第10章　控制单元的设计

系统结构研究所· 计算机组成原理

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

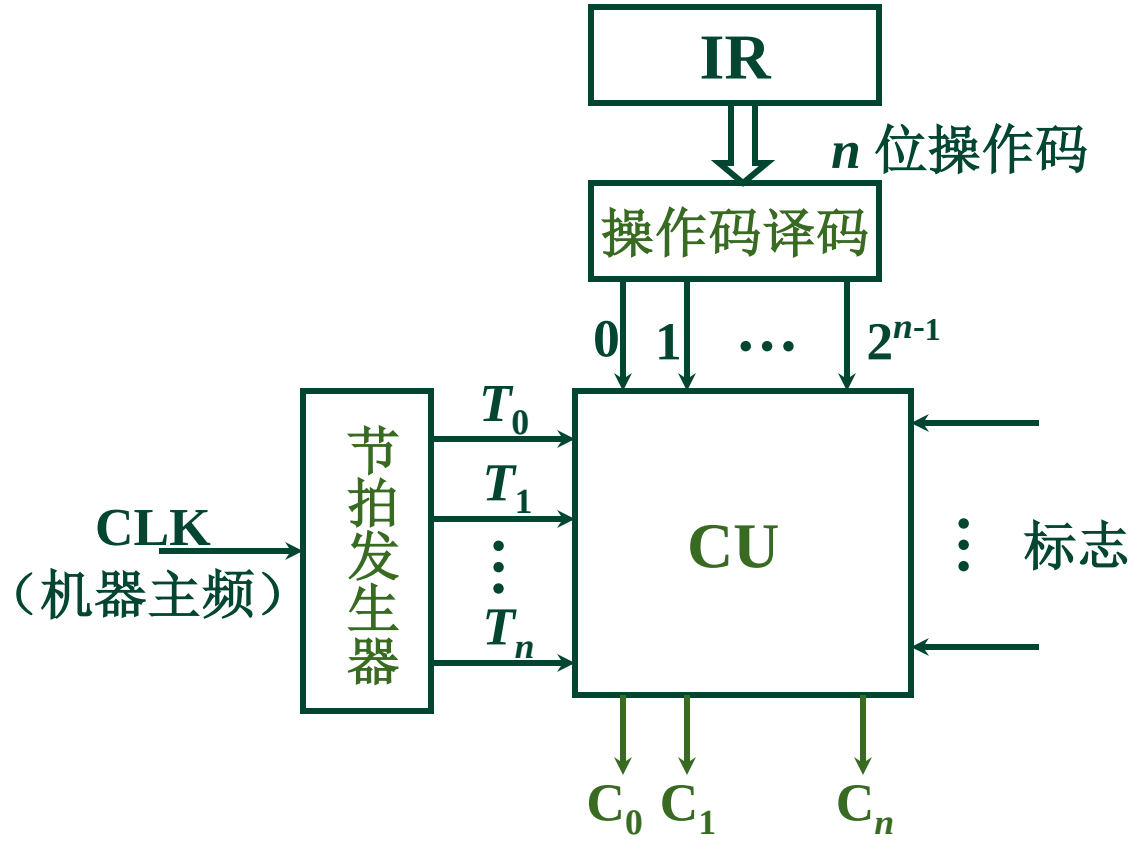
控制器的类型

- 组合逻辑型：核心是微操作产生部件，用组合逻辑设计思想，以布尔代数为工具设计。输入信号来自指令译码器的输出、时序发生器的时序信号和程序运行结果特征及状态，输出为带有时间标志的微操作控制信号。

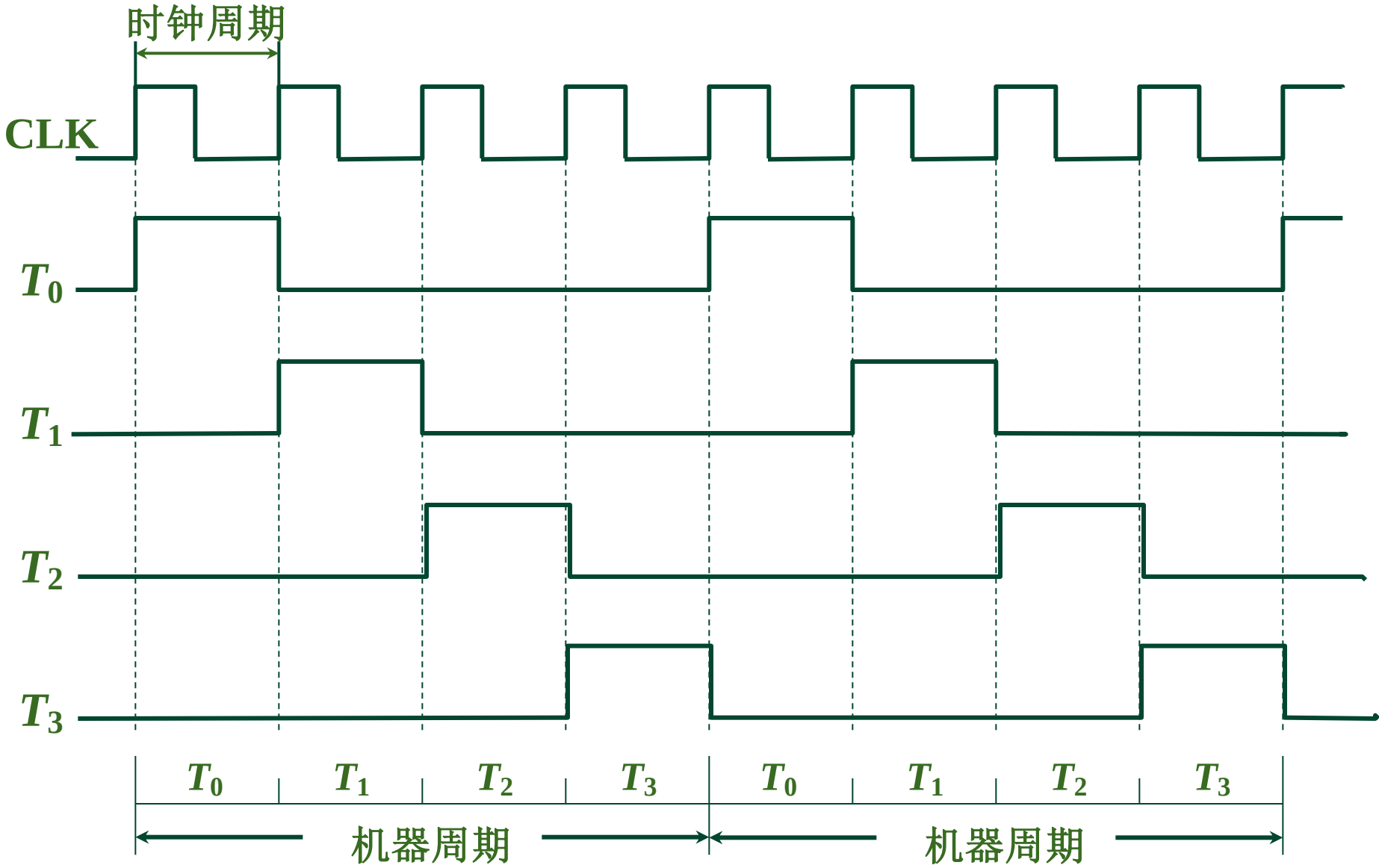
- 微程序控制型：将机器指令分解为基本微命令序列，用二进制码表示微命令，并编成微指令，多条微指令形成微程序。每种机器指令对应一段微程序，在制造CPU时固化在CPU中的一个控制存储器中（CS或CM）中。执行一条机器指令时，CPU依次从CS中取微指令，从而产生微命令。

