北京郵電大學
BEIJING UNIVERSITY OF POSTS AND TELECOMMUNICATIONS

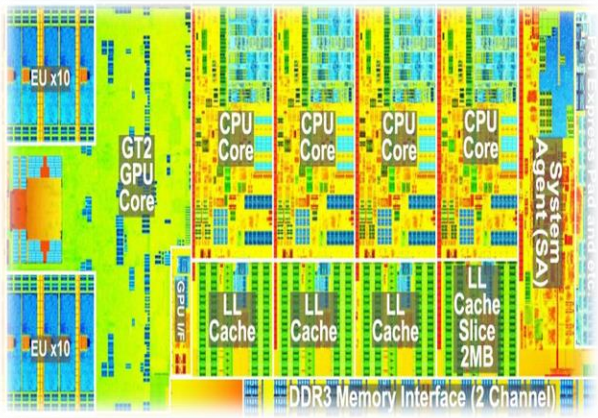
计算机系统结构

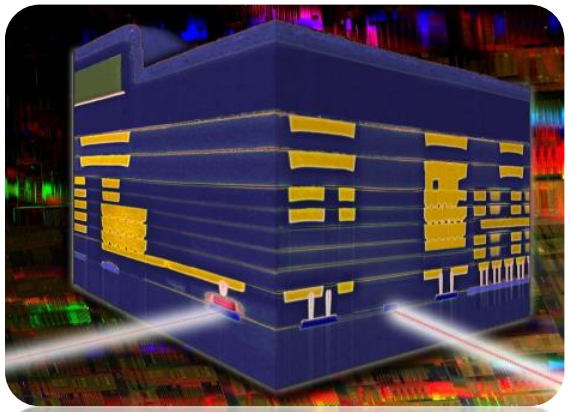
Computer Architecture

黄智瀕

574832909@qq.com

教三楼1017 / 计算机学院体系结构中心





总复习

- 1.计算机系统结构基础知识
- 2.流水线技术
- 3.指令集并行技术
- 4.向量处理机
- 5.互联网络
- 6.阵列处理机
- 7.机群系统
- 8.多处理机

主要的概念

- 什么是计算机系统结构？
- Flynn分类法？
- Amdahl定律？

$$S_n = \frac{T_0}{T_n} = \frac{1}{(1-F_e)+F_e/S_e}$$

$$S_n = \frac{T_0}{T_n} = \frac{1}{(1-\Sigma F_i)+\Sigma\frac{F_i}{S_i}}$$

- 平均指令执行周期数CPI（Cycle Per Instruction）

$$CPI = \frac{\text{执行程序所需的时钟周期数}}{\text{指令数}}$$

- SimpleScalar模拟器的特征
- 基准测试程序的概念
- 并行性的概念

5. 并行

- 并行性的概念
  - 并行性的实现途径

1. 时间重叠

引入时间因素，让多个处理过程在时间上相互错开，轮流重叠地使用同一套硬件设备的各个部分，以加快硬件周转而赢得速度。

最典型的例子：流水线技术

5. 并行

- 并行性的概念
  - 并行性的实现途径

2. 资源重复

引入空间因素，以数量取胜。通过重复设置硬件资源，大幅度地提高计算机系统的性能。

3. 资源共享

这是一种软件方法，它使多个任务按一定时间顺序轮流使用同一套硬件设备。

例如：多道程序、分时系统

例题： 某台主频为400MHz的计算机执行标准测试程序，程序中指令类型、执行数量和平均时钟周期数如下：

指令类型	指令执行数量	平均时钟周期数
整数	45000	1
数据传送	75000	2
浮点	8000	4
分支	1500	2

求该计算机的有效CPI、MIPS和程序执行时间。

解： CPI ＝(45000×1＋75000×2＋8000×4＋1500×2) / 129500＝1.776

MIPS速率＝f/ CPI ＝400/1.776 ＝225.225MIPS

程序执行时间= (45000×1＋75000×2＋8000×4＋1500×2) ／ 400=575μs

例题： 将计算机系统中某一功能的处理速度加快10倍，但该功能的处理时间仅为整个系统运行时间的40%，则采用此增强功能方法后，能使整个系统的性能提高多少？

解： 可改进比例 = 40% = 0.4　　部件加速比 = 10

根据Amdahl定律可知：

$$\text{系统加速比} = \frac{1}{(1-0.4)+\frac{0.4}{10}} = 1.5625$$

采用此增强功能方法后，能使整个系统的性能提高到原来的1.5625倍。

例题： 计算机系统中有三个部件可以改进，这三个部件的部件加速比为：

部件加速比$_1$=30； 部件加速比$_2$=20； 部件加速比$_3$=10

(1) 如果部件1和部件2的可改进比例均为30%，那么当部件3的可改进比例为多少时，系统加速比才可以达到10？

(2) 如果三个部件的可改进比例分别为30%、30%和20%，三个部件同时改进，那么系统中不可加速部分的执行时间在总执行时间中占的比例是多少？

解： （1）在多个部件可改进情况下，Amdahl定理的扩展：

$$S_n = \frac{1}{(1-\sum F_i)+\sum \frac{F_i}{S_i}}$$

已知S_1=30，S_2=20，S_3=10，S_n=10，F_1=0.3，F_2=0.3，得：

$$10 = \frac{1}{1-(0.3+0.3+F_3)+(0.3/30+0.3/20+F_3/10)}$$

得F_3=0.36，即部件3的可改进比例为36%。

例题： 计算机系统中有三个部件可以改进，这三个部件的部件加速比为：

部件加速比$_1$=30； 部件加速比$_2$=20； 部件加速比$_3$=10

(1) 如果部件1和部件2的可改进比例均为30%，那么当部件3的可改进比例为多少时，系统加速比才可以达到10？

(2) 如果三个部件的可改进比例分别为30%、30%和20%，三个部件同时改进，那么系统中不可加速部分的执行时间在总执行时间中占的比例是多少？

解： （2）设系统改进前的执行时间为T，则3个部件改进前的执行时间为：（0.3+0.3+0.2）T = 0.8T，不可改进部分的执行时间为0.2T。

已知3个部件改进后的加速比分别为S_1＝30，S_2＝20，S_3＝10，因此3个部件改进后的执行时间为：

$$T_n' = \frac{0.3T}{30} + \frac{0.3T}{20} + \frac{0.2T}{10} = 0.045T$$

改进后整个系统的执行时间为：Tn = 0.045T+0.2T = 0.245T

那么系统中不可改进部分的执行时间在总执行时间中占的比例是：$\frac{0.2T}{0.245T} = 0.82$

例题： 假设某应用程序中有4类操作，通过改进，各操作获得不同的性能提高。具体数据如下表所示：

操作类型	程序中的数量（百万条指令）	改进前的执行时间（周期）	改进后的执行时间（周期）
操作 1	10	2	1
操作 2	30	20	15
操作 3	35	10	3
操作 4	15	4	1

（1）改进后，各类操作的加速比分别是多少？
（2）各类操作单独改进后，程序获得的加速比分别是多少？
（3）4类操作均改进后，整个程序的加速比是多少？

解： 根据Amdahl定律

$$S_n = \frac{1}{(1-Fe)+\frac{Fe}{Se}}$$

操作类型	各类操作的指令条数在程序中所占的比例 F_i	各类操作的加速比 S_i	各类操作单独改进后，程序获得的加速比
操作 1	11.1%	2	1.06
操作 2	33.3%	1.33	1.09
操作 3	38.9%	3.33	1.37
操作 4	16.7%	4	1.14

4类操作均改进后，整个程序的加速比：$S_n = \frac{1}{(1-\sum F_i)+\sum \frac{F_i}{S_i}} \approx 2.16$

主要的概念

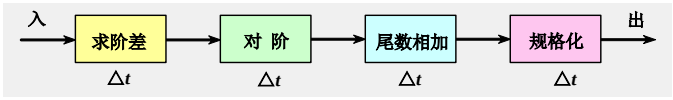
- 什么是指令流水线？
- 流水线的时空图
- 流水线需要有通过时间和排空时间
- 流水线的性能指标

