国防科技大学学报
JOURNAL OF NATIONAL UNIVERSITY OF DEFENSE TECHNOLOGY
第11卷第1期 1989年3月 Vol.11,No.1

数据流模型机SDS-1的实现

吴 涛 李 勇

（计算机系）

摘要 文中全面地介绍了数据流模型机SDS—1的特点和实现方法。SDS-1模型机是一种同步和异步相结合、数据流和控制流相结合的数据流模型机，它能开发复合函数级的并行性。该模型机在Intel86/310微机和8个Intel86/12单板机组成的多机系统上实现，达到了80年代初国际先进水平。

关键词 数据流，模型机，复合函数，并行处理，函数语言

分类号 TP37，TP399

引 言

新一代计算机的研究旨在突破传统的 Von Neumann 结构，研制更高性能的计算机，以满足各个领域，特别是人工智能应用的要求。数据流计算机是大有前途的研究课题，因为数据流计算机具有潜在的高性能，有希望成为新一代计算机的主要结构之一。

数据流计算模型是60年代末提出来的，至今数据流计算机还没有诞生第一代商业产品。这和计算机的发展是很不匹配的。人们普遍认为数据流计算机的开销过高。造成数据流计算机开销过大的基本原因是并行的级别太低，或者说并行的粒度太细。一个系统中的并行可以划分为多个级别，如作业级、进程级、函数级、指令级等。但在数据流计算机中，主要是通过指令级的并行来实现计算并行性，因此，有必要研究如何提高并行的级别，以降低系统的开销。

SDS(Synchronous Dataflow System)模型是一个同步和异步相结合、数据流和控制流相结合的数据流计算模型[1,2,3]。在SDS模型中能开发两个层次的并行，即复合函数级和复合函数内部的并行；复合函数内部采用同步操作，有效地降低了系统的开销；引入控制标志，冗余操作被消除；可正确地执行非严格的函数，保证语义无误。SDS—1是实现SDS计算模型的计算机体系结构，SDS-1是一种树型层次结构，这种结构降低了对互连网和存贮器的设计要求，减小了系统开销。实现模型机SDS-1是为了客观地评估SDS模型及相应的SDS-1体系结构的正确性和可行性，并为研究函数语言VAL的编译器，特别是研究复合函数划分的有效算法和优化算法提供一个良好的数据流运行的环境。

1988年4月10日收稿

1 SDS模型与SDS-1体系结构

1.1 SDS模型

SDS计算模型是在数据流计算模型中引入同步控制机制，使数据流和控制流相结合的数据流计算模型。

通过对数据流计算模型深入、全面地分析可知，造成数据流系统开销太大的主要原因有两条：

（1）指令级并行级别太低，或者说并行的粒度太细；

（2）无控制的异步操作，增加了复杂性，使开销增加。

因此，欲设计高效、实用的数据流计算机，必须做到：

（1）提高并行级别；

（2）在保证并行的前提下，引入必要的控制，利用同步操作减小指令级并行的开销。

SDS模型正是在此思想指导下提出的，它具有下述特点：

（1）SDS模型开发两个层次的并行。在高层次，即复合函数级，相关函数类似异步的数据流图，函数之间仅有数据驱动关系，不存在传统多机系统中多进程间同步和通讯关系。这样做，一方面方便程序设计，另一方面可以简化体系结构。这是数据流异步操作带来的优点。由于执行的单位是复合函数而不是指令，因而异步操作引起的开销平均到指令已被大大地减小。

（2）复合函数内部采用同步操作，有效地降低了系统的通讯开销。在一个复合函数的DFGC图中[1,3]，如果它不含函数调用操作，则它的DFGC图可以变换为平衡的DFGC图。由于平衡的DFGC图可以按序执行，因此可以采用按序点火(firing)的规则。相当于同时点火一排指令，这就是同步的含义。此外，可以把一个平衡的DFGC映射到一组处理器阵列上。假定不考虑存贮器和互连网的延迟；任何操作的延时都是单位时间；全部的处理器由一个共同的控制器控制，自上而下地按序启动每排处理器，就形成了整个部件在控制器管理下同步运行的宏流水线。这样即开发了并行，又保持了小的开销。

（3）引入控制标志，冗余操作可以消除。大量的冗余操作对实际的数据流系统来说，会造成系统资源的紧张和各种冲突的增加。引起这一问题的原因是，在数据流图中起控制作用的标志与一般的数据标志不加区别，而控制标志必须在操作点火后方能起作用，不能在操作之前决定一个函数是否进行操作。因此，引入控制标志以有别于数据标志可以减小冗余操作，提高系统的效率。

