第5章 中央处理器

主要内容：

（一） CPU 的功能和基本结构

（二） 指令执行过程

（三） 数据通路的功能和基本结构

（四） 控制器的功能和工作原理

1. 硬布线控制器
2. 微程序控制器

微程序、微指令和微命令；微指令的编码方式；微地址的形成方式。

（五） 指令流水线

1. 指令流水线的基本概念
2. 超标量和动态流水线的基本概念

5.4 指令流水

如何提高机器速度

- 提高访存速度

  高速芯片　　Cache　　多体并行

- 提高 I/O 和主机之间的传送速度

  中断　　DMA　　通道　　I/O 处理机　　多总线

- 提高运算器速度

  高速芯片　　改进算法　　快速进位链

- **提高整机处理能力**

  高速器件　　改进系统结构，挖掘系统的并行性

系统的并行性

- 并行的概念

并行
- 并发：两个或两个以上事件在**同一时间段**发生
- 同时：两个或两个以上事件在**同一时刻**发生

- 并行性的等级

过程级（进程）	粗粒度	软件实现（单核处理器）
指令级（指令）	细粒度	硬件实现

流水线工作原理

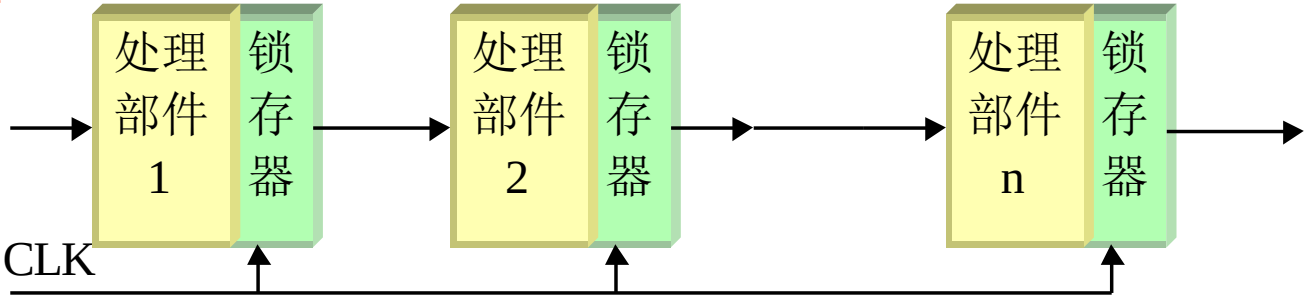
- 冯·诺依曼型计算机工作原理
  - 依序逐条串行执行程序指令，每条指令的各个操作也按顺序串行执行。例如，加法指令依序分成取指令/指令译码/取数操作/运算处理和写结果五个步骤。

取指$_1$ 译码$_1$ 取数$_1$ 运算$_1$ 存数$_1$
取指$_2$ 译码$_2$ 取数$_2$ 运算$_2$ 存数$_2$……

  - 特点：控制简单，速度低，各功能部件利用率低。
    例如，在取指令时主存忙，译码器和运算器等都空闲。
  - 若能把程序中的多条指令在时间上重叠起来执行，是否会显著提高机器速度呢？

流水线工作方式

- 计算机流水线（Pipelines）：将一个任务细分成若干个子任务，每个子任务由专门的部件处理，可与其他子任务并行进行处理。
- 流水线技术现已成为计算机中普遍使用的一种并行处理技术。



- 指令流水线：指令的执行过程采用流水线。
- 操作部件流水线：运算器中操作部件采用流水线。
- 宏流水线：多台计算机间通过存储器连接，采用流水线。
- 访存部件流水线

