



系統科學叢書  
Series of System Science



陳寶根 李 清

# 并行處理語言 OCCAM 及其應用

PARALLEL  
PROCESSING  
LANGUAGE  
(OCCAM) AND  
ITS APPLICATION

東南大學出版社  
SOUTHEAST UNIVERSITY  
PRESS

# 并行处理语言OCCAM及其应用

陈宝根 李 清 编著

东南大学出版社

## 内 容 提 要

并行处理技术是计算机科学工程发展中最新成就之一, 并已进入实用阶段。

Occam是一种新型的高级并行处理语言, 能够很好地描述并行进程间的并发和同步通信概念。

全书共十四章。从并行处理的基本概念入手, 比较全面、系统地介绍了Occam语言的各种进程结构、数据类型、通道通信、实时程序设计、并行系统配置、语言的编程风格及应用举例。同时也扼要介绍了它的开发系统TDS和高度集成的英国第五代并行处理芯片Transputer(晶片计算机)。

本书内容全面, 系统性强, 层次清楚, 概念突出, 是国内首次向广大读者系统介绍Occam并行处理语言的专业书籍。

本书可作为大专院校计算机、自动控制和计算数学等专业的本科生、研究生的教科书。也是从事计算机软件设计、并行处理技术开发和计算机仿真、控制及图象处理等方面科技人员的一本很好的参考用书。

**责任编辑** 陈天授

### 并行处理语言OCCAM及其应用

陈宁根 本 洁 编者

东南人 出版社出版

南京四牌楼2号

江苏省新华书店发行 涟水印刷厂印刷

开本850×1168毫米 1/32 印张10.3125字数285千

1990年6月第1版 1990年6月第1次印刷

印数: 1—1500册

ISBN7-81023-295-9

TP.22

定价: 2.40元

## 《系统科学丛书》出版说明

当今人类社会已经由“机器时代”跨入“系统时代”。系统科学是新时代占主导地位的科学，它是范围广、渗透性强的综合性学科。现实世界错综复杂、千变万化，但是，只要站在系统科学的高度就能透过其复杂的表象，抓住其主要特征，研究其相互关系，找出其共同规律，探求事物的本质。

为了适应科学发展的形势，加强出书的计划性和系统性，在有关专家的大力支持下，我社决定出版《系统科学丛书》。本丛书的选题围绕系统科学的基础理论、方法及其工程应用，主要收录有关学术专著、研究生教材或参考书，是侧重理论性和方法论的高层次系列书。相信本丛书的出版不仅会对从事系统科学、控制论、信息论、运筹学、系统工程等学科的专业人员、研究生、高年级大学生有所帮助，而且也便于其它领域的科技工作者拓宽思路、有所借鉴，从而促进边缘学科、交叉学科的发展。东南大学出版社愿为繁荣系统科学尽绵薄之力，努力扶植学术著作的出版，欢迎国内外专家学者踊跃投稿。

本丛书将陆续出版，各册相互独立，自成体系，编号以出版先后为序。

东南大学出版社

## 《系统科学丛书》编审委员会

**主任委员：**钱钟韩

**副主任委员：**冯纯伯 徐南荣

**常务编委：**陈天授

**编委：**（以姓氏笔划为序）

史 维 邢汉承 宋文忠 盛昭瀚

黄可鸣

## 前 言

近几年来, 32位微处理机在各个方面都取得了巨大进展。含有多个微处理机的芯片已经出现, 系统集成亦已从各个角度实现, 其中尤以英国INMOS公司开发的Transputer(晶片计算机)最为突出。

Transputer一词乃派生于*transistor*和*computer*, 其含意是指象使用晶体管那样, 可以很方便地使用晶片计算机, 以组成各种计算机系统, 包括海量并行系统。

实际上, Transputer是一个采用RISC结构的单片微型计算机, 其内部具有局部存贮器以及用于联接其他Transputer的连接口。

一个标准的并行处理计算机系统配有十块B003-2板和一块用于联接IBM-PC/XT(AT)及其兼容机的接口板B004-4时, 它的处理能力可达400MIPS, 即每秒四亿次。如果将B003-2板改为T800-20板, 浮点处理能力将达60MFLOPS, 即每秒六千万次浮点指令。