（4）可正确地执行非严格函数，保证语义无误。条件函数COND定义为：if B then F elseG．它是一个非严格的函数。如果B为真，COND的值由F决定，如果G的输入无定义，COND实际上也有定义。但对数据流图来讲，若COND为图中的一个结点，COND结点的任意一个输入没有定义，COND就不能点火，因此就没有定义。故数据流图不能表示非严格的函数，而在SDS模型中，可以正确地执行非严格的函数。

1.2 VAL语言简介

VAL语言和ID语言是专为数据流计算机而设计，特别是VAL语言，更适合大规模数值计算领域，尽管它也有不足之处，但与LISP和FP相比仍略胜一筹(LISP不适合数

值计算领域，FP不易被硬件支持），故我们选择了VAL语言。

VAL语言是美国MIT的J. B. Dennis教授领导的一个研究小组在1979年提出的。VAL是a Value-oriented Algorithmic Language的缩写，即面向值的算法语言[4]。VAL特别适合高性能计算机受限制的数值计算领域，它的基本目标是为MIT开发高性能数值计算的数据驱动计算机。VAL语言有以下六个方面的优点：

（1）VAL语言中没有变量的概念，仅有值的名称。因此，它没有Von Neumann结构中的全局存贮器和状态的概念，它遵循单赋值规则，没有副作用，任何操作都是函数，故它是函数语言。

（2）VAL语言提供了丰富的数据类型。基本类型有布尔型、整型、实型和字符型。数据结构可以是数组或记录，且容许数组和记录嵌套定义，深度不限。

（3）VAL语言是一种强类型语言。VAL中任何函数的变元和计算结果的数据类型在函数定义的首部加以说明，函数内部值名的类型也都在函数内部加以说明。所以编译程序很容易发现任何非法类型的使用。

（4）VAL程序是一组分别编译的模块集合。每一模块包含一个外部函数，可供其它调用，一个模块也包含一些仅仅容许模块内部调用的内部函数。VAL程序的模块结构适合程序设计的结构化。

（5）VAL语言给每种数据类型都提供了一些错误值，如undef是无定义值，供程序运行时的例外处理。这一方面使VAL语言语义完整；另一方面也有利于程序调试和并行处理。

（6）VAL语言中没有规定语句的执行顺序，也没有GOTO之类的控制语句。语句的执行顺序不影响计算的结果。同时VAL语言还提供了forall/construct和forall/eval结构来明显表达算法中的并行成份。这些都有利于开发程序的并行性。

VAL语言也存在一些问题。VAL仍是一种试验性语言，不成熟，还有一些问题没有解决。在VAL提出之后一直在修改；VAL没有输入、输出手段，特别是交互式输入、输出手段；用VAL编写的程序的形式不够方便自然等。但对数据流计算机来讲，VAL不失为一种良好的实用语言。

1.3 VAL语言与SDS模型的抽象机

VAL语言是基于数据流计算模型而设计的函数语言，这与SDS模型的要求相一致。SDS模型可以恰当地表达无定义的值，这为VAL语言中引入的多种错误值和例外处理提供了基础。

VAL语言编译程序对程序的每个模块进行独立地编译，最后再链接为可运行的目标代码。编译程序的任务是把每个源程序模块变换为函数相关图，再把每个复合函数变换为平衡的或一般的DFGC图。一般说来，编译程序首先将源程序变换为原子操作构成的DFGC图，再根据其结构，划分为不同的复合函数。

为了支持VAL的实现，在SDS模型之上又提供了与语言成份相对应的模式，这可以视为实现VAL语言的SDS模型抽象机[3]。这些模式包括模块的模式、条件结构的模式、forall/construct结构模式，forall/eval结构模式和其它模式。编译程序的任务之一是将源程序变换为这些模式。

1.4 SDS-1体系结构

SDS-1体系结构是根据SDS模型，为支持高级函数语言而提出来的。SDS-1结构充分反映了SDS模型的特点。

SDS-1是树型层次结构，这与SDS模型的层次结构相适应。它的抽象结构如图1所示。根结点中有全局控制器和存贮器，它们与多个局部控制器和存贮器相连接，通过局部控制器和局部存贮器再与多个处理器相连。复合函数和数据都放在根结点，然后发送至局部存贮器；再在局部控制器管理下，发送至多个处理器并行执行；其结果送回局部存贮器，再送至全局存贮器。处理器之间不能直接通讯，需往局部存贮器互连，各处理部件间的通讯通过共享根结点的存贮器来实现。

