计算机组成原理

第 13讲

左德承

哈尔滨工业大学计算学部

容错与移动计算研究中心

7.5 RISC 技 术

一、RISC 的产生和发展

RISC（Reduced Instruction Set Computer）

CISC（Complex Instruction Set Computer）

80 — 20 规律　　—— RISC技术

- 典型程序中 **80%** 的语句仅仅使用处理机中 **20%** 的指令
- 执行频度高的简单指令，因复杂指令的存在，执行速度无法提高

? 能否用 **20%** 的简单指令组合不常用的 **80%** 的指令功能

二、RISC 的主要特征

7.5

- 选用使用频度较高的一些 简单指令，
  复杂指令的功能由简单指令来组合
- 指令 长度固定、指令格式种类少、寻址方式少
- 只有 **LOAD / STORE** 指令访存
- **CPU** 中有多个 通用 寄存器
- 采用 流水技术 一个时钟周期 内完成一条指令
- 采用 组合逻辑 实现控制器
- 采用 优化 的 编译 程序

三、CISC 的主要特征

7.5

- 系统指令 复杂庞大，各种指令使用频度相差大
- 指令 长度不固定、指令格式种类多、寻址方式多
- 访存 指令 不受限制
- CPU 中设有 专用寄存器
- 大多数指令需要 多个时钟周期 执行完毕
- 采用 微程序 控制器
- 难以 用 优化编译 生成高效的目的代码

四、RISC和CISC 的比较

7.5

1. RISC更能 充分利用 VLSI 芯片的面积

2. RISC 更能 提高计算机运算速度

   指令数、指令格式、寻址方式少，
   通用 寄存器多，采用 组合逻辑 ，
   便于实现 指令流水

3. RISC 便于设计，可 降低成本，提高 可靠性

4. RISC 有利于编译程序代码优化

5. RISC 不易 实现 指令系统兼容

第 8 章　CPU 的结构和功能

8.1 CPU 的结构

8.2 指令周期

8.3 指令流水

8.4 中断系统

8.1 CPU 的结构

一、 CPU 的功能

1. 控制器的功能

取指令 — 指令控制

分析指令 — 操作控制

执行指令，发出各种操作命令 — 操作控制

控制程序输入及结果的输出 — 时序控制

总线管理 — 时序控制

处理异常情况和特殊请求 — 中断处理

2. 运算器的功能

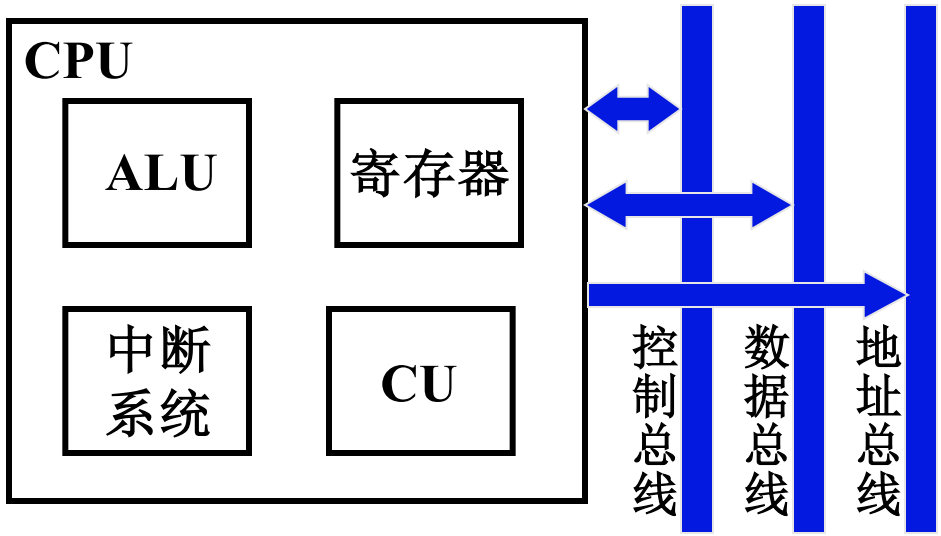
实现算术运算和逻辑运算 — 数据加工

二、CPU 结构框图

8.1

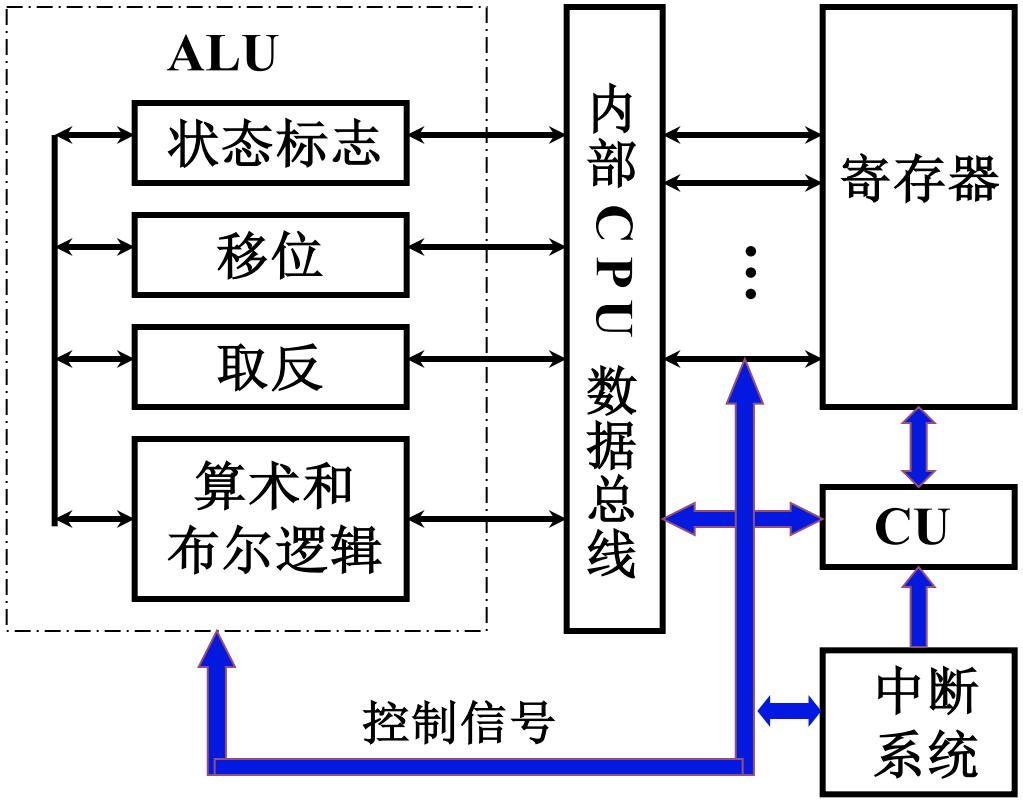
1. CPU 与系统总线

指令控制		PC　IR
操作控制	}	CU　时序电路
时间控制	}	CU　时序电路
数据加工		ALU　寄存器
处理中断		中断系统



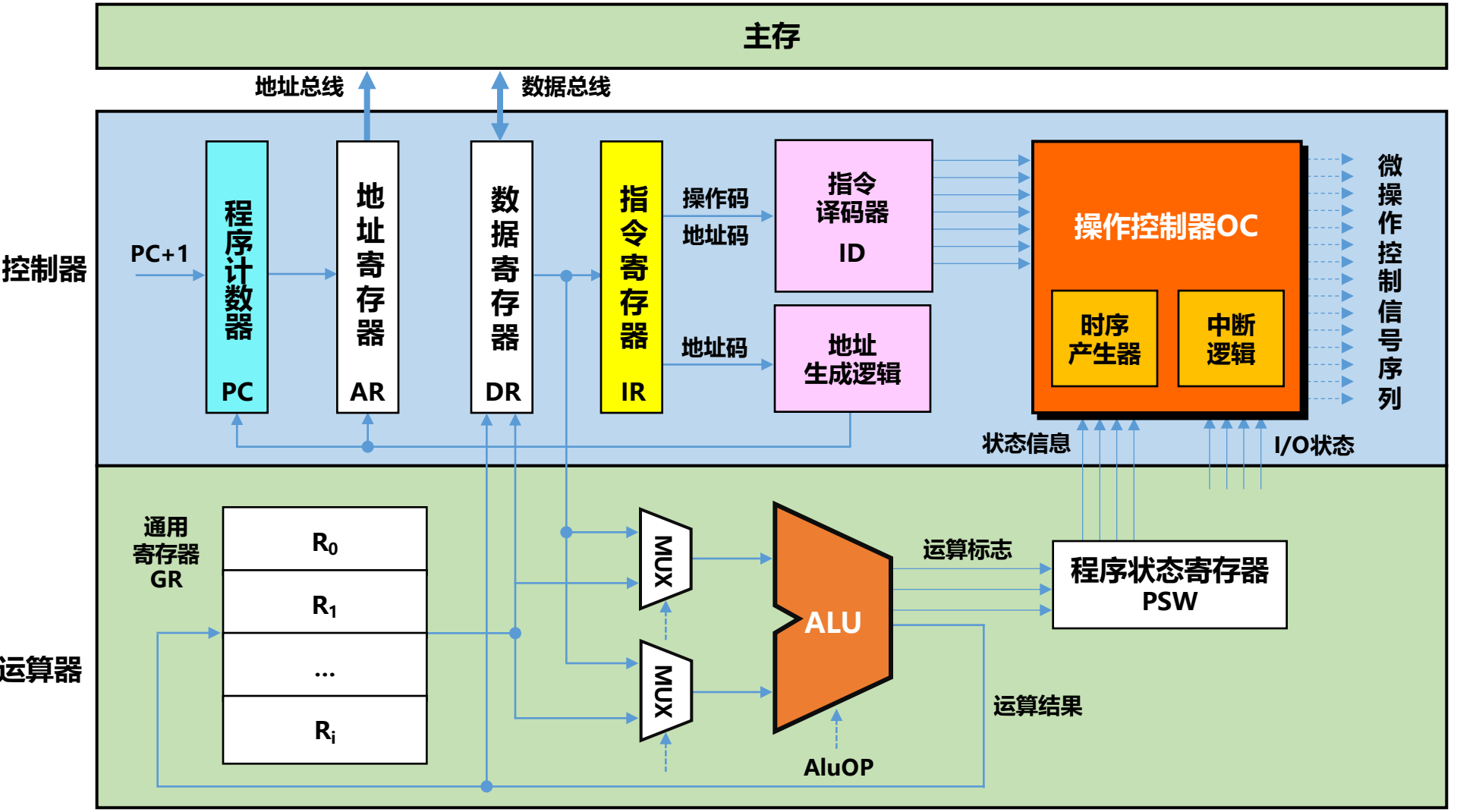
2. CPU 的内部结构

8.1



简单CPU模型

8.1


主存
地址总线
数据总线
控制器
PC+1
程序计数器
PC
地址寄存器
AR
数据寄存器
DR
指令寄存器
IR
操作码
地址码
指令译码器
ID
地址码
地址生成逻辑
操作控制器OC
时序产生器
中断逻辑
微操作控制信号序列
状态信息
I/O状态
运算器
通用寄存器
GR
R_0
R_1
...
R_i
MUX
MUX
ALU
运算标志
程序状态寄存器
PSW
运算结果
AluOP

