微处理机工程用高级语言

〔英〕D.泰勒 等 著

朱炳洋 胡 洁 译

科学技术文献出版社

微处理机工程用高级语言

[英]D.泰勒 L.毛根 著

朱炳洋 胡 洁 译

科学技术文献出版社

1986

内 容 简 介

本书介绍了微处理机工程用17种高级语言，并详细阐述了什么是高级语言，为什么要应用高级语言，高级语言与微处理机的关系，17种高级语言各自的功能、特点和用途，软件开发系统和它的设计方案与标准规格，以及如何选择语言等。

本书适合于微机工程和软件系统的研究、设计和操作人员使用，也可作为大专院校师生用的参考书。

David Taylor and Lyndon Morgan
High—Level Languages for Microprocessor Projects
The National Computing Centre, England
1980

微处理机工程用高级语言

朱炳洋 胡 洁 译
科学技术文献出版社出版
北京京辉印刷厂印刷
新华书店北京发行所发行 各地新华书店经售
*
787×1092毫米 32开本 10.25印张 220千字
1986年10月北京第一版第一次印刷
印数：1—7000册
科技新书目：131—65
统一书号：15176·753 定价：2.10元

译 者 序

微处理机以惊人的速度发展是与现代计算科学技术的成就相辅相成的。对于这一具有划时代意义的新技术——计算技术方面的丰硕成果自然受到这一领域的工作人员的极大重视。在计算技术中占有关键地位的高级语言对微处理机技术究竟能起什么样的作用？人们能够在多大程度上和怎样才能充分利用这些科技成果？正是本文要论述的课题。

本书篇幅虽不大，论述也不够深透，但在一定程度上揭示了当前这一领域内一个敏感的和具有现实意义的主题。

本书内容全面，叙述清楚，对于初始触及微处理机技术的人来说，则是一本具有一定研究价值的参考书。

由于译者水平有限，差错和遗漏之处在所难免，恳请读者批评指正。

序

在计算机和半导体技术开始独立地问世以来，经过了廿年，诞生了由这两门新技术相结合的产物——微处理机。初期的微处理机也象最初的程序存储式计算机那样的原始。它们以机器编码写程序，并依靠简易按键开关和指示灯来输入程序。这正是计算机幼年时期的情景。不久之后就开始应用汇编语言，以及与之相应的翻译和开发设备。

紧跟着对微处理机开发系统进行了研究，以期提供高级语言翻译设备。在此之前，计算机的发展已促使某些高级语言向这方面演进。它们有的是因为适应一定领域内应用的需要，有的则是由于投合一些用户的某种偏好。这类系统采用这些精心研制的语言的一些子集和变体。但是早期的微处理机不能完全适合编写完善的翻译程序，从而防碍了高级语言的应用。今天微处理机结构的发展已经为处理高级语言创造了较有利的条件。先前应用这类语言的尝试筛选出了一批可供今天微处理机考虑采用的高级语言。进行选择看来不太容易。这取决对象任务的性质和应用特点，同时也与拟采用该语言的微处理机的性能有关。

在着手考虑选择语言之前，必须弄清两种微处理机应用方式间的重大区别。这两种应用方式是：

1．作成〈产品的〉组成部件；

2．用作一个工具。

如果欲将微处理机作为某一产品内部的一个组成部分，

那么就应考虑到该产品应能为那些不是程序员的人操作使用。对用户来说，该产品只不过是一种他们自己所熟悉业务的工具。微处理机应用的另一种情况则是被用作自身专业性的工具，就是说用户能以自己编写程序来满足其自身特定的需要。将微处理机作成产品本身的组成单元，必须选用一种语言以及相应的开发系统，以能全面验证产品的性能，并使其在经济上更为合理。还需要有一些与该语言系统有关的设备来为所设计产品中备置只读存储器单元。将微处理机作为一个工具或者个人计算机使用时，则对高级语言及其开发系统的要求就不一样了。

在根据实际需要选定语言后，必须仔细地考虑对这一语言的应用处理。决不能认为任何高级语言都是非常便于编制计算机程序的。即便它是一种易于运用的语言，而且在更正设计或改变要求时修改也方便，依然存在着因为这种改写而可能严重影响整个程序工作的危险。因此，对围绕应用任何计算机语言的设计研究有关规则进行探讨是十分重要的。

可见，为微处理机设计项目选择一种高级语言要考虑很多因素。其中包括：使用特点，应用方式，设计定型所选用的微处理机，以及用于一定语言系统的有关设计思想和规约。本书是为微处理机系统着手选择高级语言提供有用信息的一个尝试。它详细地讨论了有可能采用的语言，和他们在微处理机系统中实施的一些途径。今向各位推荐此书，作为当前出版的大量有关阐述微处理机及其应用有关书籍的有益补充。

