# 第 5 章　流水线和向量处理机

本章着重讨论重叠和流水等控制方式的基本原理。讲述向量的流水处理方式、向量流水机的结构。最后介绍在指令级上发展高度并行的超标量处理机、超流水线处理机、超标量超流水线处理机及超长指令字处理机的工作原理。

## 5.0　学习指南

### 1. 知识点和学习要求

- 重叠方式。

领会顺序与重叠方式的定义和特点，以及“一次重叠”方式的含义及好处。

了解对有关条件转移指令与后继指令之间的相关、指令相关、主存数相关、通用寄存器组的数相关和变(基)址值相关等的定义及各自的处理方法。

了解设置相关专用通路的目的及其适用的场合。

在给出指令之间各种微操作时间重叠关系的要求后，能熟练计算连续执行完若干条指令所需要的时间。

- 流水方式。

领会流水方式的工作原理。

了解从不同角度对流水线的分类和定义。

熟练掌握流水线时空图的画法，能够计算出流水线的最大吞吐率、实际吞吐率、效率和加速比。

掌握消除流水线瓶颈两种方法的时空图画法，能计算出吞吐率和效率。

了解同步流动和异步流动的区别。

熟练掌握在单功能非线性流水线上，任务流入流水线的最佳调度方案，求出极限吞吐率。按此方案实际调度若干条指令，画出时空图，并求出此时实际的吞吐率和效率。

- 向量的流水处理与向量处理机。

了解处理向量的三种方式和向量的流水处理含义。

- 指令级高度并行的超级处理机。

了解超标量、超流水线、超标量超流水线、超长指令字处理机在指令级并行的工作原理。给出指令数和并行的度数，能画出各自的工作时空图，计算出所需时间、加速比。

### 2. 重点和难点

本章的重点：重叠解释方式；“一次重叠”方式中各种相关的处理；流水线的分类；流水线的性能(吞吐率、加速比、效率)分析及时空图；流水线处理机的主要性能；流水线瓶颈

段的处理；单功能非线性流水线的调度；向量的流水处理；向量流水处理机；向量处理方式；超标量处理机、超流水线处理机、超标量超流水线处理机的指令执行时序及性能。

本章的难点：针对所要求的时间重叠关系，计算全部指令完成的时间；为消除流水线瓶颈，所采取的措施及相应的流水线时空图画法，吞吐率和效率的计算；优化单功能非线性流水线的调度方案；在超标量、超流水线、超标量超流水线、超长指令字处理机上，给出指令数和并行的度数，画出时空图，计算加速比。

## 5.1 重叠方式

### 5.1.1 重叠原理和一次重叠

一条指令的执行过程可以分成多个子过程，具体分法依据各种处理器的设计。图 5-1 中对一条机器指令的执行分三个子过程，即取指令(简称取指)、分析取操作数(简称分析)、执行把结果送回(简称执行)。

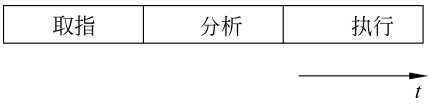



图 5-1 一条指令的执行过程

指令的顺序执行方式指的是指令与指令之间顺序串行，指令内的各个微操作之间也是顺序串行的。顺序执行方式执行 $n$ 条指令所用的时间为

$$T=\sum_{i=1}^{n}(t_{取指i}+t_{分析i}+t_{执行i})$$

如果取指、分析和执行指令的时间都相等，每段时间都为 $t$，则执行 $n$ 条指令所用的时间为 $T=3nt$。

采用顺序执行方式的主要优点是：控制简单，节省设备，转入下条指令的时间易于控制。主要缺点是：执行指令的速度慢，功能部件的利用率很低。只有上一条指令执行完，下一条指令才能执行。在取指和分析指令时，主存是忙碌的，而执行部件是空闲的。

指令的重叠解释方式是在相邻的指令之间，让取指、分析、执行各部分的操作在时间上重叠地进行，而指令内部的微操作仍然是顺序串行的。重叠解释虽不能加快每条指令的解释，但却能加快相邻两条以至整段程序的解释，使系统的性能价格比有显著提高。