三、 CPU 的寄存器

8.1

1. 用户可见寄存器

(1) 通用寄存器　　存放操作数

可作 某种寻址方式所需的 专用寄存器

(2) 数据寄存器　　存放操作数（满足各种数据类型）

两个寄存器拼接存放双倍字长数据

(3) 地址寄存器　　存放地址，其位数应满足最大的地址范围

用于特殊的寻址方式　段基址　栈指针

(4) 条件码寄存器　存放条件码，可作程序分支的依据

如 正、负、零、溢出、进位等

2. 控制和状态寄存器

(1) 控制寄存器

PC → MAR → M → MDR → IR

控制 CPU 操作

其中 MAR、MDR、IR　　用户不可见

PC　　用户可见

(2) 状态寄存器

状态寄存器　　存放条件码

PSW 寄存器　　存放程序状态字

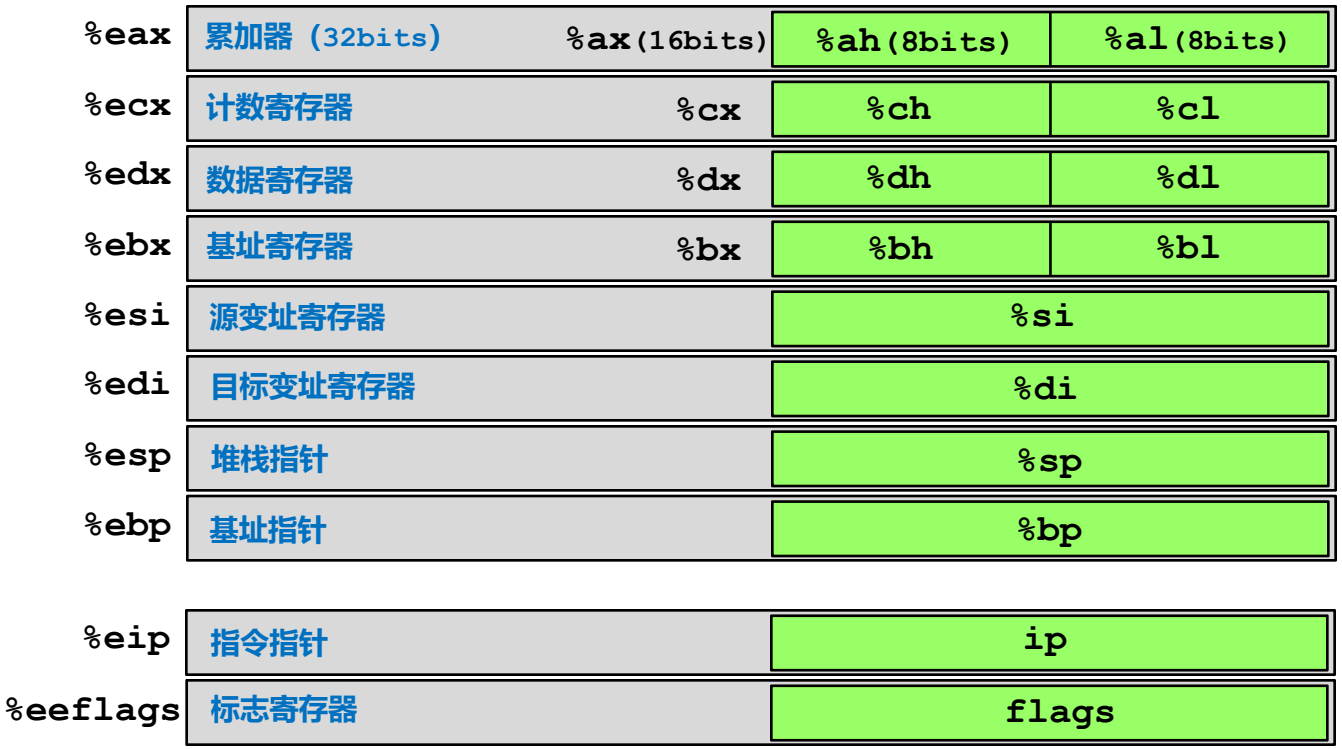
MIPS-32个寄存器

7.4

寄存器#	助记符	释义
0	$zero	常量寄存器，值固定为0，硬件实现
1	$at	临时变量，保留给汇编器使用
2~3	$v0~$v1	函数调用返回值
4~7	$a0~$a3	4个函数调用参数
8~15	$t0~$t7	暂存寄存器，子程序内可直接使用
16~23	$s0~$s7	变量寄存器，子程序返回应复原内容
24~25	$t8~$t9	暂存寄存器，子程序内可直接使用
26~27	$k0~$k1	操作系统保留，中断异常处理
28	$gp	全局指针 (Global Pointer)
29	$sp	堆栈指针 (Stack Pointer)
30	$fp	帧指针 (Frame Pointer)
31	$ra	函数返回地址 (Return Address)

- **32个32位通用寄存器$0~$31**
- **32个32位单精度浮点寄存器f_0-f_{31}**
- **2个32位乘、商寄存器 H_i 和L_0**
- **程序寄存器PC是单独的寄存器**
- **无程序状态寄存器**
- **RISC-V也有类似的32个寄存器设置**

