计算机组成原理

2017年修订

西南交通大学信息科学与技术学院
唐慧佳　hjtang@home.swjtu.edu.cn

第6章　中央处理器

第6章　中央处理器

本章要点：

1. CPU的功能和组成
2. 时序控制方式、时序系统及其作用
3. 指令执行过程的微操作序列分解
4. 组合逻辑控制器的设计思路和方法
5. 微程序控制原理、微指令的操作控制字段和顺序控制字段的常见的编码方法
6. 流水线技术

第6章　中央处理器

CPU：中央处理器（运算器 + 控制器）

现代CPU芯片内一般包含：

- 运算器
- 控制器
- Cache（高速缓存）
- MMU（内存管理单元）

……

§6.1 CPU的功能

CPU的功能：

指令控制：产生下一条指令在内存中的地址、取指令；

操作控制：把指令分解成一系列的微操作控制信号，控制各部件完成指令所要求的动作；

时序控制：对指令的各个微操作实施时间的定时，使它们能够按先后顺序来执行；

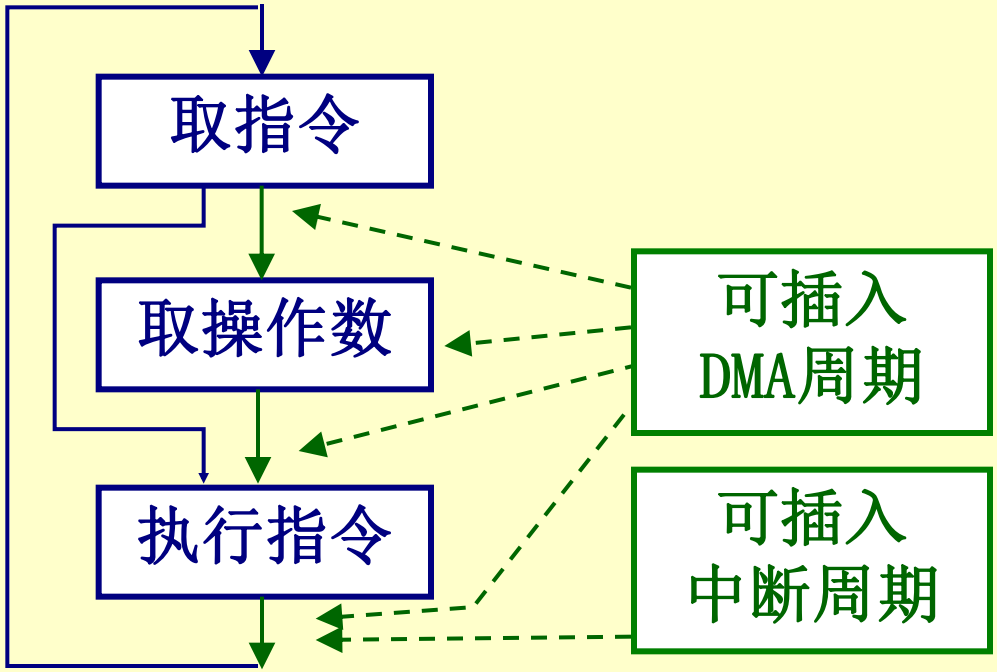
数据加工：算术运算、逻辑运算；

中断处理：处理异常情况和特殊请求。

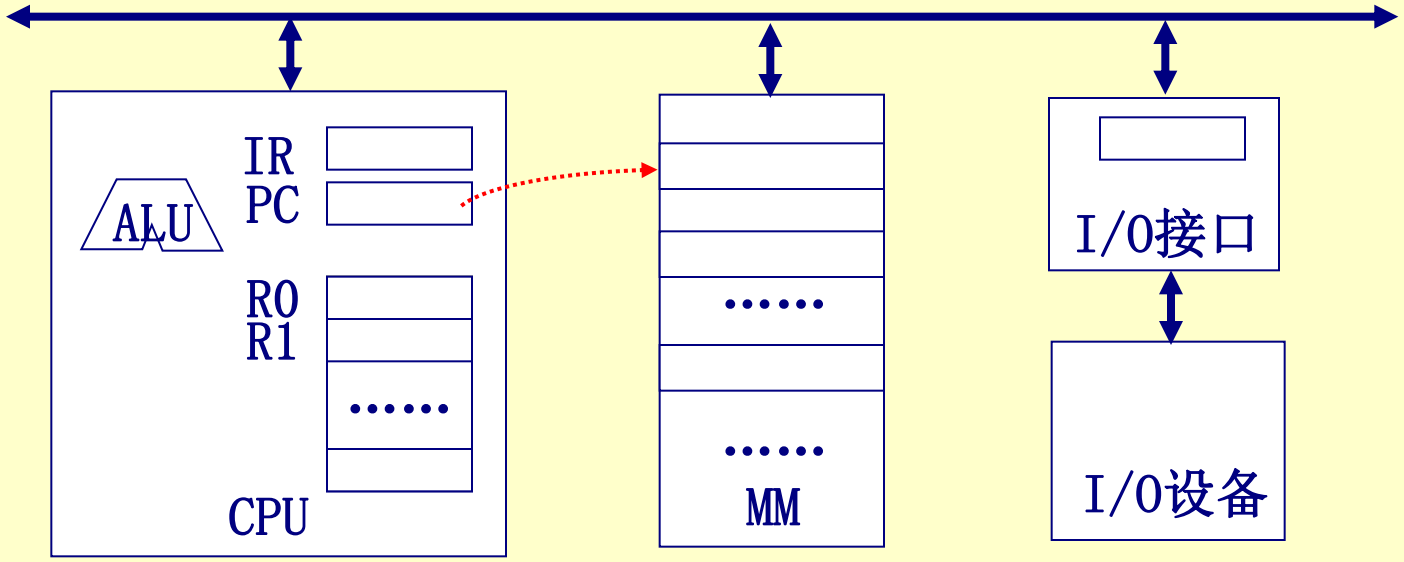
其它：　　如总线管理等。

回顾：计算机执行程序的过程

计算机执行程序的过程，即逐条执行指令的过程。



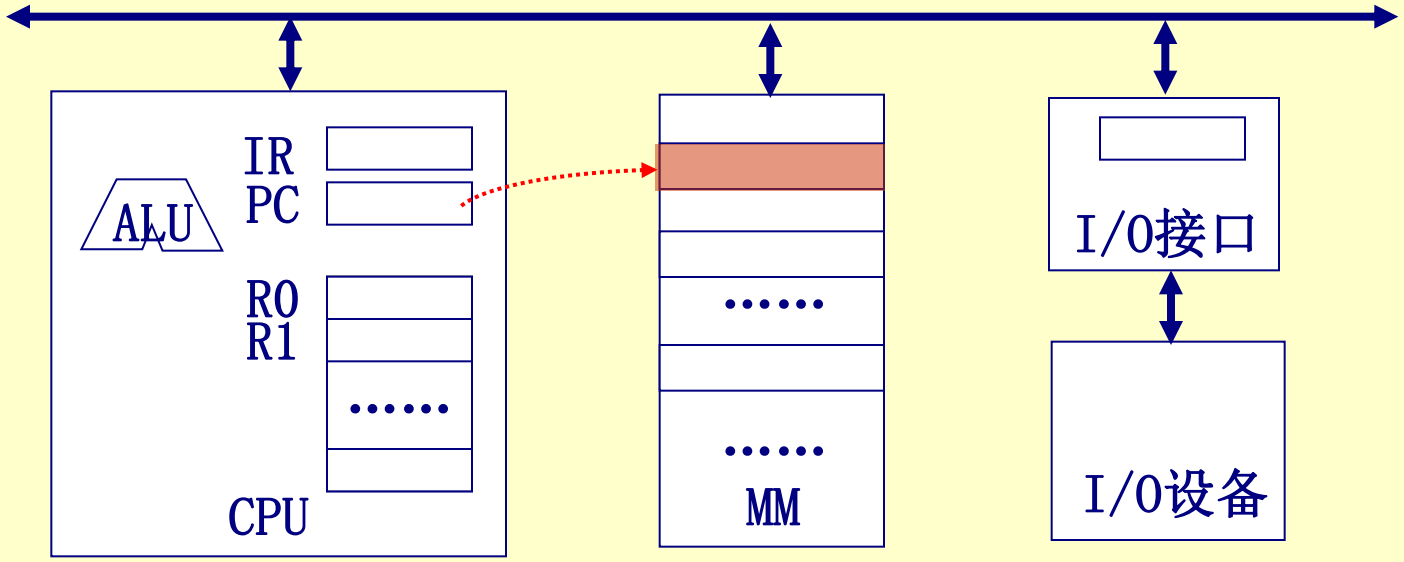
回顾：计算机执行程序的过程



以取指令过程为例，其具体操作包含以下内容：

（1）控制器以PC中的二进制编码为内存地址，通过系统总线（的地址线）向MM发送地址信息，并且通过系统总线（的控制线）向MM发“读内存”命令；

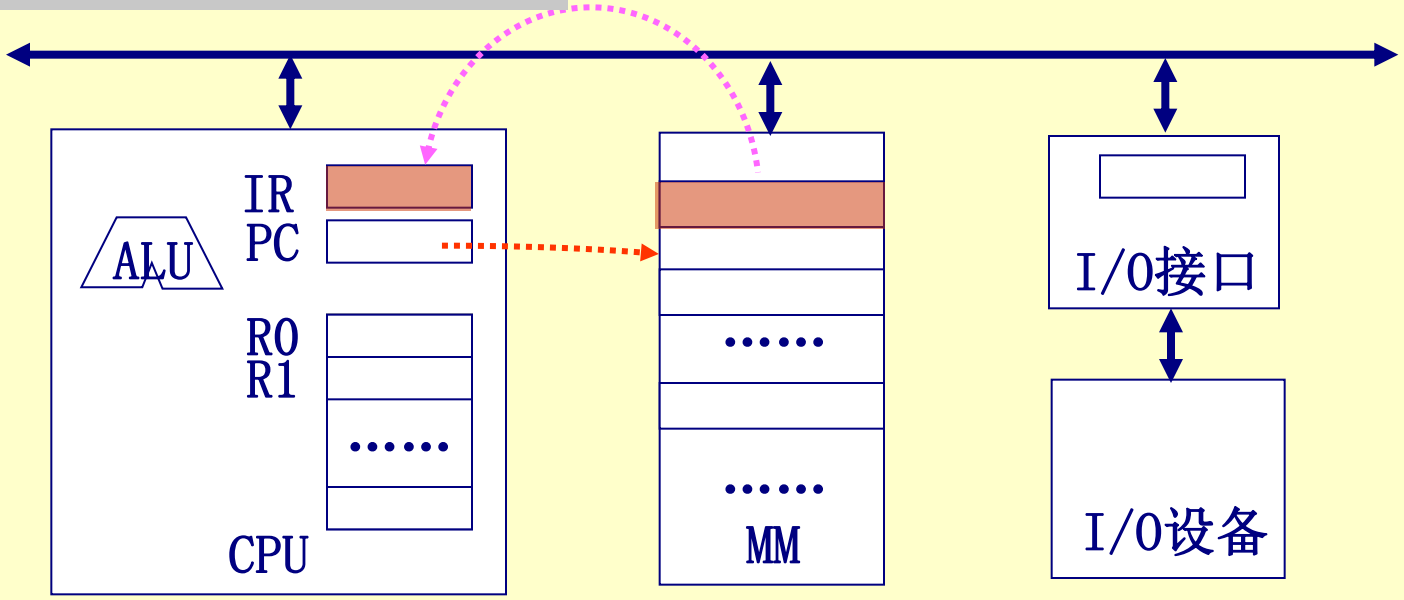
回顾：计算机执行程序的过程



以取指令过程为例，其具体操作包含以下内容：

（2）MM接到地址信息和“读内存”命令后，按第5章中所学的原理把内存相应单元中的二进制编码（即指令）读到内存的数据缓冲寄存器（MDR）中；同时，PC内容递增，为取下一条指令作准备；

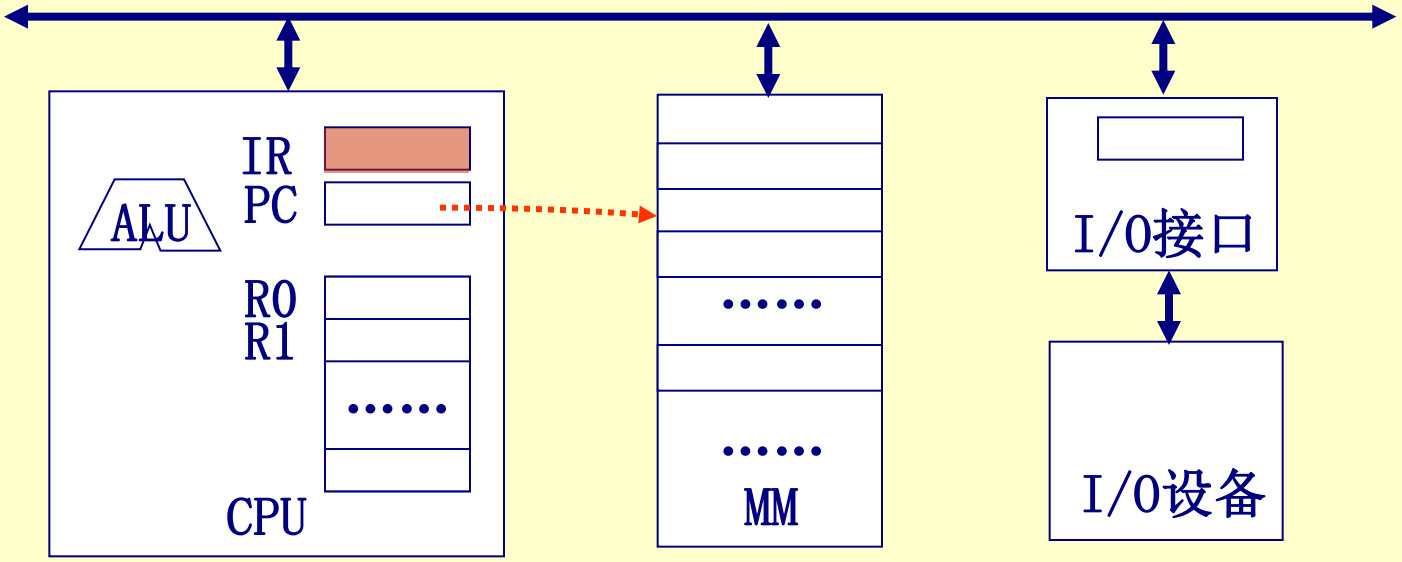
回顾：计算机执行程序的过程



以取指令过程为例，其具体操作包含以下内容：

（3）MM把MDR中的指令通过系统总线（的数据线）传到CPU的指令寄存器（IR）中，为下一步指令译码作准备。

回顾：计算机执行程序的过程



同样，“取操作数”和“执行”也要分解成若干步微操作来执行！

§6.1 CPU的功能

CPU的功能：

指令控制：产生下一条指令在内存中的地址、取指令；

操作控制：把指令分解成一系列的微操作控制信号，控制各部件完成指令所要求的动作；

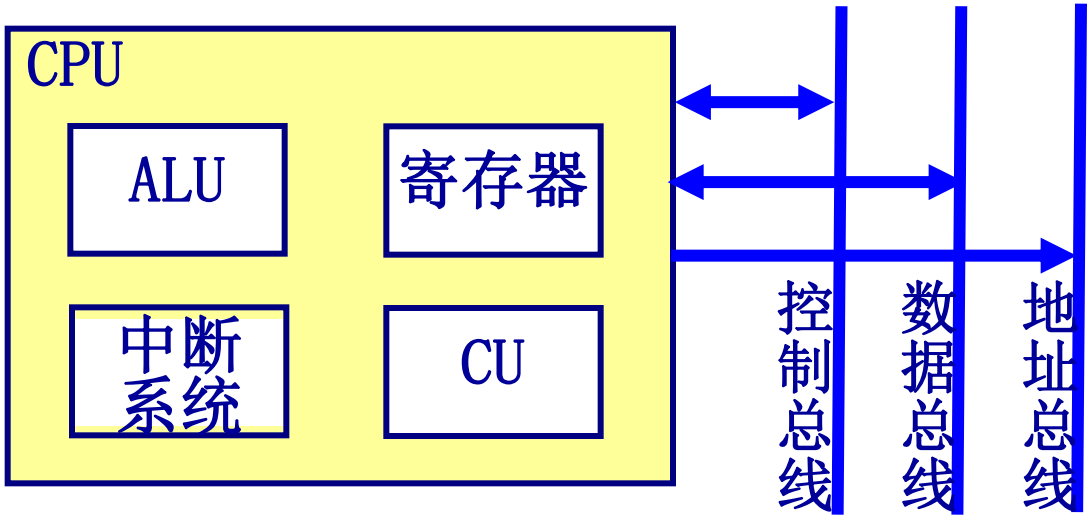
时序控制：对指令的各个微操作实施时间的定时，使它们能够按先后顺序来执行；

数据加工：算术运算、逻辑运算；

中断处理：处理异常情况和特殊请求。

其它：　　如总线管理等。

§6.2 CPU的组成



CPU内部主要包含：

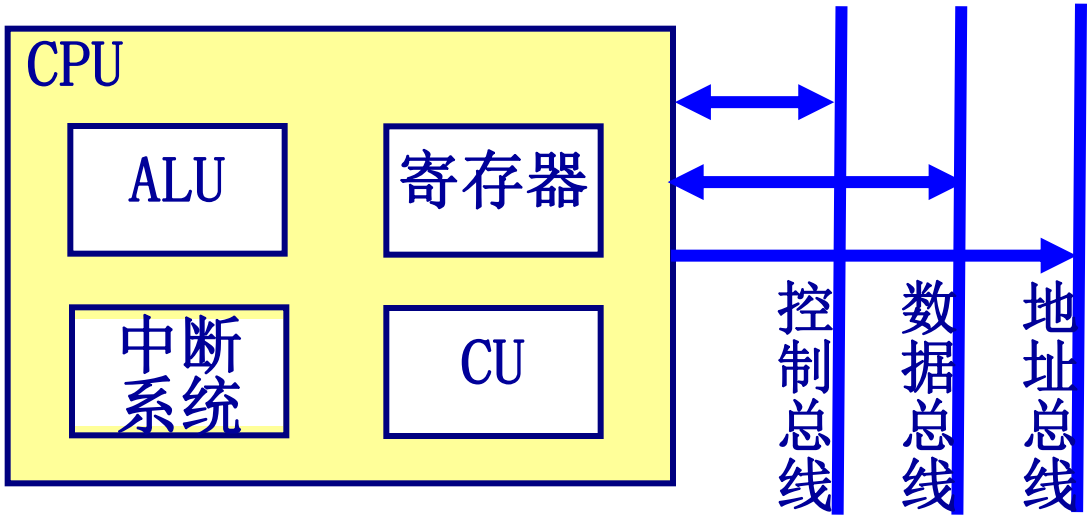
运算部件ALU

寄存器

控制部件（时序部件和微操作信号发生器）

中断控制逻辑等。

§6.2 CPU的组成



CPU总体结构的设计主要考虑：

1）设置哪些部件

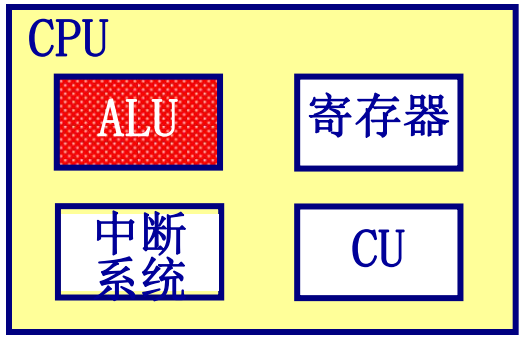
2）各部件之间如何连接（即数据通路）

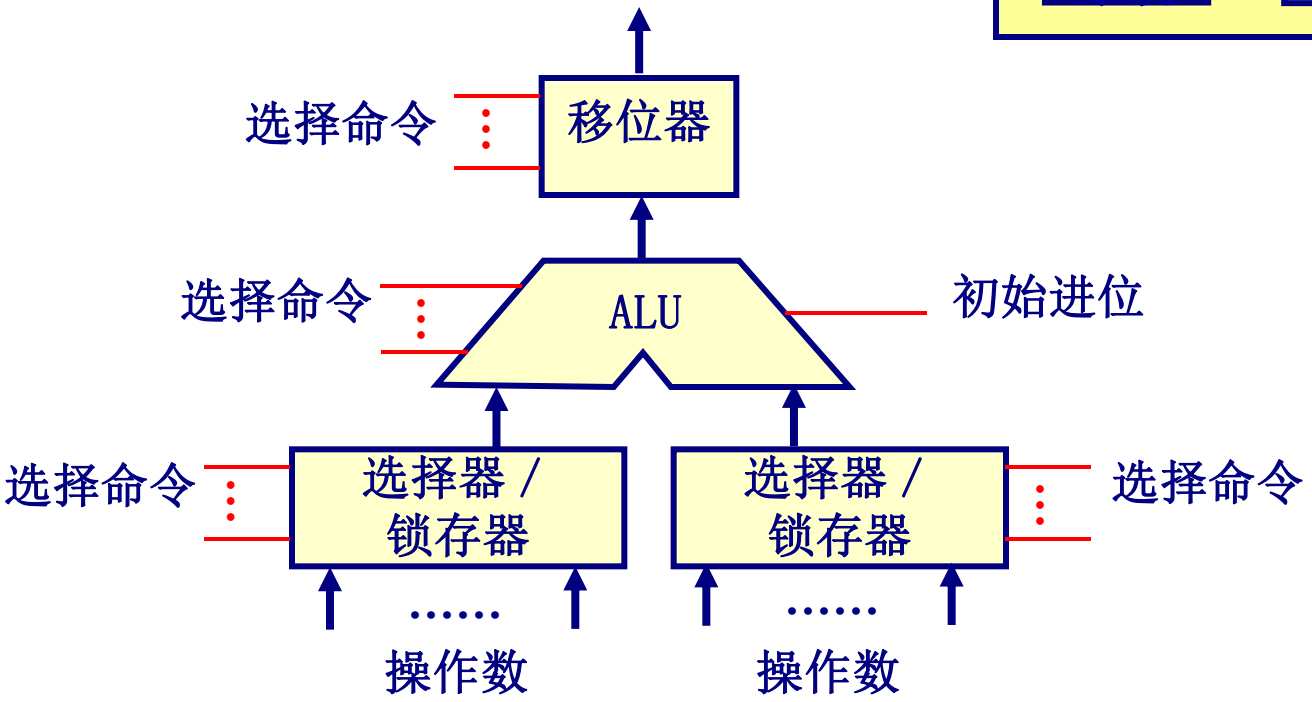
3）主机与外设之间如何传送信息

---- 与指令系统的设计有密切关系

6.2.1 运算部件

1. 基本的运算部件



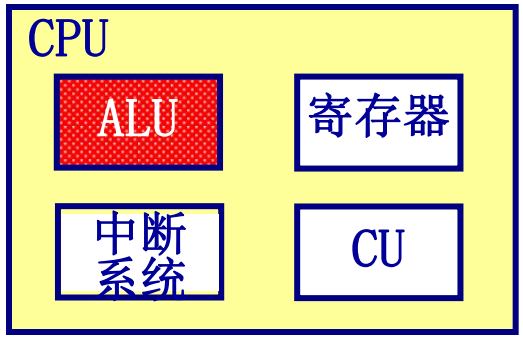


输出移位级

算逻运算级

输入级

6.2.1 运算部件



2. 不同档次的运算部件

(1) 低档处理器（早期）

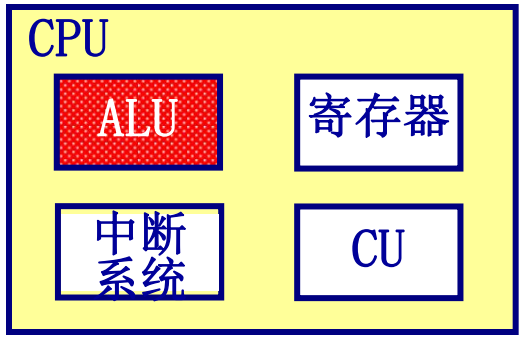
设置一个ALU，硬件级只能实现基本的逻辑运算功能。

(2) 普通处理器

设置一个ALU，硬件级可实现定点加减乘除法运算。

浮点运算可通过软件（子程序）或浮点协处理器硬件来实现。

6.2.1 运算部件



2. 不同档次的运算部件

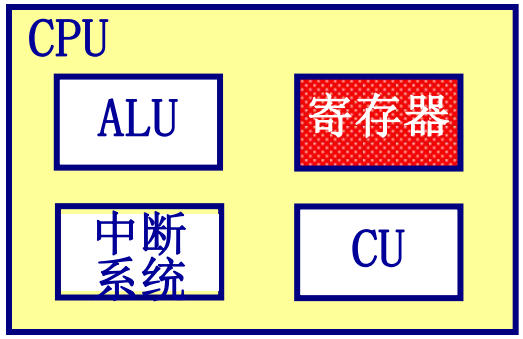
(3) 较高档处理器

单ALU，并将定点乘除和浮点部件作为基本配置。

(4) 高档处理器

包含多种运算部件，例如定点标量运算器、浮点运算器、向量运算部件等。

6.2.2 寄存器设置



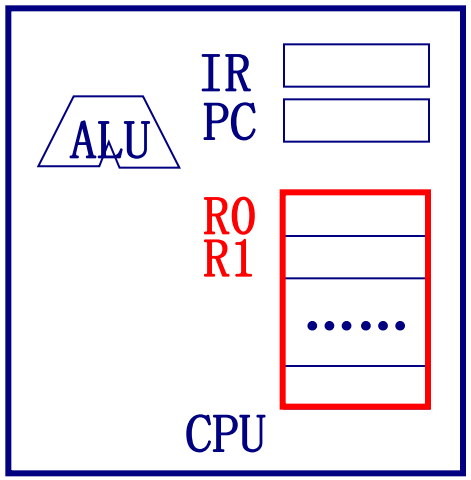
1. 通用寄存器

存放原始数据和运算结果；

用户可编程访问；

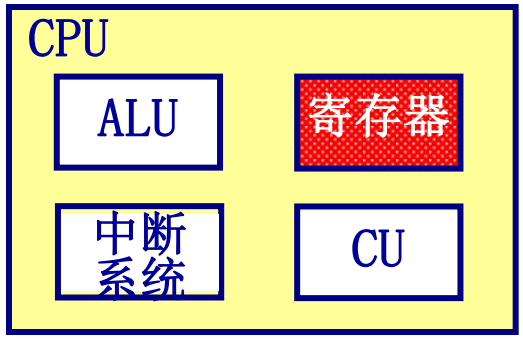
数量少则几个，多则上百个。

现代计算机常采用RAM或双口RAM来构成寄存器组。



例如：MOV R0, 25

6.2.2 寄存器设置

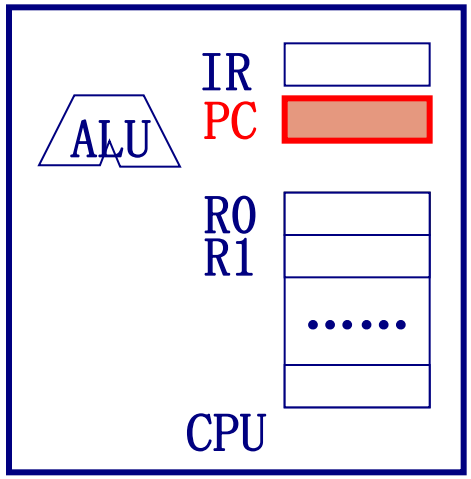


2. 专用寄存器

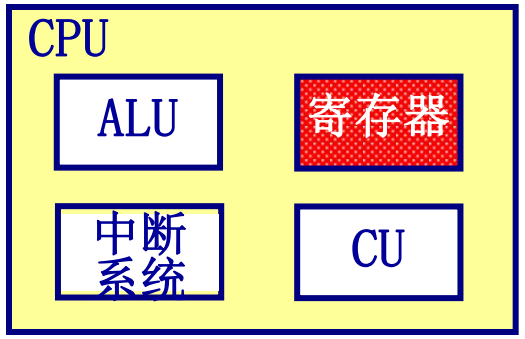
(1) 程序计数器PC （指令指针IP）

用于提供读取指令的地址。

通过PC内容的不断更新，控制执行指令序列的流向，从而产生所谓的控制流。



6.2.2 寄存器设置

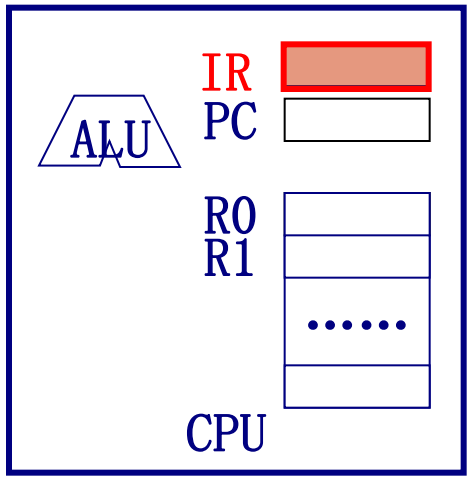


2. 专用寄存器

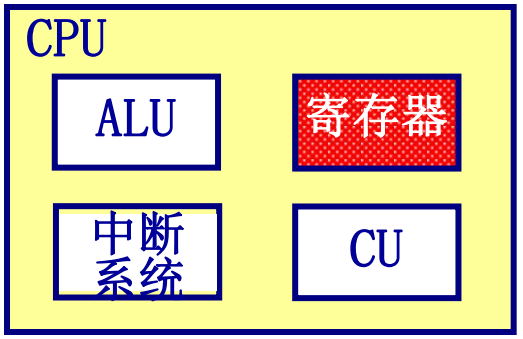
(2) 指令寄存器IR

用于存放现行指令。

为提高读取指令的速度，大多数计算机都将指令寄存器扩充为指令队列（或称指令栈），允许预取若干条指令，甚至有的还引入了指令Cache.



