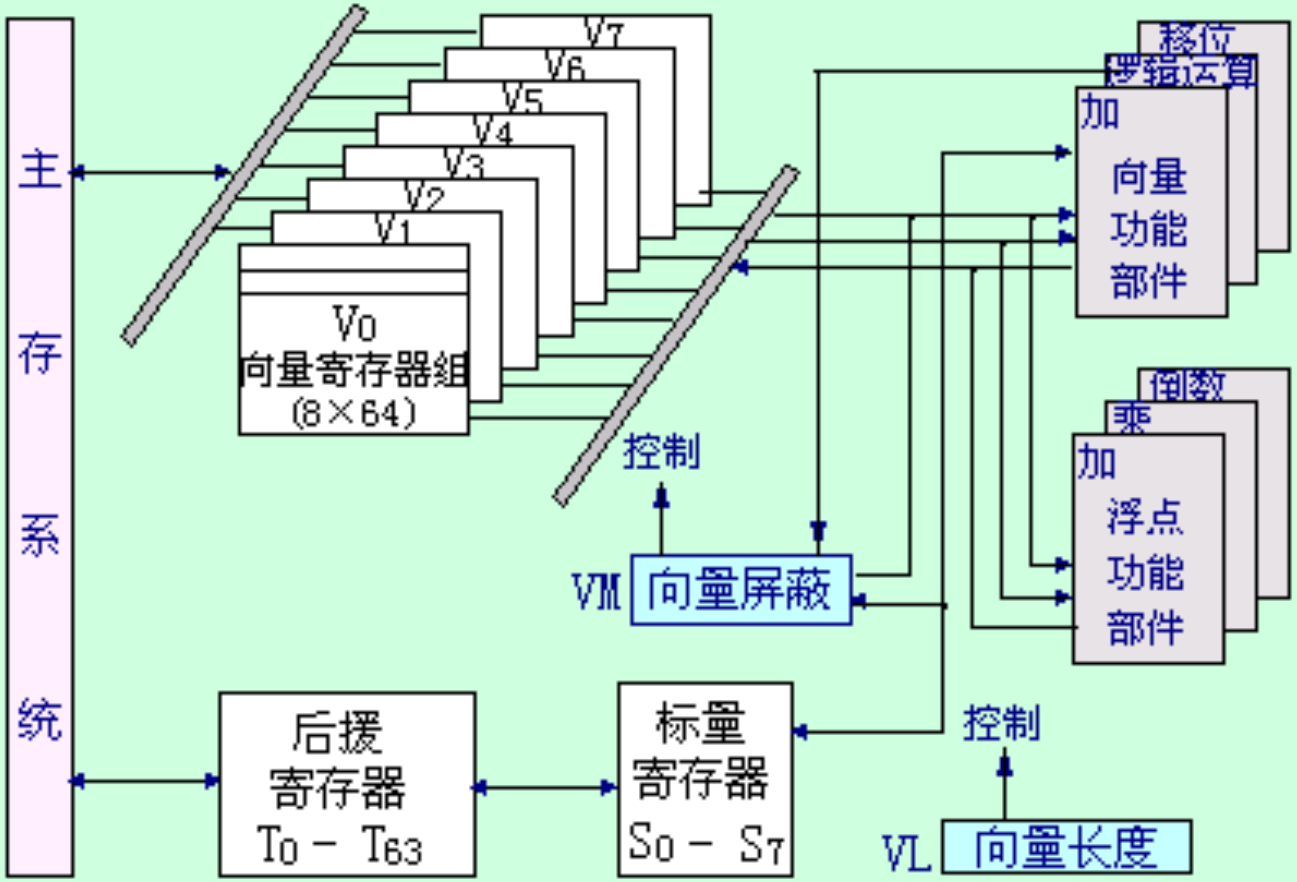
CRAY-1的基本结构


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

- 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12.5 ns。

- 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～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～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。