X86-- IA-32的寄存器组织



- 8个通用寄存器
- 两个专用寄存器
- 6个段寄存器

CS（代码段）16bits
SS（堆栈段）
DS（数据段）
ES（附加段）
FS（附加段）
GS（附加段）

四、 控制单元 CU 和中断系统

8.1

1. CU　产生全部指令的微操作命令序列

组合逻辑设计　　硬连线逻辑

微程序设计　　存储逻辑

很快会讲到^_^

2. 中断系统

参见：I/O系统

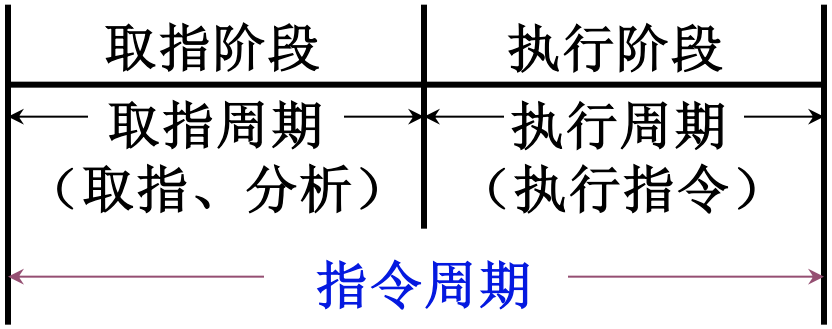
8.2 指 令 周 期

一、 指令周期的基本概念

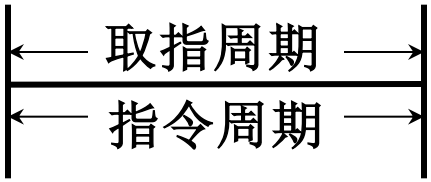
1．指令周期

取出并执行一条指令所需的全部时间

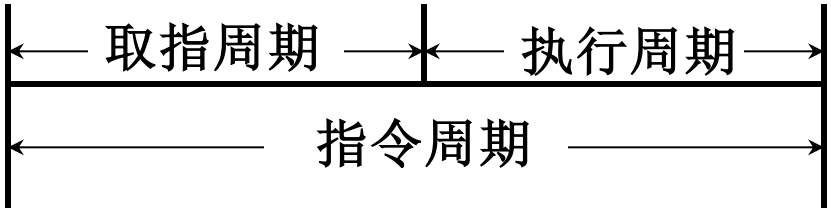
完成一条指令 { 取指、分析 —— 取指周期
执行 —— 执行周期



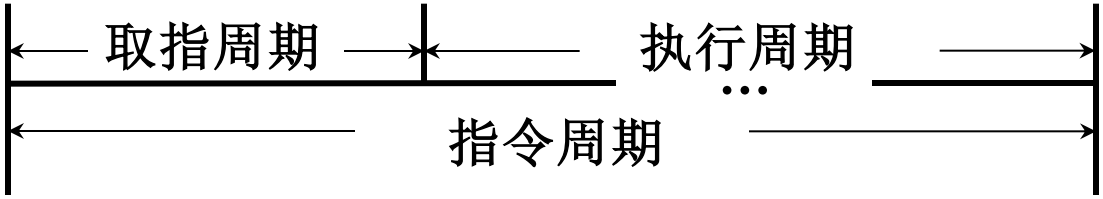
2. 每条指令的指令周期不同



NOP

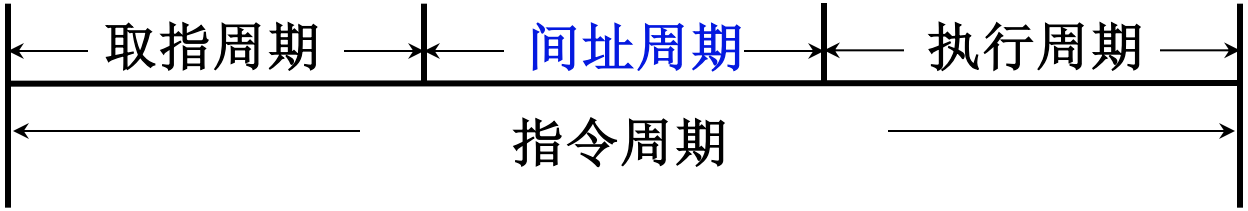


ADD mem

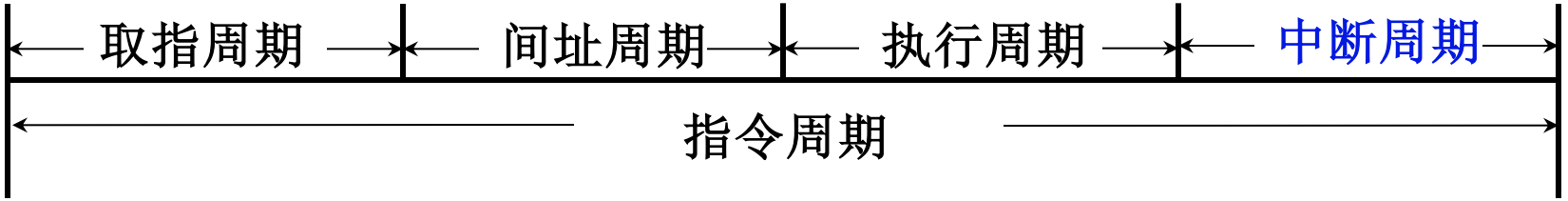


MUL mem

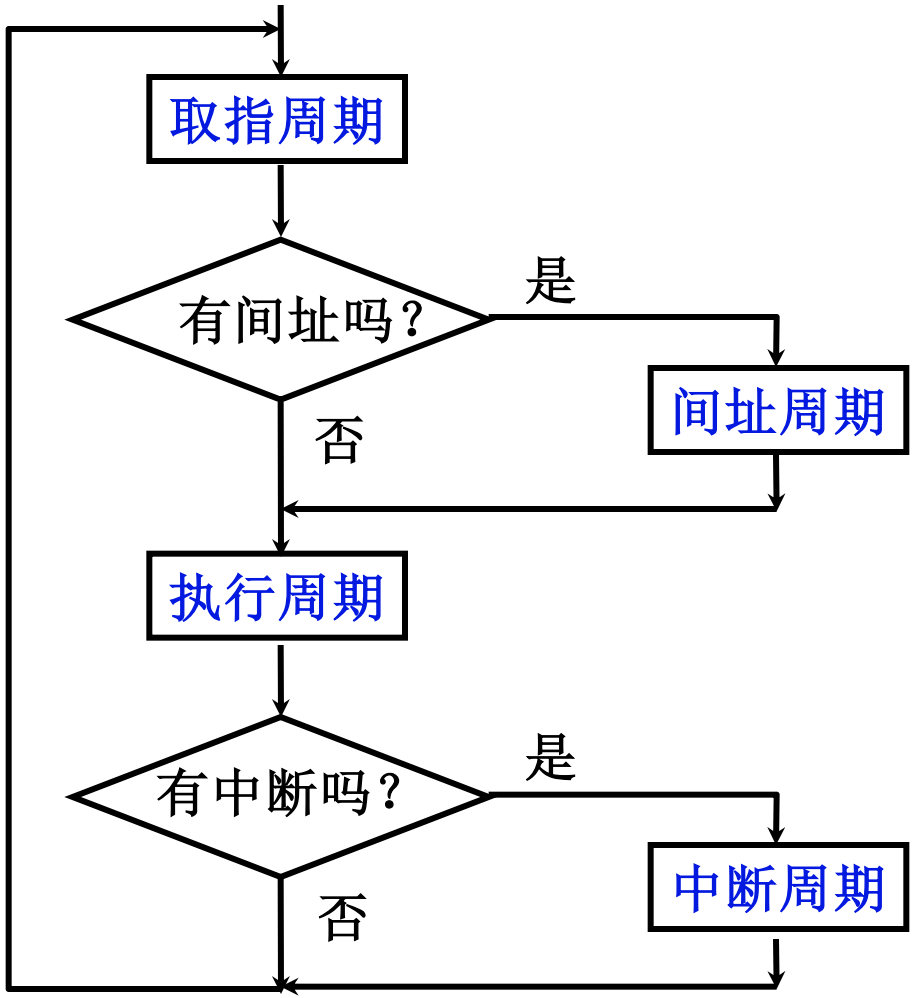
3. 具有间接寻址的指令周期



4. 带有中断周期的指令周期



5. 指令周期流程



6. CPU 工作周期的标志

CPU 访存有四种性质

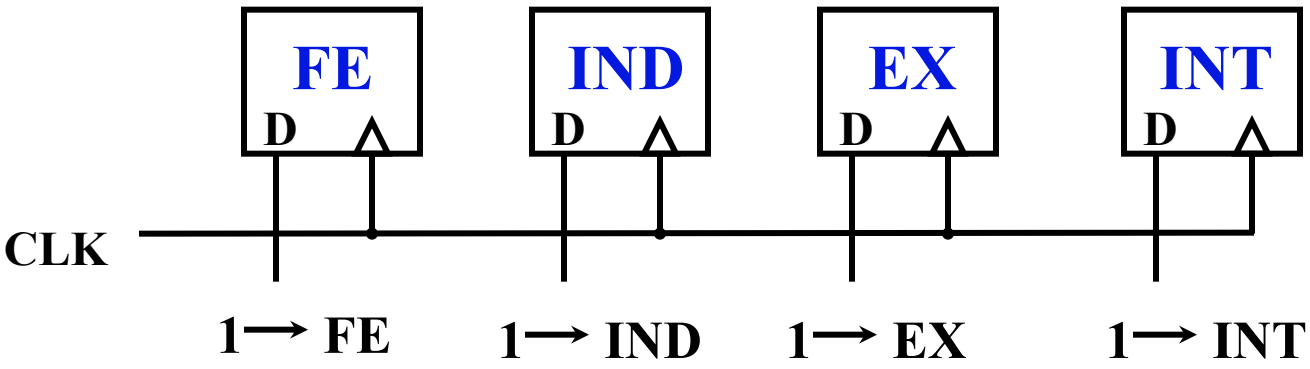
取 指令　　取指周期

取 地址　　间址周期

取 操作数　　执行周期

存 程序断点　　中断周期

CPU 的
4个工作周期



第 9 章　控制单元的功能

9.1　操作命令的分析

9.2　控制单元的功能

9.1 操作命令的分析

完成一条指令分 4 个工作周期

取指周期

间址周期

执行周期

中断周期

9.1 操作命令的分析

一、取指周期

PC ⟶ MAR ⟶ 地址线

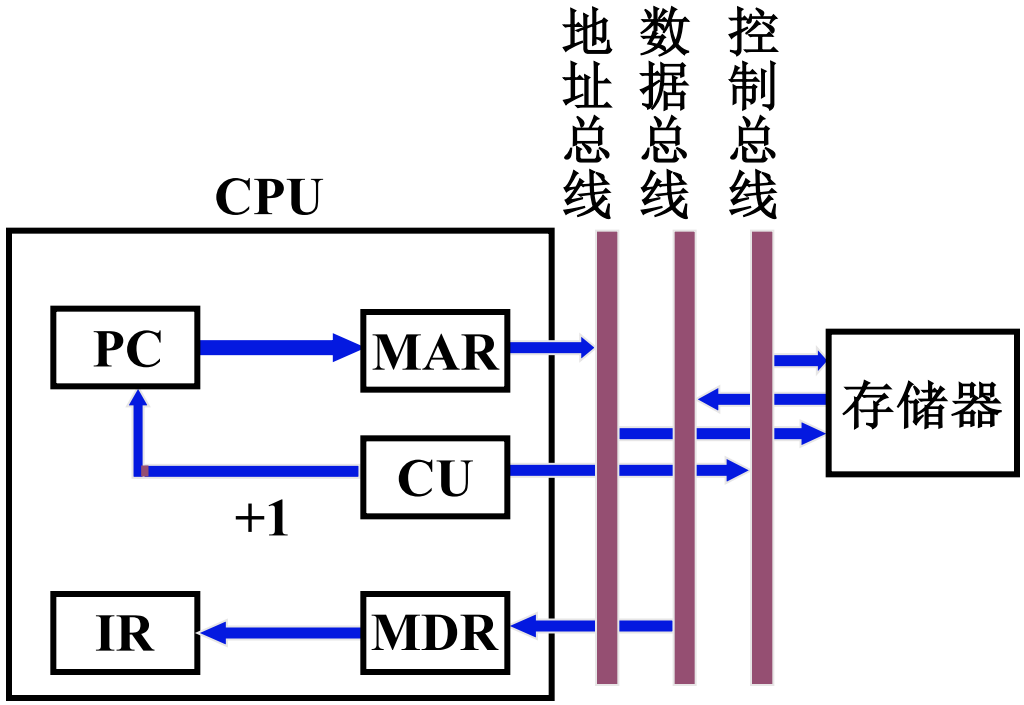
1 ⟶ R

M (MAR) ⟶ MDR

MDR ⟶ IR

OP（IR）⟶ CU

(PC) + 1 ⟶ PC



二、间址周期

9.1

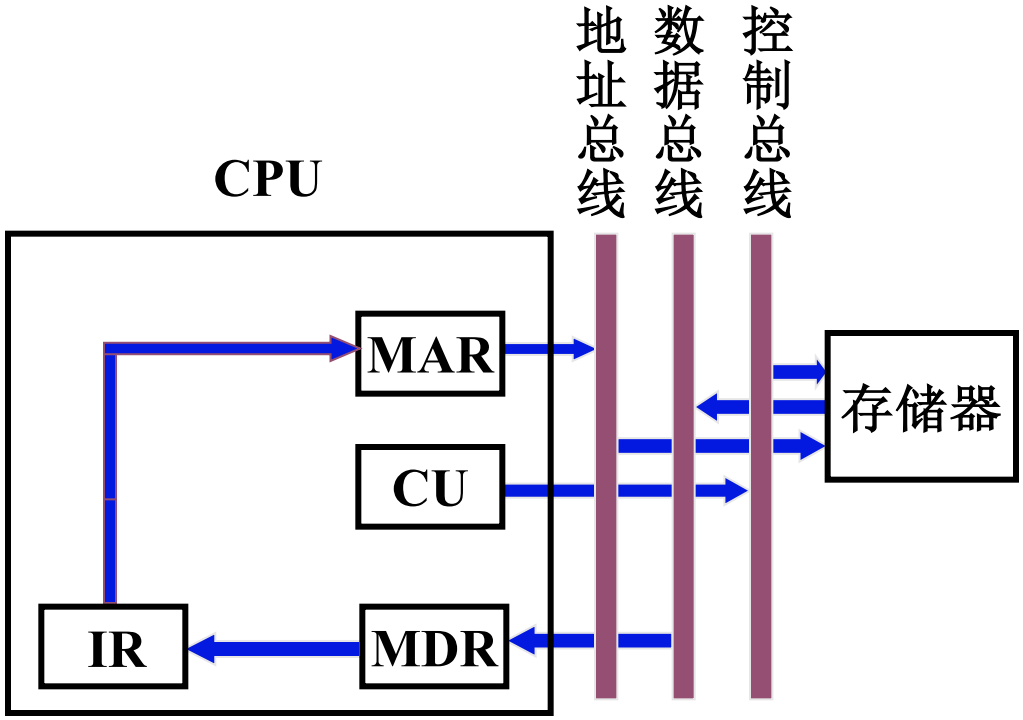
指令形式地址 ⟶ MAR

Ad (IR) ⟶ MAR

1⟶R

M (MAR) ⟶ MDR

MDR⟶ Ad (IR)



三、执行周期

9.1

1. 非访存指令

(1) **CLA** 清A　　$0 \rightarrow ACC$

(2) **COM** 取反　　$\overline{ACC} \rightarrow ACC$

(3) **SHR** 算术右移　　$L(ACC) \rightarrow R(ACC), ACC_0 \rightarrow ACC_0$

(4) **CSL** 循环左移　　$R(ACC) \rightarrow L(ACC), ACC_0 \rightarrow ACC_n$

(5) **STP** 停机指令　　$0 \rightarrow G$

2. 访存指令

(1) 加法指令　**ADD　X**

Ad(IR) ⟶ MAR

1 ⟶ R

M(MAR) ⟶ MDR

(ACC) + (MDR) ⟶ ACC

(2) 存数指令　**STA　X**

Ad(IR) ⟶ MAR

1 ⟶ W

ACC ⟶ MDR

MDR⟶ M(MAR)

(3) 取数指令　LDA　X

Ad (IR) ⟶ MAR

1 ⟶ R

M (MAR) ⟶ MDR

MDR ⟶ ACC

3. 转移指令

(1) 无条件转　JMP　X

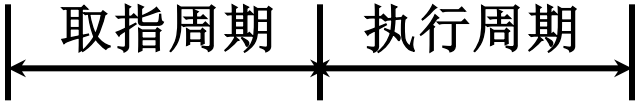
Ad (IR) ⟶ PC

(2) 条件转移　BAN　X　（负则转）

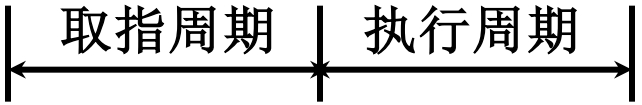
$A_0 \cdot Ad(IR) + \bar{A}_0 (PC) \longrightarrow PC$

