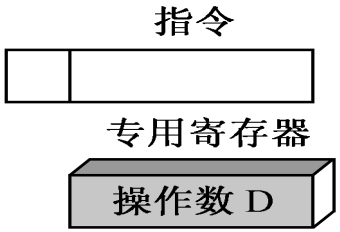
一、指令系统

1、指令格式

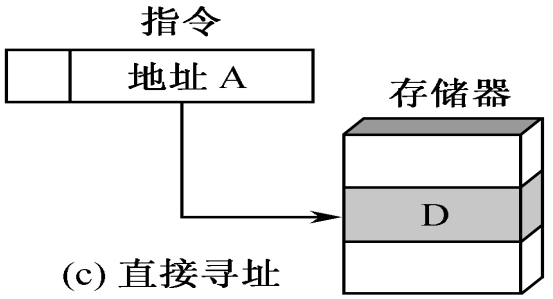
- **操作码OP**
  - ✓ 规整型：划出固定的一至二个字节代表操作码，其余部分是地址码，便于译码逻辑设计，易扩充。
  - ✓ 非规整型：操作码宽度随指令类型而变化，利用率高，但译码逻辑复杂，不易扩充。
- **地址码**
  - ✓ 四地址、三地址、二地址、单地址、零地址
- **指令长度：**一个指令字包含二进制代码的位数
  - ✓ 等长指令、非等长指令、多字长指令