6.2.2 寄存器设置



2. 专用寄存器

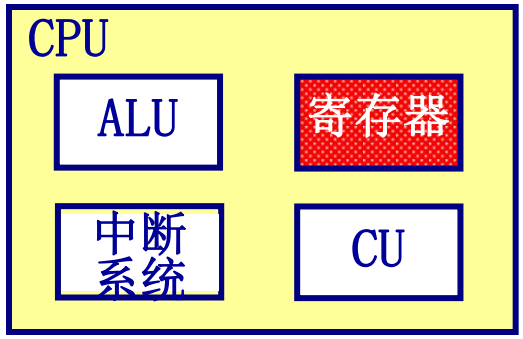
(3) 暂存器

用户不能直接访问的寄存器，用来暂存信息。

例如：源寄存器

暂存寄存器等。

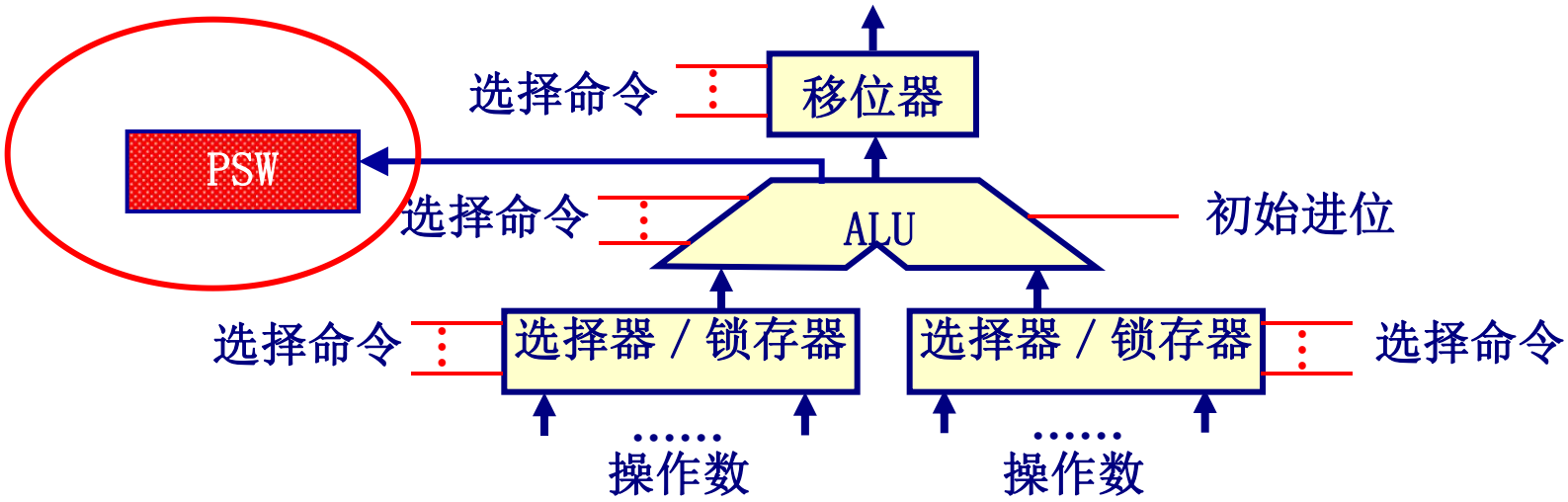
6.2.2 寄存器设置



2. 专用寄存器

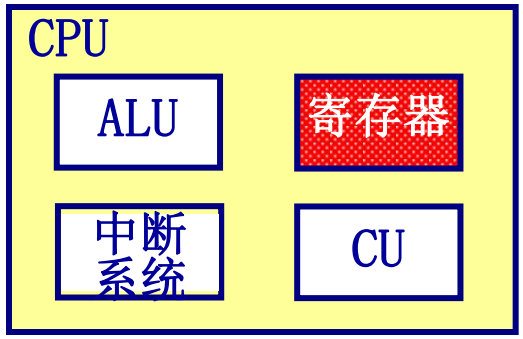
(4) 状态寄存器PSW

用于指示CPU的工作方式、算逻运算指令运行结果的特征等。



不同档次的计算机，其PSW的内容可能相差很大。

6.2.2 寄存器设置



2.专用寄存器

(4)状态寄存器PSW

例如：

PSW	工作方式	……	优先级	I	T	P	N	Z	V	C

I：允许中断　T：跟踪(陷阱)　P：奇偶位　N：结果为负　Z：结果为零　V：结果溢出　C：有进位

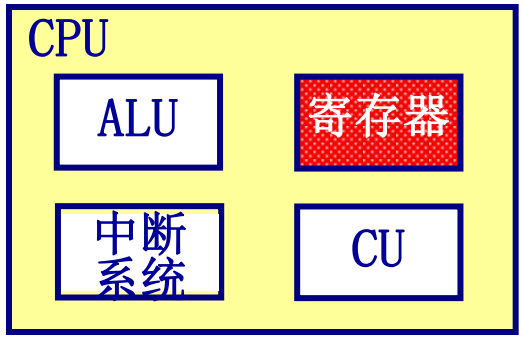
特征位：C、V、Z、N等表征算逻运算结果的特征；

跟踪位T，由编程设定断点；

允许中断I，为1时CPU允许响应外部中断请求。

有的计算机还设置有半进位AF、单步位TF等。

6.2.2 寄存器设置



2. 专用寄存器

(4) 状态寄存器PSW

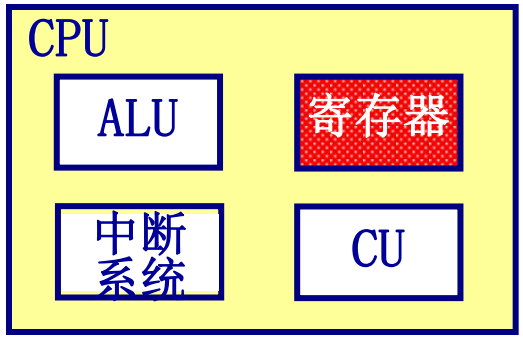
例如：

PSW	工作方式	……	优先级	I	T	P	N	Z	V	C
				允许中断	跟踪(陷阱)	奇偶位	结果为负	结果为零	结果溢出	有进位

优先级：当前运行程序的优先级。

当有外部中断请求的优先级高于它时，CPU才会响应中断请求。

6.2.2 寄存器设置



2. 专用寄存器

(4) 状态寄存器PSW

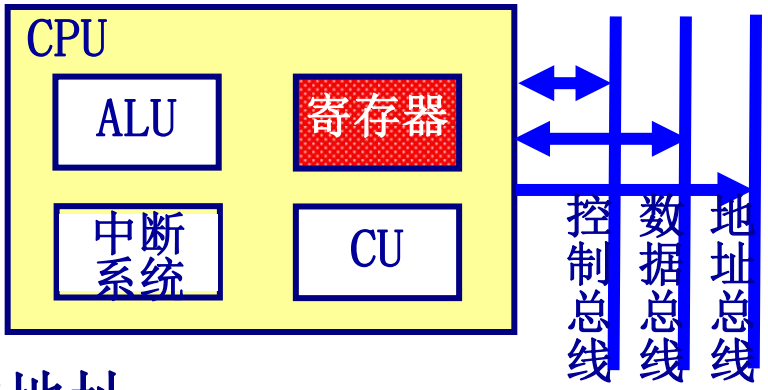
例如：

PSW	工作方式	.....	优先级	I	T	P	N	Z	V	C

I：允许中断　T：跟踪(陷阱)　P：奇偶位　N：结果为负　Z：结果为零　V：结果溢出　C：有进位

工作方式：有些计算机将CPU状态分为用户态和管态。

6.2.2 寄存器设置



2.用作主存接口的寄存器

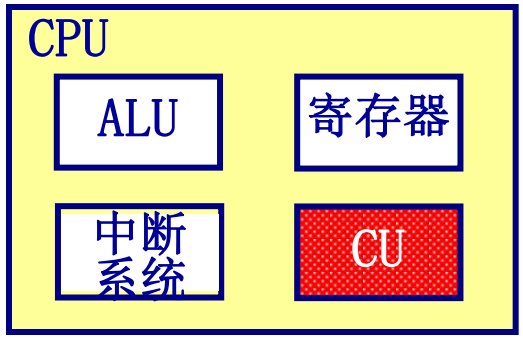
(1)地址寄存器AR（或MAR）

存放需要访问的内存单元的地址。

(2)数据寄存器DR（或MBR）

存放准备写入到内存单元的数据，或存放从内存读出后进入CPU的数据。

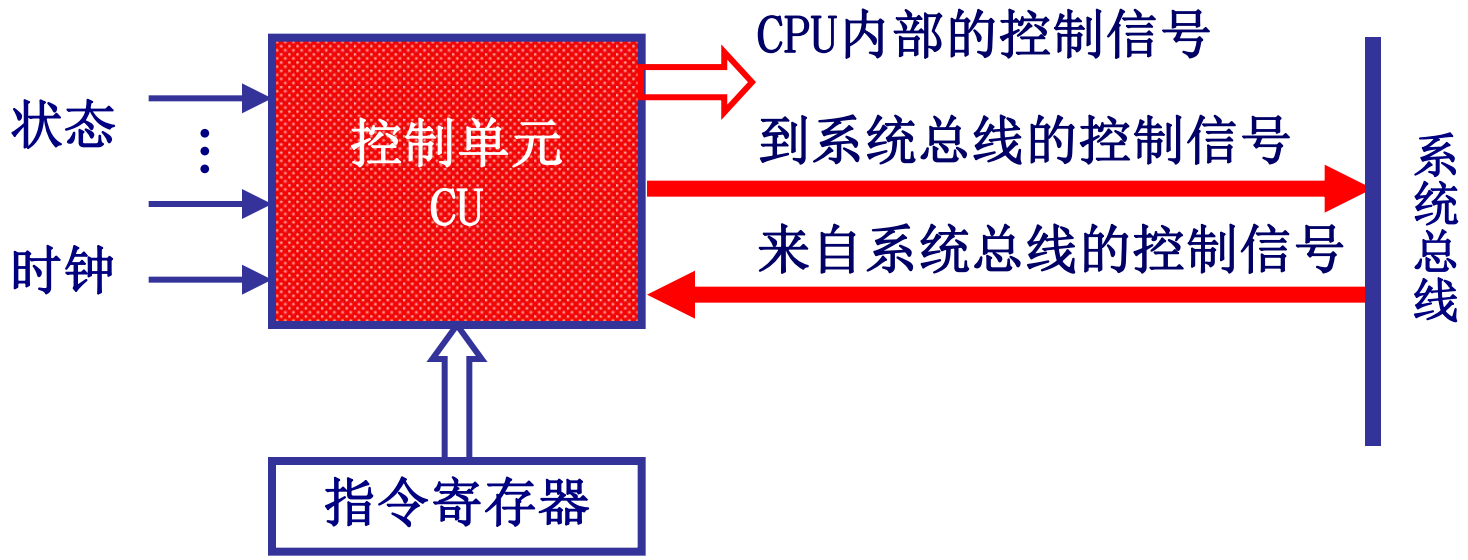
6.2.3 控制单元CU



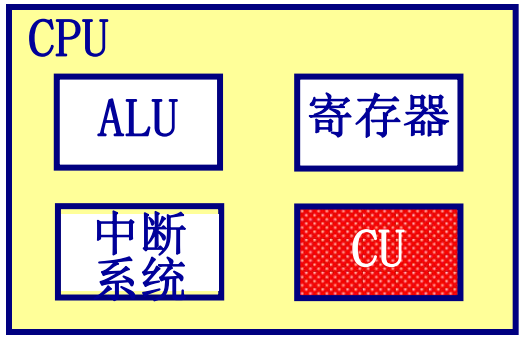
控制单元主要包括：

- 时序部件
- 微操作信号发生器

控制单元的外特性：



6.2.3 控制单元CU



1. 时序部件

时序部件一般包括：

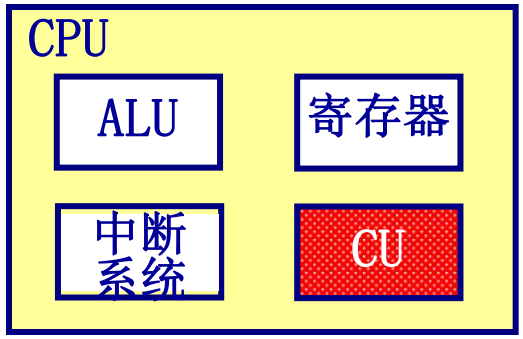
脉冲源 -- 基准时钟（主频）

启停控制逻辑

时序信号发生器等。

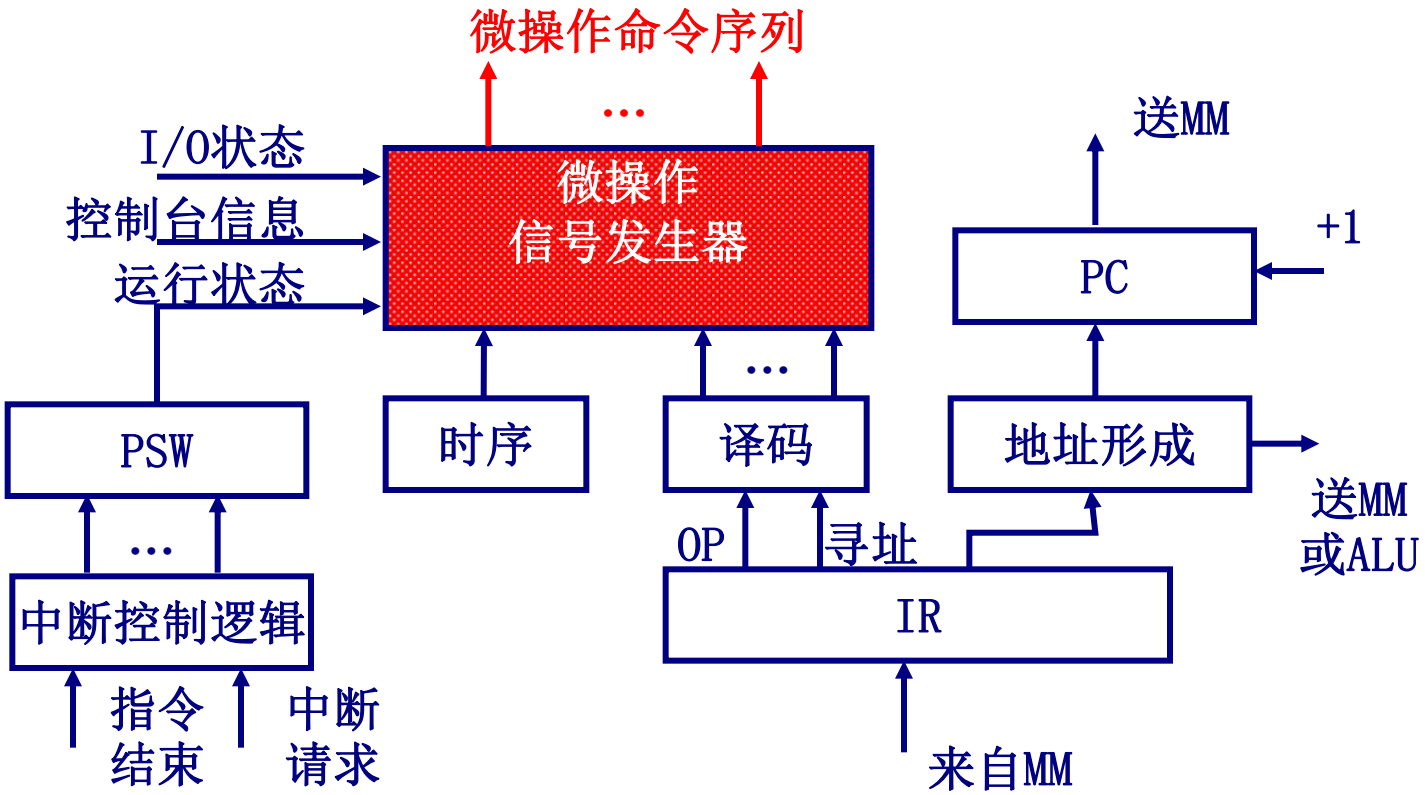
用于产生工作周期、节拍、脉冲等时间信号标志(时序信号)。

6.2.3 控制单元CU

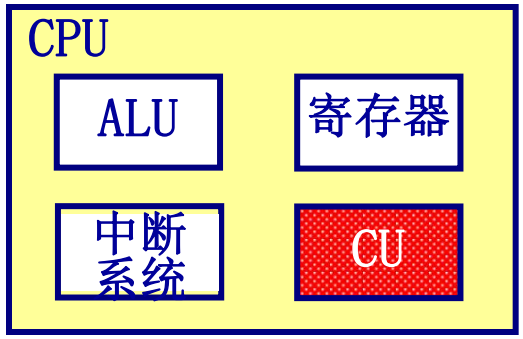


2. 微操作信号发生器

微操作信号：最基本的控制命令。



6.2.3 控制单元CU



2. 微操作信号发生器

微操作信号：最基本的控制命令。

产生微操作信号的基本依据：

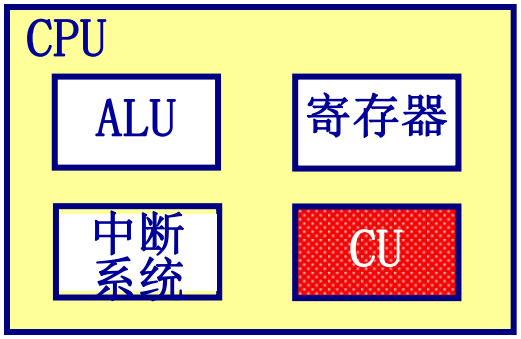
1) 时序信号（如工作周期、节拍周期、节拍脉冲等）

2) 指令代码（如操作码、寻址方式、寄存器号等）

3) 状态　　（如CPU内部的PSW、外设的状态等）

4) 外部请求（如控制台请求、外部中断请求、DMA请求等）

6.2.3 控制单元CU



2. 微操作信号发生器

微操作信号：最基本的控制命令。

根据微操作信号的形成方式，控制器可分为：

1）组合逻辑控制器

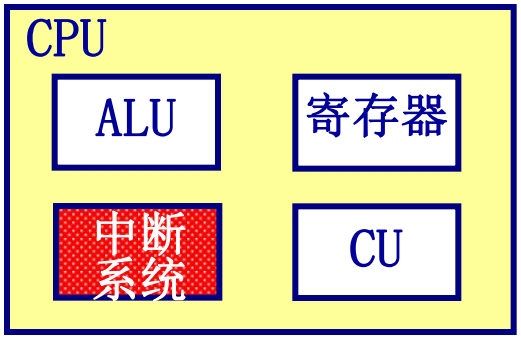
--速度快，电路很不规整，逻辑网络复杂；

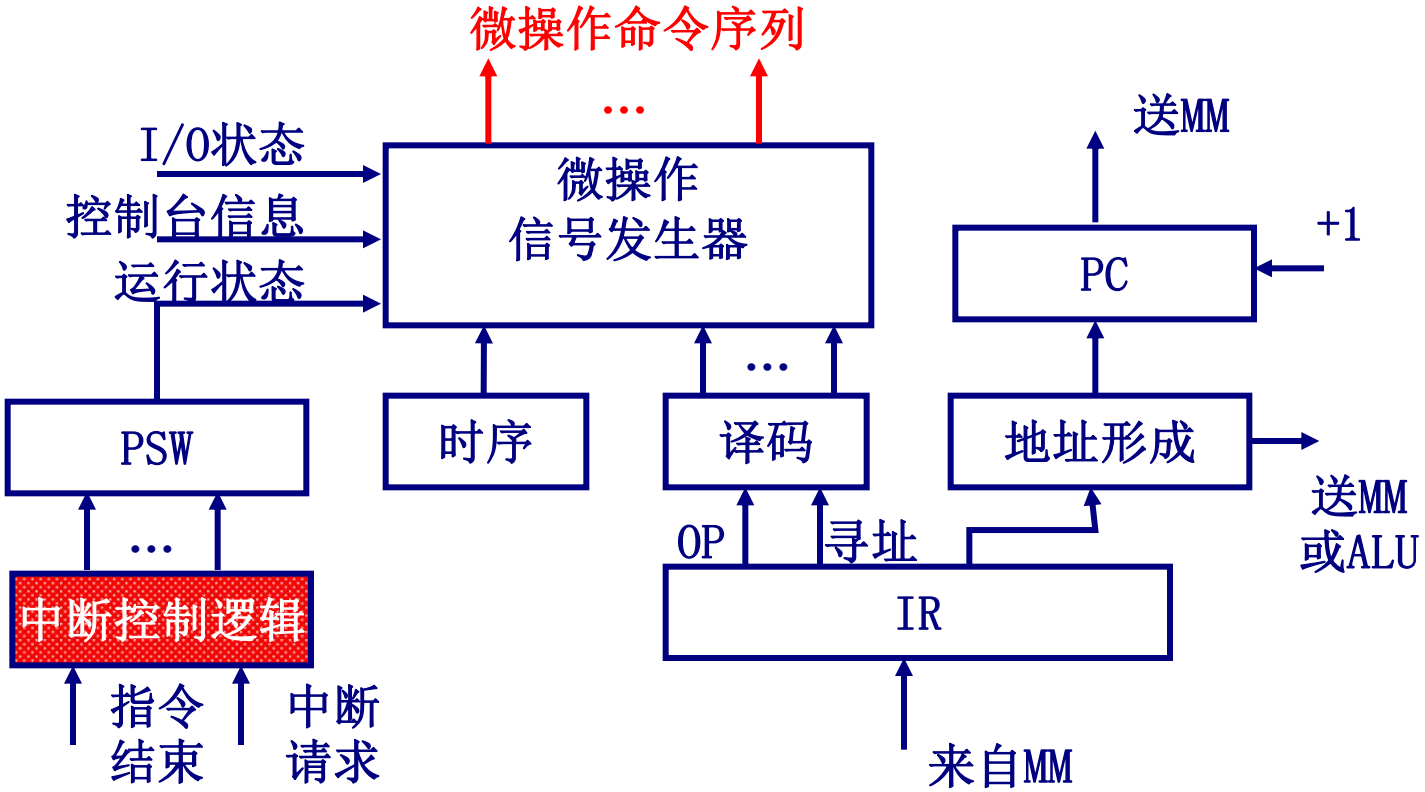
2）微程序控制器

--电路规整，易扩充，速度稍慢。

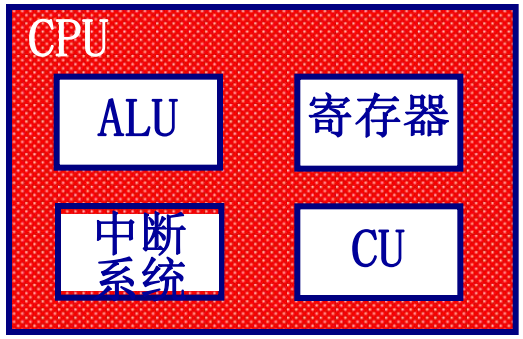