- 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。

- 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

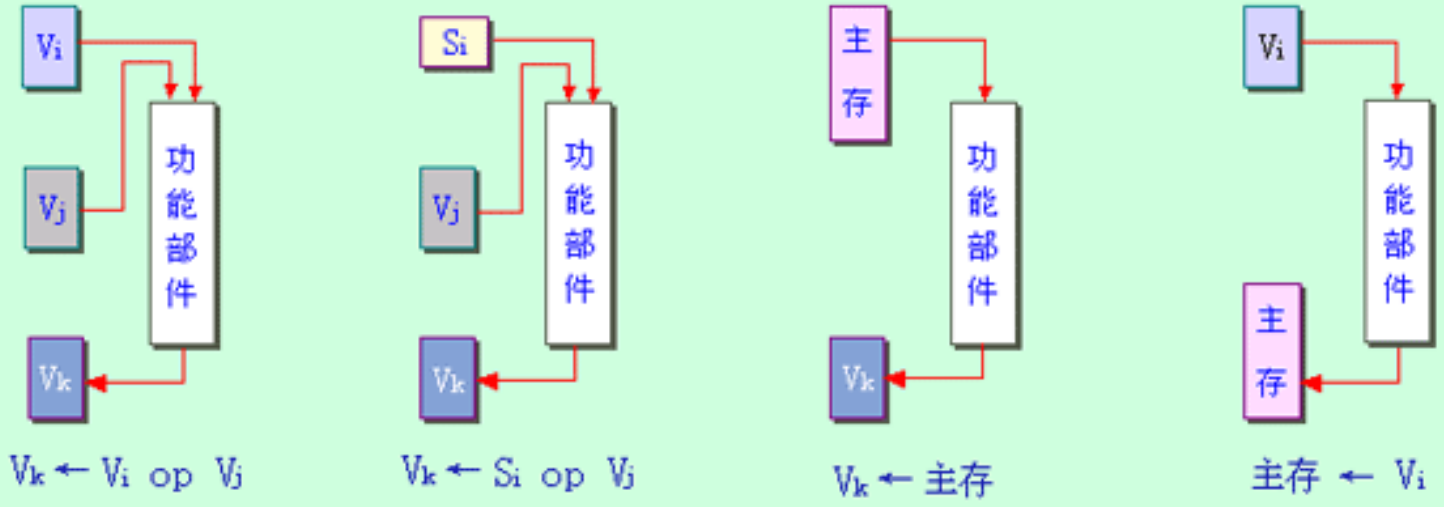
  $$V_3 \leftarrow V_1 \times V_2$$

  $$V_5 \leftarrow V_4 \times V_6$$

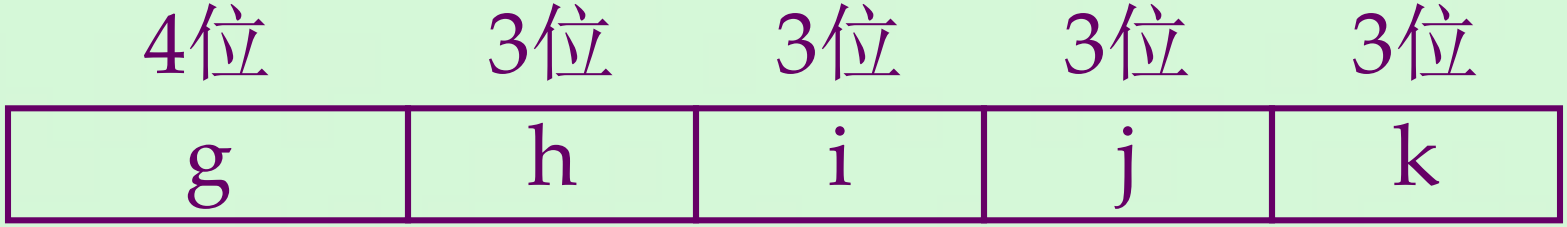
以CRAY-1向量处理机为例，有四类指令，两种指令格式

四类指令
- (1) 向量与向量操作，　Vi←Vj OP Vk
- (2) 向量与标量操作，　Vi ← Sj OP Vk
- (3) 向量取，　Vi←存储器
- (4) 向量存，　存储器← Vi