5.5.1 指令的重叠执行

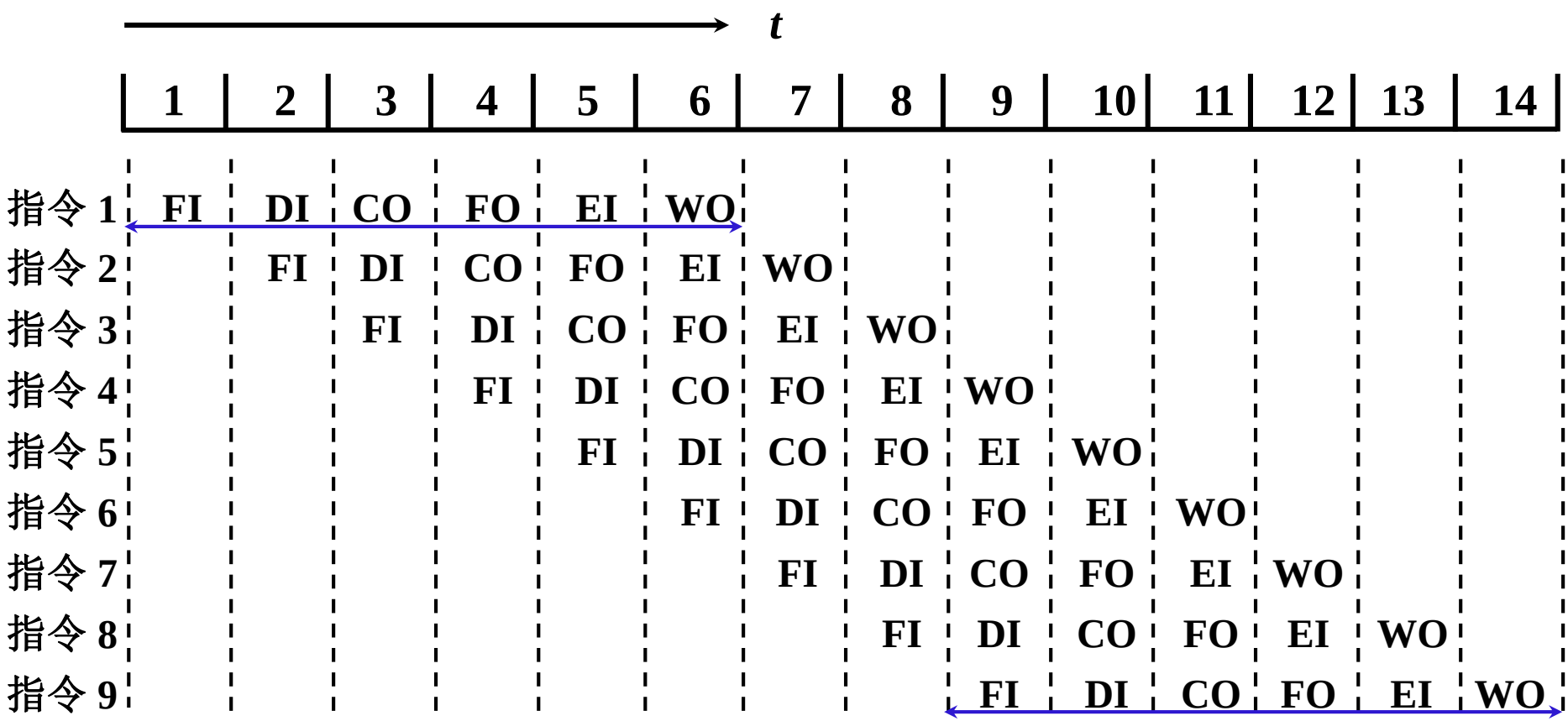
例：五条指令重叠执行情况。

T1	T2	T3	T4	T5	——机器执行时间			
取指1	译码1	取数1	运算1	存数1				
	取指2	译码2	取数2	运算2	存数2			
		取指3	译码3	取数3	运算3	存数3		
			取指4	译码4	取数4	运算4	存数4	
				取指5	译码5	取数5	运算5	存数5

- 可见，若将一条指令的执行时间分为5段，每段所用时间为T，则一条指令执行时间为5T。系统工作正常后每隔T时间就得到一条指令的处理结果。平均速度提高了4倍。这种工作方式称为流水线处理。 技术主要有三种形式：时间并行、 空间并行（资源重复）和时间并行+空间并行。

