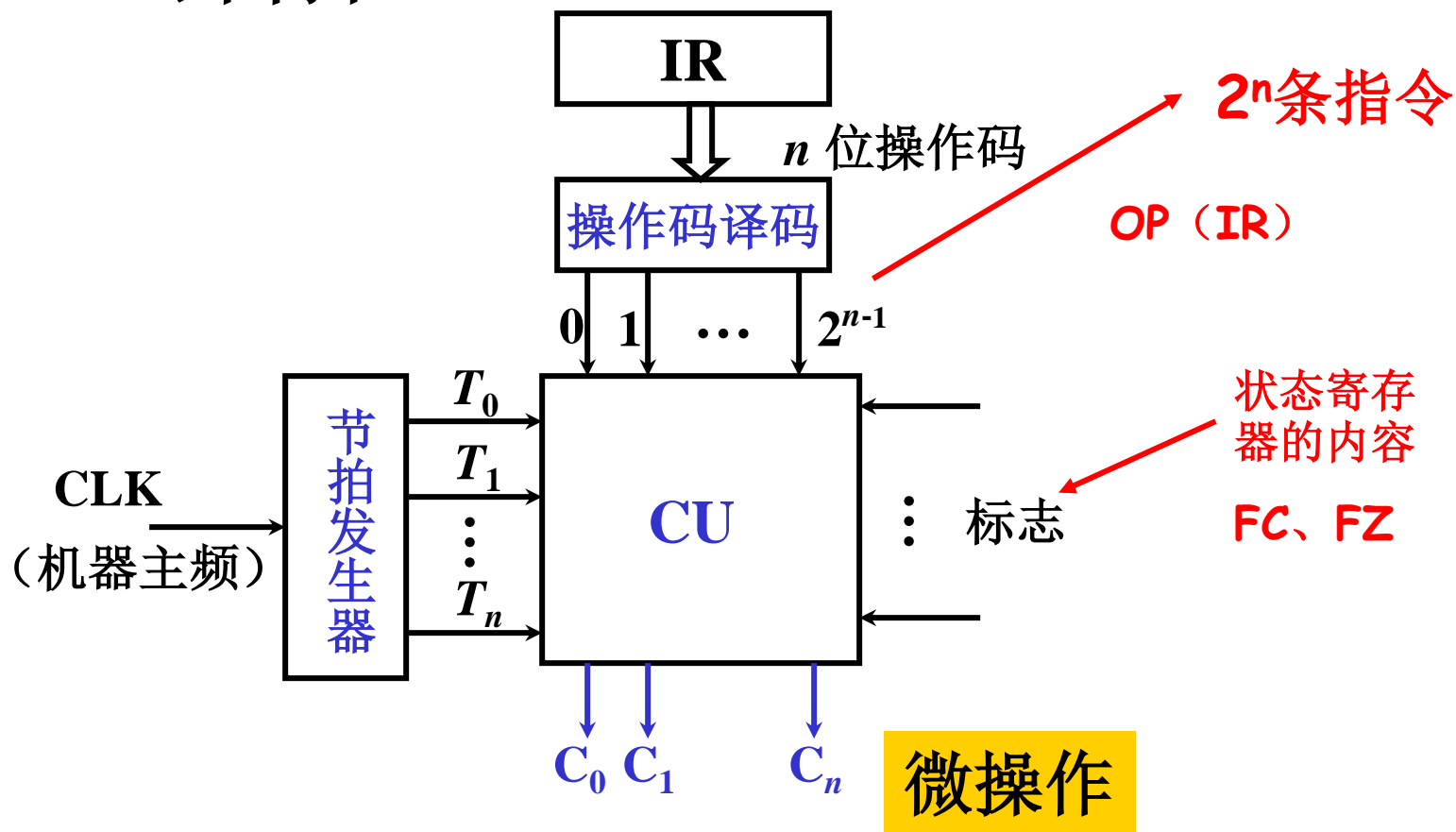


10.1 组合逻辑设计

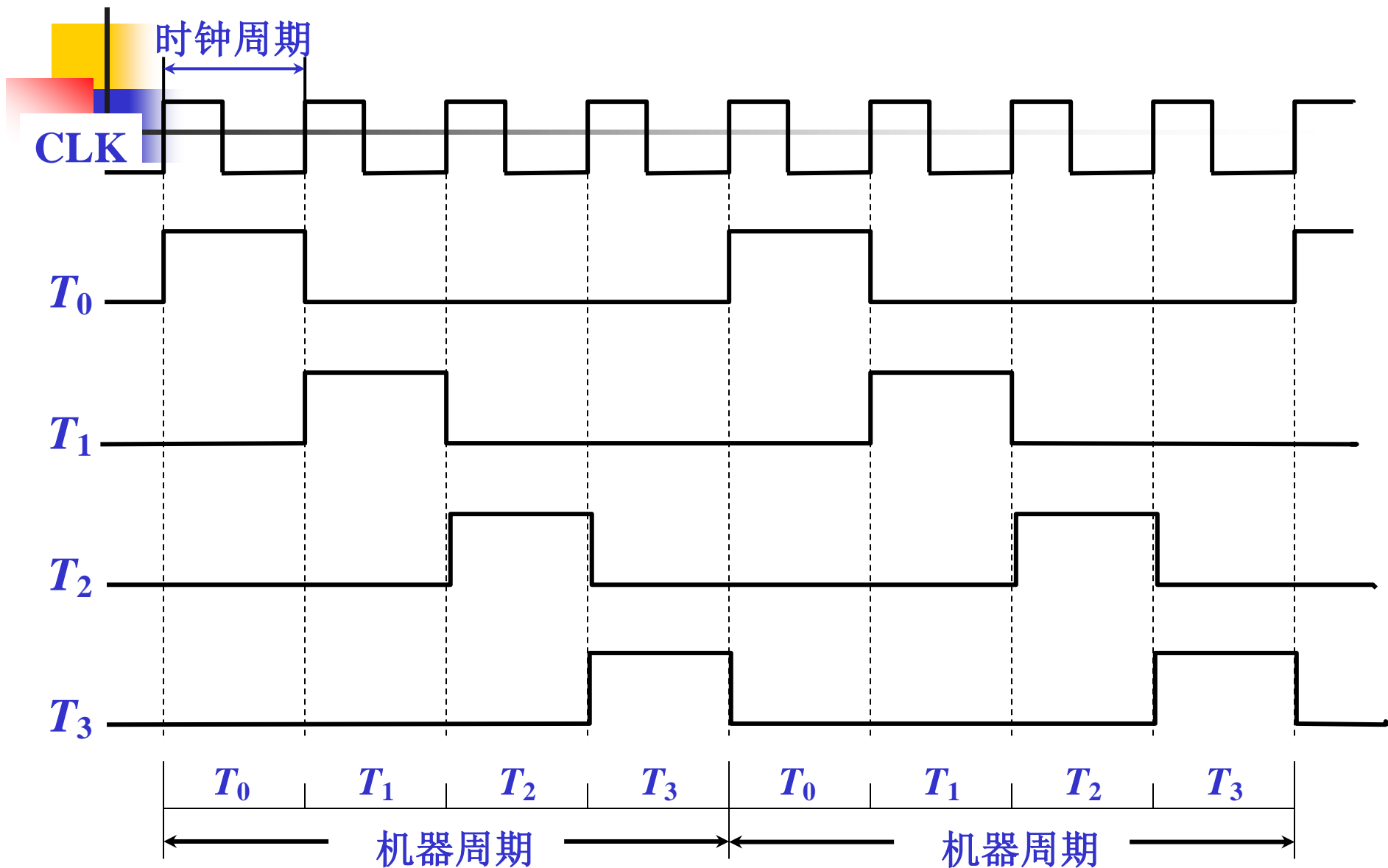
- 一、组合逻辑控制单元框图
- 二、微操作的节拍安排
- 三、组合逻辑设计步骤

一、组合逻辑控制单元框图

1. CU 外特性



2. 节拍信号



二、微操作的节拍安排

采用 同步控制方式

一个 机器周期 内有 3 个节拍（时钟周期, T_0, T_1, T_2 ）

CPU 内部结构采用非总线方式

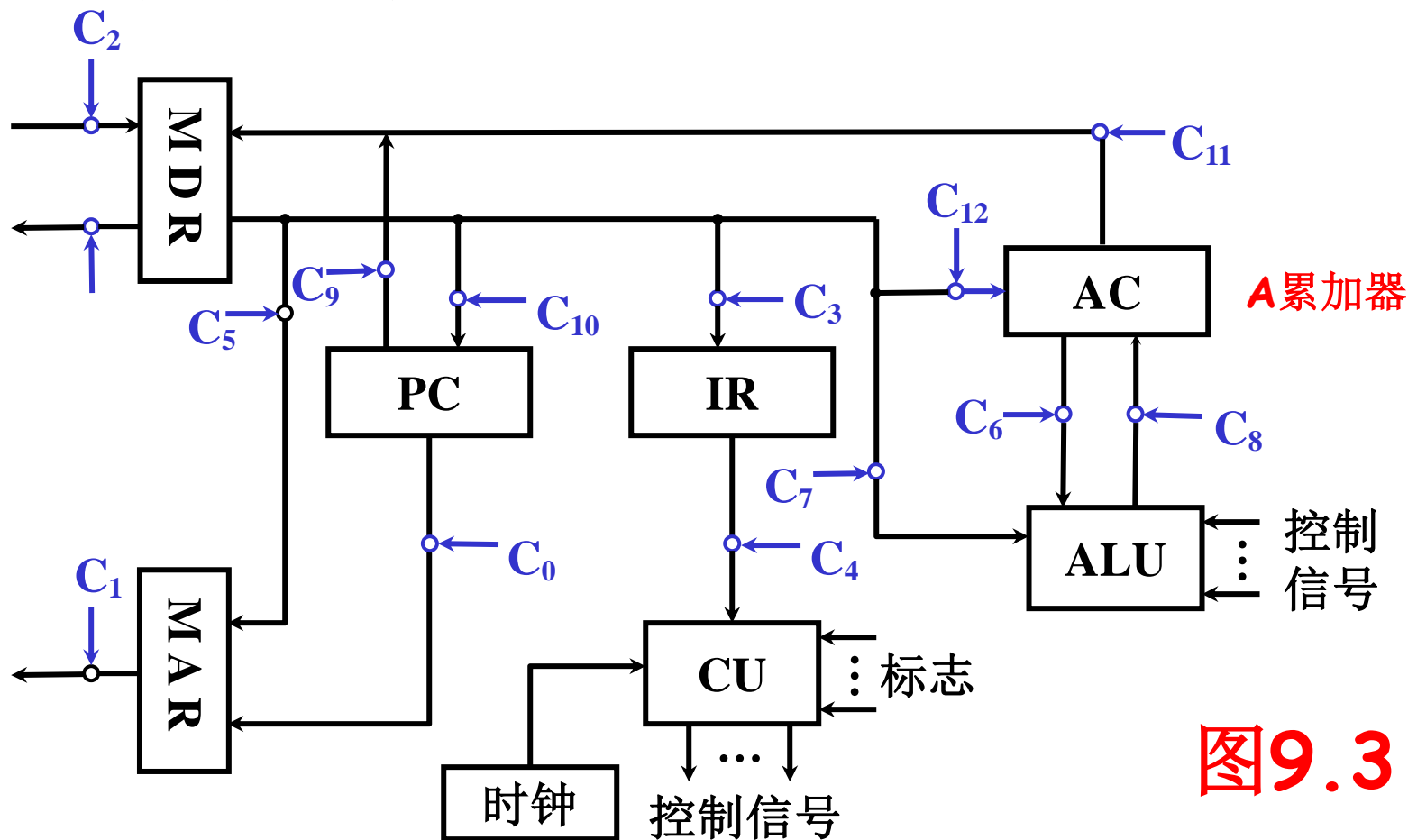


图9.3

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，可以节省时间

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 $PC \rightarrow MAR$

原则二

$1 \rightarrow R$

6个微操作安
排在3个时钟
周期内

T_1 $M(MAR) \rightarrow MDR$

原则二

$(PC) + 1 \rightarrow PC$

T_2 $MDR \rightarrow IR$

原则三

$OP(IR) \rightarrow ID(\text{指令译码})$

第八章的取指周期

- ① $PC \rightarrow MAR \rightarrow \text{地址线}$
- ② $1 \rightarrow R$
- ③ $M(MAR) \rightarrow MDR$
- ④ $MDR \rightarrow IR$
- ⑤ $OP(IR) \rightarrow CU$
- ⑥ $(PC) + 1 \rightarrow PC$

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，可以节省时间

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

3. 间址周期 微操作的 节拍安排

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

$1 \longrightarrow \text{R}$

T_1 $\text{M (MAR)} \longrightarrow \text{MDR}$

T_2 $\text{MDR} \longrightarrow \text{Ad (IR)}$

4个微操作安
排在3个时钟
周期内

第八章的间址周期

① 指令形式地址 $\longrightarrow \text{MAR}$

即: $\text{Ad (IR)} \longrightarrow \text{MAR}$

② $1 \longrightarrow \text{R}$

③ $\text{M (MAR)} \longrightarrow \text{MDR}$

④ $\text{MDR} \longrightarrow \text{Ad (IR)}$

4. 执行周期 微操作的 节拍安排

共10条
指令

① CLA T_0

T_1

T_2 $0 \longrightarrow AC$

② COM T_0

取反操作

T_1

T_2 $\overline{AC} \longrightarrow AC$

③ SHR T_0

算术右移

T_1

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

$AC_0 \longrightarrow AC_0$

④ CSL T_0

循环左移

T_1

T_2

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

$AC_0 \longrightarrow AC_n$

⑤ STP T_0

停机

T_1

T_2

$0 \longrightarrow G$

⑥ ADD X T_0

$Ad(IR) \longrightarrow MAR$

$1 \longrightarrow R$

$AC+(X) \rightarrow AC$

T_1

$M(MAR) \longrightarrow MDR$

T_2

$(AC) + (MDR) \longrightarrow AC$

⑦ STA X T_0

$Ad(IR) \longrightarrow MAR$

$1 \longrightarrow W$

$AC \rightarrow (X)$

T_1

$AC \longrightarrow MDR$

T_2

$MDR \longrightarrow M(MAR)$

⑧ LDA X T_0 $\text{Ad (IR)} \longrightarrow \text{MAR}$ $1 \rightarrow \text{R}$

(X) \rightarrow AC

T_1 $\text{M (MAR)} \longrightarrow \text{MDR}$

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

⑨ JMP X T_0

无条件转移

T_1

T_2 $\text{Ad (IR)} \longrightarrow \text{PC}$

⑩ BAN X T_0

有条件转移指令
(负则转)