一次重叠执行方式是一种最简单的流水线方式。指令执行过程分成三个子过程，如果各子过程的时间相等，把执行第 $k$ 条指令与取第 $k+1$ 条指令同时执行，如图 5-2 所示，则执行 $n$ 条指令的时间为 $T=(1+2n)t$。与顺序执行相比时间缩短近三分之一。

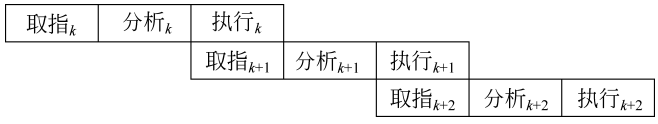



图 5-2 一次重叠执行方式

采用一次重叠方式的主要优点是：指令的执行时间缩短，功能部件的利用率明显提高，主存基本上处于忙碌状态。主要缺点是：需要增加一些硬件，控制过程稍复杂。

如果每次都可以从指令缓冲器中取得指令,则“取指”的时间很短,可把这个微操作合并到“分析”内,从而由原先的“取指$_{k+2}$”“分析$_{k+1}$”“执行$_k$”重叠变成只有“分析$_{k+1}$”与“执行$_k$”的重叠。这种在任何时间都只有“执行$_k$”和“分析$_{k+1}$”在时间上重叠的一次重叠执行方式如图 5-3 所示。

为了实现“执行$_k$”与“分析$_{k+1}$”的重叠,硬件上应有独立的指令分析部件和指令执行部件。为了使“一次重叠”方式的系统有较高的重叠效率,应使“分析”和“执行”的时间尽可能等长。

为了进一步提高指令的执行速度,采用二次重叠执行方式如图 5-4 所示,“取指$_{k+2}$”“分析$_{k+1}$”和“执行$_k$”重叠。如果执行一条指令的三个子过程的时间相等,执行 $n$ 条指令的时间为 $T=(2+n)t$。进一步提高执行速度,执行时间比顺序执行缩短近三分之二。

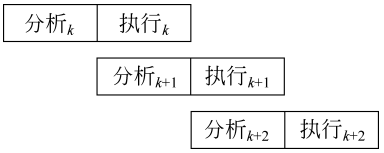



图 5-3　两功能段一次重叠执行方式

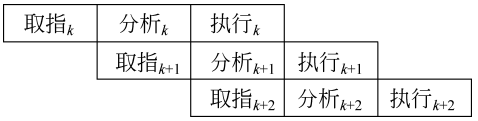



图 5-4　二次重叠执行方式

理想情况下处理机同时有三条指令在执行,处理机的结构要做比较大的改变,必须采用先行控制方式。采用二次重叠执行方式重叠必须解决以下两个问题。

(1) 要有独立的取指令部件、指令分析部件和指令执行部件。三个子过程都有独立的部件,独立的控制器包括存储控制器、指令控制器和运算控制器。

(2) 要解决访问主存储器的冲突问题。

取指令、分析指令、执行指令都可能要访问存储器,要解决主存能否同时被访问的冲突问题。

假设解释一条机器指令的微操作被分为取指、分析和执行三个部分,这三个部分都可能需要访问主存储器。为解决“取指$_{k+2}$”“分析$_{k+1}$”“执行$_k$”在重叠时的访存冲突,可采取的解决访存冲突方法有:

(1) 分别设置各自独立编址的数据存储器和指令存储器,让存、取操作数和取指令可同时访存。其不足之处在于增加了总线控制和软件设计的负担。

(2) 仍维持指令和数据混存,但采用多体交叉主存结构,只要第 $k$ 条指令的操作数与第 $k+1$ 条指令不在同一个存储体内,仍可在一个主存周期内取得。其不足之处在于可能会发生分体冲突。