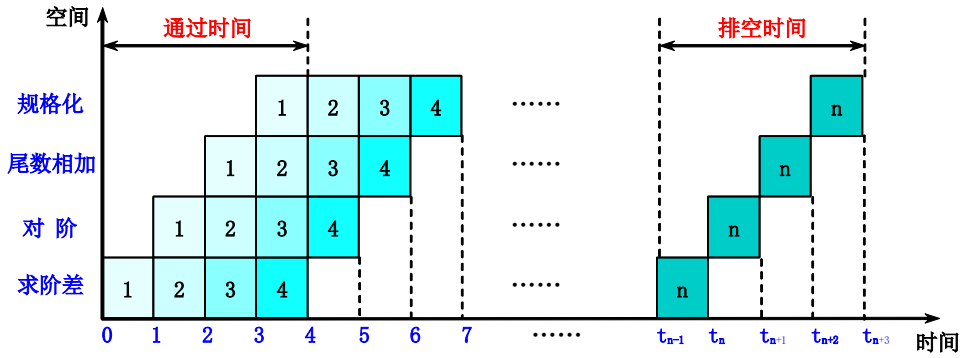
吞吐率

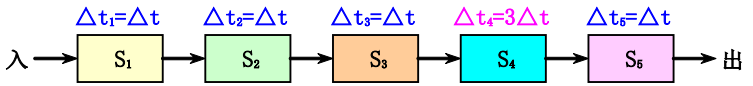
$$TP = \frac{n}{T_k}$$

n：任务数

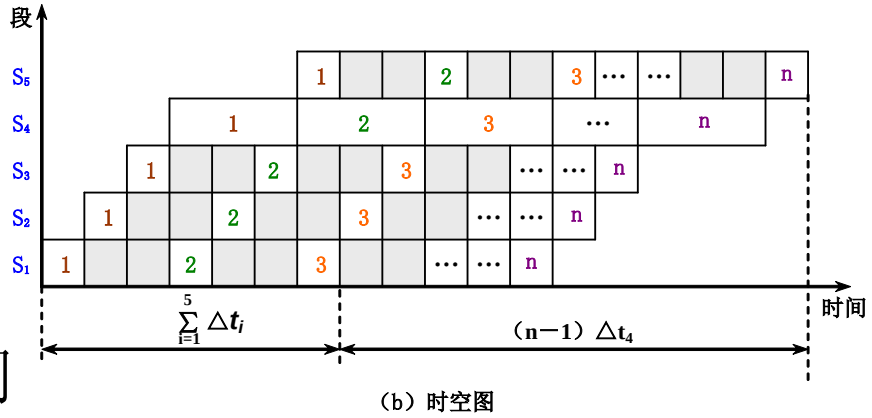
T_k：处理完成n个任务所用的时间







（a）流水线



（b）时空图

主要的概念

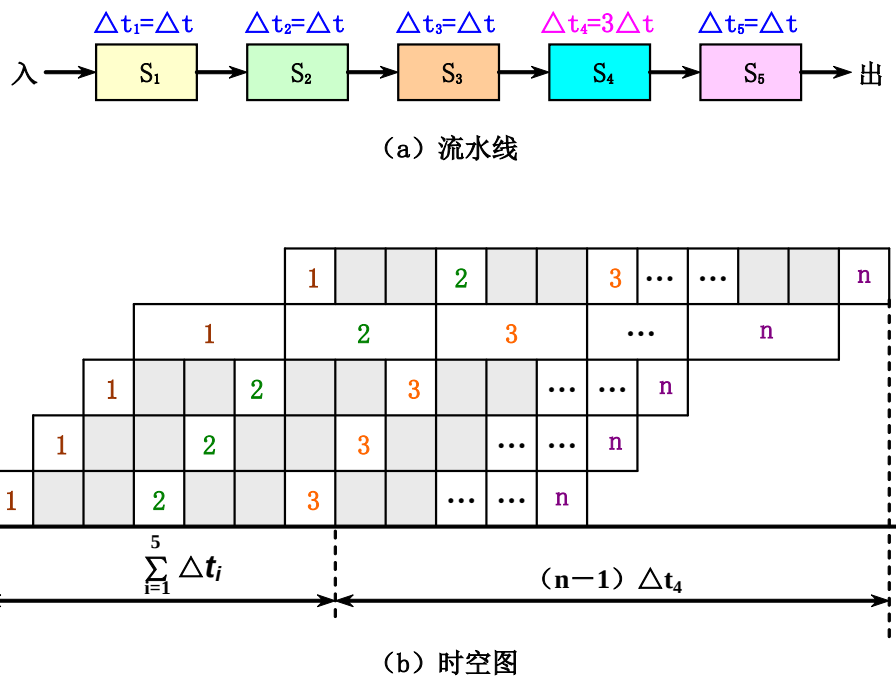
- 流水线的性能指标

吞吐率

$$TP = \frac{n}{T_k}$$

加速比

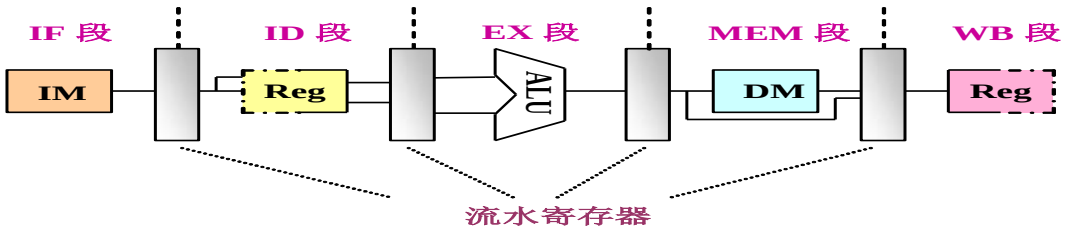
$$S = \frac{T_s}{T_k}$$



流水线的效率

$$E = \frac{n\text{个任务实际占用的时空区}}{k\text{个段总的时空区}}$$

- 经典的5段流水线



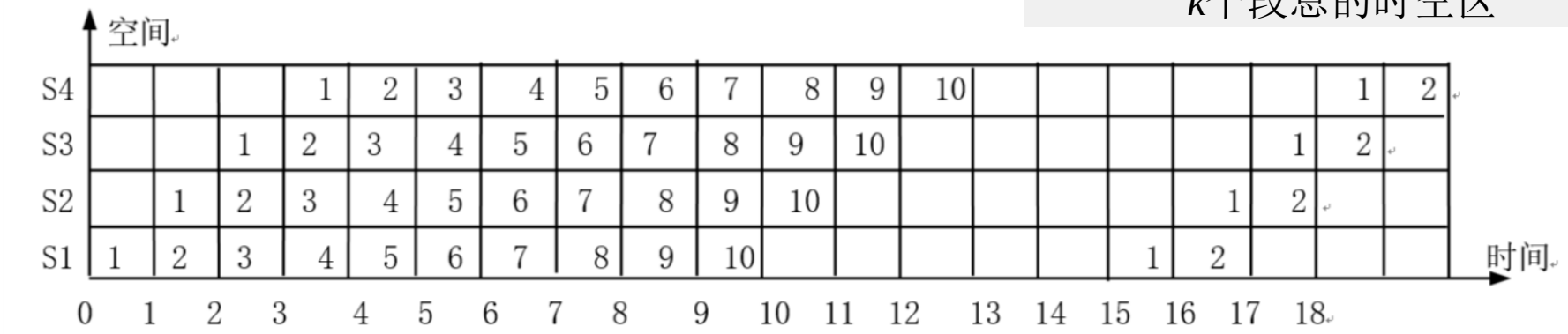
主要的概念

- 相关与流水线冲突
  - 相关有3种类型
    - 数据相关（也称真数据相关）
    - 名相关
    - 控制相关

例题： 某流水线由4个功能部件组成，每个功能部件的执行时间都为△t。当输入10个数据后，停顿5t，又输入10个数据，如此重复。计算流水线的实际吞吐率、加速比和效率，并画出时空图。

解： 该题中的流水线的任务是非连续流入的，因此不能直接应用公式直接计算，要利用时空图。题意的流水线时空图如下图所示。

$$E=\frac{n\text{个任务实际占用的时空区}}{k\text{个段总的时空区}}$$



TP = 10/15Δt = 0.67/Δt

S = T_0/T_K = 10×4Δt/15Δt = 2.67

E = 4×10Δt/4×15Δt = 0.67

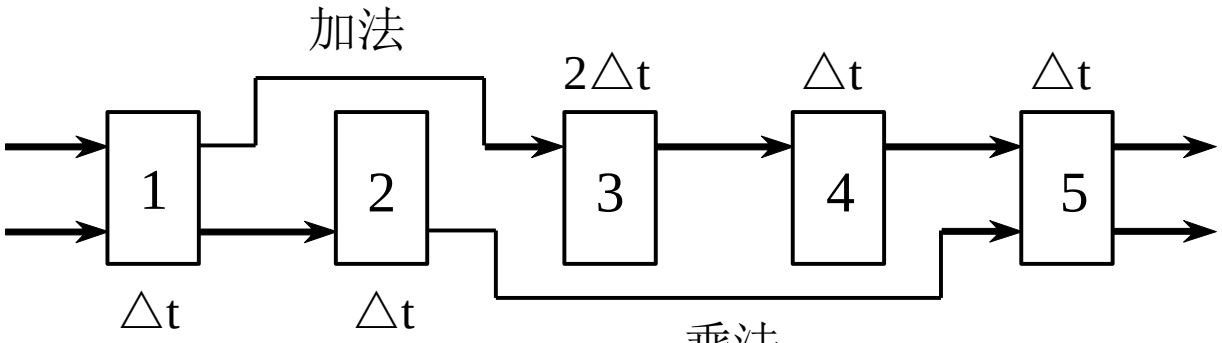
$$TP=\frac{n}{\sum_{i=1}^{k}\Delta t_i+(n-1)\max(\Delta t_1,\Delta t_2,\cdots,\Delta t_k)}$$

$$S=\frac{T_s}{T_k}$$

例题： 有一条静态多功能流水线由5段组成，加法用1、3、4、5段，乘法用1、2、5段，第3段的时间为2△t，其余各段的时间均为△t，而且流水线的输出可以直接返回输入端或暂存于相应的流水寄存器中。现要在该流水线上计算

$$\prod_{i=1}^{4}(A_i + B_i)$$

画出其时空图，并计算其吞吐率、加速比和效率。



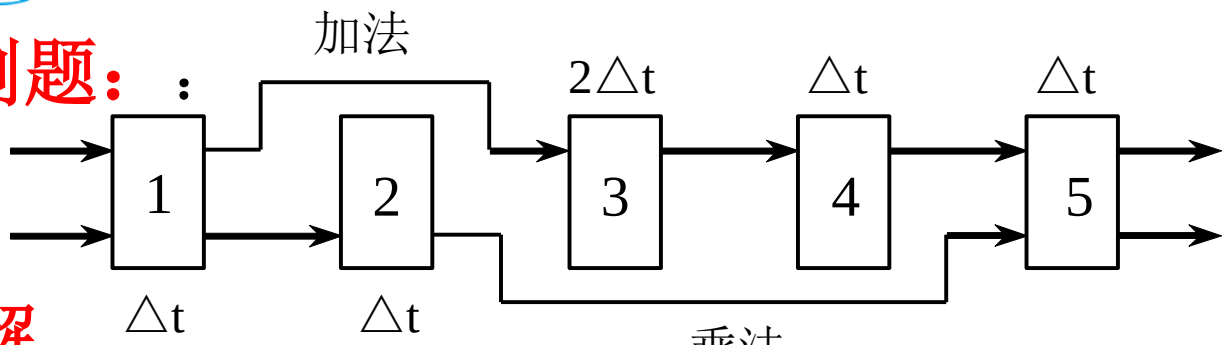
解：

$$TP = \frac{n}{\sum_{i=1}^{k}\Delta t_i + (n-1)\max(\Delta t_1, \Delta t_2, \cdots, \Delta t_k)}$$

$$S = \frac{T_s}{T_k}$$

$$E = \frac{n\text{个任务实际占用的时空区}}{k\text{个段总的时空区}}$$

例题：

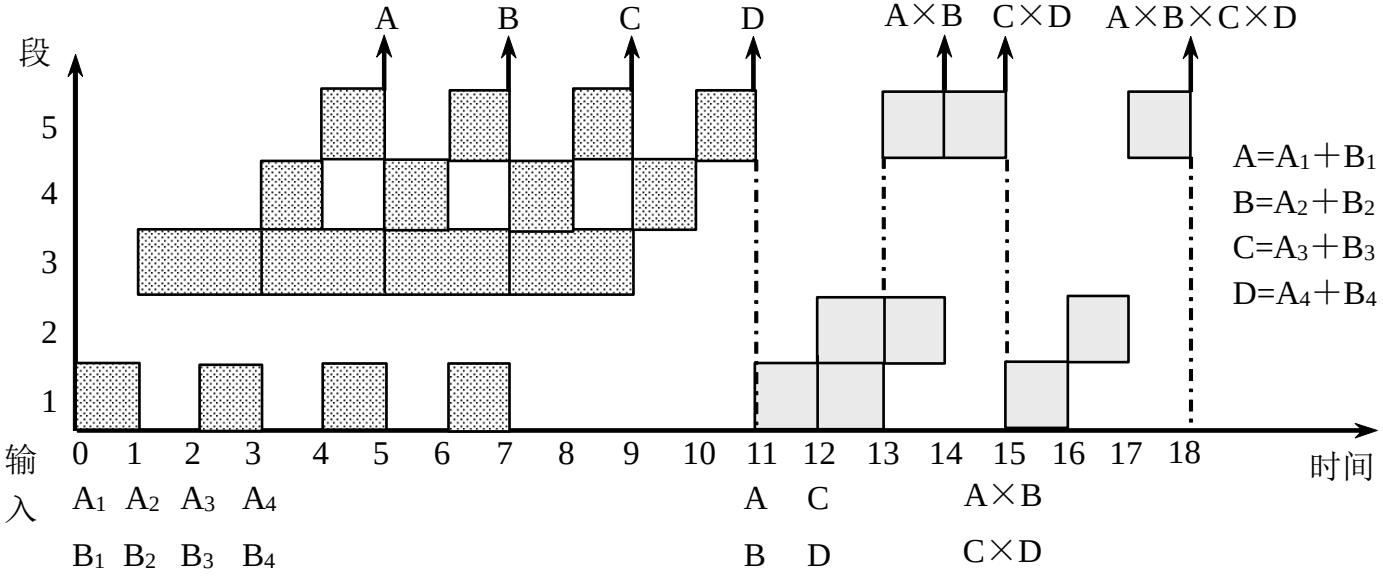


$$\prod_{i=1}^{4}(A_i + B_i)$$

解：

应选择适合于流水线工作的算法。对于本题，应先计算A_1+B_1、A_2+B_2、A_3+B_3和A_4+B_4；再计算$(A_1+B_1)\times(A_2+B_2)$和$(A_3+B_3)\times(A_4+B_4)$；然后求总的结果。

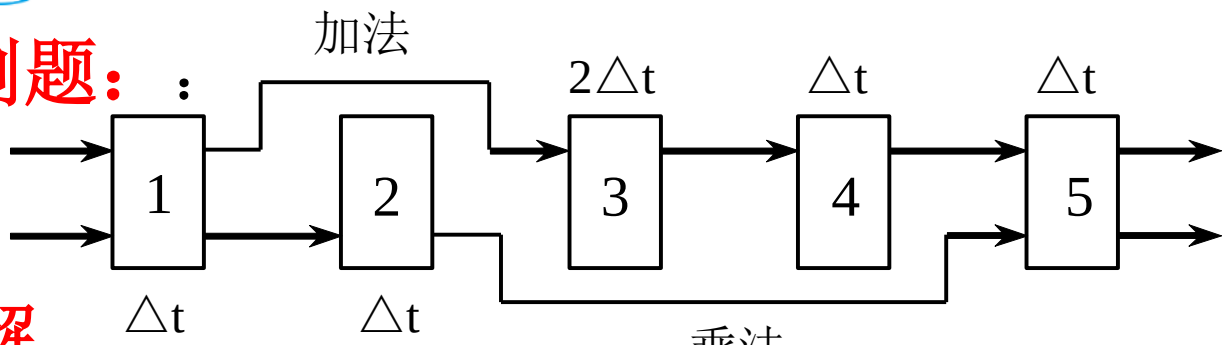
其次，画出完成该计算的时空图，如图所示，图中阴影部分表示该段在工作。



由图可见，它在18个△t时间中，给出了7个结果。所以吞吐率为：

$$TP = \frac{7}{18\Delta t}$$

例题：



$$\prod_{i=1}^{4}(A_i + B_i)$$

解：

应选择适合于流水线工作的算法。对于本题，应先计算A_1+B_1、A_2+B_2、A_3+B_3和A_4+B_4；再计算$(A_1+B_1)\times(A_2+B_2)$和$(A_3+B_3)\times(A_4+B_4)$；然后求总的结果。

如果不用流水线，由于一次求积需3△t，一次求和需5△t，则产生上述7个结果共需（4×5+3×3）△t =29△t。所以加速比为：

$$S = \frac{29\Delta t}{18\Delta t} = 1.61$$

该流水线的效率可由阴影区的面积和5个段总时空区的面积的比值求得：

$$E = \frac{4 \times 5 + 3 \times 3}{5 \times 18} = 0.322$$

例题： 有一个流水线由4段组成，其中每当流经第3段时，总要在该段循环一次，然后才能流到第4段。如果每段经过一次所需要的时间都是Δt，问：

（1） 当在流水线的输入端连续地每 Δt 时间输入任务时，该流水线会发生什么情况？

（2） 此流水线的最大吞吐率为多少？如果每$2\Delta t$ 输入一个任务，连续处理10个任务时的实际吞吐率和效率是多少？

（3） 当每段时间不变时，如何提高该流水线的吞吐率？仍连续处理10个任务时，其吞吐率提高多少？

解：（1）会发生流水线阻塞情况。

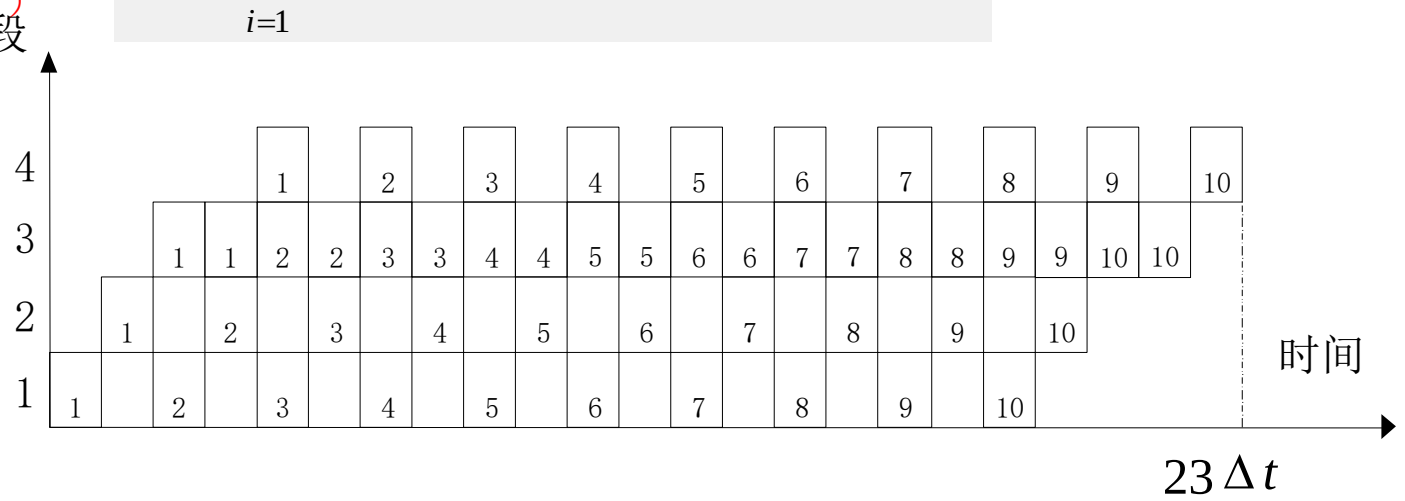
第 1 个任务	S1	S2	S3	S3	S4						
第 2 个任务		S1	S2	stall	S3	S3	S4				
第 3 个任务			S1	stall	S2	stall	S3	S3	S4		
第 4 个任务					S1	stall	S2	stall	S3	S3	S4