2、寻址方式

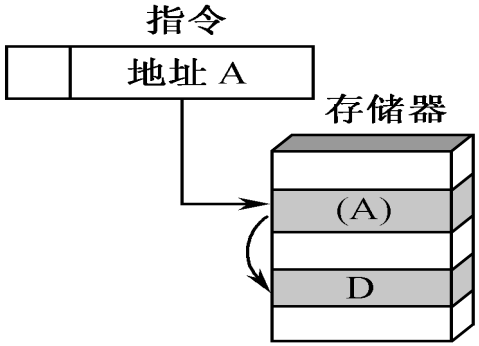


(a) 隐含寻址

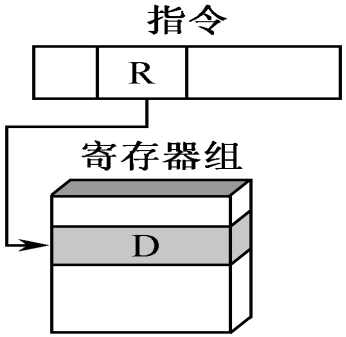
(b) 立即寻址



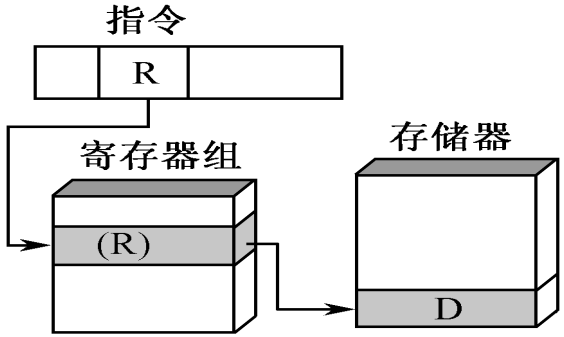
(c) 直接寻址



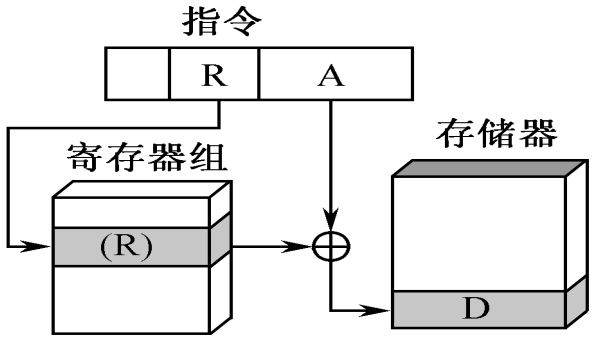
(d) 间接寻址



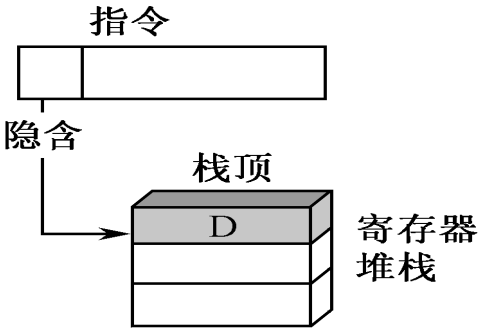
(e) 寄存器寻址



(f) 寄存器间接寻址



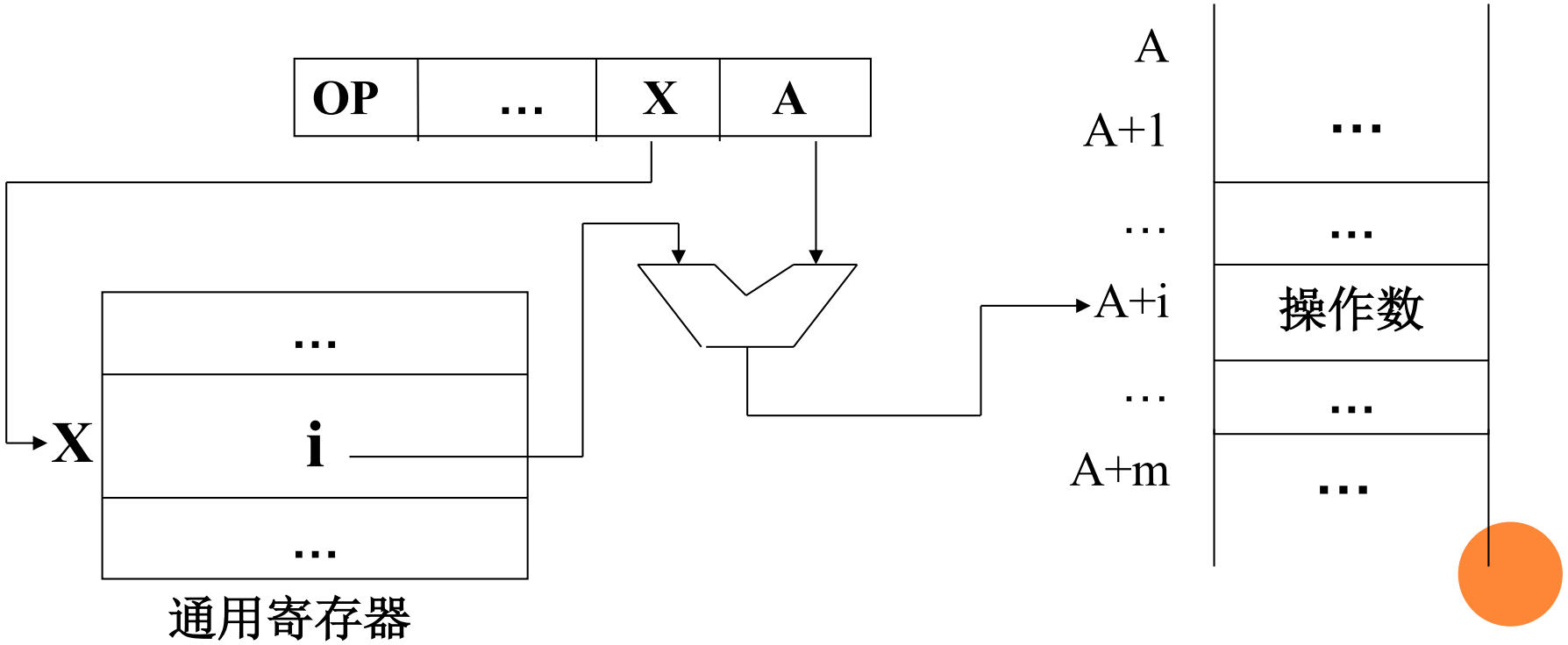
(g) 偏移寻址



(h) 堆栈寻址

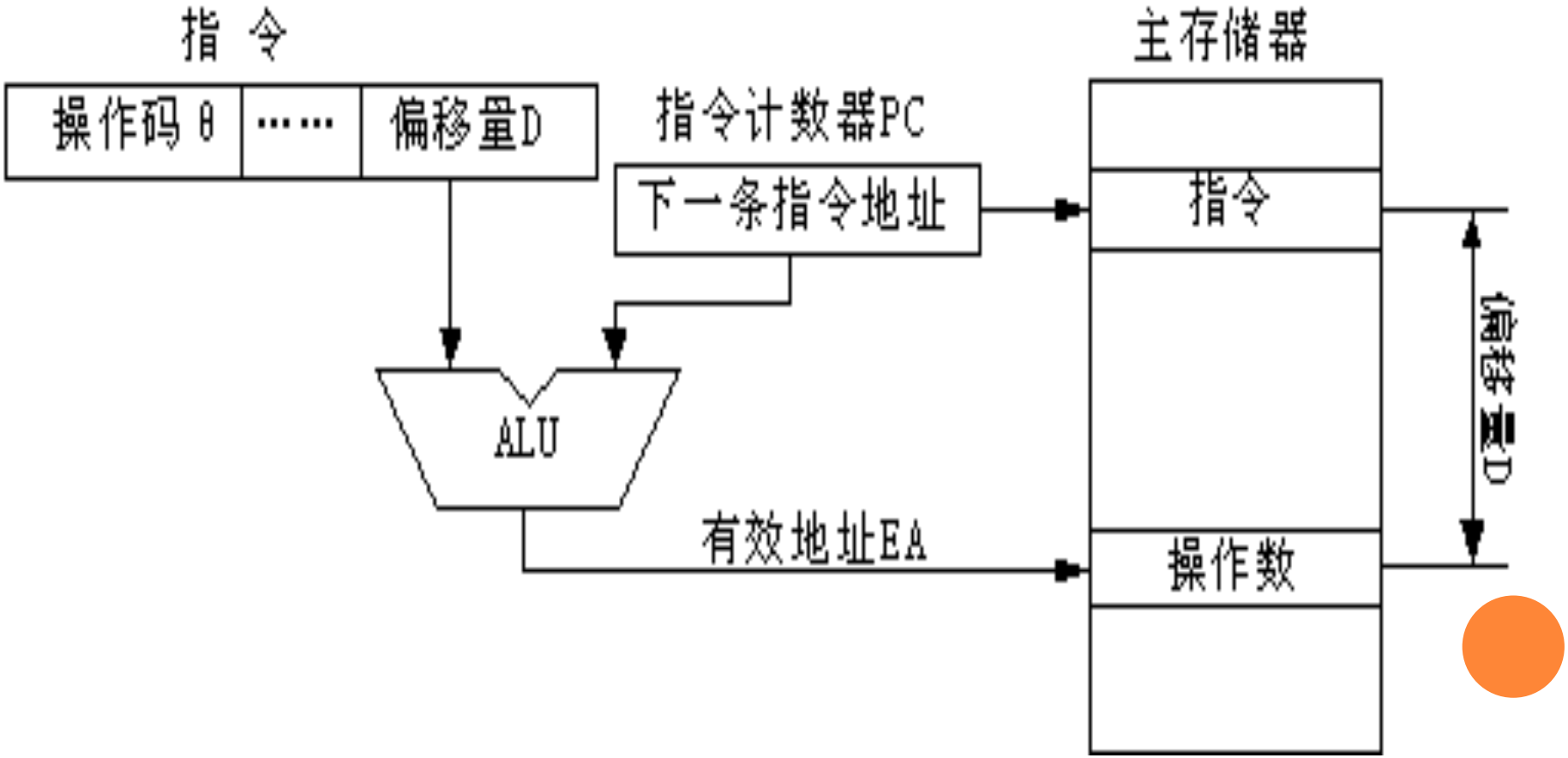
变址寻址

- 地址码给出基准地址A，变址寄存器Rb给出变址值，操作数的有效地址等于A加上变址值，即：EA = A + (Rx)。
- 典型应用：将Rx中的内容作为修改量。在需要频繁修改操作数地址时，无须修改指令，只要修改Rx值即可。广泛用于软堆栈、线性表、数组、字符串等需要成批数据处理的运算。



相对寻址

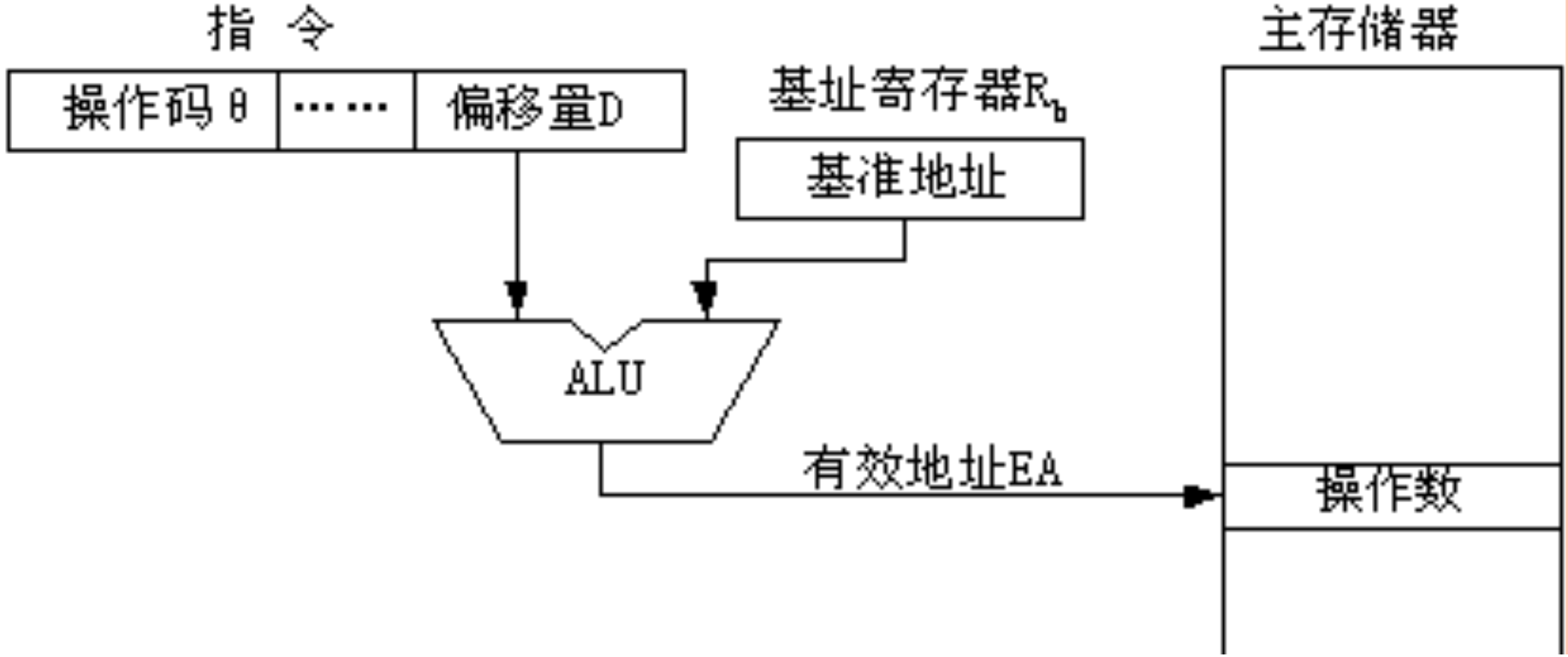
- 由程序计数器PC提供基准地址，而指令的地址码部分给出相对的位移量D，两者相加后作为操作数的有效地址，即：EA = (PC) + D。
- 应用：实现子程序浮动。编程时只要确定操作数与指令间的相对距离，可将子程序安排在主存任意位置，不会影响程序执行的正确性。