① 采用低位交叉存取方式。

这种方法不能根本解决冲突问题,有取指令、读操作数、写结果的访存冲突。存储器分成几个模块,连续单元在不同模块,但要求三个操作在不同模块。

② 两个独立的存储器:独立的指令存储器和数据存储器。

如果再规定,执行指令所需要的操作数和执行结果只写到通用寄存器,那么取指令、分析指令和执行指令就可以同时进行了。

在许多高性能处理机中,有独立的指令 Cache 和数据 Cache,这种结构被称为哈佛结

构。这种结构取指令和取数据不会冲突。

③ 采用先行控制技术。

先行控制技术的关键是缓冲技术和预处理技术。缓冲技术是在工作速度不固定的两个功能部件之间设置缓冲栈,用以平滑它们的工作。预处理技术是早些取来指令和操作数。在采用了缓冲技术和预处理技术之后,运算器能够专心于数据的运算,从而大幅度提高程序的执行速度。增加了指令缓冲栈、数据缓冲栈,执行速度快了。可以把取指和分析结合在一起,每次从指缓中取指令,取指时间很短。

(3) 增设采用先进先出方式工作的指令缓冲寄存器,让主存利用空闲时间将预取的指令存入指令缓冲器。

### 5.1.2 相关处理

因机器语言程序中邻近指令之间出现了关联,为防止出错让它们不能同时解释的现象称为发生了“相关”。相关又可细分为数据相关和指令相关。

数据相关是指第 $k$ 指令和第 $k+1$ 指令的数据地址之间有关联。例如,第 $k+1$ 条指令的源操作数地址正好是第 $k$ 条指令存放运算结果的地址。数据相关不仅会发生在主存空间,也会发生在通用寄存器空间。

指令相关是因为指令在程序的执行过程中允许被修改造成的。例如,经第 $k$ 条指令的执行形成第 $k+1$ 条指令。

无论发生何种相关,或者使解释出错,或者使重叠效率显著下降,都必须加以正确处理。

#### 1. 指令相关的处理

当条件转移成功时,重叠效率会下降。假设第 $k$ 条指令是条件转移指令,并成功转移到第 $m$ 条指令去。如果第 $m$ 条指令已在指令缓冲器(简称指缓)中,则在“分析$_{k+1}$”之后接着“分析$_m$”和“执行$_m$”;若第 $m$ 条指令还没取到指令缓冲器,则在“分析$_{k+1}$”之后还需先进行“取指$_m$”才能“分析$_m$”和“执行$_m$”。图 5-5 给出第 $k$ 条指令是条件转移指令时的时间关系示意图。条件转移成功时,重叠实际变成了顺序执行。

如果冯·诺依曼(Von Neumann)型机器上指令可修改的办法是经第 $k$ 条指令的执行来形成第 $k+1$ 条指令,例如:

```
k:      STORE 通用寄存器,k+1          ;(通用寄存器)→k+1
k+1:    …
```

由于在“执行$_k$”的末尾才形成第 $k+1$ 条指令,按照一次重叠的时间关系,“分析$_{k+1}$”所分析的是早已预取进指缓的第 $k+1$ 条指令的旧内容,这就会出错。为了避免出错,第 $k$ 条和第 $k+1$ 条指令就不能同时解释,称此时这两条指令之间发生了“指令相关”。特别是当指令缓冲器可缓冲存放 $n$ 条指令的情况下,执行到第 $k$ 条指令时,与已预取进指缓的第 $k+1$ 到第 $k+n$ 条指令都有可能发生指令相关。指缓的容量越大,或者说指令预处

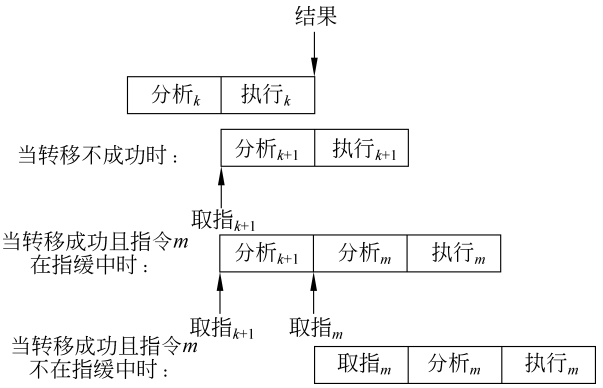



图 5-5　第 $k$ 条指令是条件转移指令时的时间关系

理能力越强的机器发生指令相关的概率就越高。

重叠方式的机器在程序中应尽量减少使用条件转移指令，在需要使用条件转移指令时，可采用“延迟转移”技术，由编译程序将条件转移指令与其前面的指令交换位置，可使重叠效率不下降。对于指令相关：