T_1

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

5. 中断周期 微操作的 节拍安排

由中断隐指令完成

T_0 $0 \rightarrow \text{MAR}$ $1 \rightarrow \text{W}$ 硬件关中断

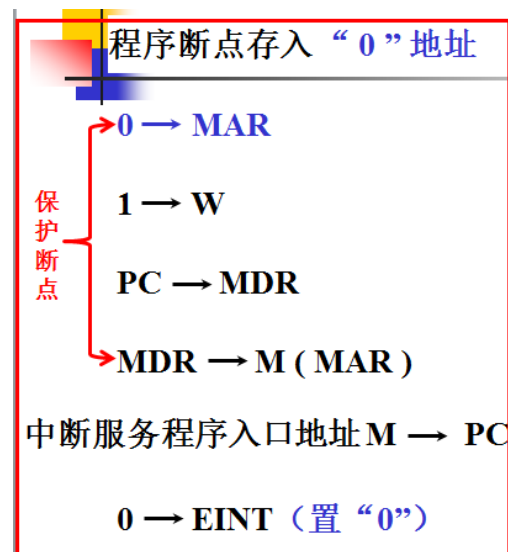
T_1 $\text{PC} \rightarrow \text{MDR}$

6个微操作安
排在3个时钟
周期内

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

向量地址 $\rightarrow \text{PC}$

第八章的中断周期



通常 $(PC)+1 \rightarrow PC$ ，是PC自动完成的，这里不是！需要由ALU完成

- **例10.1**：设CPU中各部件及其相互连接关系如图10.2所示：
- (1)假设要求在取指周期由ALU完成 $(PC)+1 \rightarrow PC$ 的操作，要求以最少的节拍写出取指周期全部微操作命令及节拍安排。
- (2)写出指令ADD #a(#为立即寻址特征，隐含的操作数在ACC中)在执行阶段所需的微操作命令及节拍安排。
- 解：

ADD ACC,80H

#：立即寻址
@：间接寻址
*：相对寻址

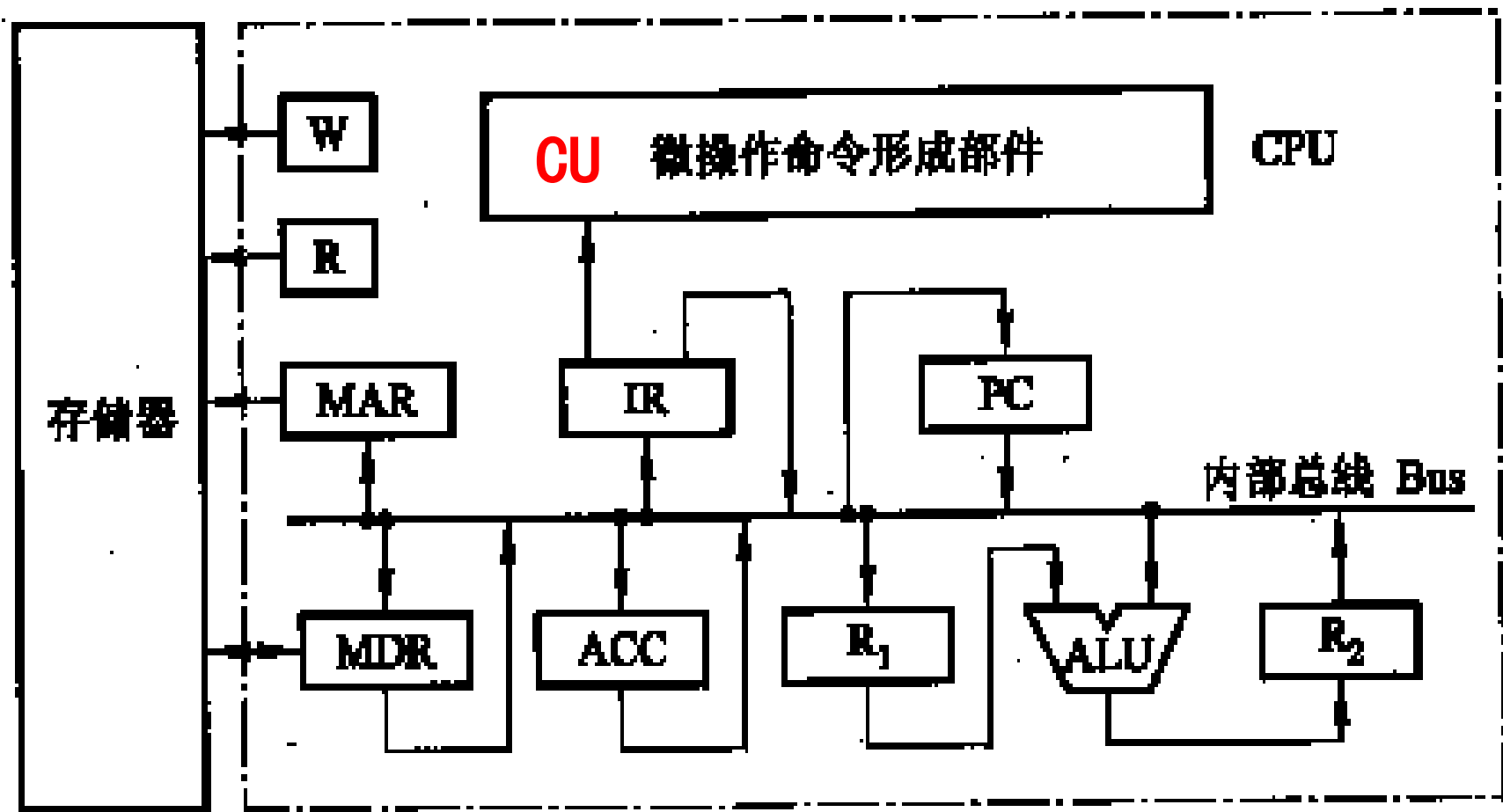


图 10.2 例 10.1 CPU 内部结构框图

一个机器周期内有 4 个节拍（时钟周期, T_0, T_1, T_2, T_3 ）
或3个节拍（时钟周期, T_0, T_1, T_2 ）

■ **(1)取指周期**的微操作命令及节拍安排如下:

■ **T_0 $PC \rightarrow Bus \rightarrow MAR, 1 \rightarrow R$**

■ **T_1 $M(MAR) \rightarrow MDR,$
 $(PC) \rightarrow Bus \rightarrow ALU_{+1} \rightarrow R2$**

■ **T_2 $MDR \rightarrow Bus \rightarrow IR,$
 $OP(IR) \rightarrow$ 微操作命令形成部件(CU)**

■ **T_3 $R2 \rightarrow Bus \rightarrow PC$**

由ALU完成
 $(PC)+1 \rightarrow PC$
的操作

取指周期

T_0	$PC \rightarrow MAR$
	$1 \rightarrow R$
T_1	$M(MAR) \rightarrow MDR$
	$(PC) + 1 \rightarrow PC$
T_2	$MDR \rightarrow IR$
	$OP(IR) \rightarrow ID(\text{指令译码})$



ADD ACC, 80H

- **(2)立即寻址的加法指令执行周期的微操作命令及节拍安排如下：**
 - **T0 Ad(IR)->Bus->R1 ;立即数送R1**
 - **T1 (ACC)+(R1)->ALU->R2**
 - **T2 R2->Bus->ACC**

ADD X 指令的执行周期

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

- **例10.2:** 设**CPU**中各部件及其相互连接关系如图**10.2**所示, 且**PC**有自动加**1**功能。此外还有**B、C、D、E、H、L**等**6**个寄存器, 它们各自的输入端和输出端都与内部总线**Bus**相连, 并分别受控制信号控制。要求写出完成下列指令组合逻辑控制单元所发出的微操作命令及节拍安排:

- **(1) ADD B, C ;(B)+(C)->B**

- **(2) SUB E,@H ;(E)-((H))->E**

寄存器间接寻址

- **(3) STA @mem ;ACC->((mem))**

存储器间接寻址

- **解:**

还有：B、C、D、E、H、L等6个寄存器

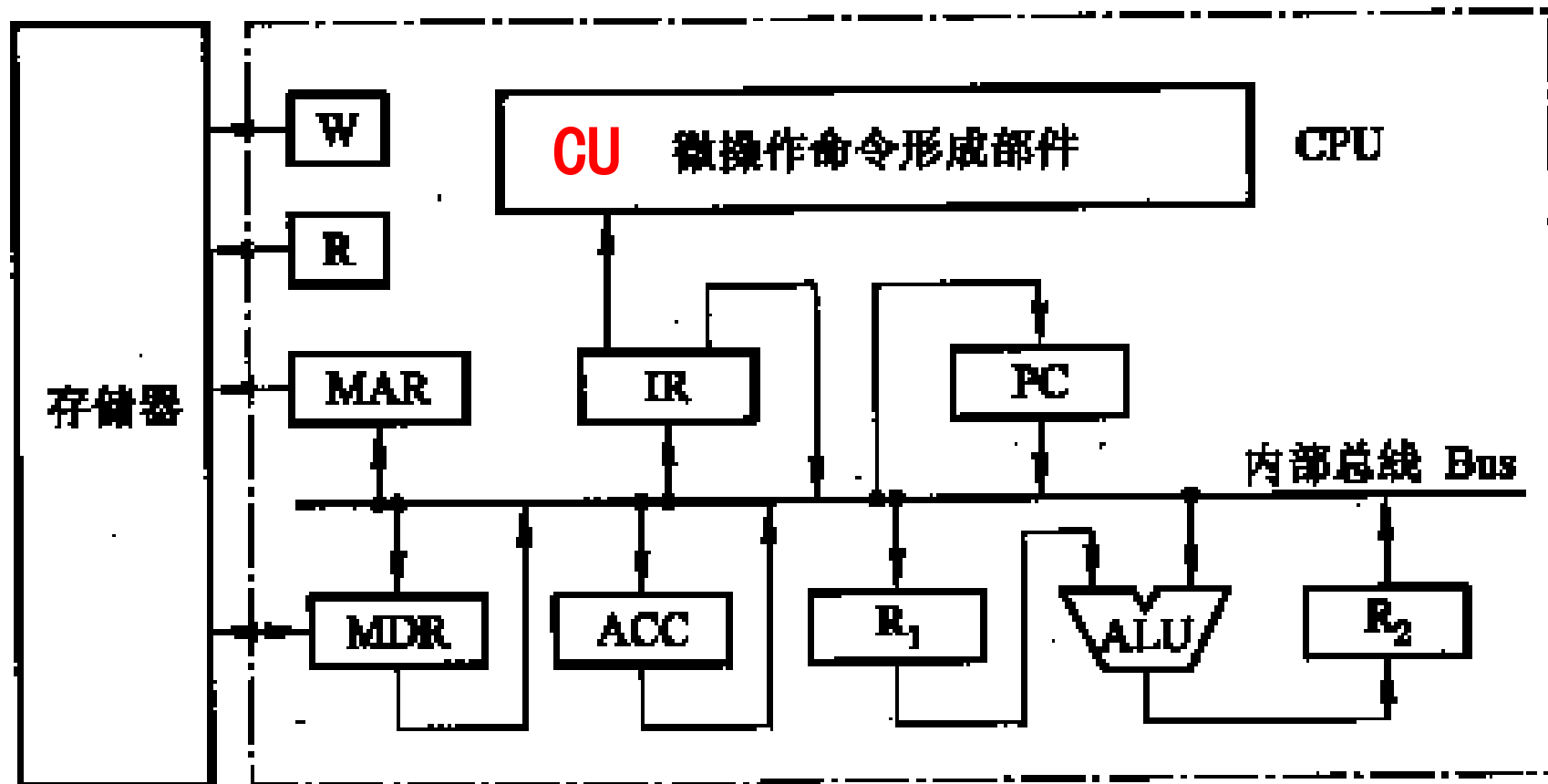


图 10.2 例 10.1 CPU 内部结构框图

ADD B,C

;2个寄存器相加，结果送B

- (1)完成“**ADD B,C**”指令所需的微操作命令及节拍安排如下：

- 取指周期

- **T0 PC→Bus→MAR, 1→R**

- **T1 M(MAR)→MDR, (PC)+1→PC**

- **T3 MDR→Bus→IR, OP(IR)→微操作命令形成部件(CU)**

- 执行周期

- **T0 C→Bus→R1**

- **T1 (B)+(R1)→ALU→R2**

- **T3 R2→Bus→B**

取指周期

T_0	PC → MAR 1 → R
T_1	M (MAR) → MDR (PC) + 1 → PC
T_2	MDR → IR OP (IR) → ID(指令译码)

ADD #a 指令的执行周期

T0 Ad(IR)→Bus→R1 ;立即数送R1

T1 (ACC)+(R1)→ALU→R2

T2 R2→Bus→ACC

- (2)完成“**SUB E,@H**”指令所需的微操作命令及节拍安排如下:

- 取指周期

- T0 PC->Bus->MAR, 1->R

- T1 M(MAR)->MDR, (PC)+1->PC

- T3 MDR->Bus->IR, OP(IR)->微操作命令形成部件(CU)

1个寄存器(E)的内容减去1个存储单元(以H为地址的存储单元)的内容, 结果送E寄存器

- 间址周期

- T0 H->Bus->MAR, 1->R

寄存器间接寻址

- T1 M(MAR)->MDR

访问存储器

得到另一个操作数

- 执行周期

- T0 MDR->Bus->R1

- T1 (E)-(R1)->ALU->R2

- T2 R2->Bus->E

一个机器周期内有3个节拍(时钟周期, T_0, T_1, T_2)或2个节拍(时钟周期, T_0, T_1)

- (3)完成“**STA @mem**”指令所需的微操作命令及节拍安排如下：

取指周期

- T0 PC->Bus->MAR, 1->R
- T1 M(MAR)->MDR, (PC)+1->PC
- T3 MDR->Bus->IR, OP(IR)->微操作命令形成部件

例如：MOV ((2000H)), ACC

间址周期

- T0 Ad(IR)->Bus->MAR, 1->R
- T1 M(MAR)->MDR

2000H

存储器间接寻址

访问存储器

得到有效地址=3000H

3000H

执行周期

- T0 MDR->Bus->MAR, 1->W
- T1 ACC->Bus->MDR
- T2 MDR->M(MAR)