基址寻址

- 地址码给出偏移量D，基址寄存器Rb给出基准地址，操作数的有效地址等于基准地址A加上偏移量D，即：EA = (Rb) + D。
- 采用基址寻址时，指令中要给出基址寄存器的代码和偏移量；
- 在Pentium中，EBX(BX)为基址寄存器（基址指针）；

  例子：MOV AL，[BX+10H]　　BX为基址指针，10H为偏移量
- 应用：用于程序装入时，逻辑地址向物理地址的转换



二、CPU的功能和构成

1、功能

- 指令控制（程序的顺序控制）
- 操作控制（一条指令有若干操作信号实现）
- 时间控制（指令各个操作实施时间的定时）
- 数据加工（算术运算和逻辑运算）

2、CPU组成：运算器 + 控制器+总线 + Cache

3、控制器组成

- 寄存器组：通用寄存器，专用寄存器
- 操作控制部件：指令译码器，微操作控制器
- 时序部件

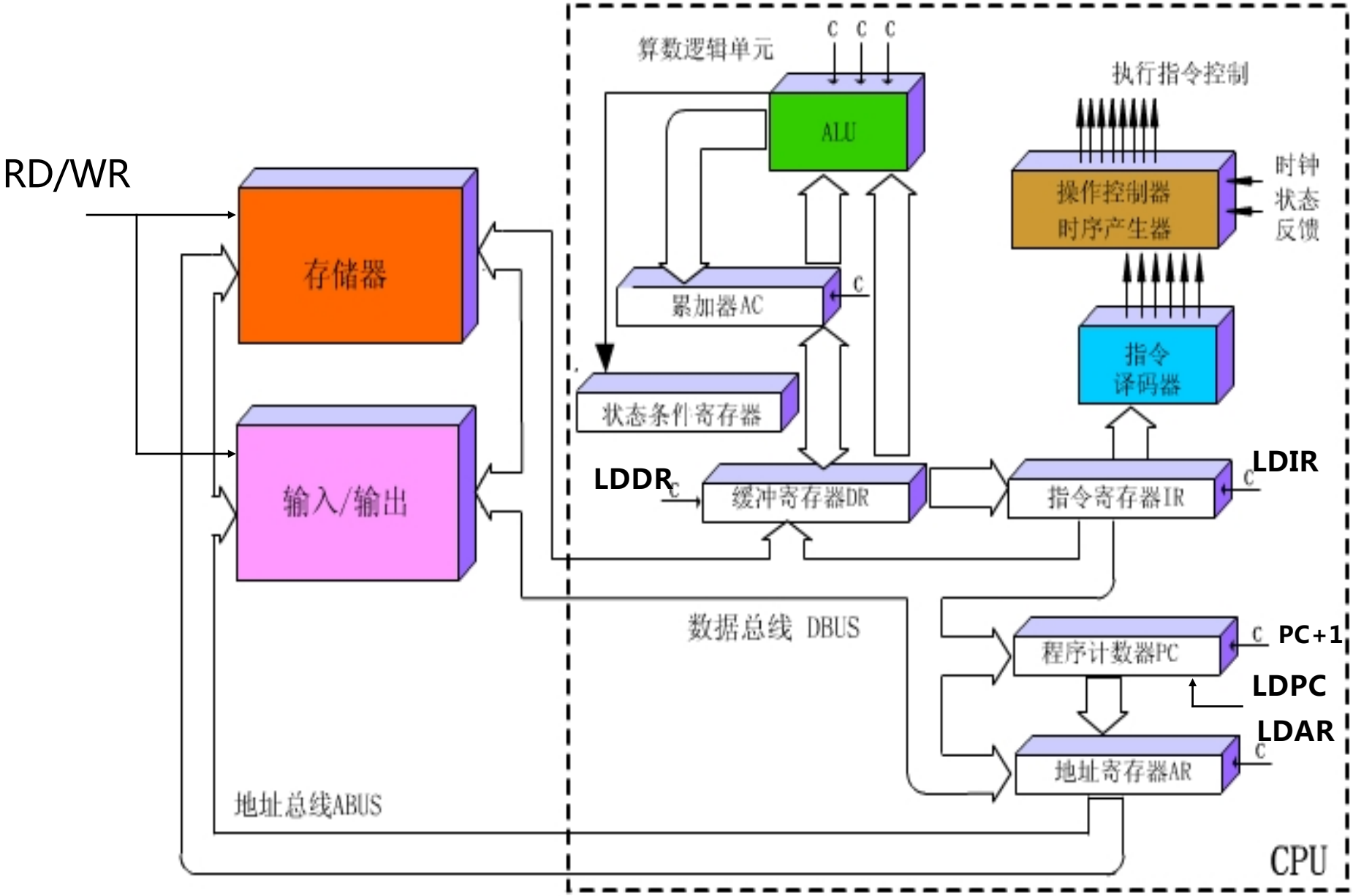
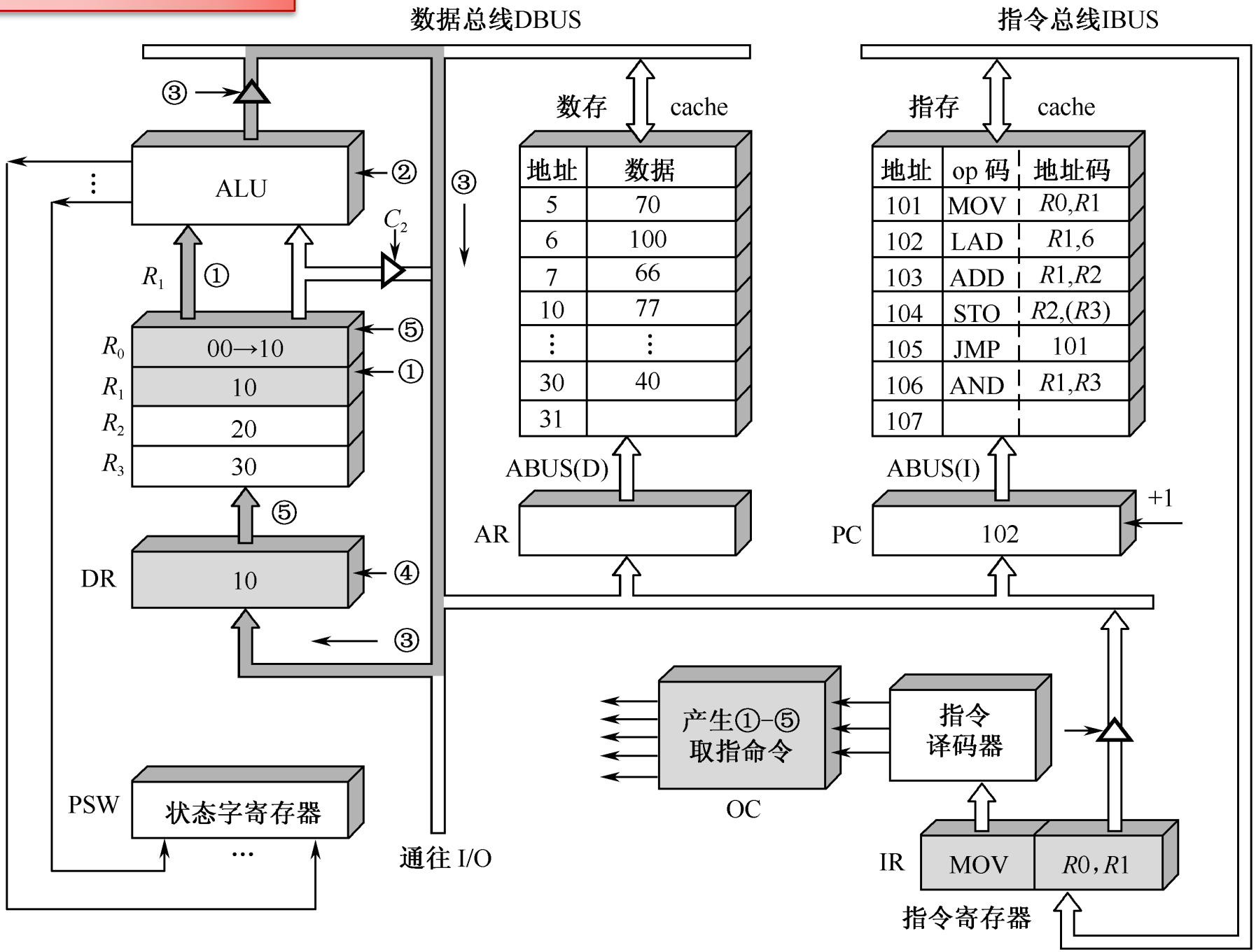

