一种新的体系结构——数据流计算机*

中国科学技术大学　李 国 杰

一、引　言

电子计算机从诞生到现在经历了几代变革，但冯·诺依曼计算机一直占统治地位。它是基于控制流概念设计的，以指令指针（即程序计数器）的移动来激发指令执行。数据通过访问指令指定的存储单元而取得。众所周知，冯·诺依曼型计算机处理并行计算问题存在着本质上的困难。近年来各种新型的计算机体系，如阵列机、流水线机、多处理机等都采用了并行技术。但这些体系结构仍旧没有摆脱传统的以控制流为主的设计思想。由于指令相关、地址空间相关等的矛盾，并行性受到限制。这些计算机的并行是局部的。并行理论的研究导致了计算机体系结构又一次重大革新，出现了以数据流概念为主的设计思想，即数据流计算机。它由操作数的到达来驱动指令执行，取消了程序计数器，摆脱了传统的同步控制的束缚，采用异步控制，从而以一种很自然的方式表示算法中的并行性，提供了全局并行的可能性，其计算速度可以比通常的计算机速度高1到2个数量级。由于数据流计算机具有上述明显的优点，它引起了计算机界较浓厚的兴趣。近年来美、英、法等国都在从事这方面的研制，在80年10月召开的IFIP第八届世界计算机大会上，数据流计算机也受到重视。目前研制的数据流计算机大多是专用机，但长远的目标是数据流通用机。数据流计算机是异步并行系统，可以用Petri网作理论模型。一般是先研究数据流语言，然后在这种语言基础上考虑硬件实现。

二、数据流语言

所谓数据流语言是一种用图表示的机器语言。数据流计算机是面向这种语言的计算机，它的硬件是数据流语言的解释器。数据流语言通过编码成为机器指令存于计算机中。

数据流语言表示为有向偶图。它有两种不同的节点，分别称为链(Iink)和动作(actor)。由偶图性质可知，两个动作之间必有一链，两个链之间必有一动作，如图1所示。其中圆圈表示动作，小圆点表示链。此图表示快速傅里叶变换中蝶式运算。粗略地看，每一动作表示进行一步运算，其结果由链传送到下一动作。根据Petri网的规定，信息由记号(token)运载，图1中用大黑圆点表示。在数据流语言中，指令的执行顺序由所谓点火规则(firing

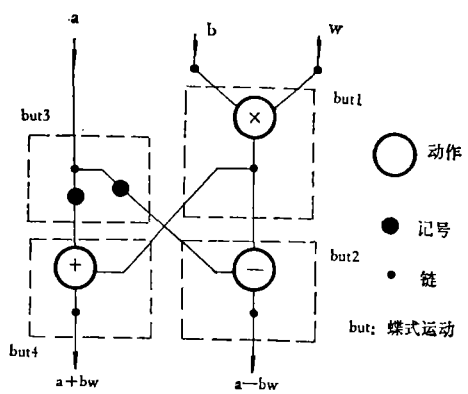

图 1

rule)规定。下面以J·B·Dennis 1973年提出的第一个数据流语言方案[1]为例加以说明：(1) 一个节点(动作或链)的每一输入弧上都有记号，而任何输出弧上都没有记号，则称此节点就绪。(2) 任何就绪的节点都可以点火。(3) 链的点火是从输入弧上移去记号放到它的每一条输出弧上。(4) 一个动作点火是从它的每一输入弧上移去记号，用输入记号传送来的