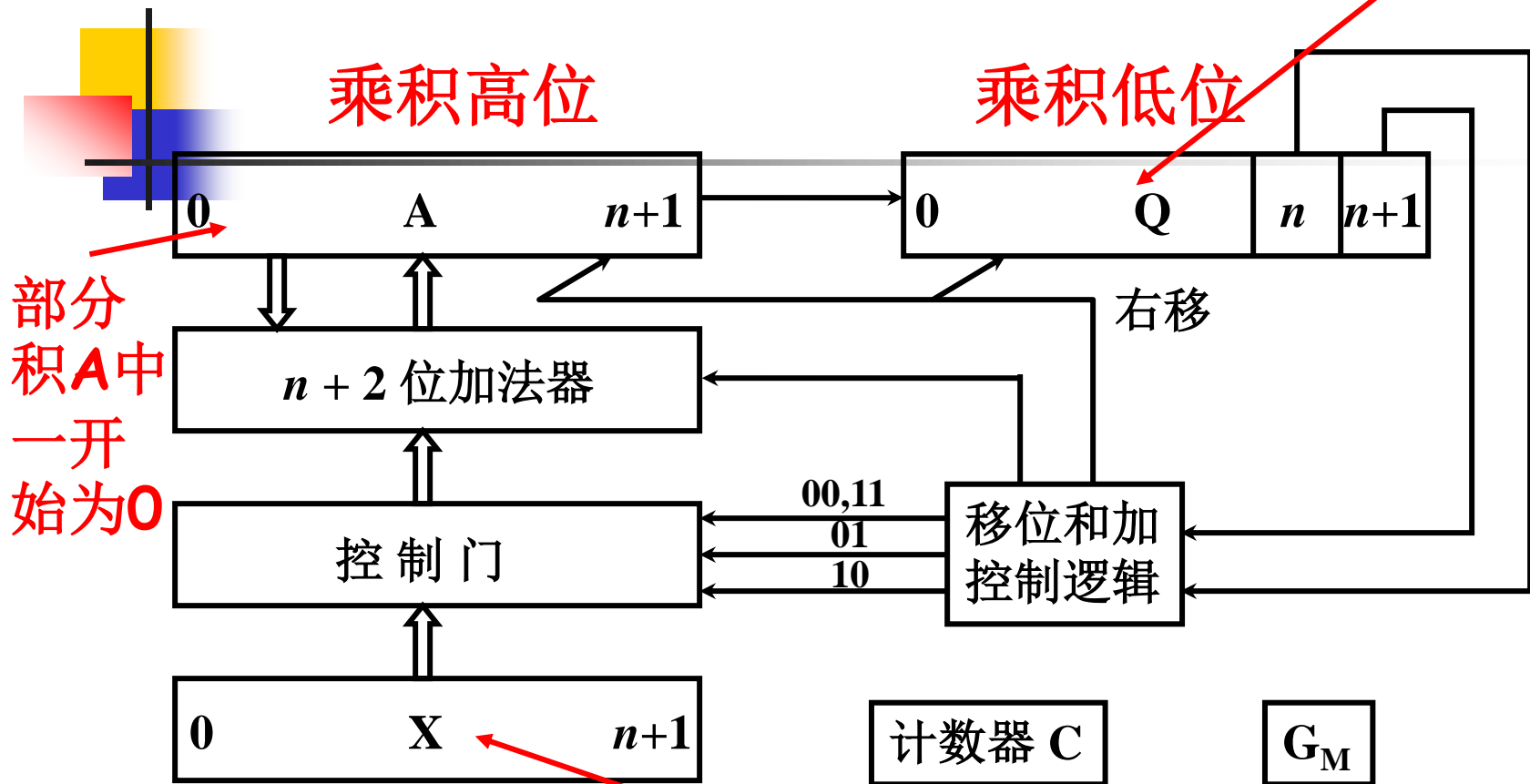
一个机器周期内有3个节拍
(时钟周期, T_0 , T_1 , T_2) 或2个
节拍 (时钟周期, T_0 , T_1)

- **例10.3**：设寄存器均为**16**位，实现补码**Booth**算法（补码一位乘法）的运算器如图**6.9**所示。其中寄存器**A, X**最高**2**位**A0、A1**和**X0、X1**为符号位，寄存器**Q**最高位**Q0**为符号位，最末位**Q15**为附加位。假设上条指令的运行结果存于**A**（即被乘数）中。
- **(1)**若**CU**为组合逻辑控制，且采用中央和局部控制相结合的方法，写出完成**MUL a(a为主存地址)**指令的全部微操作命令及节拍安排

MUL A, (2000H) ; a=2000 被乘数=A 乘数=(2000H)
- **(2)**指出哪些节拍属于**中央控制节拍**，哪些节拍属于**局部控制节拍**，局部控制最多需要几拍？
- **解：**

Booth 算法的硬件配置

图6.9



A、X、Q 均 $n+2$ 位

移位和加受末两位乘数控制

被乘数的补码

(1)

MUL a

■ 取指阶段

- T0 PC->Bus->MAR, 1->R
- T1 M(MAR)->MDR, (PC)+1->PC
- T2 MDR->Bus->IR, OP(IR)->ID

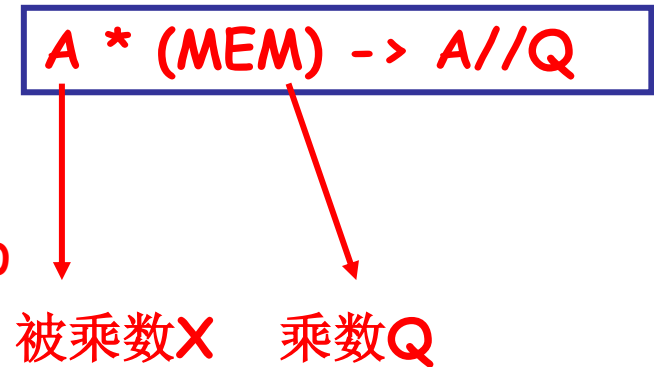
例如: MUL A, (2000H)

■ 执行周期

- T0 Ad(IR)->MAR, 1->R, A->X
- T1 M(MAR)->MDR, 0->Q₁₅, 0->A
- T2 MDR->Q₀₋₁₄
- T0*
$$\frac{1}{Q_{14}} * \frac{1}{Q_{15}} * (A+X) + \frac{1}{Q_{14}} * \frac{1}{Q_{15}} * (A+X+1) + \frac{1}{Q_{14}} * \frac{1}{Q_{15}} * A + \frac{1}{Q_{14}} * \frac{1}{Q_{15}} * A \rightarrow A$$
- T1* L(A//Q)->R(A//Q) (A//Q算术右移一位)
- .
- .
- .

被乘数送X

附加位置0 部分积置0



0 → A

表 6.17 例 6.22 求 $[x \cdot y]_{\text{补}}$ 的过程0 → Q₁₅

部分积	乘数 y_n	附加位 y_{n+1}	说 明
00.0000 $+ 00.1011$	10011	0	$y_n y_{n+1} = 10$, 部分积加 $[-x]_{\text{补}}$
00.1011 00.0101 00.0010 $+ 11.0101$	11001 11100	1 1	$\rightarrow 1$ 位, 得 $[z_1]_{\text{补}}$ $y_n y_{n+1} = 11$, 部分积 $\rightarrow 1$ 位, 得 $[z_2]_{\text{补}}$ $y_n y_{n+1} = 01$, 部分积加 $[x]_{\text{补}}$
11.0111 11.1011 11.1101 $+ 00.1011$	11 11110 11111	0 0	$\rightarrow 1$ 位, 得 $[z_3]_{\text{补}}$ $y_n y_{n+1} = 00$, 部分 $\rightarrow 1$ 位, 得 $[z_4]_{\text{补}}$ $y_n y_{n+1} = 10$, 部分积加 $[-x]_{\text{补}}$
00.1000	1111		最后一步不移位, 得 $[x \cdot y]_{\text{补}}$

MDR → Q₀₋₁₄

例 6.22 已知 $[x]_{\text{补}} = 1.0101$, $[y]_{\text{补}} = 1.0011$, 求 $[x \cdot y]_{\text{补}}$

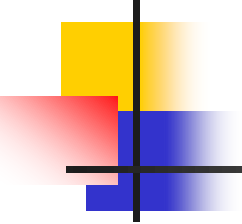
表 6.15 $y_i y_{i+1}$ 的状态对操作的影响

$y_i y_{i+1}$	$y_{i+1} - y_i$	操 作
0 0	0	部分积右移一位
0 1	1	部分积加 $[x]_{\text{补}}$, 再右移一位
1 0	-1	部分积加 $[-x]_{\text{补}}$, 再右移一位
1 1	0	部分积右移一位

$$/Q14*Q15*(A+X)+Q14*/Q15*(A+/X+1)+/Q14*/Q15*A+Q14*Q15*A \rightarrow A$$

$$L(A//Q) \rightarrow R(A//Q)$$

($A//Q$ 算术右移一位)

- 
- **(2)中央控制节拍**包括取指阶段所有节拍和执行阶段的**T0、T1、T2**等**3**个节拍，完成取指令和取操作数及乘法运算前的准备工作。
 - 局部控制节拍是执行阶段的**T0***和**T1***节拍，其中**T0***为重复加操作，受Q寄存器末2位**Q₁₄Q₁₅**控制，最多执行**15次**；**T1***为移位操作，共执行**14次**。

最后一步不移位

三、组合逻辑设计步骤

6条指令

1. 列出每个微操作命令的操作时间表

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	$1 \rightarrow IND$	有间指					
		\bar{I}	$1 \rightarrow EX$	无间指					

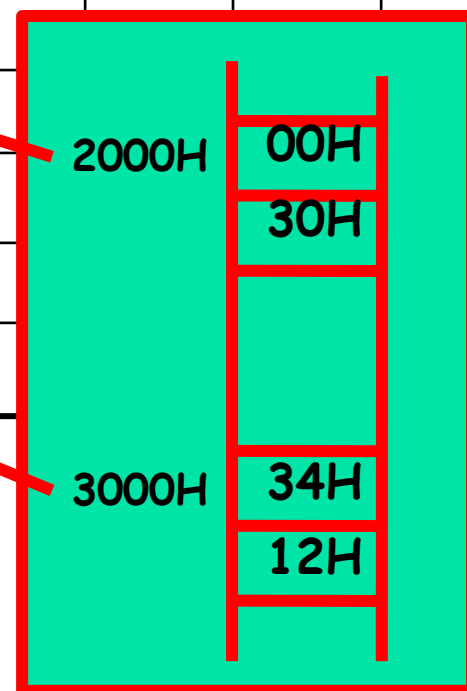
间址特征 I=1有间址

三、组合逻辑设计步骤

1. 列出每个微操作命令的操作时间表 **6条指令**

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) → MAR						
			1 → R						
	T_1		M(MAR) → MDR						
	T_2		MDR → Ad (IR)						
		$\overline{\text{IND}}$	1 → EX						

间址周期标志 IND=0表示一次间址寻址，转EX执行周期；IND=1表示多次间址寻址，继续间接寻址



三、组合逻辑设计步骤

6条指令

1. 列出每个微操作命令的操作时间表

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) → MAR						
			1 → R						
			1 → W						
	T ₁		M(MAR) → MDR						
			AC → MDR						
	T ₂		(AC)+(MDR) → AC						
			MDR → M(MAR)						
			MDR → AC						
			0 → AC						

AC → AC
Ad(IR)->PC

三、组合逻辑设计步骤

6条指令，其中4条为间接寻址

1. 列出每个微操作命令的操作时间表

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

6条指令

1. 列出每个微操作命令的操作时间表

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
		$\overline{\text{IND}}$	$1 \rightarrow \text{EX}$			1	1	1	1

这4条间址指令都是一次间接寻址

三、组合逻辑设计步骤

6条指令

1. 列出每个微操作命令的操作时间表

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					
			AC \rightarrow AC Ad(IR) \rightarrow PC		1				1

表 10.1 操作时间表

10条指令

工作周期 期标记	节拍	状态条件	微操作命令信号	CLA	COM	SHR	CSL	STP	ADD	STA	LDA	JMP	BAN
FE (取指)	T_0		PC \rightarrow MAR	1	1	1	1	1	1	1	1	1	1
			$1 \rightarrow R$	1	1	1	1	1	1	1	1	1	1
	T_1		$M(MAR) \rightarrow MDR$	1	1	1	1	1	1	1	1	1	1
			$(PC) + 1 \rightarrow PC$	1	1	1	1	1	1	1	1	1	1
	T_2		MDR \rightarrow IR	1	1	1	1	1	1	1	1	1	1
			$OP(IR) \rightarrow ID$	1	1	1	1	1	1	1	1	1	1
		\overline{I}	$1 \rightarrow IND$						1	1	1	1	1
		$\overline{1}$	$1 \rightarrow EX$	1	1	1	1	1	1	1	1	1	1
IND (间接寻址)	T_0		$Ad(IR) \rightarrow MAR$						1	1	1	1	1
			$1 \rightarrow R$						1	1	1	1	1
	T_1		$M(MAR) \rightarrow MDR$						1	1	1	1	1
	T_2	\overline{IND}	MDR $\rightarrow Ad(IR)$						1	1	1	1	1
			$1 \rightarrow EX$						1	1	1	1	1
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									
			$\overline{AC} \rightarrow AC$		1								
			$1(AC) \rightarrow R(AC), AC_0$ 不变			1							
			$P^{-1}(AC)$				1						
			$Ad(IR) \rightarrow PC$									1	
		A_0	$Ad(IR) \rightarrow PC$										1
			$0 \rightarrow G$					1					

