计算机系统结构
第四章 向量处理机

主　讲：刘超

中国地质大学（武汉）计算机学院

向量流水处理机

- 4.1向量的处理方式
- 4.2向量处理机的结构
- 4.3提高向量处理机性能的常用技术
- 4.4向量处理机的性能评价
- 4.5向量处理机实例
- 4.6习题

向量流水处理机

- 向量由一组有序、具有相同类型和位数的元素组成。
- 在流水线处理机中，设置向量数据表示和相应的向量指令，称为向量处理机。
- 不具有向量数据表示和相应的向量指令的流水线处理机，称为标量处理机。
- 典型的向量处理机
  - 1976年 Cray-1超级计算机，浮点运算速度达到了每秒1亿次
  - CDC Cyber 205，Cray Y-MP，NEC SX-X/44，Fujitsu VP2600等，性能达到了每秒几十亿～几百亿次浮点运算

4.1 向量的处理方式

以计算表达式 D=A×（B－C）为例

A、B、C、D —— 长度为 N 的向量

4.1 向量的处理方式

- 横向(水平)处理方式
  - 向量计算是按行的方式从左到右横向地进行。
    - 先计算：　$d_1 \leftarrow a_1 \times (b_1 - c_1)$
    - 再计算：　$d_2 \leftarrow a_2 \times (b_2 - c_2)$
    - ……
    - 最后计算：　$d_N \leftarrow a_N \times (b_N - c_N)$
  - 组成循环程序进行处理。

    $q_i \leftarrow b_i - c_i$

    $d_i \leftarrow q_i \times a_i$
    - 数据相关：N次　　功能切换：2N次
  - 不适合于向量处理机的并行处理。

4.1 向量的处理方式

2. 纵向 (垂直)处理方式

- 向量计算是按列的方式从上到下纵向地进行。

先计算
$$\begin{cases} q_1 \leftarrow b_1 - c_1 \\ \cdots\cdots \\ q_N \leftarrow b_N - c_N \end{cases}$$
再计算
$$\begin{cases} d_1 \leftarrow q_1 \times a_1 \\ \cdots\cdots \\ d_N \leftarrow q_N \times a_N \end{cases}$$

- 表示成向量指令：

  $Q=B-C$

  $D=Q\times A$

- 两条向量指令之间：

  数据相关：1次　　功能切换：1次

4.1 向量的处理方式

3. 纵横 (分组)处理方式

- 又称为分组处理方式。
- 把向量分成若干组，组内按纵向方式处理，依次处理各组。
- 对于上述的例子，设：

$$N=S\times n+r$$

  - 其中N为向量长度，S为组数，n为每组的长度，r为余数。
  - 若余下的r个数也作为一组处理，则共有S+1组。
- 运算过程为：

4.1 向量的处理方式

- 先算第1组：

$$Q_{1\sim n} \leftarrow B_{1\sim n} - C_{1\sim n}$$

$$D_{1\sim n} \leftarrow Q_{1\sim n} \times A_{1\sim n}$$

- 再算第2组：

$$Q_{(n+1)\sim 2n} \leftarrow B_{(n+1)\sim 2n} - C_{(n+1)\sim 2n}$$

$$D_{(n+1)\sim 2n} \leftarrow Q_{(n+1)\sim 2n} \times A_{(n+1)\sim 2n}$$

- 依次进行下去，直到最后一组：第S+1组。
- 每组内各用两条向量指令。

数据相关：1次　　功能切换：2次

4.2 向量处理机的结构

- **向量处理机的结构因具体机器不同而不同。**

  由所采用的向量处理方式决定。

- **有两种典型的结构**
  - 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

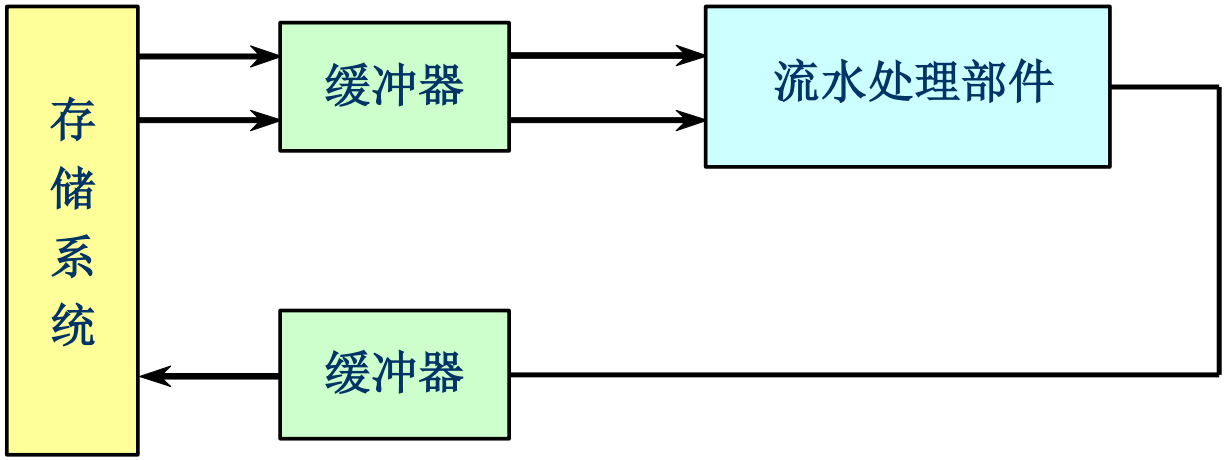
4.2 向量处理机的结构

4.2.1 “存储器-存储器”结构

1. 采用纵向处理方式的向量处理机对处理机结构的要求：存储器－存储器结构

   - 向量指令的源向量和目的向量都是存放在存储器中，运算的中间结果需要送回存储器。
   - 流水线运算部件的输入和输出端都直接（或经过缓冲器）与存储器相联，从而构成存储器-存储器型操作的运算流水线。
     - 例如：STAR-100、CYBER-205

4.2 向量处理机的结构



“存储器—存储器”型操作的运算流水线

4.2 向量处理机的结构

2. **要充分发挥这种结构的流水线效率，存储器要不断地提供源操作数，并不断地从运算部件接收结果。**

（每拍从存储器读取两个数据，并向存储器写回一个结果）

- 对存储器的带宽以及存储器与处理部件的通信带宽提出了非常高的要求。
- 解决方法：一般是通过采用多体交叉并行存储器和缓冲器技术。

例如，70年代初问世的Star 100

- 存储器：32个体交叉
- 每个体的数据宽度：8个字（字长64位）
- 最大数据流量：每秒2亿字

4.2 向量处理机的结构

4.2.2 “寄存器-寄存器”结构

在向量的分组处理方式中，对向量长度N没有限制，但组的长度n却是固定不变的。

- 对处理机结构的要求：寄存器—寄存器结构
- 设置能快速访问的向量寄存器，用于存放源向量、目的向量及中间结果。让运算部件的输入、输出端都与向量寄存器相联，就构成了“寄存器—寄存器”型操作的运算流水线。
  - 典型的寄存器—寄存器结构的向量处理机