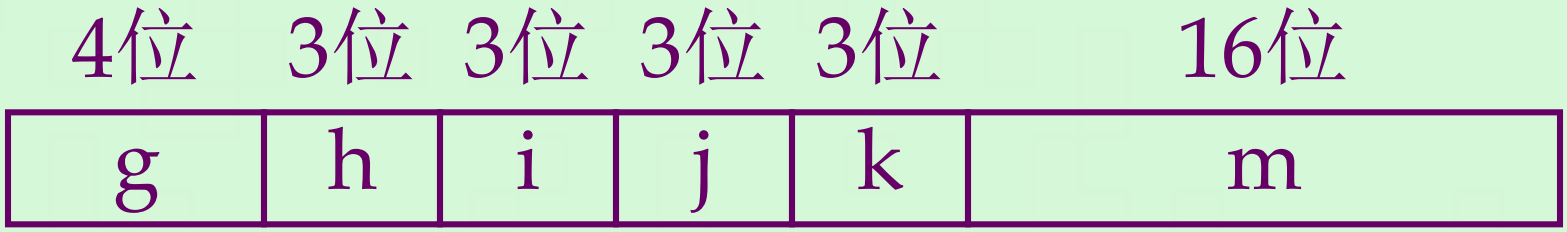
CRAY-1的向量指令类型



CRAY向量处理机的指令格式



g, h为操作码，i为目的寄存器编号
j, k为源寄存器编号



g为操作码，h为变址寄存器A的编号
i为目的寄存器编号，j, k, m为形式地址

向量运算中的相关和冲突

- 向量运算中的数据相关和功能部件冲突：
  采用顺序发射顺序完成方式
  (1) 写读数据相关。
  (2) 读读数据相关，或向量寄存器冲突。
  (3) 运算部件冲突。

- **V0 ← V1＋V2**
  V3 ← V4 × V5
  (a)不相关的指令

  V0 ← V1＋V2
  V3 ← V0 × V4
  (b)写读数据相关

- V0 ← V1＋V2

  ↓

  V3 ← V4＋V5

  (c)功能部件冲突

  V0 ← V1＋V2

  ↓

  V3 ← V1 × V4

  (d)读读数据相关

3.5.3 提高向量处理机性能的方法

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。

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

2. 链接技术

- 链接特征：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。
- 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。

例3.3 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

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

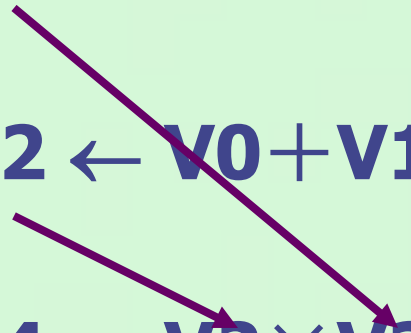
$V_4 \leftarrow V_2 \times V_3$　　// 浮点乘，结果存入V_4