例题：：

$$TP=\frac{n}{\sum_{i=1}^{k}\Delta t_i+(n-1)\max(\Delta t_1,\Delta t_2,\cdots,\Delta t_k)}$$

$$TP_{\max}=\frac{1}{\max(\Delta t_1,\Delta t_2,\cdots\Delta t_k)}$$

解：（2）



$$\mathrm{TP}_{\max}=\frac{1}{2\Delta t}$$

$$\mathrm{T}_{\text{pipeline}}=23\Delta t$$

$$\mathrm{Tp}=\mathrm{n}/\mathrm{T}_{\text{pipeline}}=10/23\Delta t$$

$$\Delta\mathrm{E}=\mathrm{TP}\cdot 5\Delta t/4=50/92\approx 54.35\%$$

$$E=\frac{n\text{个任务实际占用的时空区}}{k\text{个段总的时空区}}$$

·2.流水线技术

例题：：

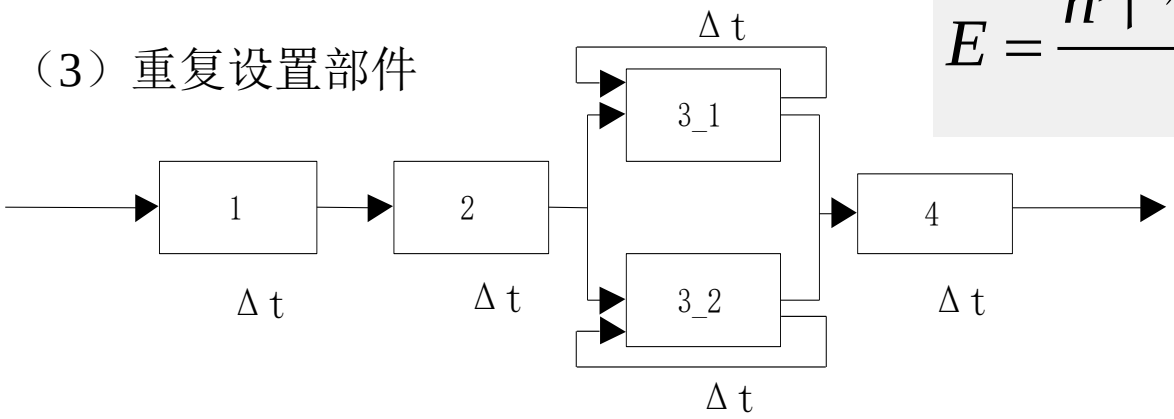
解：

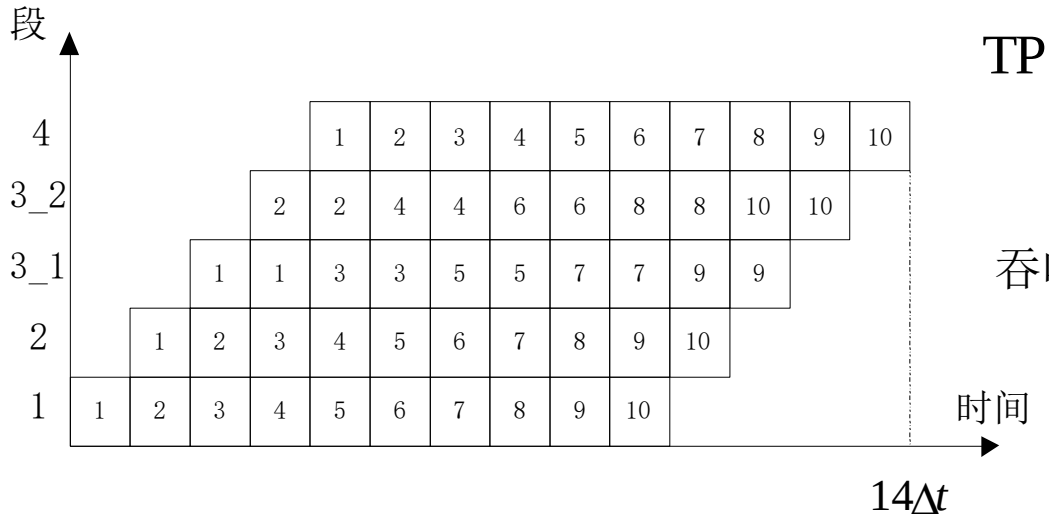
$$TP=\frac{n}{\sum_{i=1}^{k}\Delta t_i+(n-1)\max(\Delta t_1,\Delta t_2,\cdots,\Delta t_k)}$$

$$TP_{max}=\frac{1}{\max(\Delta t_1,\Delta t_2,\cdots\Delta t_k)}$$

$$E=\frac{n\text{个任务实际占用的时空区}}{k\text{个段总的时空区}}$$

（3）重复设置部件





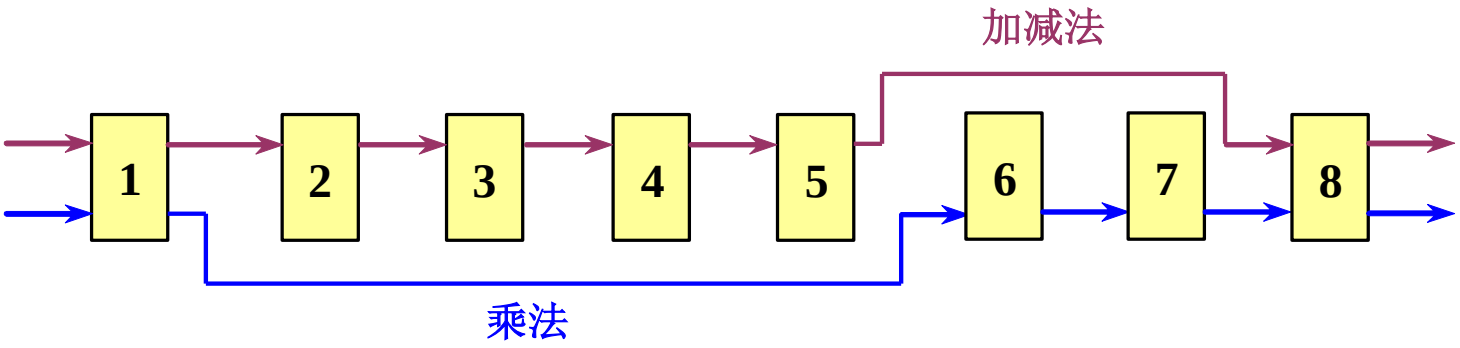
$$TP=n/T_{pipeline}=10/14\cdot\Delta t=5/7\cdot\Delta t$$

吞吐率提高倍数 $\dfrac{5/7\Delta t}{10/23\Delta t}$

例1 设在下图所示的静态流水线上计算：

$$\prod_{i=1}^{4}(A_i + B_i)$$

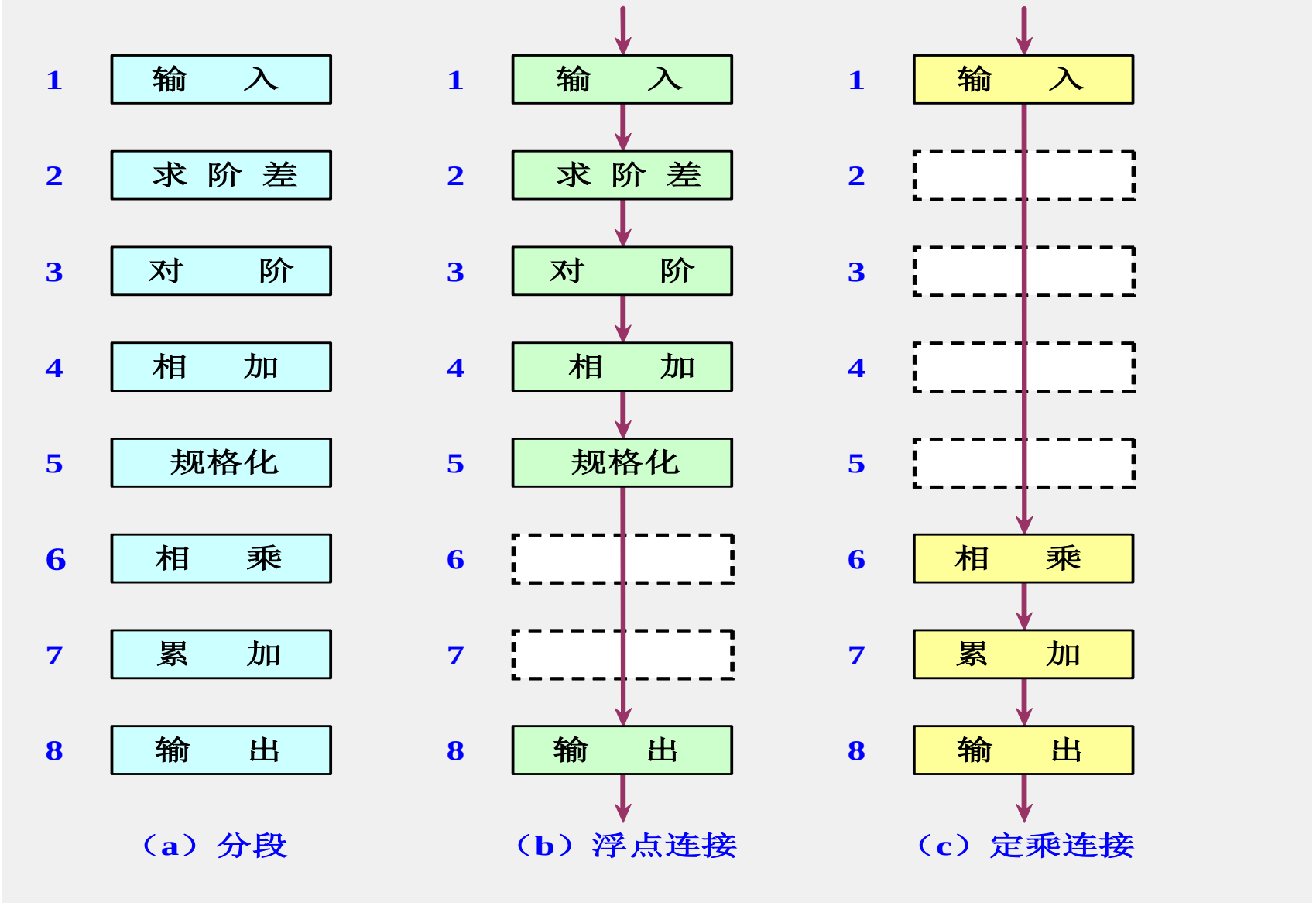
流水线的输出可以直接返回输入端或暂存于相应的流水寄存器中，试计算其吞吐率、加速比和效率。