    美国的CRAY-1、我国的YH-1巨型机

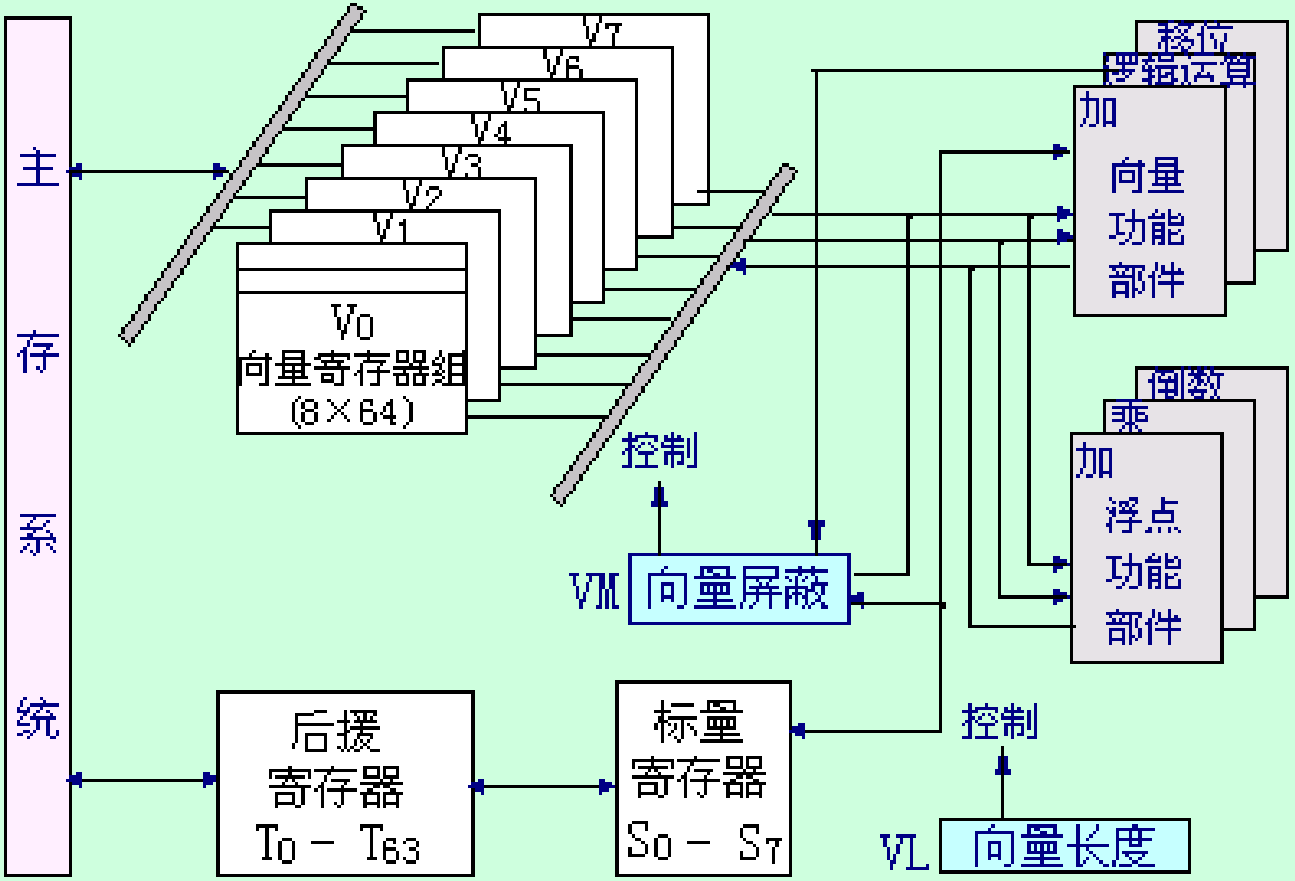
4.2 向量处理机的结构

- 以CRAY-1机为例
  - 美国CRAY公司
  - 1976年
  - 每秒1亿次浮点运算
  - 时钟周期：12.5ns

- **CRAY-1的基本结构**
  - 功能部件

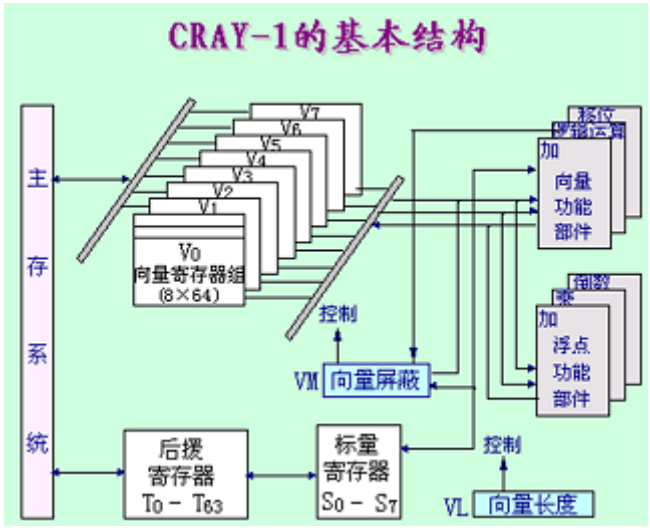
    共有12条可并行工作的单功能流水线，可分别流水地进行地址、向量、标量的各种运算。

CRAY-1的基本结构


主存系统
V7
V6
V5
V4
V3
V2
V1
V0
向量寄存器组
(8×64)
移位
逻辑运算
加
向量
功能
部件
倒数
乘
加
浮点
功能
部件
控制
VM
向量屏蔽
后援
寄存器
T0 - T63
标量
寄存器
S0 - S7
控制
VL
向量长度

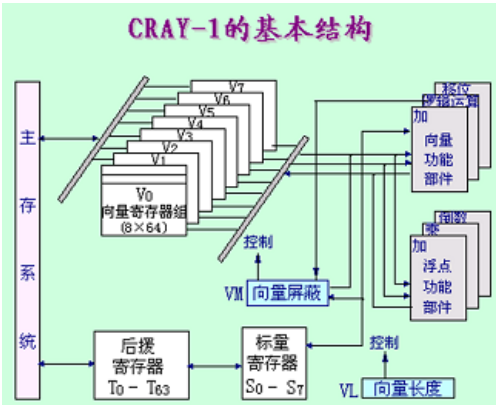
4.2 向量处理机的结构

- 6个单功能流水部件：进行向量运算
  - 整数加（3拍）
  - 逻辑运算（2拍）
  - 移位（4拍）
  - 浮点加（6拍）
  - 浮点乘（7拍）
  - 浮点迭代求倒数（14拍）



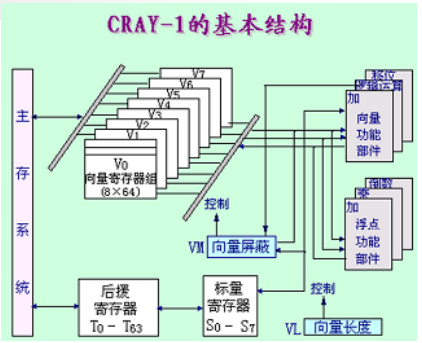
括号中的数字为其流水经过的时间，每拍为一个时钟周期，即12.5ns。

4.2 向量处理机的结构



- 向量寄存组V
  - 由512个64位的寄存器组成，分成8块。
  - 编号：$V_0 \sim V_7$
  - 每一个块称为一个向量寄存器，可存放一个长度（即元素个数）不超过64的向量。
  - 每个向量寄存器可以每拍向功能部件提供一个数据元素，或者每拍接收一个从功能部件来的结果元素。
- 标量寄存器S和快速暂存器T
  - 标量寄存器有8个：$S_0 \sim S_7$　　64位
  - 快速暂存器T用于在标量寄存器和存储器之间提供缓冲。