① 规定指令在执行过程中不允许修改。

② 在需要修改指令时，可以设置类似 IBM 370 的“执行”指令，将指令相关转成操作数相关，统一按操作数相关来处理。

“执行”指令是 IBM 370 机器为此设置的一条指令，其形式为

| 执行 | $R_1$ | $X_2$ | $B_2$ | $D_2$ |
|---|---|---|---|---|

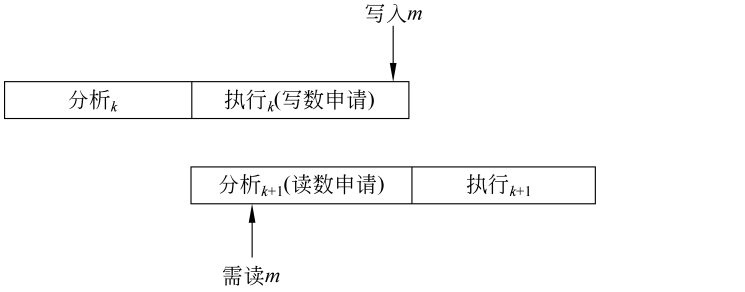
当执行到“执行”指令时，将第二操作数地址($X_2$)+($B_2$)+$D_2$ 取出操作数区中单元的内容作为指令来执行。指令相关是因为机器指令允许被修改引出的。被修改的指令是以“执行”指令的操作数形式出现的，将指令相关转化成了数据相关，统一按数据相关进行处理。

### 2. 主存空间数据相关的处理

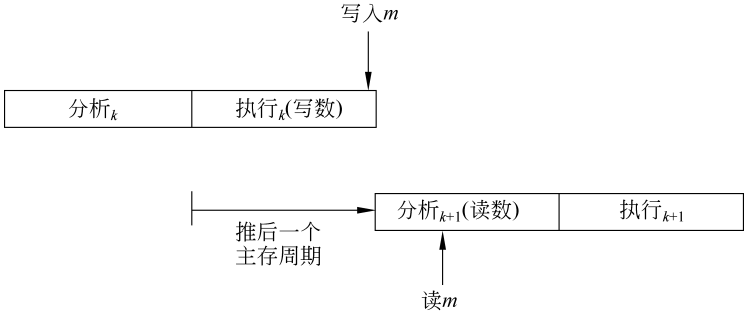
主存空间数据相关是相邻两条指令之间出现对同一单元要求先写而后读的关联，可采取推后后续指令对相关单元的读操作的方法。例如，在存储器的控制器内，让访存的“写”申请优先于“读”申请得到响应。主存数据相关的处理如图 5-6 所示，第 $k+1$ 条指令分析读出的 $m$ 不是第 $k$ 条指令写入的结果，对 $m$ 存在先写后读的关联，就可采用推后读进行处理。

### 3. 通用寄存器组相关的处理

通用寄存器一般除了放操作数、运算结果外，也可能放形成访存操作数物理地址的变址值或基址值，因此，通用寄存器组的相关又有操作数的相关和变址值/基址值的相关两种。设机器的基本指令格式为




(a) 主存数相关的时间关系




(b) 由存控推后“分析$_{k+1}$”的读

图 5-6　主存数据相关的处理

| 操作码 | $L_1$ | $L_3$ | $B_2$ | $d_2$ |
|---|---|---|---|---|

或

| 操作码 | $L_1$ | $L_3$ | | $L_2$ |
|---|---|---|---|---|

指令解释过程中与通用寄存器内容有关的微操作时间关系如图 5-7 所示。用基/变址值在分析的前半段，取操作数在分析的后半段，存结果在执行的最后。$L_3$ 为结果，指令有操作数的相关和基址/变址值相关。“执行$_k$”和“分析$_{k+1}$”重叠时访问通用寄存器的时间关系如图 5-8 所示 。当出现 $L_1(k+1)=L_3(k)$时，就发生了 $L_1$ 相关。一种方法是推后读，另一种方法是相关专用通路(Forwarding Path)。用相关专用通路解决通用寄存器组的数据相关如图 5-9 所示。在运算器的输出到 B 或 C 输入之间增设“相关专用通路”，