4. 三类指令的指令周期

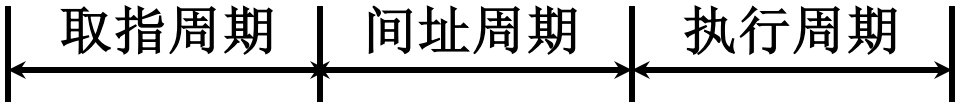
非访存 指令周期



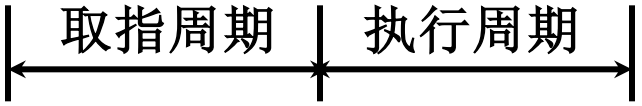
直接访存 指令周期



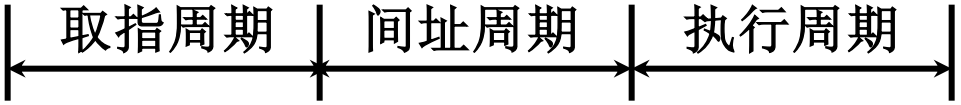
间接访存 指令周期



转移 指令周期



间接转移 指令周期



四、中断周期

9.1

程序断点存入 “ 0 ” 地址　　程序断点 进栈

0 → MAR　　(SP) – 1 → MAR

1 → W　　1 → W

PC → MDR　　PC → MDR

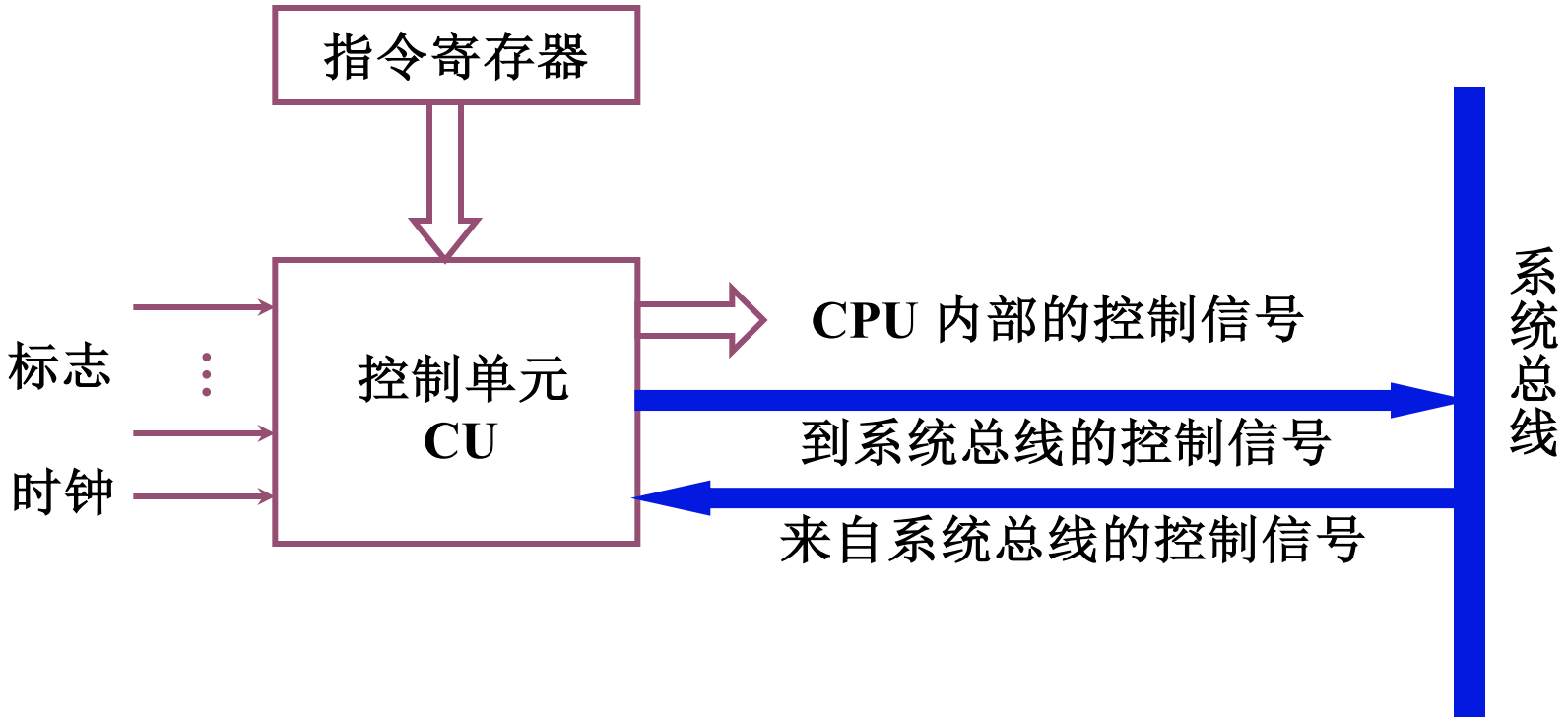
MDR → M (MAR)　　MDR → M (MAR)

中断识别程序入口地址 M → PC

0 → EINT（置“0”）　　0 → EINT（置“0”）

9.2 控制单元的功能

一、控制单元的外特性



1. 输入信号

(1) 时钟

CU 受时钟控制

一个时钟脉冲

发一个操作命令或一组需同时执行的操作命令

(2) 指令寄存器 OP (IR) → CU

控制信号 与操作码有关

(3) 标志

CU 受标志控制

(4) 外来信号

如 INTR 中断请求

HRQ 总线请求

2. 输出信号

(1) CPU 内的各种控制信号

$R_i \longrightarrow R_j$

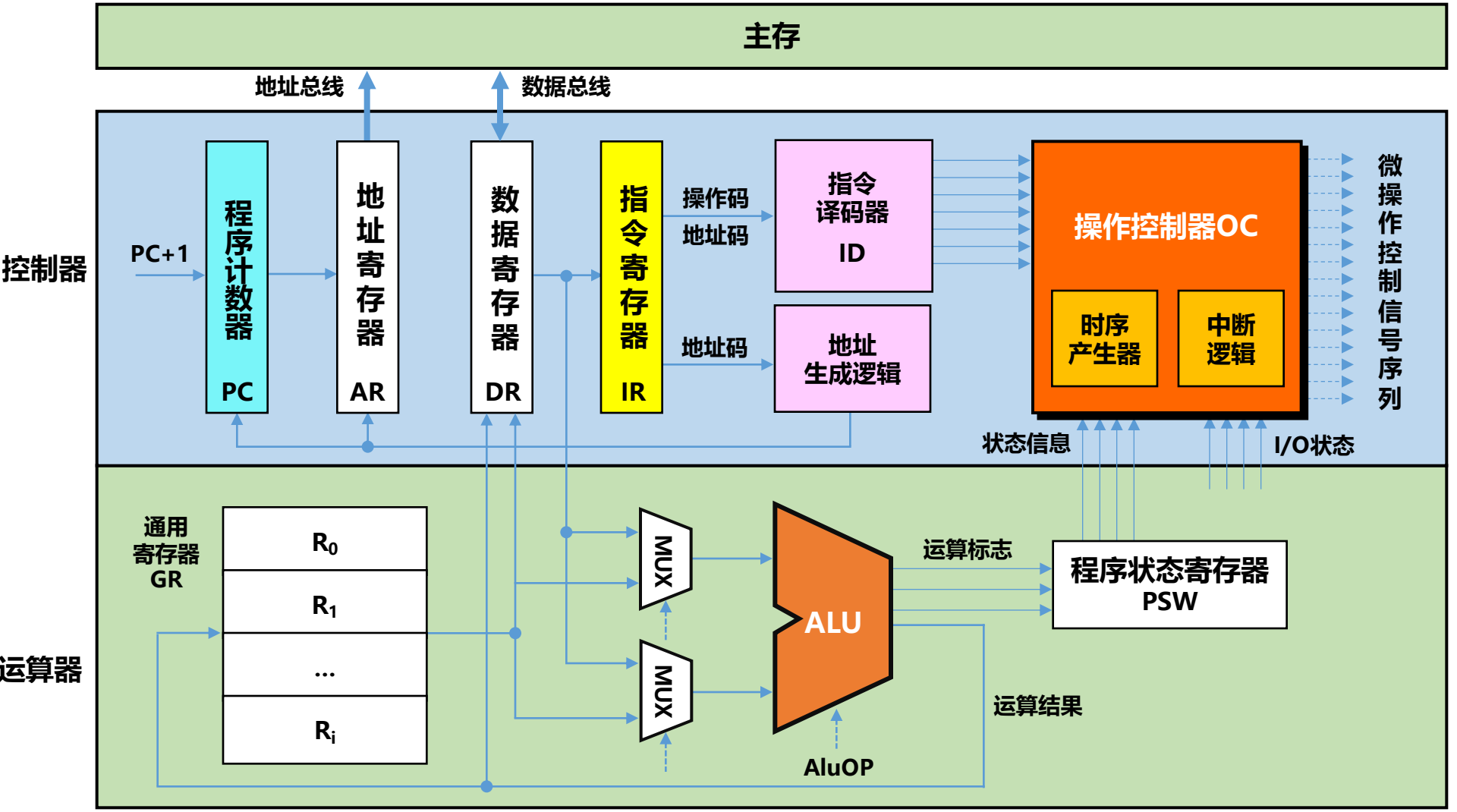
(PC) + 1 ⟶ PC

ALU　＋、－、与、或 ……

(2) 送至控制总线的信号

$\overline{\text{MREQ}}$	访存控制信号
$\overline{\text{IO}}$/M	访 IO/ 存储器的控制信号
$\overline{\text{RD}}$	读命令
$\overline{\text{WR}}$	写命令
INTA	中断响应信号
HLDA	总线响应信号