4.2 向量处理机的结构



- 向量屏蔽寄存器VM
  - 64位，每一位对应于向量寄存器的一个单元。
  - 作用：用于向量的归并、压缩、还原和测试操作、对向量某些元素的单独运算等。

2. CRAY-1向量处理的一个显著特点

- 每个向量寄存器V_i都有连到6个向量功能部件的单独总线。
- 每个向量功能部件也都有把运算结果送回向量寄存器组的总线。

4.2 向量处理机的结构

- 只要不出现V_i冲突和功能部件冲突，各V_i之间和各功能部件之间都能并行工作，大大加快了向量指令的处理。
  - V_i冲突：并行工作的各向量指令的源向量或结果向量使用了相同的V_i。

  例如：源向量相同

  $$V_3 \leftarrow V_1 + V_2$$

  $$V_5 \leftarrow V_4 \wedge V_1$$

  - 功能部件冲突：并行工作的各向量指令要使用同一个功能部件。

  例如：都需使用乘法功能部件

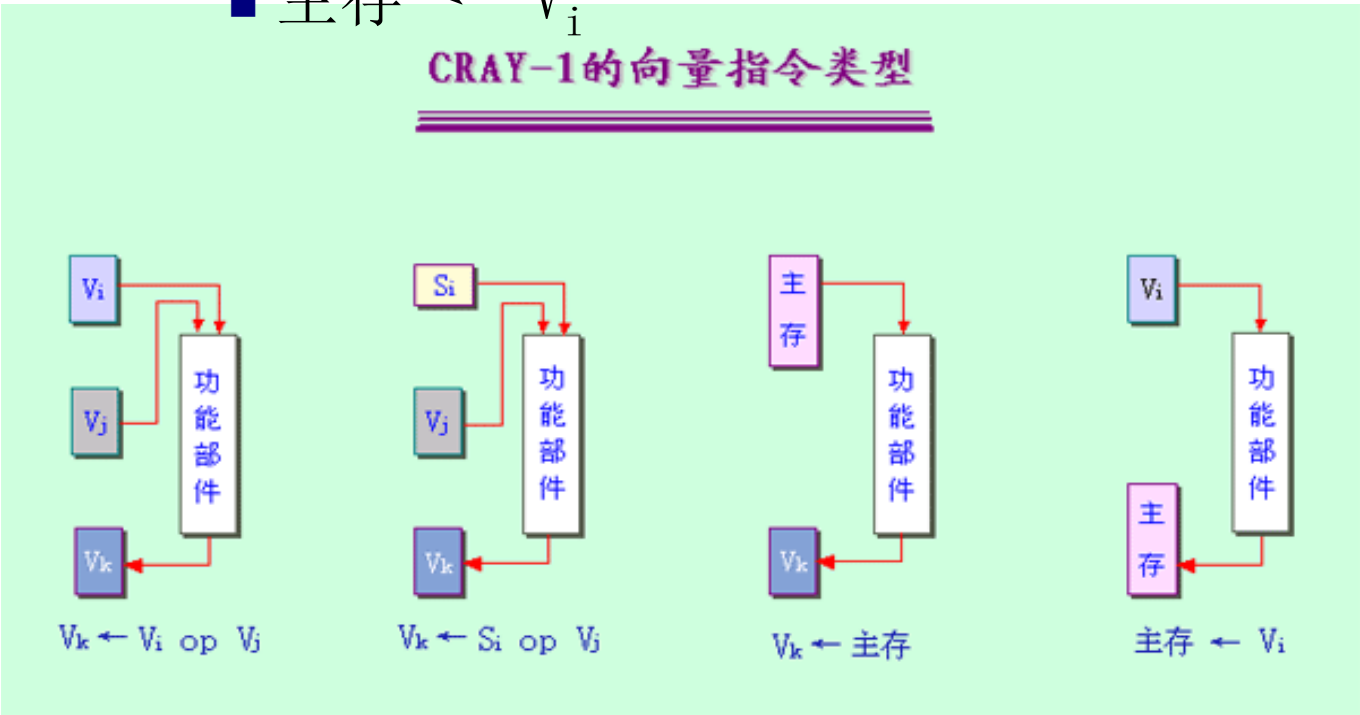
  $$V_3 \leftarrow V_1 \times V_2$$

  $$V_5 \leftarrow V_4 \times V_6$$

4.2 向量处理机的结构

3. CRAY-1向量指令类型

- $V_k \leftarrow V_i$ op V_j
- $V_k \leftarrow S_i$ op V_j
- $V_k \leftarrow$ 主存
- 主存 $\leftarrow V_i$



4.3 提高向量处理机性能的常用技术

提高向量处理机性能的方法

- 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。

4.3 提高向量处理机性能的常用技术

4.3.1 设置多个功能部件

- 设置多个独立的功能部件。这些部件能并行工作，并各自按流水方式工作，从而形成了多条并行工作的运算操作流水线。

例如：CRAY-1向量处理机有4组12个单功能流水部件：

- 向量部件：向量加，移位，逻辑运算
- 浮点部件：浮点加，浮点乘，浮点求倒数
- 标量部件：标量加，移位，逻辑运算，数“1”/计数
- 地址运算部件：整数加，整数乘

4.3 提高向量处理机性能的常用技术

4.3.2 链接技术

- 两条向量指令占用功能流水线和向量寄存器的4种情况
  - 指令不相关

    例如：V0←V1＋V2

    V6←V4*V5

    - 这两条指令分别使用各自所需的流水线和向量寄存器，可以并行执行。
  - 功能部件冲突

    例如：V3←V1＋V2

    V6←V4＋V5

4.3 提高向量处理机性能的常用技术

- 这两条指令都要使用加法流水线，发生了功能部件冲突（但向量寄存器不冲突）。当第一条指令流出时，占用加法流水线。第二条指令要等加法流水线变成空闲后，才能流出。

V3←V1＋V2

V6←V4＋V5

- 源寄存器冲突

例如：V3←V1＋V2

V6←V1*V4

- 这两条向量指令的源向量之一都取自V1。由于两者的首元素下标可能不同，向量长度也可能不同，所以难以由V1同时提供两条指令所需要的源向量。
- 这两条向量指令不能同时执行。只有等第一条向量指令执行完、释放V1之后，第二条向量指令才能开始执行。

4.3 提高向量处理机性能的常用技术

- 结果寄存器冲突

  两条向量指令使用了相同的结果向量寄存器。

  例如：V4←V1＋V2

  V4←V3*V5

  - 这两条指令都要访问目的寄存器V4。由于第一条指令在先，所以它先占用V4直到运算完成，然后再流出后一条指令。

2. 当前一条指令的结果寄存器是后一条指令的源寄存器、且不存在任何其他冲突时，就可以用链接技术来提高性能。

例如：V3←V1＋V2

V6←V3*V4

4.3 提高向量处理机性能的常用技术

- 向量流水线链接：具有先写后读相关的两条指令，在不出现功能部件冲突和源向量冲突的情况下，可以把功能部件链接起来进行流水处理，以达到加快执行的目的。
  - Cray-1向量处理的一个显著特点
  - 链接特性的实质