6.2.4 中断控制逻辑

用来控制中断处理的硬件逻辑。





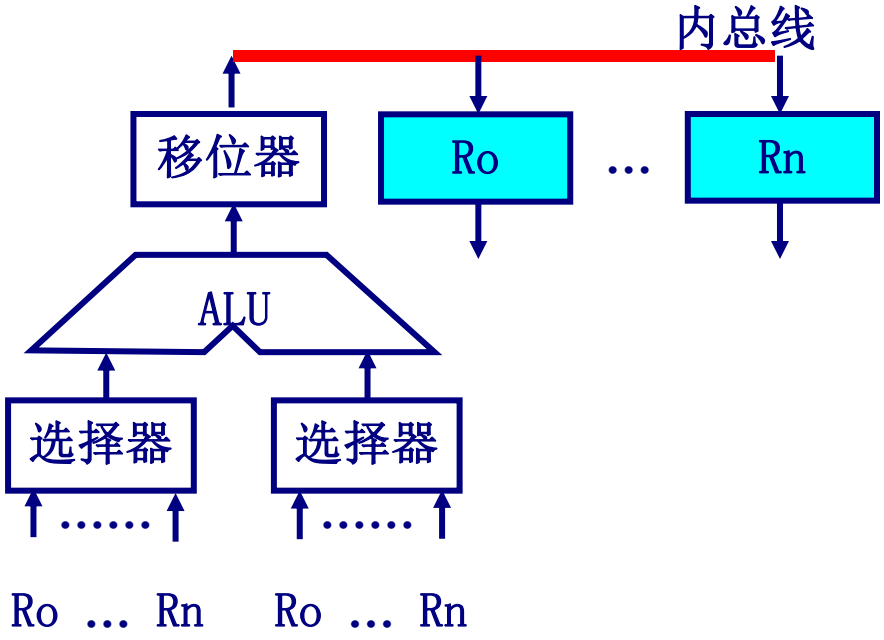
6.2.5 CPU内部数据通路结构



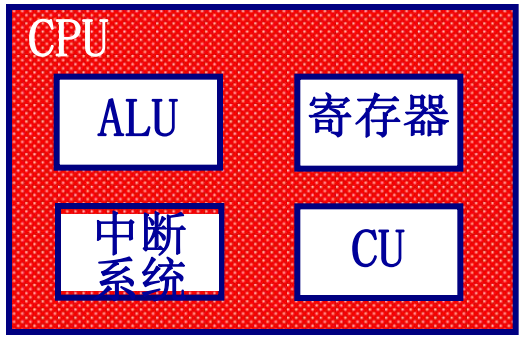
通常用总线的连接方式。

1） 单组内总线数据通路结构

例：分立寄存器结构



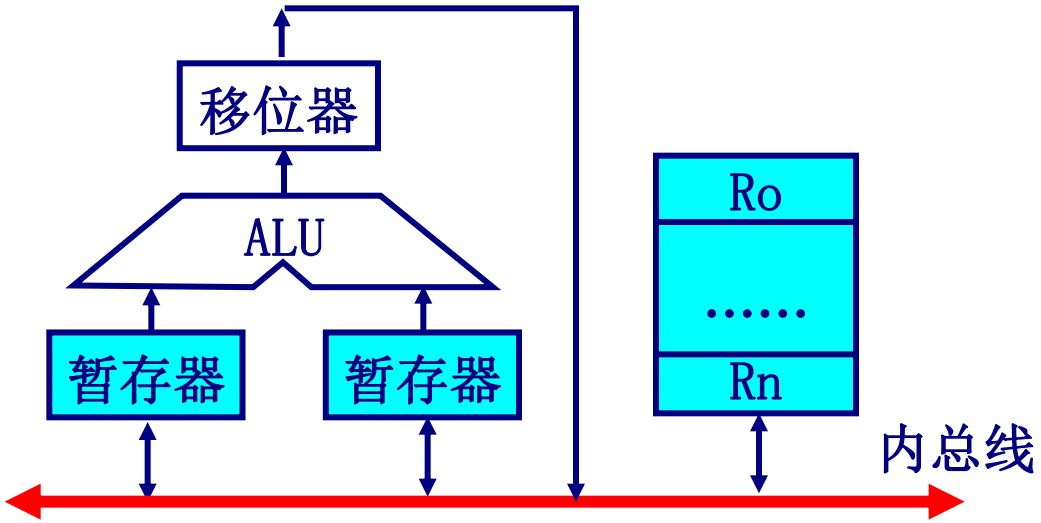
6.2.5 CPU内部数据通路结构



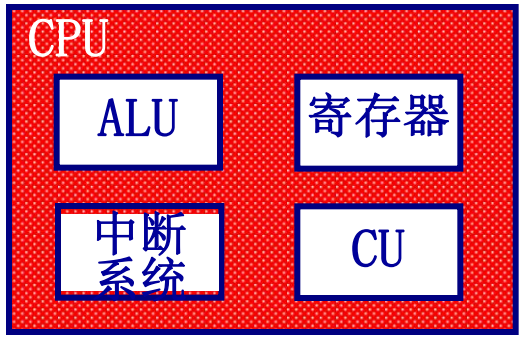
通常用总线的连接方式。

1） 单组内总线数据通路结构

例：集成寄存器结构



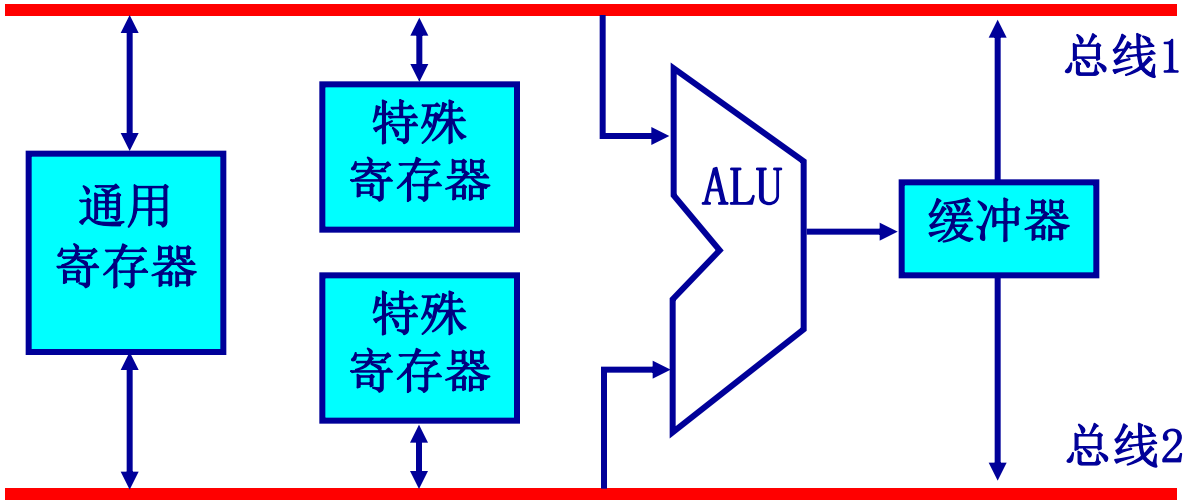
6.2.5 CPU内部数据通路结构



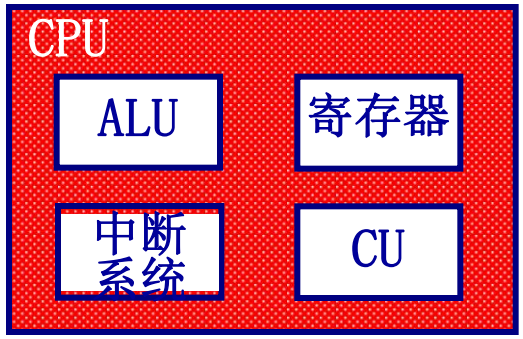
通常用总线的连接方式。

2) 多组内总线结构

例：双总线结构



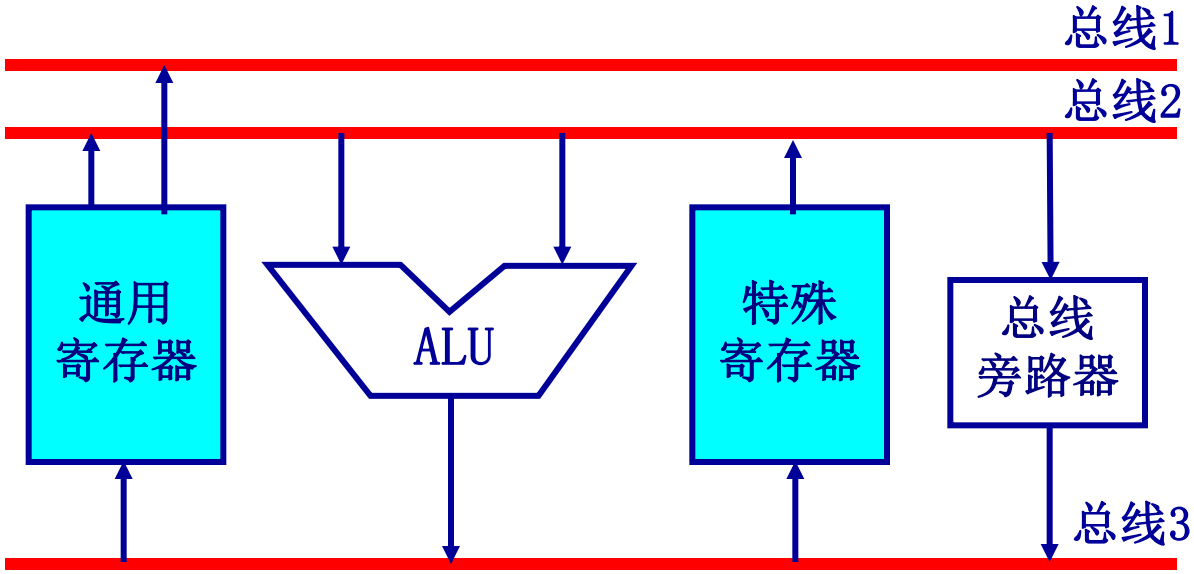
6.2.5 CPU内部数据通路结构



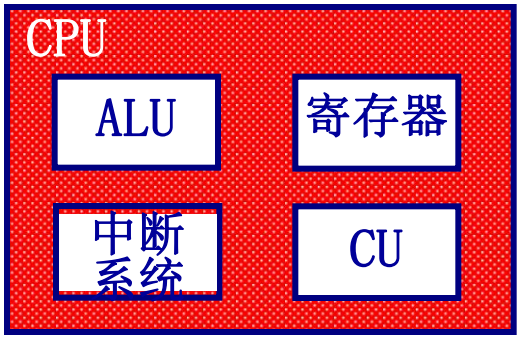
通常用总线的连接方式。

2）多组内总线结构

例：三总线结构



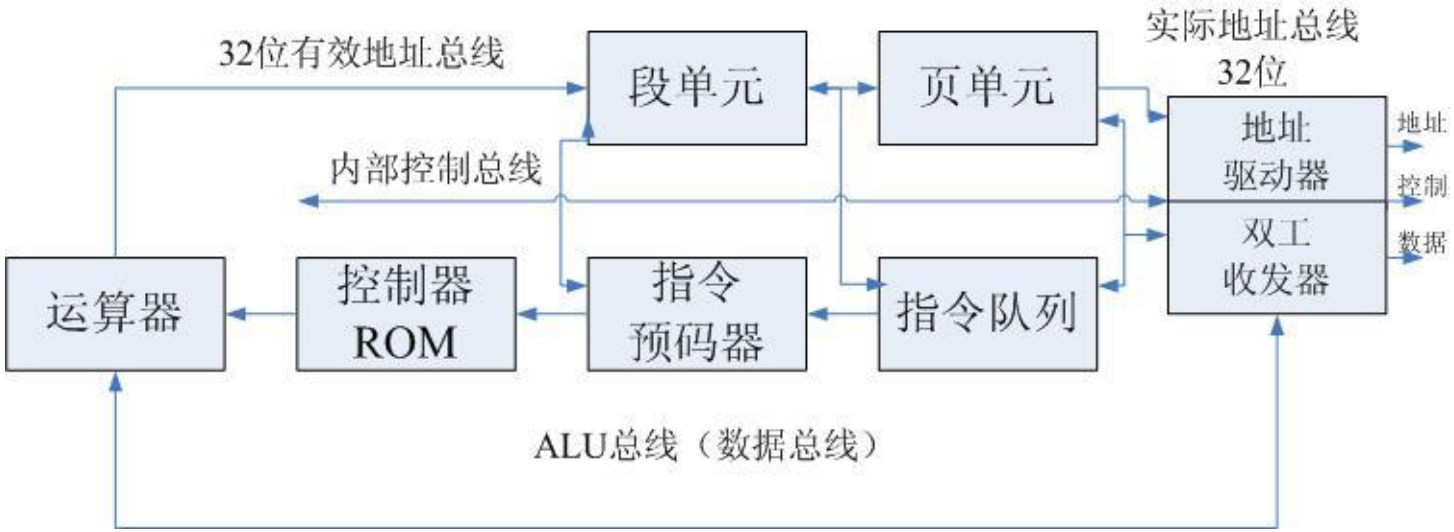
6.2.5 CPU内部数据通路结构



通常用总线的连接方式。

2) 多组内总线结构

例：Intel80386 的多组、多种内部总线。



6.2.6 CPU芯片的主要技术参数

1. 字长
2. 内部工作频率
3. 外部工作频率

   也叫前端总线频率或系统总线时钟频率。

   内频=外频×倍频
4. 片内Cache的容量和速率

   片内Cache的运行速度与内频相同或接近，

   容量可达几十KB～几百KB。

6.2.6 CPU芯片的主要技术参数

5. 工作电压

工作电压指的是CPU正常工作所需的电压。

6. 地址总线宽度

决定了CPU可以访问的最大的物理地址空间。

7. 数据总线宽度

8. 制造工艺

线宽越小，意味着芯片上包括的晶体管数目越多。

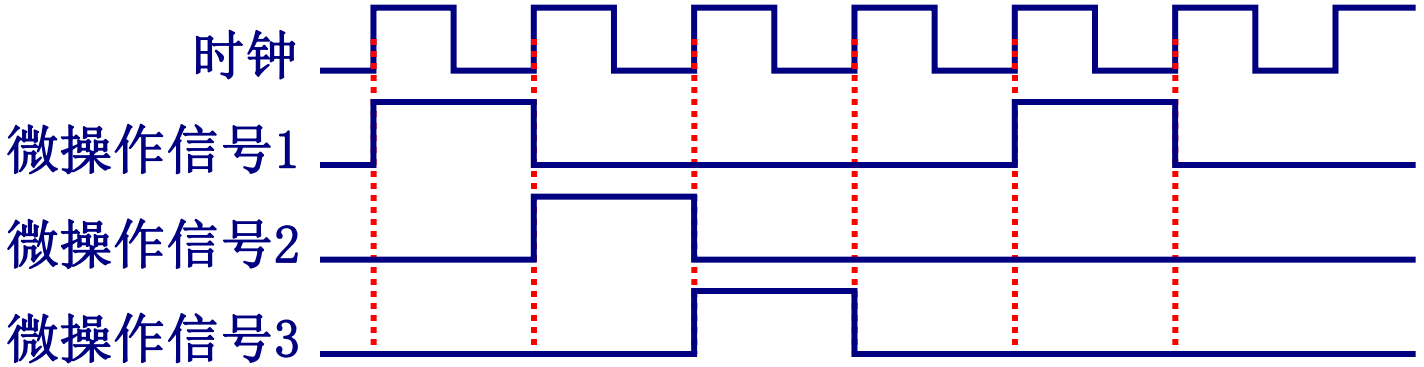
§6.3 时序控制方式与时序系统

6.3.1 时序控制方式

指完成指令的各微操作在与时序上采用何种协调关系。

1）同步控制方式

各项微操作都由固定的，统一的时序进行控制。



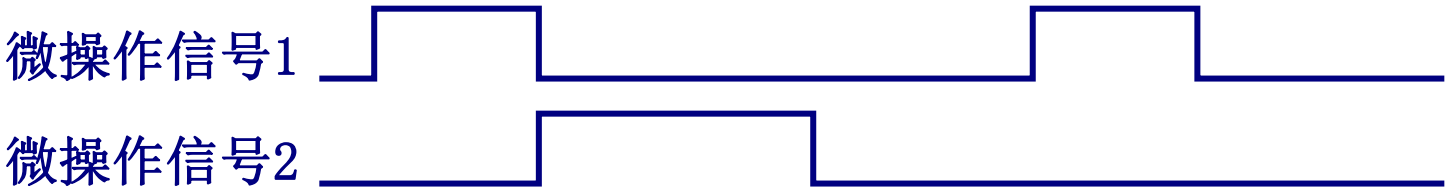
特点：控制方式简单，容易实现，存在时间浪费。

应用：CPU内部或设备内部。

6.3.1 时序控制方式

2）异步控制方式

各微操作按其需要选择不同的时间间隔，不受统一的时间的约束；各微操作之间的衔接与各部件之间的信息交换采用应答方式。



特点：没有时间上的浪费，因而提高了机器的效率，但是控制比较复杂。

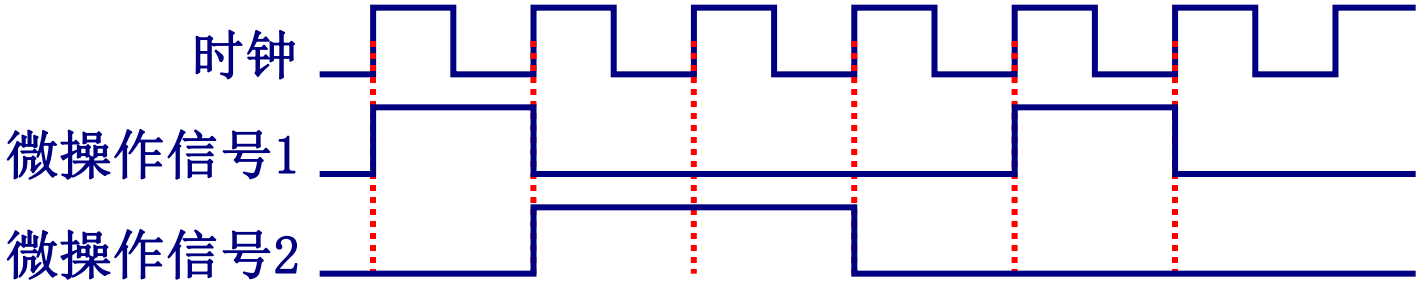
应用：用于系统总线操作控制

6.3.1 时序控制方式

3）准同步控制方式

异步方式的同步化

（只在节拍结束时查询异步应答信号）。



4）联合控制方式

同步控制和异步控制相结合的方式。

6.3.2 同步控制方式下的多级时序系统

1. 时序的层次划分

指令周期：从取指到一条指令执行结束所需的时间。

在同步控制方式下，常将时序关系把指令周期划分为几个层次，称为多级时序。

最常见的是二级时序和三级时序。

6.3.2 同步控制方式下的多级时序系统

2. 多级时序划分举例

（1）三级时序举例（常用在组合逻辑控制器中）

<table>
<tr><td rowspan="6">指令周期</td><td rowspan="3">工作周期0</td><td>节拍0</td><td>节拍脉冲</td></tr>
<tr><td>……</td><td></td></tr>
<tr><td>节拍n</td><td>节拍脉冲</td></tr>
<tr><td>……</td><td></td><td></td></tr>
<tr><td rowspan="3">工作周期m</td><td>……</td><td></td></tr>
<tr><td>……</td><td></td></tr>
<tr><td>……</td><td></td></tr>
</table>

