普通高等教育“十一五”国家级规划教材

国家精品课程主讲教材

高等学校软件工程系列教材

计算系统基础

陈道蓄　主编

王浩然　葛季栋　编著

高等教育出版社
HIGHER EDUCATION PRESS

普通高等教育“十一五”国家级规划教材
国家精品课程主讲教材
高等学校软件工程系列教材

计算系统基础

Jisuan Xitong Jichu

陈道蓄　主编
王浩然　葛季栋　编著

高等教育出版社·北京
HIGHER EDUCATION PRESS　BEIJING

内容提要

本书是国家精品课程“计算系统基础”的主讲教材。该课程作为软件工程专业的第一门专业课程，以一个经典计算机指令集 MIPS 的简化版本 DLX 为线索，选择 C 语言作为载体，采用程序设计与系统级认识双优先的方式，使初学者在对计算机系统有基本了解的基础上，更好地掌握结构化程序设计的基本思想和方法。

本书采用自顶向下和自底向上相结合的方式介绍计算系统，全书分为三部分，共 17 章，其中第一部分为第 1 章~第 5 章，主要介绍程序设计基础，使没有编程基础的学生获取对程序设计的感性认识；第二部分为第 6 章~第 14 章，主要介绍计算机硬件和系统软件的基础知识，并通过介绍一个简单的 DLX 计算机系统的工作原理，来帮助读者理解高级语言程序是如何在计算机系统中运行的；第三部分为第 15 章~第 17 章，主要介绍函数、指针、数组和递归等内容，以及一些复杂程序在 DLX 系统下的运行机制。

本书有与“计算系统基础”课程配套的 DLX 套件。该套件提供了 C 语言（部分非常用功能除外）编译和基本运行环境，能够满足本课程教学的需要。此外，DLX 模拟器提供了较完整的模拟 CPU，有兴趣的学生可以尝试在 DLX 模拟器上设计和实现基于 DLX 指令集的仿真操作系统 DLX OS。

图书在版编目（CIP）数据

计算系统基础/陈道蓄主编，王浩然，葛季栋编著. --北京：高等教育出版社，2013.3

ISBN 978-7-04-036760-7

Ⅰ.①计… Ⅱ.①陈… ②王… ③葛… Ⅲ.①电子计算机-高等学校-教材 Ⅳ.①TP3

中国版本图书馆 CIP 数据核字（2013）第 012019 号

策划编辑	倪文慧	责任编辑	倪文慧	封面设计	于文燕	版式设计	王艳红
插图绘制	尹 莉	责任校对	刘 莉	责任印制	朱学忠		

出版发行	高等教育出版社	咨询电话	400-810-0598
社 址	北京市西城区德外大街 4 号	网 址	http://www.hep.edu.cn
邮政编码	100120		http://www.hep.com.cn
印 刷	北京玥实印刷有限公司	网上订购	http://www.landraco.com
开 本	787mm×1092mm 1/16		http://www.landraco.com.cn
印 张	23.5	版 次	2013 年 3 月第 1 版
字 数	530 千字	印 次	2013 年 3 月第 1 次印刷
购书热线	010-58581118	定 价	36.00 元

物 料 号 36760-00

前　言

对于计算机类专业的初学者，第一门专业课（计算机导论除外）的教学目标是使学生掌握用计算机解决问题的基本方法，并且能用一种高级程序设计语言编写简单的应用程序。实现这一教学目标的方法有两种：一种是暂时避开计算机硬件细节，直接介绍高级语言程序设计的编程技术，使学生能够在很短的时间内编写出可以在计算机上运行的程序；另一种方法是先让学生对计算机的组成和结构有基本的了解，在编程的同时更好地理解计算机软硬件间不可分割的关系，使编写的软件能够更好地发挥硬件的性能，进一步培养学生的基本计算思维能力。后一种方法虽然对培养学生在本科学习的初始阶段就具有计算思维和有效解决问题的能力将会产生积极的作用，但是由于现在的计算机内部结构相对复杂，要求学生在很短的时间内掌握计算机的内部结构，并且学会在其基础上进行程序设计，在教学实践中难以实现。面对这种情况，我们尝试采用虚拟硬件平台来支持这种教学方法。该虚拟硬件平台结构简单，易于展开教学，保留了主流计算机的基本结构和对程序运行的支撑方式，让学生在最初的专业学习中，逐渐养成在编写程序的同时考虑硬件环境影响的习惯。