一、组合逻辑控制单元框图

1. CU 外特性



2. 节拍信号

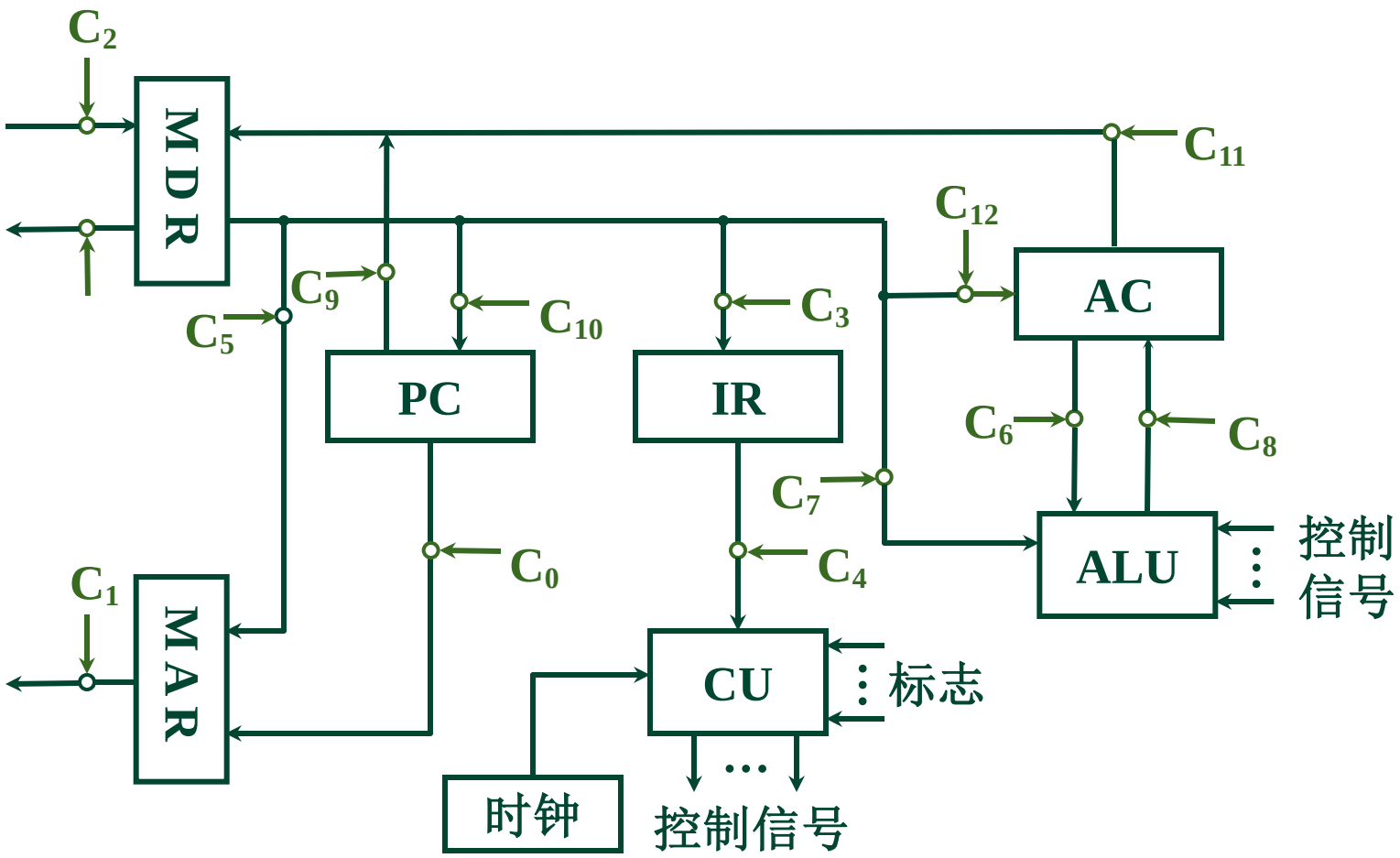


二、微操作的节拍安排

假设：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 ⟶ MAR　　原则二

1 ⟶ R

T_1 M (MAR) ⟶ MDR　　原则二

(PC) + 1 ⟶ PC

T_2 MDR ⟶ IR　　原则三

OP (IR) ⟶ ID

3. 间址周期 微操作的 节拍安排

T_0 Ad (IR) ⟶ MAR

1 ⟶ R

T_1 M (MAR) ⟶ MDR

T_2 MDR ⟶ Ad (IR)

4. 执行周期 微操作的 节拍安排

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) ⟶ L (AC)	$AC_0 \longrightarrow AC_n$
⑤ STP	T_0		
	T_1		
	T_2	0 ⟶ G	
⑥ ADD X	T_0	Ad (IR) ⟶ MAR	1 ⟶ R
	T_1	M (MAR) ⟶ MDR	
	T_2	(AC) + (MDR) ⟶ AC	
⑦ STA X	T_0	Ad (IR) ⟶ MAR	1 ⟶ W
	T_1	AC ⟶ MDR	
	T_2	MDR ⟶ M (MAR)	

⑧ LDA X

T_0 Ad (IR) ⟶ MAR　1 ⟶ R

T_1 M (MAR) ⟶ MDR

T_2 MDR ⟶ AC

⑨ JMP X

T_0

T_1

T_2 Ad (IR) ⟶ PC

⑩ BAN X

T_0

T_1

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

5. 中断周期 微操作的 节拍安排

T_0　0 ⟶ MAR　　1 ⟶ W　　硬件关中断

T_1　PC ⟶ MDR

T_2　MDR ⟶ M (MAR)　　向量地址 ⟶ PC

中断隐指令完成

三、组合逻辑设计步骤

- 确定指令功能和指令格式。
- 确定机器周期、节拍和时钟周期，确定机器周期是固定的还是可变长的。
- 根据指令功能和CPU的结构图，进行微操作命令分析
- 对指令微操作进行节拍安排，确定每条指令所需的机器周期及在各机器周期需完成的操作，排出操作时间表。
- 根据操作时间表写出微操作的逻辑表达式，即

  微操作＝周期·节拍·指令码·其他条件
- 根据微操作的表达式，画出组合逻辑电路

三、组合逻辑设计步骤

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⟶ MAR						
			1⟶ R						
	T_1		M(MAR) ⟶MDR						
			(PC) +1 ⟶PC						
	T_2		MDR⟶ IR						
			OP(IR) ⟶ID						
		I	1⟶ IND						
		$\bar{I}$	1⟶ EX						

间址特征

三、组合逻辑设计步骤

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

间址周期标志

三、组合逻辑设计步骤

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) ⟶ MAR						
			1 ⟶ R						
			1 ⟶ W						
	T_1		M(MAR) ⟶ MDR						
			AC ⟶ MDR						
	T_2		(AC)+(MDR) →AC						
			MDR ⟶ M(MAR)						
			MDR ⟶ AC						
			0 ⟶ AC						

三、组合逻辑设计步骤

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⟶ MAR	1	1	1	1	1	1
			1⟶ R	1	1	1	1	1	1
	T_1		M(MAR) ⟶MDR	1	1	1	1	1	1
			(PC) +1⟶ PC	1	1	1	1	1	1
	T_2		MDR⟶ IR	1	1	1	1	1	1
			OP(IR)⟶ ID	1	1	1	1	1	1
		I	1⟶ IND			1	1	1	1
		$\bar{I}$	1⟶ EX	1	1	1	1	1	1

三、组合逻辑设计步骤

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

三、组合逻辑设计步骤

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) ⟶ MAR			1	1	1	
			1⟶ R			1		1	
			1 ⟶ W				1		
	T_1		M(MAR) ⟶ MDR			1		1	
			AC⟶ MDR				1		
	T_2		(AC)+(MDR) ⟶AC			1			
			MDR⟶ M(MAR)				1		
			MDR ⟶ AC					1	
			0⟶ AC	1					