指令的六级流水

取指、译码、计算操作数地址、
取数、执行，写回



完成 一条指令

串行执行

六级流水

k=6 个时间单位，k级流水线

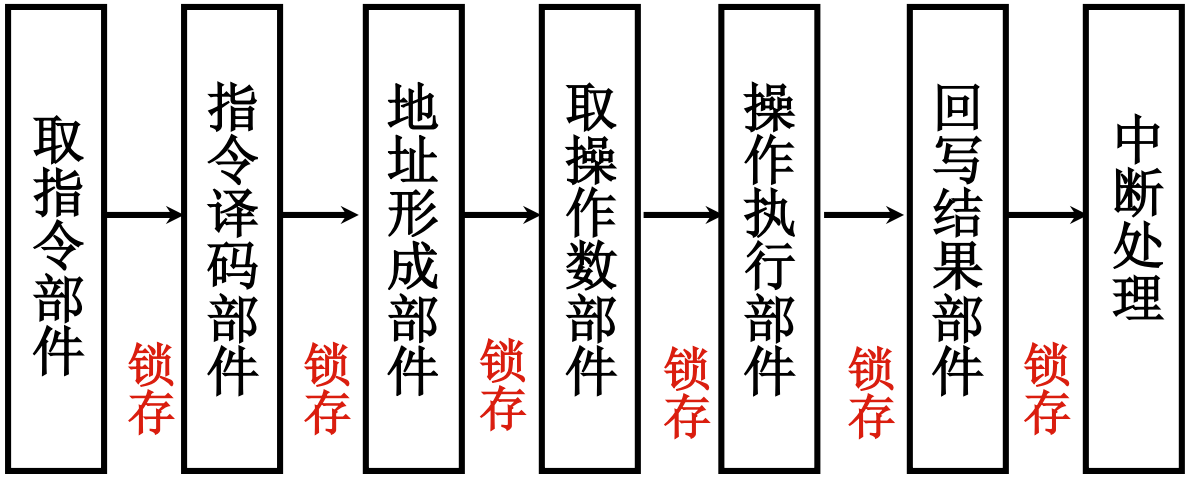
6 × 9 ＝ 54 时间单位，n=9 条指令

k+（n-1）=14 个时间单位

指令流水线结构

流水线的每一阶段完成一条指令的一部分功能，每一阶段称为一个流水阶段，或流水节拍、流水步、功能段、流水级等。

例：完成一条指令分 7 段， 每段需一个时钟周期



若 流水线不出现断流 1 个时钟周期出 1 结果

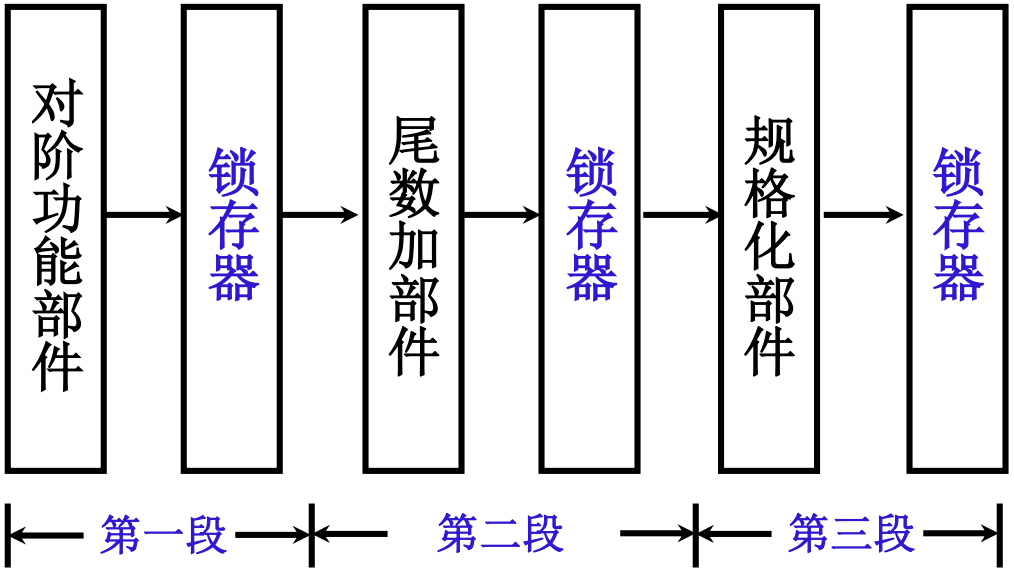
分段原则：每段操作时间尽量一致。

每个流水步的时间长度（时钟周期）应以最长功能段的执行时间为准，否则用时长的流水段的功能将不能正确完成。

运算流水线

完成 浮点加减 运算 可分

对阶、尾数求和、规格化 三段



流水线段数：根据总价、性能价格比选择最佳段数。
一般把大于等于8段的称为超流水线。

流水线性能

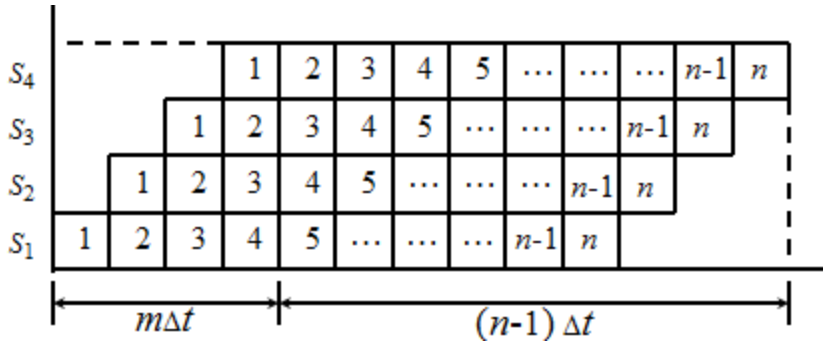
吞吐率

单位时间内 流水线所完成指令 或 输出结果 的 数量

设 m 段的流水线，各段时间为Δt

- 最大吞吐率

$$T_{pmax} = \frac{1}{\Delta t}$$



- 实际吞吐率

连续处理 n 条指令的吞吐率为

$$T_p = \frac{n}{m \cdot \Delta t + (n\text{-}1) \cdot \Delta\ t}$$

加速比 S_p

m 段的 流水线的速度 与等功能的 非流水线的速度 之比

设流水线各段时间为 Δt

