

计算机逻辑结构

山 东 电 子 学 会

一九八二年六月

序 言

《计称机逻辑结构》是一本介绍计称机结构的教材，它用深入浅出方式从计称机系统结构、传统计称机级、微程序级、操作系统计称机级、汇编语言级、到多级计称机，系统地叙述了第三代电子计称机的发展过程和设计思想，随着电子计称机科学的迅速发展，以往只教学生们就某个机型编写程序的时代已经过去了，必须介绍更多有关计称机结构的新知识，这些知识仅在几年前以前还属于研究阶段，例如，分段虚拟存贮，平行处理，微程序，自行虚拟化计称机等问题，本书试图满足这些要求当然，更新的研究成果又在不断出现了。但本书仍然是一本较好的基础教材。

学习本书的唯一先决条件是：对计称机科学应有一点入门知识或实践经验，同时也应该懂一点使用FORTRAN, COBOL, 或PL等高级语言编程序的知识。它可以作为大学教材使用，也可以作为广大科技工作者，设计人员学习电子计称机的参考读物。

译者文笔流畅，增加了读者学习兴趣，这本书的出版，为四个现代化建设中最关键的一门科学，电子计称机，提供了学习工具，它是值得欢迎和推荐的。

王运丰

1981年8月10日于北京

目 录

第一章 概述

- 1.1 语言, 层次和虚拟计算机 (2)
- 1.2 现代多层计算机 (4)
- 1.3 多层计算机的历史发展述评 (7)
- 1.4 硬件, 软件, 多层计算机 (10)
- 1.5 进程 (12)
- 1.6 本书概述 (16)

第二章 计算机系统结构

- 2.1 处理机 (20)
 - 2.1.1 指令的执行过程
 - 2.1.2 指令的并行操作
- 2.2 存储器 (22)
 - 2.2.1 位
 - 2.2.2 存储器编址
 - 2.2.3 无位
 - 2.2.4 辅助存储器
- 2.3 输入/输出设备 (36)
 - 2.3.1 I/O设备
 - 2.3.2 I/O处理机
 - 2.3.3 字符码
 - 2.3.4 误差纠正码
 - 2.3.5 频率相关码
- 2.4 信息传输 (45)
 - 2.4.1 数据通路

2.4.2 远距离通信

调制

异步与同步传输

单工、半双工和全双工传输

2.5 计算机网络------(51)

2.6 分布式计算机------(54)

第三章 常规机四层

3.1 常规机四层举例------(56)

3.1.1 IBM系统/360和系统/370

3.1.2 CDC6000, Cyber70, Cyber170

3.1.3 DEC PDP-11

3.2 指令格式------(67)

3.2.1 指令格式的设计标准

3.2.2 扩充操作码

3.2.3 指令格式举例

3.3 编址------(76)

3.3.1 立即型地址编排

3.3.2 直接地址编排

3.3.3 寄存器地址编排

3.3.4 间接地址编排

3.3.5 变址

3.3.6 基地址寄存器

3.3.7 堆栈地址编排

逆波兰表示法

逆波兰表达式的计算

3.3.8 PDP-11的寻址方式

3.3.9 编址方式讨论

3.4 指令分类 ----- (91)

3.4.1 数据传送指令

3.4.2 二元操作

3.4.3 一元操作

3.4.4 比较和条转移指令

3.4.5 转子指令

3.4.6 循环计数

3.4.7 输入/输出(I/O)指令

3.5 数据表示方法 ----- (109)

3.5.1 整数

3.5.2 浮点数

3.5.3 布尔变量

3.5.4 字符

3.5.5 字符串

3.5.6 阵列

信息向量

边界变址

3.6 控制流程 ----- (115)

3.6.1 控制流程和转移

3.6.2 过程

3.6.3 联立子程序

3.6.4 陷阱

3.6.5 中断

3.7 附录 ----- (132)

第四章 微程序层

4.1 处理机部件 ----- (151)