基于上述思想，我们建设了“计算系统基础”课程，作为软件工程专业的第一门专业课程。该课程以一个经典计算机指令集 MIPS 的简化版本 DLX 为线索，以 C 语言为载体，采用程序设计与系统级认识双优先的方式，使初学者在对计算机系统有基本了解的基础上，能更好地掌握结构化程序设计的基本思想和方法。

本书采用自顶向下和自底向上相结合的方式介绍计算系统，可分为 3 个部分，其中，第一部分包括第 1 章～第 5 章，主要介绍程序设计基础，使没有编程基础的学生对程序设计有感性的认识。第二部分包括第 6 章～第 14 章，主要介绍计算机硬件和系统软件的基础知识，有助于学生理解程序在计算机系统中是如何运行的。为便于初学者理解，本部分以一个简单的 DLX 计算机为例进行讲解。在了解了 DLX 计算机系统的工作原理之后，就可以理解高级语言程序是如何在 DLX 上运行的了。第三部分包括第 15 章～第 17 章，主要介绍函数、指针、数组和递归等深入主题，以及一些复杂程序在 DLX 系统下的运行机制。

通过本课程的学习，学生应该能建立起完整的计算概念，了解经典计算系统的工作原理，理解计算系统自底向上、逐次构造的过程；了解计算机系统的底层机制，包括数据的机器级表示、数字逻辑、冯·诺依曼模型、机器语言、汇编语言、输入和输出、TRAP 机制等；掌握结构化程序设计语言基础，包括变量和运算符、类型、表达式、简单 I/O、控制结构、函数、数组和指针等；理解结构化程序设计，能够利用自顶向下、逐步求精的方法完成小规模的结构化程序；掌握系统地测试小规模程序以及调试程序的技术、方法与工具；了解基本的数据结构与一些简单的算法，包括数组、

栈、链表、递归、顺序查找、二分法查找、冒泡排序等。

“计算系统基础”课程已经在南京大学开设10年，通过在实践中的不断改进，已经取得了预期的效果，并且对软件工程专业后续课程的改革提供了一个良好的基础。

在本课程的教学实施过程中，我们开发了DLX套件。该套件提供了C语言（部分非常用功能除外）编译和基本运行环境，能够满足本课程教学的需要。此外，DLX模拟器提供了较完整的模拟CPU，有兴趣的学生可以尝试在DLX模拟器上设计和实现基于该DLX指令集的仿真操作系统DLX OS。

“计算系统基础”是南京大学软件学院重点建设的课程，并入选2008年国家级精品课程，在教学过程中得到了学院的大力支持，在此表示衷心感谢。在本书撰写过程中得到骆斌教授的指导与帮助，金志权教授、刘海涛老师和宋健建老师对本书的编写也提出了许多极为宝贵的意见，上海交通大学臧斌宇教授审阅了全书并提出了中肯的意见，在此向他们表示衷心的感谢。

特别感谢南京大学软件学院2009级研究生周赵锋、赵科在模拟器设计方面所做的探索工作，2010级研究生吴金虎、马龙、朱文嘉、卢明、袁高龙、刘求索、张鹤在DLX套件方面所做的探索工作，2011级研究生赵奕龙、王诚、韩廷明、徐思豪在C－DLX编译器方面所做的研发工作以及在DLX OS方面所做的探索工作，2008级本科生葛馨阳、葛羽航、胡亮、龚晨、李翔宇、李业、杜昕同学为实现DLX套件所做的工作。