完成 n 条指令在 m 段流水线上共需

$$T = m \cdot \Delta t + (n-1)\Delta t$$

完成 n 条指令在等效的非流水线上共需

$$T' = nm \cdot \Delta t$$

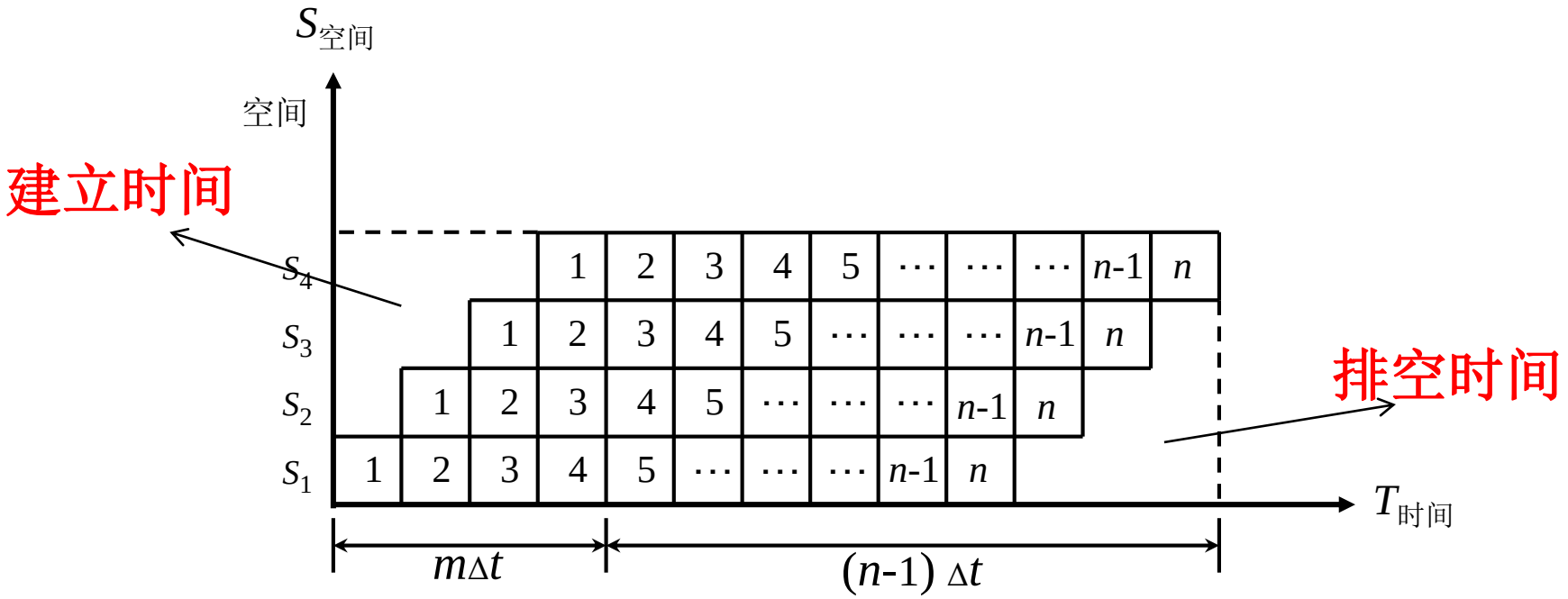
则

$$S_p = \frac{nm \cdot \Delta t}{m\Delta t + (n-1)\Delta t} = \frac{nm}{m+n-1}$$

效率

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

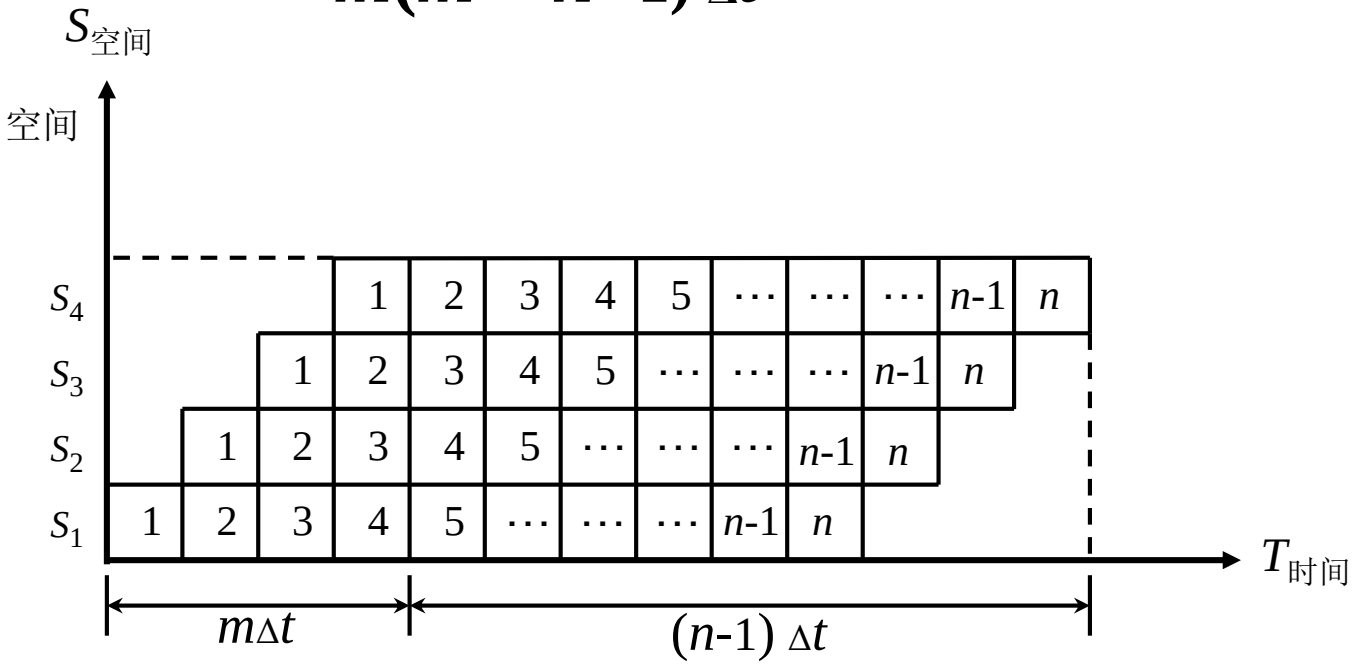


效率

流水线中各功能段的 利用率

$$效率 = \frac{流水线各段处于工作时间的时空区}{流水线中各段总的时空区}$$

$$= \frac{mn\Delta t}{m(m+n-1)\,\Delta t}$$

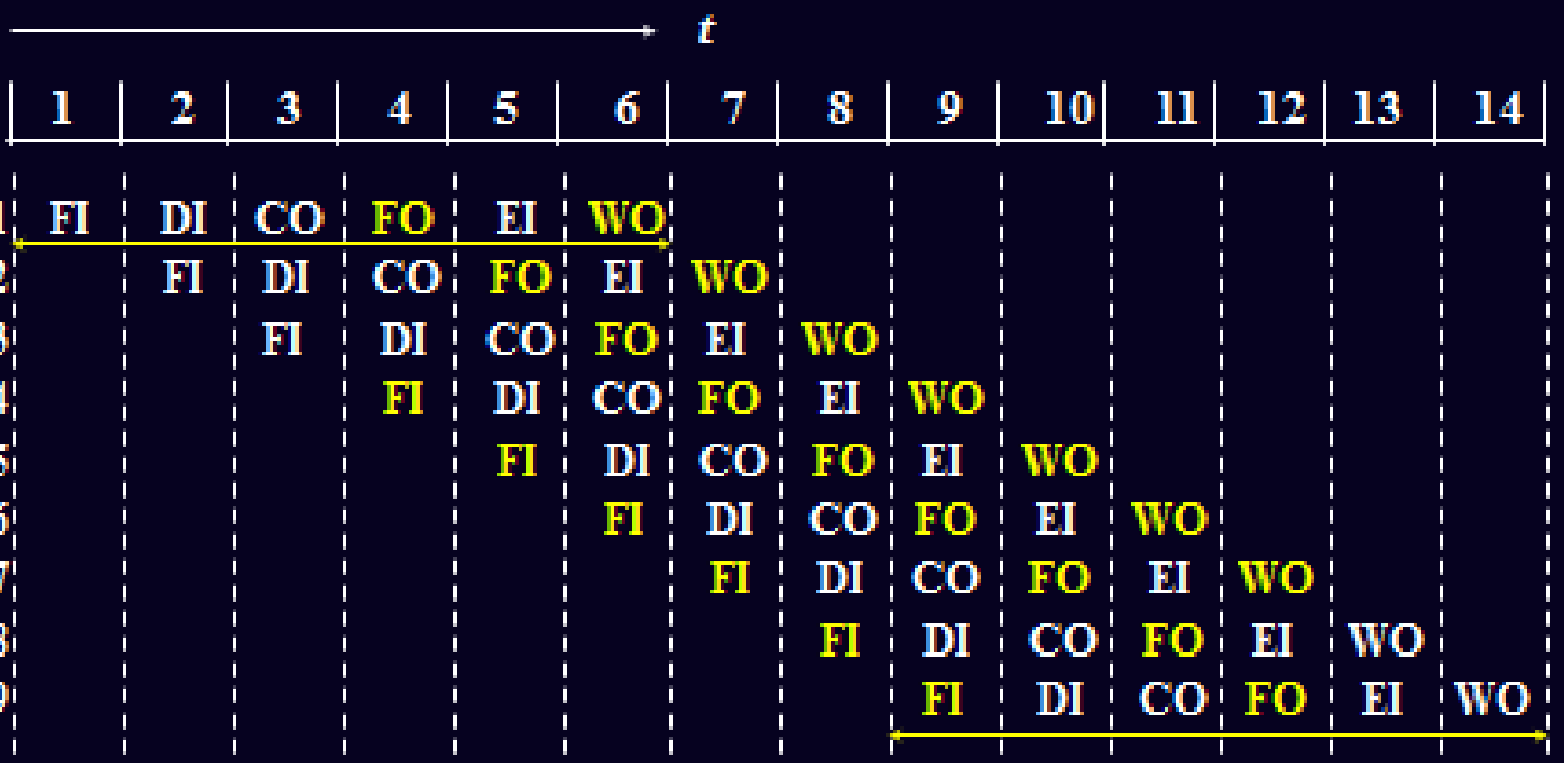


5.5.2 流水线的一些问题解决

流水线把取指与执行分开，使取指与执行同时进行，减少了取指等待时间，大大提高了CPU的利用率。同时降低了对与之匹配的存储器的存取速度要求。但流水线处理方式也存在一些困难：

- **结构相关：**由于多条指令在同一时刻争用同一资源而形成的冲突称为结构相关。
- **数据相关：**后续指令要使用前面指令的操作结果，而这一结果尚未产生或未送到，就出现了流水线阻塞，称为数据相关。
  - 两级流水线不存在数据相关现象。
- **控制相关：**在遇到条件转移指令时，存在是顺序执行还是转移执行两种可能。