第1级　　第2级　　第3级

在三级时序系统中，一个指令周期分为三级时序：

① CPU工作周期（又称为机器周期）

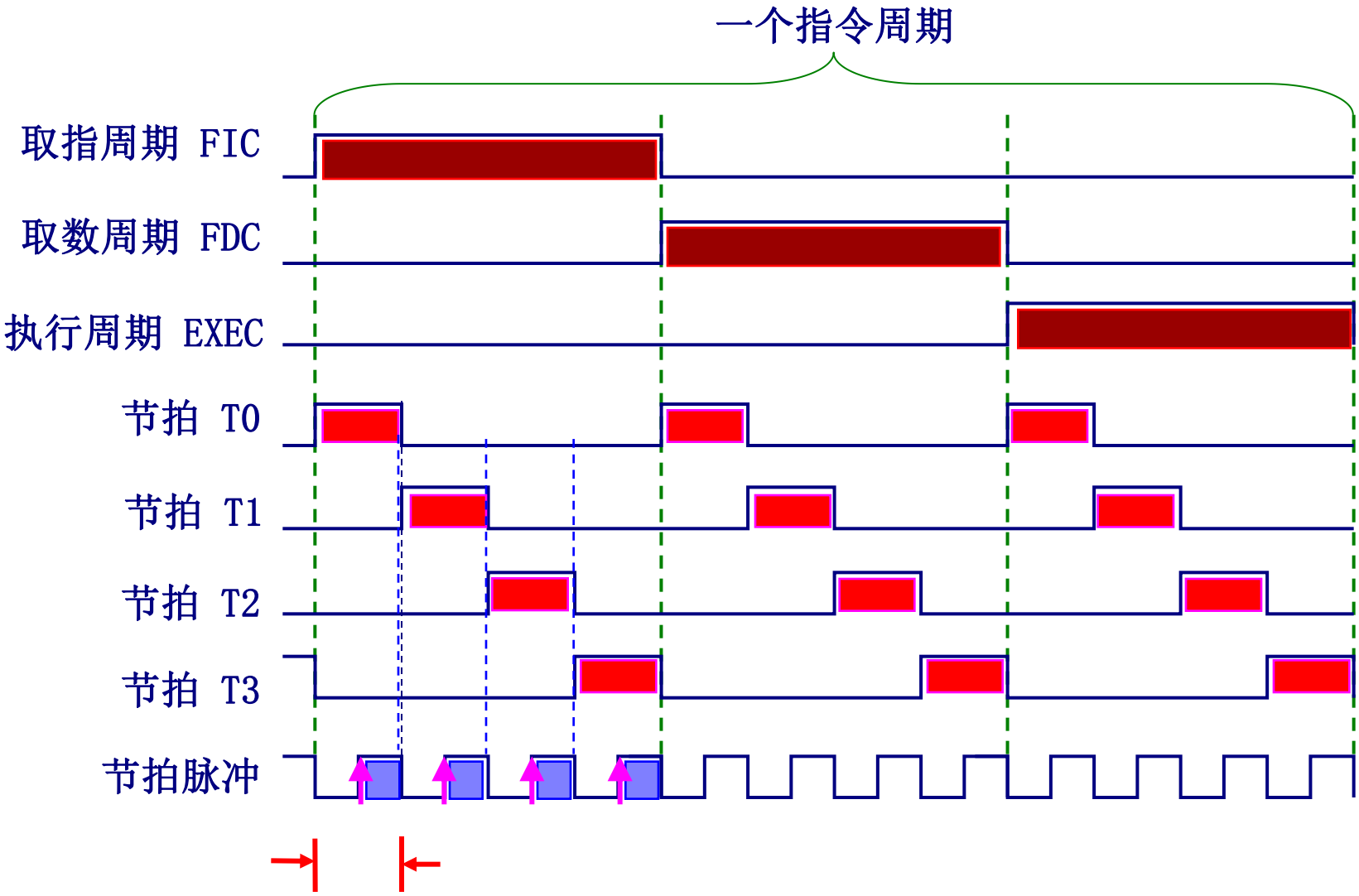
对应于指令执行的某个阶段（如取指、取操等），通常伴随着一次总线操作（访内）。

② 节拍（又称为时钟周期）

完成CPU内部一些最基本操作所需的时间。比如寄存器之间的数据传送等。

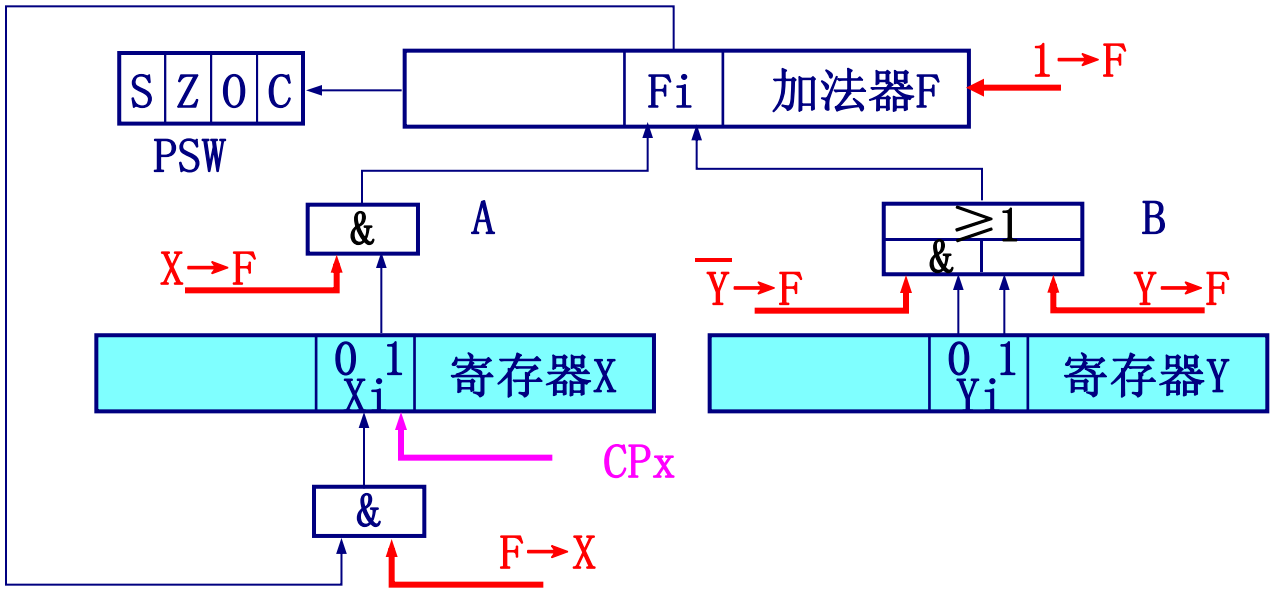
③ 节拍脉冲（又称为工作脉冲、定时脉冲）

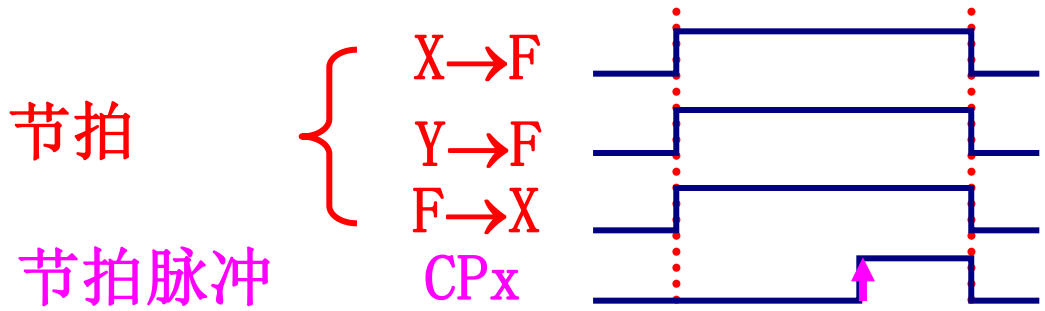
通常作为触发器（寄存器）的打入脉冲，与节拍相配合完成一次数据传送。



一个微操作持续的时间，为一个节拍的时间（即一个时钟周期的时间）。

例：X←X+Y

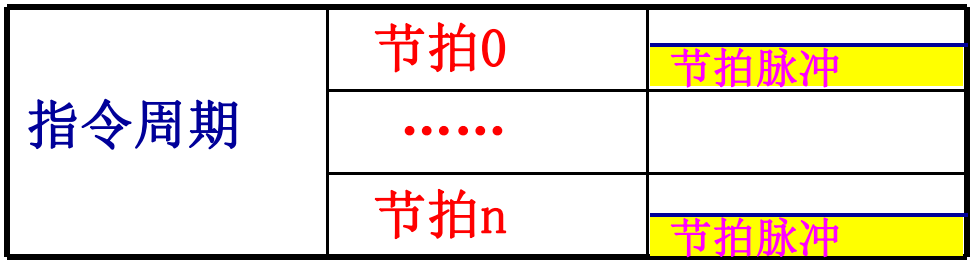

S Z O C
PSW
Fi
加法器F
1→F
&
A
X→F
≥1
&
B
Y→F
Y→F
0 1
Xi
寄存器X
0 1
Yi
寄存器Y
CPx
&
F→X


节拍
节拍脉冲
X→F
Y→F
F→X
CPx

6.3.2 同步控制方式下的多级时序系统

2. 多级时序划分举例

（2）二级时序举例 （常用在为程序控制器中）



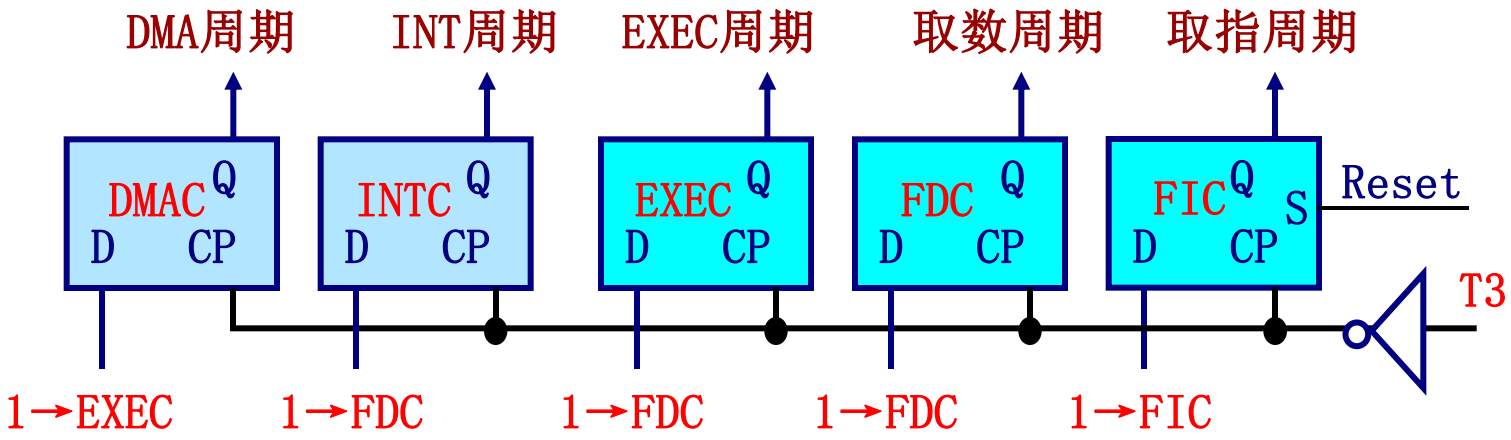
6.3.2 同步控制方式下的多级时序系统

3. 工作周期与节拍的信号生成

（1）CPU工作周期信号发生器

每个工作周期都用一个触发器与之对应。

在一条指令运行的任何时刻，只能处于一种工作周期状态，因此，有一个且仅有一个触发器被置“1”。

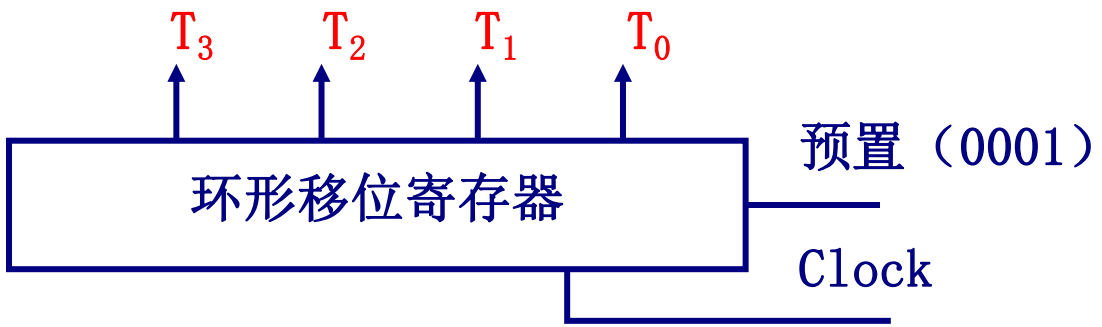


6.3.2 同步控制方式下的多级时序系统

3. 工作周期与节拍的信号生成

（2）节拍信号发生器（以统一节拍法为例）

可采用环形一位寄存器等。



T3

思考题：P233 1-5，8，9

P209 6,7，13，14，15，19，20

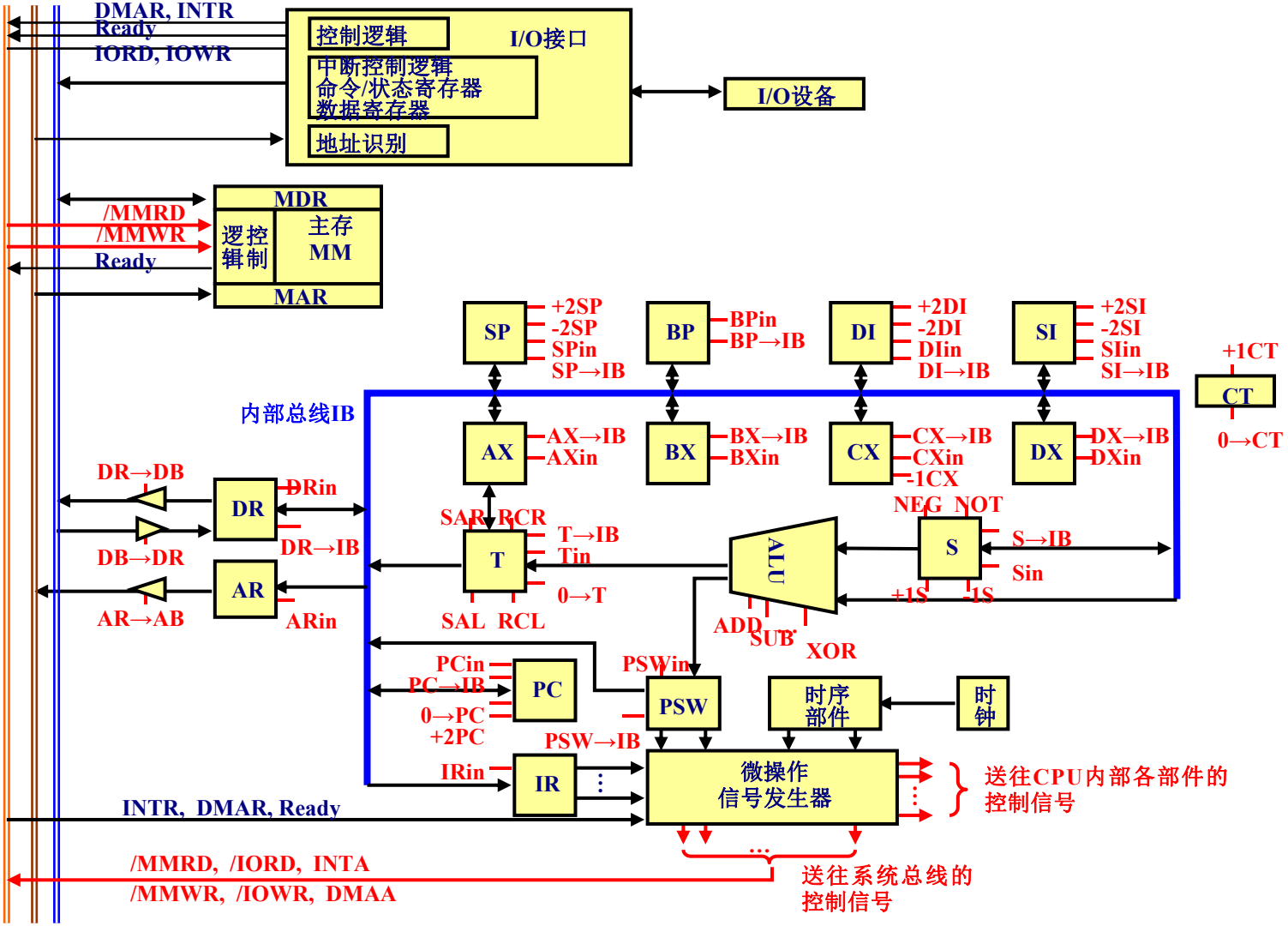
习题： P233 1-5，8，9

P209 6,7，13，14，15，19，20

§6.4 指令的微操作序列

大图

6.4.1 模型机



§6.4 指令的微操作序列

6.4.1 模型机

1. 硬件结构（PC机的简化） 16位机

(1)总线

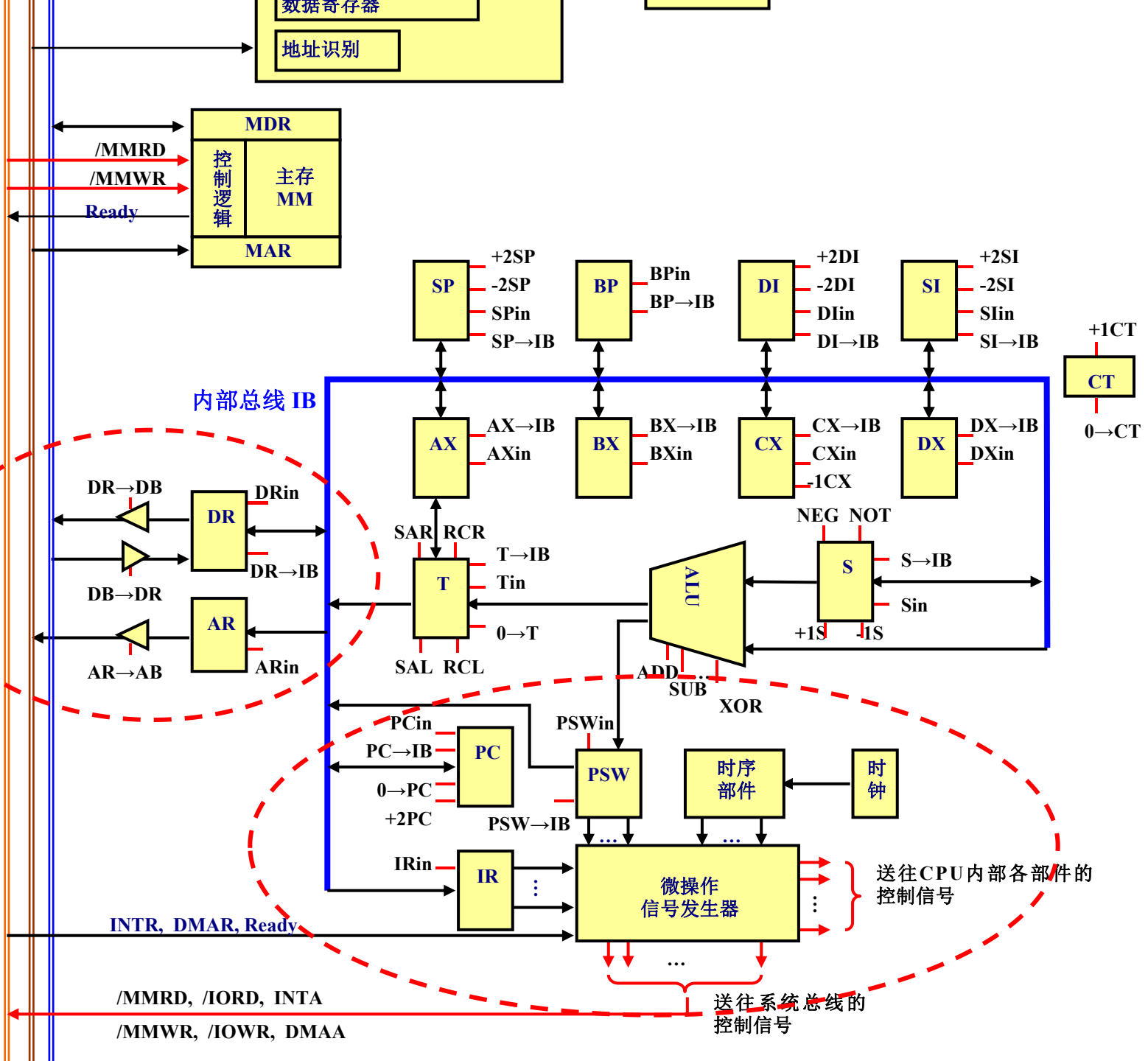
系统总线：单总线（地址线、数据线、控制线）。

内部总线：各寄存器用三态门接入。

二者之间通过地址寄存器AR和数据寄存器DR沟通信息。

(2)控制器

包括PC, IR, PSW, 时序部件、微操作形成部件等。


数据寄存器
地址识别
MDR
/MMRD
/MMWR
Ready
控制逻辑
主存MM
MAR
SP
+2SP
-2SP
SPin
SP→IB
BP
BPin
BP→IB
DI
+2DI
-2DI
DIin
DI→IB
SI
+2SI
-2SI
SIin
SI→IB
+1CT
CT
0→CT
内部总线 IB
AX
AX→IB
AXin
BX
BX→IB
BXin
CX
CX→IB
CXin
-1CX
DX
DX→IB
DXin
NEG NOT
DR→DB
DR
DRin
DR→IB
DB→DR
AR
ARin
AR→AB
SAR RCR
T
T→IB
Tin
0→T
SAL RCL
ALU
S
S→IB
Sin
+1S -1S
ADD SUB
XOR
PCin
PC→IB
PC
0→PC
+2PC
PSWin
PSW
PSW→IB
时序部件
时钟
IRin
IR
微操作信号发生器
送往CPU内部各部件的控制信号
INTR, DMAR, Ready
/MMRD, /IORD, INTA
/MMWR, /IOWR, DMAA
送往系统总线的控制信号

(3) 运算器

ALU：算逻运算功能

(ADD, ADC, SUB, SBB, IMUL, ...,

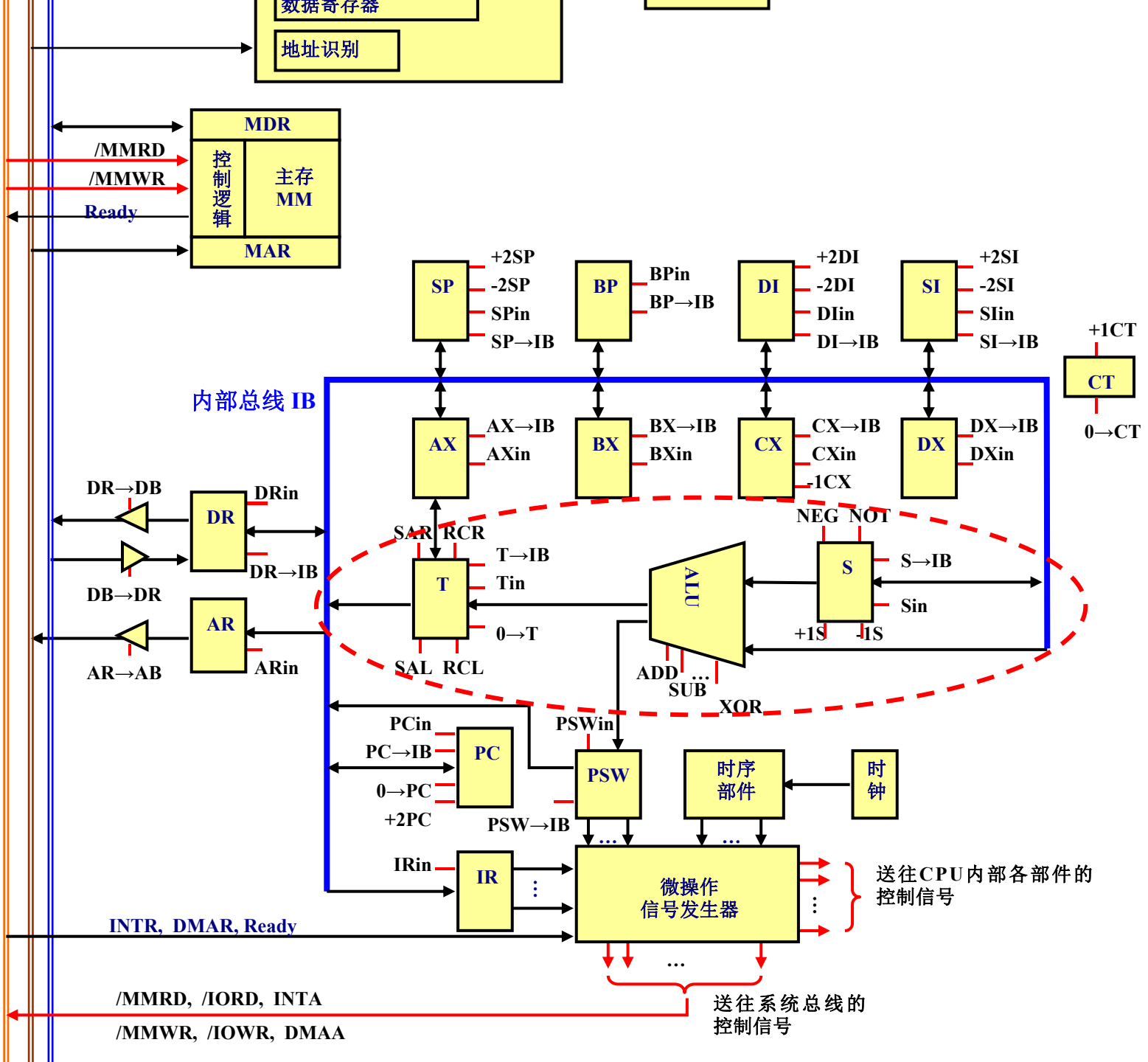
OR, XOR, CMP, TEST, ...

寄存器：AX, BX, CX, DX, SI, DI, BP, SP　　16位

暂存器：S ——具有加 1 、减 1 功能；

T ——具有移位功能。

乘除步数计数器：CT ——可清0, +1。


数据寄存器
地址识别
MDR
/MMRD
/MMWR
Ready
控制逻辑
主存MM
MAR
SP
+2SP
-2SP
SPin
SP→IB
BP
BPin
BP→IB
DI
+2DI
-2DI
DIin
DI→IB
SI
+2SI
-2SI
SIin
SI→IB
+1CT
CT
0→CT
内部总线 IB
AX
AX→IB
AXin
BX
BX→IB
BXin
CX
CX→IB
CXin
-1CX
DX
DX→IB
DXin
NEG NOT
DR→DB
DR
DRin
DR→IB
DB→DR
SAR RCR
T
T→IB
Tin
0→T
SAL RCL
ALU
S
S→IB
Sin
+1S -1S
AR
AR→AB
ARin
ADD …
SUB
XOR
PCin
PC→IB
PC
0→PC
+2PC
PSWin
PSW
PSW→IB
时序部件
时钟
IRin
IR
微操作信号发生器
送往CPU内部各部件的控制信号
INTR, DMAR, Ready
/MMRD, /IORD, INTA
/MMWR, /IOWR, DMAA
送往系统总线的控制信号

(4) 内存和I/O设备

寄存器： MAR 存储器的地址寄存器；

MDR 存储器的数据寄存器；

控制信号：/MMRD 存储器读；

/MMWR 存储器写；

READY 存储器（或I/O）工作完成；

/IORD I/O读；

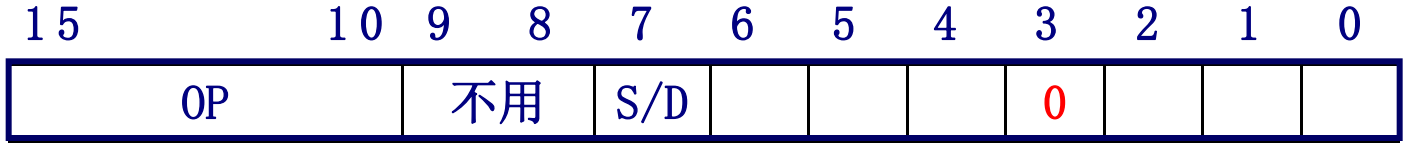
/IOWR I/O写。

2. 模型机指令系统及寻址方式

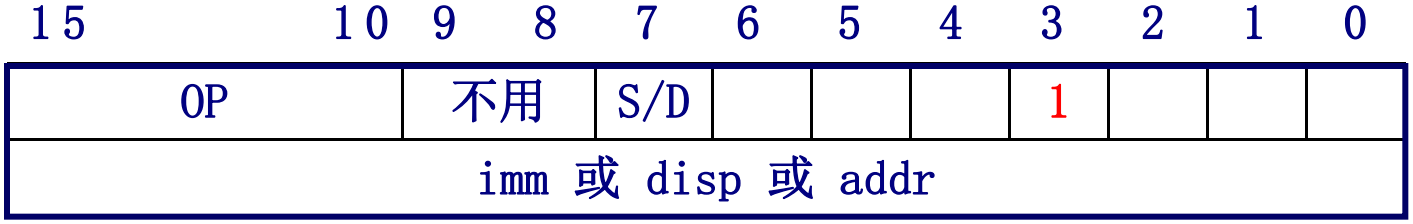
(1)指令系统

①双操作数指令

单字指令：

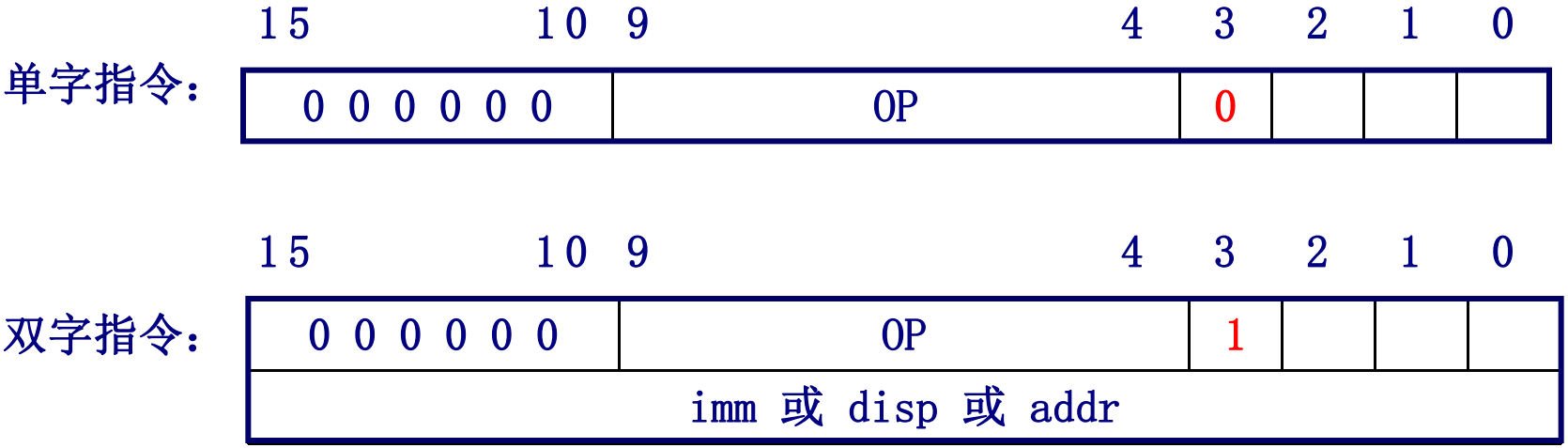


双字指令：



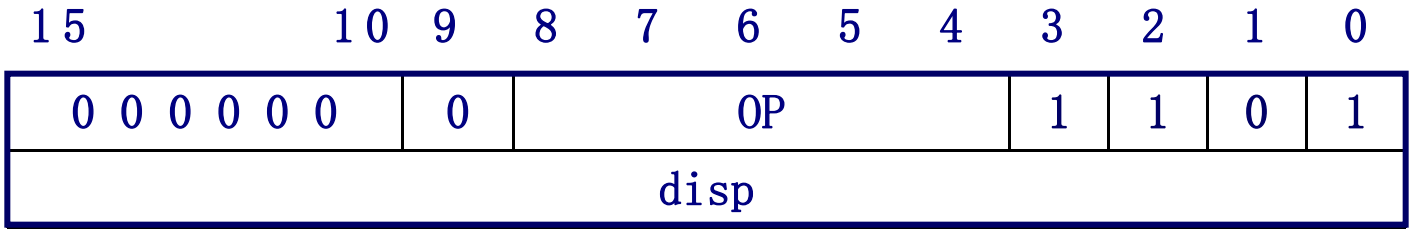
如：MOV、IN/OUT、ADD、ADC、SUB、SUBB、CMP、AND、OR、XOR、TEST

②单操作数指令



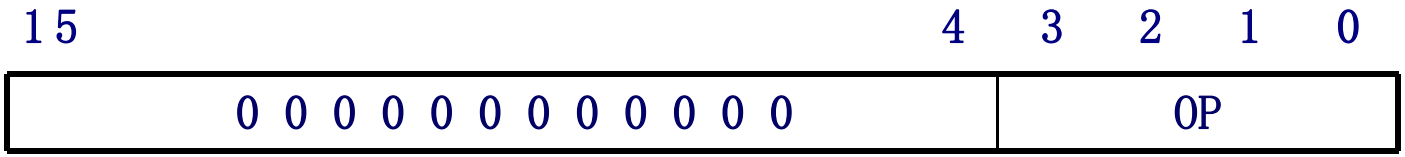
如：PUSH/POP、INC/DEC、NEG、NOT、
MUL、IMUL（带符号乘法指令）、DIV、IDIV、
SHL(SAL)、SHR(SAR)、ROL/ROR、RCL/RCR；

③转移类指令



如：JMP、LOOP、JZ/JNZ、CALL等。

④无操作数指令



如：NOP、RET等

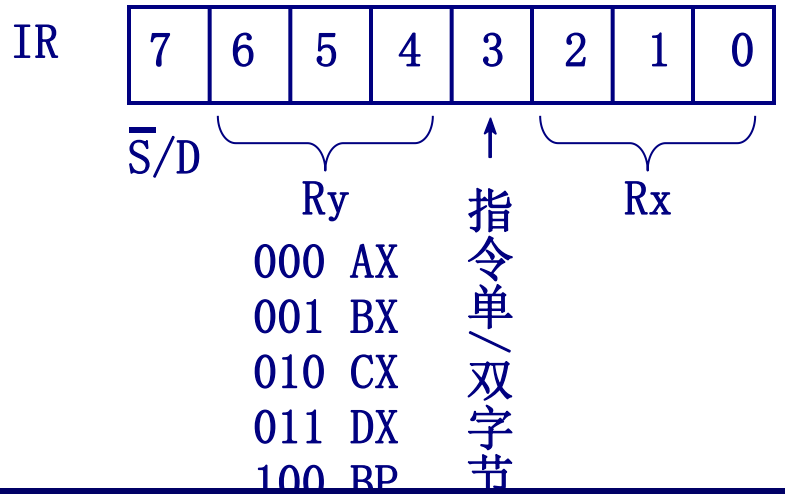
2. 模型机指令系统及寻址方式

(2) 寻址方式及编码

双操作数指令中，必须有一个操作数来自寄存器（寄存器寻址），用Ry表示（IR6-4）。Ry可以是源操作数也可以是目的操作数，由IR7（$\overline{S}$/D位）指明。

双操作数的另一个操作数，其寻址方式由IR3-0指明

2. 模型机指令系统及寻址方式



寻址字段 $IR_{3—0}$				寻址方式及所用寄存器	
0	0	0	0	AX	寄存器寻址
0	0	0	1	BX	
0	0	1	0	CX	
0	0	1	1	DX	
0	1	0	0	[BP]	寄存器间接寻址
0	1	0	1	[BX]	
0	1	1	0	[SI]	

例. 传送指令

```
MOV AX,[BX] ; Ry即AX，是目的操作数
            ; 源操作数是寄存器间址。   1 000 0101
MOV [BX],AX ; Ry即AX，是源操作数
            ; 目的操作数是寄存器间址。0 000 0101
```

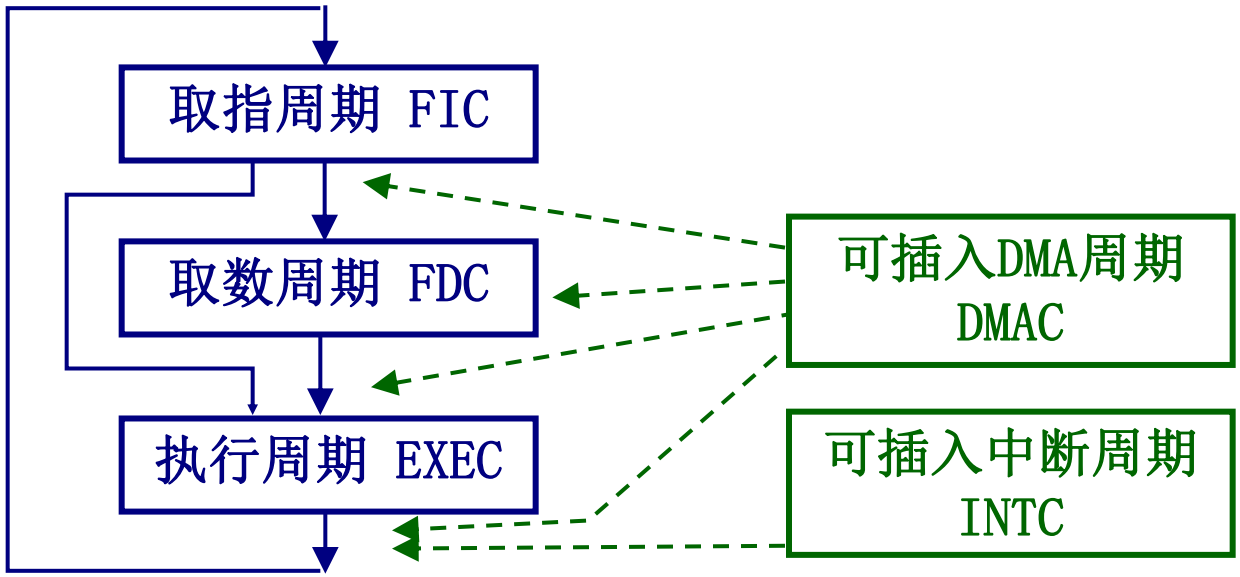
6.4.2 模型机时序系统与控制方式

1. 三级时序

(1) 工作周期

设有5个CPU工作周期。

其中取指FIC、取数FDC、执行EXEC为基本工作周期.