与本书配套的教学网站为http://software.nju.edu.cn/haoranwang，有关本课程的教学课件以及DLX套件可以从该网站下载。限于编者的水平，书中难免存在不妥之处，恳请读者指正和赐教，编者的电子邮件地址为cdx@nju.edu.cn、haoranwang@software.nju.edu.cn和gjd@software.nju.edu.cn。

编者

2012年12月于南京

目　录

第1章 引 言

1.1 本书的目标

本书可作为计算类专业本科生的入门课程教材。通过入门课程的学习，初学者应该对计算有一个系统的认识，并为后续课程的学习打下一个良好的基础。

对于初学者而言，计算机无疑是一个复杂的机器，但是实际上它只是一个由数量巨大的简单元件（晶体管）组成的集合。本书的目标就是向读者介绍计算系统，即如何由简单的元件组成计算机，并使其能执行用计算机语言编写的程序。

学习完本书后，读者可以使用一种高级计算机语言——C 语言编写比较复杂的程序，并能理解这些程序在计算机内部是如何执行的。

1.2 计算机与计算机系统

1. 计算机

前面提到的“计算机”指的是“现代计算机”，全称为“通用电子数字计算机（General Purpose Electronic Digital Computer）”。世界上第一台通用电子数字计算机是 1946 年在美国宾夕法尼亚大学研制成功的 ENIAC（Electronic Numerical Integrator and Calculator，电子数字积分器和计算器）。

（1）通用计算设备

“通用”是指计算机是一种通用计算设备，而不是一种专用设备。在现代计算机出现之前出现过加法器、乘法器等设备，这些设备只能执行专门的计算，如加法、乘法等。而计算机则既可以做加法，也可以做乘法，还可以实现排序或者其他计算。

通用计算设备的思想要归功于英国数学家阿兰·图灵。图灵在 1936 年发表了一篇题为“论可计算数及其在判定问题中的应用”的论文，给出了通用计算设备的数学描述。

基于通用计算设备思想设计出来的现代计算机具有以下两个特点。

① 所有的计算机（无论大小，快慢，昂贵还是便宜），如果给予足够的时间和足够容量的存储器，都可以做相同的计算。换句话说，所有的计算机都能做几乎完全相同的事情，只是在计算速度上有差别。

② 如果想做一种新的计算，不需要对计算机重新进行设计，只需要在计算机上安装合适的

软件，就可以达到目的。只要安装了合适的软件，利用计算机进行计算的领域可以说是非常广阔的，例如安装了拨打电话软件后，就可以利用计算机实现打电话的功能。

（2）电子设备

1642 年，法国数学家帕斯卡发明了世界上第一台真正意义上的计算机。它利用齿轮传动原理制造而成，通过手摇方式操作运算，即手摇机械计算机。随着电子技术的发明与发展，电子元件逐渐演变成为计算机的主体，成为现代计算机硬件实现的物理基础。

计算机是非常复杂的电子设备，计算机执行的计算最终都是通过电子电路中的电流、电位等实现的。

（3）数字设备

“数字”是现代计算机的一种基本特征，也是计算机通用性的一个重要基础。

在现代计算机出现之前，还出现过许多计算机器，大多属于模拟机。模拟机通过测量物理量（如距离或电压）得出计算结果。例如，计算尺通过读取两个标有对数的尺子之间的“距离”计算出乘法结果。模拟机的缺点是很难提高精度，例如，模拟手表通过时针、分针和秒针的移动，根据测量的角度来表示时间。如果要将其精度提高到 0.01 s，可以采用数字手表。因为数字手表使用数字表示时间，只要增加位数就可以提高精度。