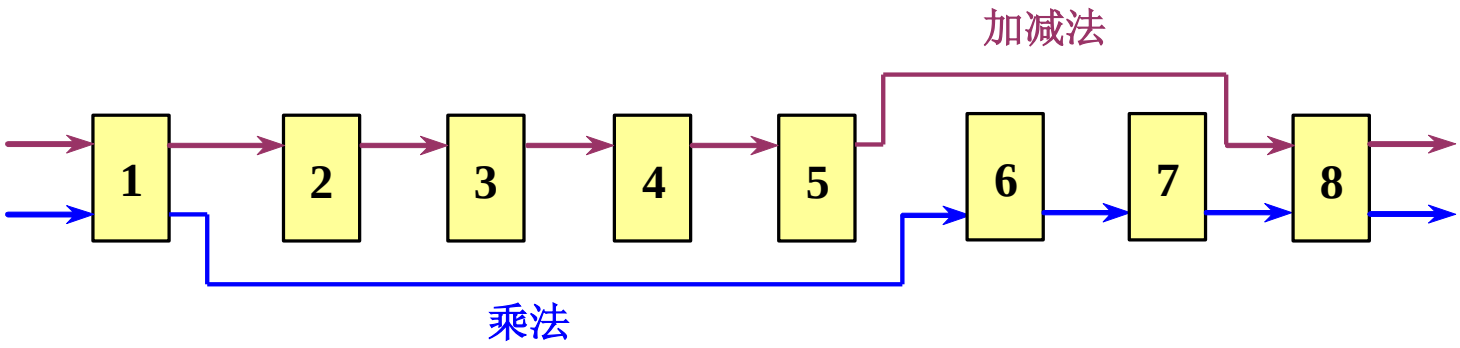
（每段的时间都为△t）

2.3流水线的性能指标
>>流水线的性能分析举例

例1



例1



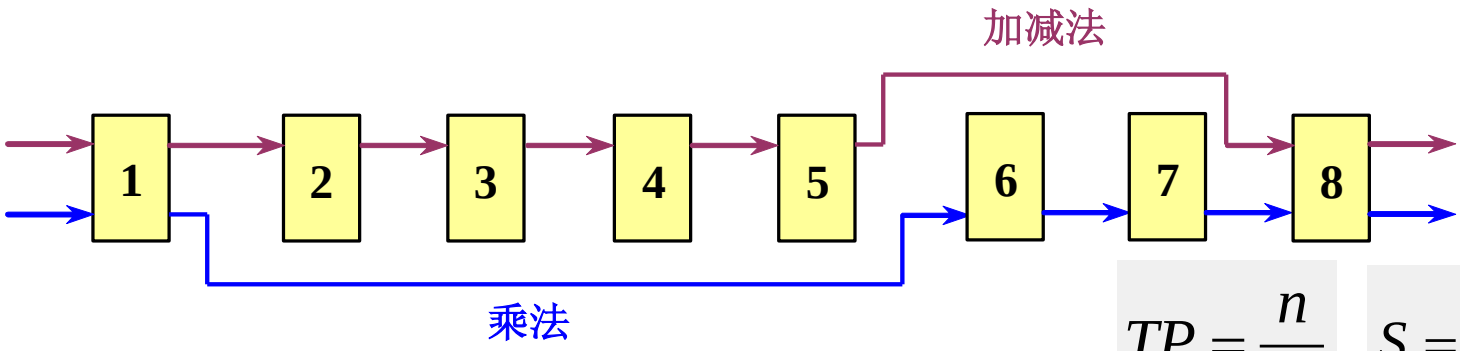
解：（1）选择适合于流水线工作的算法

- 先计算A_1+B_1、A_2+B_2、A_3+B_3和A_4+B_4；
- 再计算$(A_1+B_1)\times(A_2+B_2)$和$(A_3+B_3)\times(A_4+B_4)$；
- 然后求总的乘积结果。

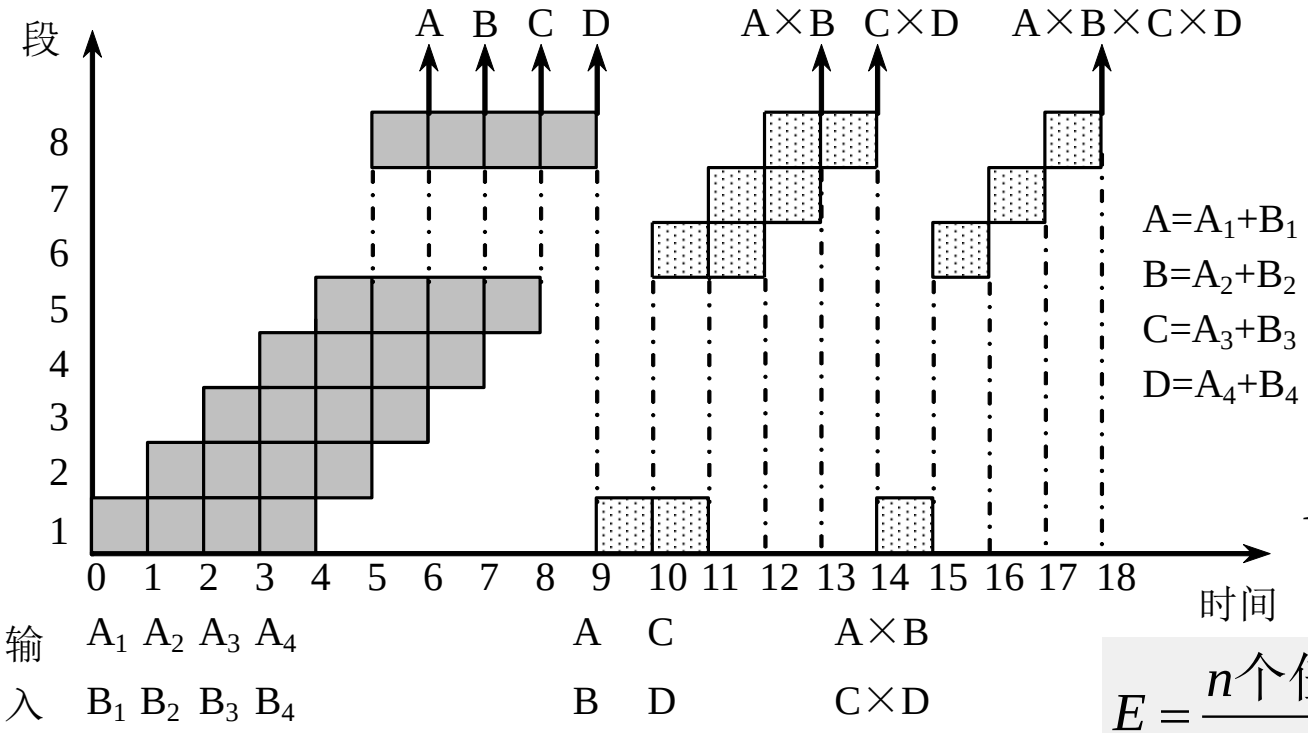
（2）画出时空图

2.3流水线的性能指标 >>流水线的性能分析举例

例1



解：



$$TP=\frac{n}{T_k} \qquad S=\frac{T_s}{T_k}$$

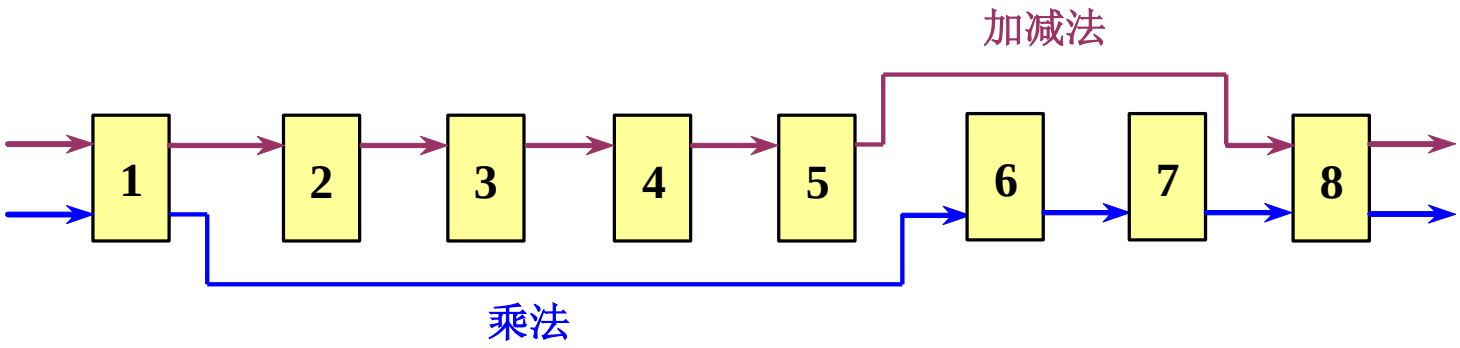
实际吞吐率 $TP=\dfrac{7}{18\Delta t}$

$$S=\frac{36\Delta t}{18\Delta t}=2$$

$$E=\frac{4\times 6+3\times 4}{8\times 18}=0.25$$

$$E=\frac{n\text{个任务实际占用的时空区}}{k\text{个段总的时空区}}$$

例1



解：（3）计算性能

- 在18个△t时间中，给出了7个结果。吞吐率为：

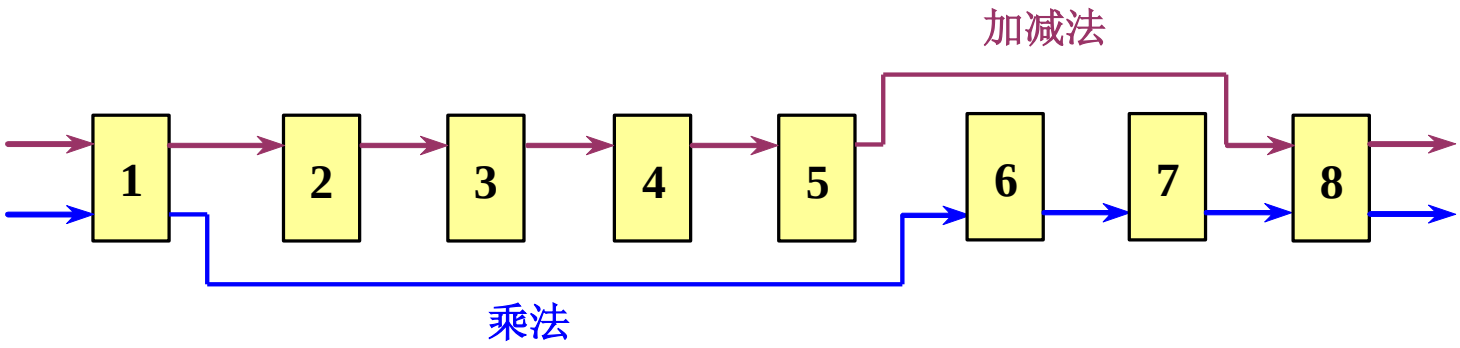
$$TP = \frac{7}{18\Delta t}$$

- 不用流水线，由于一次求和需6△t，一次求积需4△t，则产生上述7个结果共需（4×6+3×4）△t = 36△t 加速比为：

$$S = \frac{36\Delta t}{18\Delta t} = 2$$

2.3流水线的性能指标 >>流水线的性能分析举例

例1



解：（3）计算性能 •流水线的效率：$E = \frac{4\times6+3\times4}{8\times18} = 0.25$

可以看出，在求解此问题时，该流水线的效率不高。

- 主要原因
  - 多功能流水线在做某一种运算时，总有一些段是空闲的；
  - 静态流水线在进行功能切换时，要等前一种运算全部流出流水线后才能进行后面的运算；
  - 运算之间存在关联，后面有些运算要用到前面运算的结果；
  - 流水线的工作过程有建立与排空部分。

- **例题：** 在一台单流水线多操作部件的处理机上执行下面的程序，每条指令的取指令、指令译码需要一个时钟周期，MOVE、ADD和MUL操作分别需要2个、3个和4个时钟周期，每个操作都在第一个时钟周期从通用寄存器中读操作数，在最后一个时钟周期把运算结果写到通用寄存器中。

  k： MOVE R1，R0 ；R1←(R0)

  k+1： MUL R0，R2，R1 ；R0←(R2)×(R1)

  k+2： ADD R0，R2，R3 ；R0←(R2)+(R3)

  就程序本身而言，可能有哪几种数据相关?

解：（1）就程序本身而言，可能有三种数据相关。若3条指令顺序流动，则：
k指令对R1寄存器的写与k+1指令对**R1寄存器**的读形成的“先写后读”数据相关。
k指令对R0寄存器读与k+1执行对R0寄存器的写形成“先读后写”反相关
k+2指令对R0寄存器的写与k+1指令对R0寄存器的写形成的“写后写”输出相关。

主要的概念

- 动态调度的基本思想
- 记分牌动态调度算法的思想及表格填写
- Tomasulo算法思想及表格填写
- 动态的分支预测算法的思想
- 超标量流水处理机
- 超流水线处理机
- 超标量超流水处理机
- 超长指令字处理机

- 数据相关及其处理技术
  - 记分牌动态调度算法
    - 每条指令的执行过程分为4段 （主要考虑浮点操作 ）

➢读操作数

记分牌监测源操作数的可用性，如果数据可用，它就通知功能部件从寄存器中读出源操作数并开始执行。

动态地解决了RAW冲突（必须等待写完成之后才能读操作数），并导致指令可能乱序开始执行。

➢执行

取到操作数后，功能部件开始执行。当产生出结果后，就通知记分牌它已经完成执行。

在浮点流水线中，这一段可能要占用多个时钟周期。

- 数据相关及其处理技术
  - 记分牌动态调度算法
    - ➢ 写结果

记分牌一旦知道执行部件完成了执行，就检测是否存在WAR冲突。如果不存在，或者原有的WAR冲突已消失，记分牌就通知功能部件把结果写入目的寄存器，并释放该指令使用的所有资源。

- 如果检测到WAR冲突，就不允许该指令将结果写到目的寄存器。这发生在以下情况：
  - 前面的某条指令（按顺序流出）还没有读取操作数；而且：其中某个源操作数寄存器与本指令的目的寄存器相同。
  - 在这种情况下，记分牌必须等待，直到该冲突消失。

- 数据相关及其处理技术
  - 记分牌动态调度算法
    - 记分牌通过与功能部件的通信控制指令的逐步执行。
    - 记分牌中记录的信息由3部分构成
      - 指令状态表：记录正在执行的各条指令已经进入到了哪一段。
      - 功能部件状态表：记录各个功能部件的状态。每个功能部件有一项，每一项由以下9个字段组成：
        - Busy：忙标志，指出功能部件是否忙。初值为“no”；
        - Op：该功能部件正在执行或将要执行的操作；
        - Fi：目的寄存器编号；
        - Fj，Fk：源寄存器编号；
        - Qj，Qk：指出向源寄存器Fj、Fk写数据的功能部件；
        - Rj，Rk：标志位，为“yes”表示Fj，Fk中的操作数就绪且还未被取走。否则就被置为“no”。