6.4.2 模型机时序系统与控制方式

1. 三级时序

(1) 工作周期

设有5个CPU工作周期。

其中取指FIC、取数FDC、执行EXEC为基本工作周期。

(2) 节拍

一个工作周期最多可有64个节拍，每个节拍等长，变长CPU工作周期。

(3) 节拍脉冲

在每个节拍中有一个对应的节拍脉冲。

6.4.2 模型机时序系统与控制方式

2.时序控制方式

CPU内采用同步控制；

CPU与MM及I/O之间采用准同步。

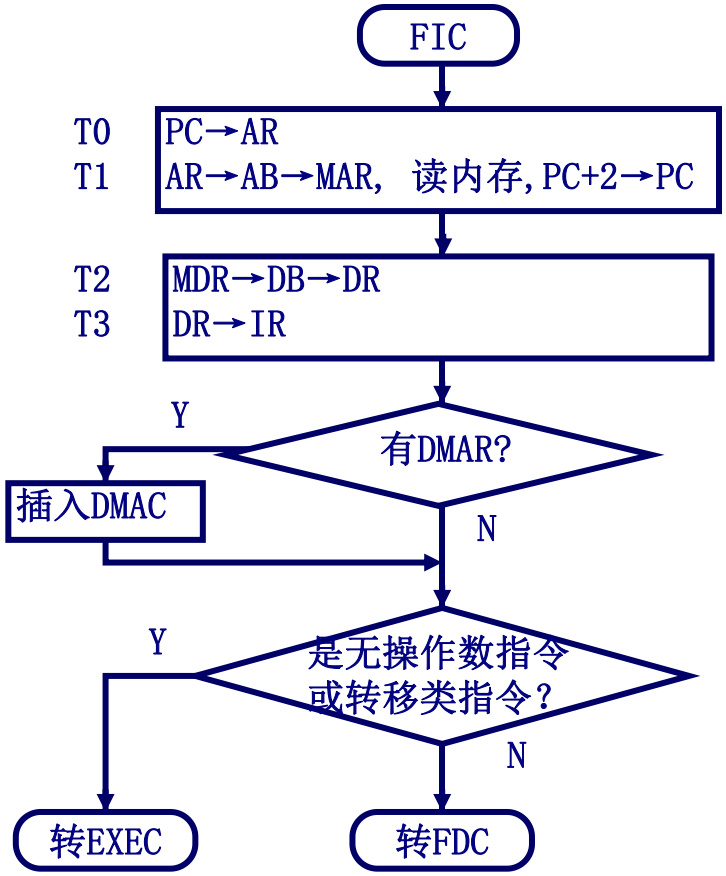
6.4.3 模型机指令的微操作流程

1. 取指周期FIC

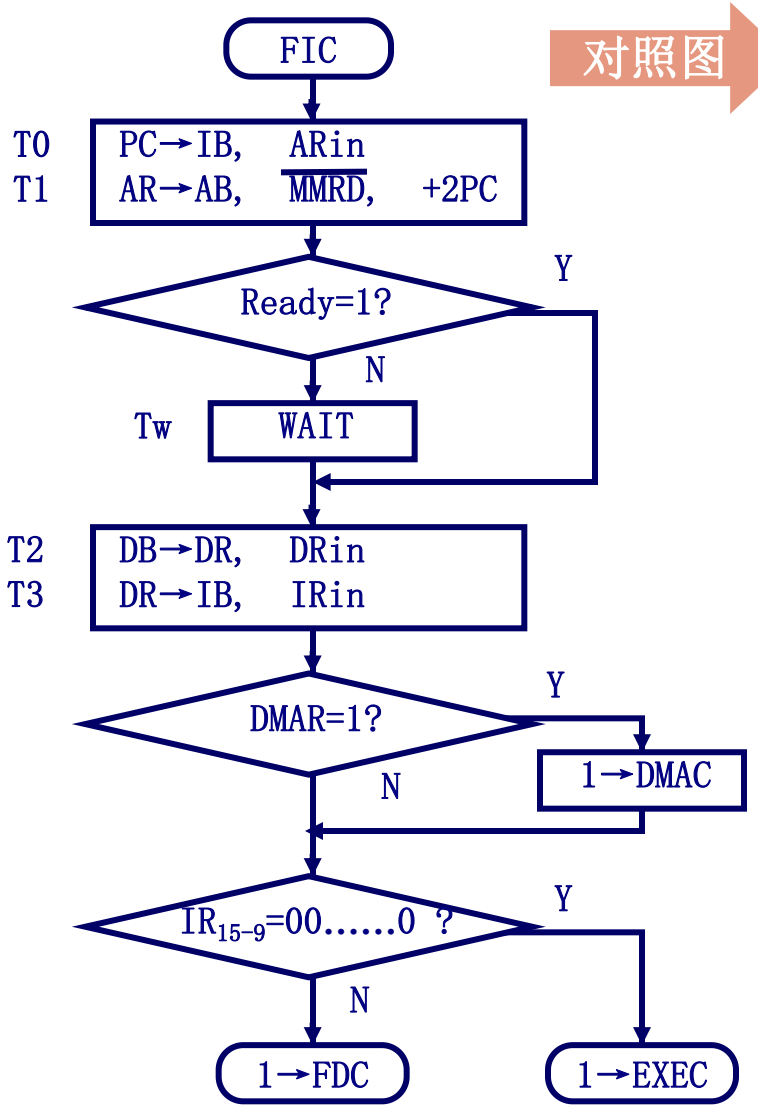
根据PC的值从内存把指令取回到指令寄存器IR中；

修改PC值；

转入下一个CPU周期。


FIC
T0
T1
PC→AR
AR→AB→MAR, 读内存,PC+2→PC
T2
T3
MDR→DB→DR
DR→IR
Y
有DMAR?
插入DMAC
N
Y
是无操作数指令
或转移类指令?
N
转EXEC
转FDC

对照图


FIC
T0
T1
PC→IB, ARin
AR→AB, MMRD, +2PC
Y
Ready=1?
N
Tw
WAIT
T2
T3
DB→DR, DRin
DR→IB, IRin
Y
DMAR=1?
N
1→DMAC
Y
IR15-9=00……0 ?
N
1→FDC
1→EXEC

6.4.3 模型机指令的微操作流程

2. 取操作数周期FDC

把操作数取到DR中；

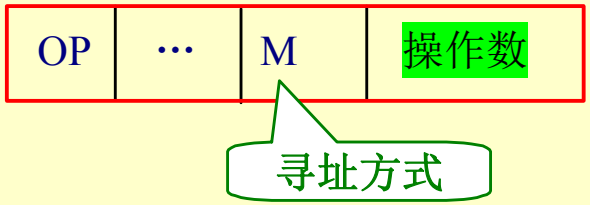
若从内存取数，则内存地址放在AR中；

转到相应的执行周期。

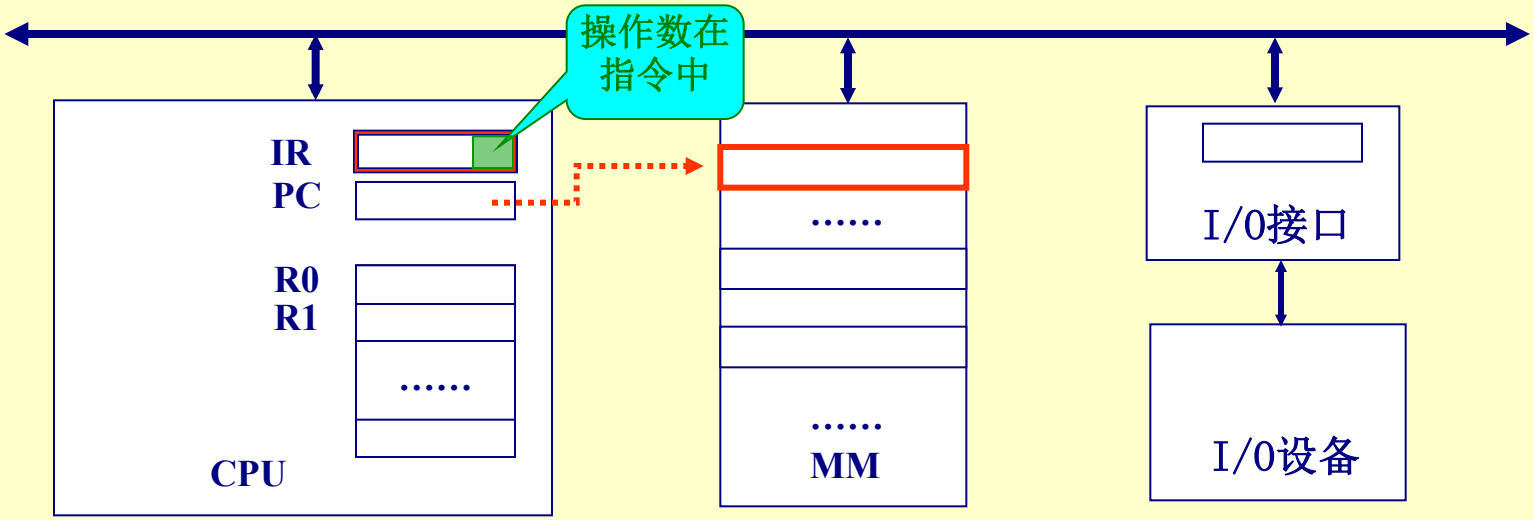
复习：常用的基本寻址方式

(1)立即寻址（Immediate Addressing）

指令中的地址字段存放的就是操作数。



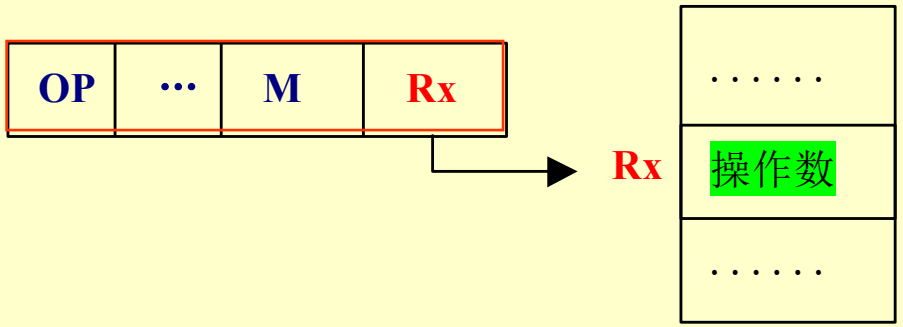
（取指令的同时取出数据）

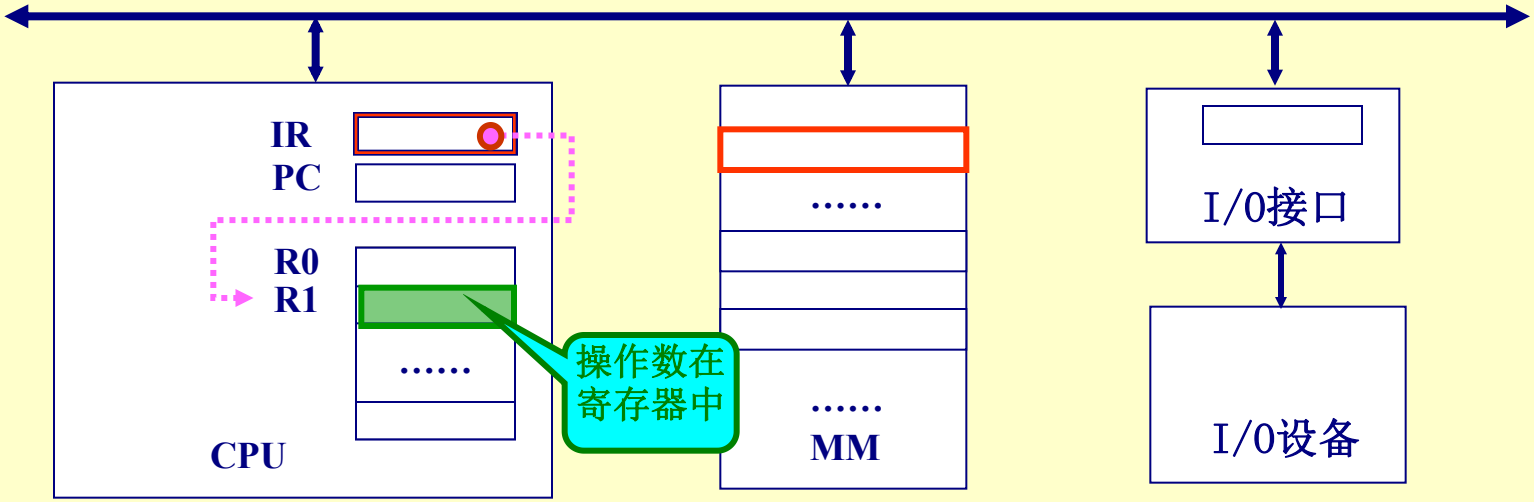


复习：常用的基本寻址方式

(2)寄存器(直接)寻址(Register Addressing)

操作数在某一通用寄存器（在 CPU）中。

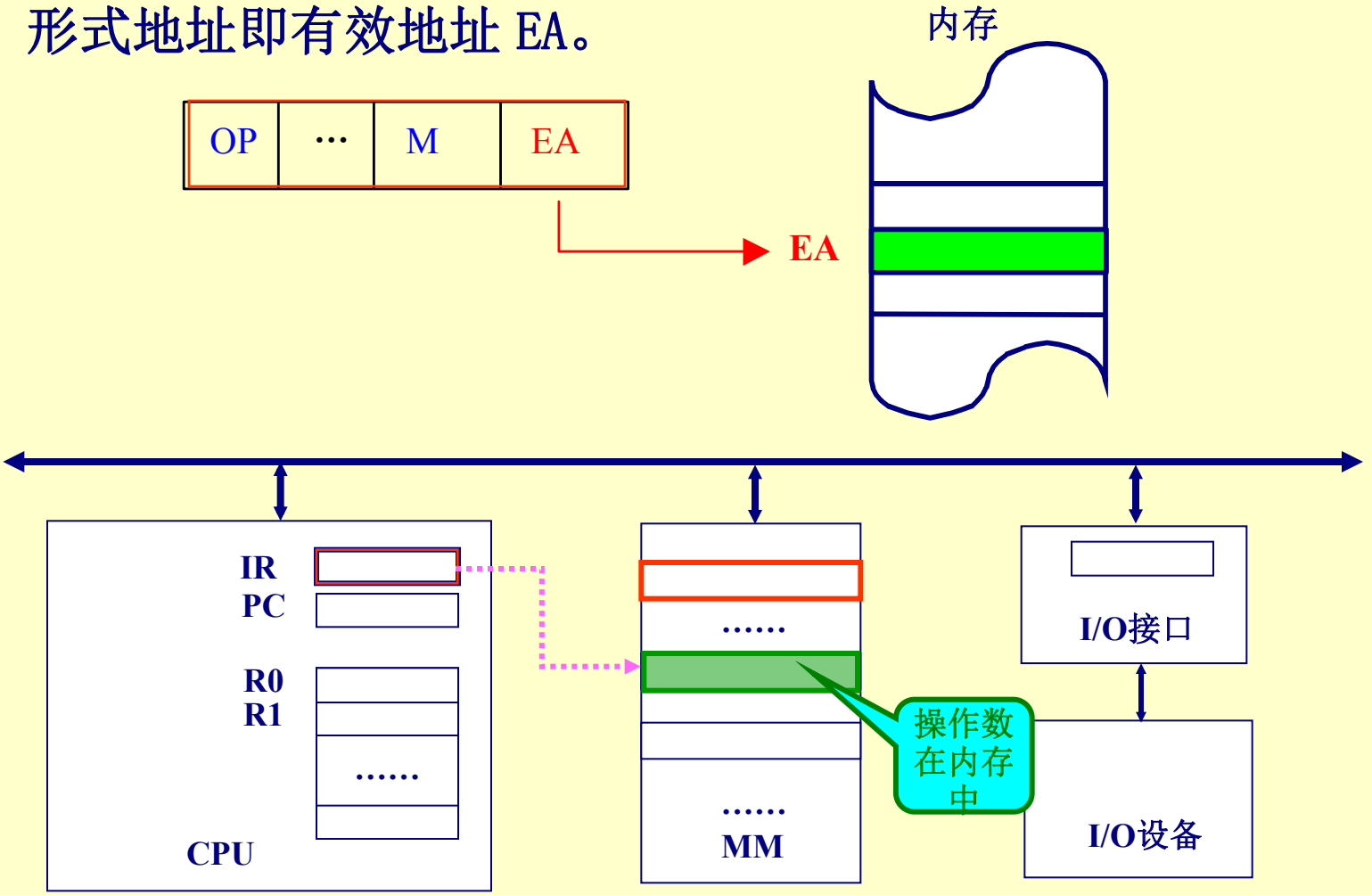




复习：常用的基本寻址方式

(3)直接寻址(Direct Addressing)

形式地址即有效地址 EA。

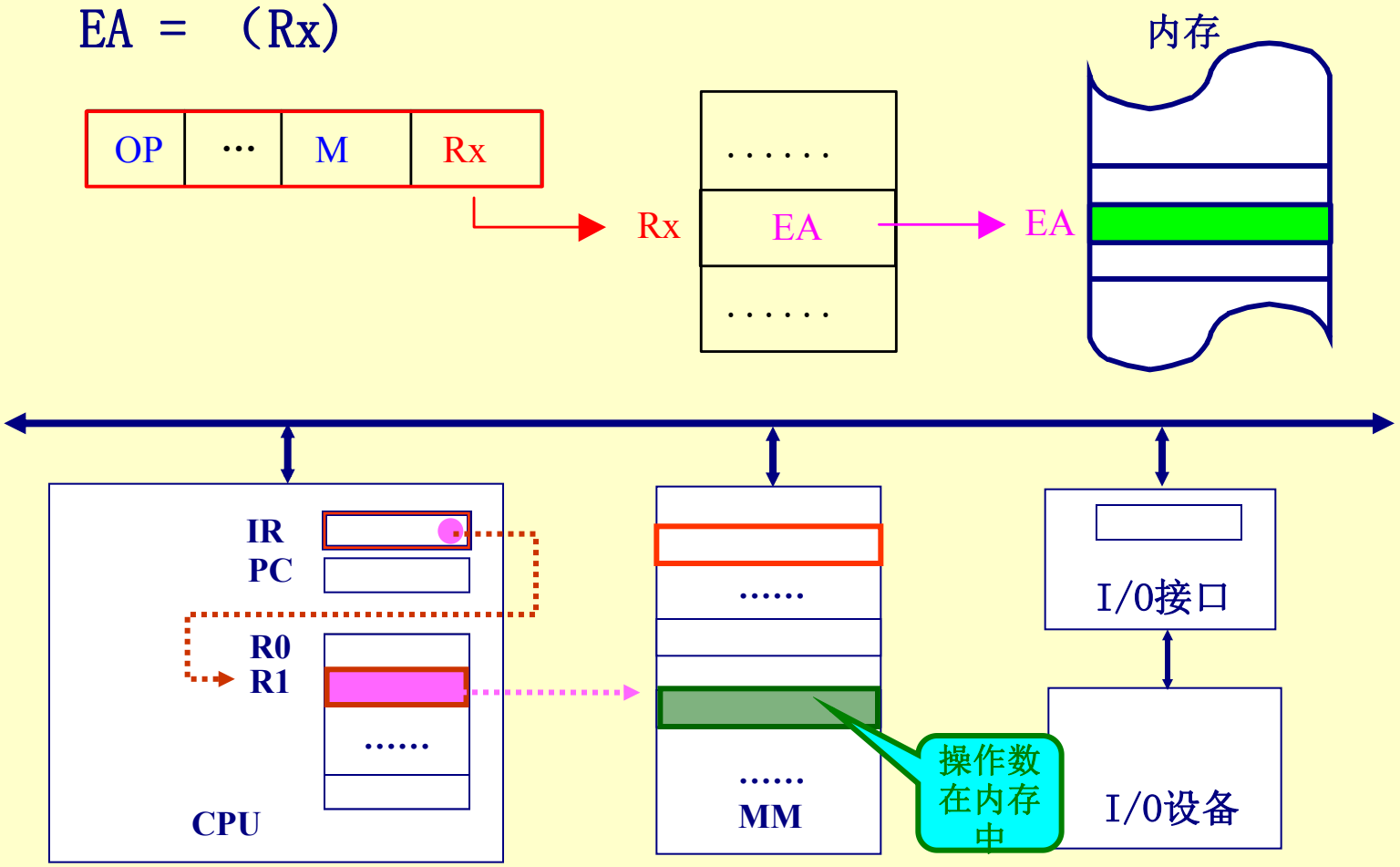


复习：常用的基本寻址方式

(4)寄存器间接寻址(Register Indirect Addressing)

寄存器中存放操作数的有效地址。

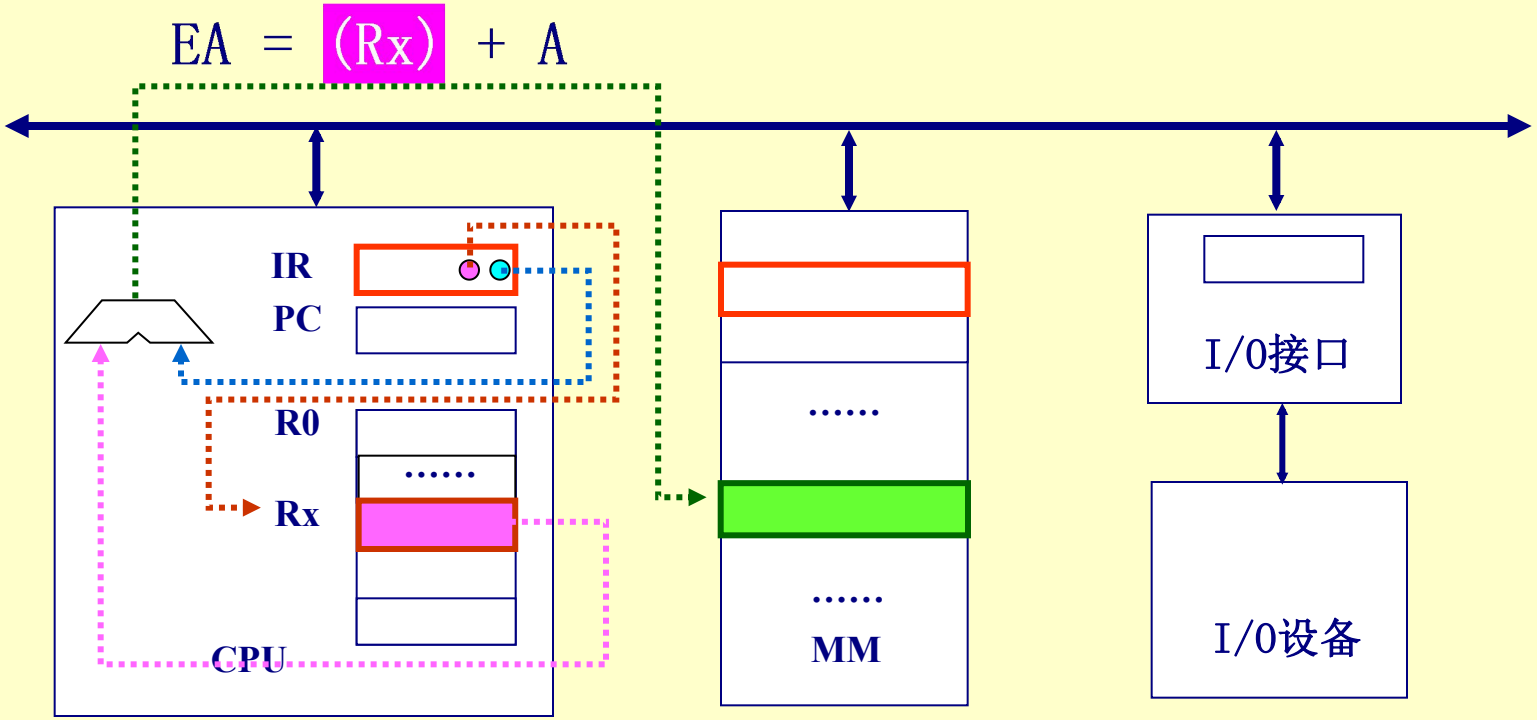
EA = (Rx)



复习：常用的基本寻址方式

(5)变址寻址(Indexed Addressing)

