第10章　控制单元的设计

10.1　组合逻辑设计

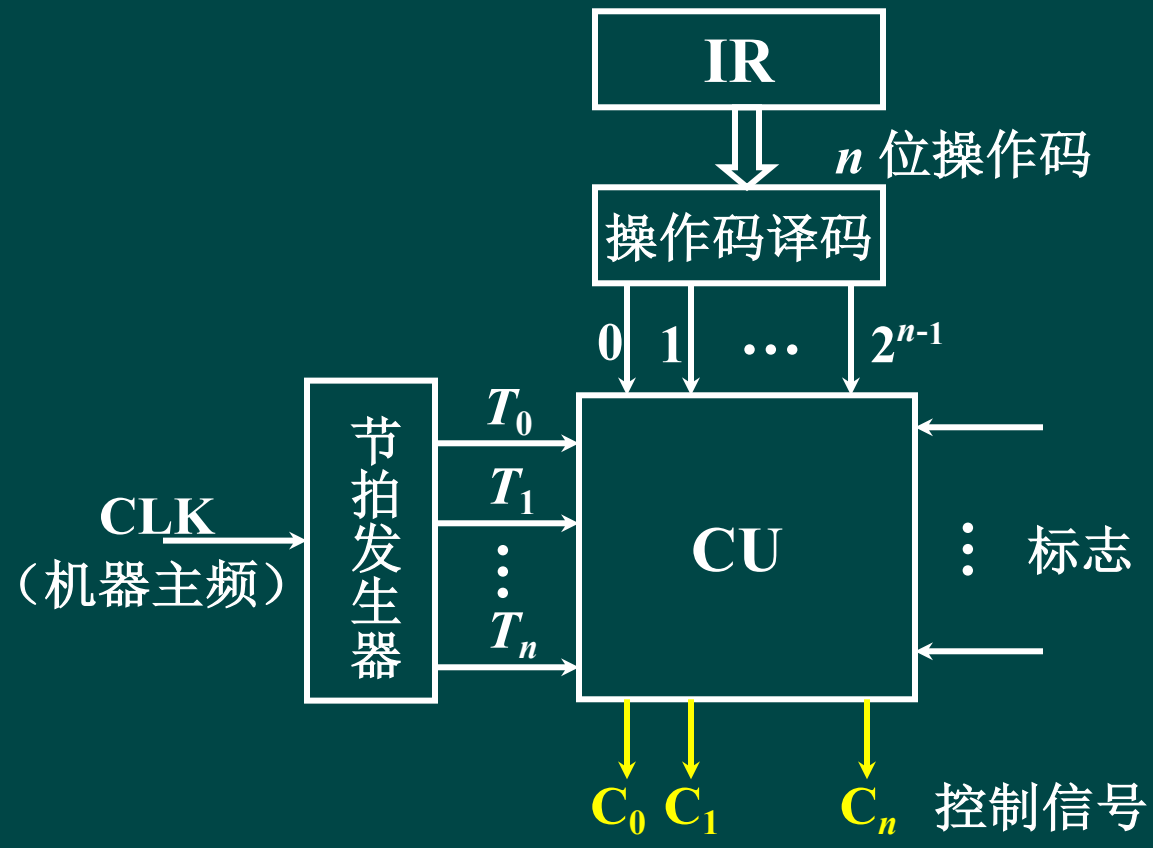
10.2　微程序设计

邵清
上海理工大学

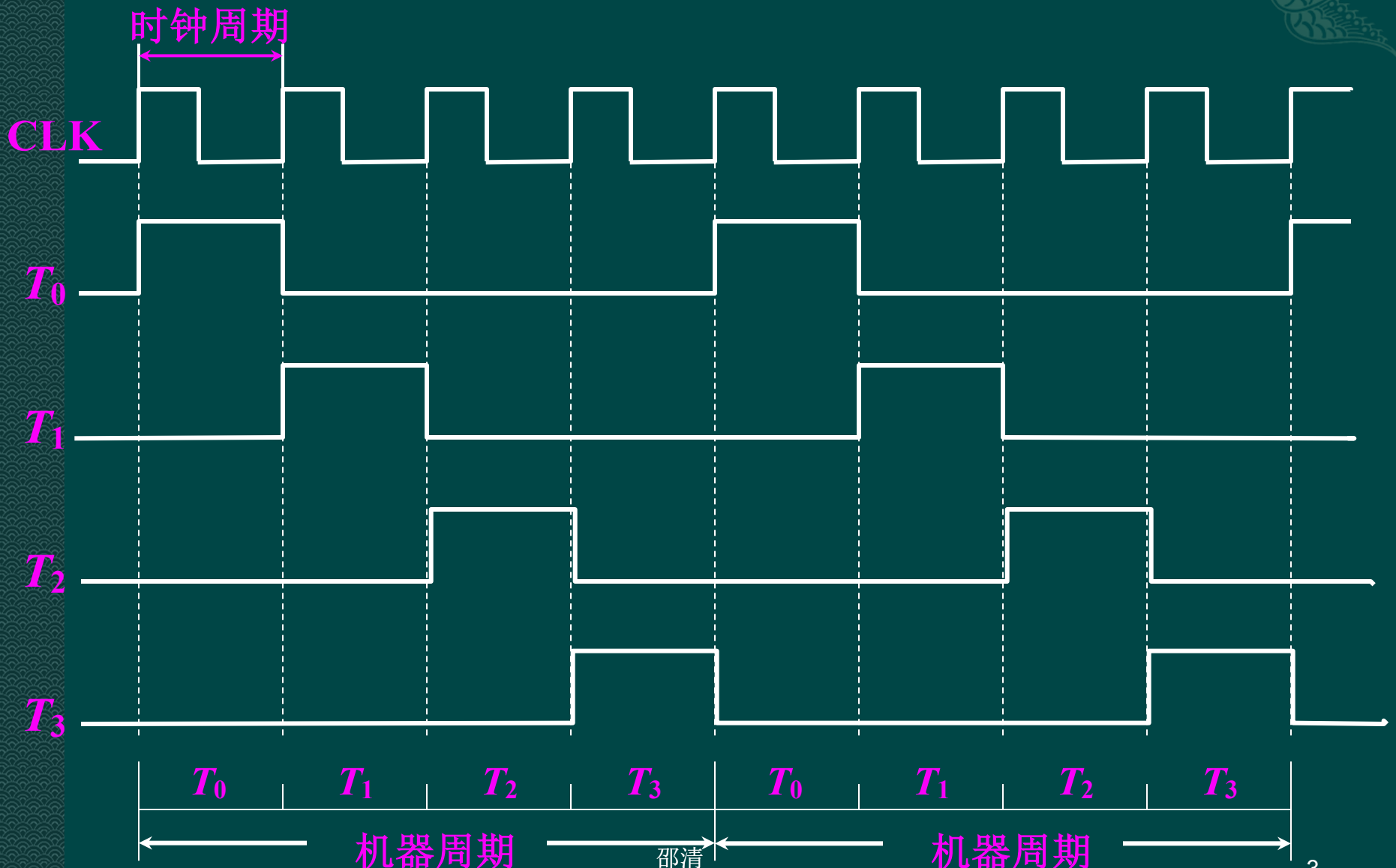
10.1 组合逻辑设计

一、组合逻辑控制单元框图

1. CU 外特性



2. 节拍信号

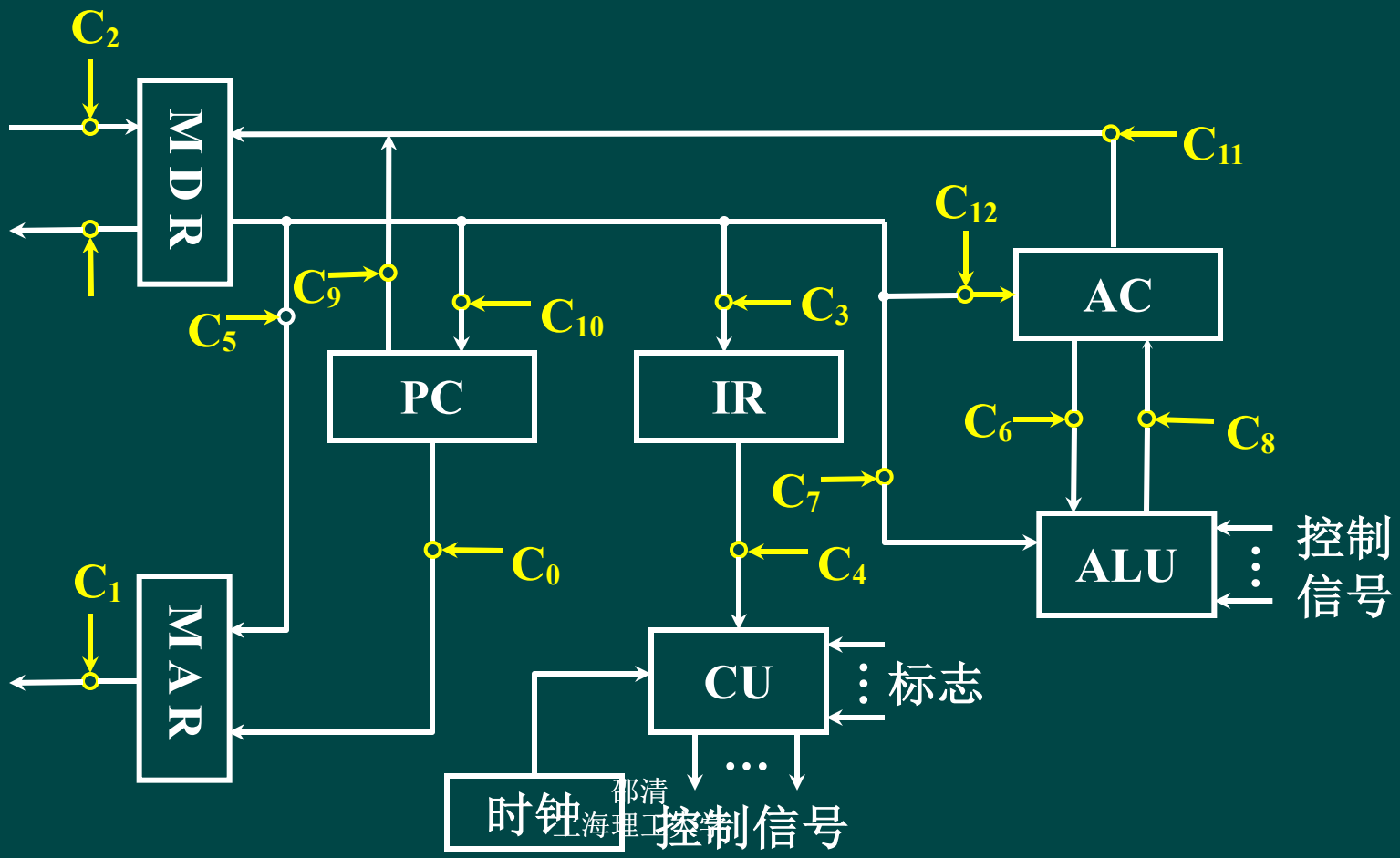


二、微操作的节拍安排

任一微操作均由 统一基准时标 的时序信号控制

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	原则二
	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

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
		$\overline{\text{IND}}$	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					