简单CPU模型

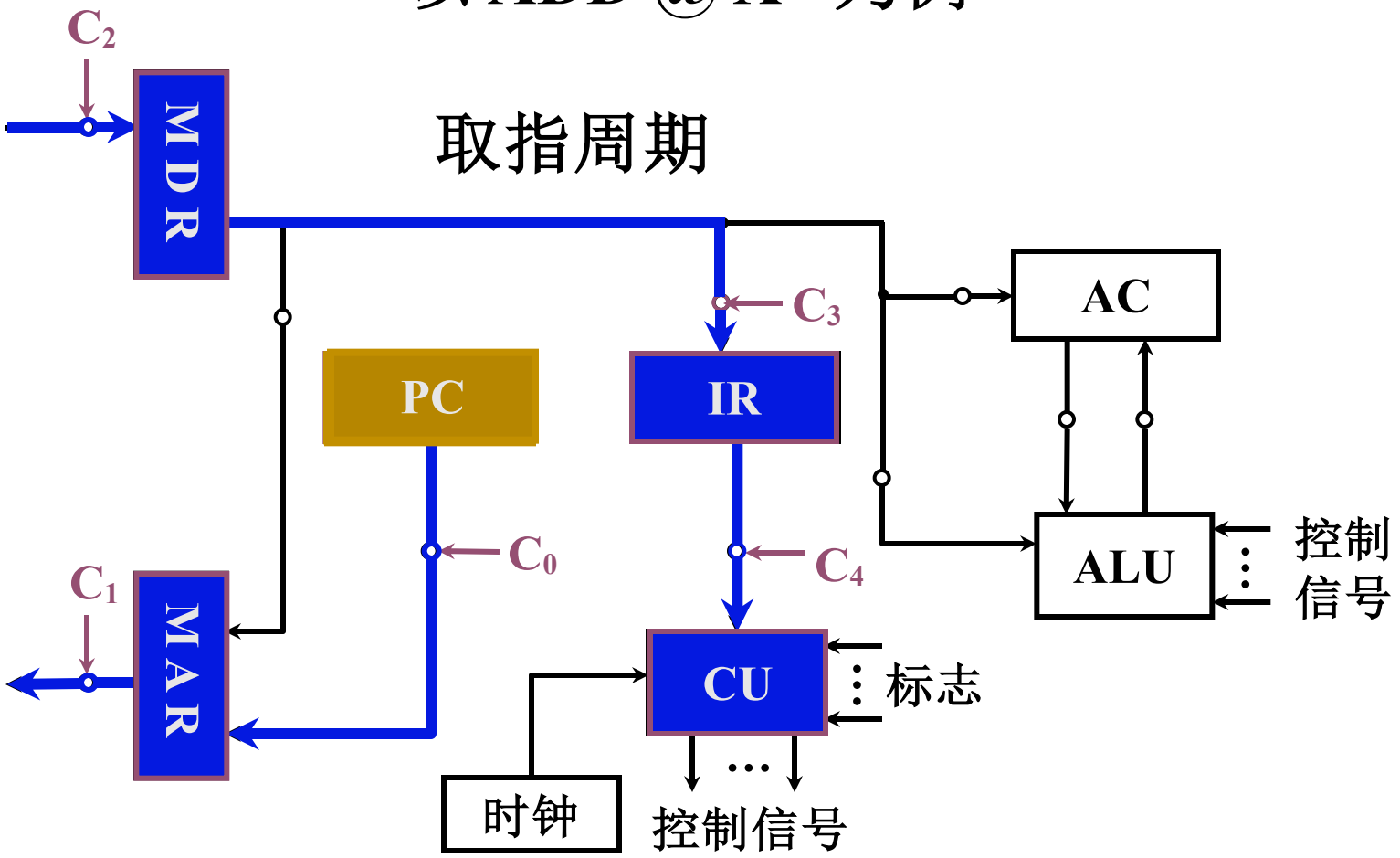

主存
地址总线
数据总线
控制器
PC+1
程序计数器
PC
地址寄存器
AR
数据寄存器
DR
指令寄存器
IR
操作码
地址码
指令译码器
ID
地址码
地址生成逻辑
操作控制器OC
时序产生器
中断逻辑
微操作控制信号序列
状态信息
I/O状态
运算器
通用寄存器
GR
R_0
R_1
...
R_i
MUX
MUX
ALU
运算标志
程序状态寄存器
PSW
运算结果
AluOP