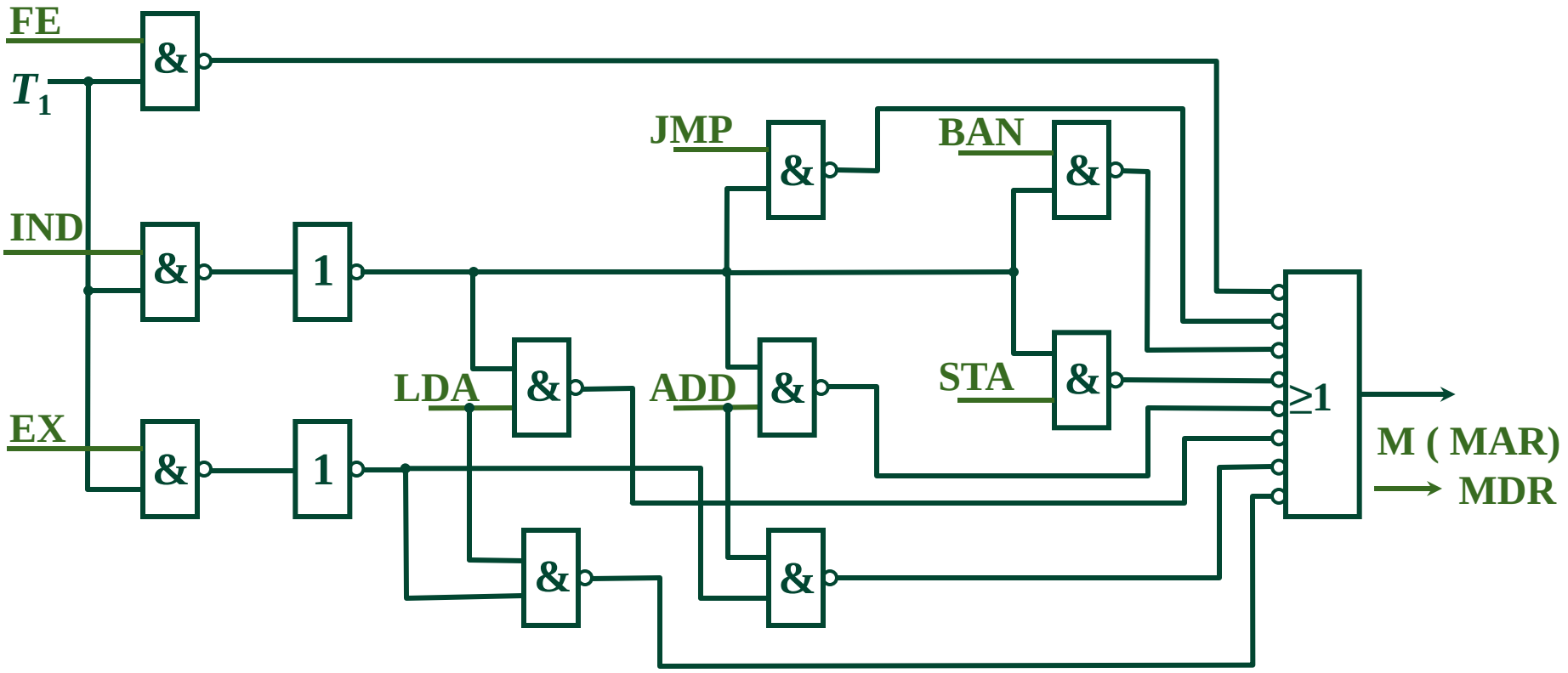
2. 写出微操作命令的最简表达式

$$M(MAR) \longrightarrow MDR$$

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

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

3. 画出逻辑图



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——基本概念

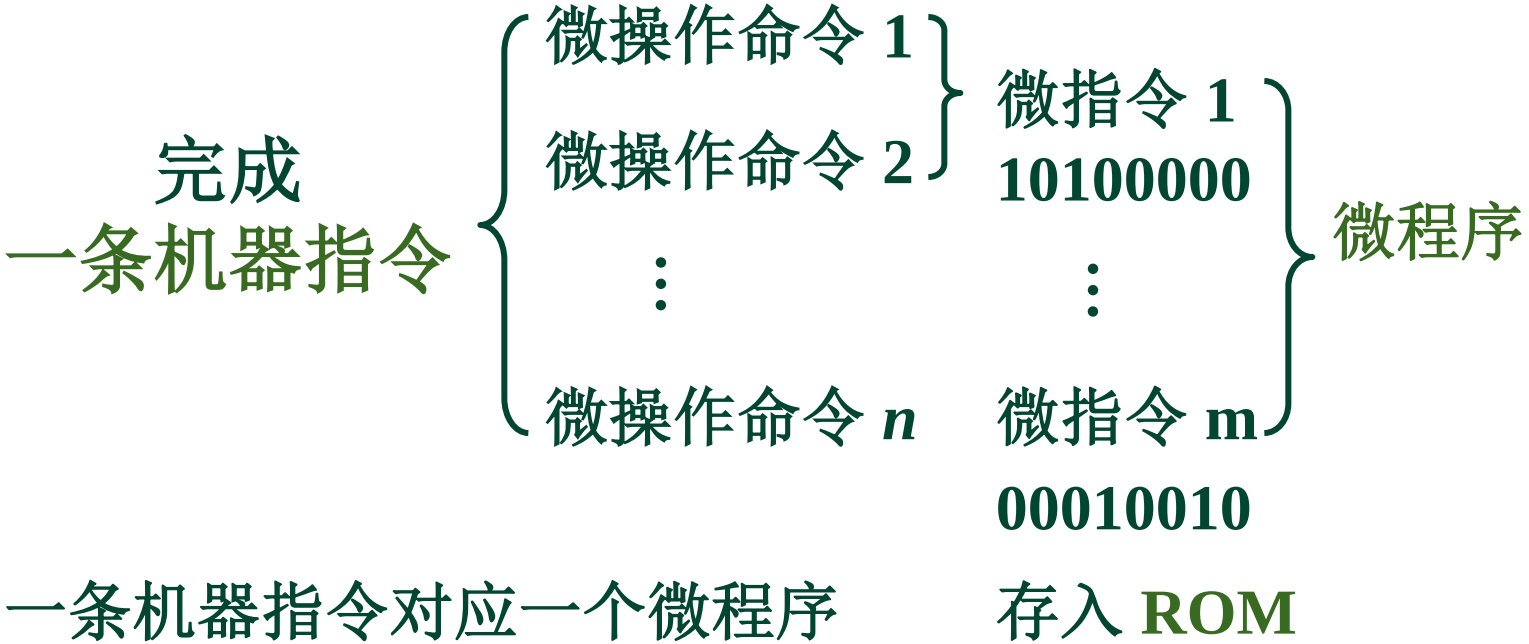
- 微命令：微程序控制计算机中的微操作控制信号。
- 微操作：执行部件接受微指令后所进行的操作，是指令序列中最基本、不可分割的动作。
  - 例如，一条加法指令要分成四步完成：取指令，计算地址，取数，加法运算，每一步实现需若干个微操作
- 微指令：在微程序控制的计算机中，同时发出的控制信号所执行的一组微操作。
  - 是在机器的一个节拍中，一组实现一定操作功能的微命令
  - 将一条指令分解成若干条微指令，按次序执行微指令，即可实现指令的功能
- 微程序：由微指令组成的序列称为微程序。一个微程序的功能对应一条机器指令的功能。

10.2 微程序设计—— 基本概念

- 控制存储器：微程序存放于存储器中，由于该存储器主要存放控制命令（信号）和下一条执行的微指令地址（简称下址），因此被称为控制存储器。
  - 执行一条指令就是执行一段存放在控制存储器中的微程序。
  - 用ROM实现，因为一台计算机指令系统是固定的，所以微程序是固定的，所以控制存储器可以用ROM实现
- 微周期：执行一条微指令和取出下一条微指令所需时间
  - 通常一个微周期与一个CPU周期时间相等
- 相斥性微命令：不能在一个微周期出现的微命令
  - 如读命令和写命令
- 相容性微命令：能在一个微周期出现的微命令。