由此可见, 晶片计算机的性能是相当优越的。正如清华大学李三立教授所说: “Transputer创始者的功绩在于他们有效地实现了系统的集成, 以及有效地设计并实施了处理机之间的通讯结构。”

由于Transputer不仅可以单独组机, 而且业已开发出可供直接插入IBM-PC、VAX11(含Micro VAX11)、M68000和Sun-3等微型计算机中的单板, 从而可大幅度地提高微型机的速度与性能。

可以认为, Transputer是微处理器大发展的必然产物, 是计算技术的新飞跃。它是小巨型机的基础。因而, 近年来, 受到极大的重视。英国的Meiko公司, 曾用80块板子, 共计314个IMST800, 构成了一台巨型机, 其性能达到了美国著名的巨型机Cray-2的水

平,但造价却不足它的 1/40。

为了从Transputer结构中获得最大的效益和加速并行处理技术的发展,INMOS公司还同步开发了Occam语言,使之整个系统网络都可用Occam语言来编程。Occam不仅具有高级语言的优点,同时也提供了充分利用Transputer特点的能力。

Occam是一种新型的高级并行处理编程语言,能够很好地描述并行进程间的并发和同步通信概念。Occam语言适用于当今先进技术和系统科学的发展,特别是有关科学计算,复杂系统的控制和仿真、高性能图形处理及计算机成象等。

Occam可使系统层次化、模块化。各种型号的Transputer都设计得能够对Occam程序进行简单、直接而有效的编译。不同型号的Transputer,在Occam水平上,接口对它们都是标准的。

综上所述,虽然Transputer1985年才研制成功,开始投入小批量生产,但发展迅速,在西欧应用普遍,美国和加拿大对其亦颇为重视。迄今为止,英国已有许多大学和研究机构应用Transputer和Occam语言做出了许多实际应用成果。如光线跟踪图象的快速实现、被噪声混淆了的图象识别、计算机实时成象、分子模型建立以及波的传播特性研究等等,其结果都是卓有成效的。

鉴于Occam语言和Transputer的先进性,业已引起国内许多专家、学者的青睐与反响,并已在不少高等院校和科研机构中得到应用。去年九月,作者曾有幸被邀请参加了一个关于Transputer和Occam语言的专家座谈会。与会人员一致认为,这一新技术的应用前景广阔,我国应加快这方面的推广应用步伐。一年多的实践表明,Occam语言和Transputer的应用热潮正在悄然兴起,并正在影响着控制论、信息论等系统科学的发展。

为了适应这一情况,及时出版一本系统而又完整地介绍Occam语言的著作是适宜的,是能对这一新技术的应用与发展作出微薄贡献的。Occam是以描述并行处理为特长的高级语言。显然,本书的出版,对推动国内并行计算技术和系统科学的研究与发展也是会有

帮助的。

全书共十四章。它主要包括两大部分的内容：

第一部分——Occam语言介绍。这部分内容共有十章。它从并行处理的基本概念入手，比较全面、系统地阐述了Occam语言的各种进程结构、数据类型、通道通信，实时程序设计以及过程和函数等语法规则。

第二部分——Occam语言的应用。从第十一章到第十四章，共四章。在扼要介绍并行系统的配置和高度集成的Transputer的基础上，着重叙述了开发、应用Occam语言和Transputer的开发系统（TDS）以及Occam语言的编程风格和程序设计实例。