---

* 1980年7月收到

值确定一个结果作输出记号，放在它的输出弧上。(注意链只有一条输入弧，而动作只有一条输出弧。)链和动作的点火规则如图 2 所示。

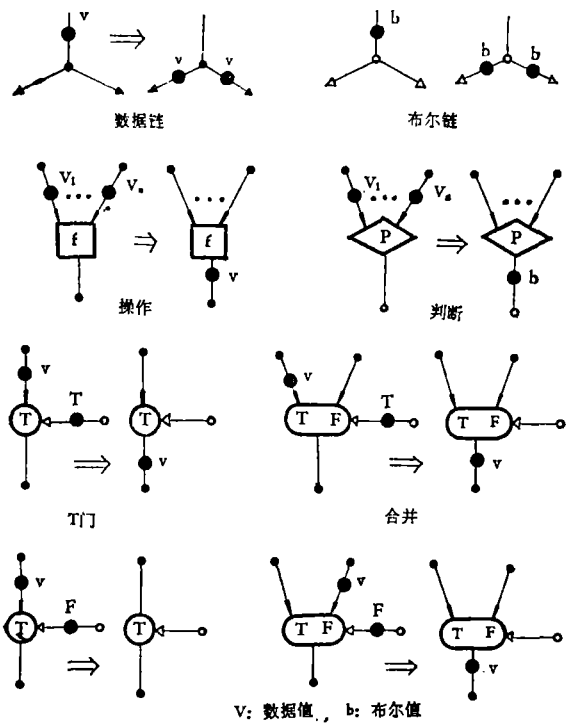

图 2

Dennis 方案中有两种链：数据链和布尔链。前者在数据弧上传送数据，记号的值可以是整数、实数、复数或字符串等。后者在控制弧上传送布尔值，记号的值只有真 (T) 和假 (F) 两种。动作分几种类型：操作 f 进行各种运算，输出数据值；判断 P 输出布尔值；控制型动作(门和合并)的点火规则与其他动作有区别。对于 T 门，如果控制记号为真，则让输入记号通过；如果控制记号为假，则吸收输入记号，不放到输出弧上，即阻止记号通过。F 门则反之，控制记号为假时让输入记号通过。合并 (merge) 动作将 T 门与 F 门合在一起，点火规则与单独的 T、F 门一样。控制型动作在条件语句和循环语句中起开关作用。下面将一段类似 ALGOL 语言写的源程序用数据流语言表示出来。源程序如下：

```
input(W, X)
  y:=X; t:=0;
  While t≠W do
  begin
      if  y>1  then  y:=y÷2
               else  y:=y×3;
      t:=t+1;
  end
output  y
```

对应的数据流语言如图 3 所示。图 3 中 x，w 是输入链，虚线小框内的部分表示条件语句 if-then-else，其余右边部分表示循环迭代语句 While-do。其中三个合并动作的控制弧用 ⟵ 表示其初始状态控制记号为假值，即允许 x，w 及 t 的初值 0 进入循环程序。除了这三个控制记号外，初始时其他弧上没有记号。

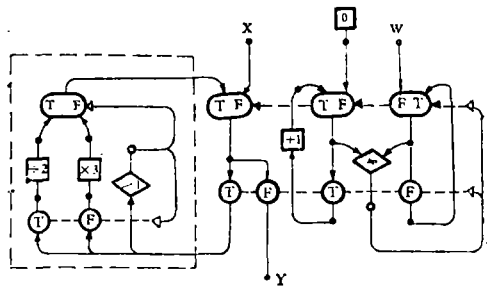

图 3

人们自然会提出一个问题：用图表示的数据流语言如何被计算机接受，成为硬件可执行的指令？解决的办法是将数据流程序图中一个或几个动作及其输出链直接编码成一条机器指令。(注意！不是输入链。)如图 1 的小框 *but* 1 可编码为一指令，占用一个指令单元，也有人称为模板(template)，如图 4 所示。一个指令单元包括以下几部分：(1) 操作码 (C-mul 表

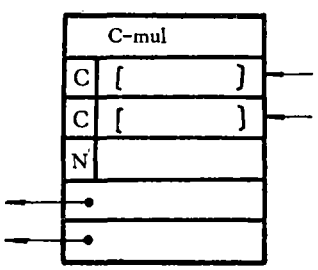

图 4

示复数乘法)；(2) 操作数。放在接收器中，此处为三个，其中两个放乘数与被乘数，它们等待数据链把要计算的值送来，第三个N表示空。(3) 目标地址。它指出运算结果送到那个单元。小黑点表示输出的数据记号。目标信息是指令必不可少的组成部分。