一、微程序设计思想的产生

1951 英国剑桥大学教授 Wilkes



存储逻辑

微指令的格式

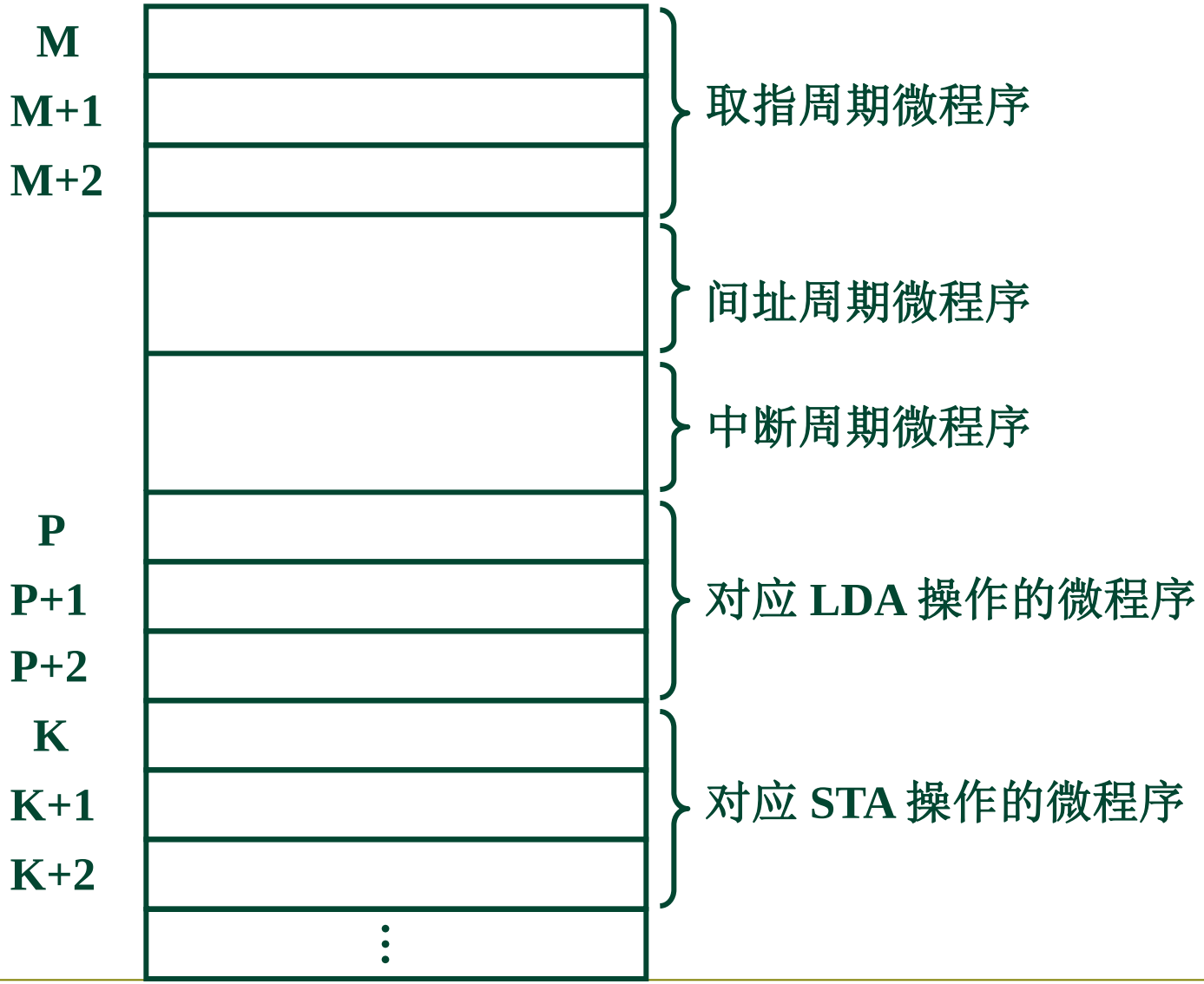
微指令：

控制字段	下址字段

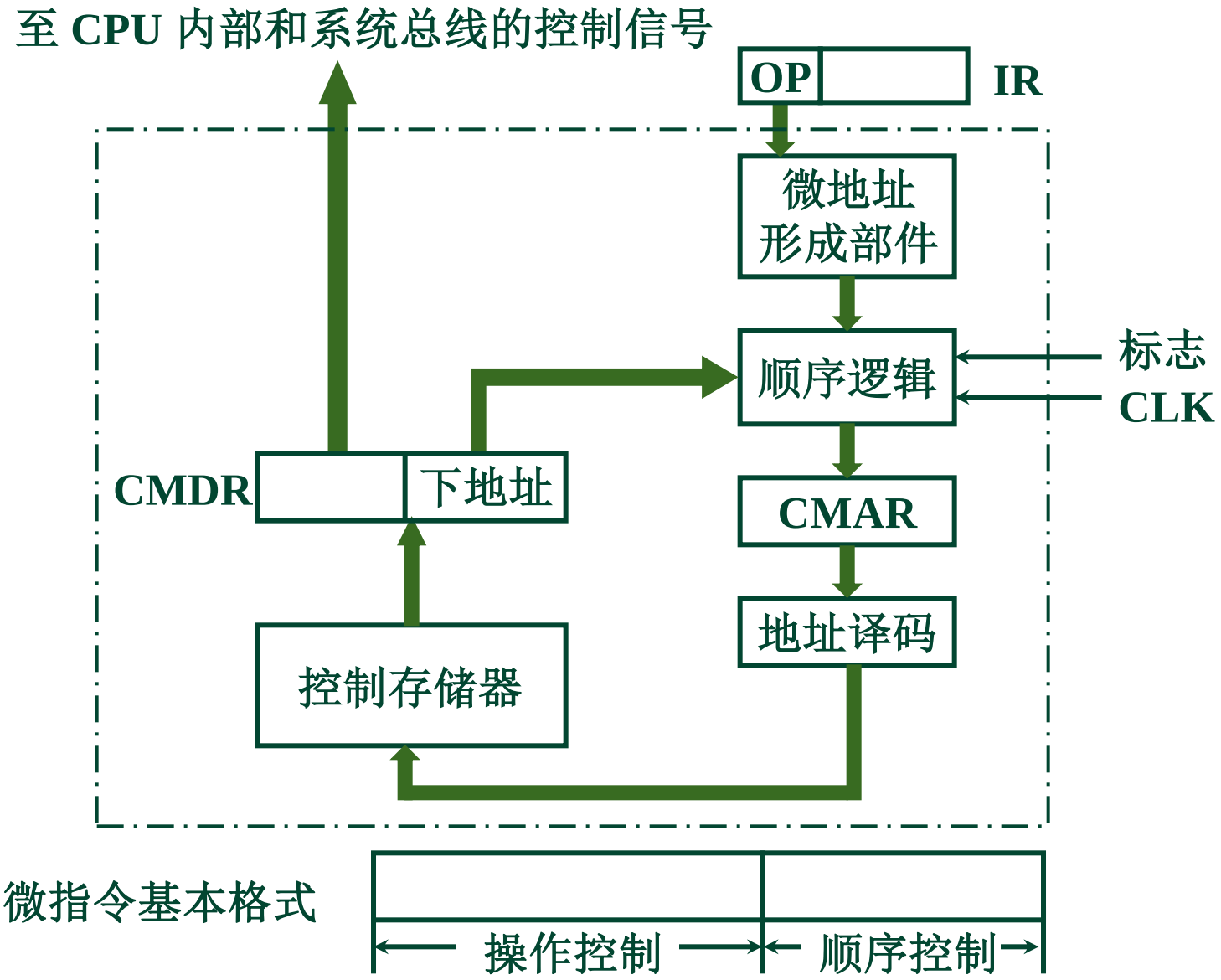
- 控制字段：操作控制，发出各种控制信号
- 下址字段：顺序控制，指出下条微指令地址，以控制微指令序列的执行顺序