为便于读者理解，在附录中完整地给出了中、英文对照的语法规则。

本书的初稿曾作为东南大学研究生课程《并行处理语言》进行讲授。出版前，作者又作了系统的整理和充实。在撰写过程中，作者力图保持程序语言的系统性和语法定义的严密性，同时又尽量避免语言的抽象和枯燥，从结构安排到具体内容描述，皆遵循由浅入深的原则。采用简明、通俗的语言和各种程序例子，把对Occam语言的概念性阐述和具体编程、应用结合起来，使读者易于理解、易于掌握，以期雅俗共赏，不仅使计算机专业人员有兴趣，也能对初次接触程序设计语言的读者有所裨益。但是，由于我们水平有限，又缺乏经验，对语法的解释难免带有片面性，所提供的编程例子也不尽完善，疏漏及错误之处，敬请有关专家、计算机工作者和广大读者指正。

本书承蒙东南大学邢汉承教授和孙志挥副教授进行审阅，提出许多宝贵意见，谨致由衷的谢意！

陈宝根 李 清

1989年12月于南京



# 目 录

## 第一章 引论

### 1.1 并行处理的基本概念

1.1.1 并行处理计算机..... 1

1.1.2 并行性..... 4

1.1.3 并行算法与并行处理语言..... 6

### 1.2 OCCAM并行处理语言的基本特点..... 12

1.2.1 OCCAM语言简况..... 12

1.2.2 基本特点..... 13

### 1.3 OCCAM语言的语法说明与程序格式..... 14

1.3.1 语法说明..... 14

1.3.2 程序格式..... 15

## 第二章 基元进程

2.1 赋值进程 ..... 20

2.2 输入进程与输出进程 ..... 23

2.2.1 输入进程..... 23

2.2.2 输出进程..... 24

2.3 空操作进程与停滞进程 ..... 25

2.3.1 空操作进程..... 25

2.3.2 停滞进程..... 27

## 第三章 组合进程结构

3.1 结构 ..... 29

3.2 串行结构 (SEQ) ..... 30

3.3 并行结构 (PAR) ..... 32

3.4 条件结构 (IF) ..... 35

3.5 循环结构 (WHILE) ..... 41

|     |             |    |
|-----|-------------|----|
| 3.6 | 择一结构 (ALT)  | 45 |
| 3.7 | 选择结构 (CASE) | 47 |
| 3.8 | 进程结构小结      | 51 |

## 第四章 数据、字符和数组

|       |         |    |
|-------|---------|----|
| 4.1   | 基本数据类型  | 53 |
| 4.1.1 | 整型      | 53 |
| 4.1.2 | 实型      | 54 |
| 4.1.3 | 字节型和布尔型 | 58 |
| 4.2   | 字符和字符串  | 59 |
| 4.3   | 常数和变量   | 61 |
| 4.3.1 | 常数      | 61 |
| 4.3.2 | 变量      | 63 |
| 4.4   | 变量数组    | 70 |
| 4.4.1 | 变量数组类型  | 71 |
| 4.4.2 | 数组段     | 75 |
| 4.4.3 | 多维数组    | 77 |

## 第五章 表达式

|       |                 |    |
|-------|-----------------|----|
| 5.1   | 表达式的一般描述        | 81 |
| 5.2   | 表               | 83 |
| 5.3   | 操作运算符           | 85 |
| 5.3.1 | 算术运算符           | 87 |
| 5.3.2 | 模算术运算符          | 88 |
| 5.3.3 | 求取最大的正整数和负整数操作符 | 91 |
| 5.3.4 | 位操作符            | 92 |
| 5.3.5 | 移位操作符           | 93 |
| 5.3.6 | 布尔运算符           | 95 |
| 5.3.7 | 关系运算符           | 96 |
| 5.3.8 | 求数组大小操作符        | 97 |
| 5.4   | 数据类型转换          | 98 |