2. 写出每个微操作命令的最简表达式

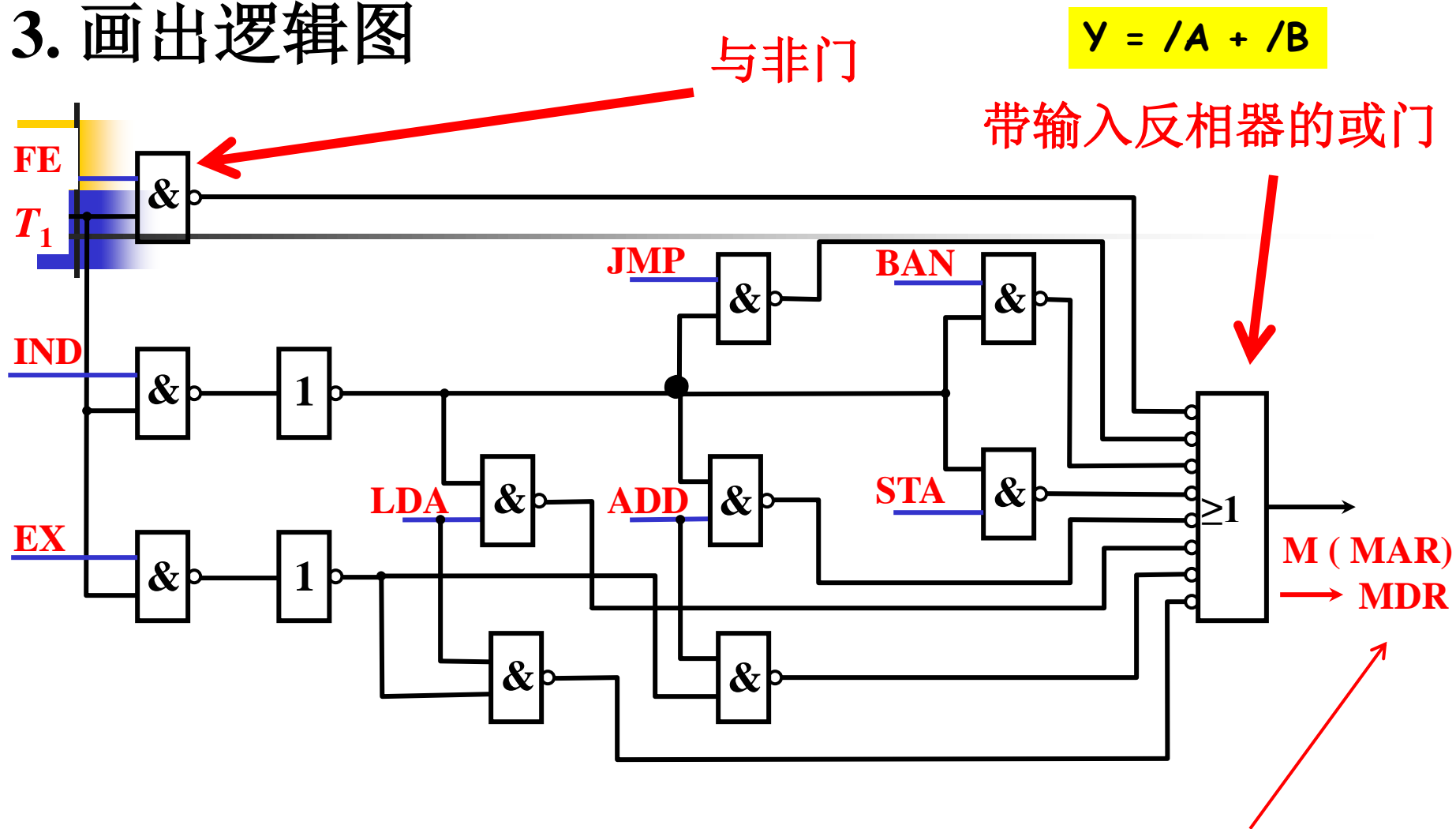


以 $M(MAR) \rightarrow MDR$ 微操作命令 为例

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

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

3. 画出逻辑图



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

微操作控制信号

组合逻辑设计

特点

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



组合逻辑设计/硬布线控制器

硬布线控制器是早期设计计算机的一种方法。硬布线控制器是将控制部件做成产生专门固定时序控制信号的逻辑电路，产生各种控制信号，因而又称为**组合逻辑控制器**。

这种逻辑电路以使用最少元件和取得最高操作速度为设计目标，因为该逻辑电路由门电路和触发器构成的复杂树型网络，所以称为硬布线控制器。

10.2 微程序设计



一、微程序设计思想的产生

二、微程序控制单元框图及工作原理

三、微指令的编码方式（控制方式）

四、微指令序列地址的形成

五、微指令格式

六、静态微程序设计和动态微程序设计

七、毫微程序设计

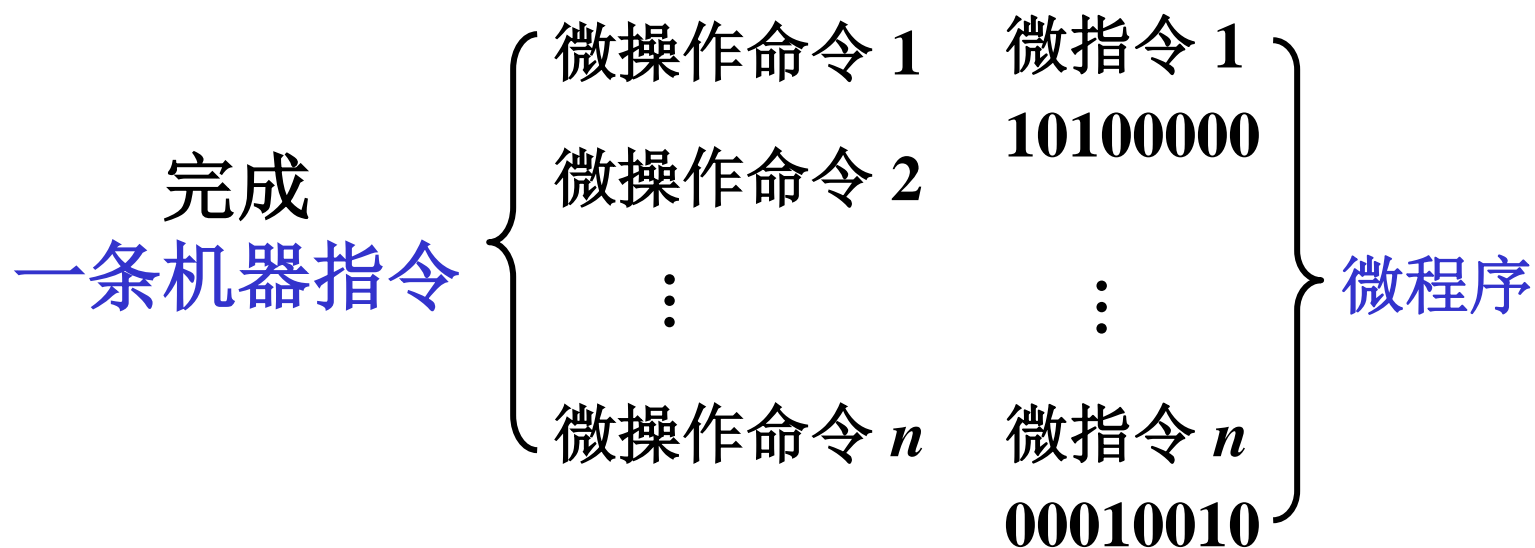
八、串行微程序控制和并行微程序控制

九、微程序设计举例

一、微程序设计思想的产生



1951 英国剑桥大学教授 M.V. Wilkes 提出



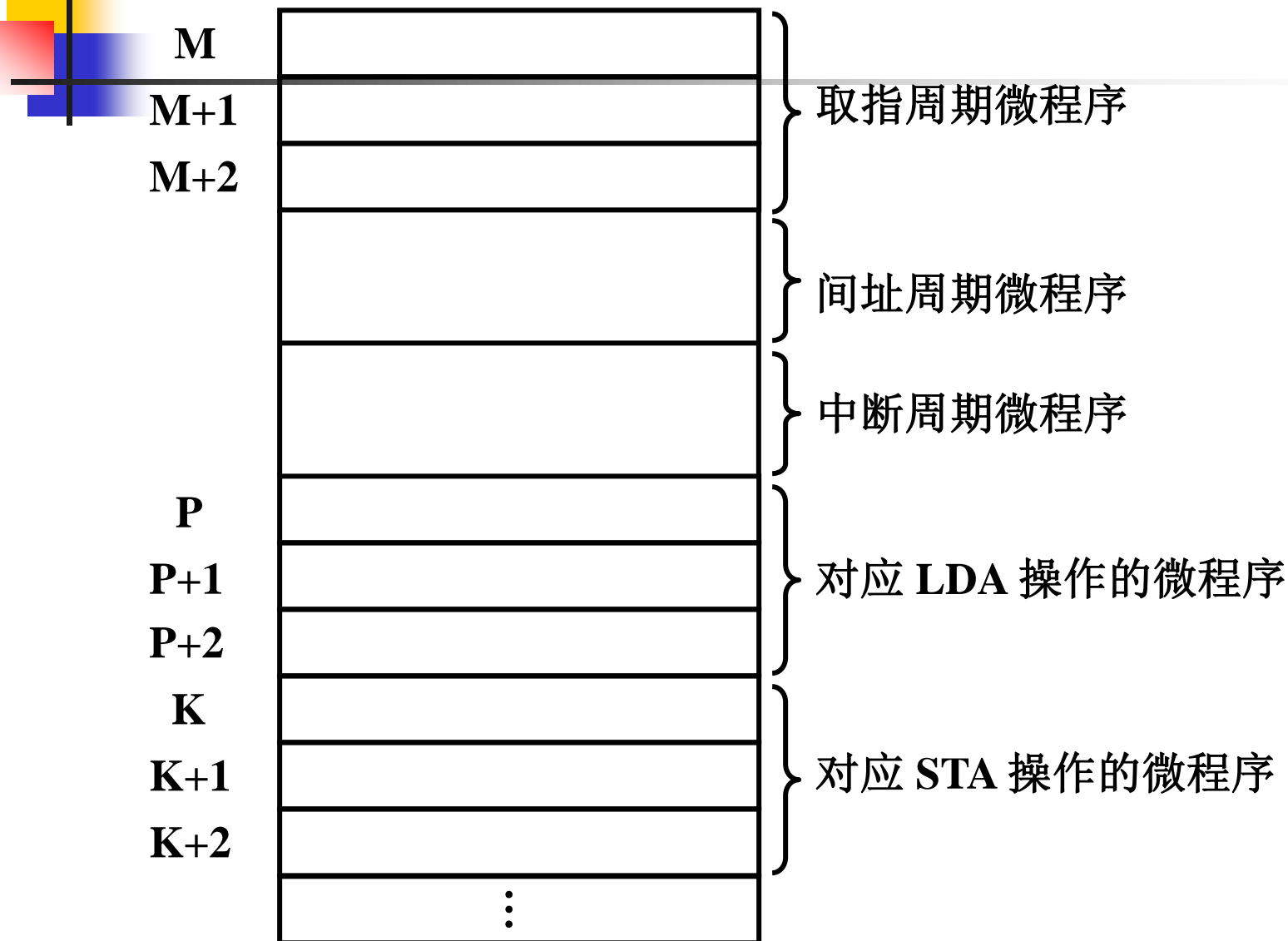
一条机器指令
对应一段微程序(若干条微指令)