有效地址 = 变址寄存器的内容 + 形式地址(位移量)。



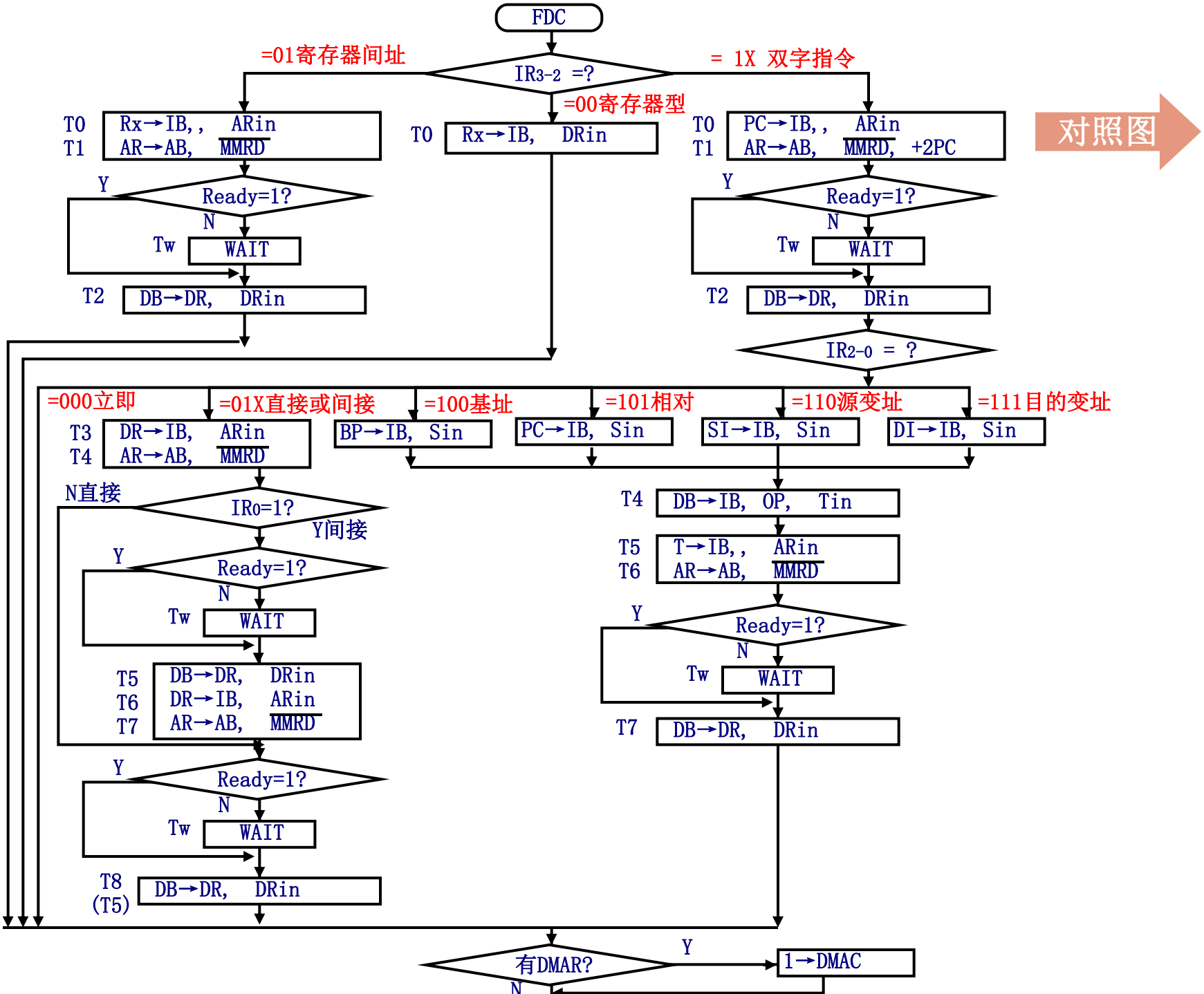
6.4.3 模型机指令的微操作流程

2. 取操作数周期FDC

把操作数取到DR中；

若从内存取数，则内存地址放在AR中；

转到相应的执行周期。

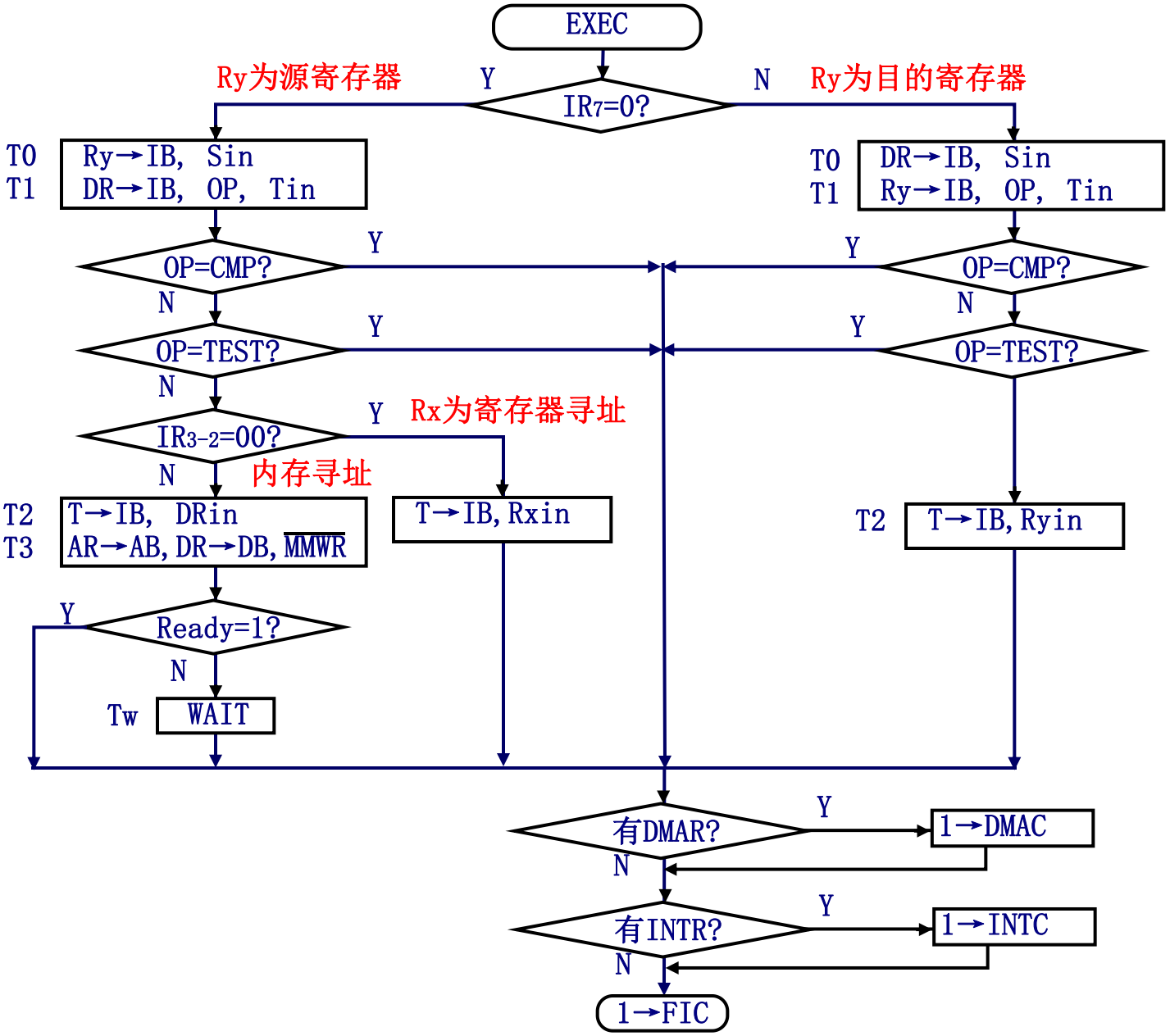

FDC
IR3-2 =?
=01寄存器间址
=00寄存器型
= 1X 双字指令
对照图
T0 Rx→IB, , ARin
T1 AR→AB, MMRD
Y
Ready=1?
N
Tw
WAIT
T2 DB→DR, DRin
T0 Rx→IB, DRin
T0 PC→IB, , ARin
T1 AR→AB, MMRD, +2PC
Y
Ready=1?
N
Tw
WAIT
T2 DB→DR, DRin
IR2-0 = ?
=000立即
=01X直接或间接
=100基址
=101相对
=110源变址
=111目的变址
T3 DR→IB, ARin
T4 AR→AB, MMRD
BP→IB, Sin
PC→IB, Sin
SI→IB, Sin
DI→IB, Sin
N直接
IR0=1?
Y间接
Y
Ready=1?
N
Tw
WAIT
T5 DB→DR, DRin
T6 DR→IB, ARin
T7 AR→AB, MMRD
Y
Ready=1?
N
Tw
WAIT
T8 (T5) DB→DR, DRin
T4 DB→IB, OP, Tin
T5 T→IB, , ARin
T6 AR→AB, MMRD
Y
Ready=1?
N
Tw
WAIT
T7 DB→DR, DRin
有DMAR?
Y
1→DMAC
N

6.4.3 模型机指令的微操作流程

3.执行周期EXEC

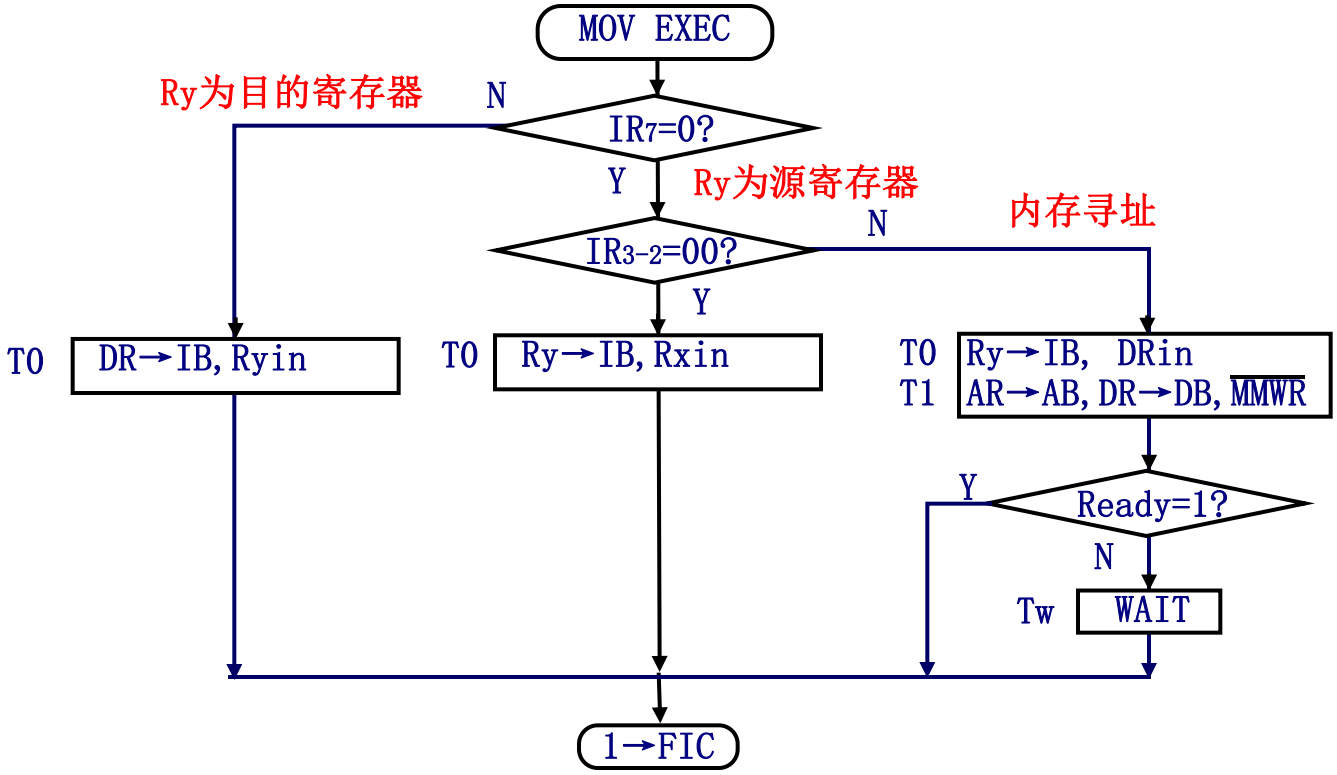
例1.双操作数加法运算指令

```
ADD  AX,Addr      ;  AX←(Addr)+(AX)
ADD  Addr,AX      ;  Addr←(AX)+(Addr)
```


EXEC
Ry为源寄存器
Y
IR7=0?
N
Ry为目的寄存器
T0
T1
Ry→IB, Sin
DR→IB, OP, Tin
T0
T1
DR→IB, Sin
Ry→IB, OP, Tin
Y
OP=CMP?
N
Y
OP=CMP?
N
Y
OP=TEST?
N
Y
OP=TEST?
Y
IR3-2=00?
Rx为寄存器寻址
N
内存寻址
T2
T3
T→IB, DRin
AR→AB, DR→DB, MMWR
T→IB, Rxin
T2
T→IB, Ryin
Y
Ready=1?
N
Tw
WAIT
Y
有DMAR?
N
1→DMAC
Y
有INTR?
N
1→INTC
1→FIC

例2. 传送指令
MOV AX，BX

对照图



6.4.4 组合逻辑控制器的设计

设计步骤：

① 用微操作流程表示各指令的机器周期的处理流程。

② 对指令流程中各微操作分配操作时间（确定机器周期，节拍周期或节拍脉冲）。

③ 在所有指令流程中，找出每一个微操作出现的条件，写出逻辑表达式。

④ 按逻辑表达式构造硬件电路

6.4.4 组合逻辑控制器的设计

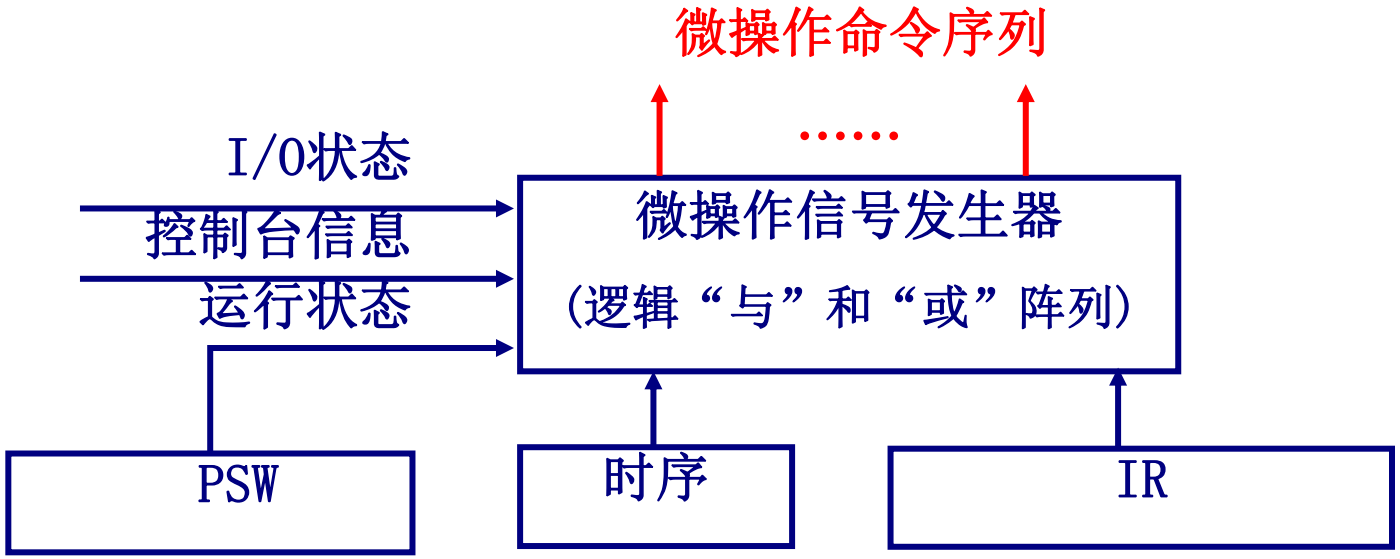
例如：

“PC→IB” = FIC • T0　　（取指周期）
+ FDC • IR3 • T0
+ FDC • (IR3 • IR2 • /IR1 • IR0) • T3　（取数周期）
+ EXEC • (IR15～9=0000000) • (IR3～0=1101) • T0
（执行转移指令）
+ EXEC • (... ...
+ ......

“ARin” = FIC • P0　　（取指周期）
+ FDC • (IR3, 2=01) • P0　（取数周期）
+ FDC • (IR3=1)P0
+ FDC • IR2, 1=01) • P3
+ ......

输入：IR的信号、时序信号、状态字寄存器PSW等；

输出：微操作控制信号。



§6.5 微程序控制原理

微程序设计技术的实质是将程序设计技术和存储技术相结合，即用程序设计的思想方法来组织操作控制逻辑，将微操作控制信号按一定规则进行信息编码（代码化），形成控制字（微指令），再把这些微指令按时间先后排列起来构成微程序，存放在一个只读的控制存储器中。