D.Aspinall 教授

1980年1月8日

目　录

第一章 绪 论

微处理机问世已经很多年了，第一片硅片微处理机于1971年制成。大概经过五年之后，人们才开始看到这门技术的重大意义。即便是那时，也决没有认识到它的计算机功能的潜力。但是最近数年来，微处理机的应用和对它们的评价均有了飞跃发展。因此，尽管前一个时期微处理机还仅仅局限于电子学领域内，今天它们则对通常的数据处理（DP），以及几乎所有的信息处理，产生日益巨大的作用。以微处理机为核心组成的产品，作为一门在1956年还根本不存在的工业项目，在极短时间内即获得了高度评价，并显示了它们的生命力。今天，甚至还把它看成是一件对人类未来有着重大意义的事件。

近廿年来计算技术所取得的一个宝贵经验是软件对计算机体系的重要意义。问题就在于编制的程序应能在整个使用期限内工作无误，并且在需要时易于修改和开发。计算机技术发展的历程充斥着大量失败的程序。它们中有的曾引起过人们的兴趣，有的则几乎毫无结果。当然，在防止发生错误方面也取得了一定程度上的成功，但也付出了巨大的代价。我们今天面临的问题则是：一些计算机系统一旦失败，可能带来严重后果。一个软件的失误有可能造成千百万元的损失。它可能导致房屋建筑的焚毁，或者使得工厂倒闭。这类系统如果采用廉价的微处理机，自然要受到更大的关注，否则就必须能最大限度地保证它的可靠性。

本书正是出于上述考虑构思而成的。计算技术发展的主要成就之一是：认识到软件是一个系统的最根本的，也是最难以完整实现的部分。高级程序语言就是为解决这类问题而诞生的一个概念。它使得程序编制工作能以较快速度进行，并且也较便宜。这类技术是否能应用于微处理机系统？如果可以的话，应该如何来利用它们呢？什么语言体系目前可以利用？在现有的计算技术发展经验中，软件方面有些什么教益是微处理机系统所能借鉴的？

本书主要针对软件，极少论及硬件问题。目的是将现今计算技术方面的经验，引用到微处理机适用的语言方面来。着重强调软件的重要性和为微处理机系统选择语言的意义。

读者对象主要是下列三类人员：

具备一定计算技术知识、而拟从事微处理机应用工程工作的工程师；

现从事于大型机或小型计算机工作、即将开始微处理机系统工作的计算机人员；

审批工程计划、并对微处理机应用工程项目的顺利实施负有直接责任的管理工作者。

毫无疑问，所提到的两类程序人员间，在专业知识面上是很不相同的。但尽管如此，他们互相间确有很多应相互学习。通常的程序员可能需要学习电子学方面的知识，并要像懂得金融管理那样了解电子控制过程。同样，那些打算应用微处理机系统的工程师应能够、而且必须掌握现行的编程技术，并考虑如何在他们的工程项目中应用。这些就是本书的基本宗旨。

鉴于读者可能具有各不相同的专业背景，本书首先阐明何谓高级语言以及为何要应用它们。随后论述大型机和小型

计算机所用高级语言与微处理机高级语言之间的差别。接着对主要的一些语言作了述评，同时并在附录1中逐一地较详细地介绍了这些语言。这些章节的目的并不是要对这些语言作详尽讨论，而是期望读者能恰当地对待它们，了解它们之间的基本差异。最后讨论采用高级语言怎样进行系统设计和编程，以及未来的用户怎样来为一个已知工程项目合理地选择语言。举例说明了一些特定对象是如何选择和应用高级语言的（附录2）。

最后必须强调指出，本书也好，大多数其他程序设计语言方面的书籍也好，都不可能讲述具体怎样来编写程序。编程是一门能学会用来解决实际问题的技巧。一个程序就是以一特定语言代码表示某一问题的解答过程。本书讨论了程序设计技术。列举了某些算法的实例。这些均不受限于任何特定语言。但是，针对某一具体问题所决定采取的解决途径，则对语言选择可能有很大影响。

本书将有助于你的设计项目选择最佳语言。不过首先你得了解并学会运用正确的编程技术。

第二章　何谓高级语言

导　言

最初制造的计算机为其编制程序的唯一办法是采用机器码。这是一项专门化的工作，生成一个工作可靠的程序需要很长时间。同时还需要对计算机本身十分熟悉。

但不久除了计算机工程师外，其他行业的人们亦提出了应用计算机的要求。加之计算机工作者本身亦希望能有一个较简单的编程方法。这样就进而研制出了能以符号表示指令、地址和标号的编程方式。这种类型的语言系统被称之为符号语言或汇编语言。他们简化了程序编制工作，但需要有一个相当复杂的被称之为汇编程序的（assembler）来进行变换。这就导致了独立于机器的高级语言的诞生，他们也必须经过翻译或编译成机器码。这些翻译程序（translater）或编译程序（compiler）都是极其复杂的。