在现代计算机里，所有信息都是采用数字化的形式表示的。无论是整数、小数、文字，还是图像、声音等，在计算机里都统一使用数字表示。

（4）计算机

“计算机”意味着它是一种能够做计算的机器。

该机器的核心处理部件是CPU（Central Processing Unit，中央处理器）。处理器完成的最重要的工作就是执行指令，即执行加法、乘法等计算工作。

指令说明了计算机执行的一件明确定义的工作。计算机程序由一组指令组成，指令是计算机程序中规定的可执行的最小工作单位。也就是说，计算机要么执行指令所说明的工作，要么什么都不做。

在早期的机器中，处理器执行的程序并不是放在机器中的，程序通常表现为一叠打了孔的卡片。机器在工作时依靠一台读卡机读取卡片，再由处理器完成卡片上程序所要求的工作。采用这种方式，受读卡机工作速度的限制，计算速度较慢。

此后，美籍匈牙利科学家冯·诺依曼提出了“存储程序控制原理”思想，现代计算机就是按照其思想构建出来的。存储程序控制计算机的核心部件除 CPU 外，还有存储器（内存）。程序存储在存储器里，CPU 负责完成两项工作：指挥信息的处理和执行信息的实际处理。“指挥信息的处理”，是指从存储器里读取下一条指令；而“执行信息的实际处理”，则是执行该指令，即进行加法、乘法等计算工作。这两项工作循环进行，即读取指令，执行指令，读取指令，执行指令……

目前各类计算机的 CPU 都是采用半导体集成电路技术制造的。其基础材料为硅片，通过复杂的工艺，在只有指甲般大小的硅片上集成了数以亿计的晶体管，称之为“微处理器”。

2. 计算机系统

当人们提到“计算机”时，往往不单指处理器、存储器等部分，还包括外部设备，如输入命令的键盘，点击菜单的鼠标，显示计算机系统生成的信息的显示器，把信息输出为纸质品的打印机，保存信息的磁盘和 CD-ROM 等，此外还有用户希望执行的程序，如操作系统 UNIX 或 Windows，数据库系统 Oracle 或 DB2，应用程序 Microsoft Office 或 Oracle Open Office。

以上集合就构成了“计算机系统”。也就是说，计算机系统由硬件和软件两部分组成，硬件包括处理器、存储器和外部设备等，软件包括程序和文档。

硬件和软件是计算机系统的两个组成部分，在设计硬件或软件时，如果能够同时考虑两者的能力和限制，可使计算机系统达到最佳工作状态。

如果微处理器设计者理解了将要在其设计的微处理器上执行的程序的需求，就可以设计出更高效的微处理器。例如，Intel、Motorola 和其他主要的微处理器厂商认识到未来的很多应用都会将视频作为 E - mail、计算机游戏、电影的一部分，对于那些应用来说，高效执行非常重要。为此，Intel 定义了称为 MMX 的指令集，为其开发了特殊的硬件。Motorola 则定义了 AltiVec 指令集并开发出相应的硬件。

这一点对于软件设计者也是一样的。如果软件设计者理解了执行程序的硬件的能力和限制，也可以设计出性能更高的程序。例如，要编写一款有竞争力的游戏软件，软件设计者就需要了解处理器的并行化特征。

1.3 计 算 系 统

人类使用自然语言（即人类所讲的语言）来描述问题，而计算机则使用流动的电子解决问题。因此，必须将人类的自然语言转换成计算机能识别的指令，进而转换为能够影响电子流动的电压，才能使计算机完成复杂的任务。这种转换是一种有序的、系统的转换。

图 1.1 显示了在计算系统中使用电子解决问题的一个过程，可以将其表示为 7 个抽象层次（其中，程序层又包括两个抽象层次）。抽象层次是硬件和软件设计者在解决问题时使用的一种方法，每一层对它的上一层隐藏了自己的技术细节。