6.5.1 微程序控制的基本组成

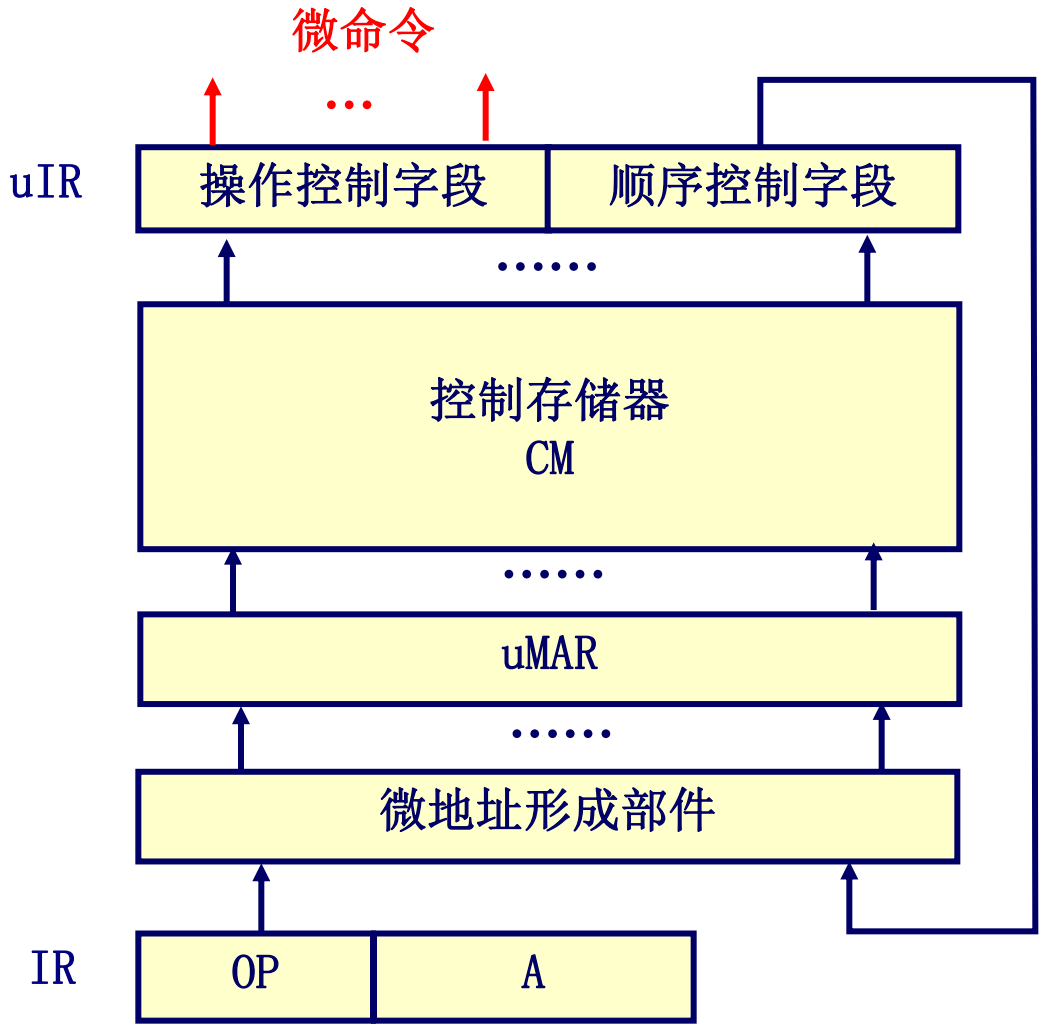
① 控制存储器（CM）

② 微指令寄存器（μIR）

③ 微地址形成部件

④ 微地址寄存器（μMAR）

6.5.1 微程序控制的基本组成



6.5.2 微程序控制的基本概念

1. 微命令和微操作

微命令是控制计算机某个部件完成某个基本微操作的命令。微命令和微操作是一一对应的。

2. 微指令、微地址

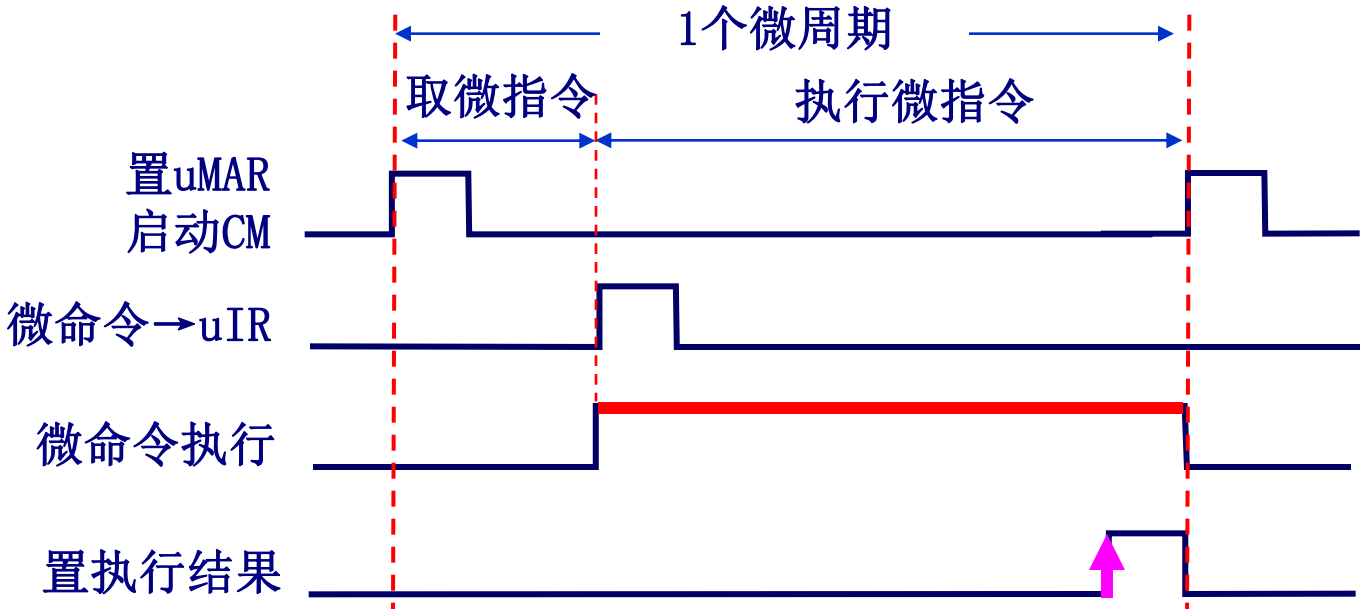
把一个节拍内所完成的微操作集合起来，用二进制的编码方式表示，形成一条微指令。

存放微指令的控制存储器的单元地址就称为微地址。

6.5.2 微程序控制的基本概念

3. 微周期

从控制存储器中读取一条微指令并执行相应的微命令所需的全部时间称为微周期。



6.5.2 微程序控制的基本概念

4. 微程序

一系列微指令的有序集合就是微程序。每一条机器指令都对应一个微程序。

微程序存放在控制存储器CM中。

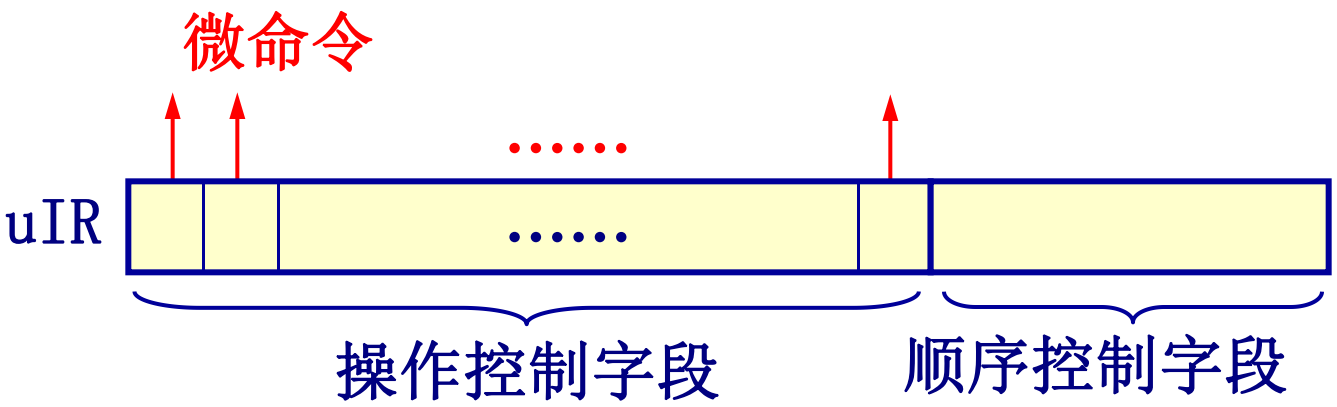
6.5.3 微指令编码法

1. 直接控制法（不译码法）

指令操作控制字段中的各位分别可以直接控制计算机的部件。

特点：结构简单，并行性强，操作速度快；

但是微指令字太长。



6.5.3 微指令编码法

2. 最短编码法

这种方法将所有的微命令统一编码，每条微指令只定义一个微命令。若微命令的总数为N，操作控制字段的长度为L，则最短编码法应满足下列关系式：

$L \geqslant \log_2 N$

特点：微指令字长最短，译码器复杂，不能充分利用机器硬件所具有的并行性。

6.5.3 微指令编码法

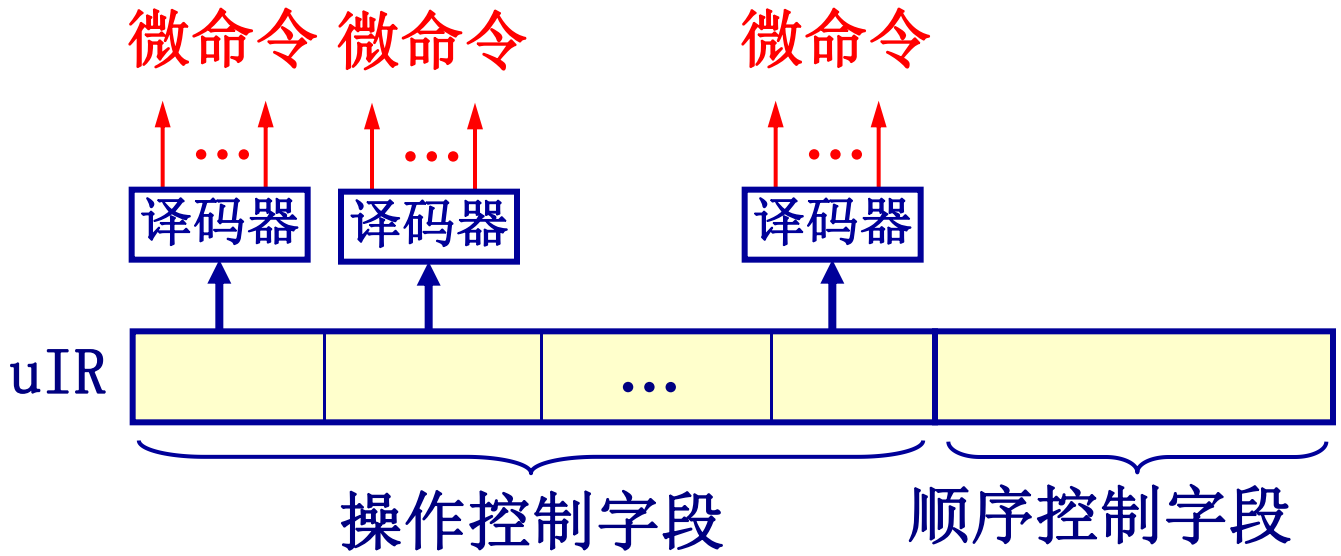
3. 字段编码法

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。

(1) 字段直接编码法

各字段都可以独立地定义本字段的微命令，而和其他字段无关。又称为显式编码或单重定义编码方法。

字段直接编码法

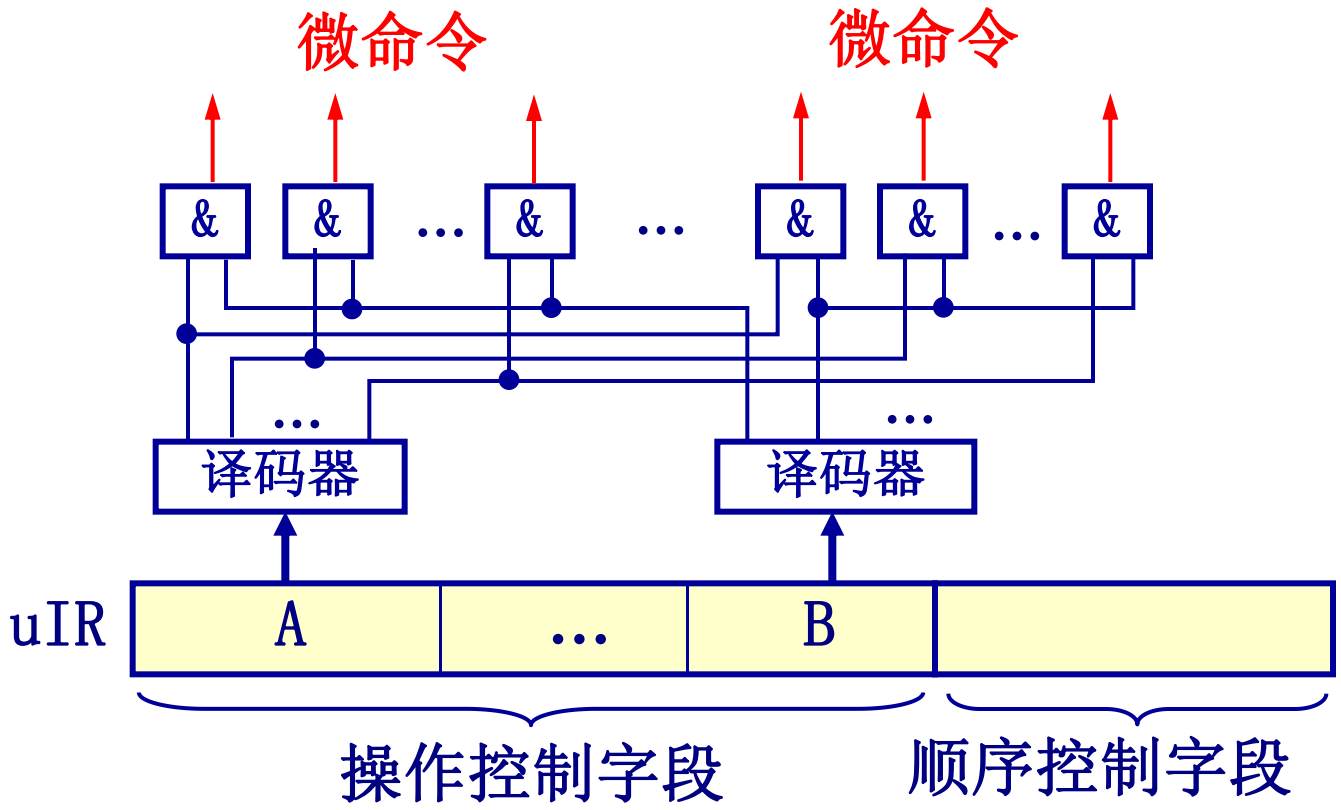


6.5.3 微指令编码法

(2) 字段间接编码法

字段间接编码法是在字段直接编码法的基础上，用来进一步缩短微指令字长的方法。间接编码的含义是，一个字段的某些编码不能独立地定义某些微命令，而需要与其他字段的编码来联合定义，因此又称为隐式编码或多重定义编码方法。

字段间接编码法



6.5.3 微指令编码法

3. 字段编码法中操作控制字段的分段原则

①把互斥性的微命令分在同一段内，相容性的微命令分在不同段内。

②应与数据通路结构相适应。

③每个小段中包含的信息位不能太多，否则将增加译码线路的复杂性和译码时间。

④一般每个小段还要留出一个状态，表示本字段不发出任何微命令。

6.5.4 微程序入口地址的形成

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个微程序的入口地址（初始微地址）。主要方式有：

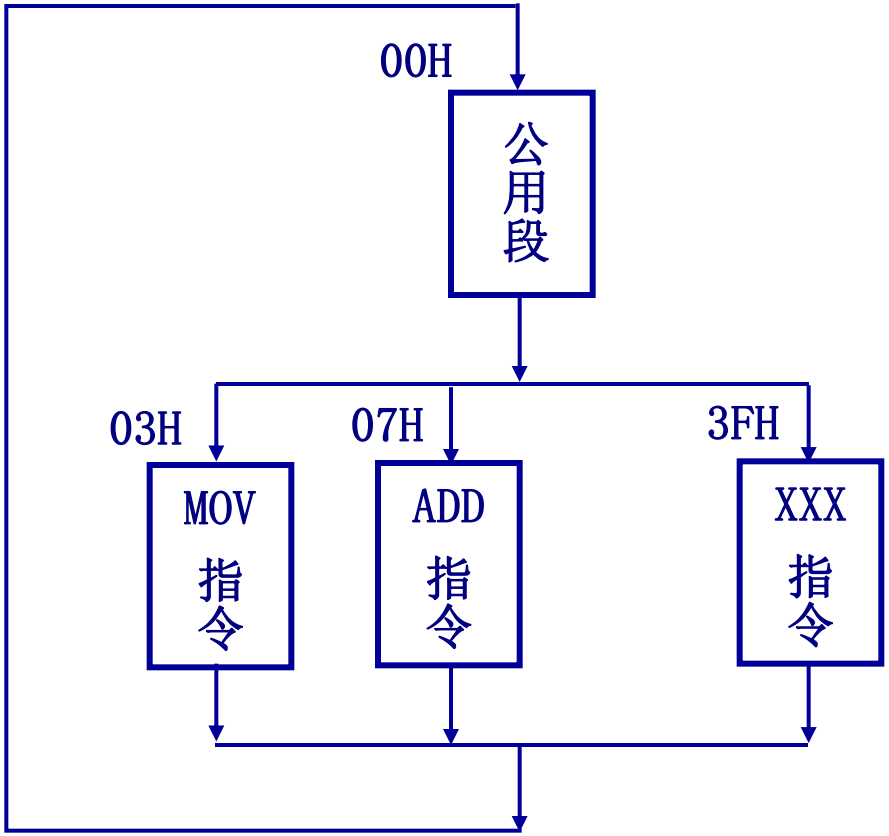
1)一级功能转换

直接使操作码与入口地址码的部分位相对应。也可用PLA 电路实现。

2)二级功能转换

第一次先按指令类型标志转移，以区分出指令属于哪一类，第二次即可按操作码区分出具体是哪条指令，找出相应微程序的入口微地址。

6.5.4 微程序入口地址的形成



一级功能转移示意图

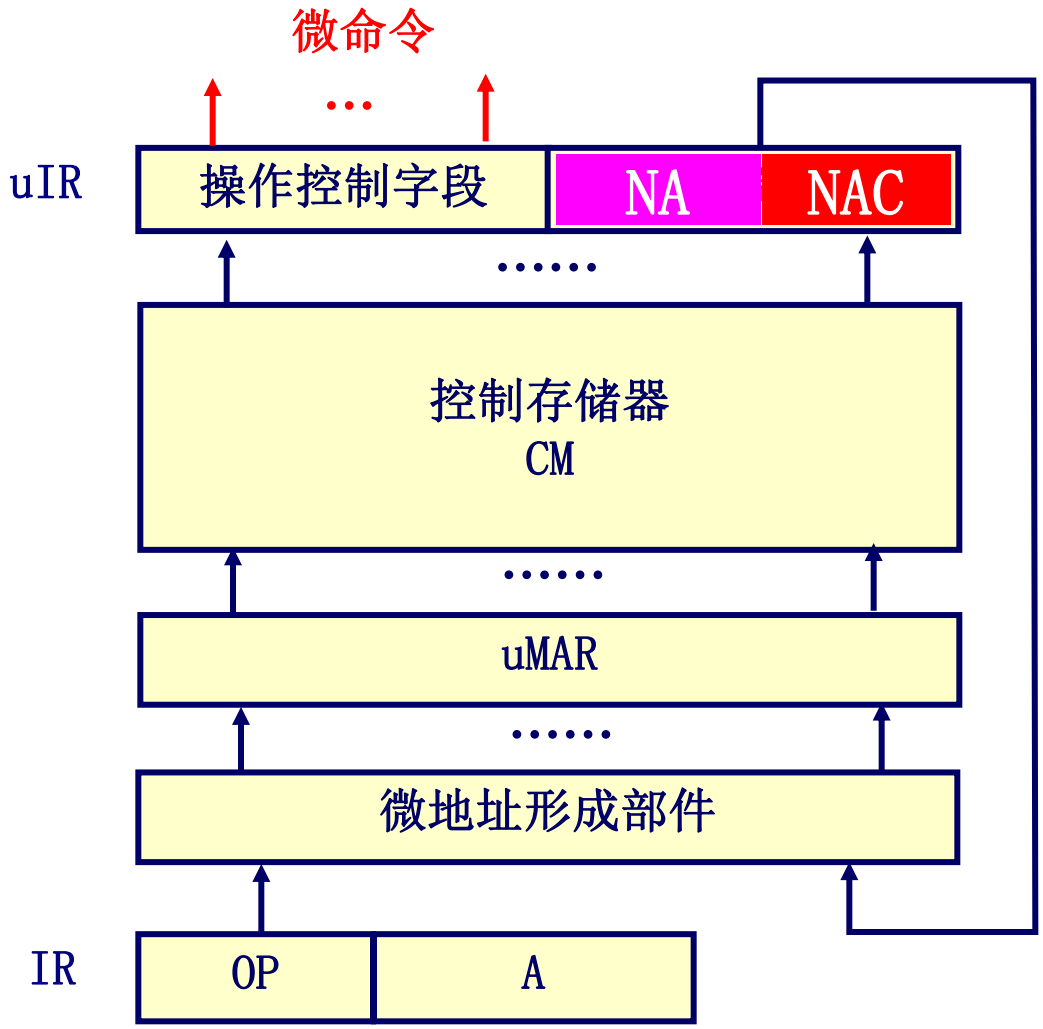
6.5.5 后继微地址的形成

微程序转移频繁，所以微指令中用次地址部分来指明下一条要执行的微指令的地址。

断定法是最常用的后继微地址形成方法，它把微指令的次地址部分又分为两个部分：

(1) NAC：次地址控制字段
（指示下一微地址的产生方式）；

(2) NA：次地址字段
（微程序转移时的后继微地址）。


微命令
…
uIR
操作控制字段
NA
NAC
……
控制存储器
CM
……
uMAR
……
微地址形成部件
IR
OP
A

例如，根据NAC的编码可以按如下方式中的一种来确定后继微地址：

1）顺序方式—— uAR递增。

2）无条件转移——uMAR←NA

3）条件转移——uMAR←uMAR+1，或 uMAR←NA

4）多分支转移

例如根据机器指令的操作码或寻址方式进行分支。

5）微地址寄存器首地址的产生

从取指入口开始，CPU被RESET时，uMAR也指向该入口。

6.5.6 模型机微程序控制器举例

对照图

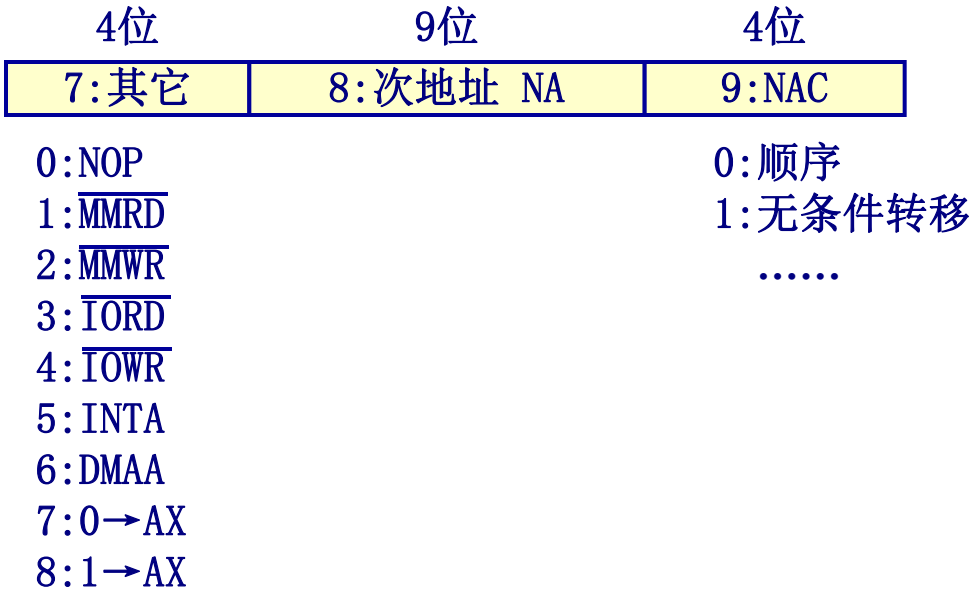
1. 微指令格式

4位	4位	2位	2位	5位	4位
1:XX→IB	2:XXin	3:DR	4:AR	5:算逻操作	6:计数
0:NOP	0:NOP	0:NOP	0:NOP	00:NOP	0:NOP
1:AX→IB	1:AXin	1:DR→DB	1:AR→AB	01:ADD	1:+2DI
2:BX→IB	2:BXin	2:DB→DR	2:ARin	02:ADC	2:-2DI
3:CX→IB	3:CXin		3:DRin	03:SUB	3:+2SP
4:DX→IB	4:DXin			04:SUBB	4:-2SP
5:SI→IB	5:SIin			05:AND	5:+2PC
6:DI→IB	6:DIin			06:OR	6:0→PC
7:BP→IB	7:BPin			07:XOR	7:-1CT
8:SP→IB	8:SPin			08:SAL	8:+1CT
9:S→IB	9:Sin			09:SAR	9:0→CT
A:T→IB	A:Tin			0A:SHR	
B:PC→IB	B:PCin			0B:ROL	
C:PSW→IB	C:PSWin			0C:ROR	
D:DR→IB	D:DRin			0D:RCL	
E:Rx→IB	E:Rxin			0E:RCR	
F:Ry→IB	F:Ryin			.....	

6.5.6 模型机微程序控制器举例

对照图

1. 微指令格式



6.5.6 模型机微程序控制器举例

对照图

1. 微指令格式

例如：

“PC→IB，ARin”微操作，其微指令的前面部分为：

4位	4位	2位	2位	5位	…
B	0	0	2	00	…

0:NOP	0:NOP	0:NOP	0:NOP	00:NOP
1:AX→IB	1:AXin	1:DR→DB	1:AR→AB	01:ADD
……	……	……	2:ARin	02:ADC
B:PC→IB			3:DRin	……
……				

§6.6 流水线技术

6.6.1 并行处理技术

1. 并行处理的几种方式

(1)时间并行（重叠）

让多个处理过程在时间上相互错开，轮流使用同一套硬件设备的各个部件，以加快硬件周转而赢得速度。

流水线是典型的时间并行处理技术，它通过采用流水处理部件来实现。

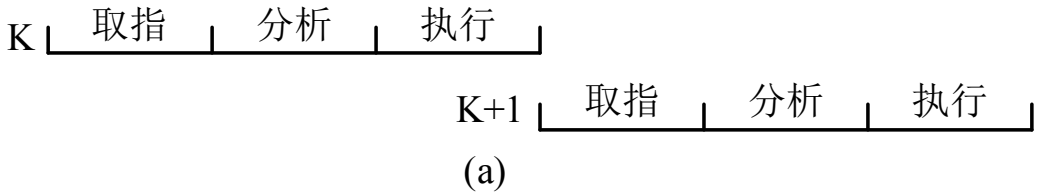
6.6.1 并行处理技术

1. 并行处理的几种方式

(1)时间并行（重叠）

让多个处理过程在时间上相互错开，轮流使用同一套硬件设备的各个部件，以加快硬件周转而赢得速度。

顺序执行方式：

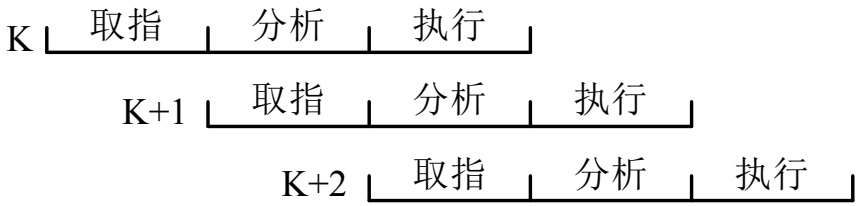


(a)

一次重叠执行方式：

(b)

二次重叠执行方式：



(c)

6.6.1 并行处理技术

1.并行处理的几种方式

(1)时间并行（重叠）

让多个处理过程在时间上相互错开，轮流使用同一套硬件设备的各个部件，以加快硬件周转而赢得速度。

(2)空间并行（资源重复）

以重复的硬件部件为基础同时进行处理。例如，设置多个运算部件来支持向量的运算。

例如：SIMD（单指令流多数据流）计算机。

(3)时间并行+空间并行

例如，Pentium中采用了超标量流水线技术

6.6.1 并行处理技术

2.流水CPU的结构

流水计算机的系统组成：

(1)存储体系

主存采用多体交叉存储器

Cache.

(2)流水方式CPU

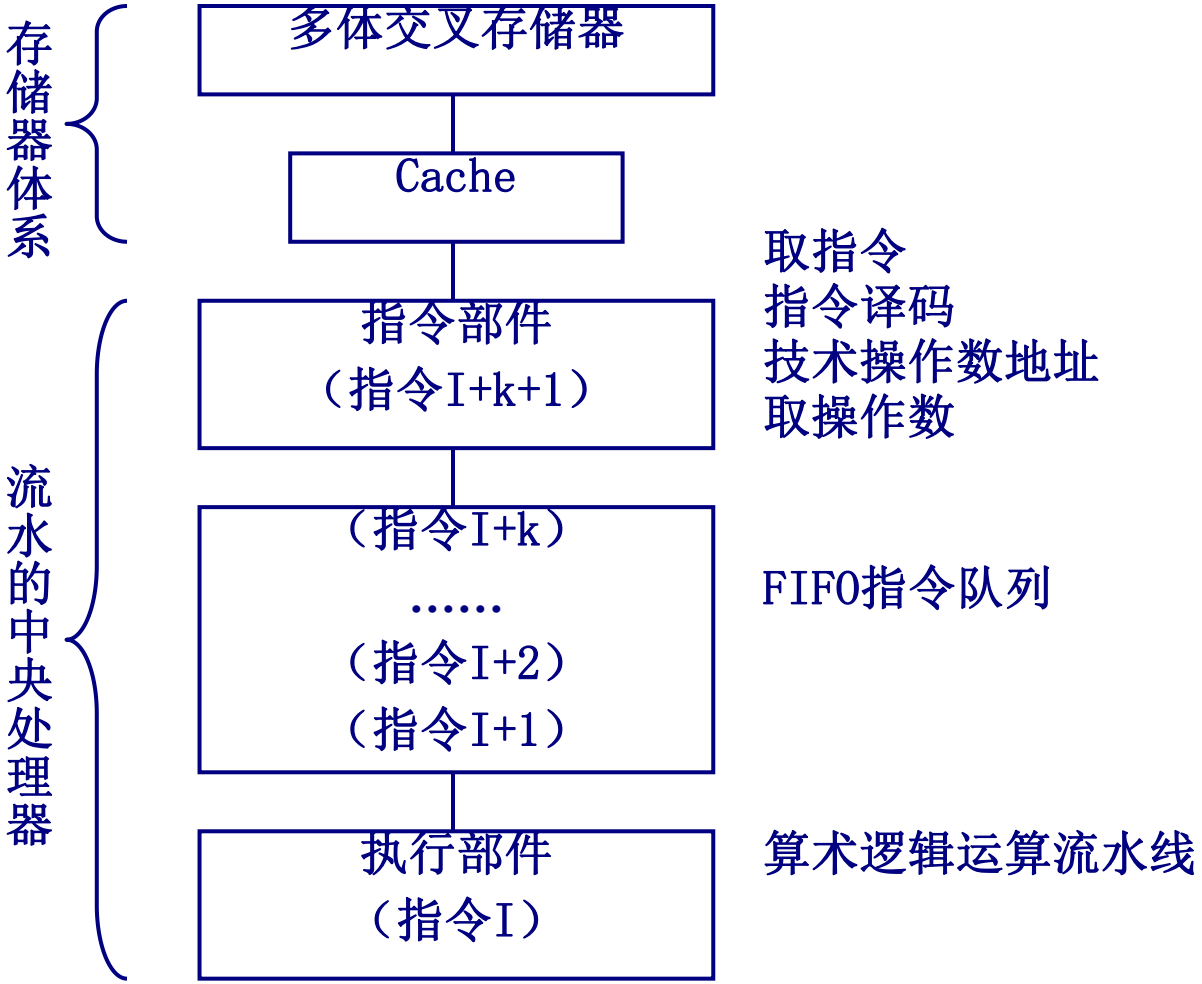
指令流水线

指令队列：FIFO

执行部件：可以有多个采用流水线方式构成的算术逻辑部件构成，可以将定点运算部件和浮点运算部件分开。

6.6.1 并行处理技术

2. 流水CPU的结构



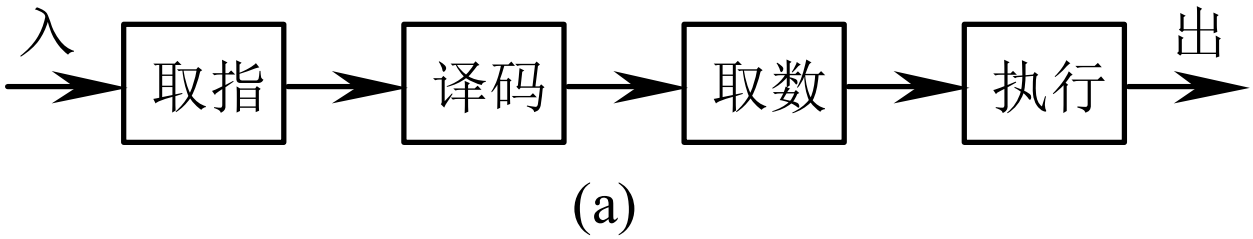
6.6.2 流水工作原理

流水处理技术是在重叠、先行控制方式的基础上发展起来的，它基于重叠的原理，但却是在更高程度上的重叠。

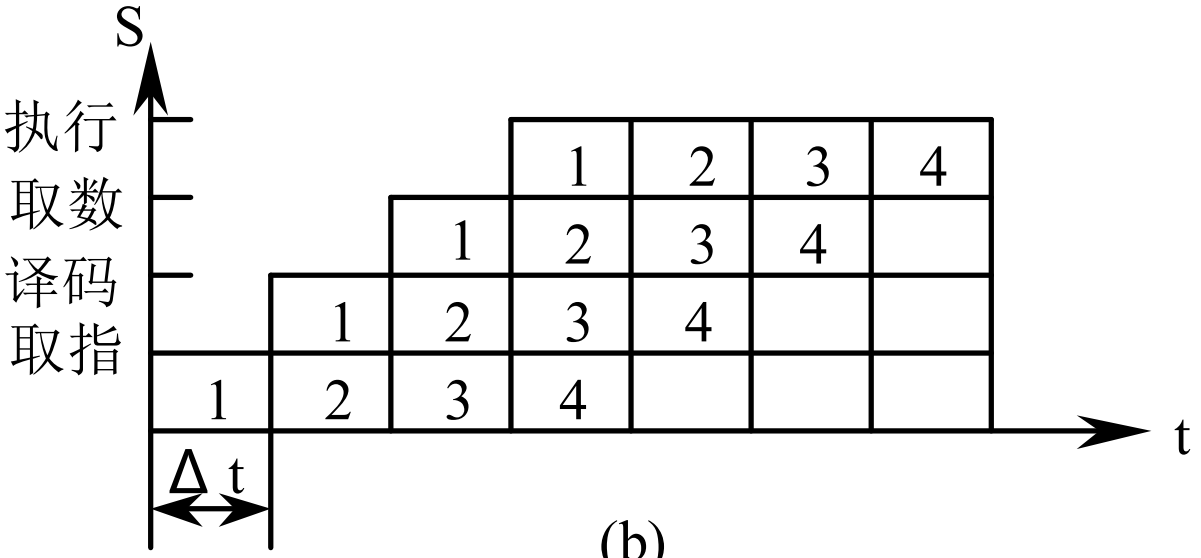
1. 流水线

流水线是将一个较复杂的处理过程分成m个复杂程度相当、处理时间大致相等的子过程，每个子过程由一个独立的功能部件来完成，处理对象在各子过程连成的线路上连续流动。在同一时间，m个部件同时进行不同的操作，完成对不同子过程的处理。

四个子过程的流水线处理：



(a)



(b)