二、微程序控制单元框图及工作原理

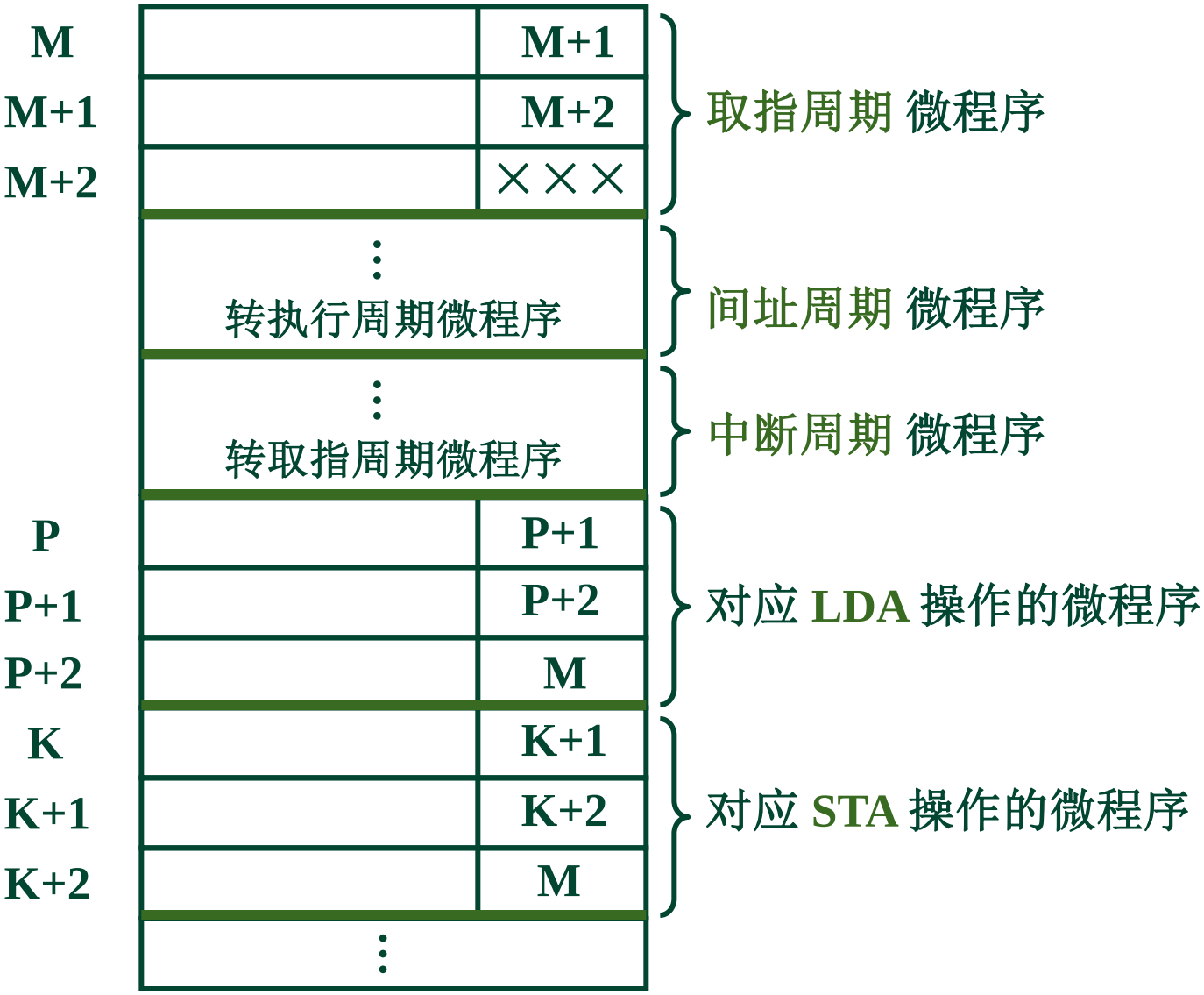
1. 机器指令对应的微程序



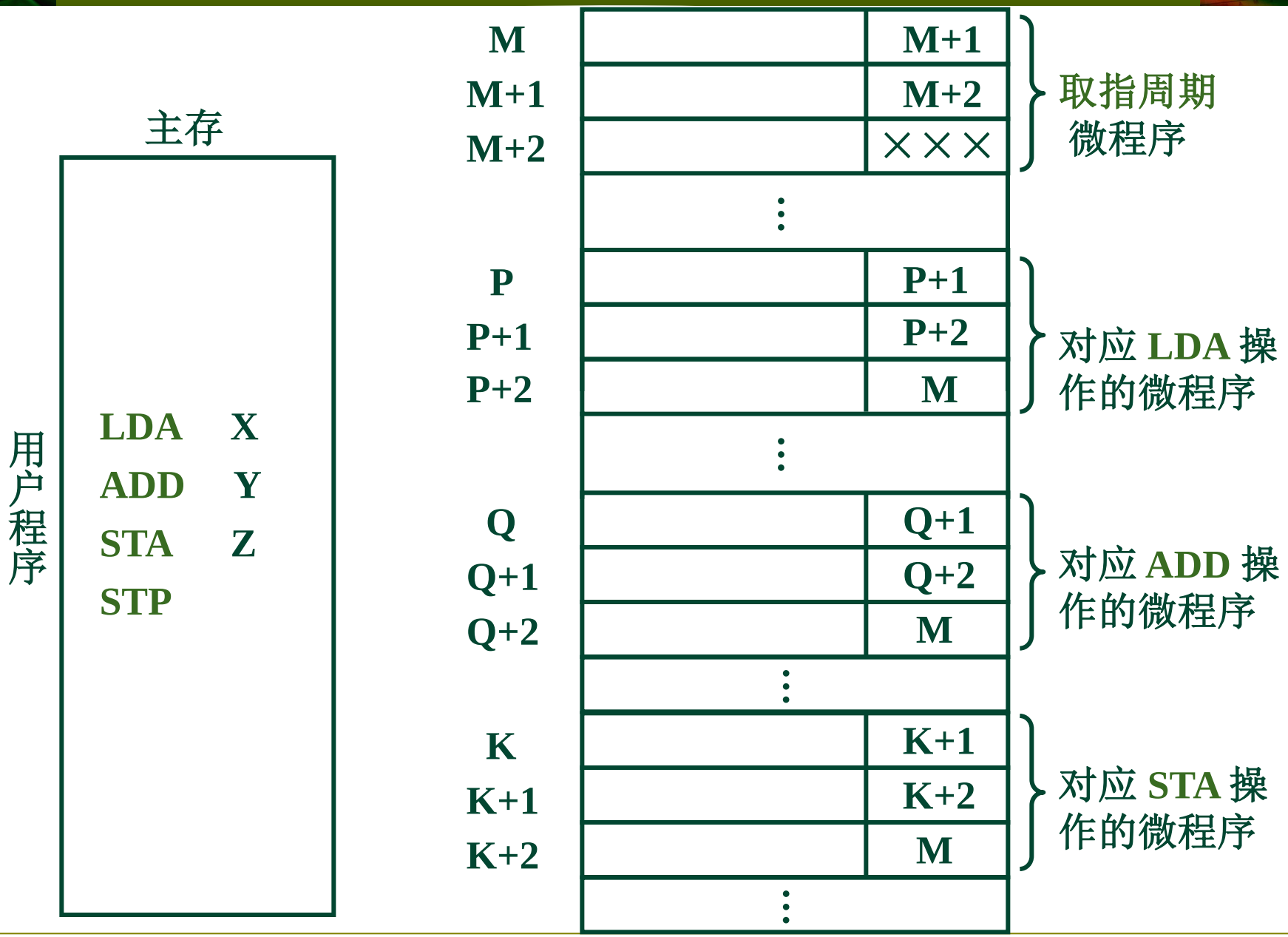
2. 微程序控制单元的基本框图



二、微程序控制单元框图及工作原理



3. 工作原理


主存
用户程序
LDA X
ADD Y
STA Z
STP
控存
M
M+1
M+2
M+1
M+2
×××
取指周期微程序
P
P+1
P+2
P+1
P+2
M
对应 LDA 操作的微程序
Q
Q+1
Q+2
Q+1
Q+2
M
对应 ADD 操作的微程序
K
K+1
K+2
K+1
K+2
M
对应 STA 操作的微程序

3. 工作原理

(1) 取指阶段　　执行取指微程序

M ⟶ CMAR

CM (CMAR) ⟶ CMDR

由 CMDR 发命令

形成下条微指令地址　M + 1

Ad (CMDR) ⟶ CMAR

CM (CMAR) ⟶ CMDR

由 CMDR 发命令

形成下条微指令地址　M + 2

Ad (CMDR) ⟶ CMAR

CM (CMAR) ⟶ CMDR

由 CMDR 发命令

(2) 执行阶段　执行 LDA 微程序

OP (IR) ⟶ 微地址形成部件 ⟶ CMAR　　(P ⟶ CMAR)