RD/WR
存储器
输入/输出
算数逻辑单元
C C C
ALU
累加器AC
C
状态条件寄存器
LDDR
C
缓冲寄存器DR
执行指令控制
操作控制器
时序产生器
时钟
状态
反馈
指令
译码器
指令寄存器IR
C
LDIR
数据总线 DBUS
程序计数器PC
C
PC+1
LDPC
LDAR
地址寄存器AR
C
地址总线ABUS
CPU

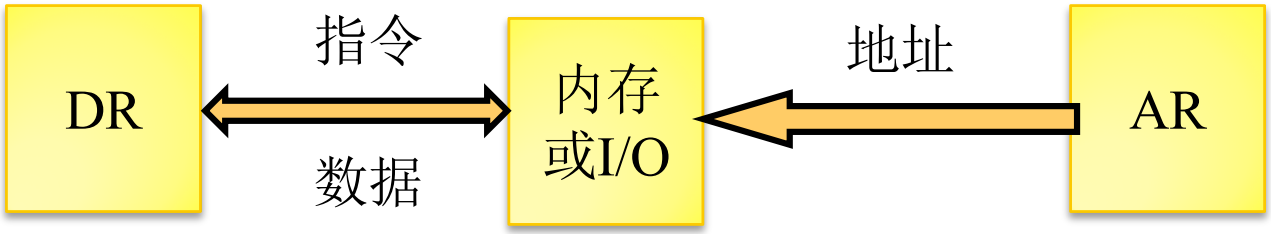
图5.1 C P U 的结构

CPU模型


数据总线DBUS
指令总线IBUS
数存
cache
指存
cache
地址
数据
5
70
6
100
7
66
10
77
30
40
31
地址
op 码
地址码
101
MOV
R0,R1
102
LAD
R1,6
103
ADD
R1,R2
104
STO
R2,(R3)
105
JMP
101
106
AND
R1,R3
107
ALU
C_2
R_1
①
②
③
④
⑤
R_0
00→10
R_1
10
R_2
20
R_3
30
DR
10
ABUS(D)
AR
ABUS(I)
PC
102
+1
PSW
状态字寄存器
通往 I/O
产生①-⑤
取指命令
OC
指令
译码器
IR
MOV
R0，R1
指令寄存器

4、主要部件功能

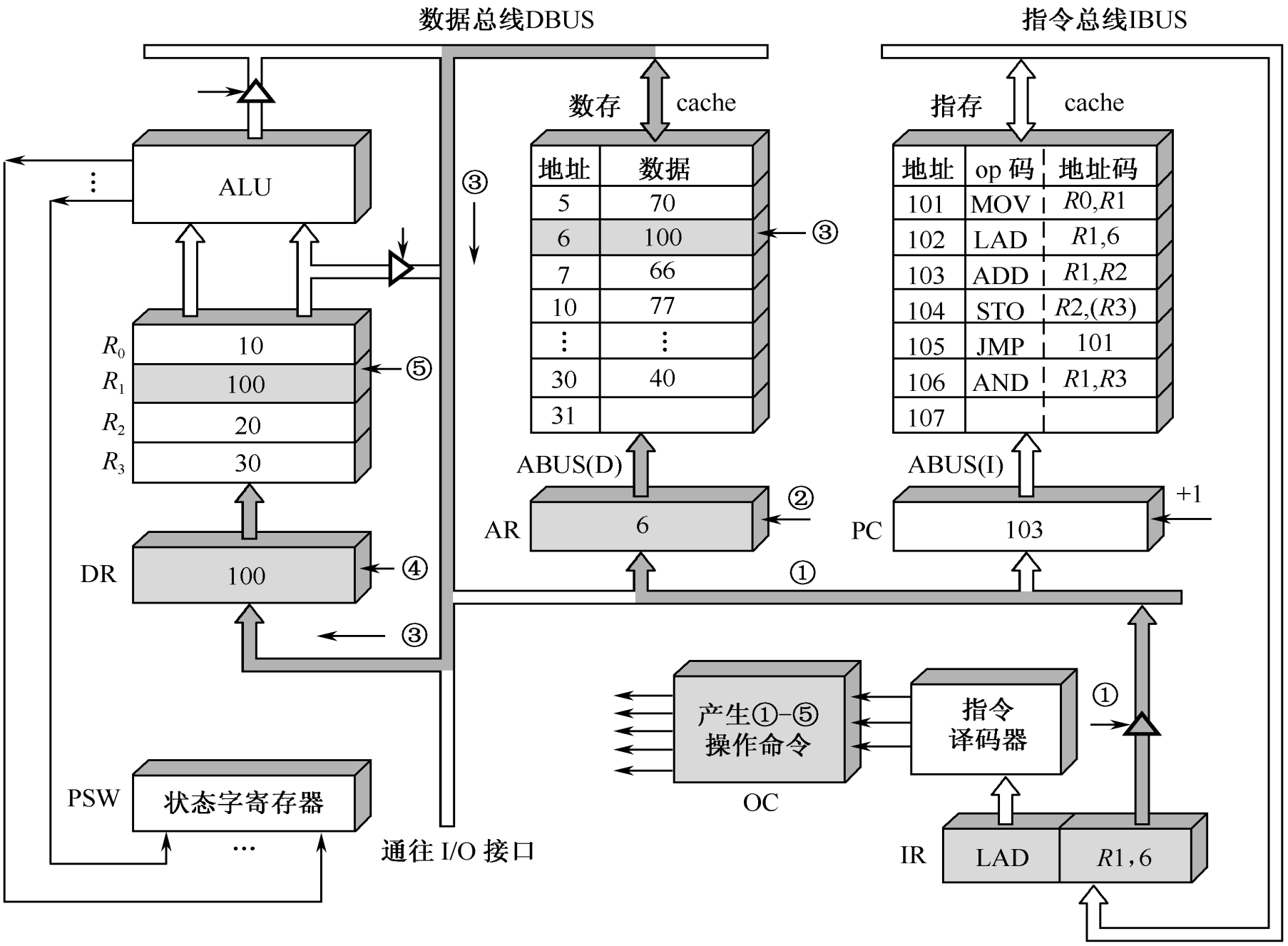
- 程序计数器PC (Programming Counter)
- 指令寄存器IR (Instruction Register)
- 指令译码器ID (Instruction Decoder)
- 累计寄存器AC和通用寄存器组
- PSW——状态条件寄存器
- AR——地址寄存器
- DR——缓冲寄存器
-