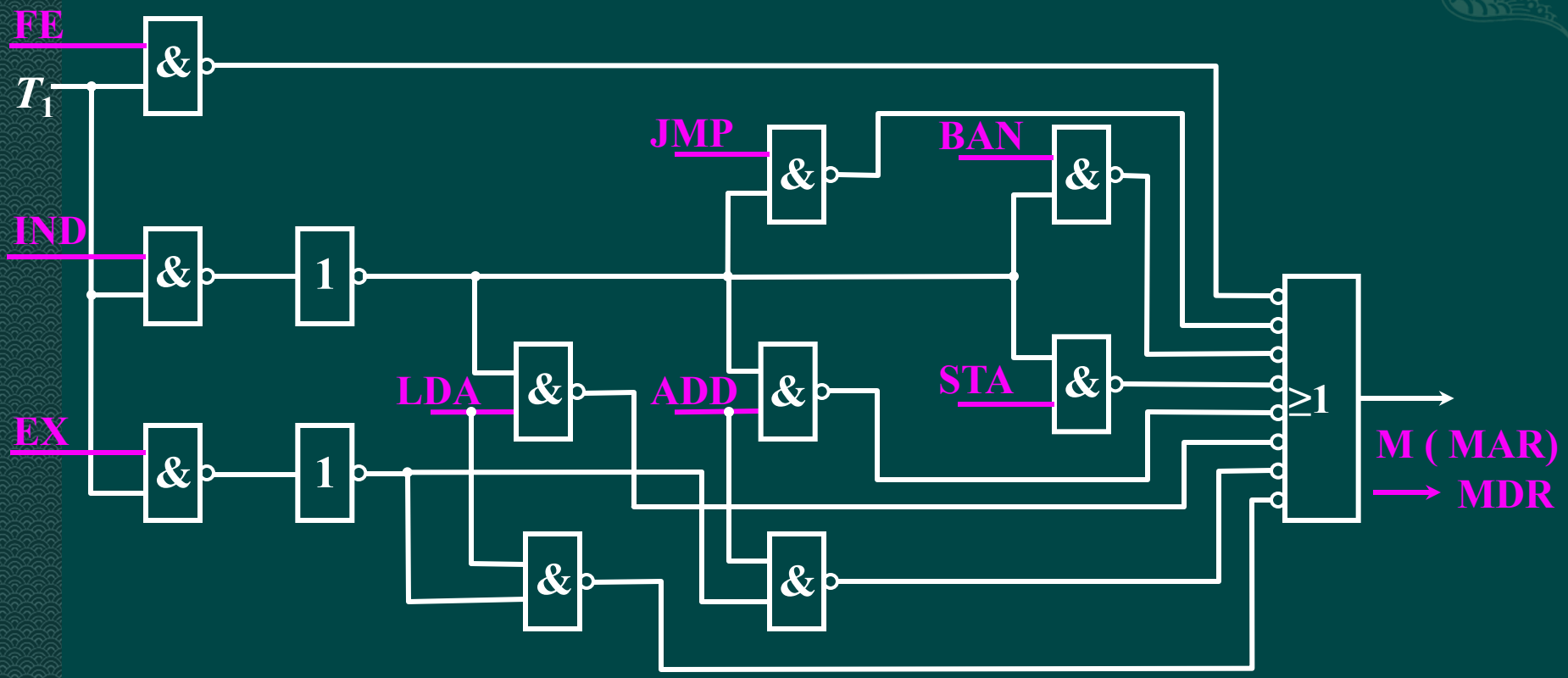
2. 写出微操作命令的最简表达式

$$M(MAR) \longrightarrow MDR$$

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

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

3. 画出逻辑图