存入 ROM 控存

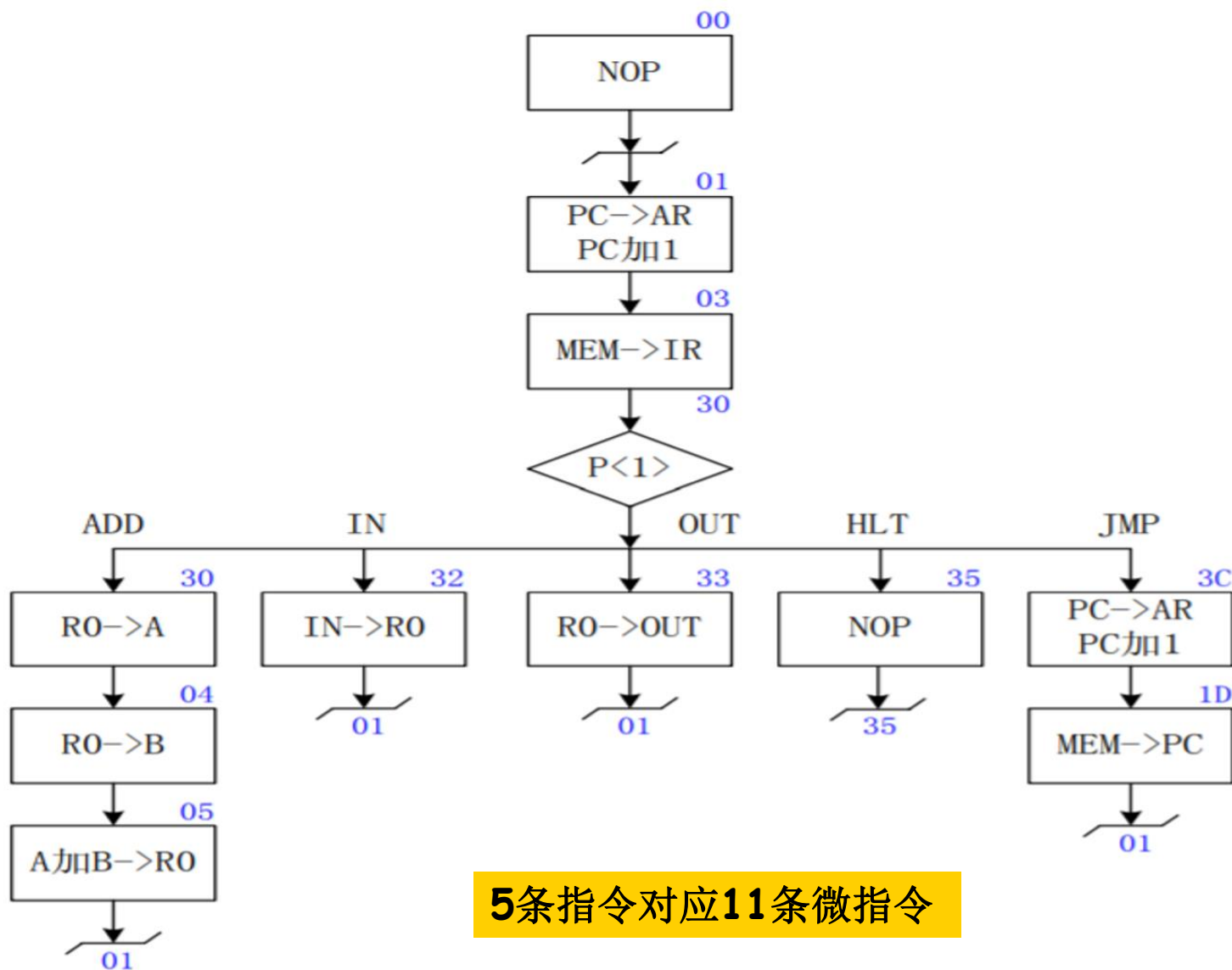
存储逻辑 控制存储器

二、微程序控制单元框图及工作原理

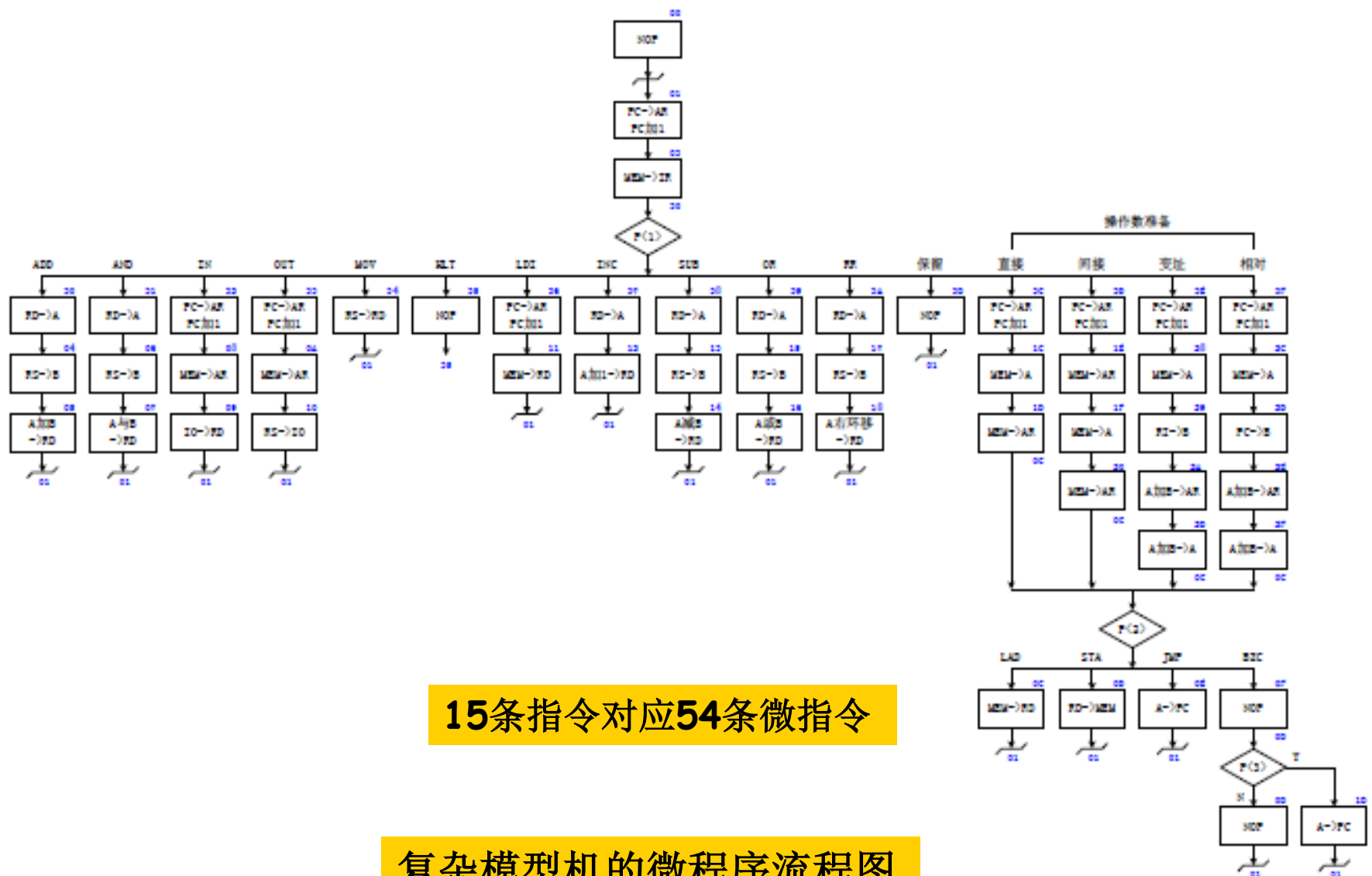
1. 机器指令对应的微程序



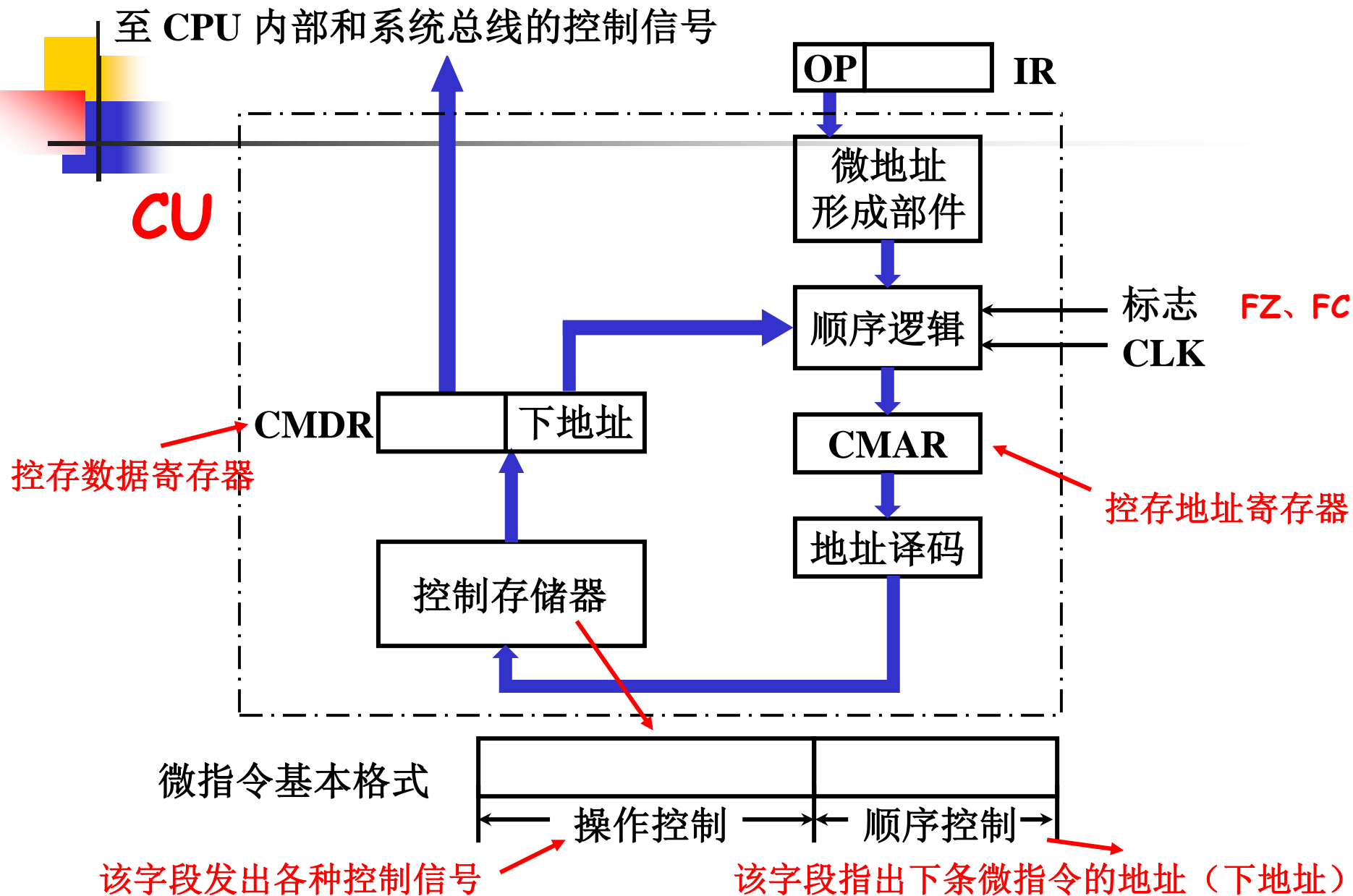
执行周期微程序



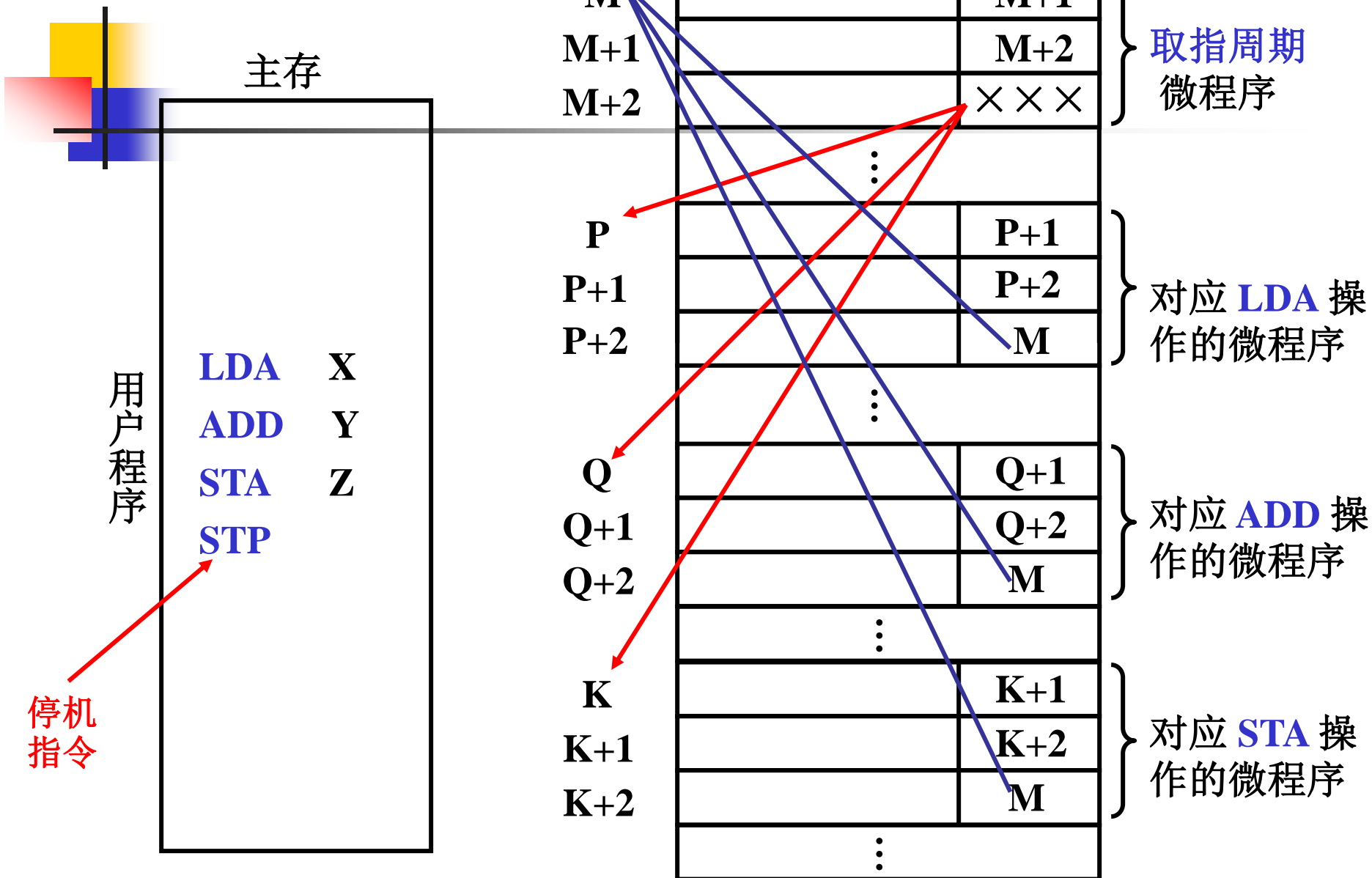
简单模型机的微程序流程图



2. 微程序控制单元的基本框图



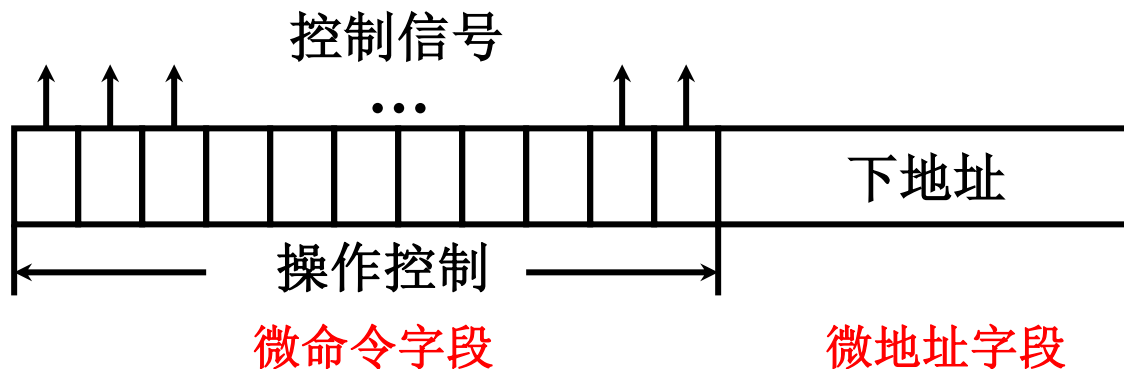
3. 工作原理



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



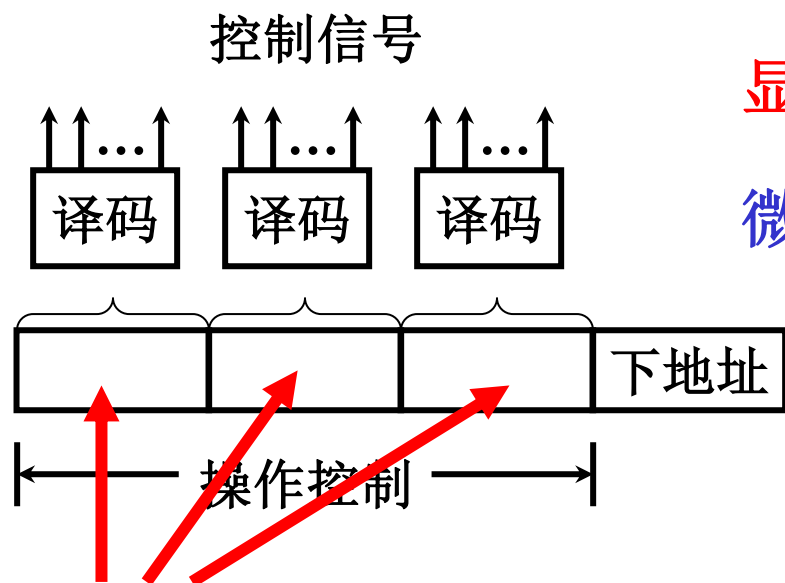
速度最快

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

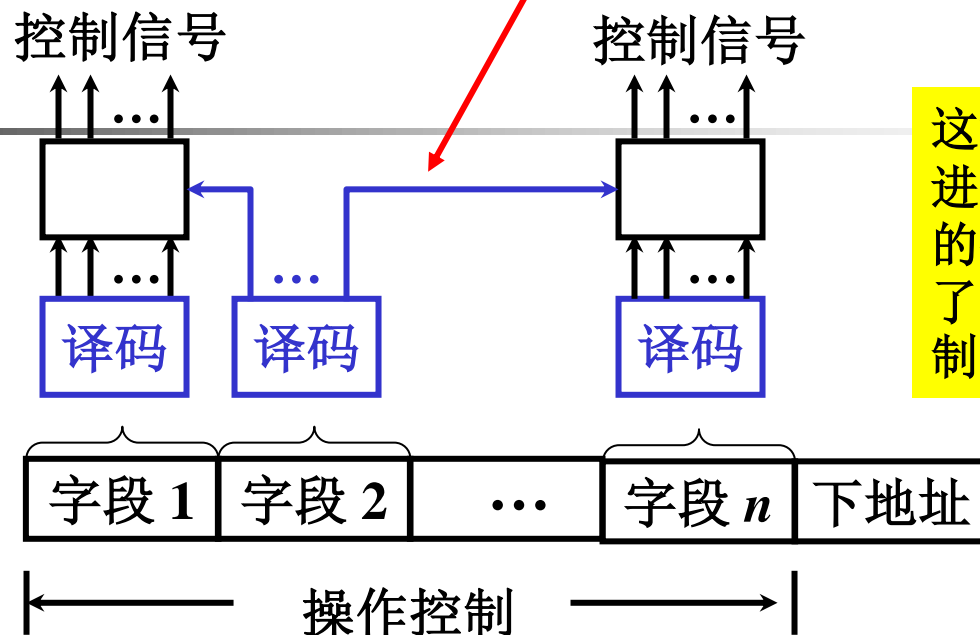
缩短了微指令字长，增加了译码时间

4位 → 2位 8位 → 3位

3. 字段间接编码方式

一个字段的某些微命令还需由另一个字段中的某些微命令来解释

隐式编码



这种方法虽然可以进一步缩短微指令的字长，但确削弱了微指令的并行控制能力

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

如设置常数字段，用来提供常数、计数器初值等

- **例10.4**：微指令格式中有**8**个控制字段，每个字段可分别激活**5、8、3、16、1、7、25、4**种控制信号。分别采用**直接编码**和**字段直接编码**，请给出两种方式的控制字段各取几位？