4.1.1 寄存器

- 4.1.2 总线
- 4.1.3 门线路
- 4.1.4 时钟信号
- 4.1.5 寄存器入口
- 4.1.6 算术逻辑部件和逻辑逻辑部件
- 4.1.7 处理机部件的组装
- 4.2 基本操作----- (157)
 - 4.2.1 寄存器传送
 - 4.2.2 寄存器读/写操作
 - 4.2.3 位测试
- 4.3 理想的目的机层----- (159)
- 4.4 理想的主机层----- (163)
 - 4.4.1 主机层寄存器
 - 4.4.2 主机层的ALU
 - 4.4.3 主机层的门电路和数据通路
- 4.5 门控时序----- (166)
 - 4.5.1 子周期
 - 4.5.2 执行ADD指令时的门控时序
- 4.6 微程序的门控时序----- (170)
 - 4.6.1 微指令
 - 4.6.2 微程序的执行
 - 4.6.3 微程序机
- 4.7 微程序设计语言----- (175)
 - 4.7.1 GATE微指令的表示法
 - 4.7.2 TEST微指令的表示法
- 4.8 目的机的解释程序----- (177)
 - 4.8.1 乘法指令的解释
 - 4.8.2 除法指令的解释

4.8.3 小结	
4.9 微程序层的设计	(186)
4.9.1 编码域段	
4.9.2 水平结构和垂直结构的比较	
4.9.3 存贮周期和重迭操作	
4.9.4 毫微秒存贮田	
4.9.5 通用微程序层与专用微程序层的比较	
4.9.6 微程序层小结	
4.10 微程序设计的优缺点	(199)
4.11 IBM370/125的微程序层	(201)
4.11.1 IBM370/125微程序层的结构格式	
4.12.2 IBM3125微指令	
4.12 PDP-11/40微程序层	(207)
4.12.1 PDP-11/40微程序层的结构格式	
4.12.2 UNIBUS 操作	
4.12.3 PDP-11/40的微指令	
4.13 巴勒斯(BURROUGHS) B1700	(217)
4.13.1 B1700的结构格式	
4.13.2 B1700指令系统	
附录1. IBM3125微指令的基本功能	(223)
附录2. B1712全部微指令的简单说明	(224)
附录3. B1700. 中寄存田和子寄存田功能	(225)

第五章 操作系统机田层

5.1 操作系统机田层的执行	(228)
5.2 虚拟 I/O 指令	(231)
5.2.1 串行文件	
5.2.2 随机存取文件	

- 5.2.3 虚拟 I/O 指令的执行
- 5.2.4 IBM370 虚拟 I/O
- 5.2.5 Cyber70 虚拟 I/O
- 5.2.6 PDP-11 虚拟 I/O
- 5.2.7 第三层 I/O 指令的比较
- 5.3 虚拟指令在并行处理中的应用 ----- (251)
- 5.3.1 进程的产生和消失
- 5.3.2 竞态条件
- 5.3.3 利用信号量进行同步
- 5.3.4 进程通信指令
- 5.4 其它的第三层指令 ----- (261)
- 5.4.1 目录管理指令
- 5.4.2 第三层机田的再设置
- 5.5 虚拟存储 ----- (265)
- 5.5.1 页面
- 5.5.2 页面的执行过程
- 5.5.3 请求式页面调度和工作方式
- 5.5.4 页面更换规则
- 5.5.5 沾污位
- 5.5.6 硬件图
- 5.5.7 页面尺寸和碎片
- 5.5.8 快存
- 5.5.9 分段
- 5.5.10 PDP-11 上的虚拟存储
- 5.5.11 MULTICS 虚存
- 5.5.12 IBM370 的虚存
- 5.5.13 段虚存和 I/O 文件
- 5.6 作业控制语言 ----- (275)

第六章 汇编语言及

6.1 汇编语言概述 ----- (299)