- 数据相关及其处理技术
  - 记分牌动态调度算法
    - 记分牌中记录的信息由3部分构成
      - 结果寄存器状态表Result：每个寄存器在该表中有一项，用于指出哪个功能部件（编号）将把结果写入该寄存器。
        - 如果当前正在运行的指令都不以它为目的寄存器，则其相应项置为“no”。
        - Result各项的初值为“no”（全0）。

- 数据相关及其处理技术
  - 记分牌动态调度算法

举例

MIPS记分牌所要维护的数据结构
下列代码运行过程中记分牌保存的信息

```
L.D      F6, 34(R2)
L.D      F2, 45(R3)
MULT.D   F0, F2, F4
SUB.D    F8, F6, F2
DIV.D    F10, F0, F6
ADD.D    F6, F8, F2
```

MIPS记分牌中的信息（完全执行完第一条指令之后的记分牌各表格记录数据）

1.指令当前执行到第二条指令，已经执行完成，但尚未更新结果寄存器。
2.由于F2寄存器的 RAW相关，后续两条指令在流出阶段等待。不能进入读操作数阶段。
3.DIV.D也只能在流出阶段，因为F0寄存器的 RAW相关。
4.ADD.D由于与指令SUB.D存在加法器的结构冲突，因此，不能流出ADD.D指令

指　令	指令状态表			
	流出	读操作数	执行	写结果
L.D F6,34(R2)	√	√	√	√
L.D F2, 45(R3)	√	√	√	
MULT.D F0, F2, F4	√			
SUB.D F8, F6, F2	√			
DIV.D F10, F0, F6	√			
ADD.D F6, F8, F2				

部件名称	功能部件状态表								
	Busy	Op	Fi	Fj	Fk	Qj	Qk	Rj	Rk
Integer	yes	L.D	F2	R3				no	
Mult1	yes	MULT.D	F0	F2	F4	Integer		no	yes
Mult2	no								
Add	yes	SUB.D	F8	F6	F2		Integer	yes	no
Divide	yes	DIV.D	F10	F0	F6	Mult1		no	yes

	结果寄存器状态表							
	F0	F2	F4	F6	F8	F10	…	F30
部件名称	Mult1	Integer			Add	Divide		

3.2 数据相关及其处理技术 >>记分牌动态调度算法

例：
假设浮点流水线各部件的延迟如下：
加法：2个时钟周期
乘法：10个时钟周期
除法：40个时钟周期；

分析MULT.D准备写结果之前的记分牌状态。

1.分析各指令之间相关性：
WAW相关：第一条LD与ADD.D；MULT.D和DIV.D
RAW相关：第二条LD与MULT.D与SUB.D; MULT.D与DIV.D；SUB.D与ADD.D
WAR相关：DIV.D和ADD.D；SUB.D和ADD.D
结构相关：ADD.D和SUB.D

2.MULT.D写结构之前：

指令状态表

指令		流出	读操作数	执行	写结果
L.D	F6, 34(R2)	√	√	√	√
L.D	F2, 45(R3)	√	√	√	√
MULT.D	F0, F2, F4	√	√	√	
SUB.D	F8, F6, F2	√	√	√	√
DIV.D	F10, F0, F6	√			
ADD.D	F6, F8, F2	√	√	√	

功能部件状态表

部件名称	Busy	Op	Fi	Fj	Fk	Qj	Qk	Rj	Rk
Integer	no								
Mult1	yes	MULT.D	F0	F2	F4			no	no
Mult2	no								
Add	yes	ADD.D	F6	F8	F2			no	no
Divide	yes	DIV.D	F10	F0	F6	Mult1		no	yes

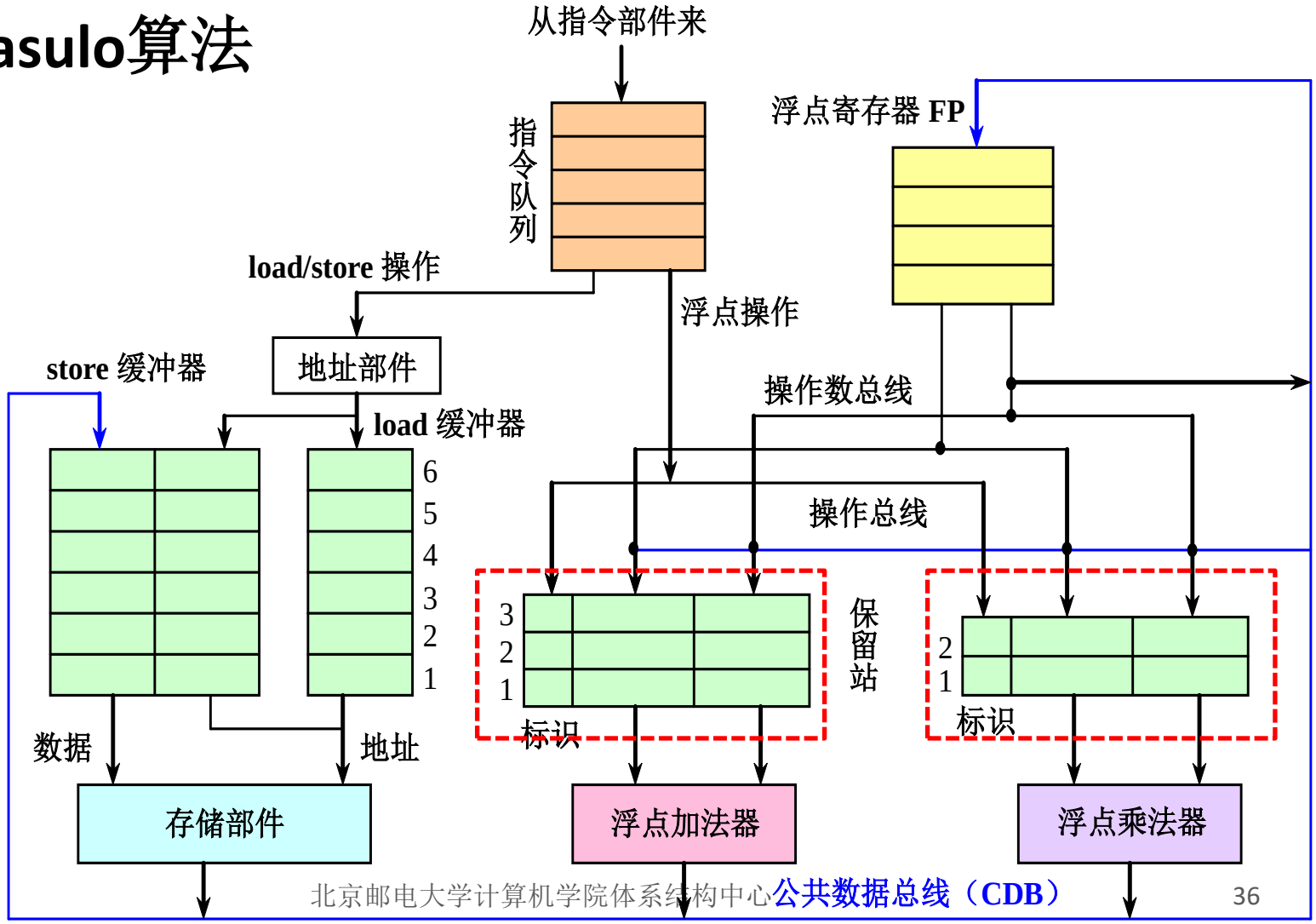
结果寄存器状态表

	F0	F2	F4	F6	F8	F10	…	F30
部件名称	Mult1			Add		Divide		

- 数据相关及其处理技术
  - Tomasulo算法
    - 基本思想
      - 核心思想
        - 记录和检测指令相关，操作数一旦就绪就立即执行，把发生RAW冲突的可能性减少到最小；
        - 通过寄存器换名来消除WAR冲突和WAW冲突。
      - IBM 360/91首先采用了Tomasulo算法。
        - IBM 360/91的设计目标是基于整个360系列的统一指令系统和编译器来实现高性能，而不是设计和利用专用的编译器来提高性能。

          需要更多地依赖于硬件。

- 数据相关及其处理技术
  - Tomasulo算法



- 数据相关及其处理技术
  - Tomasulo算法

例 对于下述指令序列，给出当第一条指令完成并写入结果时，Tomasulo算法所用的各信息表中的内容。

```
L.D    F6,34（R2）
L.D    F2,45（R3）
MUL.D F0,F2,F4
SUB.D F8,F2,F6
DIV.D F10,F0,F6
ADD.D F6,F8,F2
```

3. >>

- 数据相关及其处理技
  - Tomasulo算法

- 每个保留站有以下7个字段：
  - Op：要对源操作数进行的操作
  - Qj，Qk：将产生源操作数的保留站号
    - 等于0表示操作数已经就绪且在Vj或Vk中，或者不需要操作数。
  - Vj，Vk：源操作数的值
    - 对于每一个操作数来说，V或Q字段只有一个有效。
    - 对于load来说，Vk字段用于保存偏移量。
  - Busy：为“yes”表示本保留站或缓冲单元“忙”
  - A：仅load和store缓冲器有该字段。开始是存放指令中的立即数字段，地址计算后存放有效地址。

指令	指令状态表		
	流出	执行	写结果
L.D F6 , 34(R2)	√	√	√
L.D F2 , 45(R3)	√	√	
MUL.D F0 , F2 , F4	√		
SUB.D F8 , F6 , F2	√		
DIV.D F10 , F0 , F6	√		
ADD.D F6 , F8 , F2	√		

名称	保留站						
	Busy	Op	Vj	Vk	Qj	Qk	A
Load1	no						
Load2	yes	LD					45+Regs[R3]
Add1	yes	SUB		Mem[34+Regs[R2]]	Load2		
Add2	yes	ADD			Add1	Load2	
Add3	no						
Mult1	yes	MUL		Reg[F4]	Load2		
Mult2	yes	DIV		Mem[34+Regs[R2]]	Mult1		

➢ 当第一条指令完成并写入结果时的状态表如右图：

	寄存器状态表							
	F0	F2	F4	F6	F8	F10	…	F30
Qi	Mult1	Load2		Add2	Add1	Mult2	…	

北京郵電大學 BEIJING UNIVERSITY OF POSTS AND TELECOMMUNICATIONS

·3.指令集并行技术

- **例题：** 假设有一条长流水线，仅仅对条件转移指令使用分支目标缓冲。假设分支预测错误的开销为4个时钟周期，缓冲不命中的开销为3个时钟周期。假设：命中率为90%，预测精度为90%，分支频率为15%，没有分支的基本CPI为1。

a) 求程序执行的CPI。

b) 相对于采用固定的2个时钟周期延迟的分支处理，哪种方法程序执行速度更快？

解： （1）程序执行的CPI = 没有分支的基本CPI（1） + 分支带来的额外开销

分支带来的额外开销是指在分支指令中，缓冲命中但预测错误带来的开销与缓冲没有命中带来的开销之和。

分支带来的额外开销= 15% * (90%命中×10%预测错误×4 + 10％没命中×3)= 0.099

所以，程序执行的CPI ＝ 1 ＋ 0.099 = 1.099

（2）采用固定的2 个时钟周期延迟的分支处理CPI = 1 + 15%×2 = 1.3

由（1）（2）可知分支目标缓冲方法执行速度快。

·3.指令集并行技术

例题： 假设分支目标缓冲的命中率为90%，程序中无条件转移指令的比例为5%，没有无条件转移指令的程序的CPI值为1。假设分支目标缓冲中包含分支目标指令，允许无条件转移指令进入分支目标缓冲，则程序的CPI值为多少？假设原来的CPI=1.1

解：

（1）原来不采用分支目标缓冲器BTB情况下

实际CPI = 理想CPI+各种停顿拍数

=1 + 5%×L + 95%×0

=1.1

解出L=2

（2）现在采用分支目标缓冲器BTB情况下

实际CPI = 理想CPI+各种停顿拍数

=1 + 5%×{ 90%×0+10%×2 } +95%×0

=1.01

主要的概念

- 向量的处理方式
- 向量处理机的结构

  两种典型的结构

  存储器-存储器型结构

  纵向处理方式采用

  寄存器-寄存器型结构

  分组处理方式采用

- 提高向量处理机性能的方法
- 链接技术

设置多个功能部件，使它们并行工作；

采用链接技术，加快一串向量指令的执行；

采用循环开采技术，加快循环的处理；

采用多处理机系统，进一步提高性能。

北京郵電大學 BEIJING UNIVERSITY OF POSTS AND TELECOMMUNICATIONS

·4.向量处理机

例题： 在CRAY-1机器上，按照链接方式执行下述4条向量指令（括号中给出了相应功能部件的执行时间），如果向量寄存器和功能部件之间的数据传送需要1拍，试求此链接流水线的通过时间是多少拍？如果向量长度为64，则需多少拍才能得到全部结果？