- 3条向量指令：

  V3 ← A

  V2 ← V0＋V1

  V4 ← V2×V3



第一、二条指令没有数据相关和功能部件冲突，可以同时开始执行。

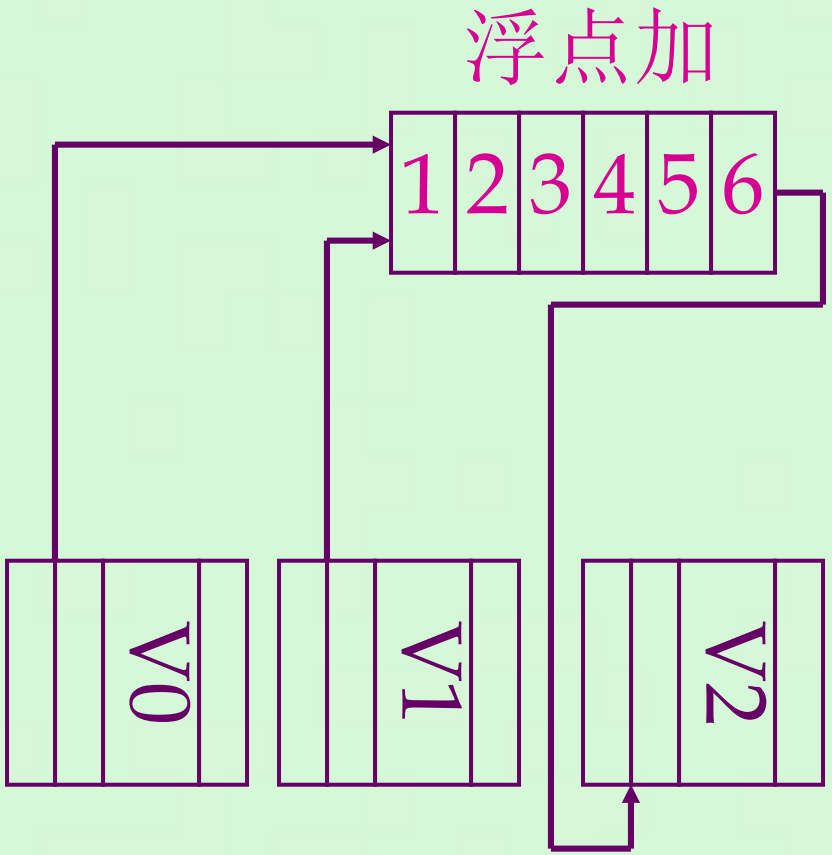
第三条指令与第一、二条指令均存在写读数据相关，可以链接执行。

向量指令：

V3 ← A

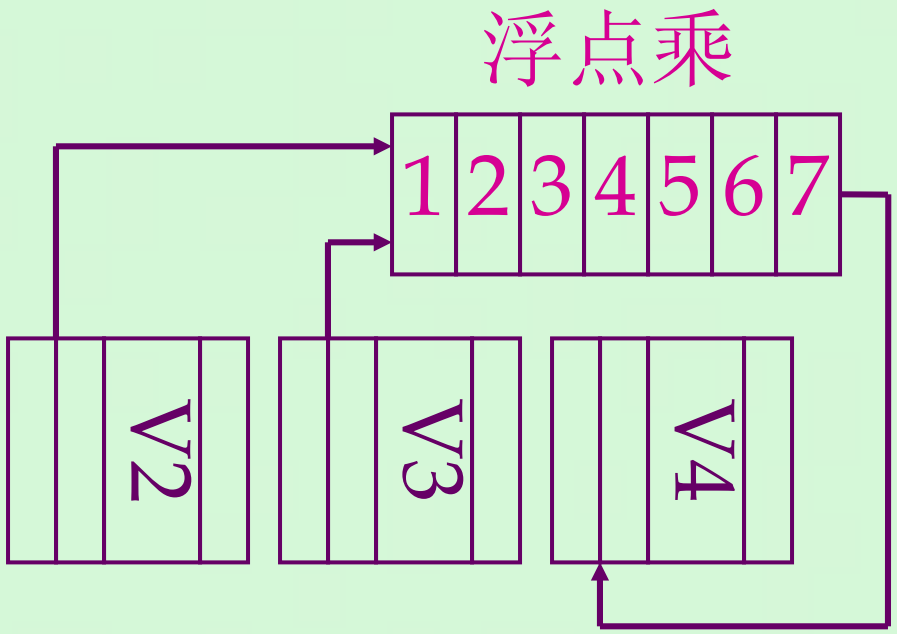
向量指令：