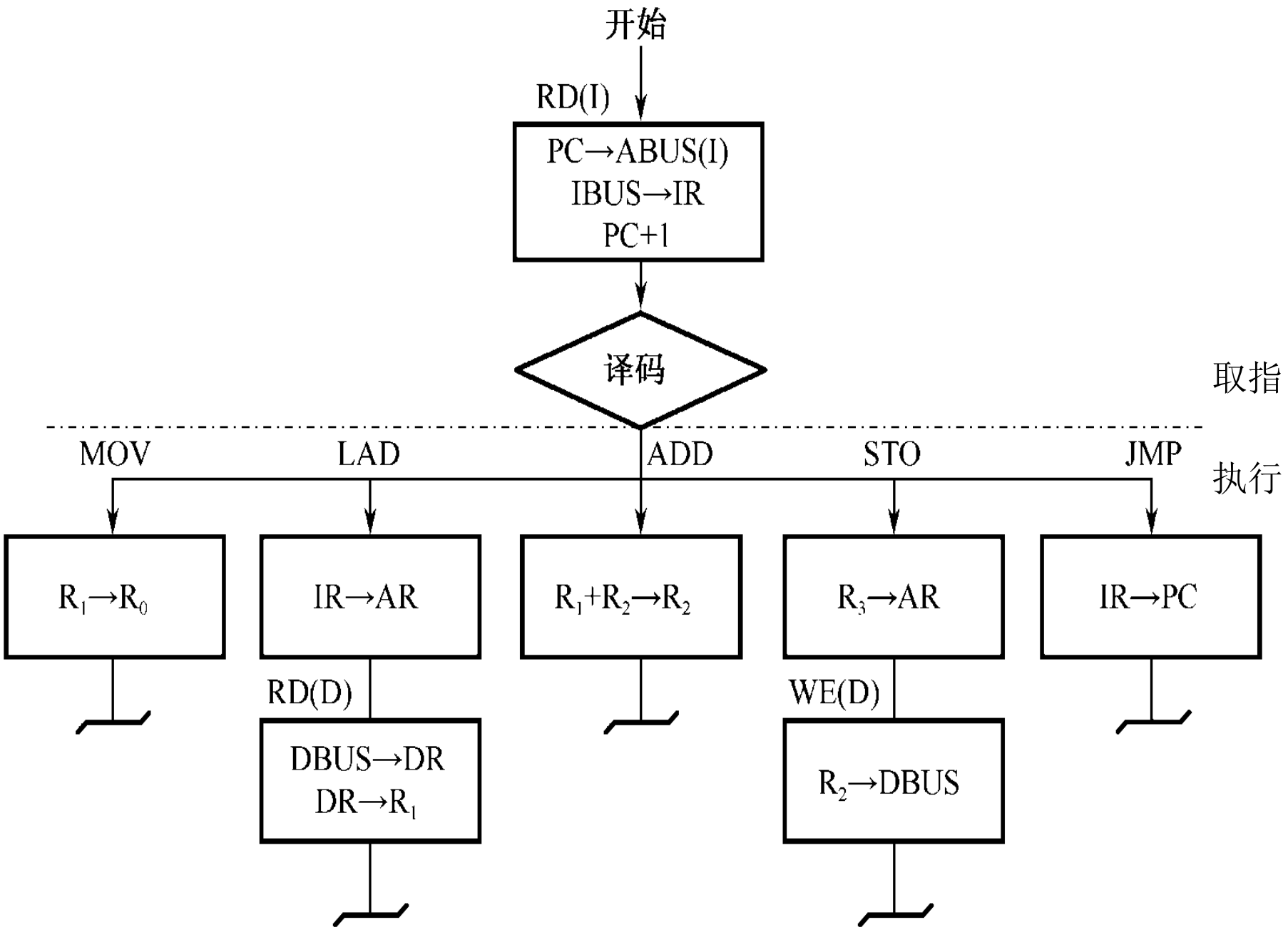


AR、DR：中转站、补偿速度差别

三、典型指令的执行流程

- 取指流程
- 执行流程
  - ✓ MOV
  - ✓ LAD
  - ✓ ADD
  - ✓ STO
  - ✓ JMP


数据总线DBUS
指令总线IBUS
数存
cache
指存
cache
地址
数据
地址
op 码
地址码
5
70
6
100
7
66
10
77
30
40
31
101
MOV
R0,R1
102
LAD
R1,6
103
ADD
R1,R2
104
STO
R2,(R3)
105
JMP
101
106
AND
R1,R3
107
ALU
R0
10
R1
100
R2
20
R3
30
DR
100
ABUS(D)
AR
6
ABUS(I)
PC
103
+1
①
②
③
④
⑤
PSW
状态字寄存器
通往 I/O 接口
产生①-⑤
操作命令
OC
指令
译码器
IR
LAD
R1,6