二、控制信号举例

9.2

1. 不采用 CPU 内部总线的方式

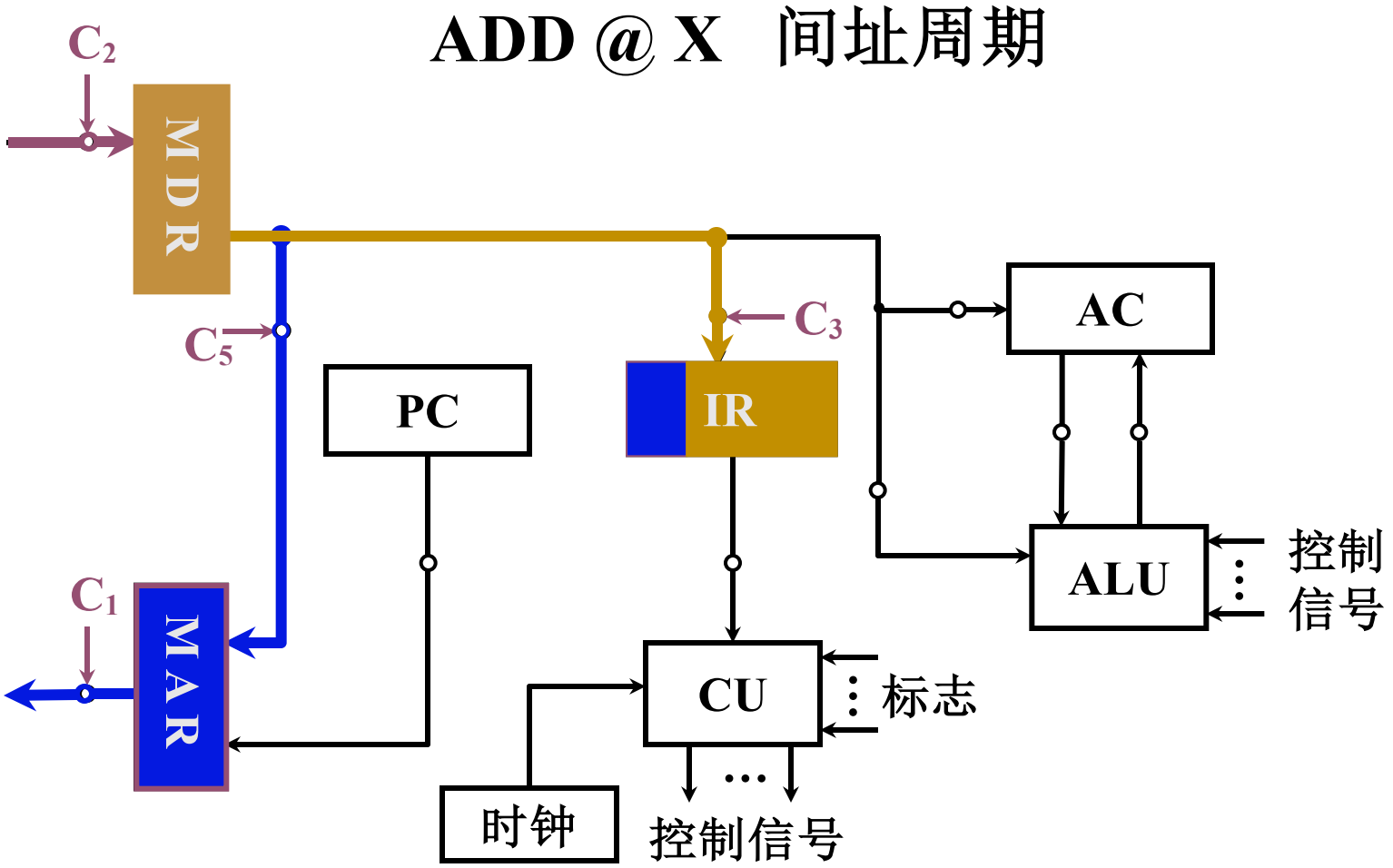
以 ADD @ X 为例



二、控制信号举例

9.2

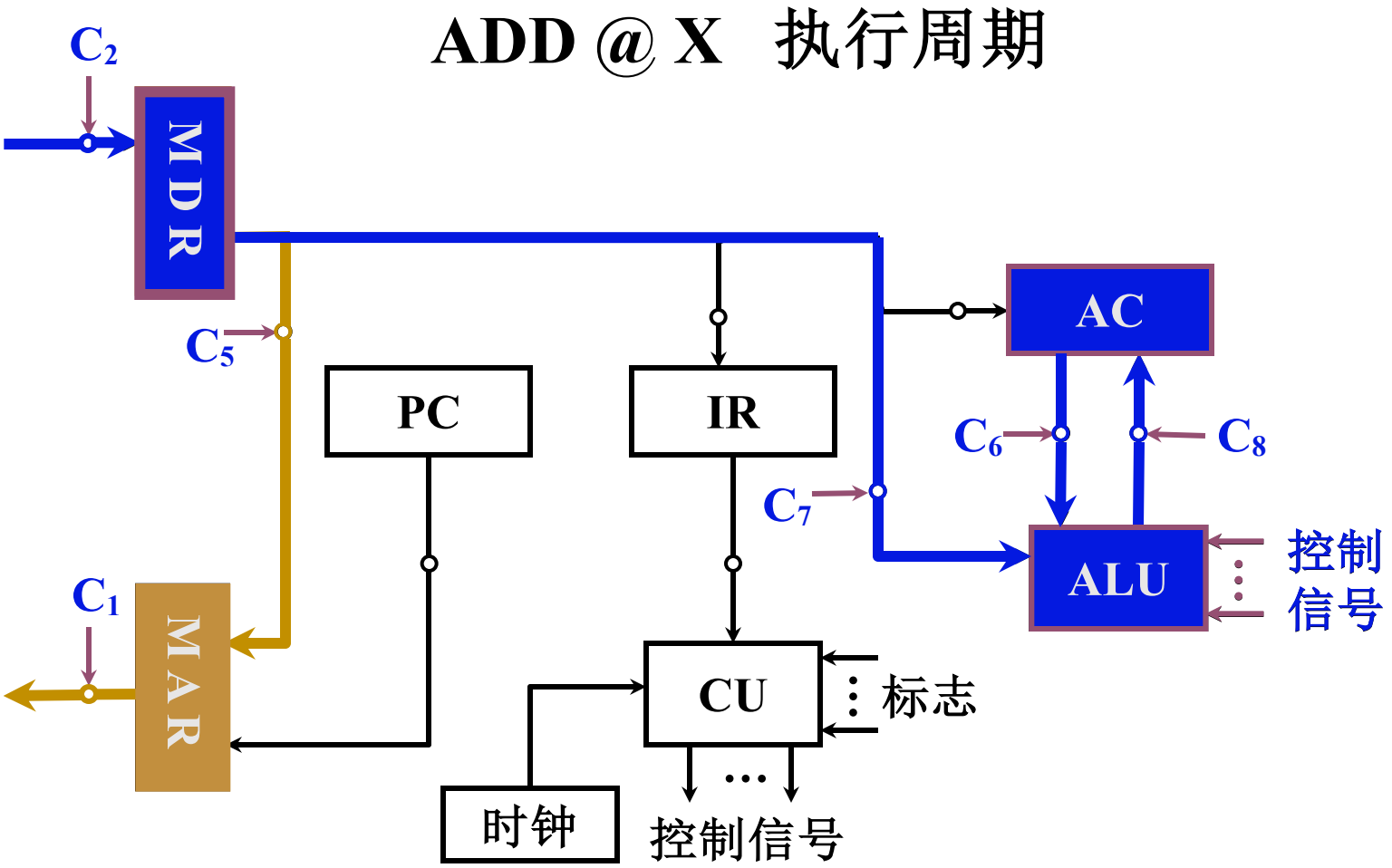
1. 不采用 CPU 内部总线的方式



二、控制信号举例

9.2

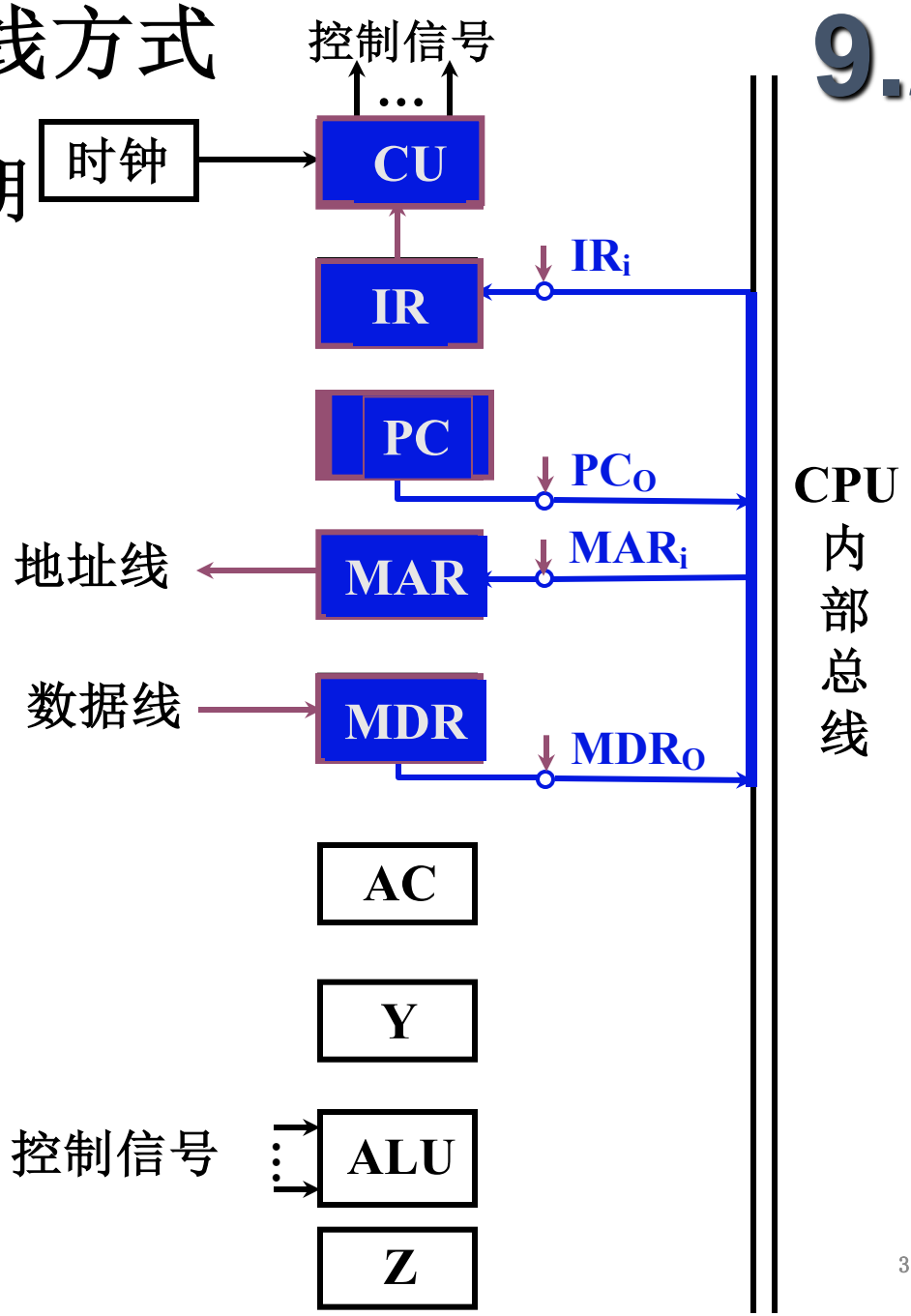
1. 不采用 CPU 内部总线的方式



2. 采用 CPU 内部总线方式

(1) ADD @ X 取指周期

- PC⟶ MAR⟶ 地址线
  PC_O MAR_i
- CU 发读命令 1⟶R
- 数据线 ⟶ MDR
- MDR ⟶ IR
  MDR_O IR_i
- OP（IR）⟶ CU
- （PC）+ 1⟶PC

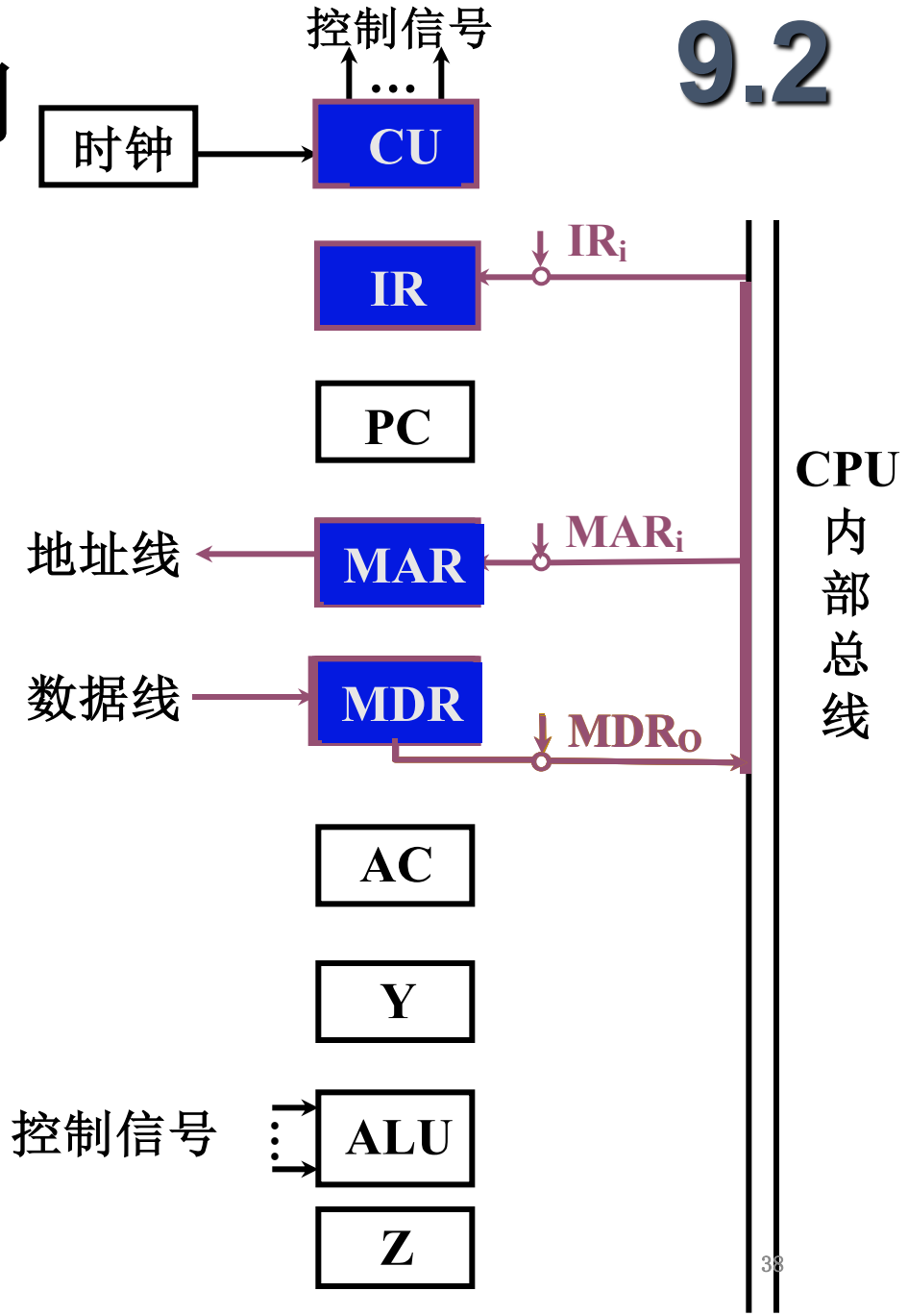


(2) ADD @ X 间址周期

形式地址 ⟶ MAR

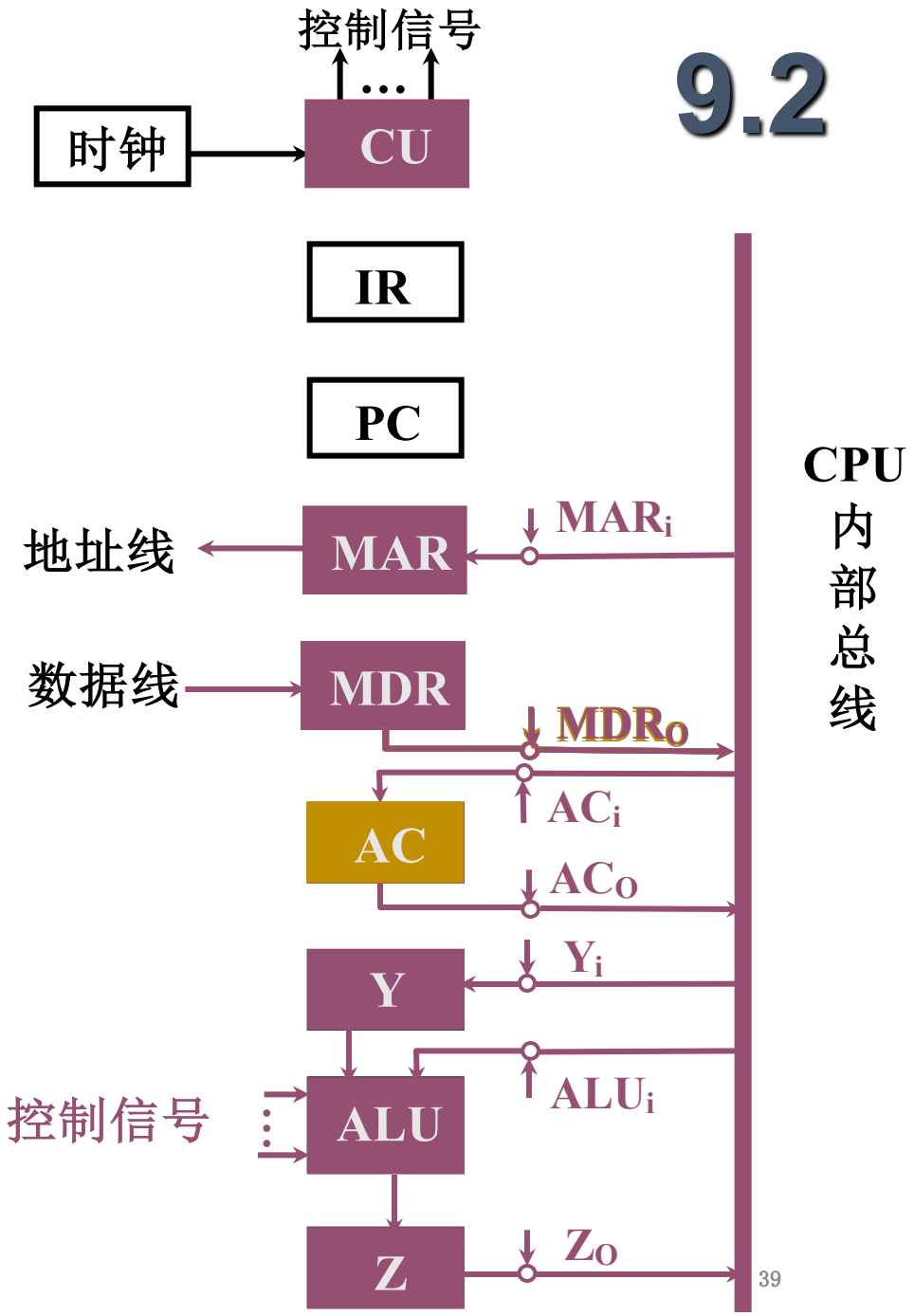
- MDR ⟶ MAR ⟶ 地址线
  MDR_O MAR_i
- 1 ⟶ R
- 数据线 ⟶ MDR
- MDR ⟶ IR
  MDR_O IR_i