从此，一个新的计算技术的分支，编译程序的编制技术诞生了。这又促进了对程序设计语言自身的进一步研究。

通常采用计算机的具体应用与计算机本身并无任何关系。因此，早在五十年代就开始进行了研制与计算机结构无关、完全面向应用对象的语言的尝试。当时主要有两种语言：FORTRAN和ALGOL。

FORTRAN（FORmula TRANslation，公式翻译语

言）是美国IBM公司（国际商业机器公司）研制成功的，1958年前在该公司7940机上运行。它是针对科学公式的计算设计的。

ALGOL（ALGOrithmic Language，算法语言）是在欧洲研制的。它是针对解决问题而指定某一算法的一种独立于机器的语言。它几乎完全脱离了计算机，以至于连Input（输入）和Output（输出）指令都没有规定。

计算机一走出数学计算的领域，就出现了商业上的编程语言的需要。COBOL（COmmon Business Orientated Language，面向商业的通用语言）已经经过了一段时间的发展。这种语言被设计得极其近似于普通的书写体英语，并且成了公认的商业用程序设计语言。它最后于1966年形成了COBOL-66版本。在此以后出现了大批程序设计语言。他们当中有的是早先语言的扩展，有的则是新研制成功的。特别是出现了那些为编制语言编译程序、操作系统、实时系统和事务管理系统所需的极其专门化的语言，以及大量较通用的语言。另一个重大发展就是，计算机的设计工作本身也考虑到高级语言的要求。

虽然原先的动机是期望能较易地获得生成机器码的语言，但经验证明，高级语言也还具有许多其他好处。

我们首先来讨论一下各种不同级别的程序设计语言。从机器码本身开始，进而讨论汇编和宏汇编程序，一直到高级语言。

机器码

计算机的处理机由一系列工作寄存器（累加器）、进行算

术和逻辑运算的算术/逻辑单元、以及一些临时存储单元组成。

来自存储器或外围设备的输入数据被置入主寄存器（Accumulator，累加器，通常以字符A来代表）。同样，数据输出亦需经由此途径。

各种基本操作包括下列这些（带△的为可选项）：

—INPUT（输入），OUTPUT（输出）；

—ADD（加），SUBTRACT（减），△MULTIPLY（乘），△DIVIDE（除）；

—MOVE（主存储器和寄存器间数据的传输）；

—逻辑运算：AND（与），OR（或）等等；

—COMPARE（寄存器比较）；

—寄存器增量和其他特殊操作；

—CONTROL PROGRAM FLOW（程序流控制，例如JUMP，RETURN）。

正常情况下程序是由一个操作到存储在主存下面字节中的一个操作按顺序进行的。JUMP（转移）操作被插在其他操作中间，而将程序流向转移到主存中的另一地点，程序即由这一点开始再继续按顺序进行。

以这些基本操作表述的程序即称之为MACHINE CODE（机器代码）表达式。

机器码指令集

尽管基本概念是类似的，但大多数处理机（以及微处理机）的具体设计细节却极不相同。以下这些方面就可能是各式各样的：

—寄存器的数量和特点；

—工作字长，即基本运算中所采用的并行（二进制）位数。微处理机基本上为 8 位和16位。不同字长自然就相应地有不同的指令；

—INPUT，OUTPUT，FETCH，LOAD 和 MOVE 操作。这些都是针对外围设备或主存储器的功能的；

—算术和逻辑运算在性质上都是一致的，但在寄存器的具体应用上则可能不同；

—测试（比较）和转移指令；

—中断处理。

就软件工程师的观点来说，一种微处理机的特点就体现在它的指令的确切的组合上，或者它的指令集（INSTRUCTION SET）上。

程 序 示 例

现在以‘上推分类（Bubble-Sort）’算法作实例来说明机器码程序。这一算法可将一组数据按其递增的顺序整理排列。其原理是对整个数的清单进行扫描，检验每每相邻的数，如果发现次序不对，就将他们对调。经过一次由‘下到上’的扫描后，最大的一个数就会‘冒升（bubble）’到顶端。下一次扫描将排列第二大的数，…如此进行下去。为简单起见，假定所有的数都在 0～255 的范围内。这样他们全部都能以一个 8 位二进制数（即一个字节）来表示。

现在只讨论此算法中检验二个相邻数并在需要时加以交换的这一部分。以 INTEL 8080微处理机的机器码为例，该算法程序如下：