    把流水线定向的思想引入到向量执行过程的结果。

4.3 提高向量处理机性能的常用技术

- 链接时，Cray-1中把向量数据元素送往向量功能部件以及把结果存入向量寄存器都需要一拍时间，从存储器中把数据送入访存功能部件也需要一拍时间。

（同步的要求）

4.3 提高向量处理机性能的常用技术

- 整数加（3拍）
- 逻辑运算（2拍）
- 移位（4拍）
- 浮点加（6拍）
- 浮点乘（7拍）
- 浮点迭代求倒数（14拍）

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

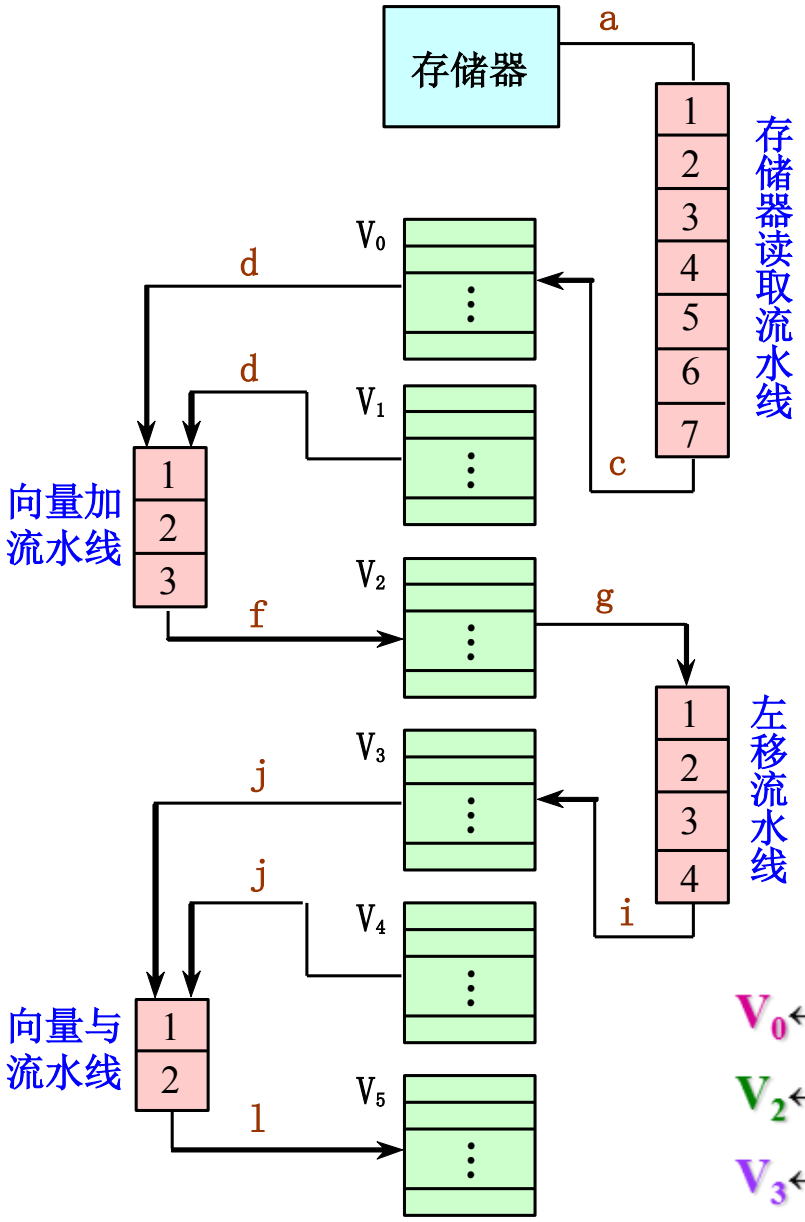
V_2←V_0＋V_1　　// 向量加：3拍

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。



Cray-1的流水线链接举例

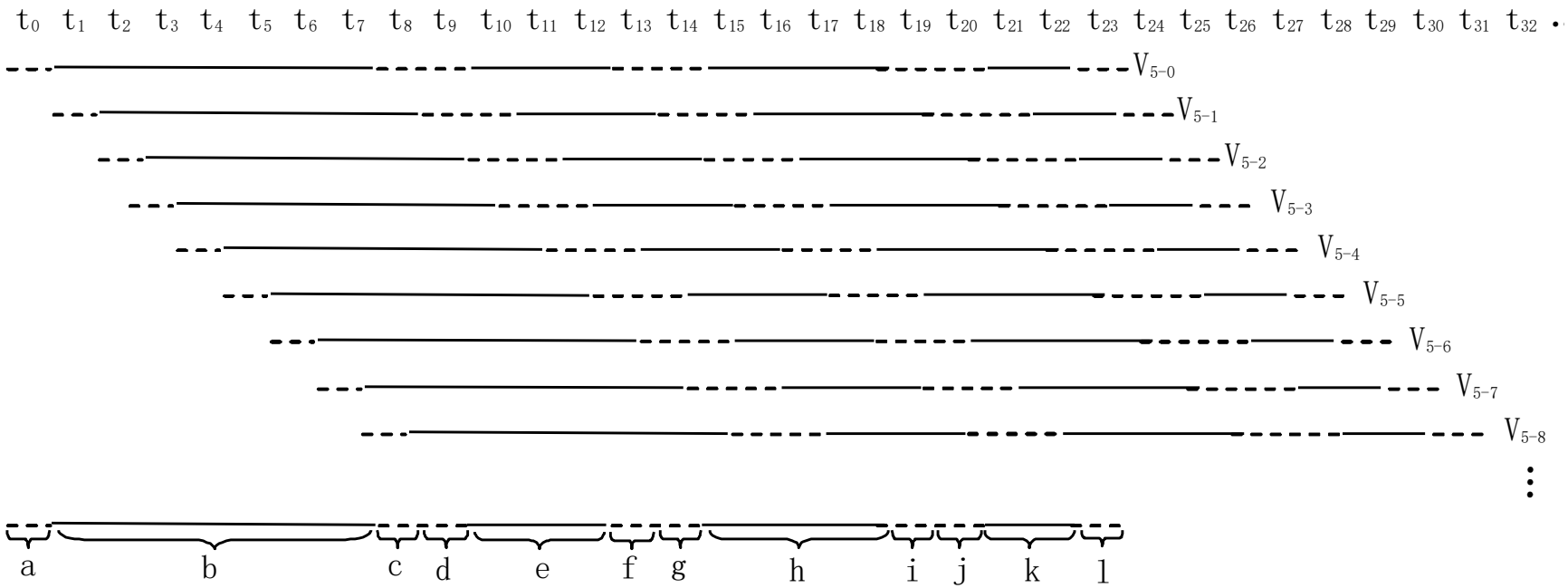
V_0←存储器　　// 访存取向量：7拍

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

链接操作的时间图：



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

4.3 提高向量处理机性能的常用技术

例4.2 在CRAY-1上用链接技术进行向量运算

$D=A\times(B+C)$

假设向量长度$N\leqslant 64$，向量元素为浮点数，且向量B、C已存放在V_0和V_1中。

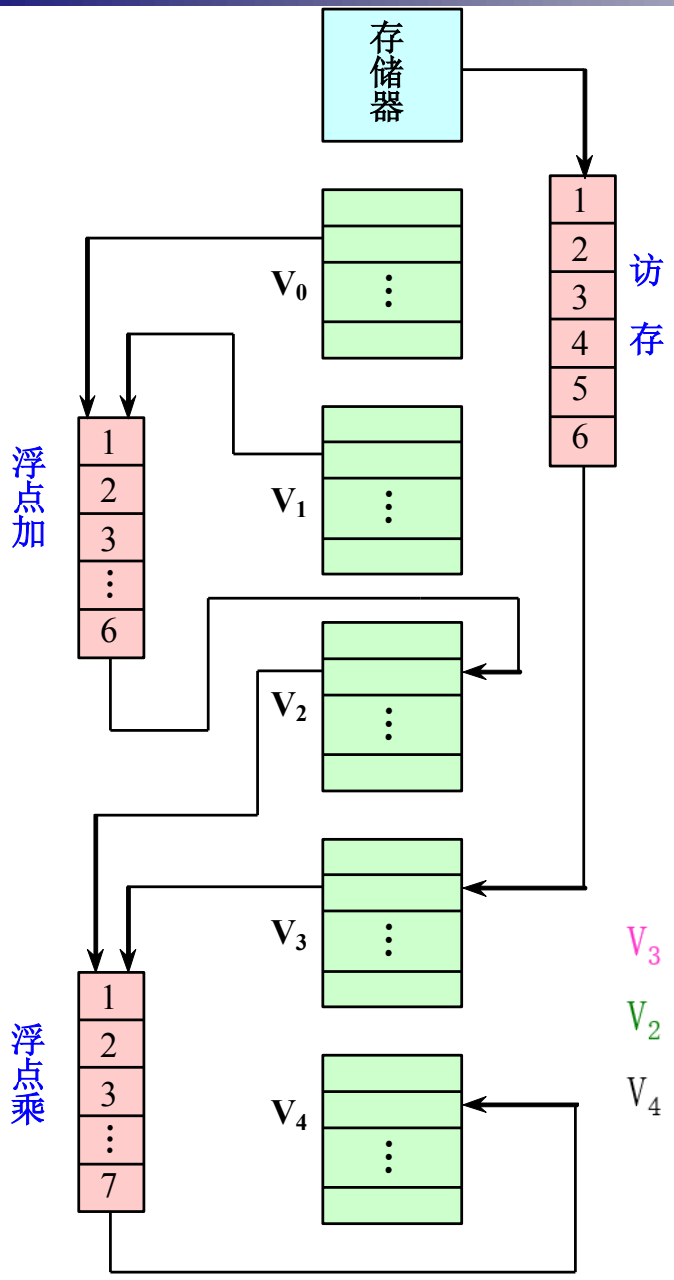
画出链接示意图，并分析非链接执行和链接执行两种情况下的执行时间。

解　用以下三条向量完成上述运算：

$V_3 \leftarrow$ 存储器　　// 访存取向量A，6拍

$V_2 \leftarrow V_0 + V_1$　　// 向量B和向量C进行浮点加，6拍

$V_4 \leftarrow V_2 \times V_3$　　// 浮点乘，结果存入V_4，7拍



链接示意图

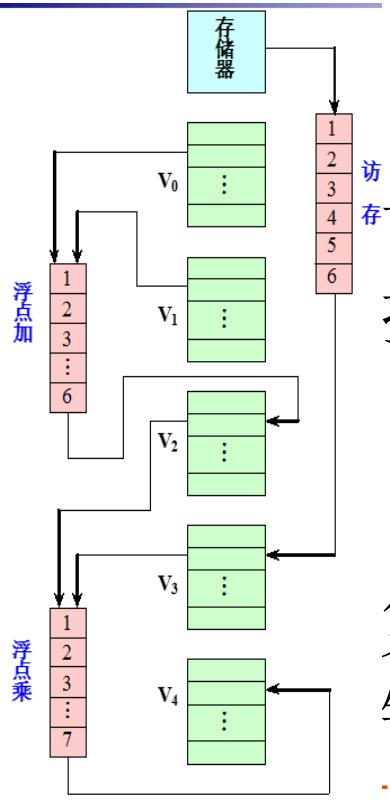
V_3 ← 存储器　　// 访存取向量A, 6拍

V_2 ← V_0 + V_1　　// 向量B和向量C进行浮点加，6拍

V_4 ← V_2 × V_3　　// 浮点乘，结果存入V_4，7拍

4.3 提高向量处理机性能的常用技术

- 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）

4.3 提高向量处理机性能的常用技术

- 进行向量链接的要求