## 第六章 通信与通道

|     |         |     |
|-----|---------|-----|
| 6.1 | OCCAM通信 | 104 |
|-----|---------|-----|

|                      |            |     |
|----------------------|------------|-----|
| 6.1.1                | OCCAM通信特点  | 104 |
| 6.1.2                | 死锁及其预防     | 109 |
| 6.2                  | 通道协议       | 114 |
| 6.2.1                | 简单协议       | 115 |
| 6.2.2                | 命名协议       | 118 |
| 6.2.3                | 序列协议       | 119 |
| 6.2.4                | 变格式协议      | 122 |
| 6.2.5                | 无格式协议      | 139 |
| <b>第七章    缩词</b>     |            |     |
| 7.1                  | 表达式缩词      | 140 |
| 7.2                  | 元素缩词       | 142 |
| 7.2.1                | 元素缩词的一般形式  | 142 |
| 7.2.2                | 数组的并行分割    | 147 |
| <b>第八章    进程阵列</b>   |            |     |
| 8.1                  | 重复符        | 149 |
| 8.2                  | SEQ阵列      | 150 |
| 8.3                  | IF阵列       | 153 |
| 8.4                  | ALT阵列      | 156 |
| 8.5                  | PAR阵列      | 158 |
| 8.5.1                | PAR阵列的一般形式 | 158 |
| 8.5.2                | 流水线处理      | 161 |
| <b>第九章    实时程序设计</b> |            |     |
| 9.1                  | 实时钟        | 173 |
| 9.1.1                | 定时器        | 173 |
| 9.1.2                | AFTER算符和延时 | 174 |
| 9.2                  | 优先级        | 176 |
| 9.3                  | 实时钟的应用     | 179 |
| 9.3.1                | 计算进程开销     | 179 |
| 9.3.2                | 实时时钟和日历    | 180 |
| 9.3.3                | 任务调度器      | 181 |

|             |                            |     |
|-------------|----------------------------|-----|
| <b>第十章</b>  | <b>过程与函数</b>               |     |
| 10.1        | 过程概念                       | 183 |
| 10.2        | 过程说明与过程调用                  | 183 |
| 10.3        | 函数说明与函数调用                  | 187 |
| <b>第十一章</b> | <b>配置</b>                  |     |
| 11.1        | 概念                         | 195 |
| 11.2        | 通道配置                       | 195 |
| 11.3        | 处理机配置                      | 198 |
| <b>第十二章</b> | <b>OCCAM语言和Transputer</b>  |     |
| 12.1        | 晶片计算机Transputer            | 201 |
| 12.1.1      | Transputer芯片               | 201 |
| 12.1.2      | Transputer系统               | 203 |
| 12.2        | OCCAM语言和Transputer         | 209 |
| <b>第十三章</b> | <b>Transputer开发系统(TDS)</b> |     |
| 13.1        | TDS概述                      | 216 |
| 13.2        | TDS的引导和使用                  | 218 |
| 13.3        | 编辑器                        | 219 |
| 13.3.1      | 折页                         | 219 |
| 13.3.2      | 编辑器功能命令                    | 222 |
| 13.3.3      | 实用功能命令                     | 227 |
| 13.4        | OCCAM开发实用程序包               | 230 |
| 13.4.1      | 实用程序                       | 232 |
| 13.4.2      | 程序的编译和运行举例                 | 237 |
| 13.5        | Transputer开发实用程序包          | 242 |
| 13.5.1      | 配置                         | 243 |
| 13.5.2      | 实用程序                       | 245 |
| 13.6        | 文件处理实用程序包                  | 251 |
| <b>第十四章</b> | <b>OCCAM语言程序设计例</b>        |     |
| 14.1        | OCCAM语言编程风格                | 255 |
| 14.2        | 程序设计实例                     | 259 |

## 附录

|                 |     |
|-----------------|-----|
| 一、OCCAM语言字符集    | 282 |
| 1.1 字符集         | 282 |
| 1.2 运算符         | 282 |
| 二、OCCAM语言保留字    | 284 |
| 三、OCCAM语法一览表    | 285 |
| 3.1 语法汇总一       | 285 |
| 3.2 语法汇总二(英文)   | 295 |
| 四、OCCAM过程库      | 305 |
| 4.1 整型算术运算过程    | 305 |
| 4.2 浮点过程        | 307 |
| 五、基本I/O过程库      | 308 |
| 5.1 数据到字符串的转换过程 | 308 |
| 5.2 字符串到数据的转换过程 | 310 |
| 5.3 通道读写过程      | 311 |
| 六、基本函数库         | 312 |
| 参考文献            | 315 |