测试公操作：停机测试，掉电测试，通道请求，中断请求

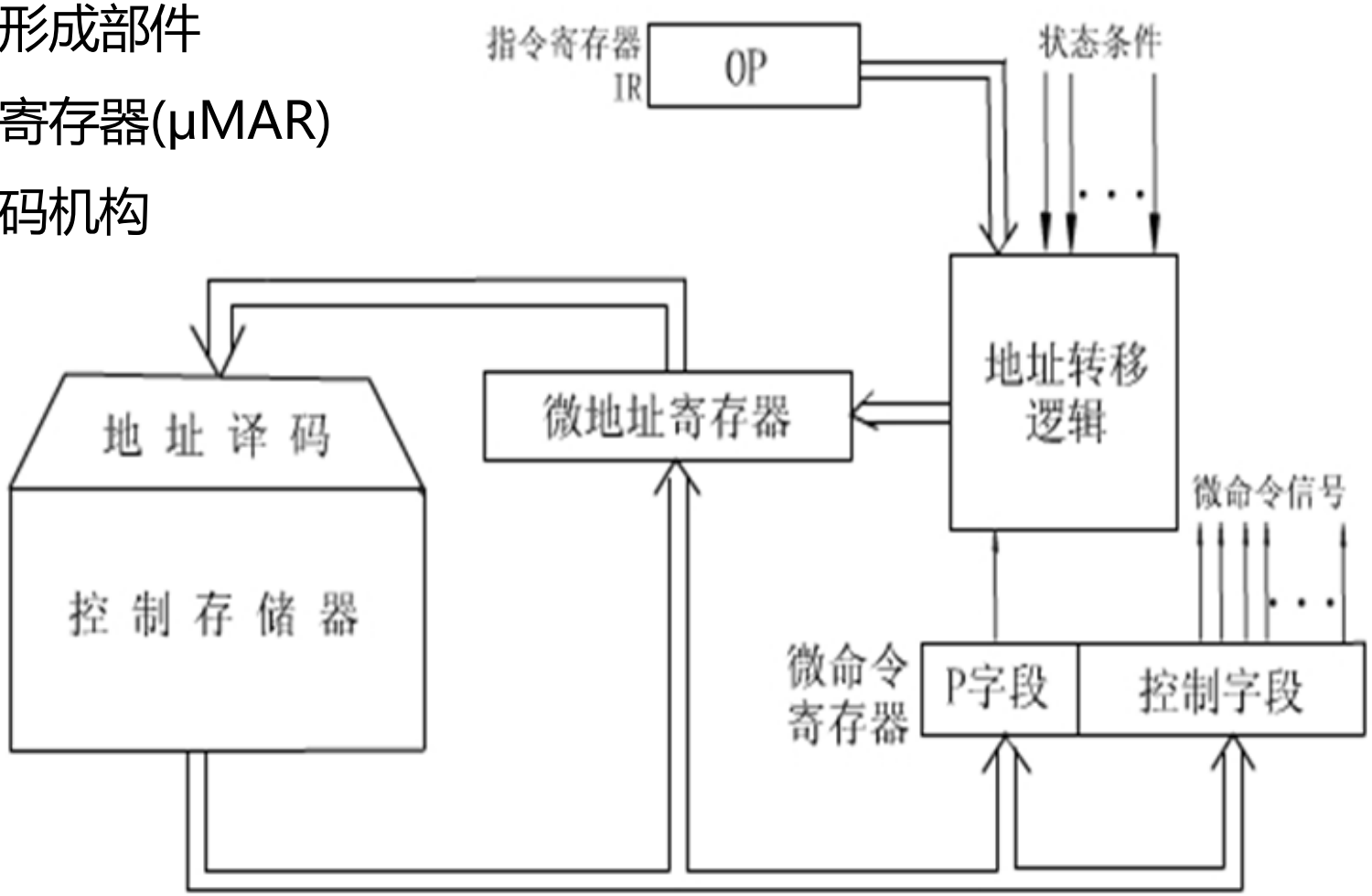
四、微程序控制器

1、基本概念

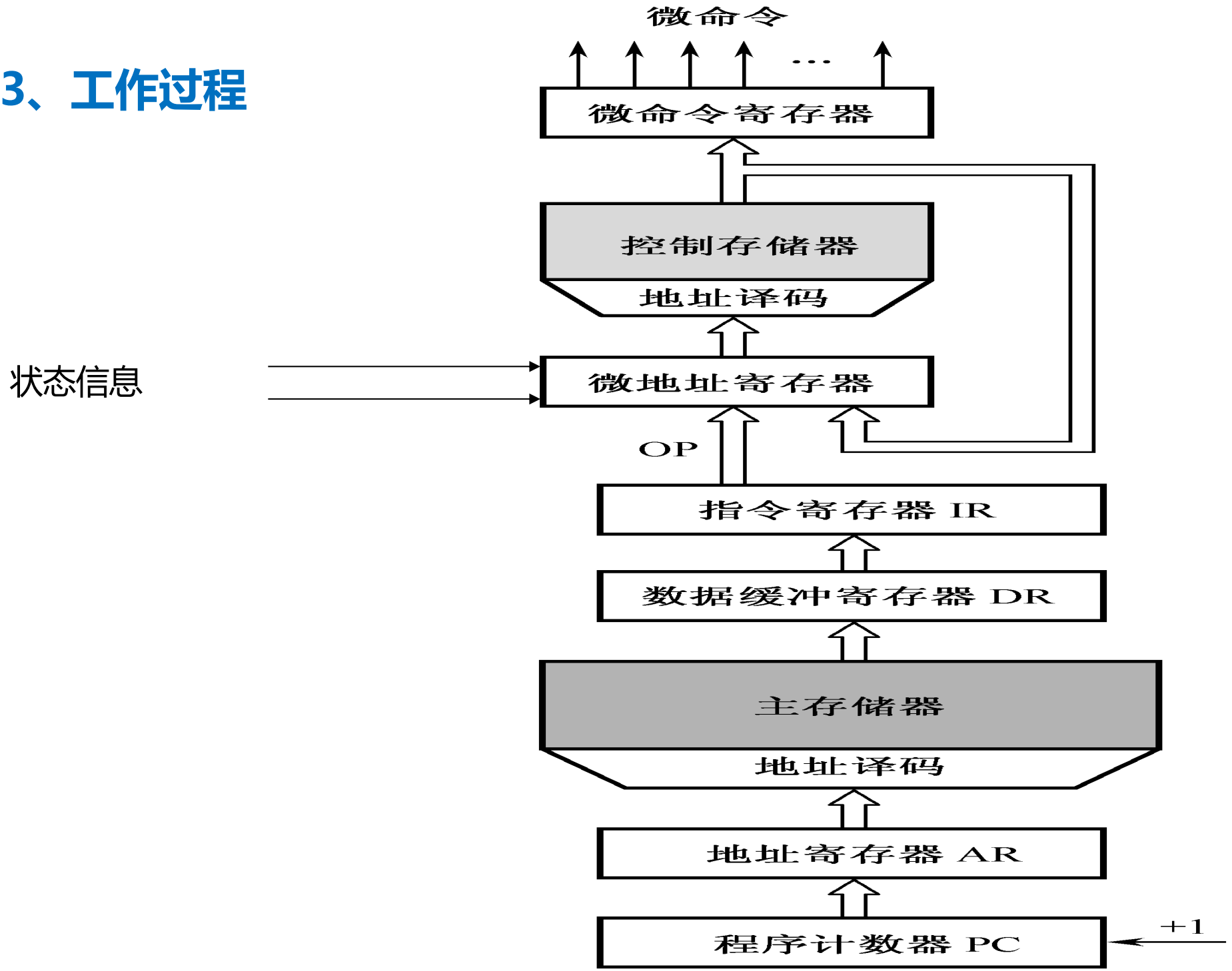
- 微命令
- 微操作（互斥微操作、相容微操作）
- 微指令（**垂直型微指令和水平型微指令**）
- 微指令周期
- 微程序