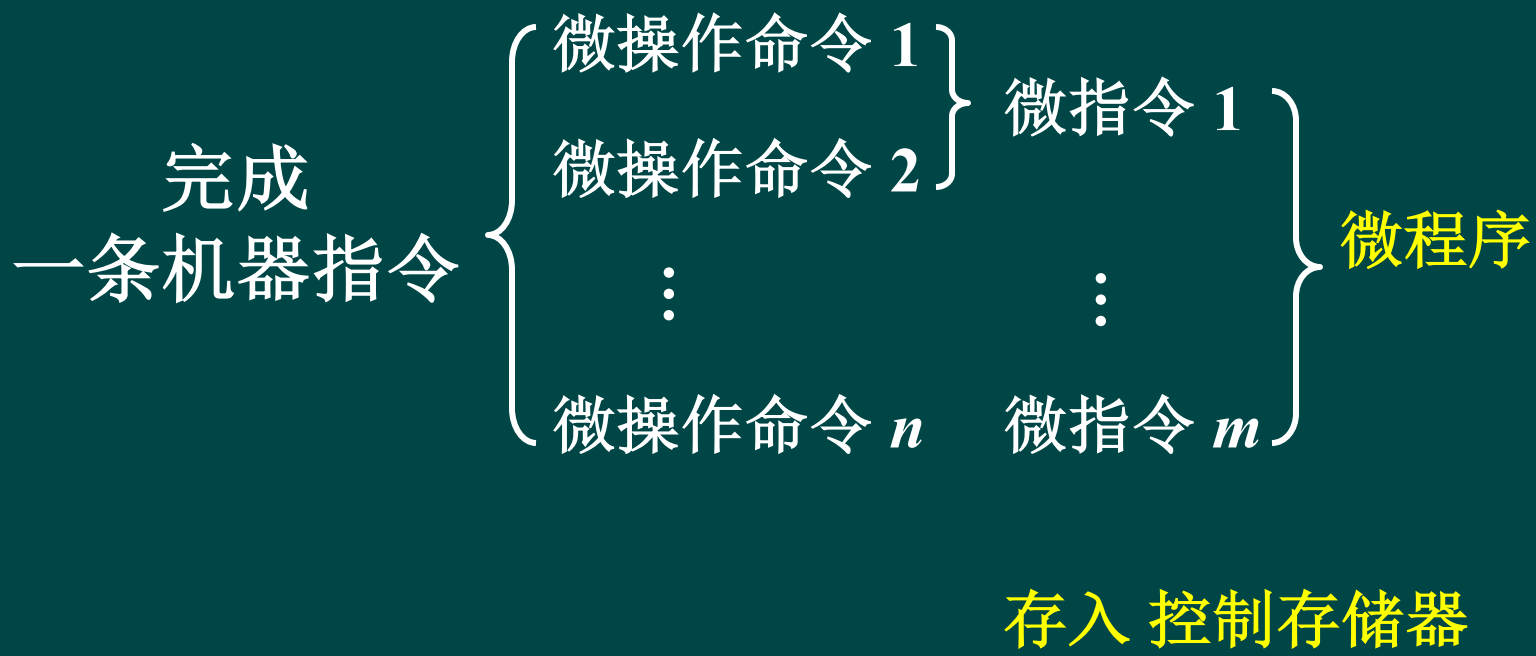


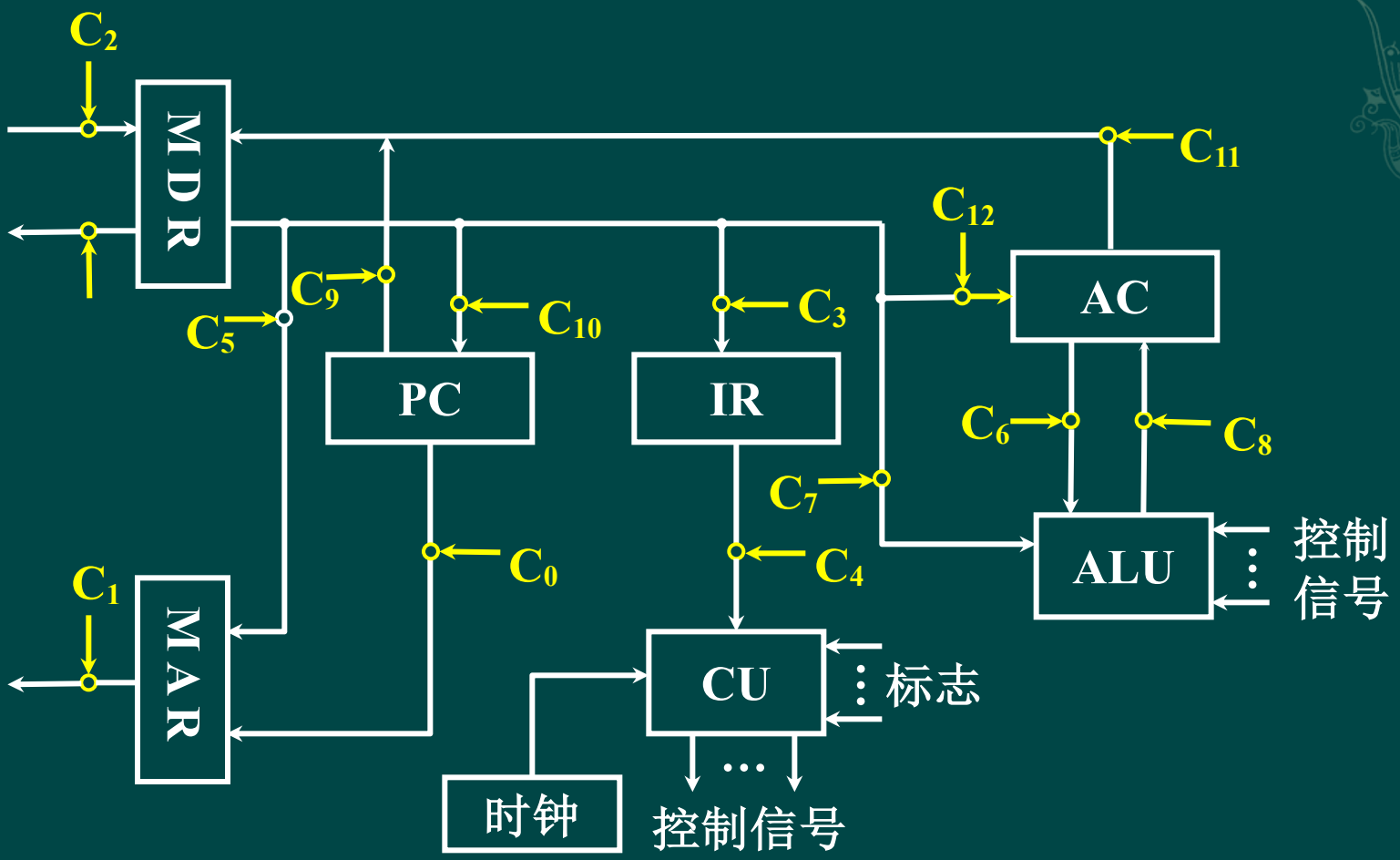
特点

- 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

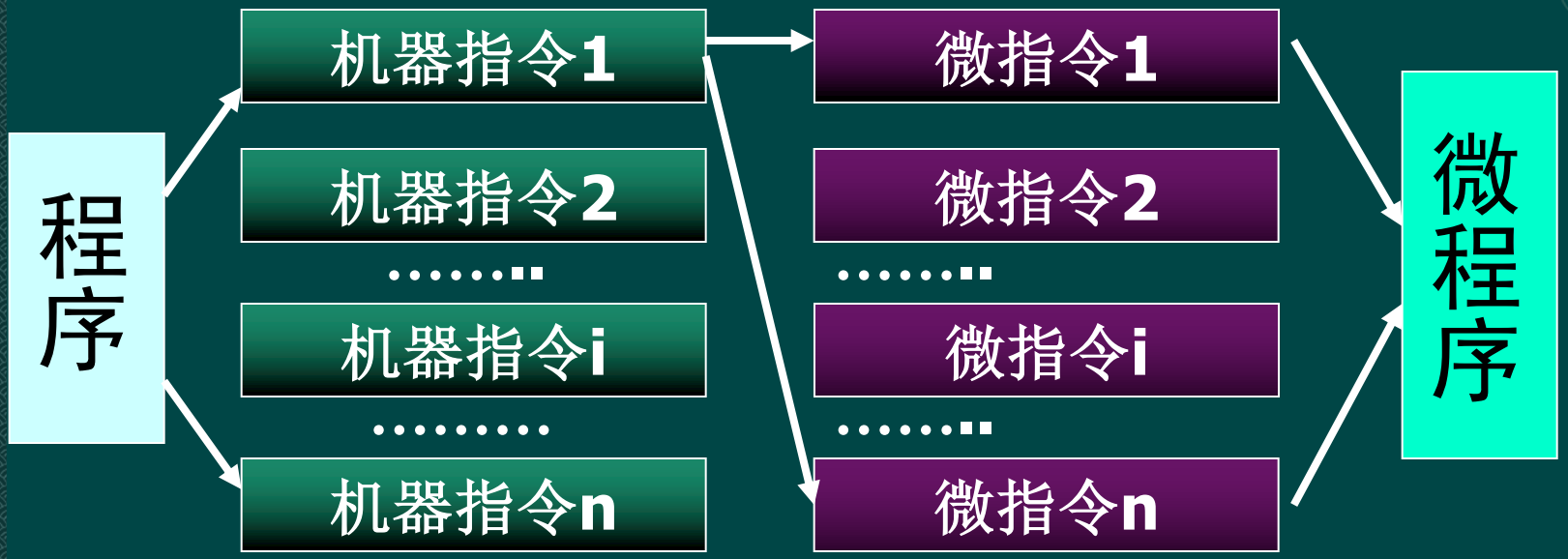
一、微程序设计思想的产生




C2
MDR
C11
C12
AC
C9