图2所示的SDS-1体系结构是对图1抽象结构的实现。

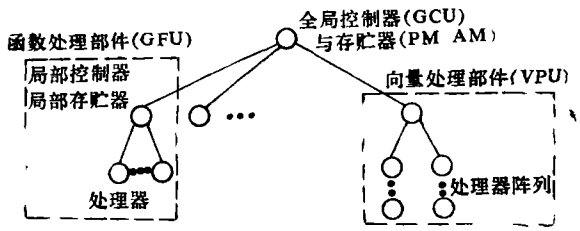

图1

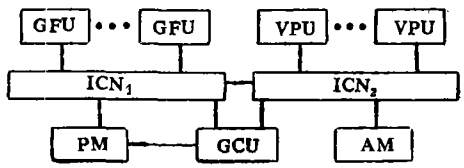

图2

GCU(Globe Control Unit)是全局控制器，它识别那些复合函数可以点火运行。GCU中有一个有效函数队列EFQ(Enabled Function Queue)。当满足点火条件且控制标志为真时，它们的函数标题指针被送入EFQ排队，GCU按照先进先出(FIFO)策略取队列头送往处理部件。

GFU(General-purpose Function(processing)Unit)是通用函数处理部件。GFU可以按照同步作业流水线的DFGC模型处理一个复合函数。其输入、输出数据可以是标量，也可以是数组元素。

VPU(Vector Processing Unit)是向量处理部件，它按照同步流水线DFGC模型处理数组。

PM(Program Memory)是程序存贮器，用来存放复合函数的程序。

AM (Array Memory) 是数组存贮器，用多叉树结构实现，负责完成各种数组操作。

ICN(Inter Connection Network) 是互连网络，它负责各处理部件之间及各处理部件和全局控制器、存贮器之间的连接。

2 模型机系统

研制SDS-1模型机系统有两个主要目的：一是通过在模型机上运行各种典型程序，实测运行速度和处理部件之间的关系，研究系统实际并行度和开销的大小，从而论证SDS-1体系结构的优劣；其二是研制函数语言VAL的编译器，特别是研究划分复合函数的有效算法。在此基础上进一步研究语言的特点，为设计更合适于数据流计算机的函数

语言打下基础。可见模型机不但是研究硬件结构的手段，也是软件研究的环境和基础。

SDS-1的结构分为两层，开发两级并行。考虑到实际条件的限制，模型机仅模拟SDS-1的高层次，开发复合函数一级的并行。

2.1 模型机的物理结构

模型机由一台主微机和八台单板机构成，它们经一组单总线互相通讯，形成紧耦合系统[5]。其结构如图3所示。

主机（主微机）管理外设，与外界通讯，并执行SDS-1的GCU部件和存贮器（包括PM和AM）的功能。八个单板机模拟SDS-1的八个处理部件，它们实际执行复合函数的各种运算。

主机选用Intel86/310机，它是16位微机，由三块插件组成。一块是主机板86/30，其中有CPU、浮点部件、128K的RAM、32K的ROM和串并行接口电路等；另一块是512K的RAM存贮扩展板；第三块是磁盘控制器板，这三块板都接在多主总线上（总线上可接多个CPU作为主设备共享，争用总线）。八块单板机都是Intel86/12，也是16位微机，全接在总线上。各单板机本身有32KRAM的双端口存贮器，供单板机自身和其它主设备访问。512K的扩展存贮器和各单板机的RAM存贮器可以被总线上的所有主设备共享，由此构成紧耦合的多机系统。各处理部件通过共享主存交换数据，实现相互通讯。这是整个系统实现并行的基础。

2.2 模型机的逻辑结构

从逻辑上讲，模型机分四个层次。核心层是Intel86/310和八个Intel86/12所组成的物理层；核心层之外是模型机系统的分布操作系统；次外层是SDS-1虚拟机；最外层是VAL语言应用层。其结构如图4所示。

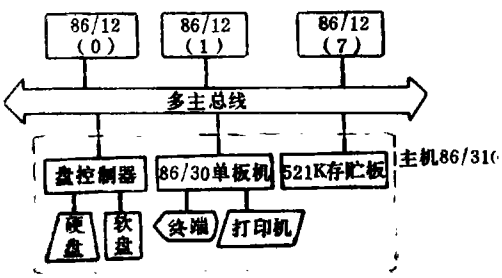

图3

图4

从语言实现的角度讲，可以把SDS-1分为七层，如图5所示。Intel86/12单板机没有OS（只有一个简单的监控程序），只能用硬件解释8086机器语言，属于M_1级机器；M_1、M_2、M_3和M_4构成了Intel86/310微机的层次结构，其中M_2是在Intel86/310系统上运行的实时多任务操作系统；M_5是用Intel86/310系统上的PLM语言实现的分布OS；M_7是应用语言VAL层机器；用VAL语言编写的程序经编译后形成复合函数的相关图CFG (Compound Function Graph)，即CFG语言机器M_6，它可以在M_5上直接运行。