V2 ← V0+V1



向量指令：

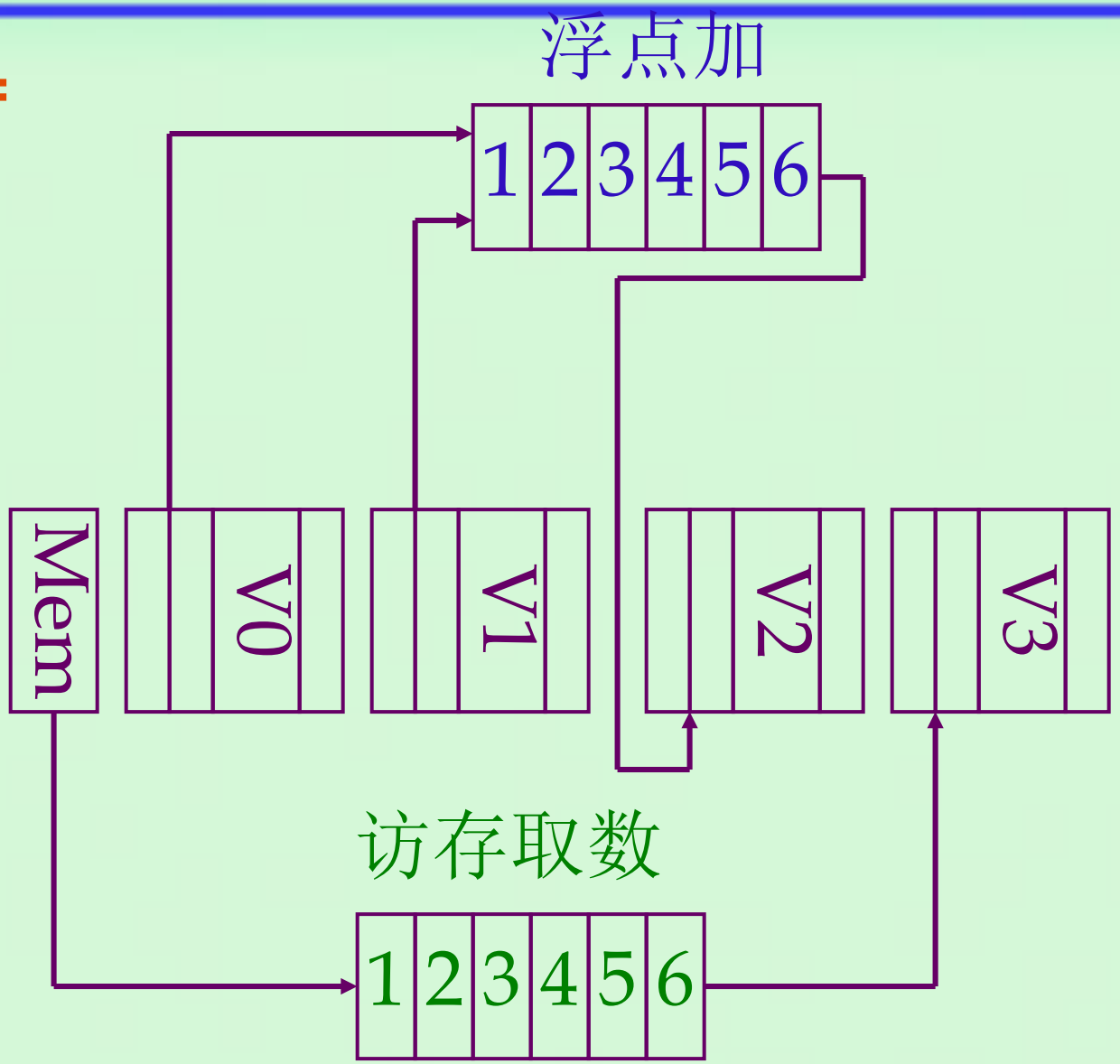
V4 ← V2×V3



向量指令的并行处理：

V3 ← A

V2 ← V0+V1



- 向量链接技术(chaining)
  - 结果寄存器可能成为后继指令的操作数寄存器

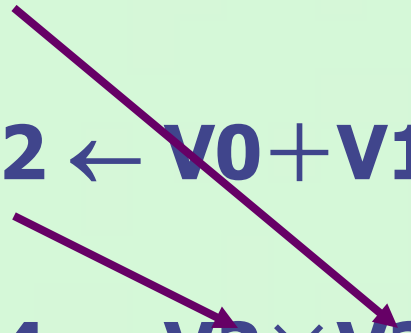
    两条有数据相关的向量指令并行执行，这种技术称为两条流水线的链接技术。