1）. 结构相关 不同指令争用同一功能部件产生资源冲突



解决办法

取指、译码、计算操作数地址、取数、执行，写回

- 停顿
- 指令存储器和数据存储器分开，多部件
- 指令预取技术（适用于访存周期短的情况）

2）. 数据相关

读/写运算器部件中的通用寄存器组、读写存储器中的数据都可能遇到数据相关问题。

不同指令因重叠操作，可能改变操作数的 读/写 访问顺序

- 写后读相关（RAW）

SUB R_0，R_1，R_2 ；（R_1）－（R_2）→R_0

ADD R_4，R_5，R_0 ；（R_5）+（R_0）→R_4

- 读后写相关（WAR）

STA M，R_2 ；（R_2）→ M 存储单元

ADD R_2，R_4，R_5 ；（R_4）+（R_5）→R_2

- 写后写相关（WAW）

MUL R_3，R_2，R_1 ；（R_2）×（R_1）→R_3

SUB R_3，R_4，R_5 ；（R_4）－（R_5）→ R_3

2）. 数据相关

读/写运算器部件中的通用寄存器组、读写存储器中的数据都可能遇到数据相关问题。

不同指令因重叠操作，可能改变操作数的 读/写 访问顺序

- 写后读相关（RAW）

SUB R_1，R_2，R_3 ；（R_2）－（R_3）→R_1

ADD R_4，R_5，R_1 ；（R_5）+（R_1）→R_4

- 读后写相关（WAR）

STA M，R_2 ；（R_2）→ M 存储单元

ADD R_2，R_4，R_5 ；（R_4）+（R_5）→R_2

- 写后写相关（WAW）

MUL R_3，R_2，R_1 ；（R_2）×（R_1）→R_3

SUB R_3，R_4，R_5 ；（R_4）－（R_5）→ R_3

取指	译码	执行	写回
	取指	译码	执行	写回

解决办法

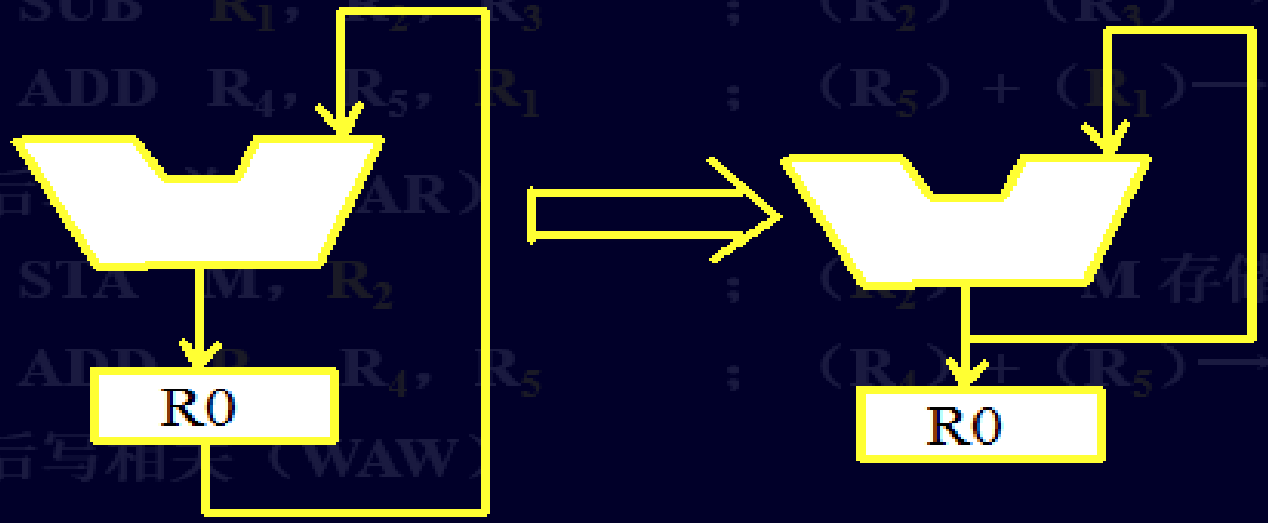
- 延迟（后推法、停顿法）
- 采用 旁路技术

• 采用 旁路技术

R2+R1->R0
R0+R5->R4

ADD R0, R1, R2
ADD R4, R5, R0

第一条指令的和已经计算出
从加法器直接取数



取指	译码	取数	执行	写回
	取指	译码	取数	执行	写回

取指	译码	取数	执行	写回
	取指	译码	取数	执行	写回

- 延迟法、后推法、停顿法

3）控制相关

由转移指令引起

```
      LDA    # 0
      LDX    # 0
M     ADD    X, D
      INX
      CPX    # N
      BNE    M
      DIV    # N
      STA    ANS
```

DIV 指令必须等BNE 指令的结果才能确定是否转移

举例

设 指令3 是条件转移指令，
等到指令2执行完毕后才知道转移地址

分支预测法

例题

- 如将一个指令周期分成取指（IF）、译码（ID）、执行（EX）、访存（MEM）和回写（WR）五个阶段，就形成了五级流水线。在流水线中，当遇到访存冲突（取指令、取操作数、存结果都需要访存）、数据相关（相邻指令共用同一个存储单元或寄存器）以及控制相关（条件转移指令需等到上一条指令产生结果才能确定转移方向）时，流水线会受阻。为了解决这个问题一般需要增加延时等待方式或增加硬件资源方式。

结构相关

- 结构相关是当指令在重复执行过程中，不同指令争用同一功能部件产生资源冲突时产生的，故也称为资源相关。举例如下：

指　　令	T1	T2	T3	T4	T5	T6	T7	T8
Load 指令	IF	ID	EX	MEM	WB			
指令 i+1		IF	ID	EX	MEM	WB		
指令 i+2			IF	ID	EX	MEM	WB	
指令 i+3				IF	ID	EX	MEM	WB
指令 i+4					IF	ID	EX	MEM

- 在T4时load指令中的访存操作MEM与指令i+3中的取指令操作IF发生访存冲突，解决办法需要指令i+3的IF操作暂缓一个周期。

- 解决访存冲突的另一种方法是设置两个独立的存储器分别存放操作数和指令，以免取指令和取操作数同时进行相互冲突。

指　　令	T1	T2	T3	T4	T5	T6	T7	T8	T9
Load指令	IF	ID	EX	MEM	WB				
指令 i+1		IF	ID	EX	MEM	WB			
指令 i+2			IF	ID	EX	MEM	WB		
指令 i+3				停顿	IF	ID	EX	MEM	WB
指令 i+4						IF	ID	EX	MEM

数据相关

- 数据相关是流水线中的各条指令因重叠操作，可能改变对操作数的读写访问顺序，从而导致了数据相关冲突。例如，流水线要执行以下二条指令：

  ADD R1，R2，R3　;(R2)+(R3)→R1

  SUB R4，R1，R5　;(R1)-(R5) →R4

- 这里第二条SUB指令中R1的内容必须是第一条ADD指令的执行结果。其流水线执行顺序如下表所示：

指　令	T1	T2	T3	T4	T5	T6
ADD	IF	ID	EX	MEM	WB	
SUB		IF	ID	EX	MEM	WB

- 指令ADD在T5时需要写入R1，但SUB指令在T3时要从R1中读数，使先写后读的顺序改变为先读后写，这样发生了数据相关冲突。如果不采取措施，就会使操作结果出错。解决数据相关的方法可以采用后推法（延迟等待），如下表所示：