6.1.1 什么是汇编语言

6.1.2 汇编语言语句格式

6.1.3 汇编语言和PL/I语言的比较

6.1.4 程序调谐

6.2 汇编进程 ----- (307)

6.2.1 汇编程序的两遍通过

6.2.2 第一遍通过

6.2.3 第二遍通过

6.3 检测和分类 ----- (315)

6.3.1 检测

6.3.2 线性检测

6.3.3 二进制检测

6.3.4 分类

6.3.5 Hash编码

6.3.6 Hash函数和冲突

6.3.7 相关技术比较

6.4 宏指令 ----- (333)

6.4.1 宏指令的定义、调用和展开

6.4.2 带有参数的宏指令

6.4.3 条件宏指令的展开

6.4.4 嵌套宏指令调用

6.4.5 循环宏指令调用

6.4.6 嵌套宏指令的定义

6.4.7 在汇编程序中宏指令的效用

6.5 连接和输入装配 ----- (348)

- 6.5.1 连接程序的任务
- 6.5.2 目的模块的结构
- 6.5.3 装配时间和动态再定位
- 6.5.4 动态连接

第七章 多层机田

7.1 实现新的机田层的方法 ----- (362)

7.1.1 解释

7.1.2 翻译

通用宏指令处理程序

7.1.3 子程序的扩展

7.2 多层机田的结构设计 ----- (368)

7.2.1 “上一下”设计法

7.2.2 “下—上”设计法

7.2.3 “中—外”设计法

7.3 程序的可携带性 ----- (373)

7.3.1 通用程序编制语言

7.3.2 强制法

7.3.3 UNCOL

7.3.4 自身虚拟机

7.3.5 模拟法

7.3.6 网络

7.4 自虚拟机 ----- (384)

7.4.1 IBM VM/370系统

7.4.2 自虚拟机的目的

自虚拟机和分时

操作系统测试

机器数据的保护

7.4.3 自虚拟机的实现
虚拟机故障和异常情况
虚拟机 I/O 的模拟
自修改通道程序
影象页面表

7.5 高层机四结构----- (398)

7.5.1 寻址方式和解说符

7.5.2 高层机四指令

7.5.3 高层机四的优缺点

第一章 概述

数字计算机是利用赋予它的执行指令代替人解决问题的一种机口。描述如何解决相应问题的指令序列称之为程序。每条指令由电子电路直接执行。通常将指令分为三类：

这称型指令：如两数相加；

逻辑型指令：如判断一个数是否为全0；

传送型指令：把信息从存贮口的一部分传到另一部分。

计算机由基本指令组成语言，利用这种语言进行人机通信。这样的语言称之为机口语言。

在设计一种新的计算机过程中，设计人员首先要决定在机口语言中包含那些指令。在满足要求的情况下，指令力求尽量简单。机口语言越简单，硬件造价越低廉，维护也越方便。但是要用简单的机口语言解决实际问题需要做大量的复杂而繁琐的工作。为了解决这个矛盾，人们设计了一种新的语言。利用这种语言解决问题比用机口语言更方便。机口语言称为 L_1 ，而这种新的语言称为 L_2 。

用语言 L_2 写的程序有不同的执行方法：第一种方法是把程序逐条变成 L_1 指令序列，而后由计算机执行 L_1 程序，这种技术称为“翻译”；另一种方法是用 L_1 写出程序，把 L_2 程序做为它的输入信息，每条 L_2 指令都用一组等价的 L_1 指令序列来执行，边翻译边执行，这种方法称为解释法，用 L_1 写的程序称为解释程序。

翻译法和解释法是十分相似的，其相似之处是： L_2 指令都要用 L_1 指令序列来执行；其不同之处是：在翻译法中，全部 L_2 程序都首先转换成 L_1 程序，这样 L_2 程序就不使用了。