- 3条向量指令：

  V3 ← A

  V2 ← V0＋V1

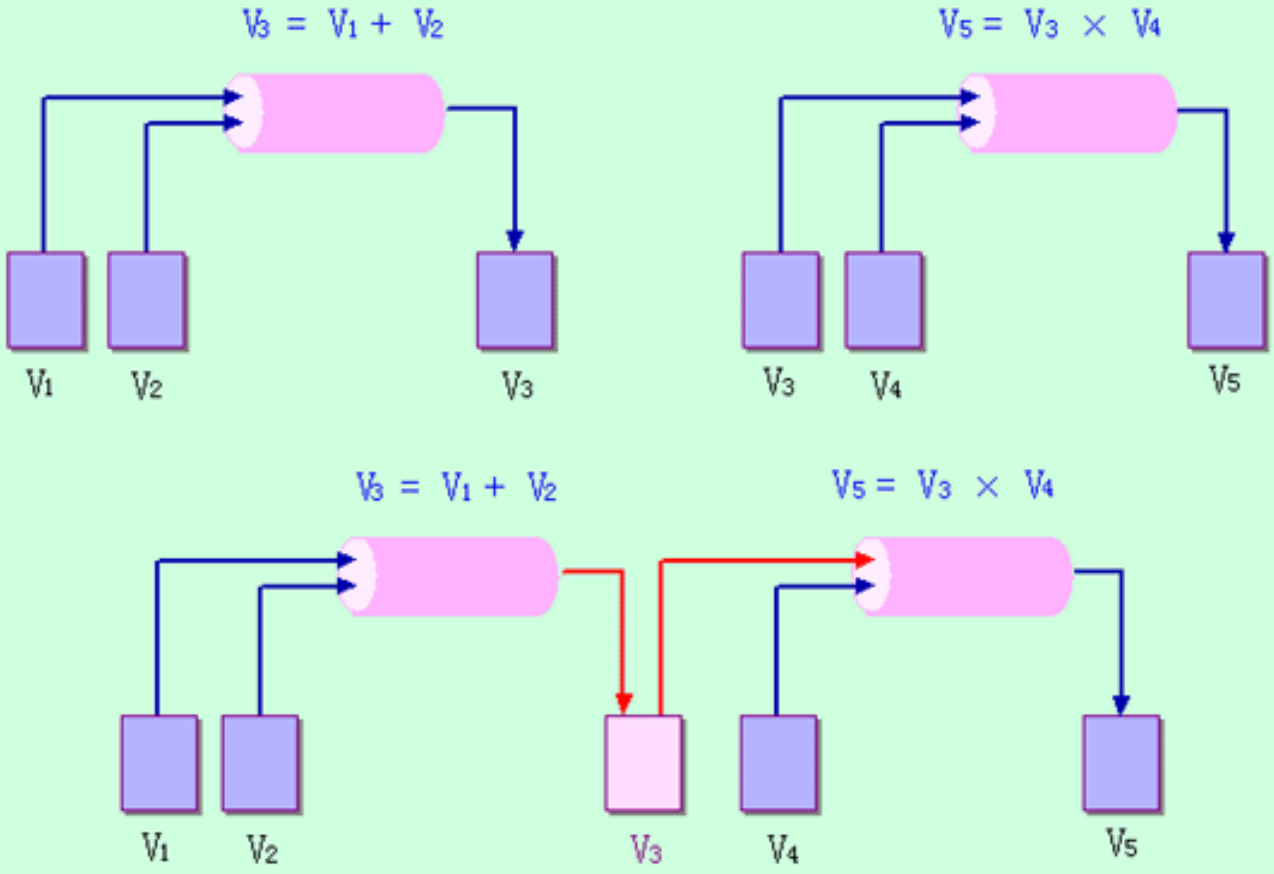
  V4 ← V2×V3



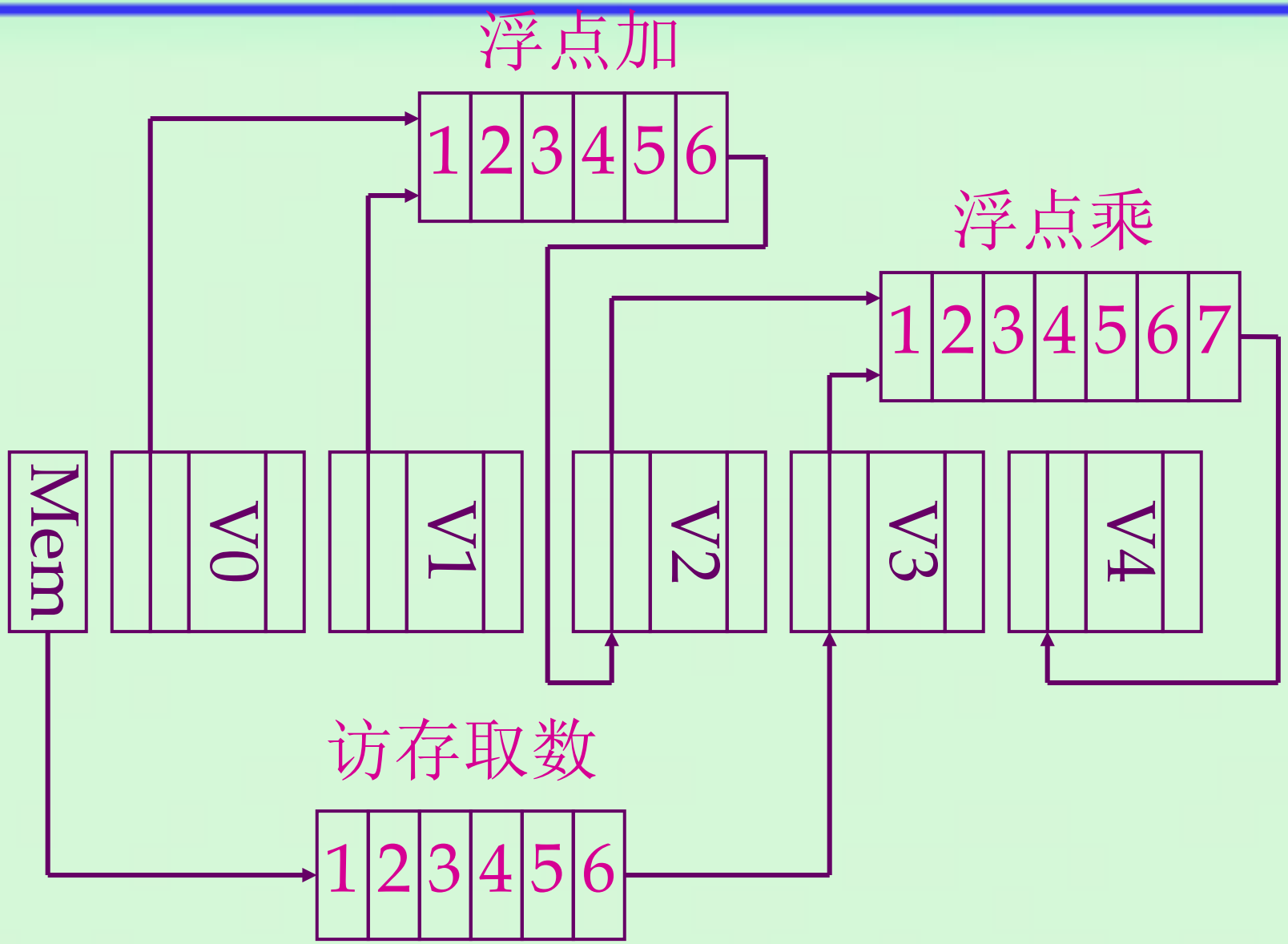
第一、二条指令没有数据相关和功能部件冲突，可以同时开始执行。

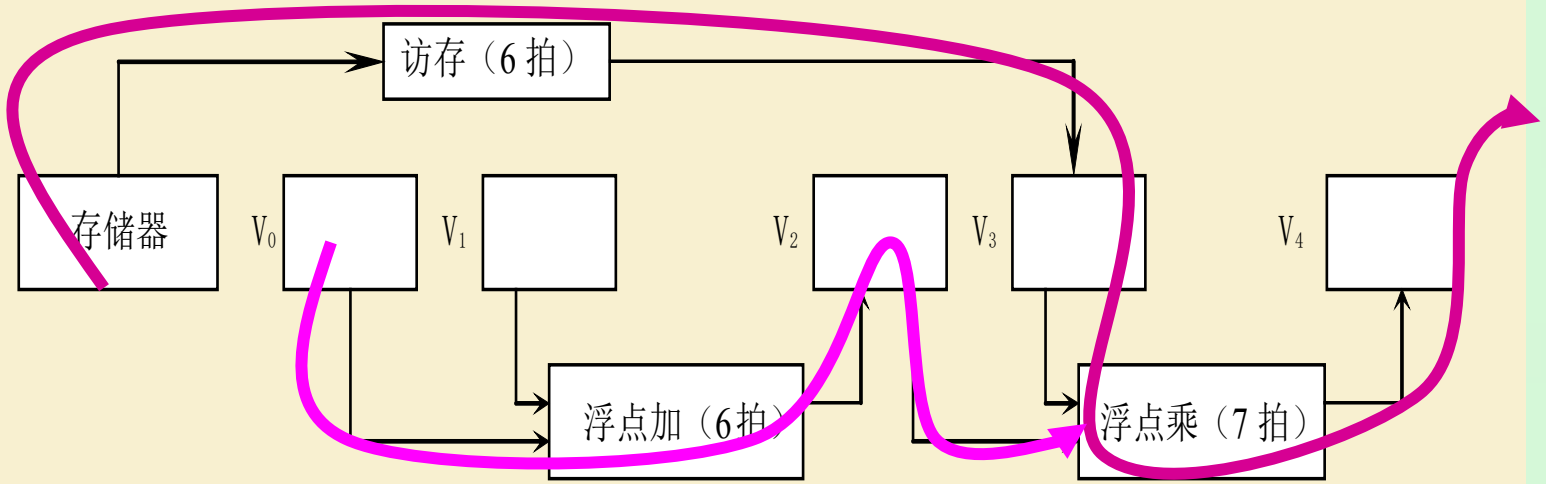
第三条指令与第一、二条指令均存在写读数据相关，可以链接执行。

链接特征



虚拟存储器的特点


浮点加
1 2 3 4 5 6
浮点乘
1 2 3 4 5 6 7
Mem
V0
V1
V2
V3
V4
访存取数
1 2 3 4 5 6



- 假设：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需要一拍时间，从存储器中把数据送入访存功能部件需要一拍时间。