# 第一章 引 论

## 1.1 并行处理的基本概念

### 1.1.1 并行处理计算机

早期计算机的全部结构都是按串行方式工作的，它遵循Burks等1946年设计的存贮程序计算机的基本概念。这就是人们常说的Von Neumann结构。这类计算机由三大部分组成：输入输出部件（I/O）、存贮器部件和中央处理部件（CPU）。中央处理部件实际上是由控制器和运算、逻辑部件这两者组成的，串行处理计算机的一个重要特点是在每一时刻只能按一条指令对一个数据进行操作：访问存贮器、运算或逻辑判断、输入或输出操作。也就是说，计算机要实现的各个操作必须按步就班地一条一条地顺序执行。

随着计算机在各个领域的广泛应用，对于计算机的性能、速度方面的要求也就愈来愈高。由于电子信息传输速度以光速（每毫秒1呎，即约30厘米）为极限，而且许多电路已达到在毫微秒的数量级内动作，因此，要再大幅度地提高串行处理计算机的速度与性能，单靠改进元器件本身的性能来实现乃是相当困难了。

人们从日常生活中体会到，如果将某件工作分解成若干个较小的部分，然后并行地去做它们则往往比一个人一步一步顺序地去做它要快得多。

在一个制造汽车的工厂中，厂长要把一辆汽车的各个部件分配到几个车间去同时生产，而不会只让一个人（一个车间）一个部件、一个部件地去生产。不言而喻，并行生产的速率远远高于串行生产

的速率。图1-1表示了汽车生产的流程示意图。

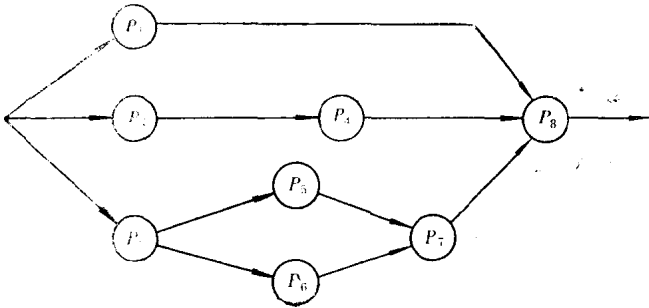


图1-1 汽车生产流程示意图

在开始时，三个车间可以同时进行加工，分别做  $P_1$ 、 $P_2$  和  $P_3$ 。在二车间中，完成了  $P_2$  后还要做  $P_4$ 。在三车间中，待  $P_3$  完成后， $P_5$  和  $P_6$  又可并行加工，然后再做  $P_7$ 。待三个车间分别把  $P_1$ 、 $P_4$  和  $P_7$  加工好后，就可以进行汽车的总装配 ( $P_8$ )。  $P_8$  一完成，一辆汽车也就造好了。

这时，生产一辆汽车所需的时间仅为完成  $(P_1 + P_8)$ 、 $(P_2 + P_4 + P_8)$ 、 $(P_3 + P_5 + P_7 + P_8)$  或者  $(P_3 + P_6 + P_7 + P_8)$  这些工作中的一个最长时间。

在这个例子中，它所以能让三个车间同时进行加工生产，是基于下述因素：

(1) 工厂拥有众多的工人和必要的厂房、设备条件，以使许多人能并行地开展生产活动。

(2) 生产汽车这一总的任务允许分割成多个分任务，而这些分任务又能在时间上重叠执行，从而使一辆汽车的生产可用较短的时间来完成。

就计算技术而言，是否亦能摆脱传统的串行结构方式，以发展并行性来提高计算机系统的性能与速度呢？设想是可行的。这主要是因为：

首先, 需要由计算机来处理的许多任务, 不仅需要而且有可能分解成多个并行任务。

**例1-1** 计算表达式  $4a^2 + (c+d)/b^2$  的值。

通常, 按串行方式来求解此表达式: 求积、和、积、商和和。我们也可以用并行的方法(如图1-2所示)来解算。这里, 把  $a^2$ 、 $c+d$ 、 $b^2$  等三种运算放在同一时刻开始执行。显然, 并行求解的时间来得少。