而直接执行新的 L_1 程序就行了。在解释法中每条 L_2 指令被检查和译码后就要直接执行。这两种方法都得到广泛的应用。

为了方便起见，我们可以不考虑翻译和解释的过程而假设存在一种虚拟机 Π ，它的机 Π 语言是 L_2 。如果这种机 Π 可以廉价创造成功，就不需要再有 L_1 ，可以把执行 L_1 程序的机 Π 取消。人们可以写出由计算机直接执行的 L_2 程序，如果直接执行这种程序因耗费太大而无法实现，就可以通过翻译法或解释法由直接执行 L_1 程序的机 Π 来完成。虚拟机 Π 是一种假想的而又能实现假想效果的机 Π 。

在实际应用中发现 L_2 语言与 L_1 语言相比並沒有太大的差别。用 L_2 语言仍然达不到方便地表达程序人员意图的要求，这就促使人们进行新的探索，创造一种新的语言，它比起语言 L_2 来更接近于人，而不接近机 Π 。这第三种语言我们称为 L_3 。人们可以用语言 L_3 写程序，如同机 Π 语言是 L_3 的计算机真正存在一样。 L_3 程序可以翻译成 L_2 程序，或者由翻译员写成 L_2 后执行之。这样我们就发明了一个语言系列，每一种语言都比它先前的语言对人更方便更合适，每一种语言又以它先前的语言做为基础。按语言系列就形成了计算机的分层结构。如图 1-1 所示：底部的语言最简单，顶部的语言最复杂。

1.1 语言，层次和虚拟计算机

语言和虚拟机 Π 之间存在着重要的关系。每种机 Π 都对应于一定的机 Π 语言，这些语言由全部机 Π 指令组成，机 Π 将执行这些指令。反之，语言也决定机 Π ，机 Π 必须能执行用某种语言写成的全部程序。然而，一些由语言决定的机 Π 太复杂，甚至不能用电子线路直接构成。尽管如此，我们仍然假定这种机 Π 的存在，这种机 Π 就是虚拟计算机。以 PL/I, ALGOL68 或 COBOL 为机 Π 语言的机 Π 是十分复杂的，或许在最近几年