保证：无向量寄存器使用冲突和无功能部件使用冲突

- 只有在前一条指令的第一个结果元素送入结果向量寄存器的那一个时钟周期才可以进行链接。
- 当一条向量指令的两个源操作数分别是两条先行指令的结果寄存器时，要求先行的两条指令产生运算结果的时间必须相等，即要求有关功能部件的通过时间相等。
- 要进行链接执行的向量指令的向量长度必须相等，否则无法进行链接。

4.3 提高向量处理机性能的常用技术

4.3.3 分段开采技术

如果向量的长度大于向量寄存器的长度，

该如何处理呢？

- 当向量的长度大于向量寄存器的长度时，必须把长向量分成长度固定的段，然后循环分段处理，每一次循环只处理一个向量段。
- 这种技术称为分段开采技术。
  - 由系统硬件和软件控制完成，对程序员是透明的。

4.3 提高向量处理机性能的常用技术

例4.3 设A和B是长度为N的向量，考虑在Cray-1向量处理器上实现以下的循环操作：

```
DO  10 I = 1, N
10  A (I) = 5.0 * B (I)  + C
```

4.3 提高向量处理机性能的常用技术

- 当$N \leqslant 64$时，可以用以下指令序列：

$S_1 \leftarrow 5.0$ ；将常数5.0送入标量寄存器S_1

$S_2 \leftarrow C$ ；将常数C送入标量寄存器S_2

$VL \leftarrow N$ ；在VL中设置向量长度N

$V_0 \leftarrow B$ ；从将向量B读入向量寄存器V_0

$V_1 \leftarrow S_1 \times V_0$；向量B中的每个元素分别和常数$S_1$相乘

$V_2 \leftarrow S_2 + V_1$ ；向量V_1的每个元素分别和常数S_2 相加

$A \leftarrow V_2$ ；将结果向量存入A数组

```
      DO  10 I = 1, N
10    A (I) = 5.0 * B (I)  + C
```

4.3 提高向量处理机性能的常用技术

- 当$N > 64$时，就需要进行分段开采。
  - 循环次数K ：

$$K = \left\lfloor \frac{N}{64} \right\rfloor$$

  - 余数L：

$$L = N - 64 \times \left\lfloor \frac{N}{64} \right\rfloor$$

$S_1 \leftarrow 5.0$
$S_2 \leftarrow C$
$VL \leftarrow N$
$V_0 \leftarrow B$
$V_1 \leftarrow S_1 \times V_0$
$V_2 \leftarrow S_2 + V_1$
$A \leftarrow V_2$