- 3条指令全部用串行方法执行，则执行时间为：

  $[(1+6+1)+N-1]+[(1+6+1)+N-1]$

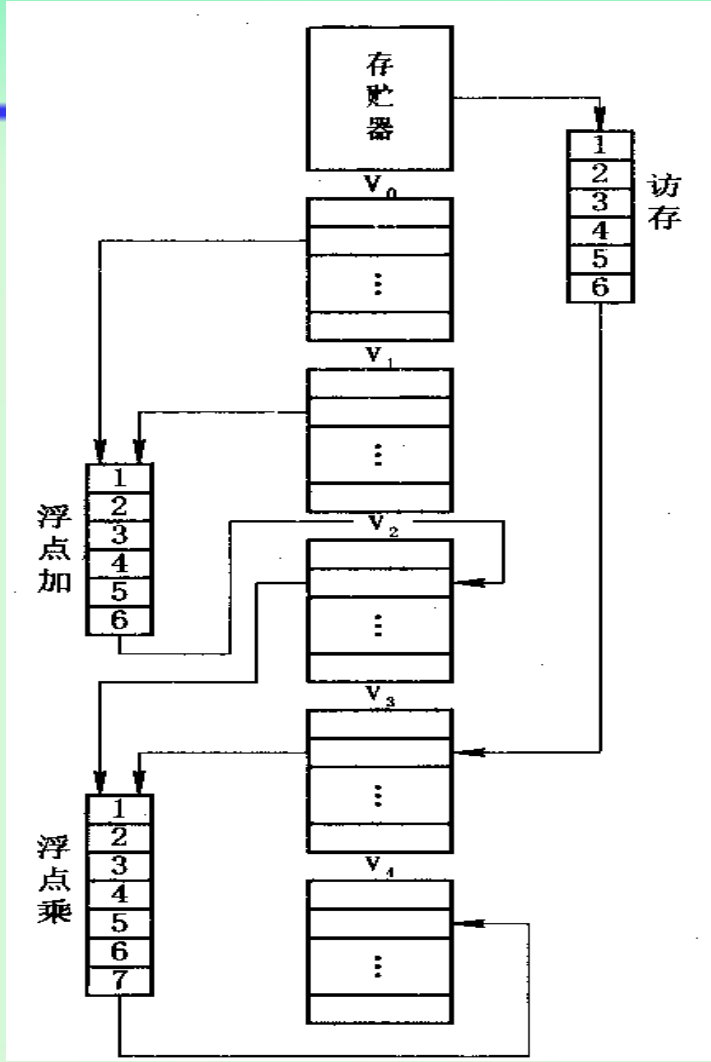
  $+[(1+7+1)+N-1]=3N+22$ （拍）

- 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。



通过链接技术实现向量指令之间大部分时间并行

- 从访存开始到把第一个结果元素存入V_4所需的拍数（亦称为链接流水线的建立时间）为：

  [（1＋6＋1）] ＋[（1＋7＋1）] = 17 （拍）

$$1\left\{\begin{matrix}\text{启动访存}\\ \text{送浮加部件}\end{matrix}\right\}+6\left\{\begin{matrix}\text{访存}\\ \text{浮加}\end{matrix}\right\}+1\left\{\begin{matrix}\text{存}V3\\ \text{存}V2\end{matrix}\right\}+1\left\{\begin{matrix}\text{送浮乘部件}\\ \text{送浮乘部件}\end{matrix}\right\}$$

$$+7\{\text{浮乘}\}+1\{\text{存}V4\}=17\text{拍}$$

- 3条指令的执行时间为：

  [（1＋6＋1）]＋ [（1＋7＋1）] ＋（*N*—1）

  = *N*＋16 （拍）