- **解**：
- **(1) 直接编码（直接控制）**： $5+8+3+16+1+7+25+4=69$ 位
- **(2) 字段直接编码**：每个控制字段至少需要留一个码字表示不激活任何一条控制线，即**8**个控制字段需要：**6**（ $=5+1$ ）、**9**（ $=8+1$ ）、**4**（ $=3+1$ ）、**17**（ $=16+1$ ）、**2**（ $=1+1$ ）、**8**（ $=7+1$ ）、**26**（ $=25+1$ ）、**5**（ $=4+1$ ）种状态；
- 故需要： $3(2^3 \geq 6) + 4(2^4 \geq 9) + 2(2^2 \geq 4) + 5(2^5 \geq 17) + 1(2^1 \geq 2) + 3(2^3 \geq 8) + 5(2^5 \geq 26) + 3(2^3 \geq 5) = 26$ 位

直接编码（直接控制）：**69**位

字段直接编码：**26**位

四、微指令序列地址的形成

1. 直接由微指令的 **下地址字段** 指出

2. 根据机器指令的 **操作码** 形成

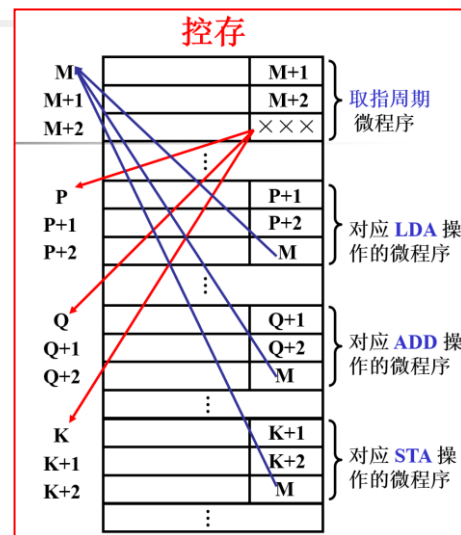
3. **增量计数器**

由取指周期转到执行周期时

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

4. **分支转移**

条件转移指令



操作控制字段

转移方式

转移地址

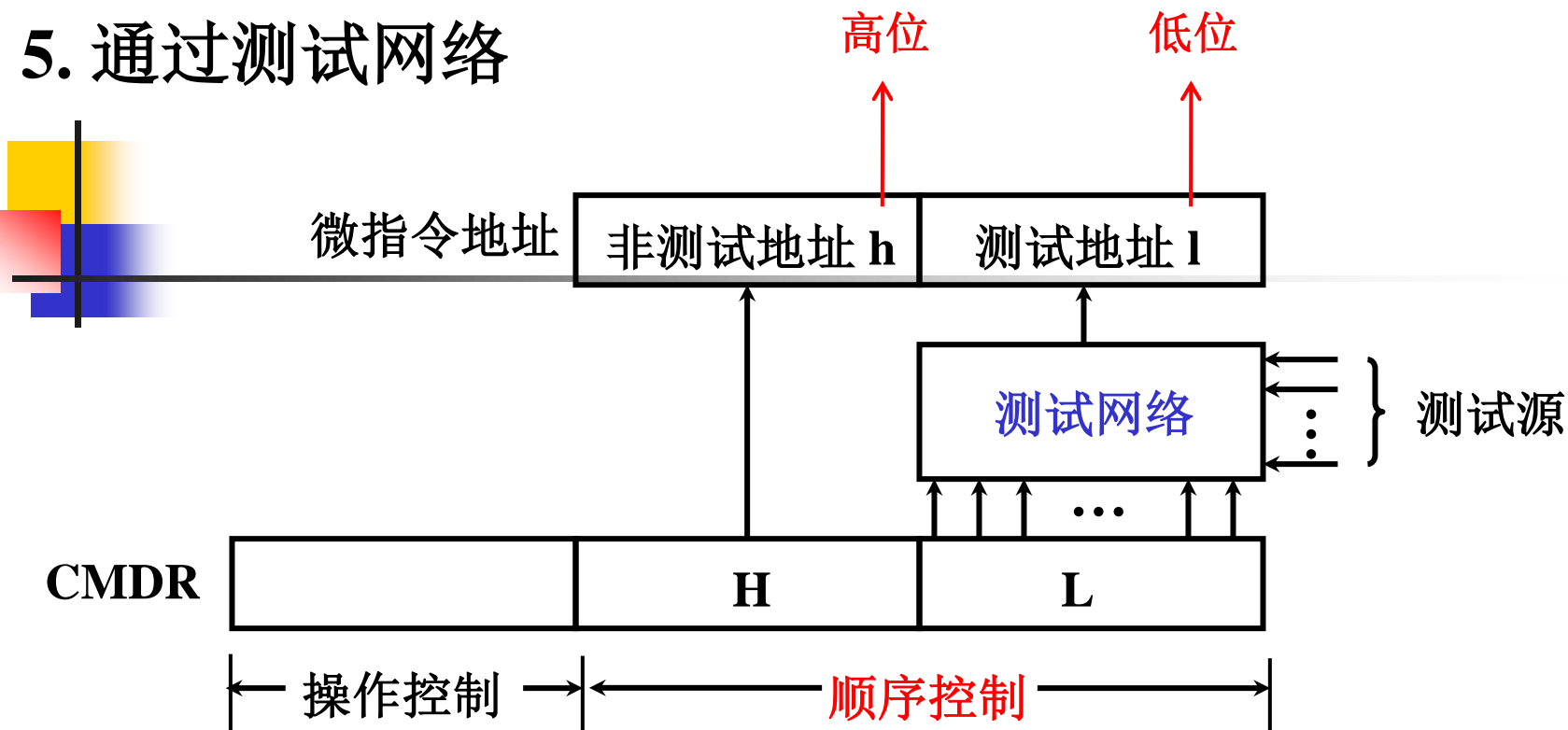
转移方式

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

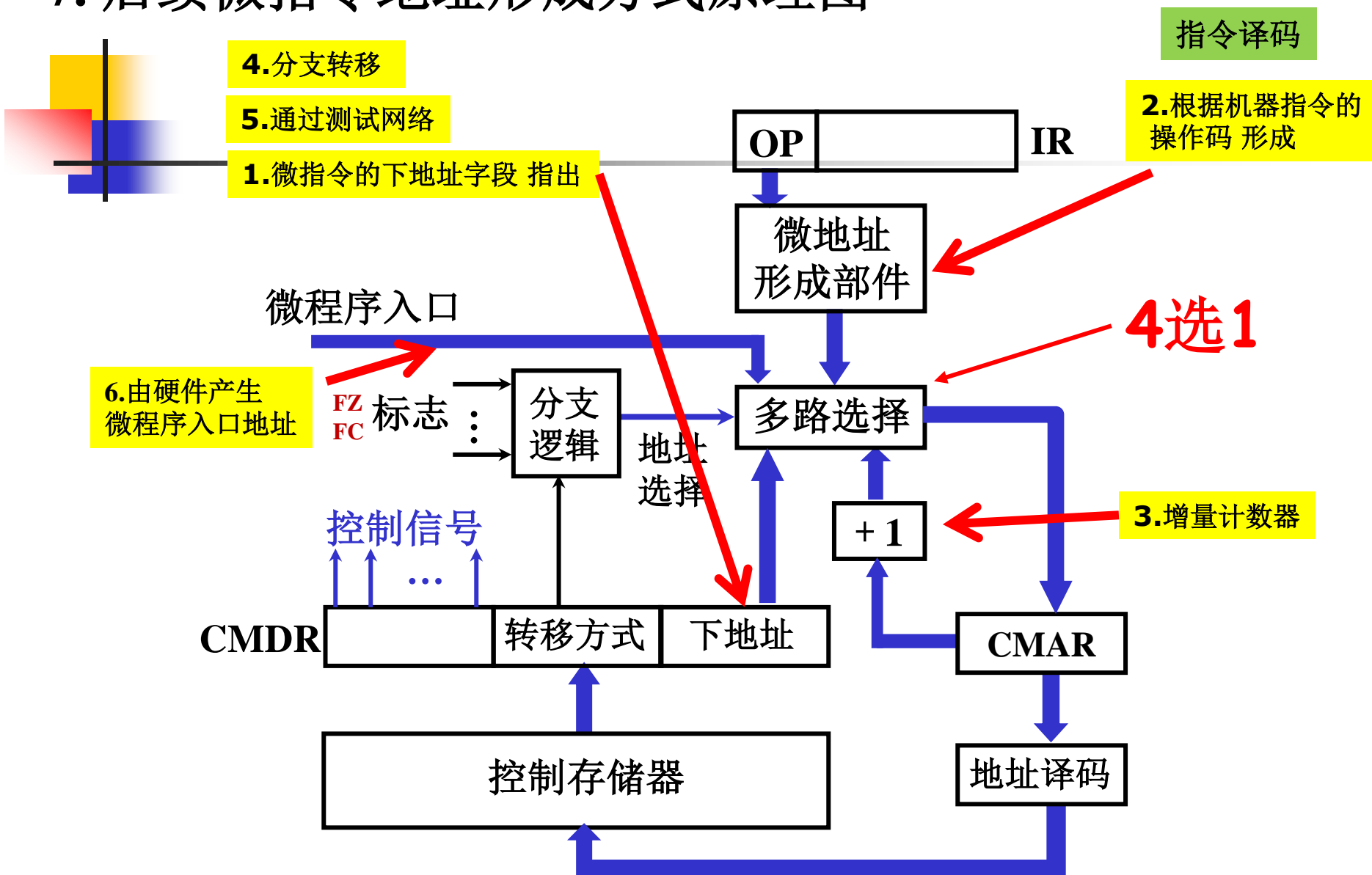


6. 由硬件产生微程序入口地址

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



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（直接控制）、字段直接编码、字段间接编码、直接编码和字段编码（字段直接编码、字段间接编码）的混合编码

2. 垂直型微指令

微操作码

地址码

其他

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位

5位

5位

3位

表 10.2 垂直型微指令示例

微操作码	地址码		其他	微指令类别及功能
0 1 2	3 ~ 7	8 ~ 12	13 15	
0 0 0	源寄存器	目的寄存器	其他控制	传送型微指令
0 0 1	ALU 左输入	ALU 右输入	ALU	运算控制型微指令 按 ALU 字段所规定的功能执行,其结果送暂存器
0 1 0	寄存器	移位次数	移位方式	移位控制型微指令 按移位方式对寄存器中的数据移位
0 1 1	寄存器	存储器	读写	访存微指令 完成存储器和寄存器之间的传送
1 0 0	D		S	无条件转移微指令 D 为微指令的目的地址
1 0 1	D		测试条件	条件转移微指令 最低 4 位为测试条件
1 1 0				可定义 I/O 或其他操作
1 1 1				第 3 ~ 15 位可根据需要定义各种微命令

垂直型微指令

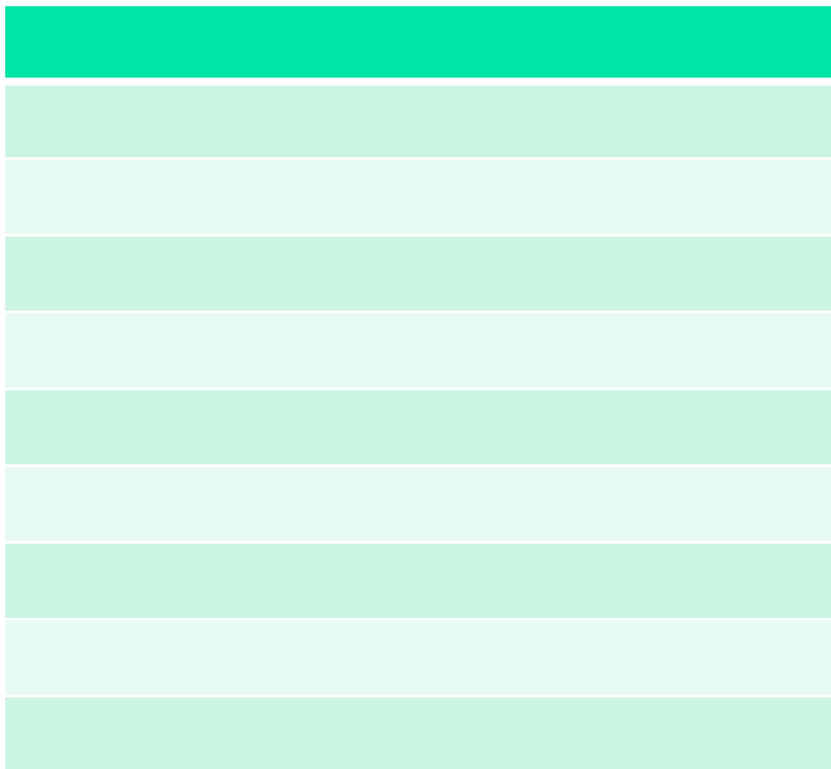
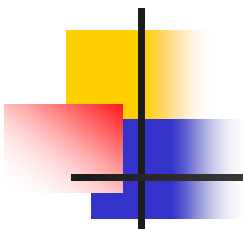
3. 两种微指令格式的比较