内也会变成现实。

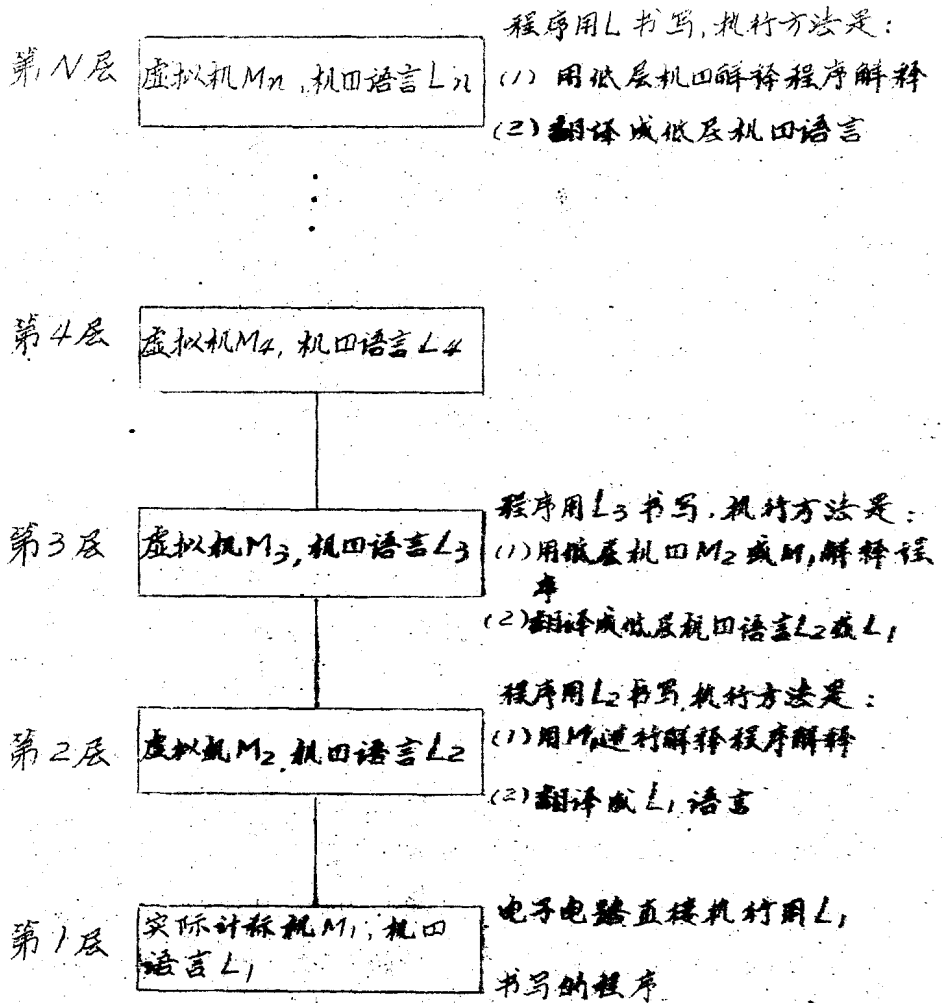


图 1-1 多层机

N层计算机可以认为由N种不同的虚拟机间所组成, 每层机间都有自己的机间语言。为了描述机间的性能, 我们将使用到“层次”和虚拟机的术语。只有用 L_1 语言写成的程序才能直接由电子线路所执行, 而并不需要使用翻译和解释方式。用 $L_2, L_3, \dots, L_N$ 写成的程序必须用解释或翻译的方法在低

层机口上运行。

用N层虚拟机口编写程序的人员可以不费低层的解释或翻译过程。无论低层机口用何种方式来执行，在它们看来都达到了执行程序的同样效果。

但是对于计算机设计者或研究人员来说仅了解N层计算机的最高层是远远不够的，必须清楚计算机的实际工作过程，分析每层虚拟机口的功能和特点。本书的重点就是研究建造多层计算机的概念和技术。用分层结构的方法来研究计算机，这对了解计算机的原理和工作过程是十分有利的。用这种方法从事设计也会创造出性能良好的机口。

1.2 现代多层计算机

现代计算机由两层或多层组成。如图1-2所示的五层机口系统并非罕见。市场上销售的许多计算机的第一层就指令系统和总体结构来说有很多共同之处。严格说来，两种机口语言不同的计算机，其第一层也不一样，但它们有许多共性存在，虽然它们已经明确定义了，但仍能使我们抽象出共同的特征进行讨论。例如，各种机口一般都有20~30种以上的指令，大多数指令是传送型指令，少数是测试型指令，这层机口称为“微程序级”，解释程序称为“微程序”，这层机口是第一层，它是建造其它各层的基础。这层指令由计算机电子线路直接执行。

第一层机口有一种或多种翻译程序。每种翻译程序都规定了它自己的第二层语言（虚拟机口的机口语言）。第二层机口也有许多共同之处，不同厂家生产的计算机，第二层机口的相同之处多于它们之间的不同之处。在本书中我们称机口的第二层为常规机口层。

计算机生产厂家，为了推销机口，都要出版一本“机口语言参考手册”，这种手册所讲的就是第二层虚拟机口，而不是第一层实际机口。这里所谈到的机口指令系统是由微程序用解释的方式来执行的，而不是由指令化的硬件来执行的。如果生产厂家对于他们的一种机口提供了能解释两类第二层语言的能力，就需要提供两种机口语言参考手册，分别对两种解释程序进行说明。

应当注意，有些机口，特别是老的机口没有微程序层，在这些机口中常规机口指令直接由电子线路来执行，不需要进行解释，因此，常规机口层是第一层而不是第二层。尽管如此，我们仍然把常规机口层看作是第二层而不管上述的特例。