值得注意的是数据流计算机的指令中，没有通常的为存取操作数所需的地址码。操作数不是指令"取"来的，而是前一步已进行的操作"送"来的。常数则事先存在指令单元中。这是数据流计算机与一般控制流计算机的主要区别。正因为操作数是自动送来的，数据的相关性就由程序直接表示出来了。操作数未全部到达，指令就不会操作，一旦操作数到齐，指令就可以执行。通过目标地址把整个程序连接成一个整体。图 5 画出了图 1 所示的程序的指令单元及其连接方式。图中 * 表示回答信号。数据流计算机的操作与流水线类似。根据点火规则，一个操作要能进行，除了操作数到齐之外，还要求输出链上没有记号，即上一次运算的结果已被取走。图 1 中加、减操作能否进行要看它的下一步操作是否完成，如果完成了，发一回答信号(*)，所以要回答信号与操作数都到齐才算真正就绪。图 5 中指令操作码后的椭圆圈内的一对数字，分子表示需要的回答信号个数，分母表示预先假定已收到的回答信号

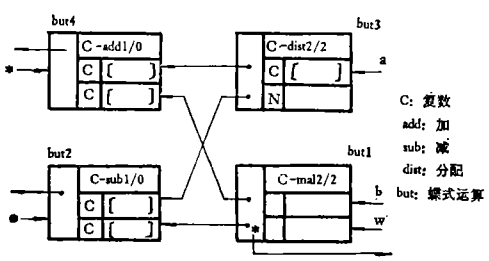

图 5

个数。如图 5 乘法结果送加、减操作，因乘法慢，加法快，不必回答。但乘法的输入数可能产生很快，所以要回答。满足上述条件的数据流语言是安全的，它保证计算机不会发生死锁。

在带有条件语句和循环迭代语句的数据流程序图中，其机器指令编码与上述情况类似。但判断动作输出的是布尔值。合并动作的输入值一个是数据值，另一个是布尔值，输出是数据值。图 3 的虚线框内的条件语句所对应的指令单元及其连接如图 6。

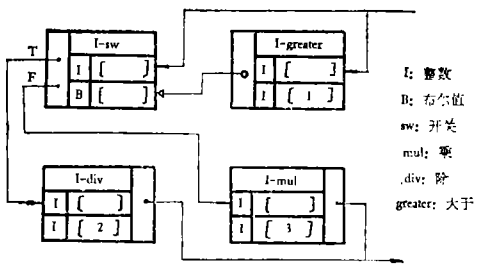

图 6 注：图中省略了回答信号

在上述的数据流程序中，记号传送的值限制在整数、实数、复数或字符串范围内，称为基本的数据流程序。这种程序等价于 While-do 和 if-then-else 写的表达式功能。以后又提出了扩展的数据流语言[1),3)]，它可以用 Apply 操作调用一个数据流过程，并具有较复杂的数据结构。限于篇幅，本文对扩展的数据流语言不作详细讨论。

在数据流语言中，没有 go to 语句，没有全程变量及其他有副作用的功能。每一操作只局限于它的输入输出链，不影响其他的操作。每个操作都是独立并行的，有多少操作就绪原则上就可以有多少操作并行。执行数据流语言表示的程序时，控制流和数据流都由数据流程序图的点火规则确定，同时有许多条控制的路径，充分发挥了并行效率。

数据流计算机的用户用高级语言写源程序，这种高级语言用一般的句文表示，但要求它易于检查程序的数据相关性，并容易编译成数据流语言。目前正在研究各种有效的源语言，如美国麻省理工学院研制的 VAL 语言和法国 Plas 等人研究的单赋值语言[2)]等。在单赋值语言中，各种语句都看成赋值语句，每一变量只赋值一次。也就是说存储单元写进一个数，在程序执行过程中不再改变，执行这种语