  - 在进入循环前，先对余数个元素进行计算，然后用循环的方式计算向量A的其他部分，每次循环计算64个元素，而循环体则是由上述第4条到第7条向量指令组成。

4.3 提高向量处理机性能的常用技术

4.3.4 采用多处理机系统

许多新型向量处理机系统采用了多处理机系统结构。例如：

- CRAY-2
  - 包含了4个向量处理机
  - 浮点运算速度最高可达1800MFLOPS
- CRAY Y-MP、C90

  最多可包含16个向量处理机

4.4 向量处理机的性能评价

衡量向量处理机性能的主要参数：

- 向量指令的处理时间
- 向量长度为无穷大时的向量处理机的最大性能
- 半性能向量长度
- 向量长度临界值

4.4 向量处理机的性能评价

4.4.1 向量指令的处理时间T_{vp}

- **一条向量指令的处理时间T_{vp}**
  - 执行一条向量长度为n的向量指令所需的时间为：

$$T_{vp} = T_s + T_e + (n-1)T_c$$

  - T_s：向量处理部件流水线的建立时间

    为了使处理部件流水线能开始工作（即开始流入数据）所需要的准备时间。

  - T_e：向量流水线的通过时间

    第一对向量元素通过流水线并产生第一个结果所花的时间 。

  - T_c：流水线的时钟周期时间

4.4 向量处理机的性能评价

- 把上式中的参数都折算成时钟周期个数：

$$T_{vp} = [s + e + (n-1)]T_c$$

  - s： T_s所对应的时钟周期数
  - e： T_e所对应的时钟周期数
- 不考虑T_s，并令T_{start}=e－1

$$T_{vp} = (T_{start} + n)T_c$$

  - T_{start}： 从一条向量指令开始执行到还差一个时钟周期就产生第一个结果所需的时钟周期数。可称之为该向量指令的启动时间。此后，便是每个时钟周期流出一个结果，共有n个结果。

4.4 向量处理机的性能评价

2. 一组向量指令的处理时间

- 对于一组向量指令而言，其执行时间主要取决于三个因素：
  - 向量的长度
  - 向量操作之间是否存在流水功能部件的使用冲突
  - 数据的相关性
- 把能在同一个时钟周期内一起开始执行的几条向量指令称为一个编队。

4.4 向量处理机的性能评价

- 可以看出：同一个编队中的向量指令之间一定不存在流水向量功能部件的冲突和数据的冲突。

- 编队后，这个向量指令序列的总的执行时间为各编队的执行时间的和 。

$$T_{all} = \sum_{i=1}^{m} T_{vp}^{(i)}$$

- $T^{(i)}_{vp}$：第i个编队的执行时间
- m：编队的个数

4.4 向量处理机的性能评价

- 当一个编队是由若干条指令组成时，其执行时间就应该由该编队中各指令的执行时间的最大值来确定。

  $T^{(i)}_{start}$：第i编队中各指令的启动时间的最大值

$$T_{all}=\sum_{i=1}^{m}T_{vp}^{(i)}=\sum_{i=1}^{m}(T_{start}^{(i)}+n)T_c=(\sum_{i=1}^{m}T_{start}^{(i)}+mn)T_c=(T_{start}+mn)T_c$$

$$T_{start}=\sum_{i=1}^{m}T_{start}^{(i)}$$ 该组指令总的启动时间（时钟周期个数）

- 表示成时钟周期个数

  $T_{all} = T_{start} + mn$ （拍）

4.4 向量处理机的性能评价

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

4.4 向量处理机的性能评价

3. 分段开采时一组向量指令的总执行时间

- 当向量长度n大于向量寄存器长度MVL时，需要分段开采。
- 引入一些额外的处理操作

  （假设：这些操作所引入的额外时间为T_{loop}个时钟周期）
  - 设 $\left\lfloor \frac{n}{MVL} \right\rfloor = p$　　q：余数
  - 共有m个编队
- 对于最后一次循环来说，所需要的时间为：

$$T_{last} = T_{start} + T_{loop} + m \times q$$

4.4 向量处理机的性能评价

- 其他的每一次循环所要花费的时间为：

$$T_{step} = T_{start} + T_{loop} + m \times MVL$$

- 总的执行时间为：

$$T_{all} = T_{step} \times p + T_{last}$$

$$= （T_{start} + T_{loop} + m \times MVL） \times p + （T_{start} + T_{loop} + m \times q ）$$