指　令	T1	T2	T3	T4	T5	T6	T7	T8	T9
ADD	IF	ID	EX	MEM	WB				
SUB		IF				ID	EX	MEM	WB

控制相关

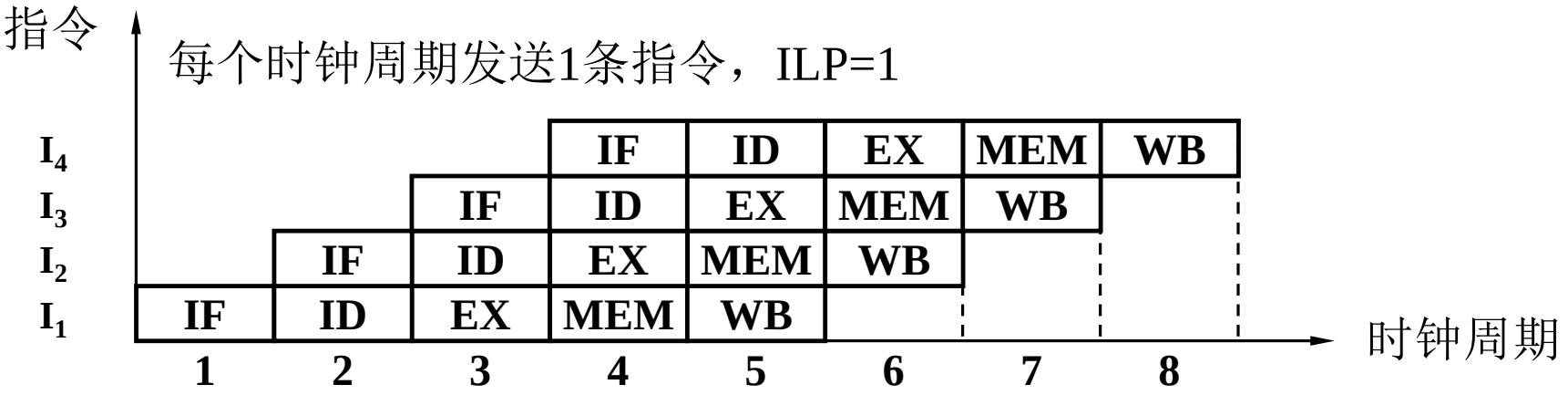
- 控制相关主要是由转移指令引起的。
- 为了解决控制相关一般采用预测方法。

5.4.4 指令级并行技术

- ILP(Instruction Level Parallelism)：指令级并行度，在一个时钟周期内流水线上流出的指令数。

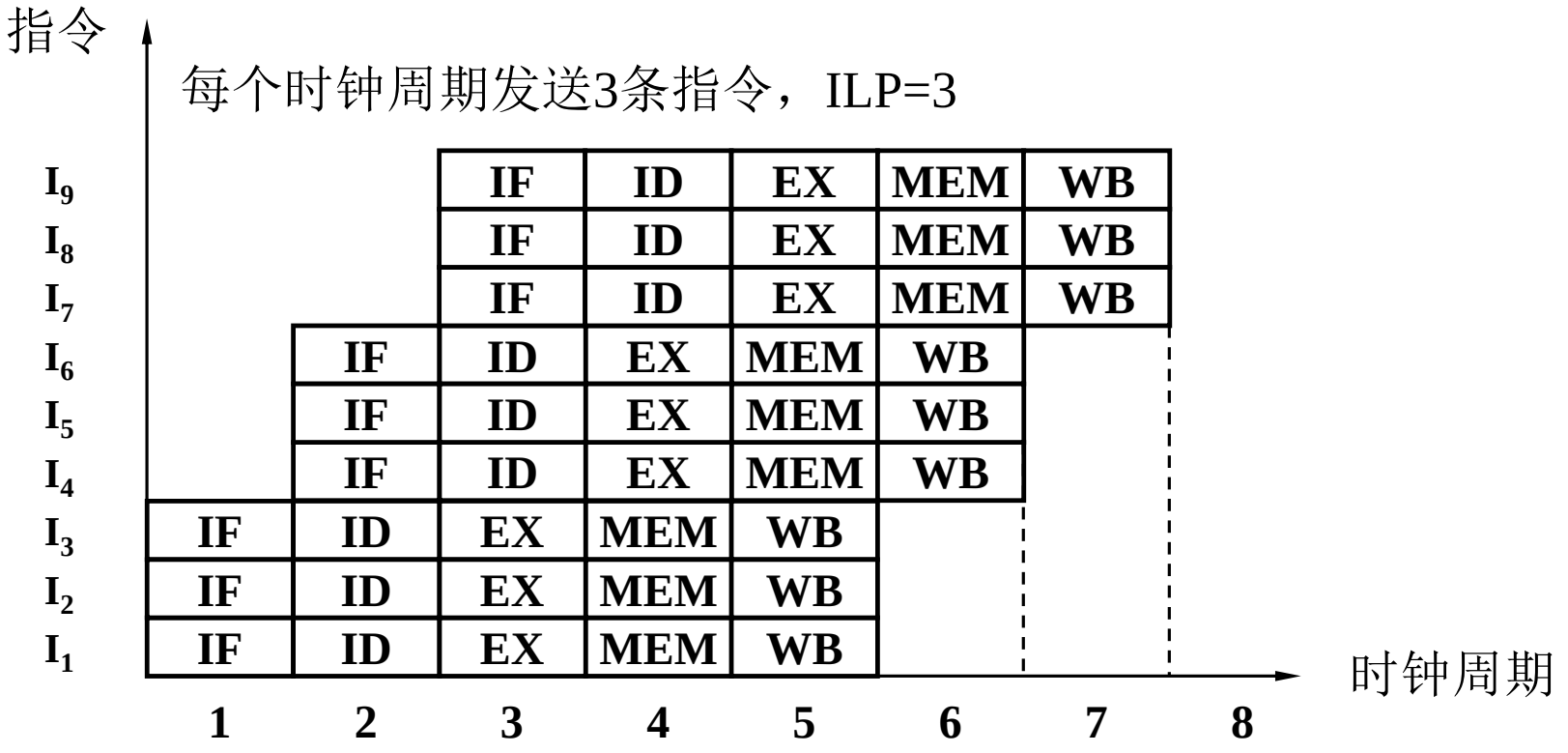
- CPI (Clock cycles Per Instruction)：流水线中平均执行一条指令所需的时钟周期数。
  - CPI是衡量指令级并行性的一个指标。