言写的程序可以不考虑指令的先后次序，几乎能直接转换成数据流语言。法国研制的 LAU 系统就是建立在单赋值语言的基础上[2]。

三、数据流计算机的体系结构和实现方法

数据流计算机由五个主要部件组成，如图 7 所示。这五大部件是：(1) 存储部件；(2) 处理部件；(3) 仲裁网络；(4) 分配网络；(5) 控制网络。

当存储器中的指令单元从分配网和控制网收到全部需要的数据包和控制包时，指令就绪。就绪的指令与它的操作数一起形成一个操作包，经过仲裁网识别操作码，按先进先出次序送到相应的处理单元，执行完一个操作所得

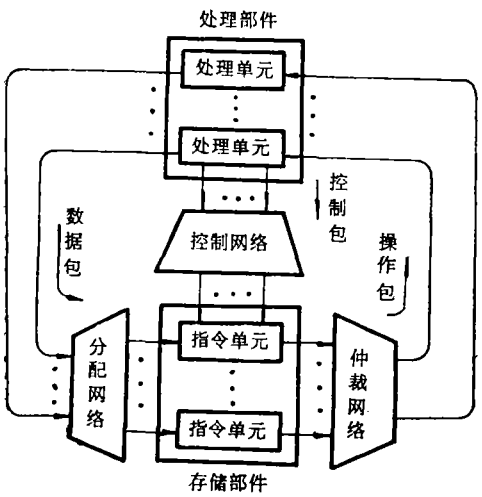

图 7

的结果又通过分配网和控制网送到存储器作为它的目标指令单元的操作数。这就是执行一条数据流指令的大致循环过程。

在数据流计算机中没有通常的中央处理器(CPU)，而把 CPU 的功能分散在各个部件中实现。操作码规定的操作在处理部件中进行；但指令顺序的控制由存储器的指令单元实现；操作码的译码由仲裁网实现。各部件之间的通讯以固定尺寸的离散的信息包形式进行。信息流方向如图 7 所示，它构成一个环形网络。信息单方向流动。各部件之间的包通讯完全是异步的。五个部件可以独立操作，不需要统一的时钟信号。因此数据流计算机有时也称包通讯计算机。

下面以美国麻省理工学院 Dennis 领导的计算机体系结构组 1979 年提出的修改方案为例[4],[5],[6]，进一步分析各个部件的功能和结构。

1) 存储部件

存储部件(即指令单元存储器)是数据流计算机的核心。它不只是存储指令和数据，而且决定哪些指令可以执行，用数据到达来驱动指令执行，取代了通常的程序计数器。怎样判别一条指令的操作数是否到齐呢？可为每一指令设一就绪计数器，开始时置应输入的记号总数，每到一个记号，计数器减 1，计数器为 0 表示所要求的数据、控制记号及回答信号已到齐。也可以在指令单元的每一接收器中设一标志位，操作数到达就改变标志位，所有的标志位都改过了，说明输入记号已到齐。

上一节已提到一个指令单元包括指令（由操作码和目标地址组成）和接收器。麻省理工学院设计的数据流计算机（以下简称 MIT 机）每一指令单元规定有三个接收器，允许一条指令最多有五个目标地址。每一指令单元有唯一的单元标识，每个接收器有接收器标识。一个目标地址包括单元标识和接收器标识两部分，即两个整数。一个指令单元的物理结构由输入接口模块，三个接收器及输出接口模块组成。

数据流语言的程序在装入时，每个指令单元的接收器根据指令要求设置相应的接收器类型和接收器模式。接收器类型分为布尔型、整数、复数三种；模式分成变量和常数两种。运行时根据接收器类型和模式接收不同的信息。记号传送来的信息必须与对应的类型和模式相符。这样可以达到校验的目的。