V_0←存储器　　（从存储器中取数：7拍）

V_2←V_0+V_1　　（向量加：3拍）

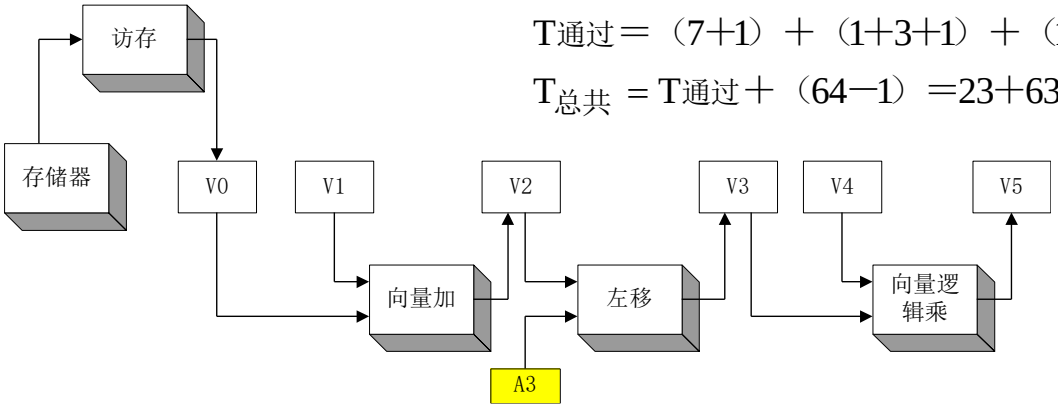
V_3←V_2<A_3　　（按（A_3）左移：4拍）

V_5←V_3∧V_4　　（向量逻辑乘：2拍）

解： 通过时间就是每条向量指令的第一个操作数执行完毕需要的时间，也就是各功能流水线由空到满的时间，具体过程如下图所示。要得到全部结果，在流水线充满之后，向量中后继操作数继续以流水方式执行，直到整组向量执行完毕。

$T_{通过}$＝（7+1）＋（1+3+1）＋（1+4+1）＋（1+2+1）＝23（拍）

$T_{总共}$ ＝$T_{通过}$＋（64−1）＝23+63=86（拍）



- 提高向量处理机性能的常用技术
  - 链接技术

例4.1 考虑在Cray-1上利用链接技术执行以下4条指令：

V_0←存储器　　// 访存取向量：7拍

V_2←V_0＋V_1　　// 向量加：3拍

V_3←V_2<A_3　　// 按（A_3）左移：4拍

V_5←V_3∧V_4　　// 与操作：2拍

画出链接示意图，并求该链接流水线的通过时间。如果向量长度为64，则需要多少拍才能得到全部结果。

解 对这4条指令进行分析可知：它们既没有部件冲突，也没有寄存器冲突，相邻两条指令之间都存在先写后读相关，因而可以把访存流水线、向量加流水线、向量移位流水线以及向量逻辑运算流水线链接成一个较长的流水线。

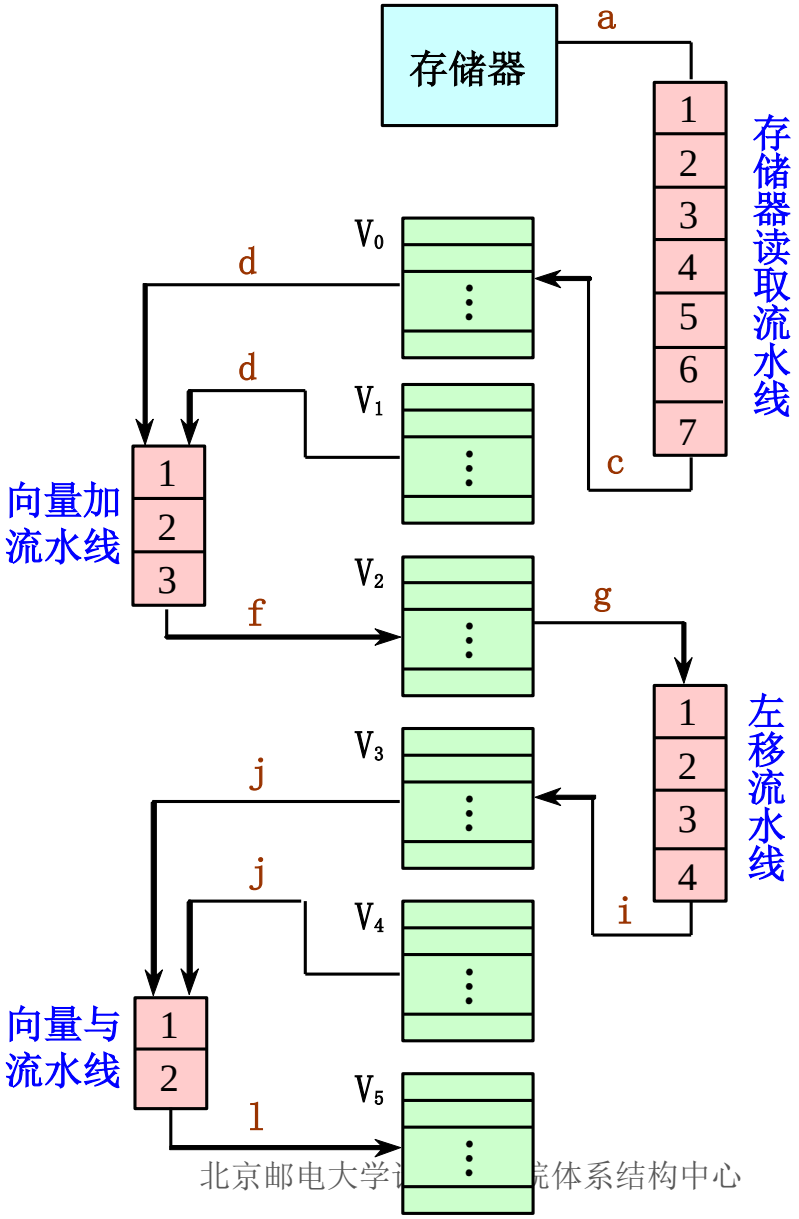
4.3提高向量处理机性能的常用技术 >>>链接技术

$V_0 \leftarrow$ 存储器

$V_2 \leftarrow V_0 + V_1$

$V_3 \leftarrow V_2 < A_3$

$V_5 \leftarrow V_3 \wedge V_4$



Cray-1的流水线链接举例

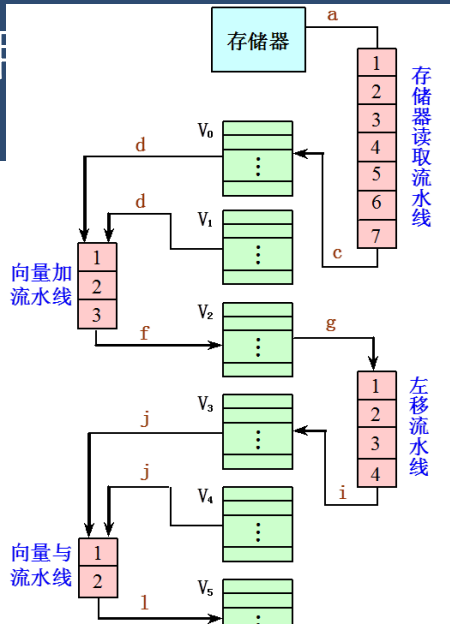
4.3提高向量处理机性能的常用 >>>链接技术

$V_0 \leftarrow$ 存储器

$V_2 \leftarrow V_0 + V_1$

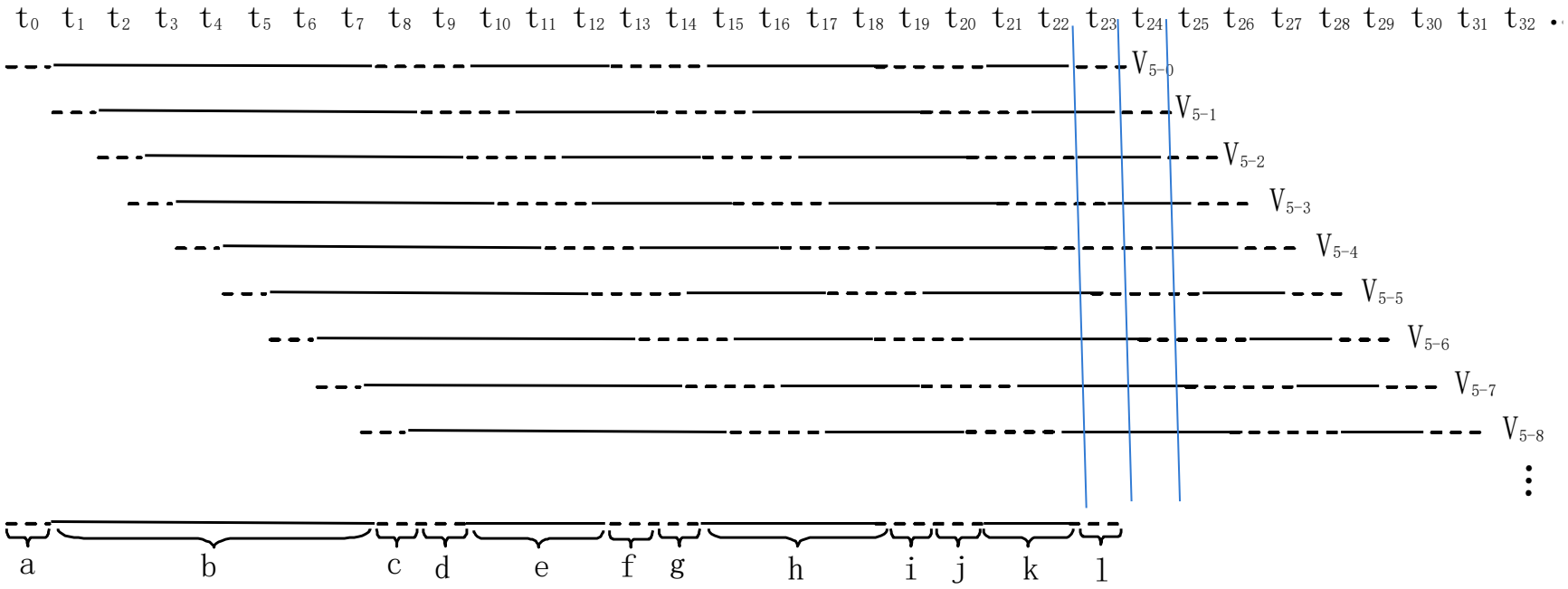
$V_3 \leftarrow V_2 < A_3$

$V_5 \leftarrow V_3 \wedge V_4$



链接操作的时间图：

24+(64-1)=87拍



4.3提高向量处理机性能的常用技术 >>>链接技术

a：存储字到“读功能部件”的传送时间

b：存储字经过“读功能部件”的通过时间

c：存储字从“读功能部件”到V_0分量的传送时间

d：V_0和V_1中操作数到整数加功能部件的传送时间

e：整数加功能部件的通过时间

f：和从整数加功能部件到V_2分量的传送时间

g：V_2中的操作数分量到移位功能部件的传送时间

h：移位功能部件的通过时间

i：结果从移位功能部件到V_3分量的传送时间

j：V_3和V_4中的操作数分量到逻辑部件的传送时间

k：逻辑功能部件的通过时间

l：最后结果到V_5分量的传送时间

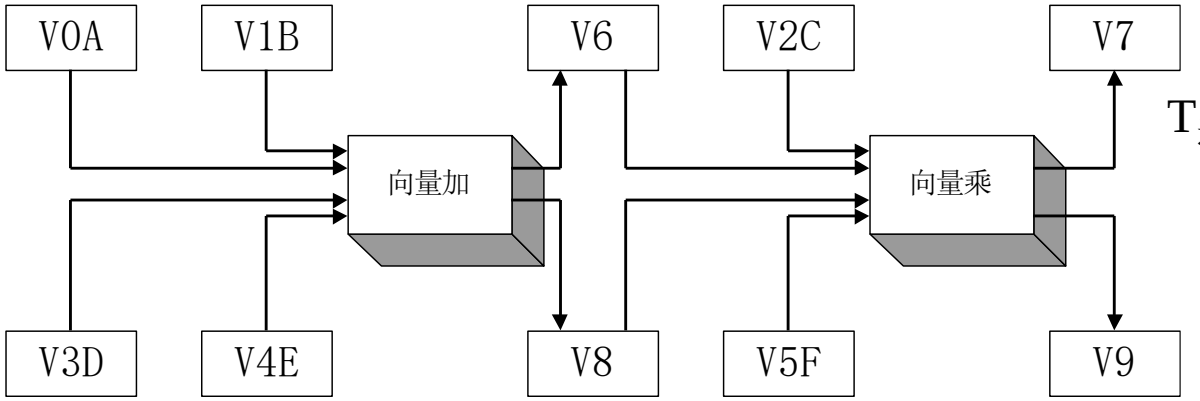
·4.向量处理机

例题： 某向量处理机有16个向量寄存器，其中V_0~V_5中分别放有向量A、B、C、D、E、F，向量长度均为8，向量各元素均为浮点数；处理部件采用两条单功能流水线，加法功能部件时间为2拍，乘法功能部件时间为3拍。采用类似于CARY-1的链接技术，先计算（A+B）*C，在流水线不停流的情况下，接着计算（D+E）*F。

(1)求此链接流水线的通过时间？（设寄存器入、出各需1拍）

(2)假如每拍时间为50ns，完成这些计算并把结果存进相应寄存器，此处理部件的实际吞吐率为多少MFLOPS？

解: （1）我们在这里假设A＋B的中间结果放在V6中，（A＋B）×C地最后结果放在V7中，D＋E地中间结果放在V8中，（D＋E）×F的最后结果放在V9中。具体实现参考下图：



通过时间应该为前者（（A＋B）×C）通过的时间：

$T_{通过}$= (1+2+1)+(1+3+1) =9（拍）

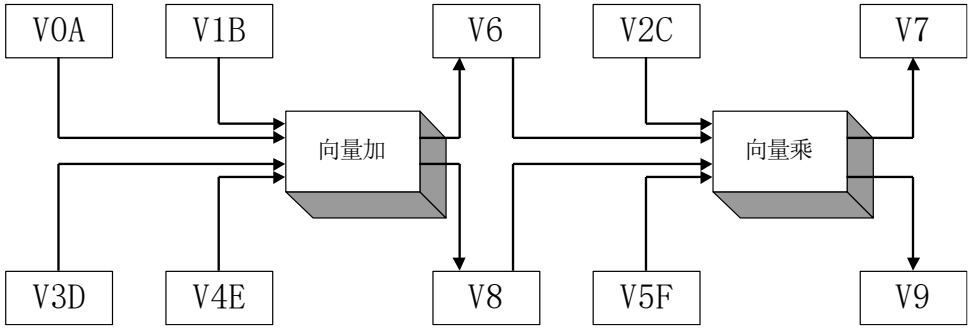
·4.向量处理机

例题： 某向量处理机有16个向量寄存器，其中V_0~V_5中分别放有向量A、B、C、D、E、F，向量长度均为8，向量各元素均为浮点数；处理部件采用两条单功能流水线，加法功能部件时间为2拍，乘法功能部件时间为3拍。采用类似于CARY-1的链接技术，先计算（A+B）*C，在流水线不停流的情况下，接着计算（D+E）*F。

(1)求此链接流水线的通过时间？（设寄存器入、出各需1拍）

(2)假如每拍时间为50ns，完成这些计算并把结果存进相应寄存器，此处理部件的实际吞吐率为多少MFLOPS？

解：



V6←A＋B

V7←V6×C

V8←D＋E

V9←V8×F.