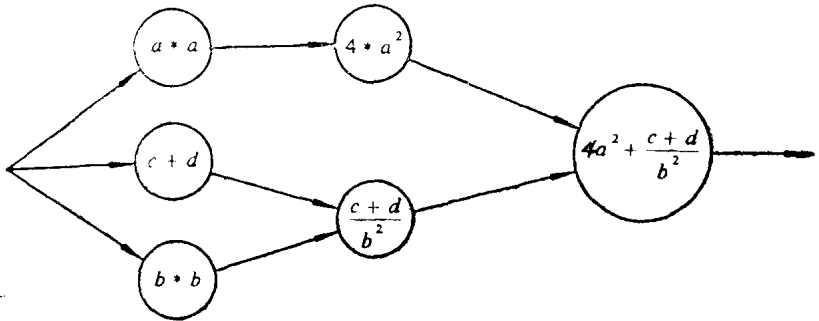


图1-2 表达式运算流程

**例1-2** 火力发电厂的计算机实时监控系統。

在这样一个计算机系统中, 它必须连续地监视对象的诸如蒸汽压力、蒸汽温度、汽轮机转速和发电机功率等上百个参数; 对现场数据要进行处理、发出调节控制信号; 遇有异常参数, 需要发出警告, 以提醒操作运行人员处理, 同时要记录打印, 以备查考; 系统要能随时响应生产过程的请求和运行人员的各种“召唤”, 并作出相应的反应。系统所配置的外部设备也比较庞杂, 除了具有完善的生产过程通道外, 还有各种外存贮器、打印机和制表机等设备。由于外部设备的信息传输率远远低于CPU的信息处理率, 因此, 如若仍然沿用串行的工作方式对它们一一依次处理, 显然是不能满足实时监控这一要求的。在软件设计上, 只要我们对每一个与外界有关的活动均建立相应的并发进程, 那就为并行的实现创造了条件。

其次, 为实现并行所需的设备条件可以获得。

就例1-2来说, 要实现那样的实时系统, 只要有一个多进程



(多任务)的实时操作系统就可以实现并发运行的要求。例如,我国早在1980年就研制成功的带页面分配保护的实时操作系统(MRTOS),它可以支持多进程的并发。它的运行环境是一台DJS100系列机,这是串行方式的。然而由于获得MRTOS这个软件系统的支持,导致各设备进程能分享CPU资源,满足并发要求,实现实时处理功能。

这是指软设备条件的获得。其他如多用户分享资源的分时系统,前、后台分享的多道程序系统等都属于此类。

另外就是硬设备条件的获得。这实际上是对计算机结构体系的变革与发展。

早在六十年代中期,美国就开始进行并行计算机的研制工作。到七十年代就陆续研制成功若干种通用或专用的并行计算机,如ILLIAC-IV、STAR-100、ASC、CRAY-10以及STARAN等。进入20世纪80年代,美国和日本等国家又相继生产了存储容量达800万字~3000万字、计算速度达5亿~10亿次的并行计算机。如Cyber 205、CRAY-1M、HITACS-810和IBM 308x等。在此期间,我国亦先后研制成功了757和YH(银河)两种并行计算机。英国的INMOS公司则在八十年代初期研制成功了具有巨大潜力的Transputer(晶片计算机)芯片,从而更为并行计算机的发展开辟了一个广阔的前景。

事实上,并行计算机是各类计算机中性能优越、功能极强的计算机,它拥有相当强大的数值计算和数据处理能力,专门用来解决那些在一般大型计算机上解决不了的科学计算、实时仿真、工程设计或者数据处理问题。正如R.W.Hocrney和C.R.Jesshope所指出的:本世纪八十年代,大规模并行技术则是在设计成批供应的计算机时出现的主要革新。

### 1.1.2 并行性

并行性,指的是多于一个事件在同一时刻发生或者在同一时间