数据流计算机中采用的存储单元不是一般的磁心或 MOS 存储器，而是一种称为位流水线(bit-pipeline)的与速度无关电路[7]。这种电路信息的流入和流出可同时进行（相当于读写同时进行），它的存储速度与其他逻辑元件工作

速度一样快。数据流计算机是异步系统，各种操作的结果不受元件及连接线传输延迟的影响。与速度无关电路是 Pefri 网的重要应用，它是数据流计算机的电路基础。

数据流计算机的每一指令占用一个指令单元(不止一个字)，比传统的计算机占用的存储空间多得多。为了节省存储空间，可采取存储器分层的办法，由位流水线组成的指令单元只做高速缓存，存放最活跃的指令。其余大部分指令存于成本较低的大容量存储器。为了减少指令单元的入口和出口，从而简化仲裁网和分配网，可将指令单元分块，每块只有一对输入输出口，同时把仲裁网分成几个子网，每个子网对应一组处理单元。

2) 处理部件

处理部件包括各种处理单元(即功能单元)，如算术逻辑运算单元，判断及扇出控制单元等。进入处理部件的操作包中有目标信息，用来形成数据包和控制包的目标地址。处理部件也采取流水线方式工作，以获得最高处理效率。为提高整个系统的性能，当然多采用一些功能单元有好处，但功能单元的多少必须与指令单元的容量匹配。而且功能单元越多，路径选择网络就越复杂。下面以整数操作单元为例简单说明功能单元的工作情况。整数操作单元具有加、减、位测试和比较等功能(为减少仲裁网的出口数，操作单元的功能应强一些，不要分得太细)，它分成为整数控制和整数运算两个子模块。整数控制子模块对接收到的操作包进行判断，看需要何种操作，然后发送命令包到整数运算子模块，运算后把结果包送回整数控制子模块形成数据包和控制包。处理部件中各个处理单元都是独立并行的。慢的操作不会影响快的操作。处理单元处理完一个操作包，马上由仲裁网供给下一个操作包。

3) 路径选择网络

数据流计算机是一个包通讯系统。除了存储部件和处理部件外，其余三部份都是通讯网络。这些网络的基本元件是仲裁单元、开关单元和缓冲器。仲裁网入口多，出口少，主要由仲裁单元构成；分配网入口少，出口多，主要由开关单元构成。这两个网的结构如图 8。

仲裁网分为三级。第一、二级的仲裁单元把来自许多指令单元的操作包汇集到较少数量的通道中；第二级的开关单元按指令所要求的处理能力把操作包流分裂为许多分枝；最后由第三级合并，使每一处理单元对应一个仲裁网的出口，只接收一个操作包流。当有几个相同类型的操作包要输出时，调度的原则是依次轮流，即先进先出。前一个操作包未处理完，后