2.3 模型机的操作系统

模型机的操作系统是分布式的。主机板上有自己的操作系统，它是在86/310机原配

置操作系统的基础上改造而成的。各单板机上都配置有管理程序，使单板机能完成模拟处理部件的功能。另外，各单板机上都有一个信箱和相应的通讯程序，按一定的通讯规程实现通讯。

在SDS-1模型机中，处理部件仅仅完成函数计算，相互之间不直接通讯，所以在模型机中各单板机只与主机通讯，相互之间不直接通讯。从功能上讲，该系统是分布式的。从形式上看，它是用信箱来实现的集中式通讯系统，如图6所示。

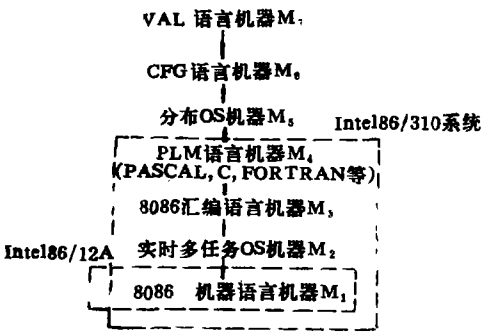

图5

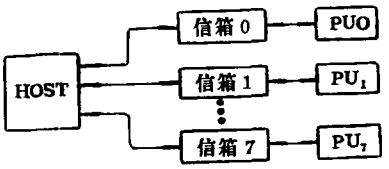

图6

信箱是通过各单板机上的共享存贮单元来实现的。从通讯的整个过程来说，HOST和相应的每一个PU(Processing Unit)不但是信箱中信息的产生者，也是消费者。但在某一确定时间，若HOST为生产者，PU必为消费者，反之亦然。

外设由主机操作系统管理。单板机如欲与外界通讯需由通讯程序把通讯请求与相应的信息经信箱传送到主机，由主机操作系统实现。

2.4 模型机的高级语言VAL

VAL语言是模型机实验用语言[4]。由于我们的目的是通过模型机研究SDS-1系统的可行性，并深入分析和掌握函数语言的特点，另外由于模型机规模的限制，因此没有必要，也不可能照搬VAL的文本。所以，我们对VAL语言的主要成分做了选择，略加修改。主要修改如下：

（1）数据类型，仅实现整型、实型和数组这三种基本类型与结构，数组至多三维。

（2）控制结构，不允许FOR循环，允许函数递归调用，不允许case语句。

（3）例外值，为简化编译，突出重点，例外值仅采用了一种undef，凡出错一律用undef表示。

（4）输入输出

在操作系统一级增加输入输出命令。

模型机上的VAL语言虽然对VAL文本做了一些修改和精选，但并没有影响和改变VAL的特色。

2.5 模型机上典型程序的运行结果

SDS-1可执行各种数值运算，如向量加、向量减、向量乘、求向量的内积、矩阵

乘、排序和求平均误差等。下面以排序、求平均误差和矩阵乘为例来说明SDS-1模型机的运行情况。

2.5.1 测试题目

(1)求平均误差(ERROR)

设平面上一点的坐标为(a,b)，现有n个测试点分布于(a,b)周围，它们与(a,b)的距离平均值为，

$$\frac{1}{n}\sum_{i=1}^{n}\sqrt{(x_i-a)^2+(y_i-b)^2}$$

当取$n=1024$时，按上式求其平均误差（平方根是通过迭代方法实现的）。

(2)矩阵乘(M-MUL)

设$A=[a_{ij}]32\times 32$，$B=[b_{ij}]32\times 32$，则$C=[c_{ij}]32\times 32=A\times B$，用外积法求矩阵$C$。

(3)排序(V-SORT)

待排序的数据被划分为n组，每组32个数据，即，$A=(a_1,a_2,\cdots,a_n)$，$a_i=(a_{i1},a_{i2},\cdots,a_{in})$，$i=1,\cdots,32$系统在运行时，将$A$分为$n$段并行排序，然后两两合并排序，最终生成一个数组。

2.5.2 运行结果

模型机系统在运行时，系统自动统计各种有关参数，以供分析评价用。如上三个题目的运行结果分别列表如下：

表1 ERROR:

部件数	1	2		3			4			
PU的编号	0	0	1	0	1	2	0	1	2	3
PU运行的函数	64	32	32	21	21	22	16	17	16	15
PU运行的指令	120896	60448	60448	44657	44657	31582	30224	30321	30224	30127
GCU用时	610	640		600			620			
总运行时间	101070	50610		36760			30660			
浮点结果	31744	31744		31744			31744			

表2 M-MUL

部件数	1	2		3			4			
PU的编号	0	0	1	0	1	2	0	1	2	3
PU运行的函数	32	16	16	11	11	10	8	8	8	8
PU运行的指令	112032	56016	56016	40559	40559	30914	28008	28008	28008	28008
GCU用时	540	490		520			470			
总运行时间	514200	319630		310120			255550			
浮点结果	66624	66624		66624			66624			

表3　V-SORT

部件数	1	2		3			4			
PU编号	0	0	1	0	1	2	0	1	2	3
	16	9	7	7	3	6	6	4	3	3
	89405	46058	43347	35214	30525	23666	25039	22328	21019	21019
GCU用时	310	300		350			300			
总运行时间	57310	31380		22980			19440			
浮点结果	17757	17757		17757			17757			

以求平均误差为例，重点分析一下加速比 r 和全局控制器GCU用时所占系统用时的比例(GCU/SDS)，其结果见表4.

表4　关于ERROR的r和GCU/SDS

部件数	1	2	3	4
r	1	1.997	2.750	3.297
GCU/SDS	0.6%	1.2%	1.3%	1.98%

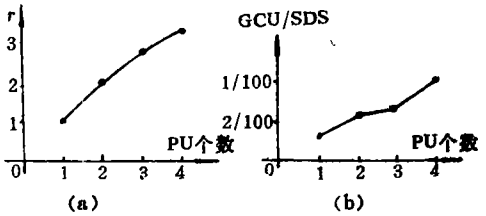

图7

如上的数据和曲线说明，

（1）系统的性能是很好的。系统在完成ERROR运算时，动态地并发出64个可执行的子函数，有效地发掘了程序中的并行性，使系统在多个PU运行下的加速比 r 达到比较理想的程度。

（2）全局控制器GCU是系统中一个重要的部件，通过它来识别可执行的复合函数，并派生到相应的处理部件，但它的时间开销是非常小的，如图7(b)所示，这说明利用复合函数以及动态派生复合函数所引入的系统开销是微不足道的，却能有效减少指令级开销。所以说，开发复合函数级并行是合理的。

3. 结　论

数据流模型机SDS-1的研制成功，不仅验证了SDS-1体系结构的可行性，而且为我们进一步研究函数语言的编译、程序划分算法、系统开销等等问题建立了良好的环境。

1987年11月来自全国的十几位专家教授对数据流模型机进行了鉴定，认为该模型机是很成功的，它首次提出并实现的数据流和控制流相结合的控制机制、开发复合函数级和复合函数内部两级并行的思想是可行的，且很有特色。该模型机是国内首创，它达到了八十年代初国际最先进的水平，对于我国进一步开展此领域的研究工作和追赶世界先进水平具有重要的价值和意义。

参考文献

[1] 刘桂仲，慈云桂．同步与异步相结合的数据流计算机SDS-1，兼评几种数据流计算模型．全国第五代计算机学术会议，1985

[2] Ci Yunqui. Architectare of the Synchronous Dataflow System SDS-1. Journal

ofComputer Science and Techaology: 1(1)
[3] 刘桂仲. 数据流与控制流相结合的计算模型与体系结构. 国防科技大学博士论文, 1986
[4] W B Ackman, J B Dennis. VAL—A Valued-oriented Algorithm Language. MIT/LSC, 1979; TR-218,
[5] Intel86/310, Intel86/12系统说明手册
[6] 数据流模型机SDS-1技术报告(鉴定资料). 国防科技大学

Implementation of Dataflow Prototype SDS-1

Wu Tao Li Yong

Abstract

This paper fully introduces the features and the approaches of the implementation for the dataflow prototype SDS-1. This prototype is the computer which combines the synchronous mechanism with the asynchronous one and the dataflow with control-flow. The parallel of composite function level's can be developed in SDS-1. The prototype SDS-1 has been implemented on the tightly-coupled multicomputer system consisting of a microcomputer Intel 86/310 and eight single-board computers Intel86/12. The prototype system has reached the advanced technical world level the beginning of 1980's.

Key Words dataflow, prototype, composite function parallel Processing, function language