CM (CMAR) ⟶ CMDR

由 CMDR 发命令

形成下条微指令地址 P + 1

Ad (IR) ⟶ MAR　　1 ⟶ R

Ad (CMDR) ⟶ CMAR

CM (CMAR) ⟶ CMDR

由 CMDR 发命令

形成下条微指令地址 P + 2

M (MAR) ⟶ MDR

Ad (CMDR) ⟶ CMAR

CM (CMAR) ⟶ CMDR

由 CMDR 发命令

形成下条微指令地址 M

MDR ⟶ AC

Ad (CMDR) ⟶ CMAR　　(M ⟶ CMAR)

(3) 取指阶段　执行取指微程序

M ⟶ CMAR

CM (CMAR) ⟶ 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

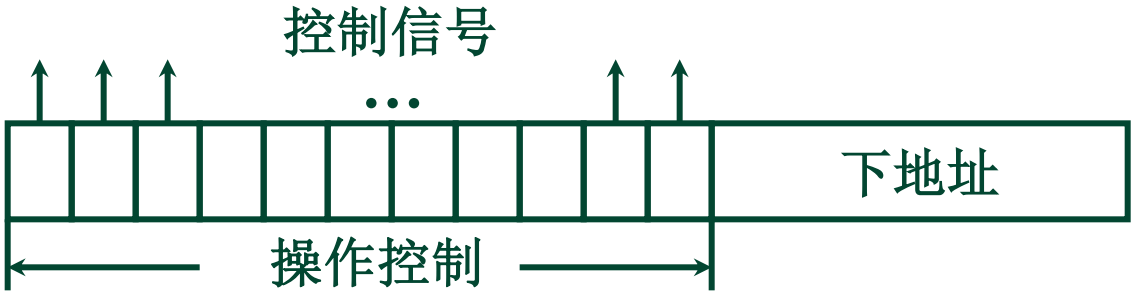
- 微指令的 操作控制字段如何形成微操作命令
- 微指令的 后续地址如何形成

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



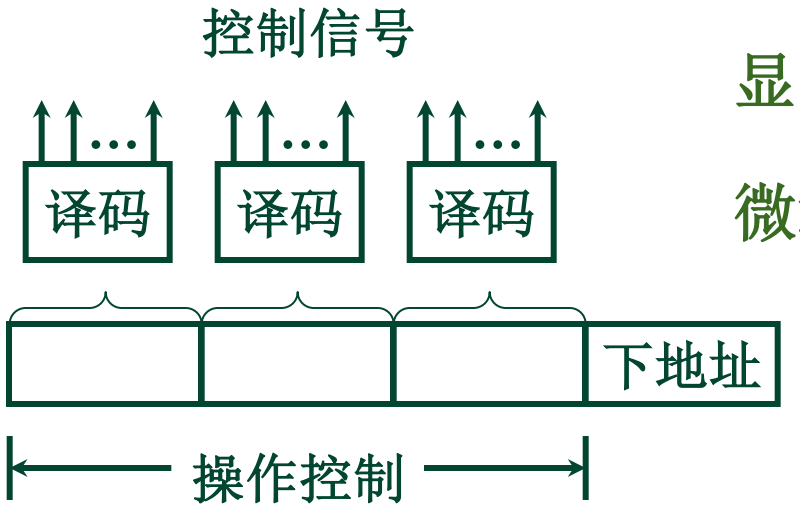
速度最快

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



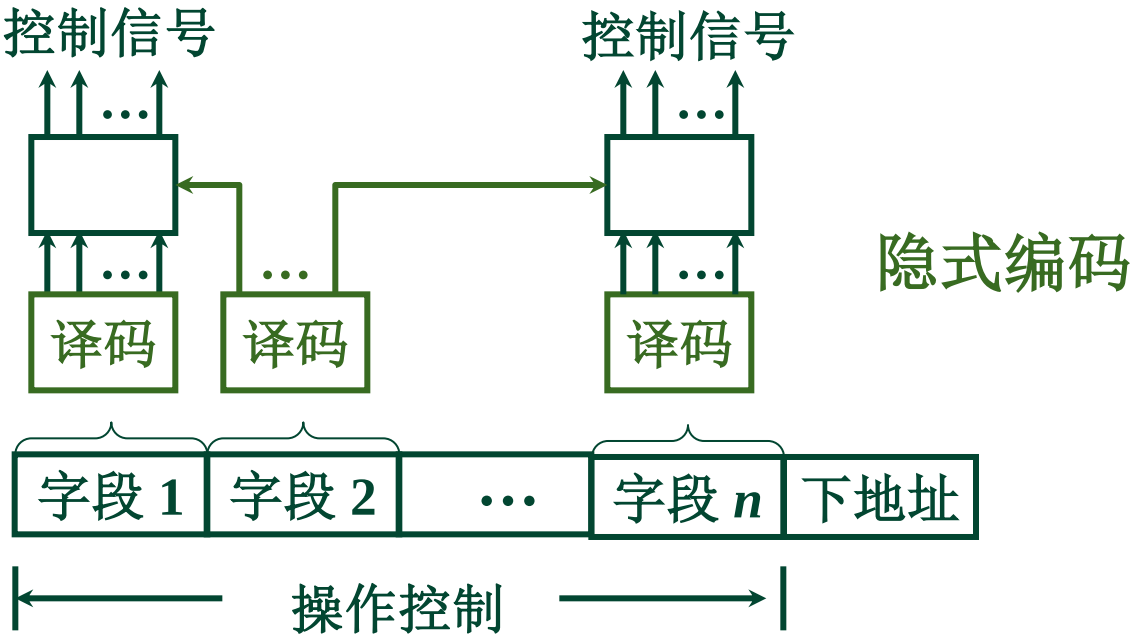
显式编码

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



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(断定方式)