有效地址 ⟶ Ad（IR）



(3) ADD @ X 执行周期

- MDR ⟶ MAR ⟶ 地址线
  MDR_O MAR_i
- 1 ⟶ R
- 数据线 ⟶ MDR
- MDR ⟶ Y ⟶ ALU
  MDR_O Y_i
- AC ⟶ ALU
  AC_O ALU_i
- （AC）+（Y）⟶ Z
- Z ⟶ AC
  Z_O AC_i



三、多级时序系统

9.2

1. 机器周期

(1) 机器周期的概念

所有指令执行过程中的一个基准时间

(2) 确定机器周期需考虑的因素

每条指令的执行 步骤

每一步骤 所需的 时间

(3) 基准时间的确定

- 以完成 最复杂 指令功能的时间 为准
- 以 访问一次存储器 的时间 为基准

若指令字长 = 存储字长　　取指周期 = 机器周期

2. 时钟周期（节拍、状态）

一个机器周期内可完成若干个微操作

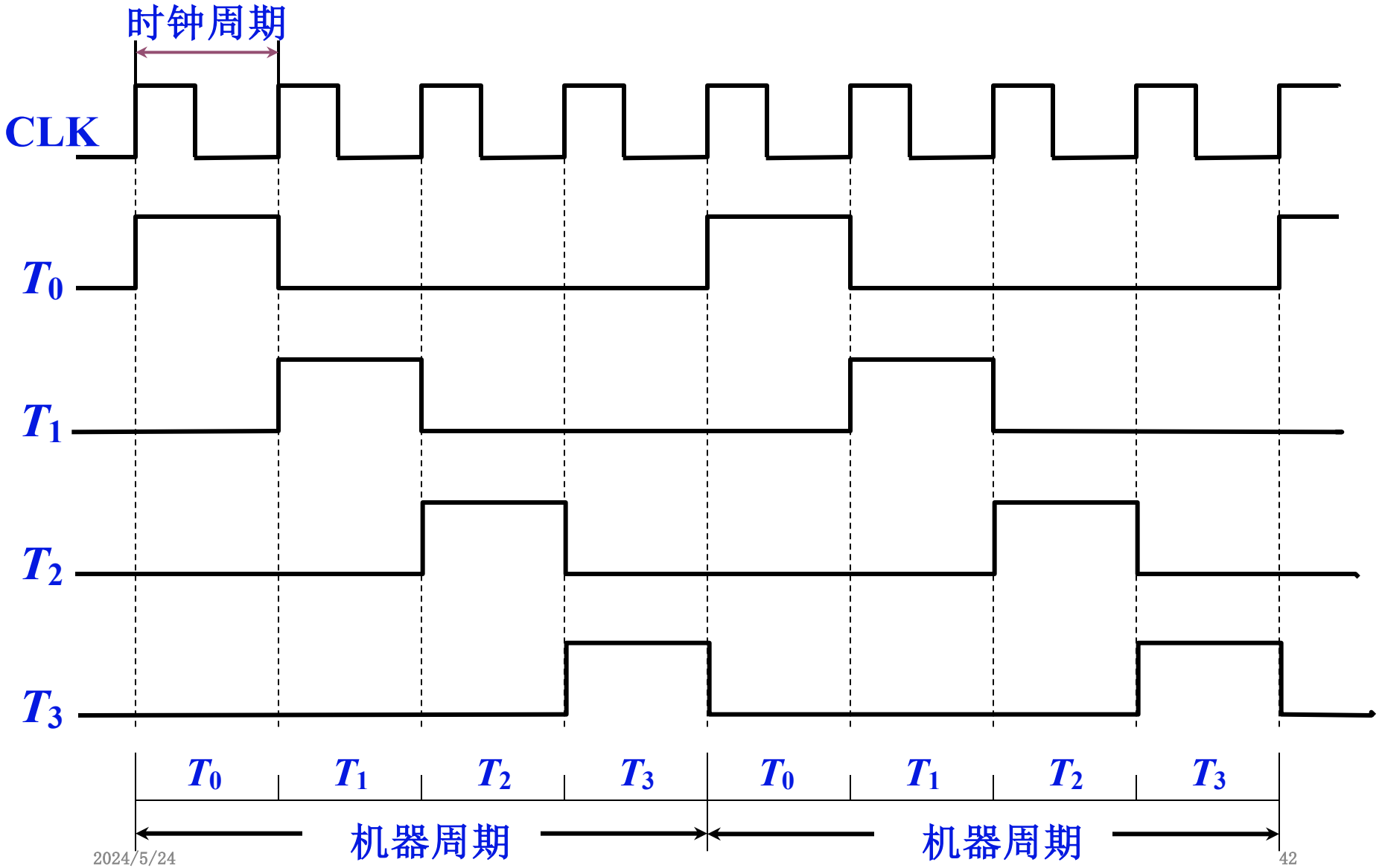
每个微操作需一定的时间

将一个机器周期分成若干个时间相等的时间段（节拍、状态、时钟周期）

时钟周期是控制计算机操作的最小单位时间

用时钟周期控制产生一个或几个微操作命令

2. 时钟周期（节拍、状态）

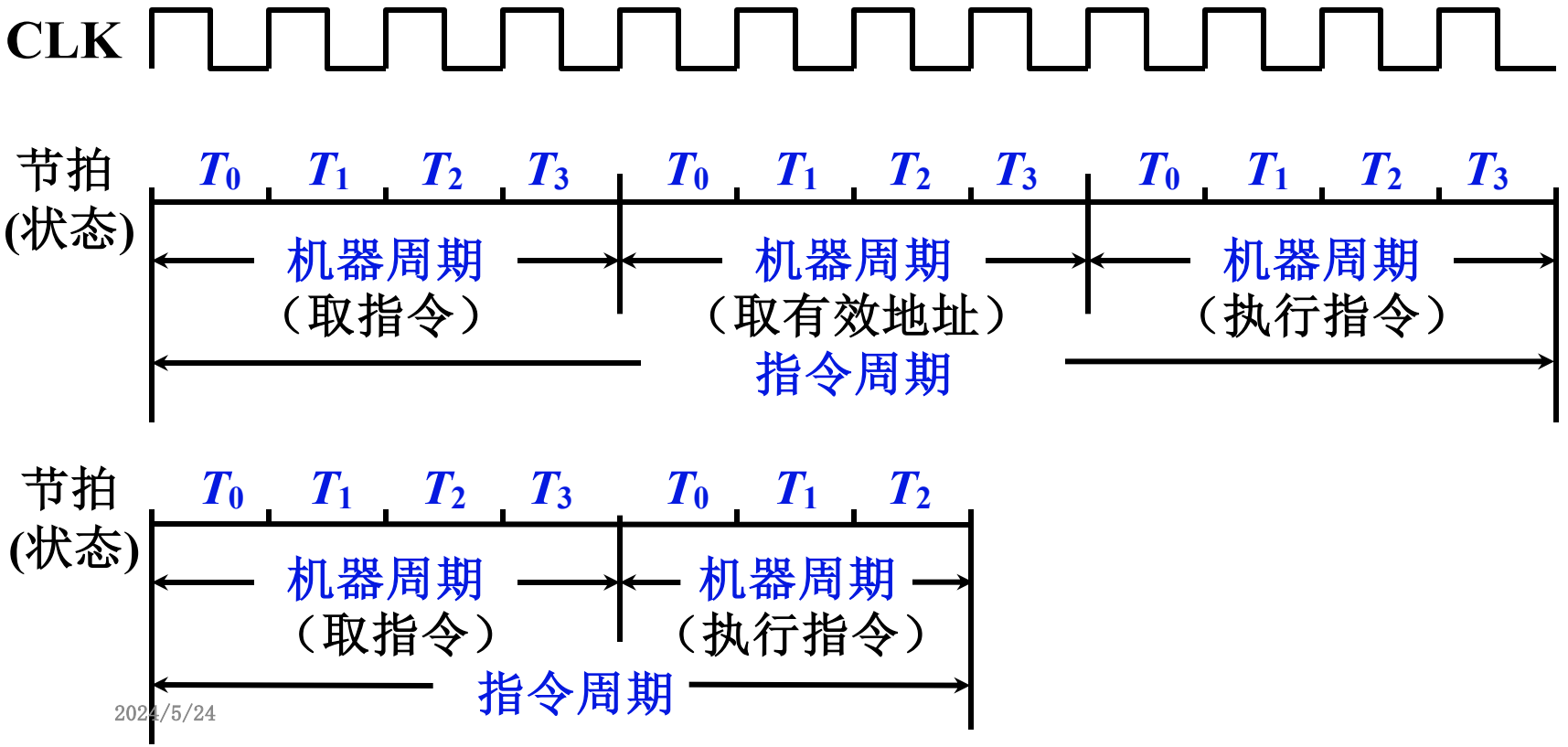


3. 多级时序系统

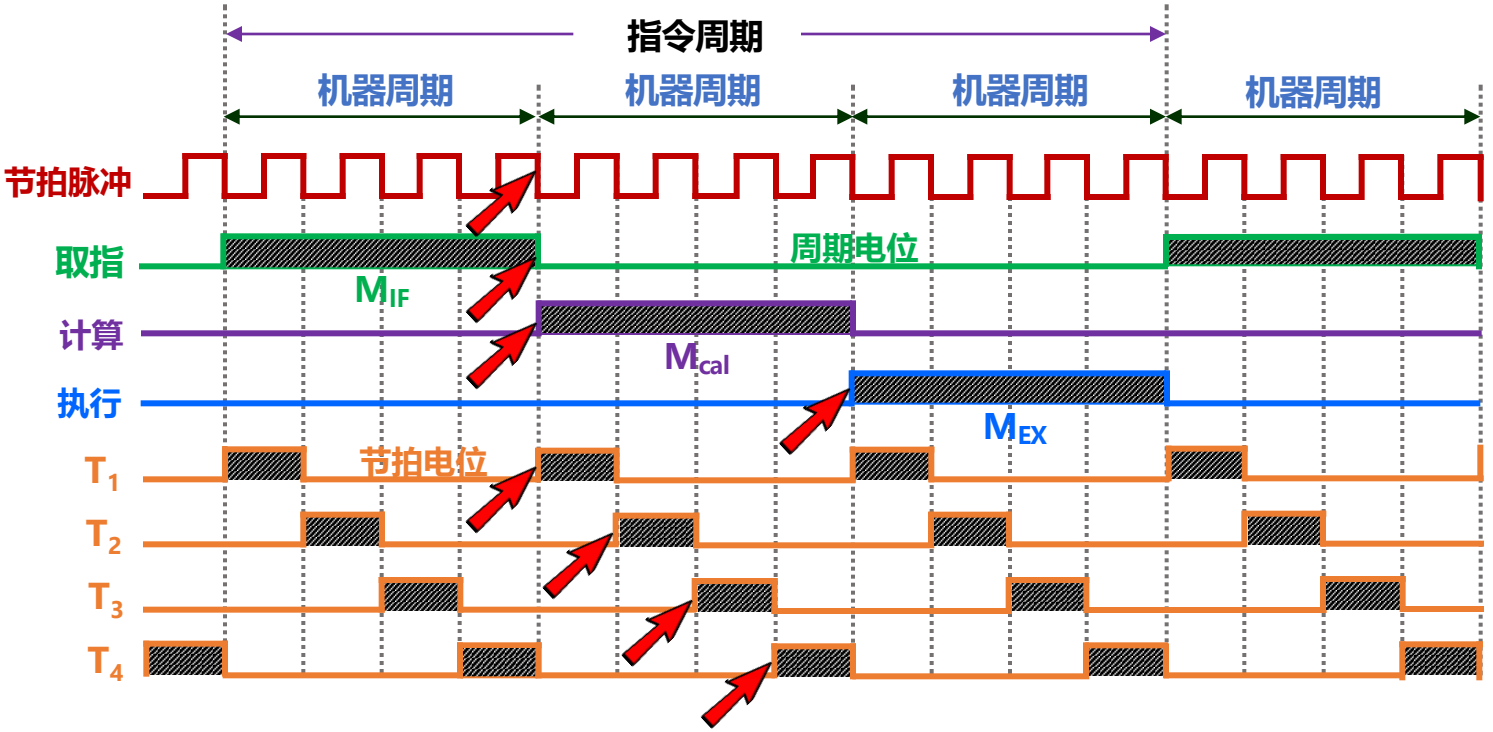
机器周期、节拍（状态）组成多级时序系统

一个指令周期包含若干个机器周期

一个机器周期包含若干个时钟周期



定长指令周期的三级时序发生器（举例）



构建时序发生器？ 输入：节拍脉冲 输出：M_{IF}，Mcal，M_{EX}，T1~T4