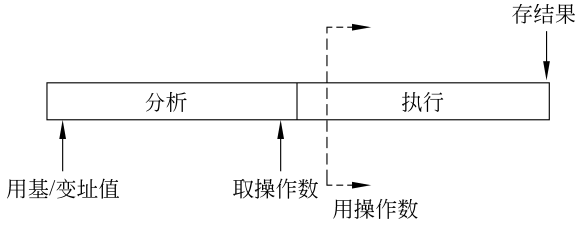



图 5-7　指令解释过程中与通用寄存器内容有关的微操作时间关系

发生 $L_1$ 或 $L_2$ 相关时,结果送寄存器的同时,直接将结果回送到 B 或 C。

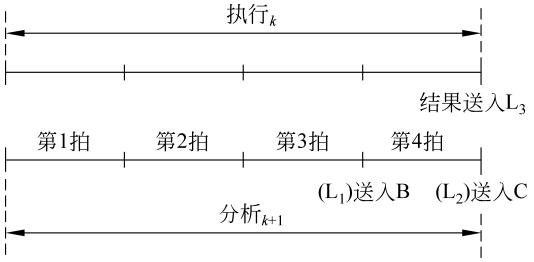



图 5-8 “执行$_k$”和“分析$_{k+1}$”重叠时访问通用寄存器的时间关系

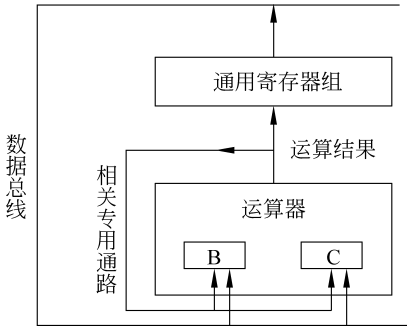



图 5-9 用相关专用通路解决数据相关

通用寄存器组的基(变)址值相关与通用寄存器组相关的处理方法相同。设操作数的有效地址为

$$(X_d)+(B_2)\cdot(B_2\neq 0000)+d_2$$

由分析器内的地址加法器形成。由于通常情况下,“分析”周期等于主存周期,所以,从时间关系上要求在“分析”周期的前半段,就能由通用寄存器输出总线取得($B_2$),送入地址加法器。由于运算结果是在“执行”周期的末尾才送入通用寄存器组的,它当然不能立即出现在通用寄存器输出总线上。也就是说,在“执行$_k$”得到的、送入通用寄存器的运算结果来不及作为“分析$_{k+2}$”的基址值用,更不用说作为“分析$_{k+1}$”的基址值用。因此,虽然是一次重叠,但基址值相关(B 相关)就不止会出现一次相关,还会出现二次相关。即当出现 $B(k+1)=L_3(k)$时,称为发生了 B 一次相关;而当出现 $B(k+2)=L_3(k)$时,称为发生了 B 二次相关,几次相关是指相隔的指令条数,如图 5-10 所示。

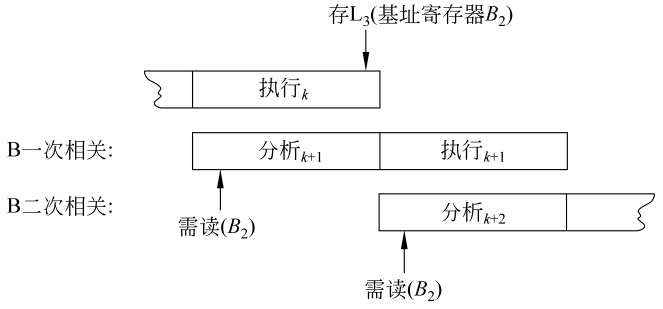



图 5-10 B 一次相关与 B 二次相关

总之,处理主存空间数据相关可采用推后“分析$_{k+1}$”和设置“相关专用通路”两种基本方法。前者以降低速度为代价,使设备基本上不增加;后者以增加设备为代价,使重叠效率不下降。