（2）在做完（A＋B）×C之后，作（C＋D）×E就不需要通过时间了。

$$T = T_{通过} + (8-1) + 8 = 24(拍) = 1200(ns)$$

$$TP = \frac{32}{T} = 26.67 MFLOPS$$

例4.2 在CRAY-1上用链接技术进行向量运算

D=A×（B+C）

假设向量长度N≤64，向量元素为浮点数，且向量B、C已存放在V_0和V_1中。画出链接示意图，并分析非链接执行和链接执行两种情况下的执行时间。

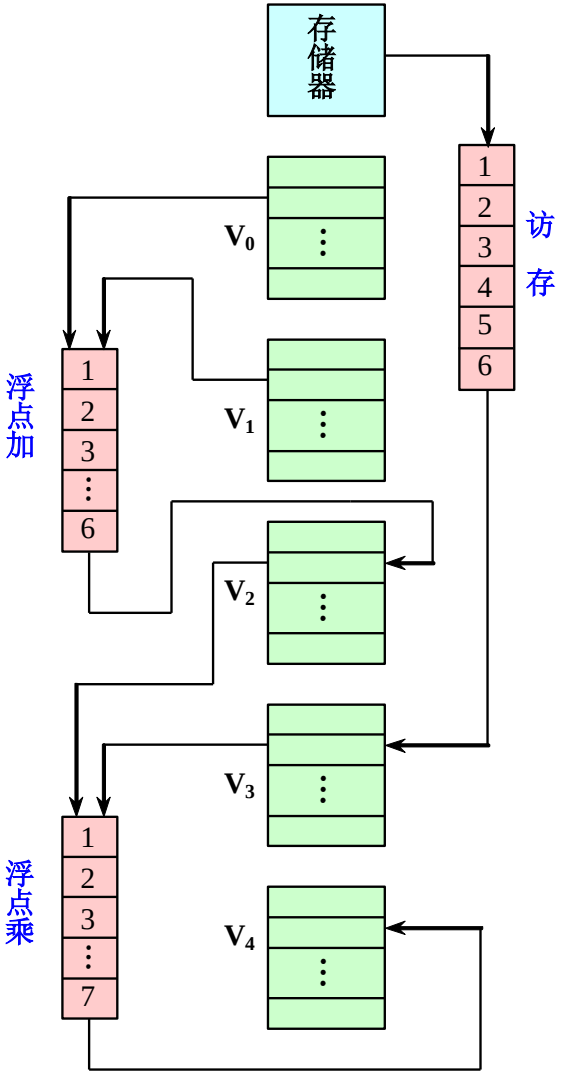
解 用以下三条向量完成上述运算：

V_3 ← 存储器 // 访存取向量A

V_2 ← V_0 + V_1 // 向量B和向量C进行浮点加

V_4 ← V_2 × V_3 // 浮点乘，结果存入V_4

4.3提高向量处理机性能的常用技术 >>>链接技术



链接示意图

$V_3 \leftarrow$ 存储器

$V_2 \leftarrow V_0 + V_1$

$V_4 \leftarrow V_2 \times V_3$

$V_3 \leftarrow$ 存储器

$V_2 \leftarrow V_0 + V_1$

$V_4 \leftarrow V_2 \times V_3$

- 3条指令全部用串行方法执行，则执行时间为：

[（1+6+1）+N−1]+[（1+6+1）+N−1]
+[（1+7+1）+N−1] = 3N +22 （拍）

- 前两条指令并行执行，然后再串行执行第3条指令，则执行时间为：

[（1+6+1）+N−1]+[（1+7+1）+N−1]
= 2N +15 （拍）

- 第1、2条向量指令并行执行，并与第3条指令链接执行。

[（1+6+1）]+ [（1+7+1）] +（N−1）
= N+16 （拍）

- 向量处理机的性能评价
  - 向量指令的处理时间T_{vp}

例4.4 假设每种向量功能部件只有一个，而且不考虑向量链接，那么下面的一组向量指令能分成几个编队？

```
LV      V1，Rx      //取向量x
MULTSV  V2，R0，V1  //向量x和标量（R0）相乘
LV      V3，Ry      //取向量y
ADDV    V4，V2，V3  //相加，结果保存到V4中
SV      Ry，V4      //存结果
```

解：分为四个编队

- 第一编队：LV
- 第二编队：MULTSV； LV
- 第三编队：ADDV
- 第四编队：SV

- 向量处理机的性能评价
  - 向量指令的处理时间T_{vp}

例4.5 在某向量处理机上执行DAXPY的向量指令序列，也即完成：

$$Y = a \times X + Y$$

其中X和Y是向量，最初保存在主存中，α是一个标量，已存放在寄存器F0中。它们的向量指令序列如下：

```
LV      V1，Rx
MULTFV  V2，F0，V1
LV      V3，Ry
ADDV    V4，V2，V3
SV      V4，Ry
```

- 向量处理机的性能评价
  - 向量指令的处理时间T_{vp}

假设向量寄存器的长度MVL=64，T_{loop}=15，各功能部件的启动时间为：

取数和存数部件为12个时钟周期；

乘法部件为7个时钟周期；

加法部件为6个时钟周期。

分别对于不采用向量链接技术和采用链接技术的两种情况，求完成上述向量操作的总执行时间。

- 向量处理机的性能评价
  - 向量指令的处理时间T_{vp}

```
LV      V1，Rx
MULTFV  V2，F0，V1
LV      V3，Ry
ADDV    V4，V2，V3
SV      V4，Ry
```

解：当不采用向量链接技术时，可以把上述五条向量指令分成4个编队：

- 第一编队：LV V1，Rx；
- 第二编队：MULTFV V2，F0，V1；LV V3，Ry；
- 第三编队：ADDV V4，V2，V3；
- 第四编队：SV V4，Ry。

T_{start}=12+12+6+12，m=4

可知，对n个向量元素进行DAXPY表达式计算所需的时钟周期个数为：

$$T_n = \left\lceil \frac{n}{MVL} \right\rceil \times (T_{loop} + T_{start}) + mn$$

$$= \left\lceil \frac{n}{64} \right\rceil \times (15+12+12+6+12) + 4n = \left\lceil \frac{n}{64} \right\rceil \times 57 + 4n$$

```
LV      V1，Rx
MULTFV  V2，F0，V1
LV      V3，Ry
ADDV    V4，V2，V3
SV      V4，Ry
```

- 向量处理机的性能评价
  - 向量指令的处理时间T_{vp}

采用向量链接技术，那么上述5条向量指令的编队结果如下（m=3）

- 第一编队：LV V1,Rx；MULTFV V2,F0,V1；
- 第二编队：LV V3,Ry；ADDV V4,V2,V3；
- 第三编队： SV V4，Ry 。

前两个编队中各自的两条向量指令都可以链接执行。根据链接的含义可知：

- 第一编队启动需要12+7=19个时钟周期
- 第二个编队启动需要12+6=18个时钟周期
- 第三个编队启动仍然需要12个时钟周期

主要的概念

- 互连函数
  - 交换函数

$$E(x_{n-1}x_{n-2}\cdots x_{k+1}x_k x_{k-1}\cdots x_1x_0)=x_{n-1}x_{n-2}\cdots x_{k+1}\bar{x}_k x_{k-1}\cdots x_1x_0$$

  - 均匀洗牌函数

$$\sigma(x_{n-1}x_{n-2}\cdots x_1x_0)=x_{n-2}x_{n-3}\cdots x_1x_0x_{n-1}$$

  - 蝶式互连函数

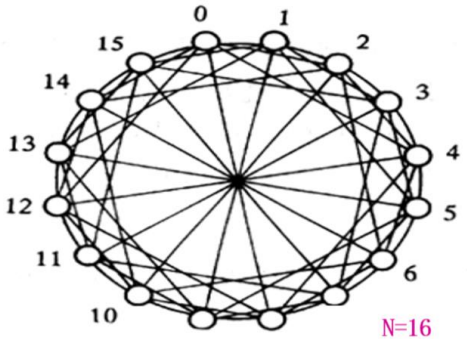
$$\beta(x_{n-1}x_{n-2}\cdots x_1x_0)=x_0x_{n-2}\cdots x_1x_{n-1}$$

  - 移数函数 $\alpha(x)=$ （x±k）mod N　1≤x≤N-1，1≤k≤N-1
  - PM2I函数

$$PM2_{+i}(x) = x+2^i \bmod N$$
$$PM2_{-i}(x) = x-2^i \bmod N$$

主要的概念

● 循环移数网络



(e)循环移数网络

- 一般地，如果 | j−i | =2^r（r=0, 1, 2, …, n−1，n=$\log_2 N$），则结点i与结点j连接。
  - 结点度：2n－1
  - 直径：n/2
  - 网络规模$N=2^n$

N=16

- 结点度：7
- 直径：2

● Omega网络

- 一个N输入的Omega网络
  - 有$\log_2 N$级，每级用N/2个2×2开关模块，共需要$N\log_2 N/2$个开关。
  - 每个开关模块均采用单元控制方式。
  - 不同的开关状态组合可实现各种置换、广播或从输入到输出的其它连接。
- N=8的多级立方体互连网络的另一种画法

主要的概念

- 四种寻径方式及其优缺点
  1. 线路交换(Circuit Switch)
  2. 存储转发(Store and Forward)
  3. 虚拟直通(Virtual cut through)
  4. 虫蚀方式（Wormhole）：把信息包“切割”成更小的单位——“片”，而且使信息包中各片的传送按流水方式进行。

5.互连网络

例题： 设函数的自变量是十进制数表示的处理机编号。现有32台处理机，其编号为0，1，2，…，31。

（1）分别计算下列互连函数

$Cube_2(12)$ $\sigma(8)$ $\beta(9)$ $PM2I_{+3}(28)$ $Cube_0(\sigma(4))$ $\sigma(Cube_0(18))$

（2）用$Cube_0$和σ构成均匀洗牌交换网（每步只能使用$Cube_0$和σ一次），网络直径是多少？从5号处理机发送数据到7号处理机，最短路径要经过几步？请列出经过的处理机编号。

（3）采用移数网络构成互连网，网络直径是多少？结点度是多少？与2号处理机距离最远的是几号处理机？

解： （1）共有32个处理机，表示处理机号的二进制地址应为5位。

N=32 的立方体交换函数

$$Cube_0\left(X_4X_3X_2X_1X_0\right) = X_4X_3X_2X_1\overline{X}_0$$

$$Cube_1\left(X_4X_3X_2X_1X_0\right) = X_4X_3X_2\overline{X}_1X_0$$

$$Cube_2\left(X_4X_3X_2X_1X_0\right) = X_4X_3\overline{X}_2X_1X_0$$

$Cube_2(12) = Cube_2(01100) = (01000)_2 = 8$

5.互连网络

例题： **σ(8) β(9) $PM2I_{+3}$(28) $Cube_0$(σ(4)) σ($Cube_0$(18))**

$$\sigma\left(x_{n-1}x_{n-2}\cdots x_1x_0\right)=x_{n-2}x_{n-3}\cdots x_1x_0x_{n-1}$$