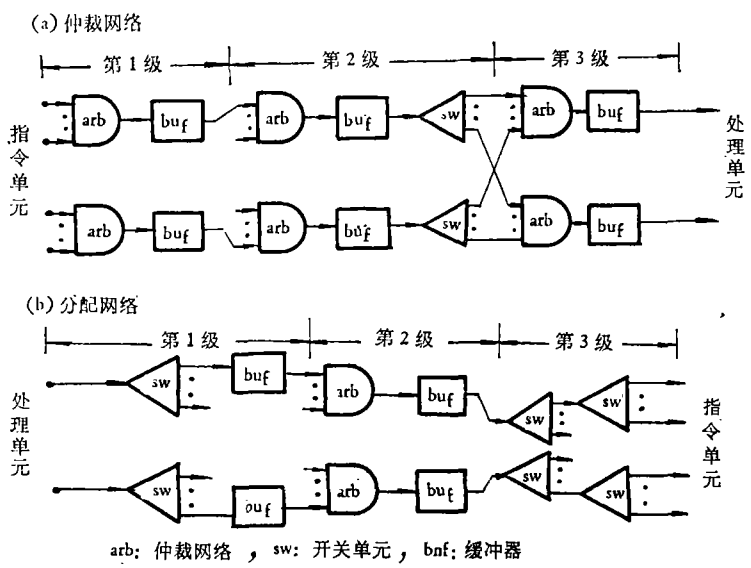

图 8

面的操作包在缓冲器中等待。所以仲裁网起到指令译码、分配处理单元和先进先出缓冲区的作用。仲裁网入口信号多，采用字节串行方式，减少与存储部件的联系。输出出口少，采用并行方式，提高传输速度使之与进口匹配。因此在仲裁网中需要有一步骤实现串行到并行的转换。相反，在分配网中需要并行到串行转换。分配网与控制网的结构与仲裁网类似，不赘述。

数据流计算机的路径选择网既不同于一般的总线结构，也不同于多处理机系统的纵横开关。在数据流计算机中，信息流的方向是单向的，所以不要求象纵横开关和总线那样必须立即给回答信号，而且往往有几个指令单元就绪，准备利用同一处理单元，所以数据包和控制包的传输延迟无关紧要，在一定范围内它不会影响系统的处理速度。数据流计算机的路径选择网络保证了存储器中各指令之间的通讯与它们在存储器中的位置无关，程序员可以随机地存放，不必考虑复杂的内存分配。数据流计算机的路径选择网络的逻辑复杂性为 O(Nlog N)，虽然比纵横开关的复杂性 $O(N^2)$好一些，但仍然大于线性关系。为了减少其复杂性，提出了许多新的想法，如动态分配资源，选择最佳级数等，这方面的研究是数据流计算机的重要课题。数据流计算机的一条指令的执行周期等于信息包通过两个网与相应的处理单元的时间。一般处理单元的延迟时间是固定的，而两个网络由于冲突，延迟时间将随机变化。这种系统的关键是仲裁网的出口和分配网的进口，这部分电路采用 ECL 代替 TTL 可以提高效率。

数据流计算机还没有真正实现，但设计方案已有许多种。除了 MIT 的基本数据流机和具有过程调用功能的数据流机外[4),5)]，还有数据流多处理机方案[3)]，法国的 LAU 系统[2)]，英国曼彻斯特大学的带名字标号数据流机和 Newcastle 大学设计的带可修改存储器的数据流计算机(JUMBO)[8)]等。JUMBO 计算机的设计结合了控制流和数据流计算机两种不同的思想，独具特色。下面将专门介绍它的设计原理。

四、数据流与控制流相结合的设计思想

在介绍 JUMBO 计算机之前，先比较一下数据流计算机和常规的控制流计算机。下面以一个赋值语句的几种不同的实现方法说明两种设计的区别。赋值语句为：

$$a:=(b+c)*(b-c)$$

图 9 画出了四种不同的实现方法。a）是冯·诺依曼方法；b）是扩展的多控制流方法；c）是数据流计算机的实现方法。其中方框表示指令，I_1，I_2，I_3，a，b，c，t_1，t_2都是名字或地址，圆圈表示存储单元，？表示数据记号，实线弧表示数据流，虚线弧表示控制流。从图 9 明显地看出在常规的控制流方法中，通过指令共享存储单元来实现数据通讯。产生数据的指令和使用这一数据的指令公用同一名字的存储单元，通过“访问”才能取得数据。而数据流计算机没有单独包含一个数的存储单元或嵌在指令中的数据名字的概念。指令不能指定到某单元去取数，而是由记号直接传送“值”，这样，一个常数如果要被不同的指令使用一千次，则要占一千个接收器，在循环过程中，常数也需要再生。数据流机实现数组运算也不方便，而控制流机可通过变址寻址很方便地对数组等数据结构加工。但控制流计算机每一个中间结果都要送存储器，每次执行指令都要取数和存数，显然这降低了效率。JUMBO 机取两者之长，采用了一个折衷方案，如图 9 d)。它保留了 a，b，c 做地址名字，而去掉了 t_1和 t_2 这些中间结果存储单元，直接用数据记号传送。因此 JUMBO 机中数据分成两类。一类是常数或需要多次使用的数，称之为要存储的数据，存于所谓半永久存储器（类似于通常的存储器）；另一类是只用一次的数据，由数据记号传送到指令激活部件中的目标指令。前一类数据是有地址的，通过访问存取，可以修改内容。这是控制流方式；后一类数据没有名字，