2、微程序控制器的结构

- ✓ 控制存储器(μCM)：
- ✓ 微指令寄存器(μIR)
- ✓ 微地址形成部件
- ✓ 微地址寄存器(μMAR)
- ✓ 地址译码机构



3、工作过程


微命令
…
微命令寄存器
控制存储器
地址译码
状态信息
微地址寄存器
OP
指令寄存器 IR
数据缓冲寄存器 DR
主存储器
地址译码
地址寄存器 AR
程序计数器 PC
+1

取出机器指令的公共操作

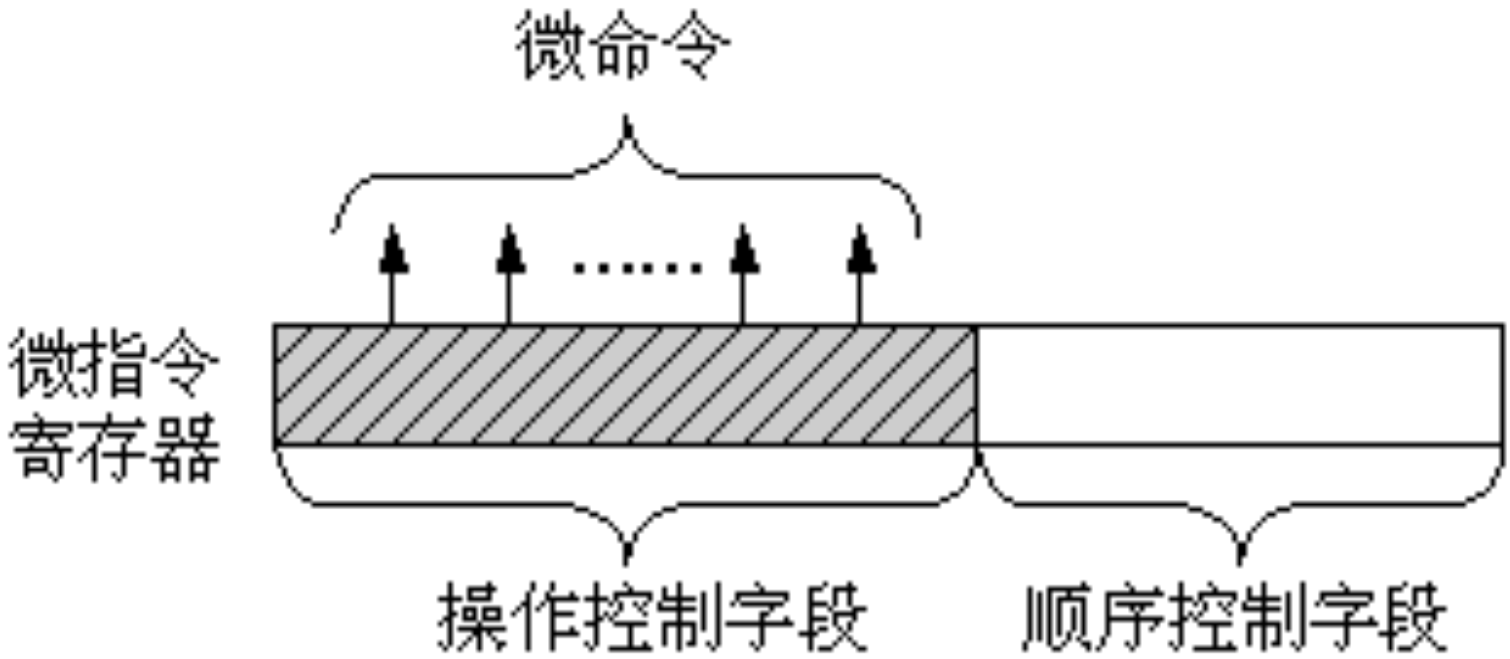
- 由一段取指微程序完成，程序开始运行时，自动将取指微程序的入口微地址送μMAR，并从μCM中读出相应微指令送入μIR。
- 取指微程序的入口地址一般为μCM的0号单元，
- 微指令的操作控制字段产生有关的微命令，用来控制实现取机器指令的公共操作。当取指微程序执行完后，从主存中取出的机器指令就已存人指令寄存器IR中了。

执行机器指令的操作流程

- 由机器指令的操作码字段通过微地址形成部件产生出该机器指令所对应的微程序的入口地址，并送入μMA
- 从μCM中逐条取出对应的微指令并执行之，每条微指令都能自动产生下一条微指令的地址。
- 一条机器指令对应微程序的最后一条微指令执行完毕后，其下一条微指令地址又回到取指微程序的人口地址，从而继续第(1)步，完成取下条机器指令的公共操作。

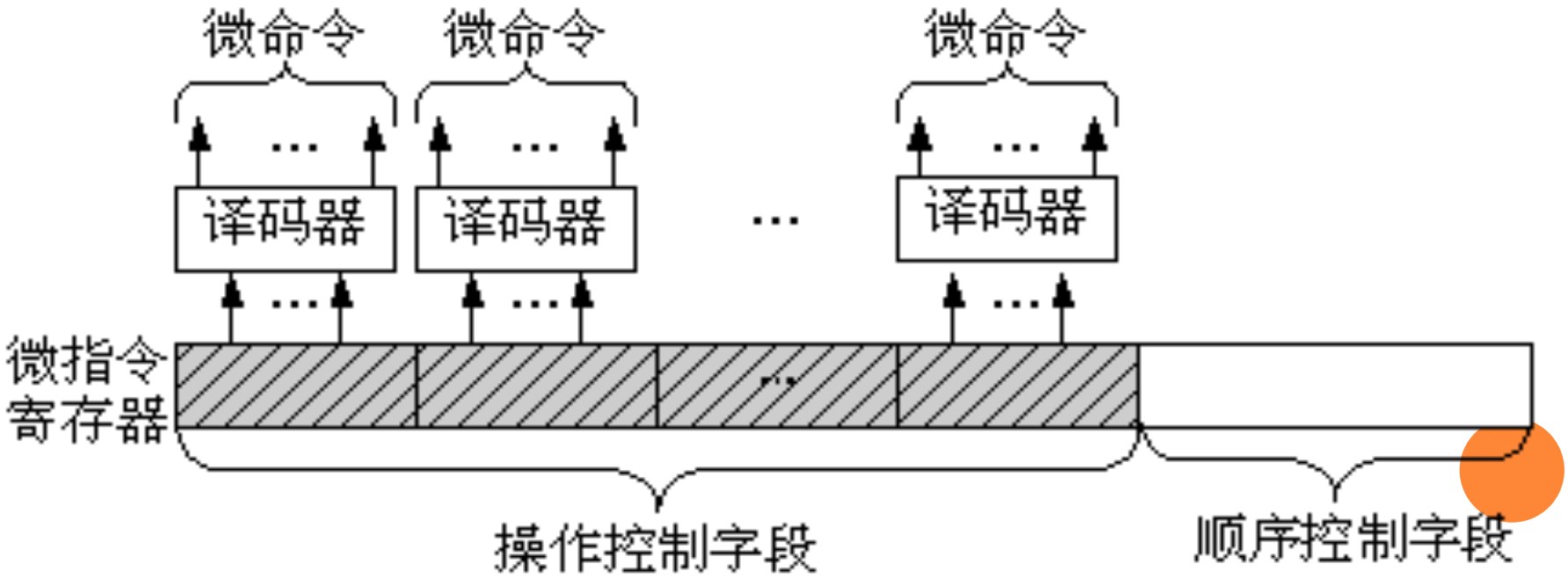
4、微命令编码

- **直接表示法**：每一位分别代表一个微命令



- **编码表示法：**将操作控制字段分为若干个小段，每段内采用最短编码法，段与段之间采用直接控制法。

- ✓ **互斥性微命令在同一字段，兼容性微命令在不同段内**
- ✓ **应与数据通路结构相适应。**
- ✓ **每个字段包含的信息位不能太多**
- ✓ **每个字段要留出一个状态，表示不发出任何微命令。**