(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

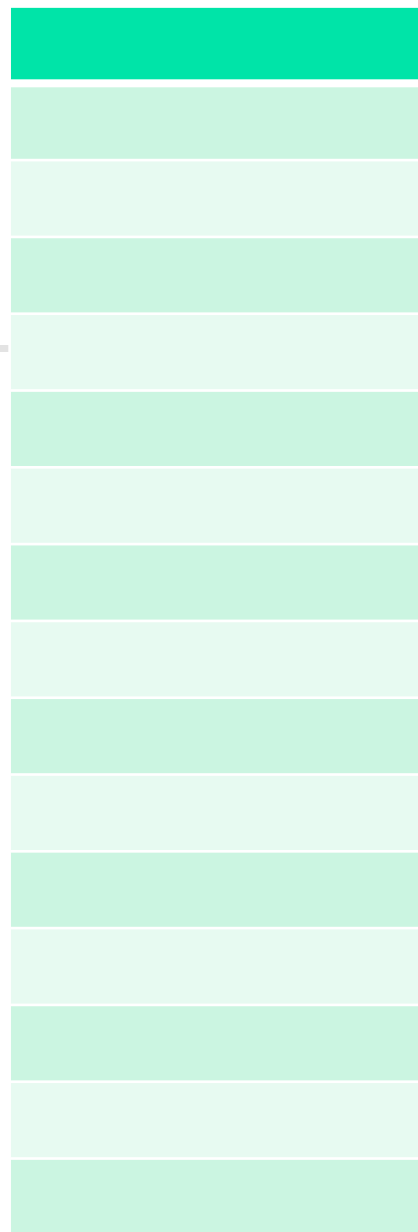
(2) 水平型微指令执行一条机器指令所要的
微指令数目少，速度快

(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；垂直型微指令正好相反

(4) 水平型微指令与机器指令差别大；垂直型微指令与机器指令相似



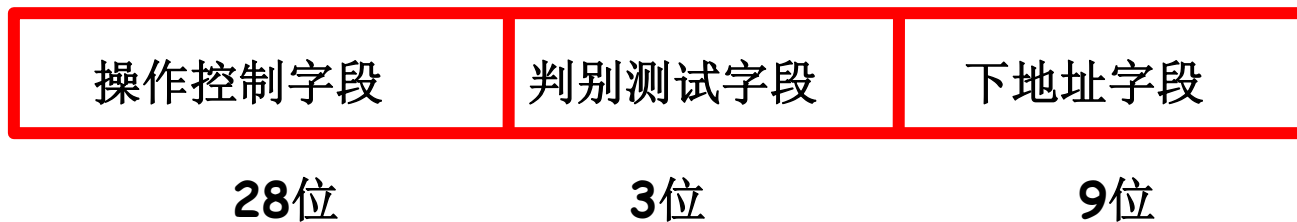
水平型微指令



垂直型微指令

- **例10.5:** 微程序控制器采用水平型直接控制（直接编码）的微指令格式，后续微指令地址由微指令的下地址字段给出。已知机器共有**28**个微命令、**6**个互斥的可判定的外部条件，控制存储器的容量为**512 X 40**位。试设计其微指令格式，并说明理由。

- **解:**
- 水平型微指令由：操作控制字段、判别测试字段、下地址字段三部分构成。
- 操作控制字段：**28位**对应**28**个微命令
- 判别测试字段：**3位**（ $2^3 \geq 6$ ）可形成**6**个互斥的可判定的外部条件
- 下地址字段：由控制存储器的容量决定，**9位**， $2^9 = 512$



- **例10.6:** 某机器共有**52**个微操作控制信号，构成**5**个互斥的微命令组，各组分别包括**5、8、2、15、22**个微命令（ **$52=5+8+2+15+22$** ）。已知可判定的外部条件有**2**个，微命令字长为**28**位。
 - **（1）**按照水平型微指令格式设计微指令，要求微指令的下地址字段直接给出后续微指令的地址。
 - **（2）**指出控制存储器的容量。



- **解：**
- **(1)** 考虑到每一组必须增加一种不发命令的情况，**5**个互斥的微命令组分别包括**5+1**、**8+1**、**2+1**、**15+1**、**22+1**个微命令，**5**个控制字段（采用**字段直接编码方式**）分别为：**3**、**4**、**2**、**4**、**5**位（共**18**位）
- 条件测试字段应包括一种不转移的情况（共**3**种情况），条件测试字段取**2**位
- 下地址字段=**28-18-2=8**位
- **(2)** 控制存储器的容量为：**256X28**位 **$2^8=256$**



六、静态微程序设计和动态微程序设计



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、Flash ROM

七、毫微程序设计

1. 毫微程序设计的基本概念

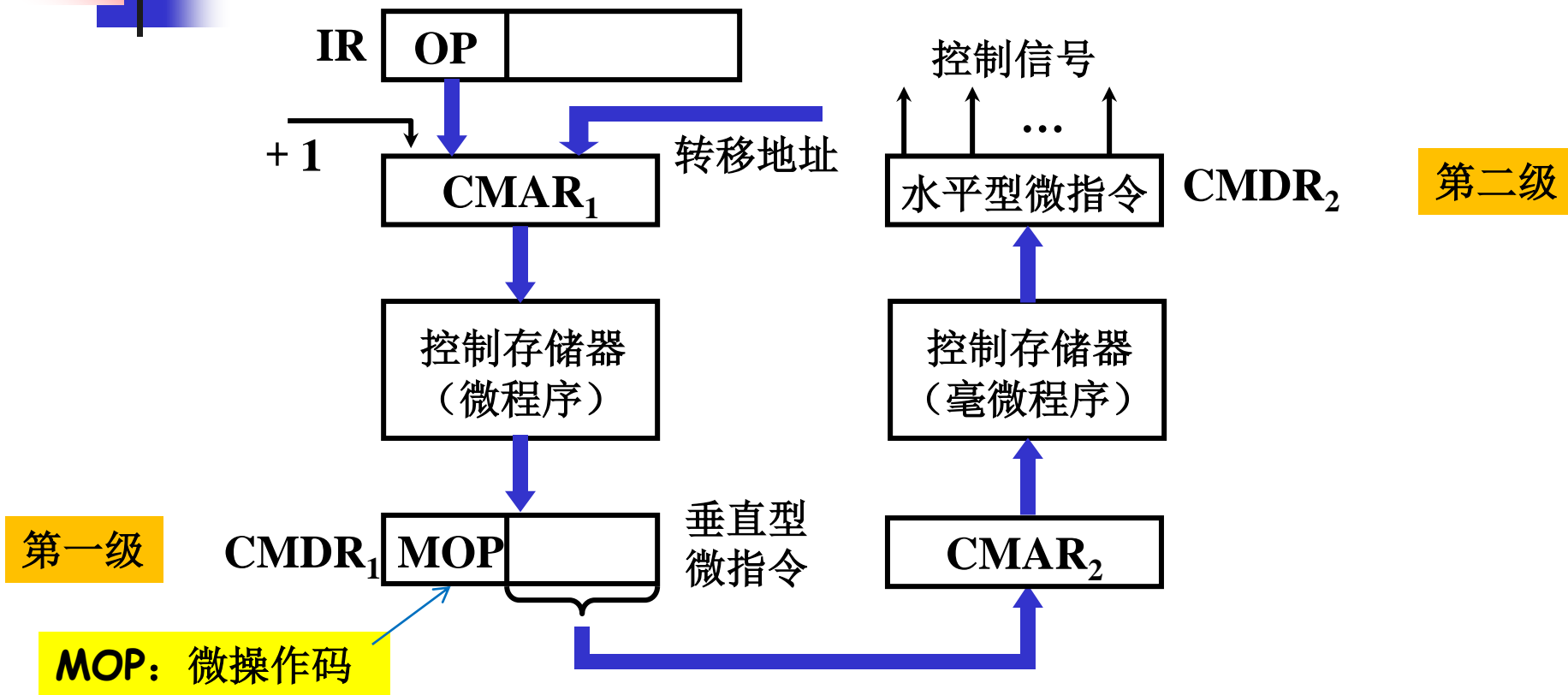
微程序设计 用 微程序解释机器指令

毫微程序设计 用 毫微程序解释微指令(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

两级微程序设计方法：第一级为垂直型微指令，第二级为水平型微指令



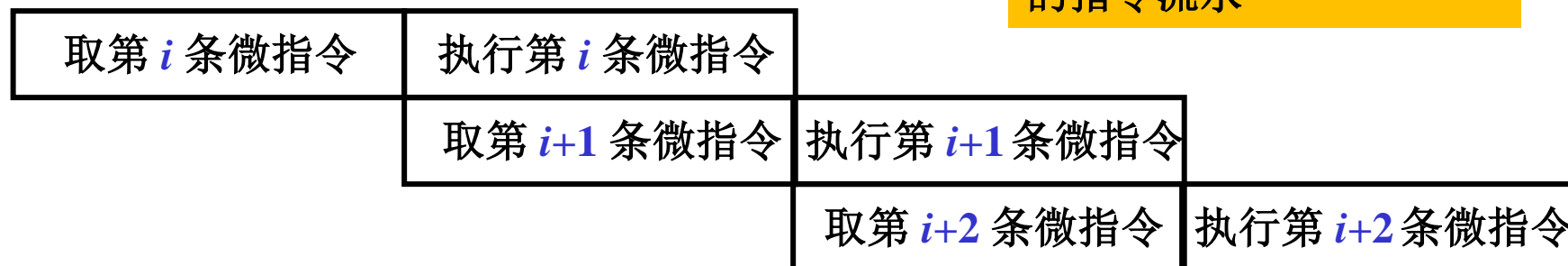
八、串行微程序控制和并行微程序控制

串行微程序控制



并行微程序控制

相当于第8章（8.3小节）
的指令流水



九、微程序设计举例

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 (6 个微操作)

T_0	$PC \rightarrow MAR$	$1 \rightarrow R$
T_1	$M(MAR) \rightarrow MDR$	$(PC) + 1 \rightarrow PC$
T_2	$MDR \rightarrow IR$	$OP(IR) \rightarrow \text{微地址形成部件}$

组合逻辑设计

(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

考虑到需形成后续微指令的地址

- 非访存指令

- ① CLA 指令

$T_0 \quad 0 \longrightarrow AC$

$T_1 \quad Ad (CMDR) \longrightarrow CMAR$

- ② COM 指令

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

$T_1 \quad Ad (CMDR) \longrightarrow CMAR$

③ SHR 指令

算术右移

T_0 $L(AC) \longrightarrow R(AC)$ $AC_0 \longrightarrow AC_0$

T_1 $Ad(CMDR) \longrightarrow CMAR$

④ CSL 指令

循环左移

T_0 $R(AC) \longrightarrow L(AC)$ $AC_0 \longrightarrow AC_n$

T_1 $Ad(CMDR) \longrightarrow CMAR$

⑤ STP 指令

停机

T_0 $0 \longrightarrow G$

T_1 $Ad(CMDR) \longrightarrow CMAR$

• 访存指令

⑥ ADD 指令

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

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

⑦ STA 指令

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

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

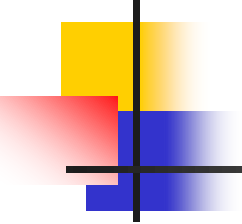
T_2 $\text{AC} \longrightarrow \text{MDR}$

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

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

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

⑧ LDA 指令



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

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

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

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（直接编码方式）

(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

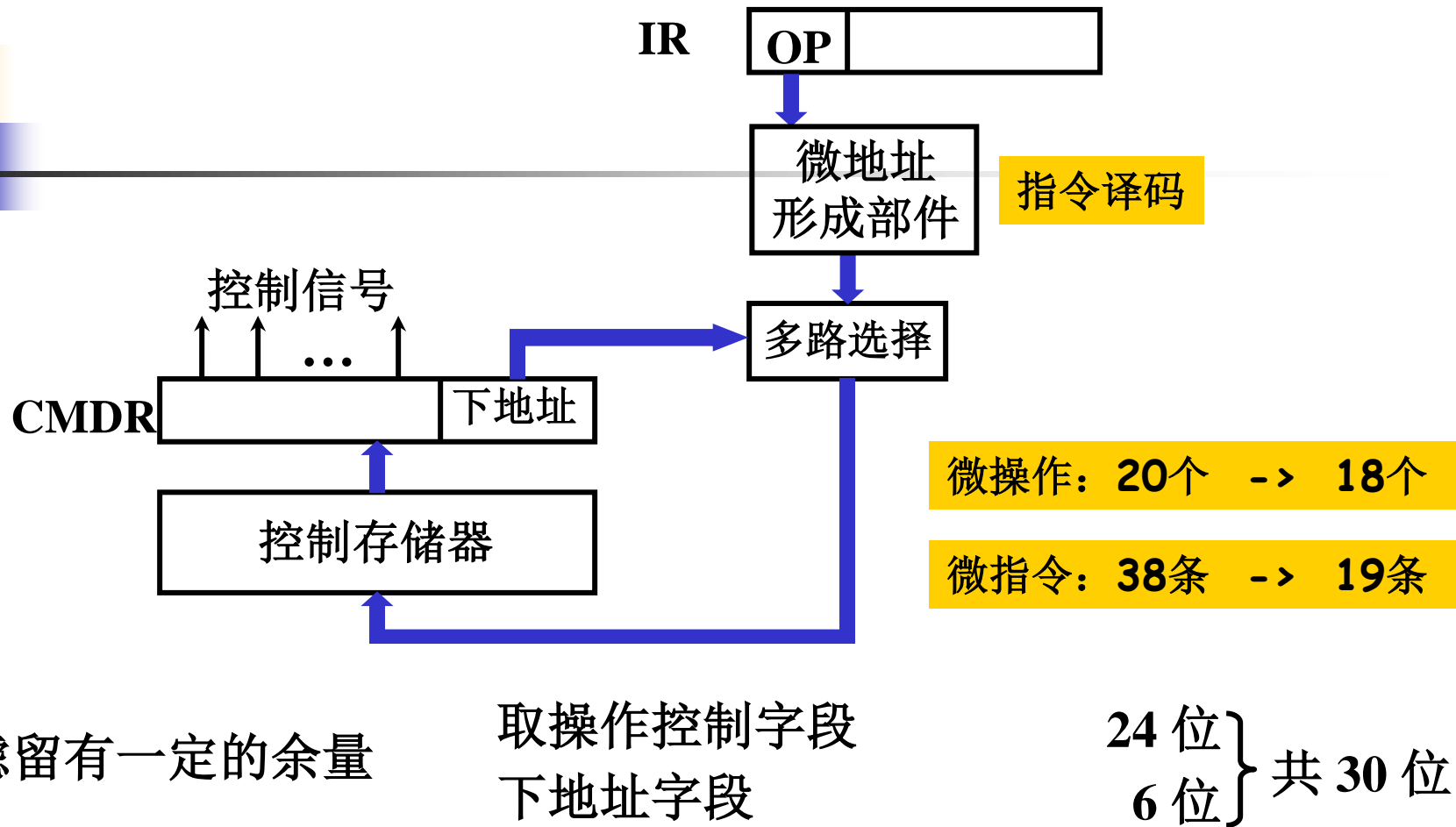
由 38 条微指令

确定微指令的 下地址字段 为 6 位 $2^6 \geq 38$ $2^5 < 38$

微指令字长 可取 $20 + 6 = 26$ 位

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

(4) 省去了 CMAR 的控制存储器



(5) 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

实验中的“微指令的代码化”

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					
		0	1	2	3	4	...	10	...	23	24	25	26	27	28	29
取指																
	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

表 10.3 对应 10 条机器指令的微指令码点

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																						顺序控制字段							
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
取指	00	1	1																												1
	01			1	1																									1	
	02					1																				x	x	x	x	x	x
PC → MAR		1 → R																													
M (MAR) → MDR		(PC) + 1 → PC																													
MDR → IR		OP (IR) → 微地址形成部件																													
ADD	10		1									1															1			1	
	11			1																							1		1		
	12												1																		
STA	13											1		1														1	1		
	14														1													1	1		1
	15																1														
LDA	16		1									1																1	1	1	1
	17			1																							1				
	20																	1													
JMP	21																		1												
BAN	22																			1											