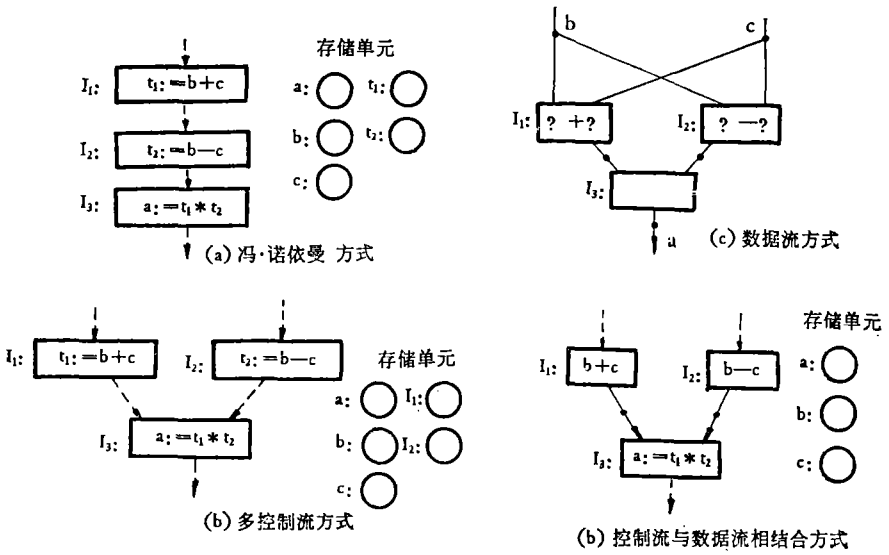

图 9

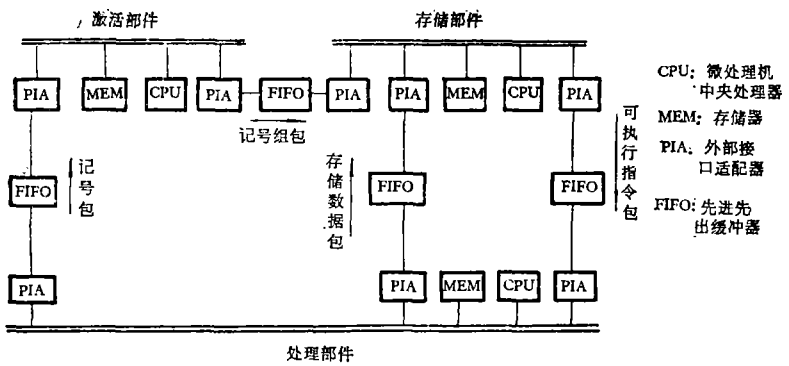

图 10

采用数据流方式。这样既节约了存储空间，便于加数据结构，又节省了读写中间结果的时间。

JUMBO的总框图如图 10。它分成三大模块。每一模块由一台微处理机实现其功能。模块之间由 8 位总线通过PIA(外部接口适配器)互相连接。三个模块独立异步地工作。第一个模块是指令激活单元。它收集数据记号，输出一个记号组包。这部分功能类似于MIT机的指令存储部件，但JUMBO机激活单元输出的各记号中的操作数不一定是值本身，可以是数的名字，即地址。第二个模块是存储器，其功能与通常的存储器相似，通过“访问”为指令提供真正的操作数，形成可执行的指令包。第三个模块是处理部件。这一部分还包括仲裁网和分配网的功能。处理部件由多个相同的处理单元组成，因而减少了仲裁网的任务。JUMBO机规模小，路径选择网络较简单。处理部件输出的结果有数据记号、控制记号和要存储的数据三种。前两种形成记号包，送激活部件；后一种形成存储数据包，送存储器。在这种折衷方案中，给一条指令提供数据有三种方式：(1) 通过数据记号；(2) 嵌在指令中的数据；(3) 按地址访问某存储单元。激活一条指令有两种方式：(1) 数据记号全部到达；(2) 控制记号通知某些事件已发生。有些指令由控制记号和数