C10
C3
C5
PC
IR
C6
C8
C7
C0
C4
ALU
控制
信号
C1
MAR
CU
标志
…
时钟
控制信号

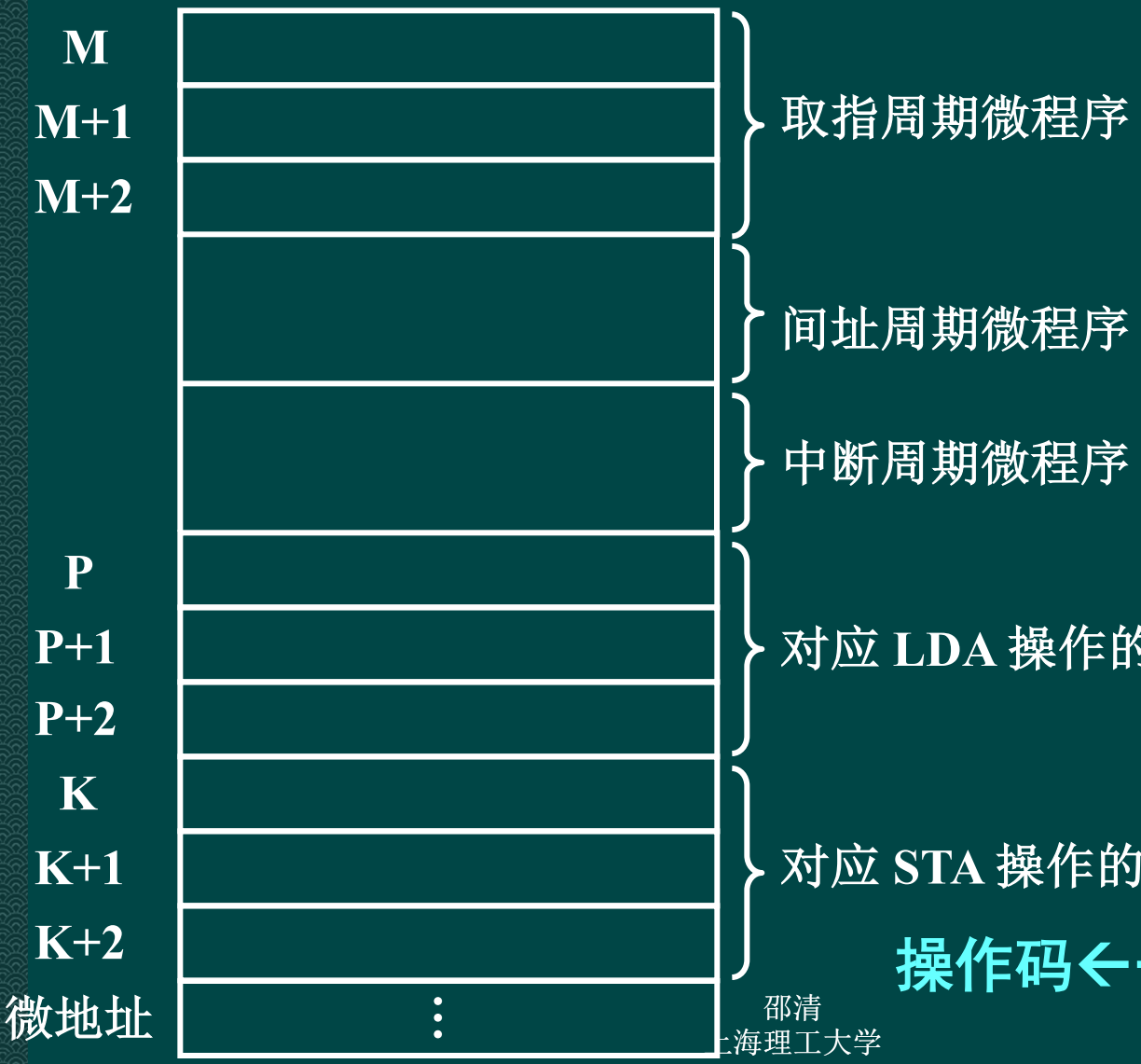

取指周期
间址周期
执行周期
中断周期
指令周期

- 微指令：同一个机器周期内发出的微操作命令的组合
- 微程序：一系列微指令的有序集合
- 一段微程序对应一条机器指令
- 控制存储器：用于存放微程序的专用存储器