2. 根据机器指令的 操作码 形成：根据机器指令的操作码，由微地址形成部件形成对应该机器指令微程序的首地址。

3. 增量计数器（顺序地址）

$$(CMAR) + 1 \longrightarrow CMAR$$

4. 分支转移（转移指令）

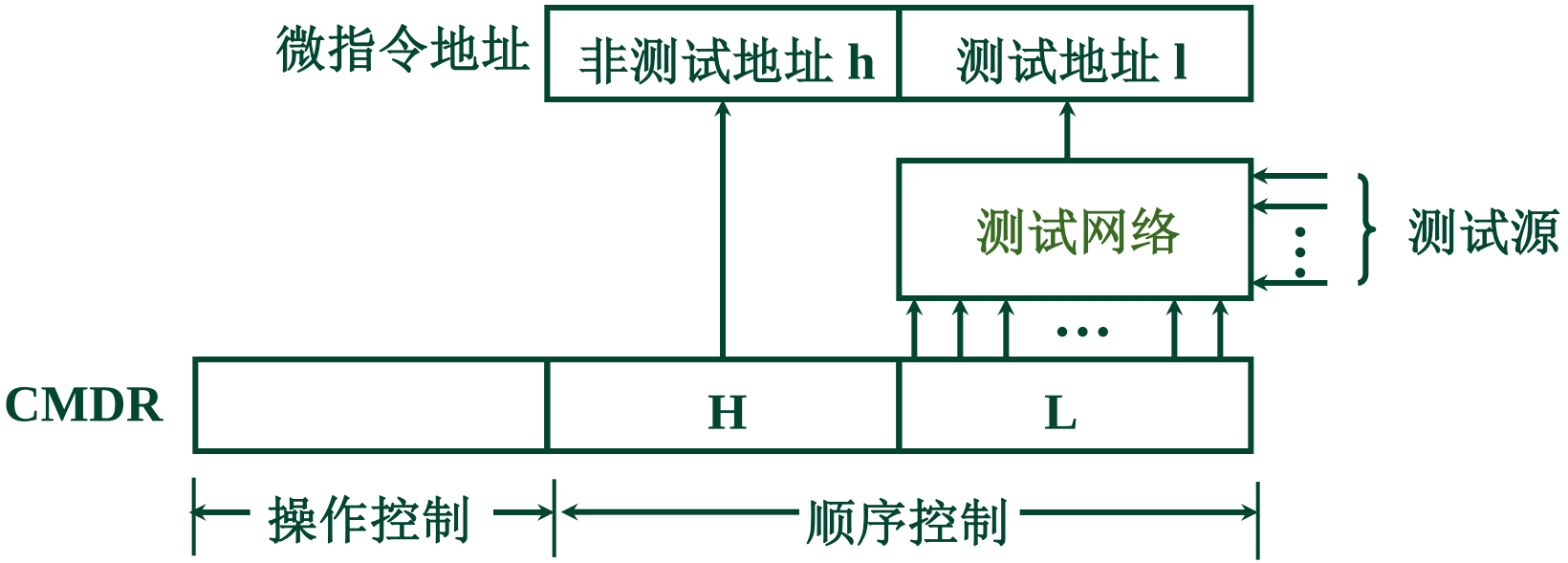
操作控制字段	转移方式	转移地址

转移方式　　指明判别条件

转移地址　　指明转移成功后的去向

四、微指令序列地址的形成

5. 通过测试网络

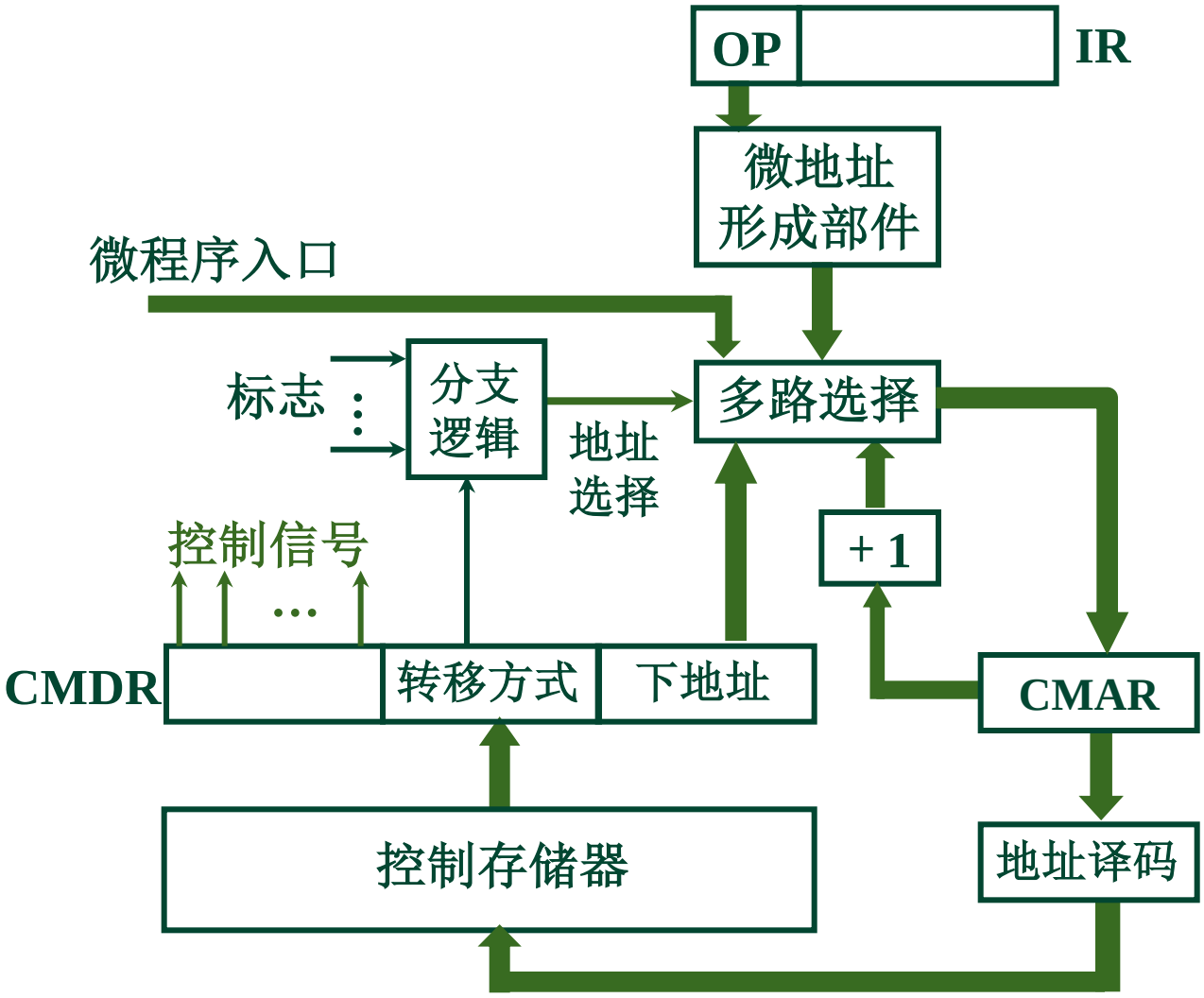


6. 由硬件产生微程序入口地址

加电后，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、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

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

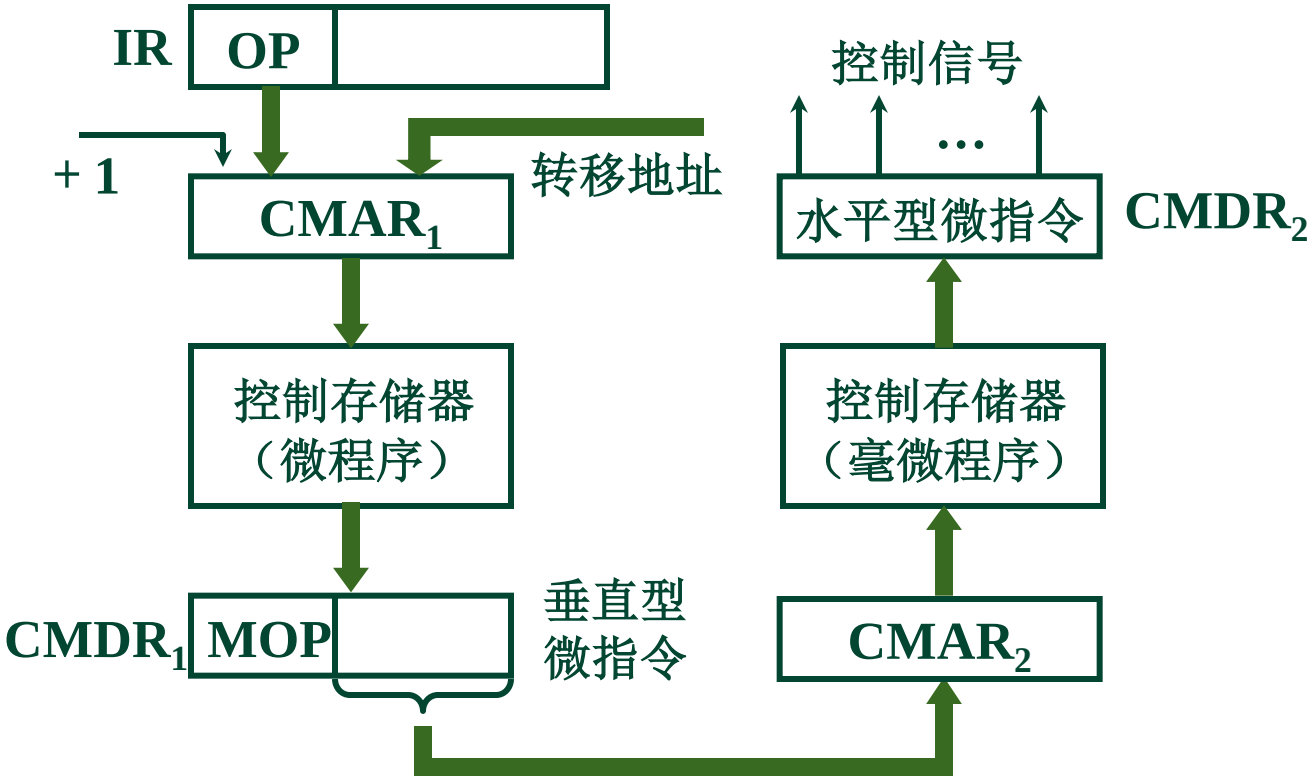
1. 毫微程序设计的基本概念

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



串行 微程序控制

取第 *i* 条微指令	执行第 *i* 条微指令	取第 *i*+1 条微指令	执行第 *i*+1 条微指令

并行 微程序控制

取第 *i* 条微指令	执行第 *i* 条微指令		
	取第 *i*+1 条微指令	执行第 *i*+1 条微指令	
		取第 *i*+2 条微指令	执行第 *i*+2 条微指令