$$= （p + 1） \times （T_{start} + T_{loop}） + m（MVL \times p + q）$$

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

4.4 向量处理机的性能评价

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
```

4.4 向量处理机的性能评价

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。

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

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。

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：

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

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

4.4 向量处理机的性能评价

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

对n个向量元素进行计算所需的时钟周期数为：

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

$$= \left\lceil \frac{n}{64} \right\rceil \times (15 + 19 + 18 + 12) + 3n = \left\lceil \frac{n}{64} \right\rceil \times 64 + 3n$$

4.4 向量处理机的性能评价

4.4.2 最大性能R_∞和半性能向量长度$n_{1/2}$

- **向量处理机的峰值性能 R_∞**
  - R_∞表示当向量长度为无穷大时，向量处理机的最高性能，也称为峰值性能。

$$R_\infty = \lim_{n\to\infty} \frac{\text{向量指令序列中浮点运 算次数} \times \text{时钟频率}}{\text{向量指令序列执行所需 的时钟周期数}}$$

  - 对于上述例题4.5向量指令序列中的操作而言，只有“MULTFV V2,F0,V1”和“ADDV V4,V2,V3”两条浮点操作向量指令。

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

4.4 向量处理机的性能评价

假设该向量处理机的时钟频率为200MHz，那么：

$$R_\infty = \lim_{n\to\infty} \frac{\text{向量指令序列中浮点运算次数} \times \text{时钟频率}}{\text{向量指令序列执行所需的时钟周期数}}$$

$$= \lim_{n\to\infty} \frac{2\times\ n\ \times 200}{\left\lceil \frac{n}{64} \right\rceil \times 64 + 3\,n}$$

$$= \lim_{n\to\infty} \frac{2\times\ n\ \times 200}{4\,n}$$

$$= 100\ \text{MFLOPS}$$

4.4 向量处理机的性能评价

2. 半性能向量长度$n_{1/2}$

- 半性能向量长度$n_{1/2}$是指向量处理机的性能为其最大性能的一半时所需的向量长度。
- 评价向量流水线的建立时间对性能影响的重要参数。

例4.6　对于例4.5，假设时钟频率为200MHz，求半性能向量长度$n_{1/2}$。

假设该向量处理机的峰值性能R_∞＝100 MFLOPS，所以根据半性能向量长度的定义有：

4.4 向量处理机的性能评价

$$\frac{2\times n_{1/2}\times 200}{\left\lceil \frac{n_{1/2}}{64} \right\rceil \times 64+3n_{1/2}} = 50$$

假设 $n_{1/2} \leqslant 64$，那么有：

$$64+3n_{1/2} = \frac{2\times n_{1/2}\times 200}{50} = 8n_{1/2}$$

$$5n_{1/2} = 64，\ n_{1/2} = 12.8$$

$$n_{1/2} = 13$$

4.4 向量处理机的性能评价

4. 向量长度临界值n_v

- 向量长度临界值n_v是指：对于某一计算任务而言，向量方式的处理速度优于标量串行方式处理速度时所需的最小向量长度。

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

- 对于上述例4.5DAXPY的例子
  - 假设，在标量串行工作方式下实现DAXPY循环的开销为10个时钟周期。那么在标量串行方式下，计算DAXPY循环所需要的时钟周期数为：

$$T_s = (10+12+12+7+6+12) \times n_v = 59n_v$$

4.4 向量处理机的性能评价

- 在向量方式下，计算DAXPY循环所需要的时钟周期数为：

$$T_v = 64 + 3n_v$$

- 根据向量长度临界值的定义，有：

$$T_v = T_s$$

$$64 + 3n_v = 59n_v$$

$$n_v = \left\lceil \frac{64}{56} \right\rceil = 2$$

4.5 向量处理机实例

4.5.1 具有代表性的向量处理机

- 美国和日本生产的一些向量处理机的简要信息

系统型号	推出时间	最大配置，时钟周期， 操作系统/编译系统	特色和要点
Cray 1S	1976年	有10条流水线的单处理机，12.5ns， COS/CF7 2.1	第一台基于ECL的超级计算机
Cray 2S/4-256	1985年	256M字存储器的4台处理机，4.1ns， COS或UNIX/CF77 3.0	16K字的本地存储器，移植了UNIX V
Cray X-MP 416	1983年	16M字存储器的4台处理机，128M字SSD，8.5ns，COS/CF77 5.0	使用共享寄存器组用于IPC
Cray Y-MP 832	1988年	128M字存储器的8台处理机，6ns， CF77 5.0	X-MP的改进型