σ(8)= σ(01000)=$(10000)_2$=16

$$\beta\left(x_{n-1}x_{n-2}\cdots x_1x_0\right)=x_0x_{n-2}\cdots x_1x_{n-1}$$

β(9)= β(01001)=$(11000)_2$=24

$PM2_{+i}(x) = x+2^i \bmod N$

$PM2_{-i}(x) = x-2^i \bmod N$

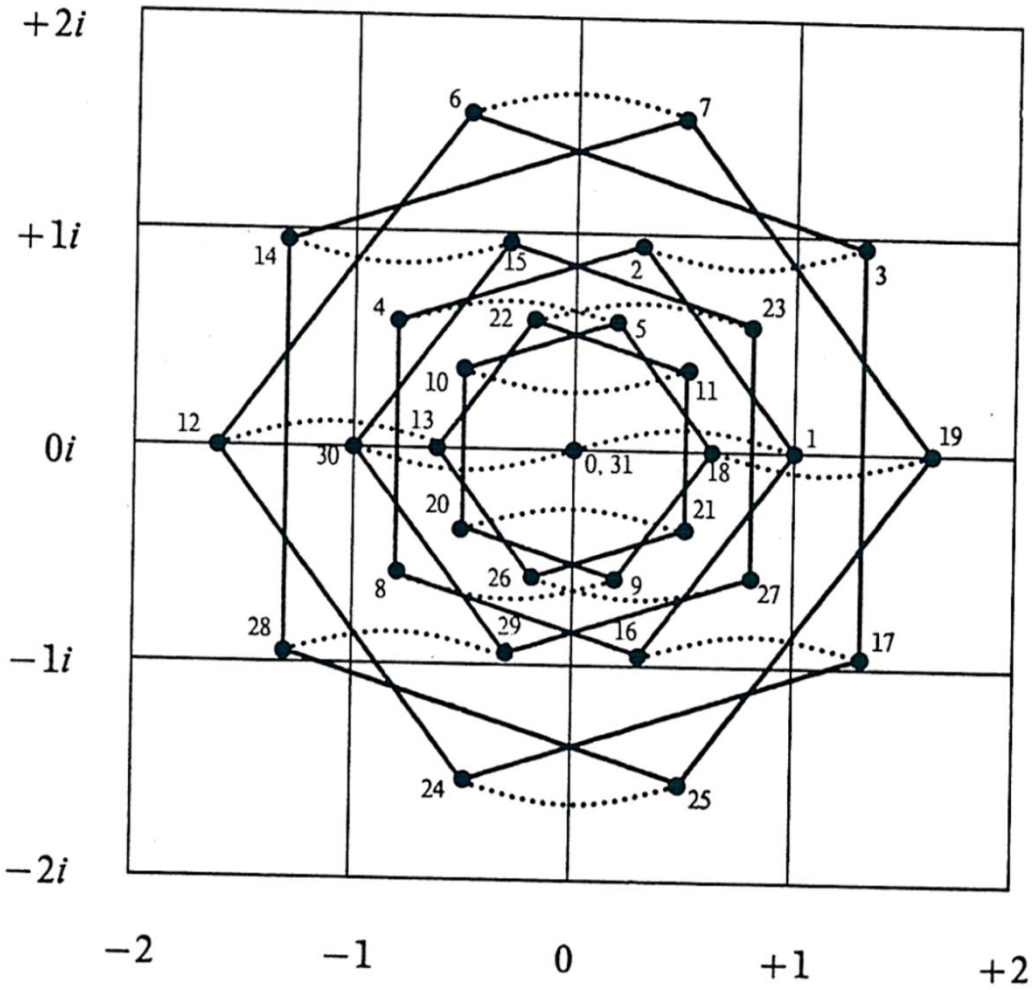
$PM2I_{+3}$(28)=28+8 mod 32=4

$Cube_0$(σ(4))= $Cube_0$(σ(00100))= $Cube_0$(01000)=$(01001)_2$=9

σ($Cube_0$(18)) = σ($Cube_0$(10010))= σ(10011)=$(00111)_2$=7

5.互连网络

例题： $\sigma(8)\ \beta(9)\ PM2I_{+3}(28)\ Cube_0(\sigma(4))\ \sigma(Cube_0(18))$



均匀洗牌交换网

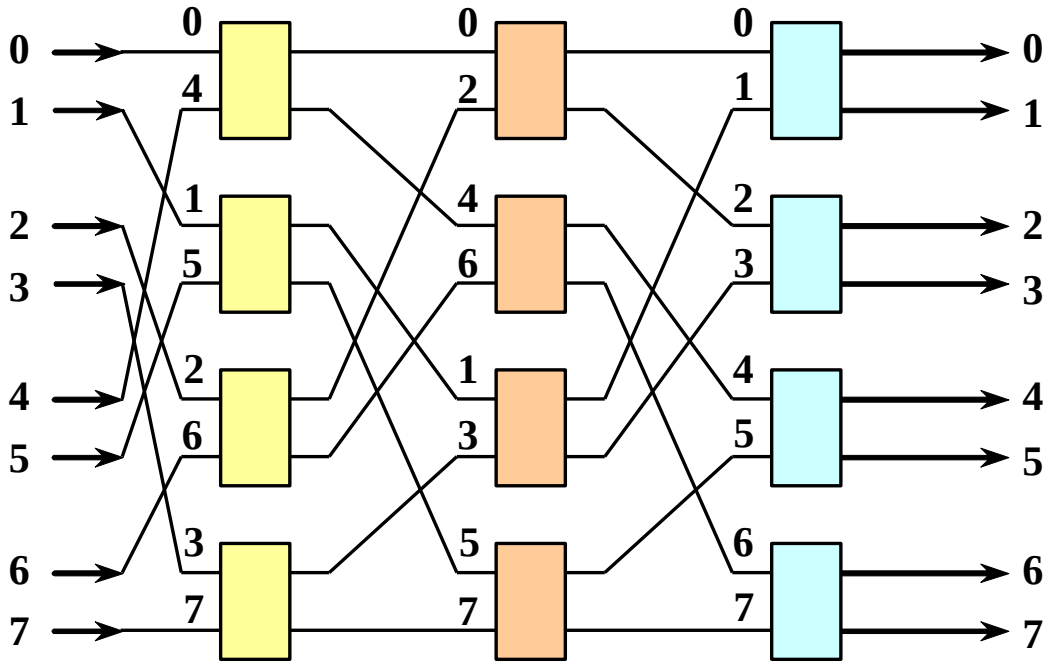
例题： $\sigma(8)\ \beta(9)\ PM2I_{+3}(28)\ Cube_0(\sigma(4))\ \sigma(Cube_0(18))$

（3）网络直径是3，结点度是9，与2号处理机距离最远的是13、15、21、23号处理机。

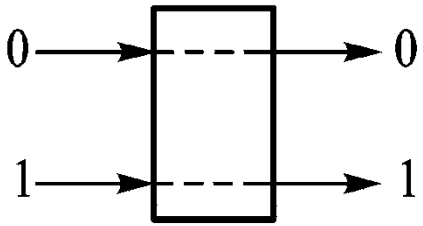
00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15

16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

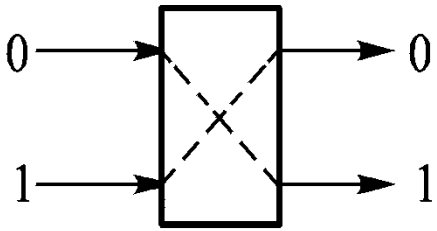
- 交叉开关网络(Crossbar network)
  - 多级互连网络(Multistage interconnection networks (MINs))
    - Omega网络
      - 一个8×8的Omega网络
        - 每级由4个4功能的2×2开关构成
        - 级间互连采用均匀洗牌连接方式



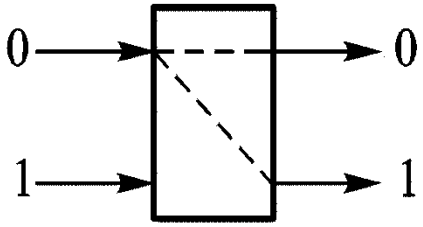
- 最简单的开关模块：2×2开关
  2×2开关的4种连接方式



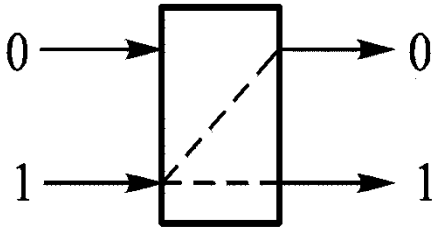
(a) 直送



(b)交叉



(c)上播



(d)下播

例题： 具有$N=2^n$个输入端的Omega网络，采用单元控制。

（1）N个输入总共应有多少种不同的排列？

（2）该Omega网络通过一次可以实现的置换总共可有多少种是不同的？

（3）若N=8，计算一次通过能实现的置换数占全部排列的百分比。

解：（1）N个输入的不同排列数为N！。

（2）N个输入端、输出端的Omega网络有$n=\log_2 N$级开关级，每级开关级有N/2个2×2的4功能开关，总共有（N/2）$\log_2 N$个开关。置换连接是指网络的输入端与输出端的一对一连接，故只考虑2×2开关的2个功能状态，即直送与交叉。网络采用单元控制，因此，每个开关都根据连接要求处于2个功能状态中的一种状态，所以，由（N/2）$\log_2 N$个开关组成的Omega网络的开关状态的种树为：

$$2^{(N/2)\log_2 N} = N^{N/2}$$

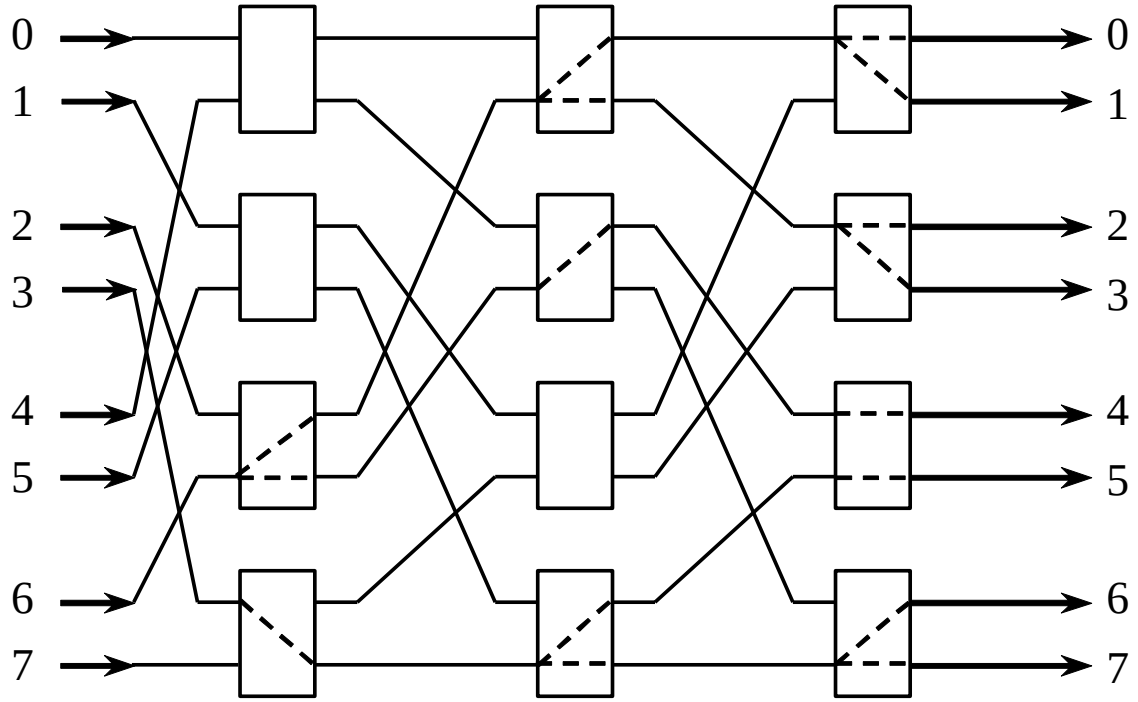
一种网络开关状态实现Omega网络的一种无冲突的置换连接，所以，一次使用Omega网络可以实现的无冲突的置换连接有$N^{N/2}$种。

（3）若N=8，则一次通过能实现的置换数占全部排列的百分比为：

$$\frac{N^{N/2}}{N!} = \frac{8^4}{8!} = \frac{4096}{40320} = 10.16\%$$

5.互连网络

例题： 用一个N=8的三级Omega网络连接8个处理机（P_0~P_7），8个处理机的输出端分别依序连接Omega网络的8个输入端0~7，8个处理机的输入端分别依序连接Omega网络的8个输出端0~7。如果处理机P_6要把数据播送给处理机P_0~P_4，处理机P_3要把数据播送给处理机P_5~P_7，那么，Omega网络能否同时为它们的播送要求实现连接？画出实现播送的Omega网络的开关状态图。



Omega网络使用的2×2开关有4种状态：直送、交叉、上播、下播。置换连接只使用直送和交叉状态，播送连接还需要使用上播和下播状态。如果没有开关状态和开关输出端争用冲突，就可以使用播送连接。实际上，它们的播送要求没有冲突，因此，可以同时实现

·6.阵列处理机

主要的概念

- 什么是阵列处理机？
- 阵列处理机的主要特点

主要的概念

- 什么是机群（Cluster of Workstations）？
- SSI包含四重含义？
- Beowulf机群的主要特点

主要的概念

- 什么是缓存一致性？
- 共享Cache的两种更新协议？
- Cache一致性协议的两大类

在多个处理器中用来维护一致性的协议。

- 关键：跟踪记录共享数据块的状态
- 两类协议（采用不同的共享数据状态跟踪技术）
  - 目录法（directory）

    物理存储器中共享数据块的状态及相关信息均被保存在一个称为目录的地方。
  - 监听法（snooping）
    - 每个Cache除了包含物理存储器中块的数据副本之外，也保存着各个块的共享状态信息。
    - Cache通常连在共享存储器的总线上，各个Cache控制器通过监听总线来判断它们是否有总线上请求的数据块。

谢谢！

祝同学们考得好分数