4. 机器速度与机器主频的关系 9.2

机器的 主频 f 越快 机器的 速度也越快

在机器周期所含时钟周期数 相同 的前提下，两机 平均指令执行速度之比 等于 两机主频之比

$$\frac{\mathbf{MIPS}_1}{\mathbf{MIPS}_2} = \frac{f_1}{f_2}$$

机器速度 不仅与 主频有关，还与机器周期中所含时钟周期（主频的倒数）数 以及指令周期中所含的 机器周期数有关

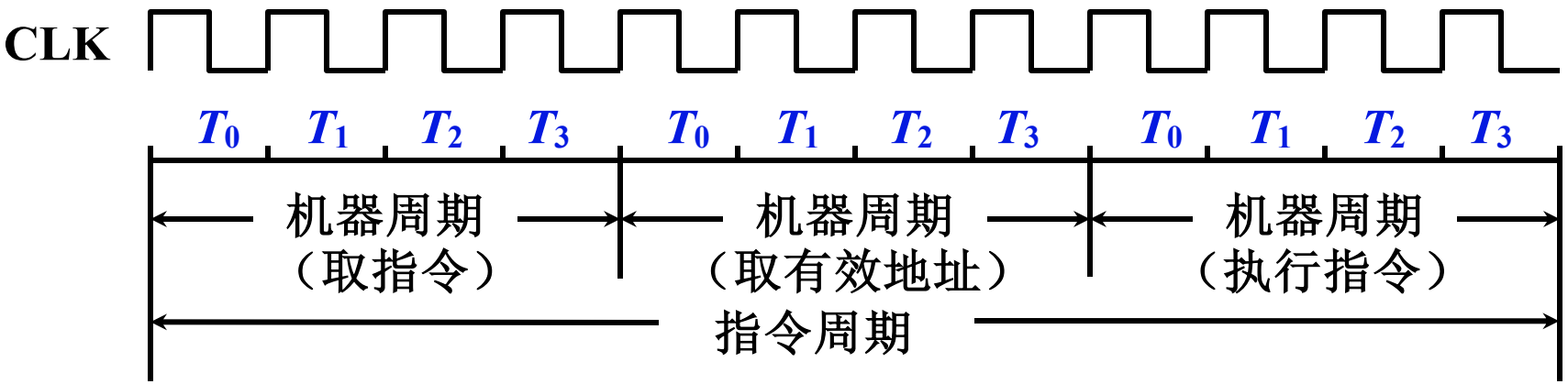
四、控制方式

9.2

产生不同微操作命令序列所用的时序控制方式

1. 同步控制方式

任一微操作均由 统一基准时标 的时序信号控制



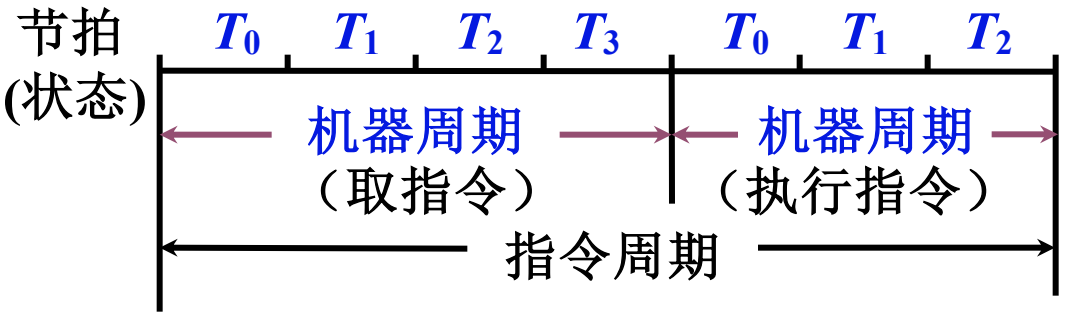
(1) 采用 定长 的机器周期

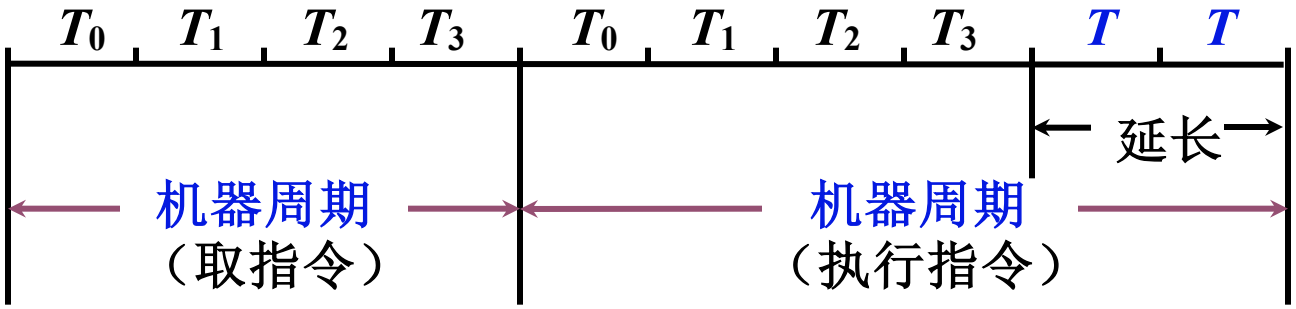
以 最长 的微操作序列和 最复杂 的微操作作为 标准

机器周期内 节拍数相同

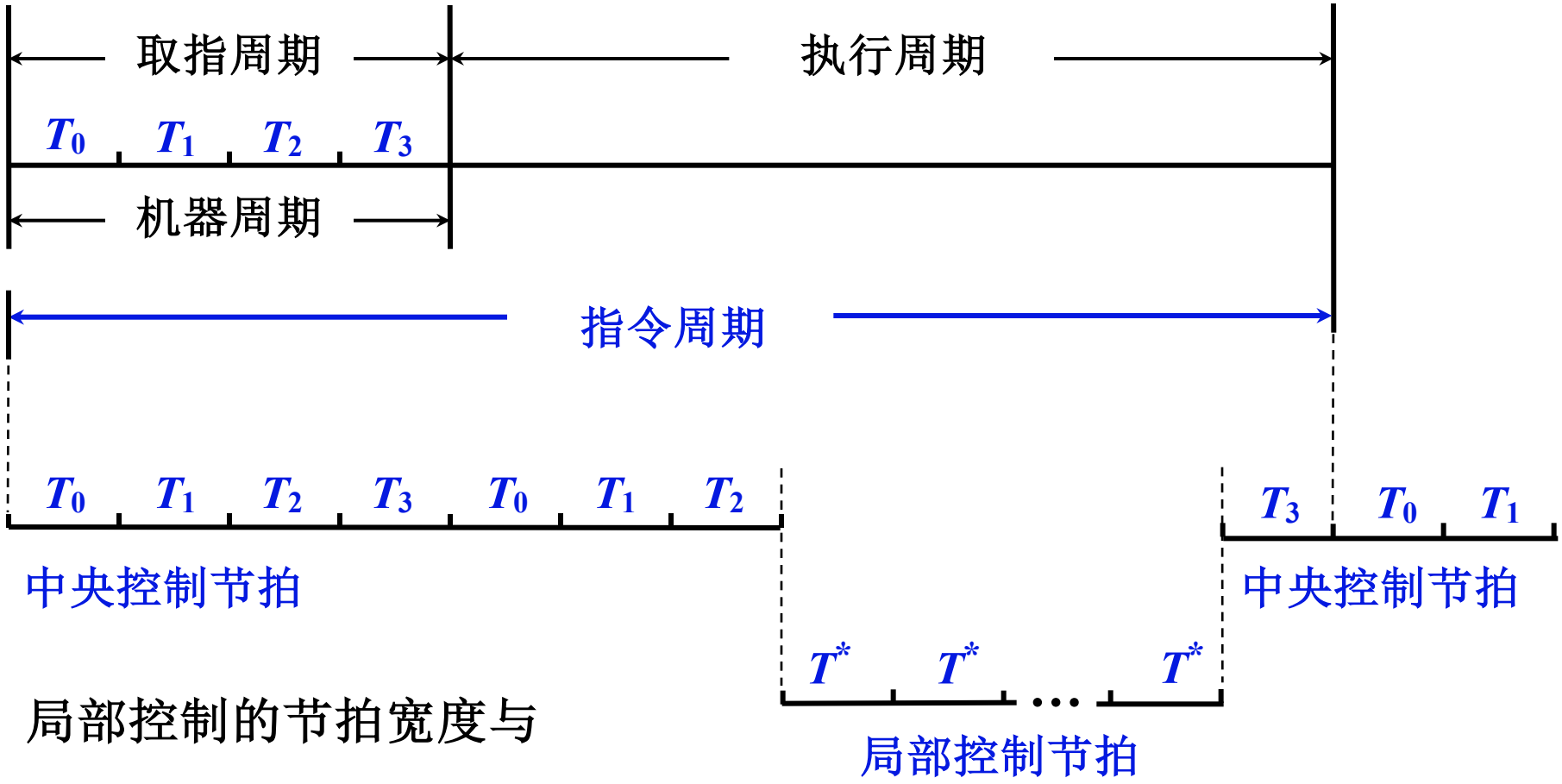
(2) 采用不定长的机器周期

机器周期内 节拍数不等





(3) 采用中央控制和局部控制相结合的方法 9.2



局部控制的节拍宽度与

中央控制的节拍宽度一致

2. 异步控制方式

无基准时标信号

无固定的周期节拍和严格的时钟同步

采用 应答方式

3. 联合控制方式

同步与异步相结合

4. 人工控制方式

(1) Reset

(2) 连续 和 单条 指令执行转换开关

(3) 符合停机开关