## 5.2 流水方式

### 5.2.1 基本概念

#### 1. 流水是重叠的引申

流水是重叠的引申。流水和重叠的差别只在于“一次重叠”是把一条指令的解释分为两个子过程，而流水是分为更多个($m$ 个)子过程。如能把一条指令的解释分解成时间相等的 $m$ 个子过程，则每隔 $\Delta t = T/m$ 就可以处理一条指令。指令流水线与工厂中的生产流水线相似，汽车装配生产线每个工段装配的不是同一辆汽车。

采用流水方式后，机器的最大吞吐率取决于子过程的经过时间 $\Delta t$，而实际吞吐率则还与连续进入流水线的任务数、各种相关等因素有关，因此，实际吞吐率总是低于最大吞吐率的。

指令分解为“分析”和“执行”子过程如图 5-11 所示。流水线中的锁存器如图 5-12 所示。

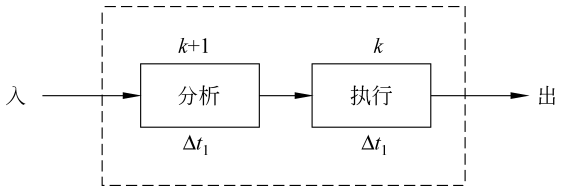



图 5-11 指令分解为“分析”和“执行”子过程

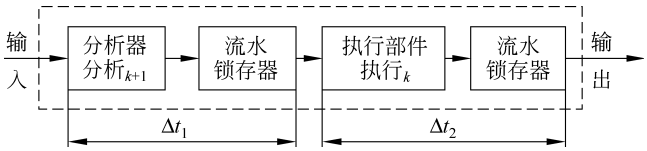


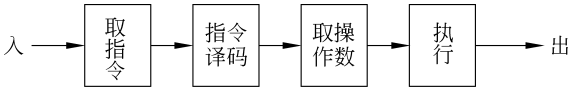

图 5-12 流水线中的锁存器

流水线的每一个子过程即阶段(Stage)，又称为流水步、流水步骤、流水段、流水线阶段、流水功能段、功能段、流水级或流水节拍等。在每一个流水段的末尾或开头必须设置一个寄存器，称为流水寄存器、流水锁存器、流水闸门寄存器等。流水锁存器会增加指令的执行时间。为了简化，在一般流水线中不画出流水锁存器。流水线的表示方法有三种：连接图、时空图和预约表。指令的流水处理如图 5-13 所示。时空图中空间是指流水线的功能段，每隔 $\Delta t$ 流水线输出一个结果。每条指令的执行时间没变，如同汽车装配流水线，装配每辆汽车的时间并没有减少。

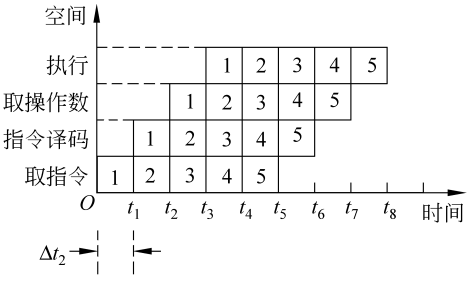
在处理机中采用流水线方式有以下主要特点。

(1) 只有连续提供同类任务才能充分发挥流水线的效率。

对于指令流水线要尽量减少因条件分支造成的“断流”，转移和不转移的分支都预取。




(a) 指令解释的流水处理




(b) 流水处理的时空图

图 5-13　指令的流水处理

对于操作部件主要通过编译技术,尽量提供连续的同类操作。

(2) 在流水线的每一个流水线段中都要设置一个流水锁存器。

从时间开销看,流水线的执行时间加长。流水锁存器是流水线中需要增加的主要硬件之一。

(3) 各流水段的时间应尽量相等,否则将引起流水线“阻塞”。

流水线处理机的基本时钟周期等于时间最长的流水段的时间长度,即“瓶颈”段的时间。

(4) 流水线需要有“装入时间”和“排空时间”。

只有完全充满时,整个流水线的效率才能得到充分发挥。

**2. 流水线的分类**

从不同的角度对流水线可进行不同的分类。按照流水线的处理级别的高低来分,将流水线分为部件级、处理机级和系统级三个不同的等级。

(1) 部件级流水线(操作流水线):指构成部件的各子部件间的流水线。例如,浮点加法器流水线,如图 5-14 所示。

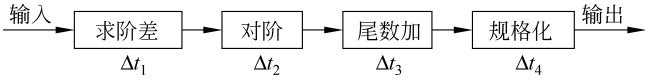



图 5-14　浮点加法器流水线

(2) 处理机级流水线:以指令为单位,又称指令流水线,是指构成处理机的各个功能部件的流水线。例如,在采用先行控制器的处理机中各功能部件之间的流水线,如图 5-15 所示。