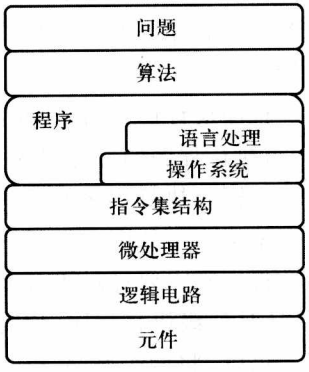

图 1.1 计算系统的抽象层次

本书的目标就是向读者解释这个系统，本章首先按照自顶向下的顺序进行简单的介绍。对于本章出现的术语和概念，并不需要读者完全理解，可以在读完本书后，再来阅读这些内容。

1. 问题

人类使用自然语言来描述那些希望计算机解决的问题。自然语言通俗易懂，但是不能直接作为计算机的指令，最重要的原因就是它具有“歧义性”：为了确定一句话的含义，听者通常需要根据说话人的发音、语调，语句的上下文来进行判断。例如，“羽

毛球拍卖完了。”可以理解为没有羽毛球拍了，被卖完了；也可以理解为没有羽毛球了，被拍卖完了。

如果把这种有歧义的自然语言作为指令提供给计算机，计算机是无法执行的，本质上，计算机仍是电子设备，它只能机械地执行明确的指令，如“Add A，B”是将两个数 A 和 B 相加。面对有歧义的自然语言，计算机将无所适从。

2. 算法

因此，第一步就是舍弃描述问题的自然语言中的歧义，将用自然语言描述的问题转换成一个无歧义的操作步骤，即算法。算法是一个逐步计算的过程，该过程一定能够结束，而且每个步骤都能够被明确描述，并能被计算机所执行。以上这 3 个特点可用专门的术语说明。

有限性(Finiteness)：计算过程最终能够结束。

确定性(Definiteness)：每个步骤都必须是明确的，不应存在歧义性。例如，“A 与一个数相加”就是“不确定”的，因为不知道 A 与哪一个数相加。

有效可计算性(Effective Computability)：每个步骤都能被计算机执行。例如，“A 除以 0”就缺乏可计算性。

要解决一个问题通常可以采用多种算法，有的算法可能需要较少的计算时间；有的算法可能需要较少的存储空间。算法分析就是对一个算法需要多少计算时间和存储空间进行定量的分析。

3. 程序

第二步是使用一种程序设计语言把算法转换为程序。程序设计语言与自然语言不同，它不是在人类的交谈中演化形成的。相反地，程序设计语言是用于表达计算机指令的语言，不能存在歧义。

迄今为止，人类为满足各种需要而发明的程序设计语言可达千种之多，例如，FORTRAN 语言主要用于科学计算，COBOL 语言可用于进行商业数据处理，Prolog 语言可用于设计专家系统，LISP 语言可用于研究人工智能问题，Pascal 语言则适用于程序设计初学者学习如何编程，等等。C 语言是为了开发系统软件(如操作系统和编译器)而发明的，本书选择 C 语言进行程序设计。

程序设计语言可以分为高级语言与低级语言两个级别。高级语言和底层计算机有一定的距离，与执行程序的计算机无关，称之为“独立于机器”。上面提到的这些语言都是高级语言。低级语言则与执行程序的计算机紧密相关，基本上每种计算机都有自己的低级语言——机器语言和汇编语言。如图 1.1 所示，不需要进行语言处理的程序为机器语言程序，汇编语言程序及高级语言程序则需要经过语言处理，翻译为机器语言程序后才能执行；大部分程序，无论高级语言程序，还是低级语言程序，均需要操作系统的支持。

仍以“将两个数 A 和 B 相加”为例，C 语言可以表示为 A + B；而用某种机器的汇编语言表示，可以为“Add A，B”，其机器语言指令则为 0001001001000000。

4. 语言处理

对于使用高级语言和汇编语言编写的程序而言，如果要在计算机上执行，必须将其翻译成执行程序作业的机器（目标机器）的指令，即机器语言。

因此，语言处理又可分为两层：高级语言处理和低级语言处理，如图 1.2 所示。高级语言处理是指将高级语言翻译为低级语言的过程；而低级语言处理是指将汇编语言翻译成机器语言的过程。这个翻译工作通常可以由翻译程序来完成。高级语言翻译程序称为编译器（又称编译程序）或解释器（又称解释程序），低级语言翻译程序称为汇编器（又称汇编程序）。