程序
机器指令1
机器指令2
机器指令i
机器指令n
微指令1
微指令2
微指令i
微指令n
微程序

二、微程序控制单元框图及工作原理

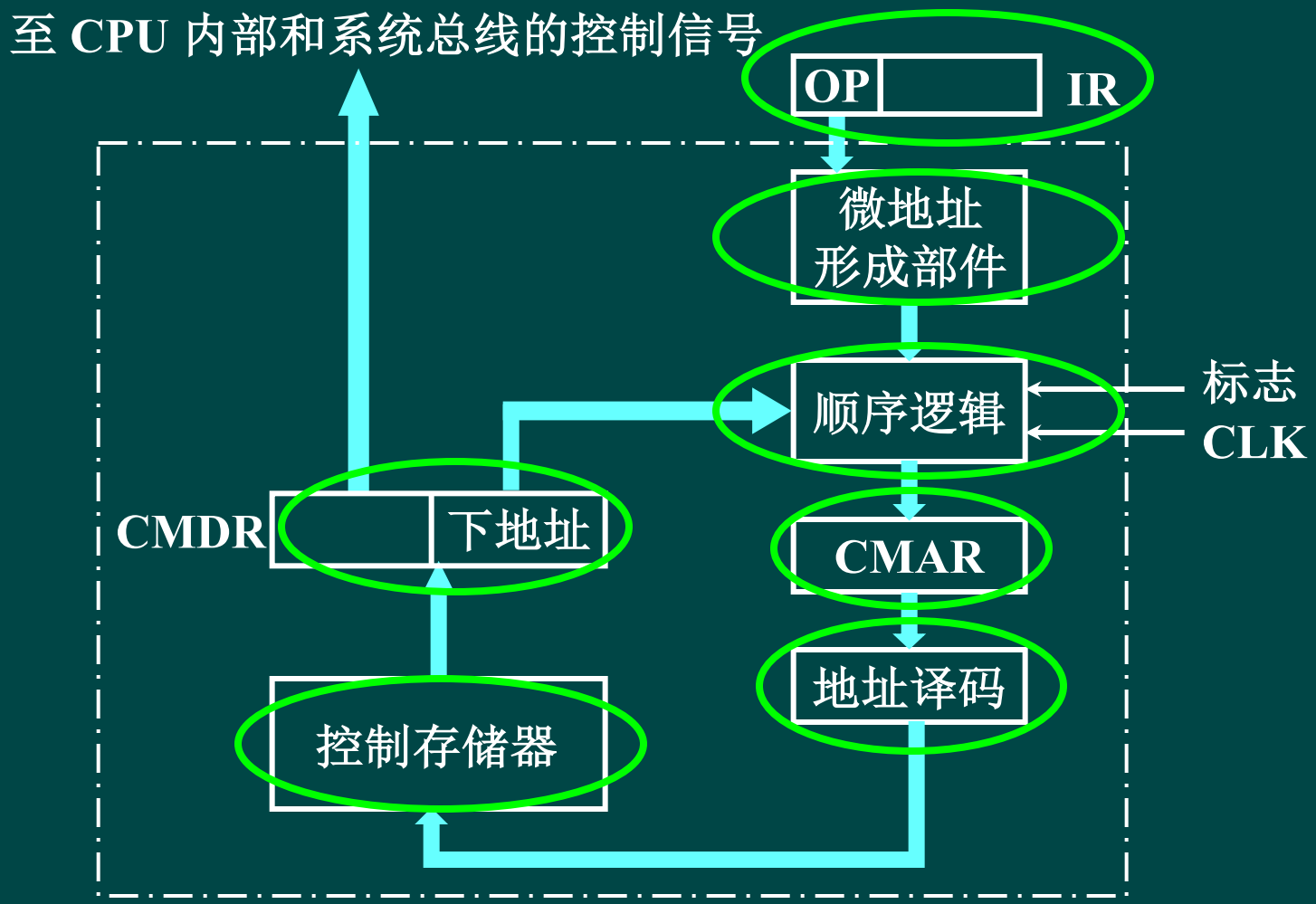
1. 机器指令对应的微程序



控制存储器

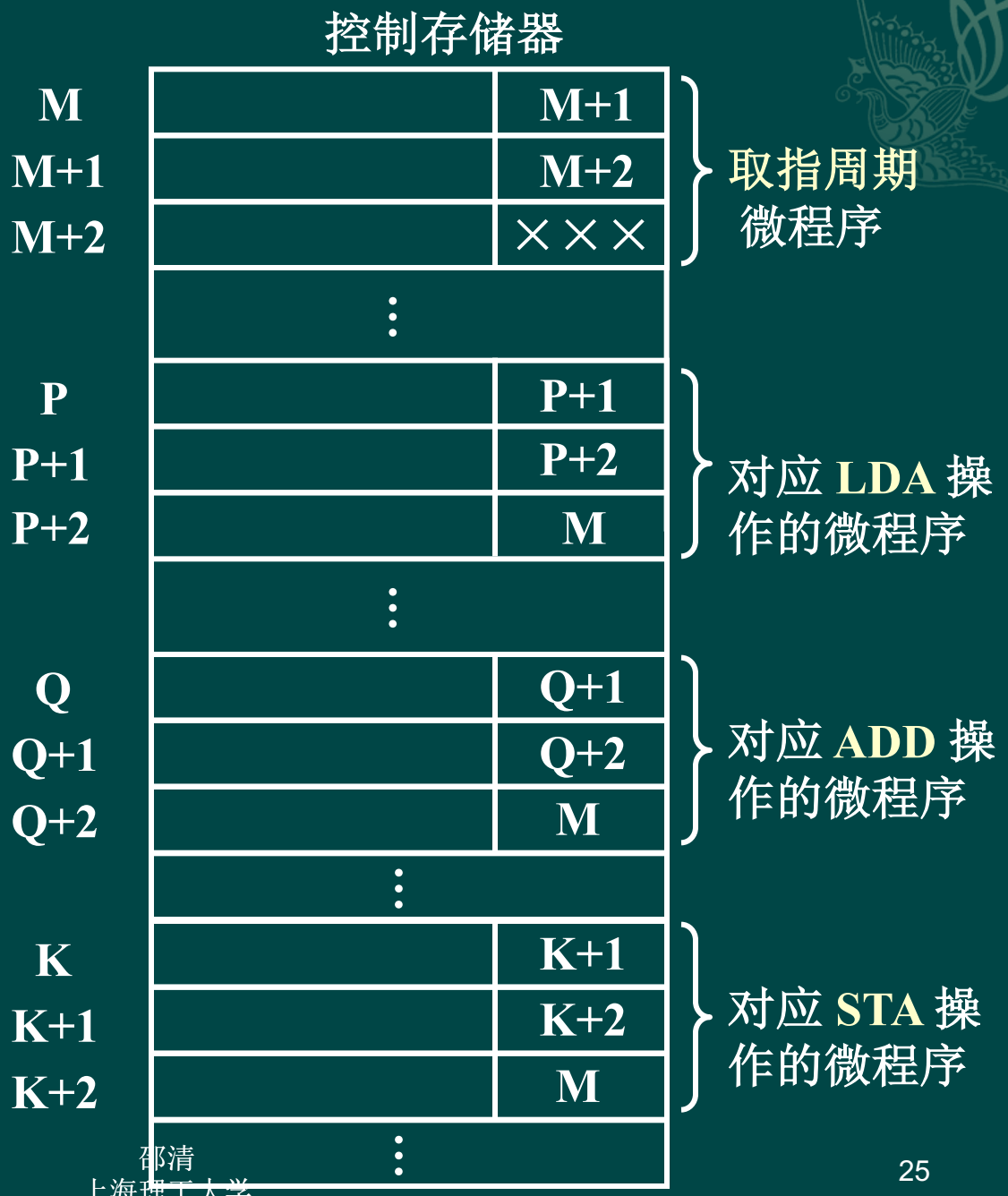
操作码←→ 微程序执行阶段

2. 微程序控制单元的基本框图



3. 工作原理


主存
用户程序
LDA X
ADD Y
STA Z
STP


控制存储器
M
M+1
M+2
P
P+1
P+2
Q
Q+1
Q+2
K
K+1
K+2
M+1
M+2
×××
P+1
P+2
M
Q+1
Q+2
M
K+1
K+2
M
取指周期微程序
对应 LDA 操作的微程序
对应 ADD 操作的微程序
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

Ad (IR) ⟶ MAR　　1 ⟶ R

由 CMDR 发命令

形成下条微指令地址　P + 1

Ad (CMDR) ⟶ CMAR

CM (CMAR) ⟶ CMDR

M (MAR) ⟶ MDR

由 CMDR 发命令

形成下条微指令地址 P + 2

Ad (CMDR) ⟶ CMAR

CM (CMAR) ⟶ CMDR

MDR ⟶ AC

由 CMDR 发命令

形成下条微指令地址 M

Ad (CMDR) ⟶ CMAR　　(M ⟶ CMAR)