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 ⟶ MAR　　1 ⟶ R

T_1　M (MAR) ⟶ MDR　　(PC) + 1 ⟶ PC

T_2　MDR ⟶ IR　　OP (IR) ⟶ 微地址形成部件

若需考虑 T_2 如何安排这条微指令？

则取指操作需 3 条微指令

Ad (CMDR) ⟶ CMAR

OP (IR) ⟶ 微地址形成部件 ⟶ 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 ⟶ MAR　　1 ⟶ R

T_1 Ad (CMDR) ⟶ CMAR

T_2 M (MAR) ⟶ MDR　(PC)+1 ⟶ PC

T_3 Ad (CMDR) ⟶ CMAR

T_4 MDR ⟶ IR　　OP (IR) ⟶ 微地址形成部件

T_5 OP (IR) ⟶ 微地址形成部件 ⟶ 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

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　0 ⟶ AC

T_1　Ad (CMDR) ⟶ CMAR

② COM 指令

T_0　$\overline{\text{AC}}$ ⟶ AC

T_1　Ad (CMDR) ⟶ CMAR

③ SHR 指令

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

T_1　　Ad (CMDR) ⟶ CMAR

④ CSL 指令

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

T_1　　Ad (CMDR) ⟶ CMAR

⑤ STP 指令

T_0　　0 ⟶ G

T_1　　Ad (CMDR) ⟶ CMAR

• 访存指令

⑥ ADD 指令

T_0 Ad (IR) ⟶ MAR　　1 ⟶ R

T_1 Ad (CMDR) ⟶ CMAR

T_2 M (MAR) ⟶ MDR

T_3 Ad (CMDR) ⟶ CMAR

T_4 (AC) + (MDR) ⟶ AC

T_5 Ad (CMDR) ⟶ CMAR

⑦ STA 指令

T_0 Ad (IR) ⟶ MAR　　1 ⟶ W

T_1 Ad (CMDR) ⟶ CMAR

T_2 AC ⟶ MDR

T_3 Ad (CMDR) ⟶ CMAR

T_4 MDR ⟶ M (MAR)

T_5 Ad (CMDR) ⟶ CMAR

⑧ LDA 指令

T_0 Ad (IR) ⟶ MAR　　1 ⟶ R

T_1 Ad (CMDR) ⟶ CMAR

T_2 M (MAR) ⟶ MDR

T_3 Ad (CMDR) ⟶ CMAR

T_4 MDR ⟶ AC

T_5 Ad (CMDR) ⟶ CMAR

• 转移类指令

⑨ JMP 指令

T_0　　Ad (IR) ⟶ PC

T_1　　Ad (CMDR) ⟶ CMAR

⑩ BAN 指令

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

T_1　　Ad (CMDR) ⟶ 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

38 条微指令中有 19 条

是关于后续微指令地址 ⟶ CMAR

其中 { 1 条　OP (IR) ⟶ 微地址形成部件 ⟶ CMAR
　　 { 18 条　Ad (CMDR) ⟶ CMAR

若用 Ad (CMDR) 直接送控存地址线

则 省去了输至 CMAR 的时间，省去了 CMAR

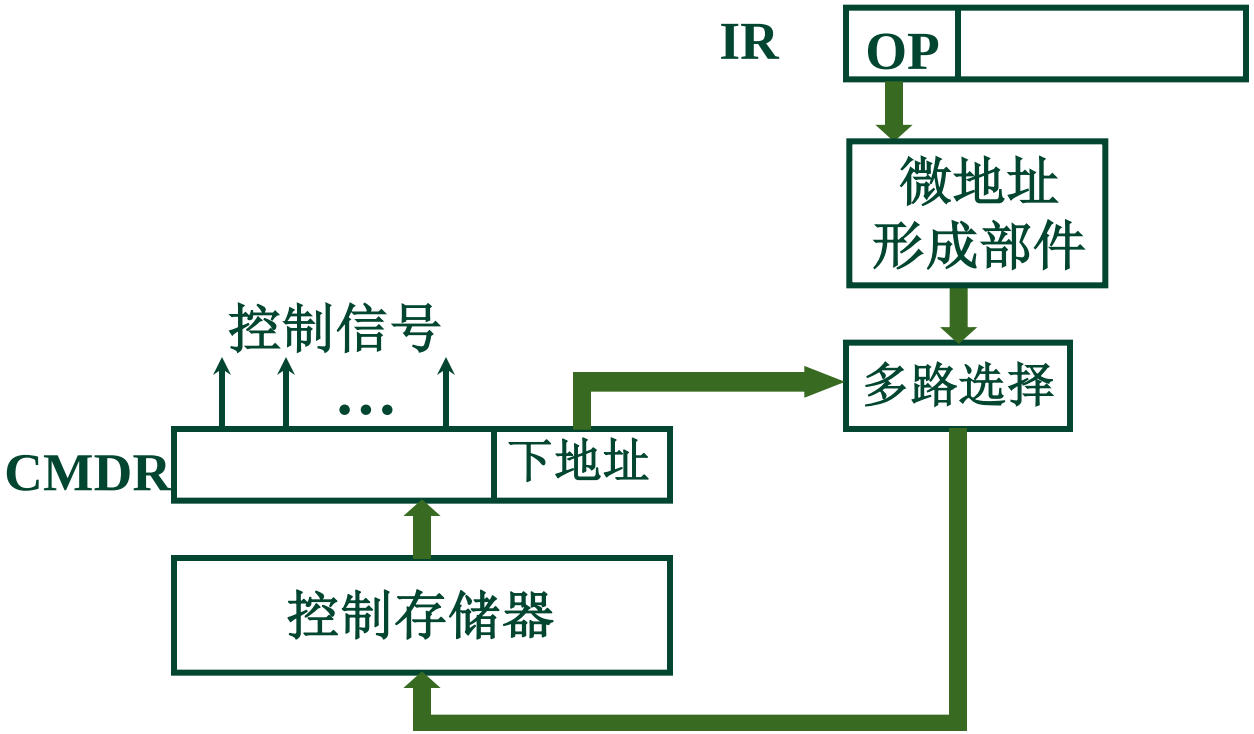
同理　OP (IR) ⟶ 微地址形成部件 ⟶ 控存地址线

可省去 19 条微指令，2 个微操作

38 － 19 ＝ 19　　20 － 2 ＝ 18

下地址字段最少取 5 位　　操作控制字段最少取 18 位

(5) 省去了 CMAR 的控制存储器



考虑留有一定的余量

取操作控制字段　18 位→24 位
下地址字段　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

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		1	0
	02					1					×	×	×	×	×	×
CLA	03										0	0	0	0	0	0
COM	04										0	0	0	0	0	0
ADD	10		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

PC→MAR

PC+1→PC

MDR→IR

Ad(IR)→MAR

1→R

M(MAR)→MDR

1→R

Ad(IR)→MAR

M(MAR)→MDR

第十章 控制单元的设计 回顾

- **10.1 组合逻辑设计**
  - **CU框图、微操作的节拍安排（三个原则）**
  - **设计步骤：**列操作时间表、写逻辑表达式、画逻辑图
- **10.2 微程序设计**
  - **设计思想：**微程序、微指令、控存
  - **CU框图和工作原理**
  - **关键：微指令编码方式**（直接、字段直接和间接编码）和**后续地址形成**（下地址、操作码经微地址形成部件、增量计数、分支转移、测试网络、硬件产生）
  - **微指令格式：**水平型、垂直型；静态、动态微程序设计
  - **设计步骤：**微操作分析和节拍安排、确定微指令格式、编写微指令码点

Computer Organization

Thank You !

Computer Architecture Research Institute · Computer Organization