Cray Y-MP C-90	1991年	每台处理机2条向量流水线，16台处理机，4.2ns，UNICOS/CF77 5.0	最大的Cray机器
CDC Cyber 205	1982年	有4条流水线的单处理机，20ns， 虚拟OS/FTN200	存储器-存储器系统结构

4.5 向量处理机实例

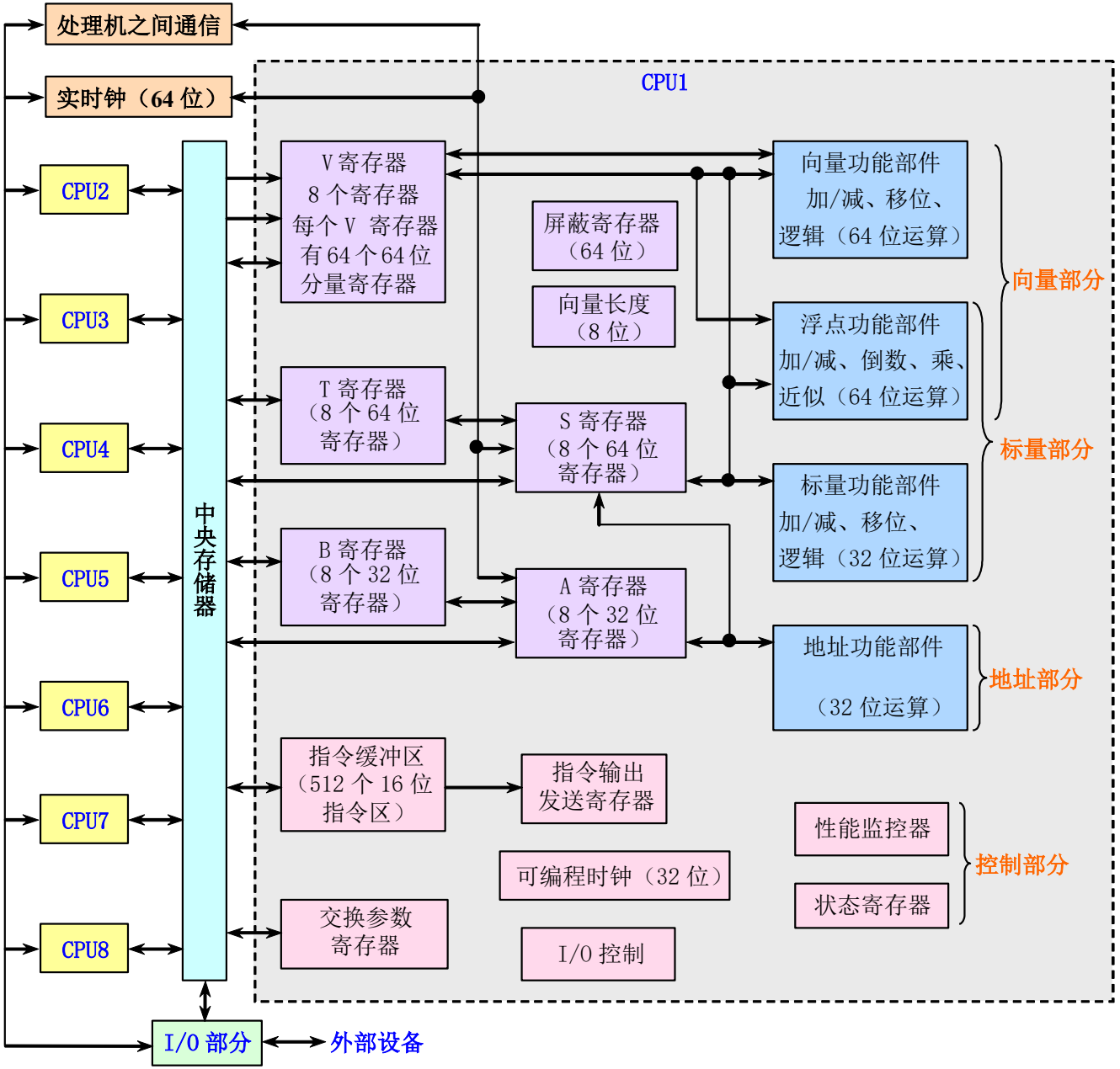
系统型号	推出时间	最大配置，时钟周期， 操作系统/编译系统	特色和要点
ETA 10E	1985年	单处理机，10.5ns，ETAV/FTN 200	Cyber 205的后继型号
NEC SX-X/44	1991年	每台处理机4组流水线，4台处理机，2.9ns，F77SX	
Fujitsu VP2600/10	1991年	5条流水线的单处理机和双标量处理机，3.2ns，MSP.EX/F77 EX/VP	使用可重构微向量寄存器和屏蔽
Hitachi 820/80	1988年	512MB存储器，18个流水线功能部件的单处理机，4ns，FORT 77/HAP V23-OC	64个I/O通道，最大传输率为288MB/秒

4.5 向量处理机实例

4.5.2 Cray Y-MP和C-90

- **Cray Y-MP 816**
  - 1991年问世
  - 系统结构图
    - 可以配置1台、2台、4台或8台处理机
    - 8个CPU共享中央存储器、I/O 子系统、处理机通信子系统和实时时钟
    - CPU的时钟周期：6ns
  - 中央存储器
    - 分成256个交叉访问的存储体

      通过每个CPU对4个存储器端口的交叉访问可以实现对存储器的重叠存取。


处理机之间通信
实时钟（64 位）
CPU1
CPU2
CPU3
CPU4
CPU5
CPU6
CPU7
CPU8
中央存储器
V 寄存器
8 个寄存器
每个 V 寄存器
有 64 个 64 位
分量寄存器
屏蔽寄存器
（64 位）
向量长度
（8 位）
T 寄存器
（8 个 64 位
寄存器）
S 寄存器
（8 个 64 位
寄存器）
B 寄存器
（8 个 32 位
寄存器）
A 寄存器
（8 个 32 位
寄存器）
指令缓冲区
（512 个 16 位
指令区）
指令输出
发送寄存器
交换参数
寄存器
可编程时钟（32 位）
I/O 控制
向量功能部件
加/减、移位、
逻辑（64 位运算）
浮点功能部件
加/减、倒数、乘、
近似（64 位运算）
标量功能部件
加/减、移位、
逻辑（32 位运算）
地址功能部件
（32 位运算）
性能监控器
状态寄存器
向量部分
标量部分
地址部分
控制部分
I/O 部分
外部设备

4.5 向量处理机实例

- 容量最大可达1GB。固态存储器的容量最大可达4GB。
- 4个存储器访问端口允许每个CPU同时执行两个标量和向量取操作、一个存储操作和一个独立的I/O操作。

  这些并行的存储器访问也采用流水线方式，使得向量读和向量写可以同时进行。

- CPU的计算系统由14个功能部件组成，分为向量、标量、地址和控制4个子系统。
  - 向量和标量指令可以并行地执行
  - 所有算术运算都是“寄存器-寄存器”类型
  - 向量指令可以使用14个功能部件中的8个

4.5 向量处理机实例

- 系统使用了大量地址寄存器、标量寄存器、向量寄存器、中间寄存器和临时寄存器。
  - 通过对寄存器及多体存储器和算术/逻辑流水线的使用，可以实现功能流水线灵活的链接。
- 浮点和整数算术运算都是64位。
- 大型指令高速缓存可同时存放512条16位的指令。
- 主机中的处理机之间的通信系统包括用于快速同步目的的共享寄存器群。
  - 每个群由共享地址寄存器、共享标量寄存器和信号灯寄存器组成。
  - CPU之间向量数据通信是通过共享存储器实现的。
- I/O子系统支持3类通道，传输速率分别：6MB/s，100MB/s和1GB/s。

4.5 向量处理机实例

2. C-90

- 由16个类似于Y-MP的CPU组成
- 16台处理机共享主存储器的容量高达256M字（2GB）
- SSD存储器的容量最多达16GB
  - 可选作第二级主存储器
- 两条向量流水线和两个功能部件可以并行操作，每个时钟周期能产生4个向量计算结果。

  意味着每台处理机有4路并行性，因此16台处理机每个时钟周期最多可以产生64个向量计算结果。

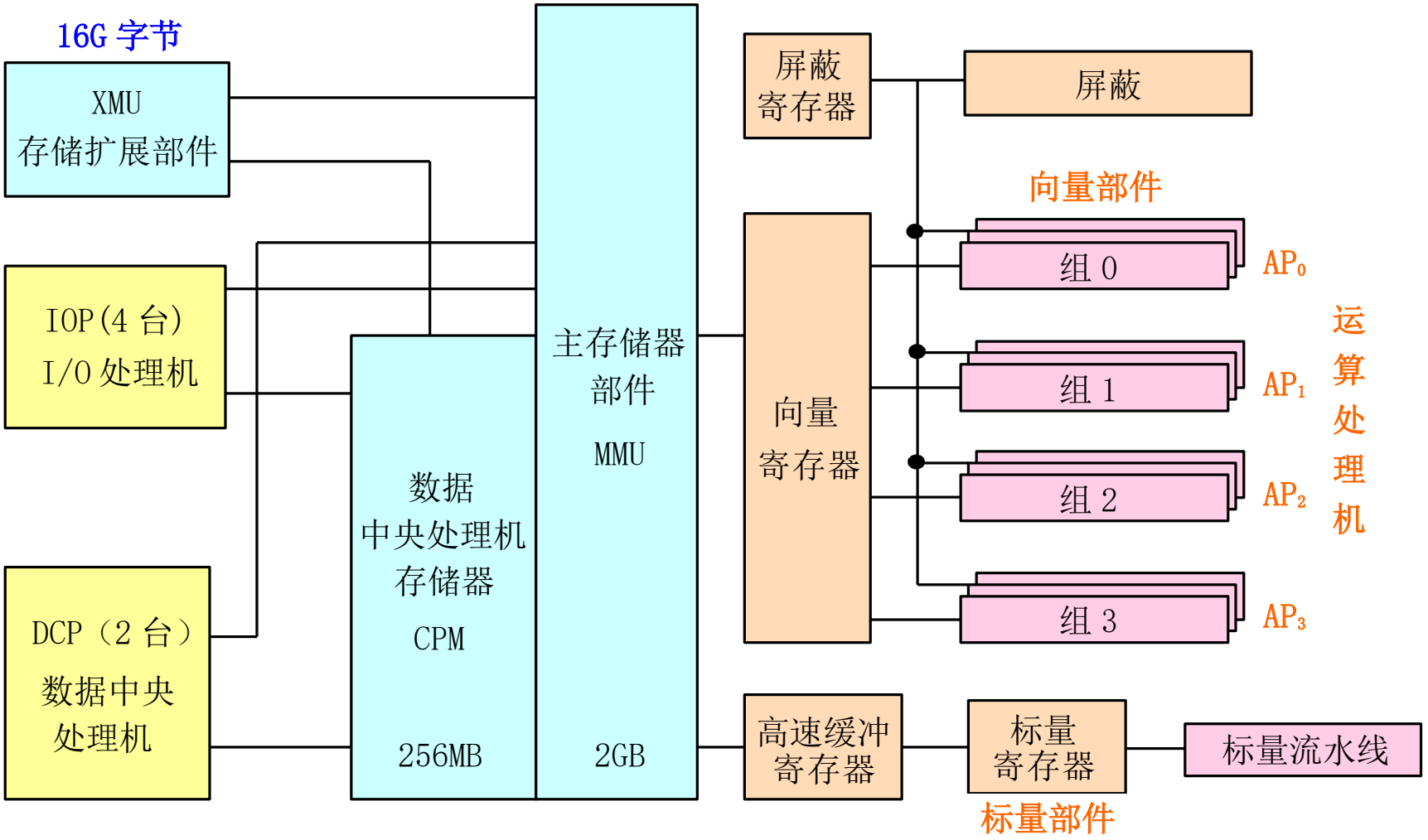
4.5 向量处理机实例

- 运行UNICON操作系统，提供向量化的FORTRAN 77和C编译器。
- 64路并行性和4.2ns时钟周期相配合，可使系统的峰值性能达到16GFLOPS，系统最大1/O吞吐率为13.6MB/s。

4.5 向量处理机实例

4.5.3 NEC SX-X44

- **NEC 1991年推出峰值速度可达到22GFLOPS**
  - 主要措施之一：使用了基于VLSI和高密度封装技术的2.9ns的时钟
- **系统结构图**
- **4台运算处理机通过共享寄存器或通过2GB的共享存储器进行通信。**
  - 每台处理机有4组向量流水线
    - 每组包括2条加法/移位流水线和2条乘法/逻辑流水线
  - 类似于C-90，4台处理机可达到64路并行


16G 字节
XMU
存储扩展部件
IOP(4 台)
I/O 处理机
DCP（2 台）
数据中央
处理机
数据
中央处理机
存储器
CPM
256MB
主存储器
部件
MMU
2GB
屏蔽
寄存器
屏蔽
向量部件
向量
寄存器
组 0
组 1
组 2
组 3
AP0
AP1
AP2
AP3
运算处理机
高速缓冲
寄存器
标量
寄存器
标量流水线
标量部件

4.5 向量处理机实例

4. **高速标量部件**
   - 采用了具有128个标量寄存器的RISC系统结构，通过把指令重新排序来开发较高的并行性。
5. **主存储器为1024路的交叉访问存储器**
   - 其扩展存储器的最大容量高达16GB
   - 最大传输率：2.75GB/s
6. **系统最多可以配置4台I/O处理机**
   - 每台I/O处理机的数据传输率：1GB/s
   - 最多可以提供256个通道，用于高速网络、图形和外围操作，支持100MB/s的通道传输。

4.6 小结与习题

- 性能分析：
  - 向量流水处理机的链接技术
  - 向量指令处理时间计算
- 作业：
  - 4.2，4.3，4.4，
  - 4.5，4.6