例：假设流水线有K个流水段，一个程序执行时共执行n条指令，CPI=（k+n-1）/n，当n>>k时， CPI≈ 1。



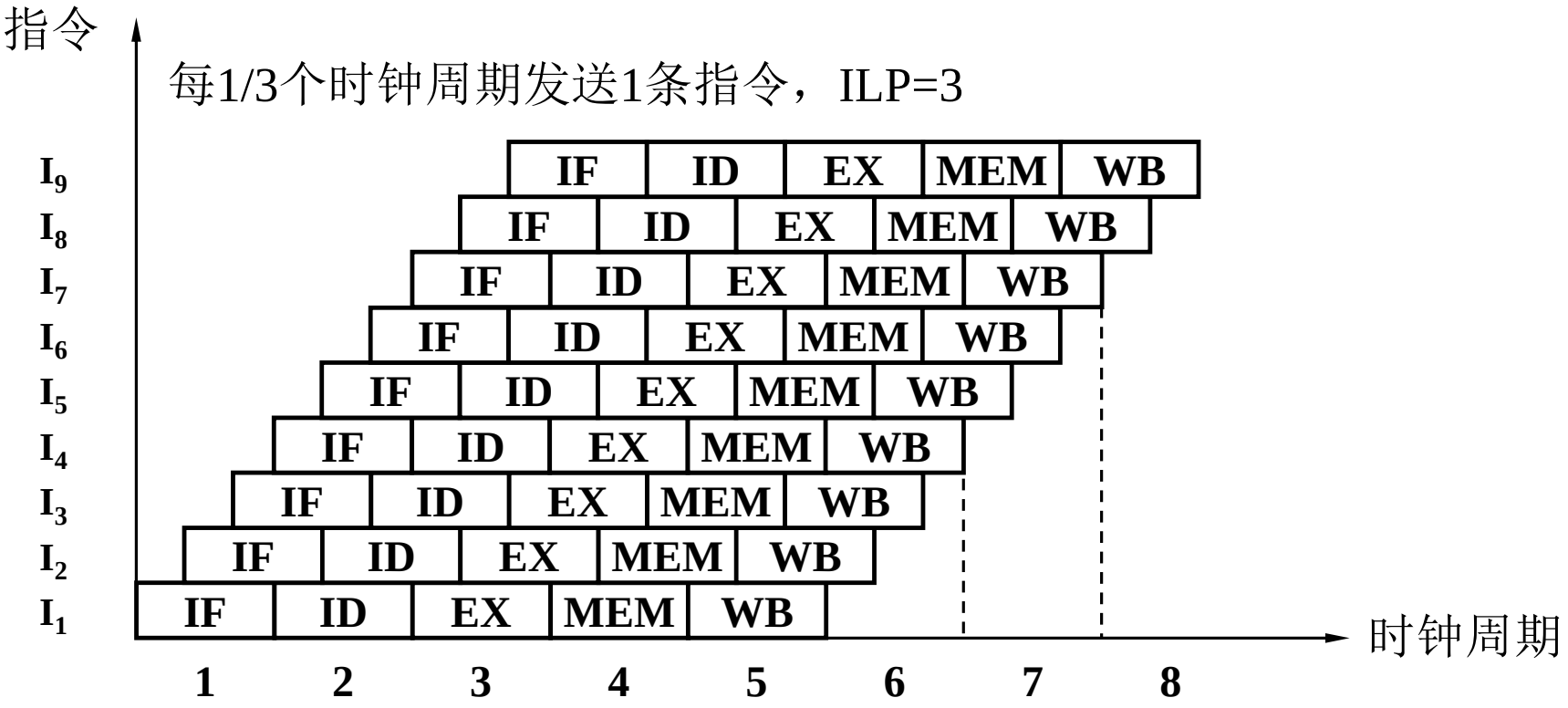
(a) 基准标量处理机时空图

- **超标量**（superscalar）CPU：集成了多个ALU、多个FPU、多个译码器和多条流水线的CPU，在一个时钟周期同时发送多条指令（CPI<1），以并行处理的方式来提高CPU的性能。Pentium 4 就有20级超标量流水线。