5. 操作系统

如何把编写的程序输入计算机中？如何把计算机执行的结果输出给用户？最初的操作系统包含的就是支持输入/输出（Input/Output，I/O）操作的设备管理例程。随着技术的发展，操作系统又扩展实现了文件管理、内存管理、进程管理等主要功能。

在计算机的发展过程中出现过许多操作系统，例如 DOS、Mac OS、Windows、UNIX、Linux、OS/2 等。

本书只介绍操作系统的 I/O 设备管理例程，在此又可分为两层：I/O 例程和系统调用，如图 1.3 所示。

图 1.2　语言处理层

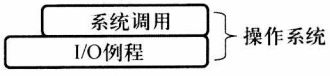

图 1.3　操作系统层

6. 指令集结构

将高级语言程序翻译成某种低级的机器语言，其依据就是目标机器的指令集结构（Instruction Set Architecture，ISA）。

指令集结构是编写的程序和执行程序的底层计算机硬件之间的接口的完整定义。指令集结构指明了计算机能够执行的指令的集合，也就是说计算机能够执行的操作和每一步操作所需的数据。所需的数据称为“操作数”，操作数在计算机中的表示方式称为“数据类型”，而操作数位于存储器的什么地方则被称为“寻址模式”。除了规定操作数的数据类型和寻址模式外，指令集结构还规定了计算机存储单元的数量，以及每个存储单元存储信息的能力。

不同的指令集结构所规定的操作数、数据类型和寻址模式是不一样的。有些指令集结构有数百种操作，而有些则只有十来种。有些指令集结构只有一种数据类型，而有些却多达十几种。有的指令集结构只有一两种寻址模式，而有些却有 20 多种。1985 年 Intel 公司设计的 IA－32 指令集结构至今仍在广泛使用，它有 100 多种操作，10 多种数据类型和 20 多种寻址模式。1986 年发布的 MIPS 也是一种典型的指令集结构，与 IA－32 相比，它规定的操作、数据类型和寻址模式要少得多。

此外，其他的指令集结构还有 PowerPC（IBM 和 Motorola）、Alpha（COPMPAQ）、PA-RISC（HP）、SPARC（SUN 和 HAL）以及最新的 IA－64（Intel）等。

如果需要将某种高级语言(如 C 语言)翻译后在某种目标机器(如 IA - 32)上执行,则必须使用相应的翻译程序。

7. 微处理器

接下来就是把指令集结构实现为微处理器。每一种指令集结构都可以采用多种微结构来实现,对于计算机设计者来说,每一种实现都是一次对微处理器的成本和性能的平衡。计算机设计就是平衡成本与性能的一种选择,使用更高(低)的成本,计算机就有更好(差)的性能表现。

以 IA - 32 指令集结构为例,从 1985 年 Intel 实现的 80386 微处理器,之后的 80486、80586 微处理器,到 1998 年推出的 Pentium(奔腾)微处理器,都是采用不同微结构对 IA - 32 指令集结构的实现。

MIPS 指令集结构由 Cisco、Nintendo、Sony 和 SGI 等公司生产,实现了不同的微处理器,用于 Sony、Nintendo 的游戏机,Cisco 的路由器和 SGI 超级计算机中。

8. 逻辑电路

下一步就是组成微处理器的每一个组件的逻辑电路实现。同样也有很多选择,因为设计者也需要考虑如何尽量平衡成本和性能。例如,对于组成微处理器的加法器的实现,选择超前进位逻辑电路比选择行波进位电路计算速度更快。