2. 流水线分类

按处理级别分类：

操作部件级 / 指令级/处理机级

按功能分类：

单功能 / 多功能（可有多种连接方式）

按工作方式分类：

静态流水线 / 动态流水线

（允许同一时间内连成不同的功能）

按流水线结构分类：

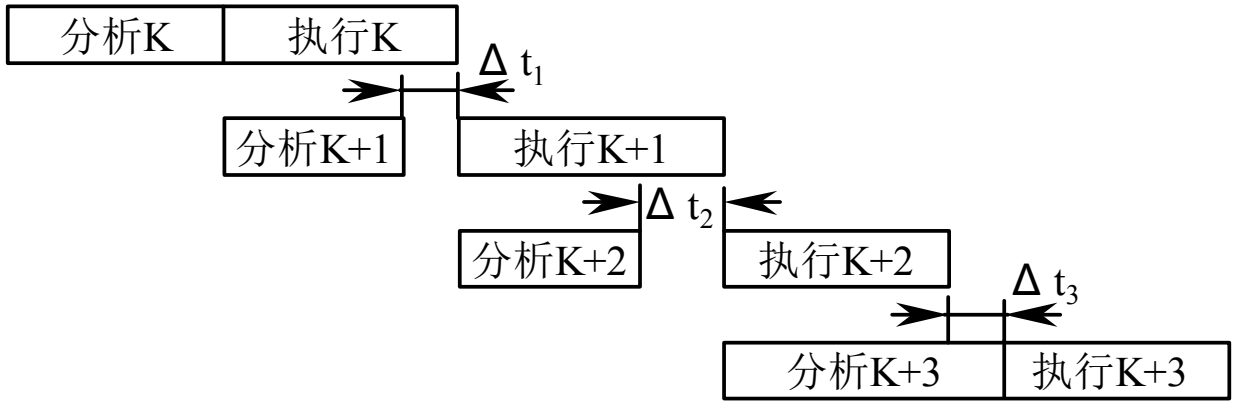
线性流水 / 非线性流水（含反馈回路）

3. 先行控制方式

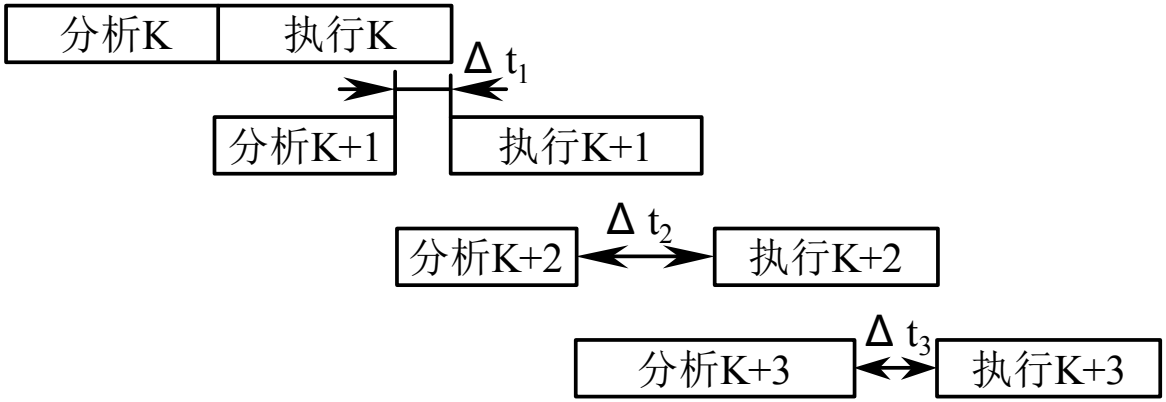
先行控制的主要目的是使各阶段的专用控制部件尽可能不间断地工作，以提高设备利用率及执行速度。

3. 先行控制方式

“分析”和“执行”时间不等的重叠：



先行控制方式：



6.6.3 流水线的性能

1. 吞吐率

指单位时间内流水线所完成指令或输出结果的数量。

最大吞吐率、实际吞吐率。

2. 加速比 Sp

指m 段的流水线的速度与等功能的非流水线的速度之比。

3. 效率

指流水线中各功能段的利用率。

由于流水线有建立时间和排空时间，因此各功能段的设备不可能一直处于工作状态 。

6.6.4 影响指令流水性能的因素

(1)流水线上每个阶段的执行时间不一样

解决办法：采用先行控制方式

(2)条件转移的影响

解决办法：猜测法预取。

(3)相近指令之间出现某种关联

1)结构相关

不同指令争用同一功能部件产生的资源冲突。

解决办法：停顿

采用哈弗结构的存储器

指令预取（适用于访存周期短的情况）

6.6.4 影响指令流水性能的因素

(1)流水线上每个阶段的执行时间不一样

解决办法：采用先行控制方式

(2)条件转移的影响

解决办法：猜测法预取。

(3)相近指令之间出现某种关联

2)数据相关

不同指令因重叠操作，可能改变操作数的读写访问顺序。

分为：写后读相关（RAW）、

读后写相关（WAR）、

写后写相关（WAW）

6.6.4 影响指令流水性能的因素

(1)流水线上每个阶段的执行时间不一样

解决办法：采用先行控制方式

(2)条件转移的影响

解决办法：猜测法预取。

(3)相近指令之间出现某种关联

3)控制相关

由转移指令引起。

§6.7 微处理器中的新技术（教材P230，自学）

1．超标量和超流水线技术

通过重复设置多个功能部件，使得计算机在每个时钟周期里可以解释多条指令。

2．EPIC的指令级并行处理

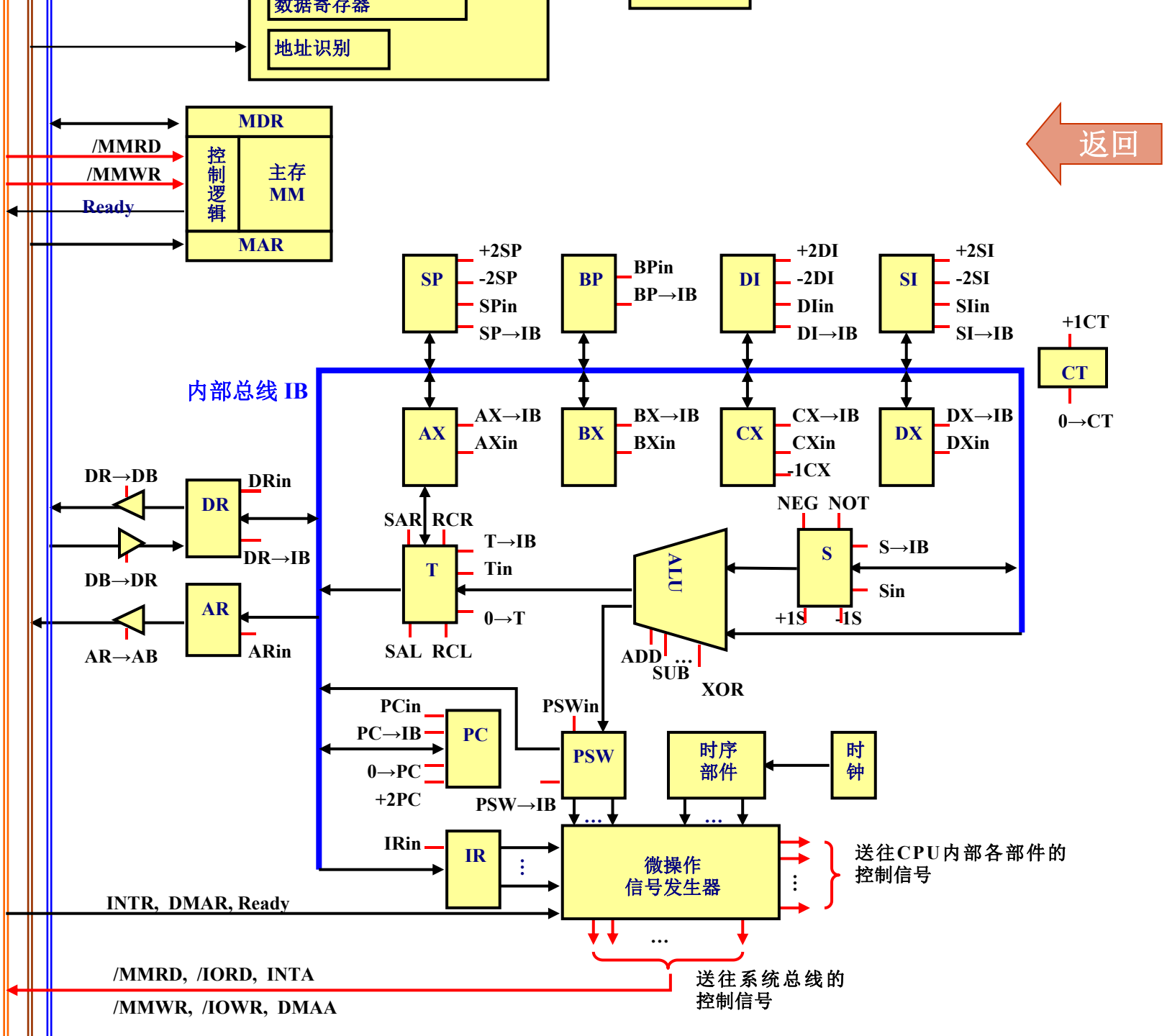
由编译器分析指令之间的依赖关系，将无依赖关系的3条指令组合成一个指令束。

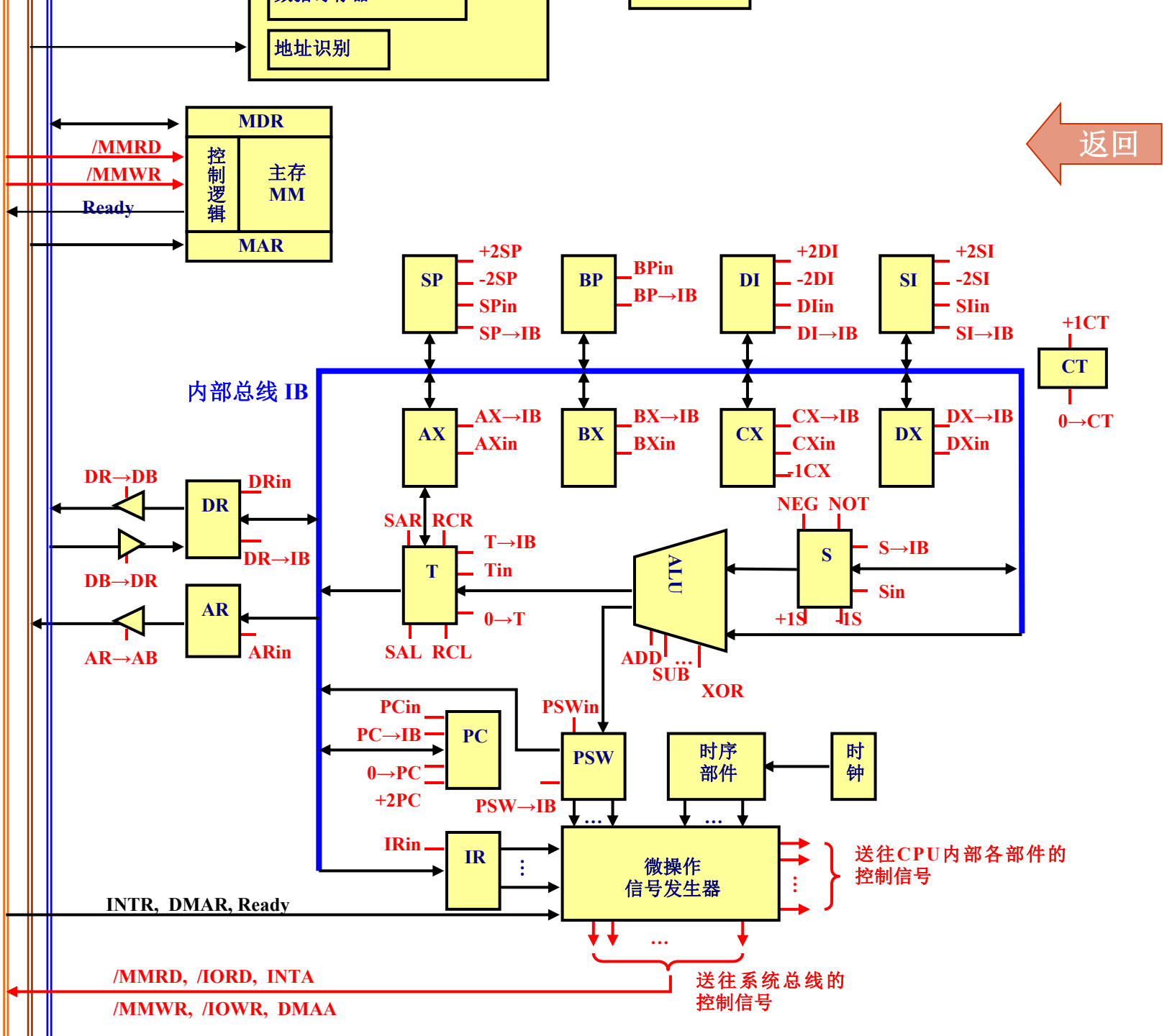
3．超线程技术

将一个物理CPU当作多个逻辑CPU使用，同时执行多重线程，以提高各部件使用效率。

4．双核与多核技术

在一个CPU芯片上集成多个CPU核。


数据寄存器
地址识别
MDR
/MMRD
/MMWR
Ready
控制逻辑
主存 MM
MAR
返回
SP
+2SP
-2SP
SPin
SP→IB
BP
BPin
BP→IB
DI
+2DI
-2DI
DIin
DI→IB
SI
+2SI
-2SI
SIin
SI→IB
+1CT
CT
0→CT
内部总线 IB
AX
AX→IB
AXin
BX
BX→IB
BXin
CX
CX→IB
CXin
-1CX
DX
DX→IB
DXin
DR→DB
DR
DRin
DR→IB
DB→DR
AR
ARin
AR→AB
SAR RCR
T
T→IB
Tin
0→T
SAL RCL
ALU
NEG NOT
S
S→IB
Sin
+1S -1S
ADD … SUB
XOR
PCin
PC→IB
PC
0→PC
+2PC
PSWin
PSW
PSW→IB
时序部件
时钟
IRin
IR
微操作信号发生器
送往CPU内部各部件的控制信号
INTR, DMAR, Ready
/MMRD, /IORD, INTA
/MMWR, /IOWR, DMAA
送往系统总线的控制信号


地址识别
MDR
/MMRD
/MMWR
Ready
控制逻辑
主存 MM
MAR
返回
SP
+2SP
-2SP
SPin
SP→IB
BP
BPin
BP→IB
DI
+2DI
-2DI
DIin
DI→IB
SI
+2SI
-2SI
SIin
SI→IB
+1CT
CT
0→CT
内部总线 IB
AX
AX→IB
AXin
BX
BX→IB
BXin
CX
CX→IB
CXin
-1CX
DX
DX→IB
DXin
DR→DB
DR
DRin
DR→IB
DB→DR
AR
ARin
AR→AB
SAR RCR
T
T→IB
Tin
0→T
SAL RCL
ALU
NEG NOT
S
S→IB
Sin
+1S -1S
ADD … SUB
XOR
PCin
PC→IB
PC
0→PC
+2PC
PSWin
PSW
PSW→IB
时序部件
时钟
IRin
IR
微操作信号发生器
送往CPU内部各部件的控制信号
INTR, DMAR, Ready
/MMRD, /IORD, INTA
/MMWR, /IOWR, DMAA
送往系统总线的控制信号

思考题：P233 1-5，8，9

P209 6,7，13，14，15，19，20

习题： P233 1-5，8，9

P209 6,7，13，14，15，19，20

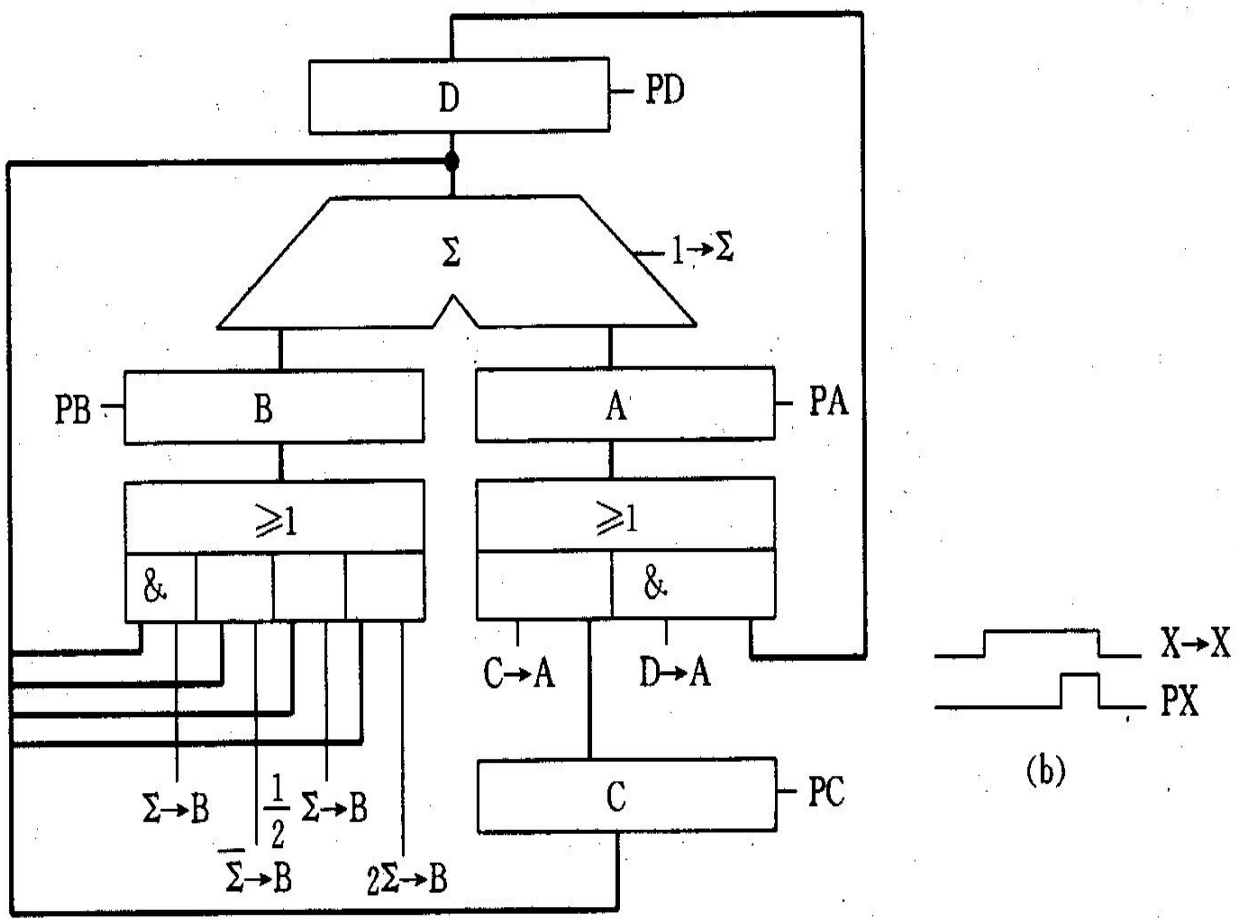
补充：

补充6-1.

下图为某模型机数据加工通路。其中∑为并行加法器，I→∑为最低位的进位。A，B，C，D为四个寄存器，PA，PB，PC，PD分别为四个寄存器的数据接收信号，且均为脉冲信号。图中其余的X→X控制信号均为电平信号，电平信号与脉冲信号的时间关系如下图(b)所示。试拟出在该图上实现下列运算所需的微操作。

(1) D - C → D
(2) D/2 + C → D
(3) D - 1 → D
(4) 2C + I → C

补充6-1.

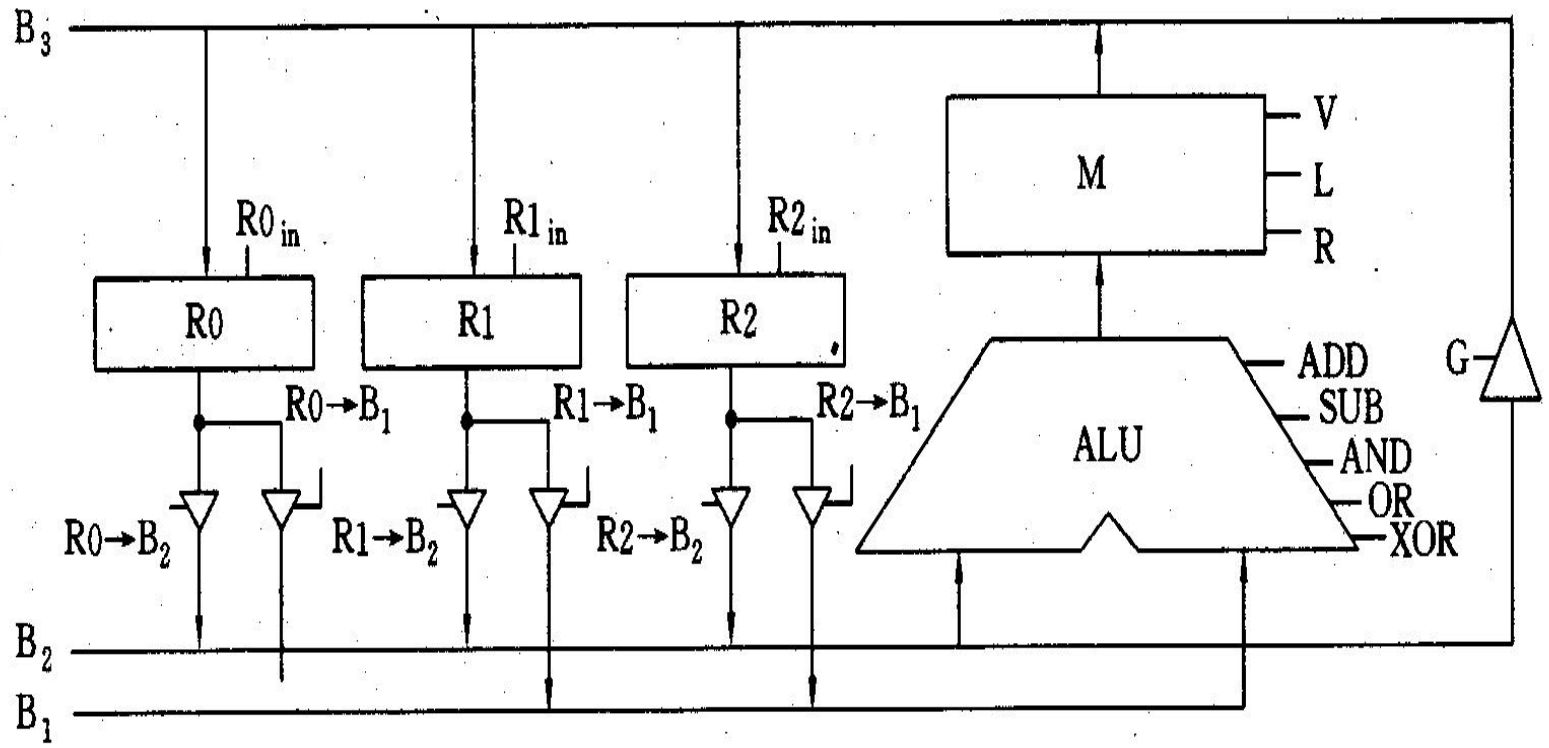


补充6-2.

某机运算器为三总线结构（如下图），三总线分别称为BI，B2，B3，连接B2，B3的控制信号为G。算逻部件ALU可进行ADD，SUB，AND, OR，XOR五种运算，输出多路器可进行直送(V)，左移一位(L)，右移一位(R)三种操作。三个通用寄存器RO，RI，R2都有输入和输出信号。

试写出实现下列功能所需的微操作：1/2[R1-R2]→R1

补充6-2.


B3
R0 in
R1 in
R2 in
M
V
L
R
R0
R1
R2
R0→B1
R1→B1
R2→B1
ADD
SUB
AND
OR
XOR
ALU
G
R0→B2
R1→B2
R2→B2
B2
B1

补充6-3.

何谓同步控制？何谓异步控制？试比较它们的特点及应用场合。

18．有一个8位机，采用单总线结构，地址总线16位（A_{15}~A_0），数据总线8位（D_7~D_0），控制总线中与主存有关的信号有MREQ（低电平有效允许访存）和R/W(高电平为读命令，低电平为写命令)。

主存地址分配如下：从0~8191为系统程序区，有ROM芯片组成；从8192~32767为用户程序区，最后（最大地址）2K地址空间为系统程序工作区。（上述地址均用十进制表示，按字节编址）。

现有如下存储芯片：8K×8的ROM，16K×1、2K×8、4K×8、8K×8的SRAM。请从上述规格中选用芯片设计该主存储器，画出主存的连接框图，并指以画出片选逻辑及与CPU的连接。

[解]

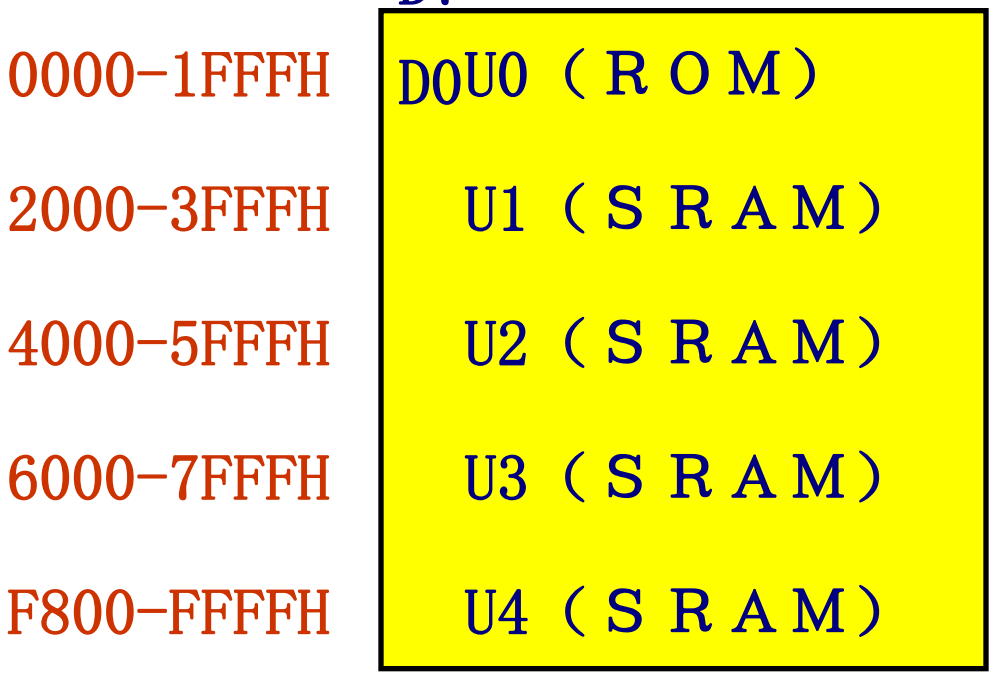
1）主存结构：系统程序区由ROM芯片组成，用户程序区和系统程序工作区均由RAM芯片组成。共需：

8K×8的ROM芯片1片，

8K×8的SRAM芯片3片，

2K×8的SRAM芯片1片。

主存地址分配如图所示。



2)片选逻辑的确定：

A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	–	–	–	–	–	–	–	–	–	–	A_0
0	0	0	–	–	–	–	–	–	–	–	–	–	–	–	–
0	0	1	–	–	–	–	–	–	–	–	–	–	–	–	–
0	1	0	–	–	–	–	–	–	–	–	–	–	–	–	–
0	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–
1	1	1	1	1	–	–	–	–	–	–	–	–	–	–	–

片内地址：U0-U3:A12-A0, U4 :A10-A0;

片选地址：A15，A14，A13，A12，A11

设各存储芯片的片选信号为低电平有效：

3)片选逻辑及与CPU的连接：