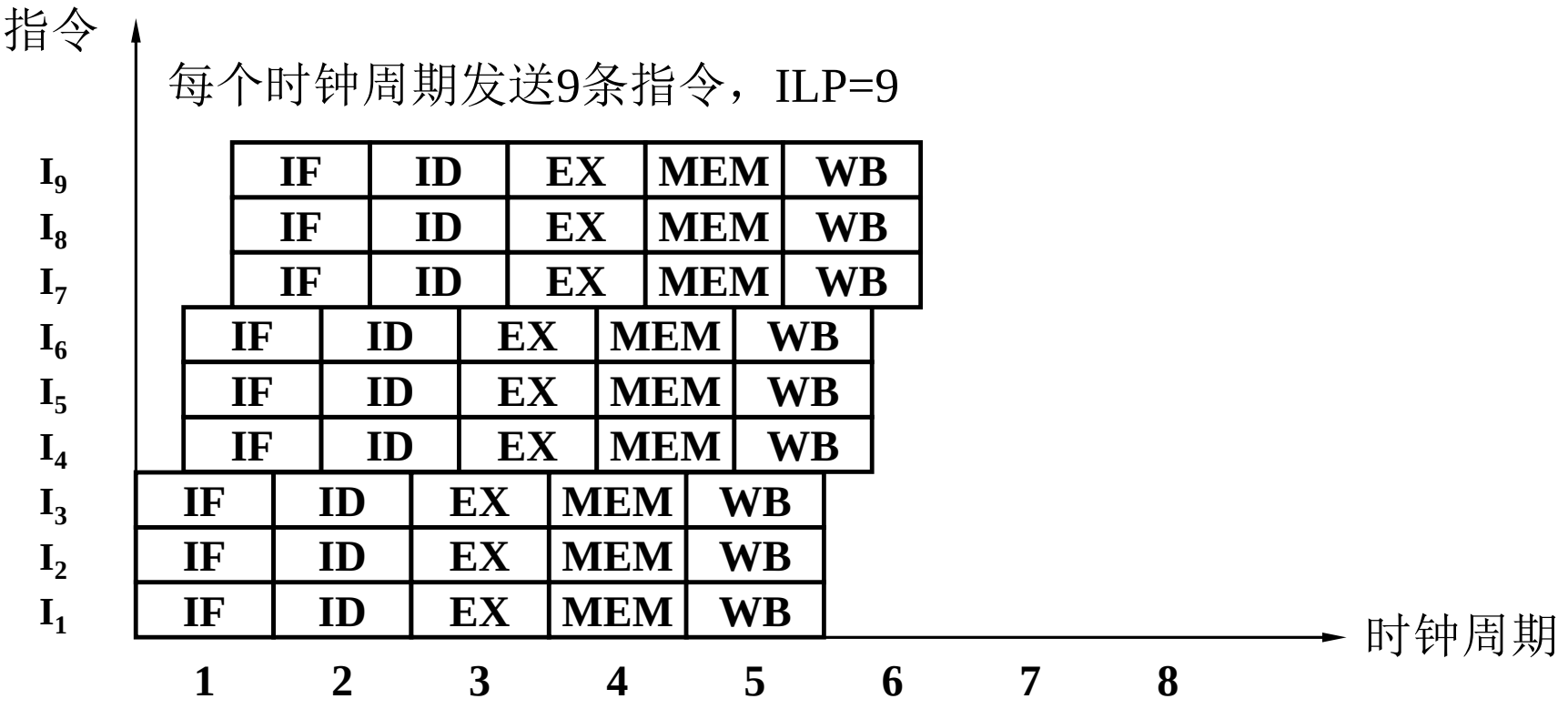
(b) 超标量处理机时空图

- **超流水线**（superpipelining）CPU：在一个时钟周期分期发送多条指令。



(c) 超流水线处理机时空图

- **超标量超流水线**（ Superscalar Superpipelining）处理机集中了超标量和超流水线两者的特点。



(d) 超标量超流水线处理机时空图

四种不同类型处理机的性能比较表

	普通标量处理机	超标量处理机	超流水线处理机	超标量流水线处理机
机器流水线周期	1个	1个	1/n个	1/n个
同时发送指令条数	1条	m条	1条	m条
指令级并行度(ILP)	1	m	n	m•n

本章小结

1、建立CPU整机概念 { 逻辑组成 ; 工作机制 }

（1）逻辑组成

寄存器、ALU设置，重在数据通路结构与功能

（2）工作机制

指令的执行过程 { 寄存器传送级：各类指令的流程 ; 微操作控制级：微命令序列 }

拟定流程的关键：清楚了解数据通路结构

熟练掌握基本寻址方式

2、有关的基本概念

（1） 微命令的产生方式

微程序控制器：控制存储器、微程序、微指令、微命令指令格式与编码方式；微地址的形成方式。基本工作原理 、优缺点、应用场合。

硬布线(组合逻辑)控制器：基本工作原理 、优缺点、应用场合

（2） 时序控制方式

同步控制方式 ：　　定义、特点、应用场合

异步控制方式：　　定义、特点、应用场合

3、指令流水线、超标量的基本概念

44. （12分）某16位计算机中，带符号整数用补码表示，数据Cache和指令Cache分离。题44表给出了指令系统中部分指令格式，其中Rs和Rd表示寄存器，mem表示存储单元地址，（x）表示寄存器x或存储单元x的内容。

题44表　指令系统中部分指令格式

名　称	指令的汇编格式	指令功能
加法指令	ADD Rs，Rd	(Rs)+(Rd)→Rd
算数/逻辑左移	SHL Rd	2*(Rd)→Rd
算数右移	SHR Rd	(Rd)/2→Rd
取数指令	LOAD Rd,mem	(mem)→Rd
存数指令	STORE Rs,mem	(Rs)→mem

该计算机采用5级流水方式执行指令，各流水线段分别是取址（IF）、译码/读寄存器（ID）、执行/计算有效地址（EX）、访问存储器（M）和结果写回寄存器（WB），流水线采用“按序发射，按序完成”方式，没有采用转发技术处理数据相关，并且同一个寄存器的读和写操作不能在同一个时钟周期内进行。请回答下列问题。

（1）若int型变量x的值为-513，存放在寄存器R1中，则执行指令“SHR　R1”后，R1的内容是多少？（用十六进制表示）

（2）若某个时间段中，有连续的4条指令进入流水线，在其执行过程中没有发生任何阻塞，则执行这4条指令所需的时钟周期数为多少？

（3）若高级语言程序中某赋值语句x =a + b ，x、a和b都为int型变量，它们的存储单元地址分别为[x]、[a]和[b]。该语句对应的指令序列及其在指令流水线中的执行过程如题44图所示。

I_1 LOAD R1, [a]

I_2 LOAD R2, [b]

I_3 ADD R1, R2

I_4 STORE R2, [x]

时间单元

指令	1	2	3	4	5	6	7	8	9	10	11	12	13	14
I_1	IF	ID	EX	M	WB									
I_2		IF	ID	EX	M	WB								
I_3			IF				ID	EX	M	WB				
I_4							IF				ID	EX	M	WB

则这4条指令执行过程中，I_3 的ID段和I_4 的IF段被堵塞的原因是什么？

（4）若高级语言程序中某赋值语句为x = 2*x+a, x和a均为unsigned int类型变量，它们的存储单元地址分别表示为[x]、[a]，则执行这条语句至少需要多少个时钟周期？要求模仿题44图画出这条语句对应的指令序列及其在流水线中执行过程示意图。

I_1 LOAD R1，[x]
I_2 LOAD R2，[a]
I_3 SHL R1 //或者 ADD R1 ,R1
I_4 ADD R1，R2
I_5 STORE R2，[x]

44.【答案要点】

（1）x的机器码为：$[x]_{补}$ = 1111 1101 1111 1111B，即执行指令前(R1) = FDFFH，右移1位后为1111 1110 1111 1111B，即指令执行后（R1）= FEFFH。

（2）至少需要5 +（4-1）= 8个时钟周期。

（3）I_3 的ID段被堵塞的原因：因为I_3 与I_1 和I_2都存在数据相关，需等到I_1 和I_2将结果写回寄存器后，I_3才能读寄存器内容，所以I_3 的ID被堵塞。

I_4的IF段被堵塞的原因：因为I_4的前一条指令I_3在ID段被堵塞，所以I_4的IF段被堵塞。

I_4 的ID段被堵塞的原因与I_3 的ID段被堵塞的原因相似。

（4）因2*x操作有左移和加法两种实现方法，故x = 2*x + a对应的指令序列为

I_1 LOAD R1，[x]

I_2 LOAD R2，[a]

I_3 SHL R1 //或者 ADD R1 ,R1

I_4 ADD R1，R2

I_5 STORE R2，[x]

这5条指令在流水线中的执行过程如下图所示。（3分）

故执行x = 2*x + a指令至少需要17个时钟周期。

指令	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
I_1	IF	ID	EX	M	WB												
I_2		IF	ID	EX	M	WB											
I_3			IF			ID	EX	M	WB								
I_4						IF				ID	EX	M	WB				
I_5										IF				ID	EX	M	WB