据记号的组合来激发。

下面以一小段程序为例说明JUMBO机的信息形式，程序如图11所示。这段程序的内容是：指令1存值26于X单元，送一控制记号到指令3，指令1的信息形式是存储数据包；指令2送一数据记号(值为10)到指令3，信息形式是记号；指令3把记号的值10与X单元相加，其信息形式先是存储的指令，记号到达后变成可执行的指令；指令3的结果由数据记号(值为36)送到指令4，指令4将36与X单元相乘，其信息形式与指令3类似。

Newcastle大学设计JUMBO机的目的是在组合的模型中研究控制流和数据流的相互影响。他们力求结构简单，只用三台微处理机构成，使每一所大学都可以研制。这种计算机的

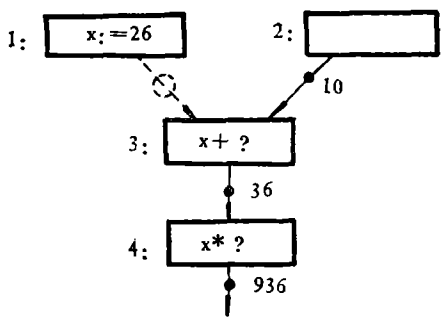

图 11

程序既可以按纯数据流方式编也可按多控制流方式编或两者的组合。

五、几点看法

1) 数据和程序的关系是计算机体系结构所涉及的主要问题。数据流计算把冯·诺依曼计算机中指令和数据的关系颠倒过来了，突出了数据的主导作用，在理论上这一变革对计算机发展是一个突破。在实现技术上，它与流水线机、多机系统、计算机网络等有密切联系。

2) 数据流计算机是否有生命力取决于它的成本。数据流计算机必须有大容量的存储器和大规模的路径选择网络。它所得到的高速度以耗费更多的存储空间和网络连接元件为代价。这是一种用空间换时间的策略。因此器件的发展程度对数据流计算机的实现有决定性的影响。

3) 数据流计算机具有模块化的特点。与当前迅速发展的微处理机技术是协调的。采用微处理机做模块，按照多机系统的方式的构成数据流计算机是一条重要的实现途径。将控制流计算机的设计思想和数据驱动原理结合起来，互相补充，将有利于新的计算机体系的实现。

参考文献

[1] J. B. Dennis, "First Version of Data Flow Procedure Language", Lecture Notes in Computer Science, Vol. 19, pp 362—376, 1974.

[2] A. Plas et al, "LAU System Architecture: a Parallel Data-Driven Processor Design Based on Single Assignment", Proc. 1976 Intl. Conf. on Parallel Processing, pp. 293—302.

[3] J. Rumbaugh, "A Data Flow Multiprocessor", IEEE Trans. on Computers, Vol. C-26, No. 2, 1977, pp. 138—146.

[4] J. B. Dennis et al, "A Preliminary Architecture for a Basic Data Flow Processor", Proc. of the Second Annual Symp. on Computer Architecture, 1975, pp. 126—132.

[5] J. B. Dennis, "The Varieties of Data Flow Computer", Collection of Papers on Advanced Computer Architecture and Processing Techniques, 黄铠编 pp. 200—209.

[6] J. B. Dennis et al, "A Highly Parallel Processor Using a Data Flow Machine Language", 同上, pp. 210—277.

[7] D. Misunas, "Petri Net and Speed Indepent Design", CACM, Vol. 16, No. 8, 1973, pp. 474—481.

[8] P. C. Treleaven, "Pricipal Components of Data Flow Computer", Euromicro, 1978.