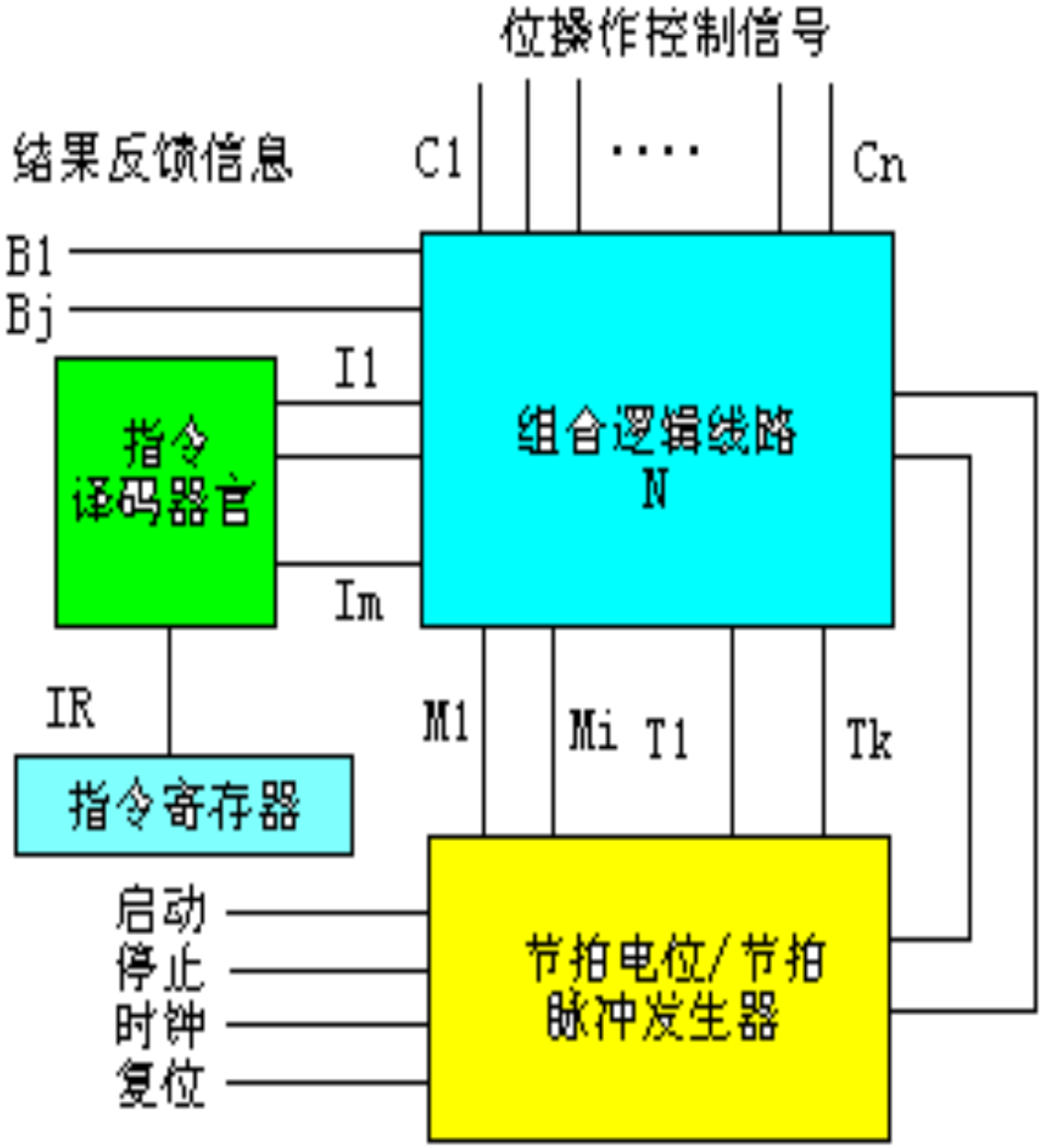


4、微指令地址的形成

- 入口地址：每条机器指令对应一段微程序，当公用的取指微程序从主存中取出机器指令之后，由机器指令的操作码字段指出各段微程序的入口地址，这是一种多分支(或多路转移)的情况。
- 机器指令的操作码转换成初始微地址的方式主要有两种。
  - 计数器的方式
  - 多路转移的方式

五、硬布线控制器（组合逻辑控制器）

1、逻辑结构



2、设计步骤

- **画出指令流程图**
- **列出微操作时间表**
  - ✓ 将指令流程图中的微操作合理地安排到各个CPU周期和节拍中；
  - ✓ 微操作时间表形象地表明：什么时间、什么条件、完成什么微操作。
- **微操作信号的综合**
  - ✓ 把执行同一微操作的所有条件(指令、 节拍电位、节拍脉冲）进行类组合，列出各微操作产生的逻辑表达式，并进行简化。
- **实现电路**
  - ✓ 根据逻辑表达式组，用逻辑门电路的组合进行实现；或用PLA或其他逻辑电路实现。

微操作时间表

	MOV	LAD	ADD	STO	JMP
M1	RD(I) PC+1	RD(I) PC+1	RD(I) PC+1	RD(I) PC+1	RD(I) PC+1
T1	…	…	…	…	…
T2	…	…	…	…	…
T3	…	…	…	…	…
T4	LDIR LDPC	LDIR LDPC	LDIR LDPC	LDIR LDPC	LDIR LDPC
M2					
T1	…	…	…	…	…
T2	…	…	…	…	…
T3	LDDR	…	LDDR	…	…
T4	…	LDAR	LDR2	LDAR	LDPC
M3	…	RD(D)			
T1	…	…	…	…	…
T2	…	…	…	…	…
T3	…	LDDR	…	WE(D)	…
T4	…	…	…	…	…

进行微操作信号的综合

RD(I) = M1

RD(D) = M3 · LAD

WE(D) = M3 · T3 · **STO**

LDPC = M1 · T4 + M2 · T4 · JMP

LDIR = M1 · T4

LDAR = M2 · T4 · (LAD+STO) = M2 · T4 · LAD + M2 · T4 · STO

LDDR = M2 · T3 · (MOV+ADD)+M3 · T3 · LAD

= M2 · T3 · MOV + M2 · T3 · ADD + M3 · T3 · LAD

PC+1 = M1

LDR2 = M2 · T4 · ADD

- M1、M2、M3：节拍电位信号
- T3、T4：节拍脉冲信号
- ADD、STA、JMP：指令译码器的输出信号