(3) 系统级流水线:即处理机之间的流水线,是指构成计算机系统的多个处理机之间的流水线,也称为宏流水线。每个处理机对同一个数据流的不同部分分别进行处理,如图 5-16 所示。

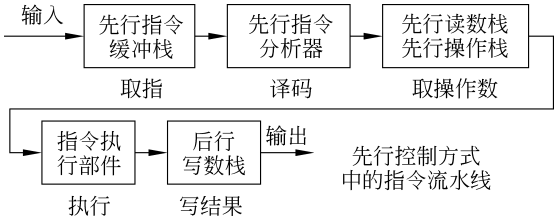


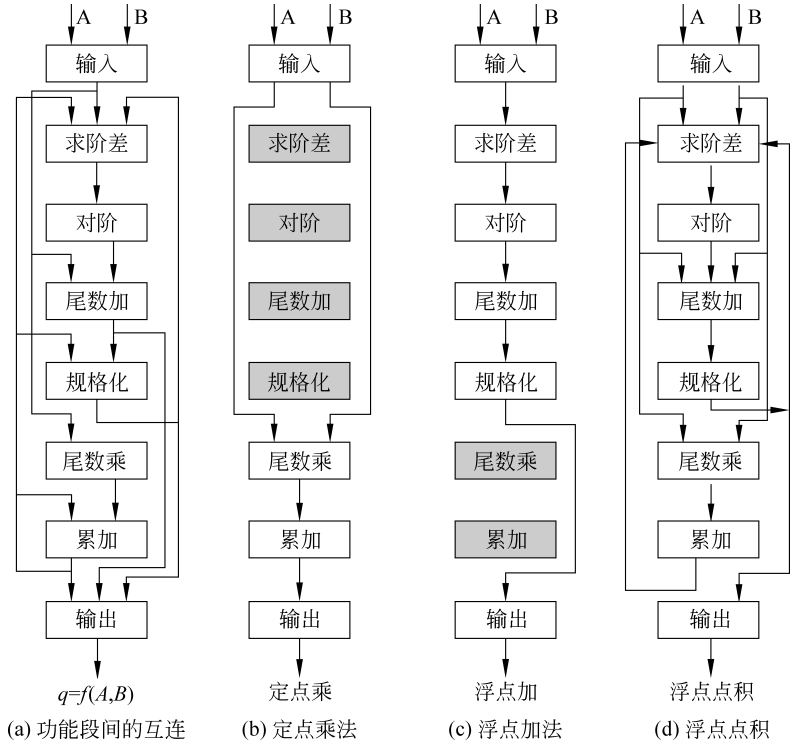

图 5-15 处理机级流水线


图 5-16 系统级流水线

按流水线具有功能的多少，将流水线分为单功能流水线和多功能流水线两类。

单功能流水线只能完成一种固定功能的流水线，只能有一种功能。例如，Cray-1 计算机中有 12 条；YH-1 计算机有 18 条；Pentium 有一条 5 段的定点和一条 8 段的浮点流水线；Pentium Ⅲ有 3 条指令流水线，其中 2 条定点指令流水线，1 条浮点指令流水线。

多功能流水线的各段通过不同连接实现多种不同的功能。Texas 公司的 ASC 计算机运算器的 8 段流水线，如图 5-17 所示，能够实现定点加减法、定点乘法、浮点加法、浮点点积、逻辑运算、移位操作、数据转换和向量运算等。多功能流水线能实现多种功能，阴影




(a) 功能段间的互连 (b) 定点乘法 (c) 浮点加法 (d) 浮点点积

图 5-17 ASC 计算机运算器的 8 段流水线