(3) 微程序分析

微程序的执行过程：将微指令从CM中读出

关键

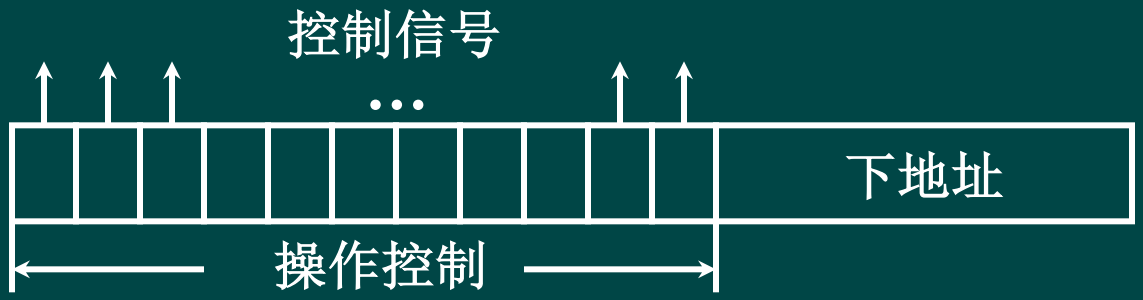
- 微指令的 操作控制字段如何形成微操作命令
- 微指令的 后续地址如何形成

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



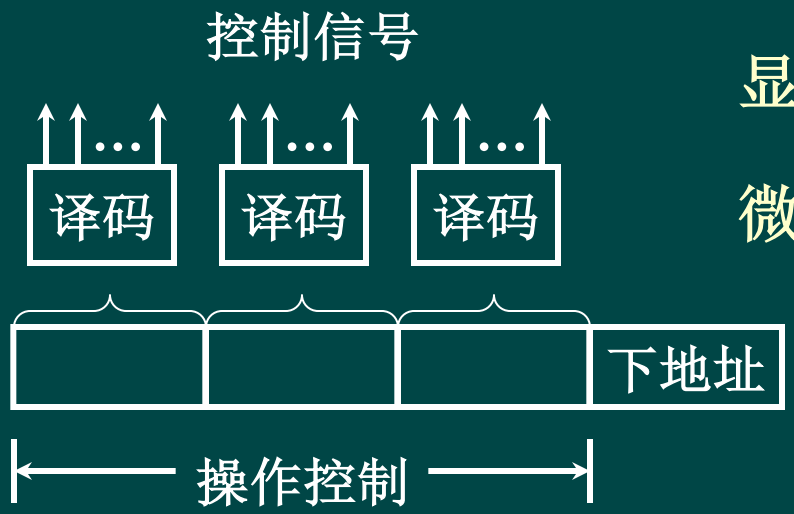
速度最快

某位为“1”表示该控制信号有效

2. 字段直接编码方式

将互斥的微操作合在一起作为一个 “段”，

每段经译码后发出控制信号

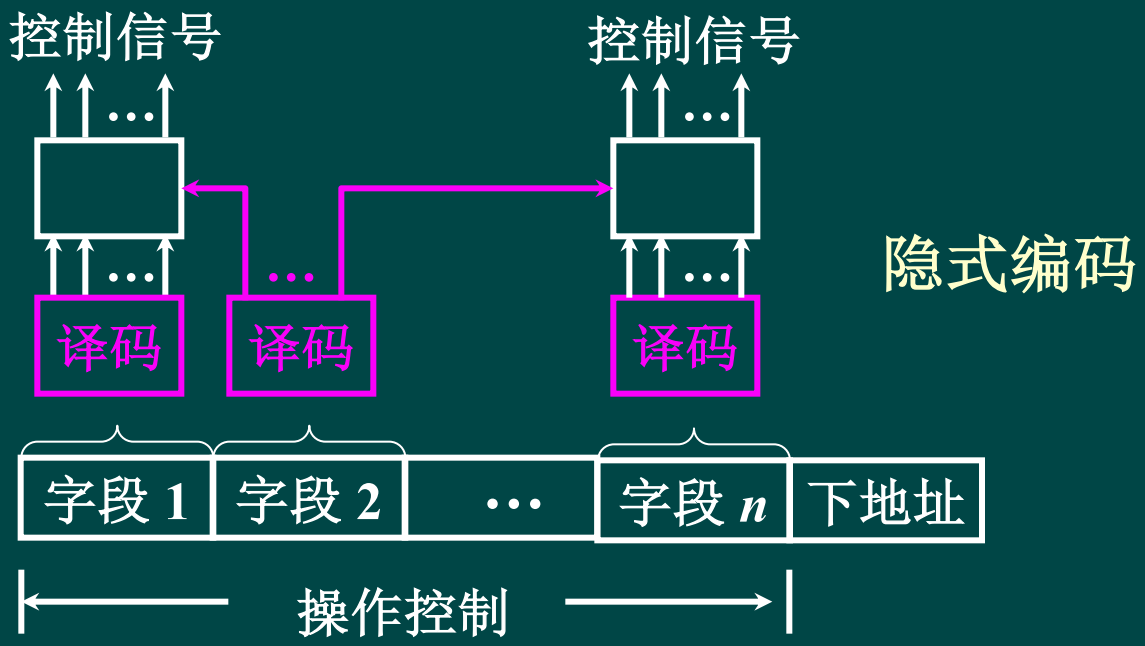


显式编码

微程序执行速度较慢

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

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

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

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

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

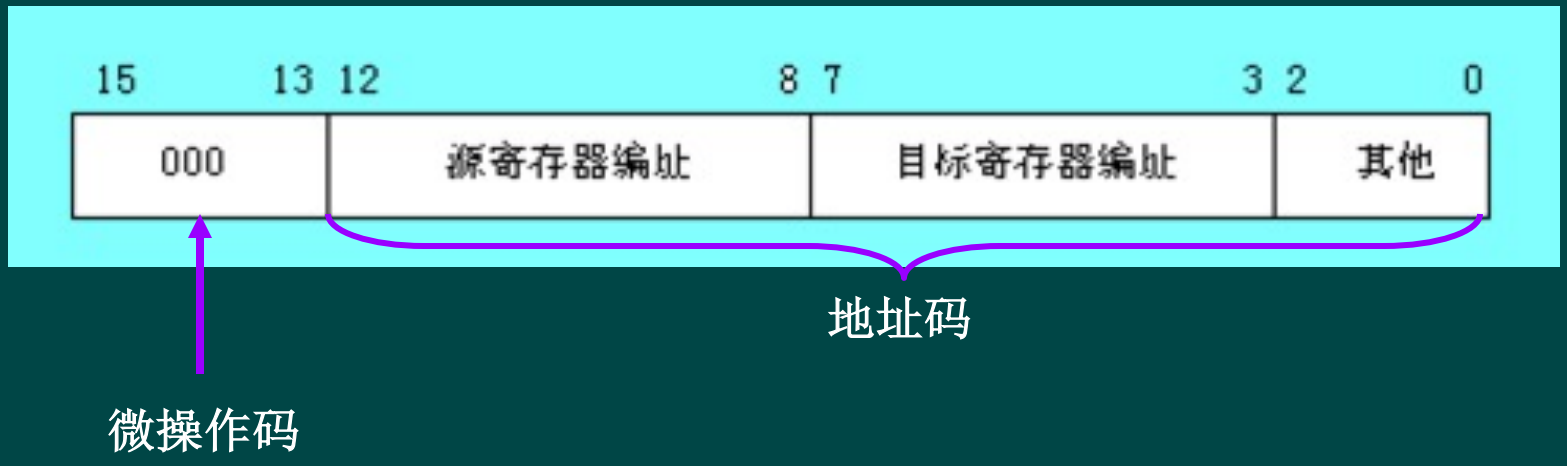
2. 垂直型微指令

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

垂直型微指令

- 采用短格式，一条微指令只能执行一、二个微命令；
- 微操作码字段确定微指令的功能；
- 地址码指定微操作数所在的寄存器地址或微指令转移地址，也可表示立即数或标志码等。



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 用较长的微指令结构换取较短的微程序结构

(4) 水平型微指令与机器指令 差别大