5个微操作

3条微指令

5个微操作

3条微指令

表 10.3 对应 10 条机器指令的微指令码点

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																						顺序控制字段							
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
取指	0 0	1	1																												1
	0 1			1	1																									1	
	0 2					1																				×	×	×	×	×	×
CLA	0 3						1																								
COM	0 4							1																							

① CLA 指令

$T_0 \quad 0 \longrightarrow AC$

~~$T_1 \quad Ad(CMDR) \longrightarrow CMAR$~~

② COM 指令

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

~~$T_1 \quad Ad(CMDR) \longrightarrow CMAR$~~

2个微操作

2条微指令

LDA	1 7			1																											
	2 0																1														
JMP	2 1																	1													
BAN	2 2																		1												

2个微操作

2条微指令

表 10.3 对应 10 条机器指令的微指令码点

[illegible]

③ SHR 指令

算术右移

$$T_0 \quad \mathbf{L}(\mathbf{AC}) \longrightarrow \mathbf{R}(\mathbf{AC}) \quad \mathbf{AC}_0 \longrightarrow \mathbf{AC}_0$$
 ~~$T_1 \quad \text{Ad (CMDR)} \rightarrow \text{CMAR}$~~

1个微操作

1条微指令

[illegible]

表 10.3 对应 10 条机器指令的微指令码点

[illegible]

④ CSL 指令

循环左移

$$T_0 \quad \mathbf{R}(\mathbf{AC}) \longrightarrow \mathbf{L}(\mathbf{AC}) \quad \mathbf{AC}_0 \longrightarrow \mathbf{AC}_n$$
 ~~$T_1 \rightarrow Ad(CMDR) \rightarrow CMAR$~~

1个微操作

1条微指令

[illegible]

表 10.3 对应 10 条机器指令的微指令码点

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																						顺序控制字段								
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	
取指	0 0	1	1																												1	
	0 1			1	1																									1		
	0 2					1																				×	×	×	×	×	×	
CLA	0 3						1																									
COM	0 4							1																								
SHR	0 5								1																							
CSL	0 6									1																						
STP	0 7										1																					
<div>⑤ STP 指令</div> <div>停机</div> <div>$T_0 \quad 0 \rightarrow G$</div> <div>$T_1 \quad Ad(CMDR) \rightarrow CMAR$</div>																																
LDA	1 7			1																												
	2 0																1															
JMP	2 1																	1														
BAN	2 2																		1													

1个微操作

1条微指令

1个微操作

1条微指令

表 10.3 对应 10 条机器指令的微指令码点

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																							顺序控制字段						
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
取指	0 0	1	1																												1
	0 1			1	1																									1	
	0 2					1																				×	×	×	×	×	×
CLA	0 3						1																								
COM	0 4							1																							
SHR	0 5								1																						
CSL	0 6									1																					
STP	0 7										1																				
ADD	1 0		1									1																1			1
	1 1			1																								1		1	
	1 2												1																		

• 访存指令

⑥ ADD 指令

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

T_1 Ad(CMDR) \rightarrow CMAR

T_2 M(MAR) \rightarrow MDR

T_3 Ad(CMDR) \rightarrow CMAR

T_4 (AC) + (MDR) \rightarrow AC

T_5 Ad(CMDR) \rightarrow CMAR

2+2个微操作

3条微指令


表 10.3 对应 10 条机器指令的微指令码点

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																						顺序控制字段								
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	
<div>⑦ STA 指令</div> <div>T_0 Ad (IR) \rightarrow MAR 1 \rightarrow W</div> <div>T_1 Ad (CMDR) \rightarrow CMAR</div> <div>T_2 AC \rightarrow MDR</div> <div>T_3 Ad (CMDR) \rightarrow CMAR</div> <div>T_4 MDR \rightarrow M (MAR)</div> <div>T_5 Ad (CMDR) \rightarrow CMAR</div>																																1
																																1
STA	1 3											1		1															1	1		
	1 4														1													1	1		1	
	1 5																1															
LDA	1 6		1									1																	1	1	1	
	1 7			1																							1					
	2 0																	1														
JMP	2 1																		1													
BAN	2 2																			1												

1+3个微操作

3条微指令

表 10.3 对应 10 条机器指令的微指令码点

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																							顺序控制字段							
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	
⑧ LDA 指令																															1	
																																
LDA	1 6		1									1																				
	1 7			1																							1					
	2 0																	1														
JMP	2 1																		1													
BAN	2 2																			1												

3+1 个微操作

3 条微指令

表 10.3 对应 10 条机器指令的微指令码点

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																								顺序控制字段					
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29

⑨ JMP 指令

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

~~T_1 Ad (CMDR) \rightarrow CMAR~~

⑩ BAN 指令

$$T_0 \quad A_0 \bullet \text{Ad}(\text{IR}) + \bar{A}_0 \bullet (\text{PC}) \longrightarrow \text{PC}$$
 ~~$T: Ad(CMDR) \rightarrow CMAR$~~ [illegible]

11条微指令

简单模型机11条微指令的代码表(码点)

地址	十六进制	高五位	S3-S0	A 字段	B 字段	C 字段	MA5-MA0
00	00 00 01	00000	0000	000	000	000	000001
01	00 6D 43	00000	0000	110	110	101	000011
03	10 70 70	00010	0000	111	000	001	110000
04	00 24 05	00000	0000	010	010	000	000101
05	04 B2 01	00000	1001	011	001	000	000001
1D	10 51 41	00010	0000	101	000	101	000001
30	00 14 04	00000	0000	001	010	000	000100
32	18 30 01	00011	0000	011	000	000	000001
33	28 04 01	00101	0000	000	010	000	000001
35	00 00 35	00000	0000	000	000	000	110101
3C	00 6D 5D	00000	0000	110	110	101	011101

复杂模型机54条微指令的代码表（码点）

序

地址	十六进制表示	高五位	S3-S0	A 字段	B 字段	C 字段	UA5-UA0
00	00 00 01	00000	0000	000	000	000	000001
01	00 6D 43	00000	0000	110	110	101	000011
03	10 70 70	00010	0000	111	000	001	110000
04	00 24 05	00000	0000	010	011	000	000101
05	04 B2 01	00000	1001	011	001	000	000001
06	00 24 07	00000	0000	010	011	000	000111
07	01 32 01	00000	0010	011	001	000	000001
08	10 60 09	00010	0000	110	000	000	001001
09	18 30 01	00011	0000	011	000	000	000001
0A	10 60 10	00010	0000	110	000	000	010000
0B	00 00 01	00000	0000	000	000	000	000001
0C	10 30 01	00010	0000	011	000	000	000001
0D	20 06 01	00100	0000	000	001	100	000001
0E	00 53 41	00000	0000	101	001	101	000001
0F	00 00 CB	00000	0000	000	000	011	001011
10	28 04 01	00101	0000	000	010	000	000001

地址	十六进制表示	高五位	S3-S0	A 字段	B 字段	C 字段	UA5-UA0
11	10 30 01	00010	0000	011	000	000	000001
12	06 B2 01	00000	1101	011	001	000	000001
13	00 24 14	00000	0000	010	011	000	010100
14	05 B2 01	00000	1011	011	001	000	000001
15	00 24 16	00000	0000	010	011	000	010110
16	01 B2 01	00000	0011	011	001	000	000001
17	00 24 18	00000	0000	010	011	000	011000
18	02 B2 01	00000	0101	011	001	000	000001
1B	00 53 41	00000	0000	101	001	101	000001
1C	10 10 1D	00010	0000	001	000	000	011101
1D	10 60 8C	00010	0000	110	000	010	001100
1E	10 60 1F	00010	0000	110	000	000	011111
1F	10 10 20	00010	0000	001	000	000	100000
20	10 60 8C	00010	0000	110	000	010	001100
28	10 10 29	00010	0000	001	000	000	101001
29	00 28 2A	00000	0000	010	100	000	101010
2A	04 E2 2B	00000	1001	110	001	000	101011
2B	04 92 8C	00000	1001	001	001	010	001100
2C	10 10 2D	00010	0000	001	000	000	101101

地址	十六进制表示	高五位	S3-S0	A 字段	B 字段	C 字段	UA5-UA0
2D	00 2C 2E	00000	0000	010	110	000	101110
2E	04 E2 2F	00000	1001	110	001	000	101111
2F	04 92 8C	00000	1001	001	001	010	001100
30	00 16 04	00000	0000	001	011	000	000100
31	00 16 06	00000	0000	001	011	000	000110
32	00 6D 48	00000	0000	110	110	101	001000
33	00 6D 4A	00000	0000	110	110	101	001010
34	00 34 01	00000	0000	011	010	000	000001
35	00 00 35	00000	0000	000	000	000	110101
36	00 6D 51	00000	0000	110	110	101	010001
37	00 16 12	00000	0000	001	011	000	010010
38	00 16 13	00000	0000	001	011	000	010011
39	00 16 15	00000	0000	001	011	000	010101
3A	00 16 17	00000	0000	001	011	000	010111
3B	00 00 01	00000	0000	000	000	000	000001
3C	00 6D 5C	00000	0000	110	110	101	011100
3D	00 6D 5E	00000	0000	110	110	101	011110
3E	00 6D 68	00000	0000	110	110	101	101000
3F	00 6D 6C	00000	0000	110	110	101	101100

- **例10.7：**某机有**5**条微指令，每条微指令发出的控制信号如表**10.4**所示。采用直接控制（直接编码）方式设计微指令的控制字段，要求其位数最少，而且保持微指令本身的并行性。

- **解：**由表**10.4**可知，控制信号**c、g、i**仅在微指令**I1**同时出现，可合并用**1**位控制字段表示；控制信号**b、h**仅在微指令**I2**同时出现，可合并用**1**位控制字段表示.这样**10**个控制信号可压缩到**7**个。

表10.4 例10.7 表格

微指令	激活的控制信号									
	a	b	c	d	e	f	g	h	i	j
I ₁	√		√		√		√		√	
I ₂	√	√		√		√		√		√
I ₃	√			√	√	√				
I ₄	√									
I ₅	√			√						√

a	b h	c g i	d	e	f	j
---	-----	-------	---	---	---	---



附录10A PC整机介绍

- 主板
- 芯片组

本章小结

■ 组合逻辑设计（硬布线控制器）

■ 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 节省时间

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

■ 微操作命令及节拍安排

取指周期

- T_0 $PC \rightarrow Bus \rightarrow MAR, 1 \rightarrow R$
- T_1 $M(MAR) \rightarrow MDR, (PC)+1 \rightarrow PC$
- T_3 $MDR \rightarrow Bus \rightarrow IR, OP(IR) \rightarrow$ 微操作命令形成部件(CU)

间址周期 微操作的节拍安排

T_0 $Ad(IR) \rightarrow MAR$

$1 \rightarrow R$

T_1 $M(MAR) \rightarrow MDR$

T_2 $MDR \rightarrow Ad(IR)$

执行周期

ADD AL, 80H

(2)立即寻址的加法指令执行周期的微操作命令及节拍安排如下:

- T_0 $Ad(IR) \rightarrow Bus \rightarrow R1$;立即数送R1
- T_1 $(ACC)+(R1) \rightarrow ALU \rightarrow R2$
- T_2 $R2 \rightarrow Bus \rightarrow ACC$

5. 中断周期 微操作的节拍安排

T_0 $0 \rightarrow MAR$ $1 \rightarrow W$ 硬件关中断

T_1 $PC \rightarrow MDR$

T_2 $MDR \rightarrow M(MAR)$ 向量地址 $\rightarrow PC$

本章小结

■ 组合逻辑设计（硬布线控制器）

■ 组合逻辑设计步骤

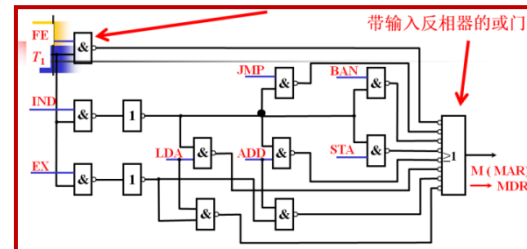
- 1. 列出每个微操作命令的操作时间表
- 2. 写出每个微操作命令的最简表达式
- 3. 画出逻辑图

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
		1	1 \rightarrow IND			1	1	1	1
		$\bar{1}$	1 \rightarrow EX	1	1	1	1	1	1

■ 以 $M(MAR) \rightarrow MDR$ 微操作命令 为例

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

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





本章小结

■ 微程序设计

- 一条机器指令对应一段微程序（若干条微指令），微程序存储在控制存储器（控存）中
- 微指令 = 操作控制字段 + 顺序控制字段（下地址）
- 微指令的编码方式（控制方式）：主要是前面**2**种方式
 1. 直接编码（直接控制）方式
 2. 字段直接编码方式：实验箱模型机采用的方式
 3. 字段间接编码方式
 4. 混合编码
 5. 其他



本章小结

■ 微程序设计

■ 微指令序列地址的形成：主要是前面**2**种方式

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
5. 通过测试网络
6. 由硬件产生微程序入口地址



本章小结

■ 微程序设计

- 水平型微指令
- 垂直型微指令

微操作码	地址码	其他
------	-----	----

- 静态微程序设计
- 动态微程序设计
- 毫微程序设计：采用两级微程序设计方法，第一级为垂直型微指令，第二级为水平型微指令
- 串行微程序控制
- 并行微程序控制



本章小结

- 微程序设计举例（微程序设计步骤）：
 - 写出机器指令对应的全部微操作及节拍安排（例**10.1**、**10.2**、**10.3**；例**9.1**、**9.2**）
 - 确定微指令格式：采用什么编码方式（直接控制？字段直接编码）？微命令字段几位？微地址字段几位？
 - 编写每条微指令的二进制代码（微指令码点，微指令代码化）



第16次作业——习题（P420-423）

- **10.7**
- **10.8**
- **10.9**
- **10.10**
- **10.11**
- **10.13**
- **10.15**
- **10.19**



关于作业的提交

- **1周内**必须提交（上传到学院的**FTP**服务器上），否则认为是迟交作业；如果期末仍然没有提交，则认为是未提交作业
 - 作业完成情况成绩=第**1**次作业提交情况*第**1**次作业评分+第**2**次作业提交情况*第**2**次作业评分+.....+第**N**次作业提交情况*第**N**次作业评分
 - 作业评分：**A**（好）、**B**（中）、**C**（差）三挡
 - 作业提交情况：按时提交（**1.0**）、迟交（**0.5**）、未提交（**0.0**）
- 请采用电子版的格式（**Word**文档）上传到**FTP**服务器上，文件名取“学号+姓名+第**X**次作业.doc”
 - 例如：**11920192203642+袁佳哲+第16次作业.doc**
- 第**16**次作业提交的截止日期为：**2021年6月18日晚上24点**



The End

Thanks