第三层通常是混合层。第三层语言中所用到的大多数指令在第二层语言中也用到（没有任何理由规定一种指令仅能被一层语言应用而不能被另一层语言应用）。此外，在这一层中还有一些新的指令，不同的存储结构，能够进行两道或多道程序并行操作，以及其它各种特征。第三层语言比第二层或第一层有更多的变异性。

用第二层解释程序来执行的第三层语言所增加的新设备，因历史的原因，这种新设备称为操作系统。操作系统是用户与计算机之间的接口。用户通过操作系统使用计算机，因此，操作系统是方便用户提高计算机利用率，加快计算机响应速度的一种软件。其主要功能是管理中央处理机、内存、外部设备和信息，控制作业的运行，以及处理中断等。此外，各种子系统（编译程序、编辑程序、装配程序）和应用程序皆在操作系统控制下运行。在第三层中，与第二层相同的指令直接由微程序来执行，而不通过操作系统。换句话说，在第三层语言中，有些由操作系统解释程序执行，有些由微程序直接执行。我们称第三层为“操作系统机口层”。

第三层和第四层之间有根本性变化。最低的三层机口通常不

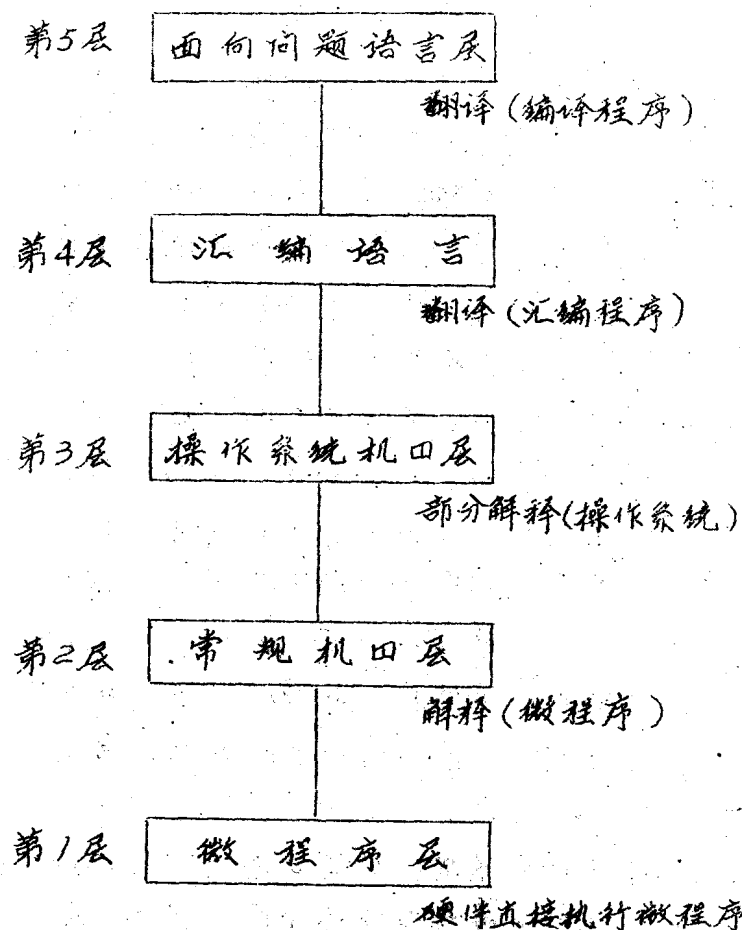


图 1—2 现代计算机的五层结构。每层都由它下面的一层支持，括号内是支持程序的名称。

是程序人员解决问题的平台。程序人员需要的是以解释程序和翻译程序为基础的更高层。解释程序和翻译程序是系统程序人员书写的，它们各具特点并能完成新的虚拟层。程序人员真正用来解决问题的是第四层或更高层的机口。

第四层机口中有方法上的变化、这种方法可以对高层计算机有所协助。虽然也有例外，但一般说来都通过翻译程序来执行，