9. 元件

最终,每一种基本的逻辑电路都是由特定的物理元件实现的。例如,CMOS(Complementary Metal-Oxide-Semiconductor,互补金属氧化物半导体)逻辑电路采用金属氧化物半导体晶体管制造,双极型逻辑电路则采用双极型晶体管构成。

1.4 本书的结构

本书采用自顶向下和自底向上相结合的方式描述了计算系统的"逐层转换"过程,可分为 3 个部分。

第一部分(第 2 章 ~ 第 5 章)描述从问题到算法,从算法到 C 语言程序的转换。

这一部分只介绍一些程序设计语言最基本的主题:C 语言程序设计简介(第 2 章)、类型和变量(第 3 章)、结构化程序设计和控制结构(第 4 章)、测试和调试(第 5 章),使读者对程序设计有一个感性认识。要深入了解 C 语言相关理论,还必须从计算机的组织结构入手,即了解计算机是如何组成的。

第二部分(第 6 章 ~ 第 14 章)描述从晶体管到逻辑电路,从逻辑电路到处理器,从处理器到指令集结构,从指令集结构到程序的转换。

这一部分主要介绍计算机的组成。首先介绍数据在计算机中是如何表示以及如何运算的(第 6 章),在此基础上说明如何使用晶体管组成可以运算的组件(如加法器)、可以存储信息的组件等逻辑单元(第 7 章),之后即可以使用经典的冯·诺依曼模型(第 8 章)将这些组件组合成

计算机(第9章)。第9章介绍的是一个MIPS指令集的简化版本DLX,DLX不是一个真实的计算机,但是其设计方法覆盖了现代真实机器实现的一般方法,而且由于其特性简单,容易被初学者理解。

本书介绍的指令集基于美国加州大学伯克利分校计算机系Patterson教授和斯坦福大学计算机系Hennessy教授在《计算机系统结构:一种定量的方法(第二版)》一书中给出的DLX指令集,并根据需要对其进行了裁减和扩充。

在了解了DLX是如何工作之后,就可以在它上面进行编程了,首先使用它自身的语言——机器语言(第10章),然后是汇编语言——一种更易于理解的语言(第11章)。第11章还以DLX汇编语言和C语言为例介绍了语言处理方法,内容包括如何将DLX汇编语言程序翻译为DLX机器语言,以及将C语言程序翻译为DLX汇编语言,为了完成C语言的翻译工作,引入一个非常重要的概念——栈。

为了处理信息进出计算机的问题,在第12、13章分别介绍DLX操作系统的I/O设备管理例程。第12章介绍I/O例程,第13章介绍两个机制:自陷(TRAP)机制,即系统调用,和一个比较复杂的机制——中断驱动的I/O,以及C语言的I/O。第14章介绍DLX的子例程。

第三部分(第15章~第17章)则是C语言的深入主题,包括函数(第15章)、指针和数组(第16章)以及递归(第17章)。在这一部分将C语言程序和底层的DLX联系起来,以便读者深入理解当在C语言程序中使用某一种结构时,在底层的计算机中是如何实现的。

通过"逐层转换",读者不仅可以用C语言程序解决问题,还能够理解C语言程序是如何通过电子的流动实现的。

在解决实际问题时,不必陷入"逐层转换"的细节之中。例如,当设计一个复杂的计算机应用程序时,如设计一个文字处理软件时,不需要考虑所有抽象层次的每一处细节,而应该集中考虑问题的本质,更多地关注从问题到算法、从算法到程序的转换。再如,在使用门电路设计逻辑电路时,也不必考虑每个门电路的内部结构。为了更好地解决问题,必须了解"逐层转换"的过程。

习 题 1

1.1 名词解释:计算、计算机、计算机系统和计算系统。

1.2 有两台计算机A和B,A有乘法指令,而B没有;两者都有加法和减法指令;在其他方面两者都相同。那么,在A和B中,哪台计算机可以解决更多的问题?

1.3 现代计算机为什么采用数字方式设计,而非模拟机?

1.4 现代计算机的核心部件有哪些?分别具有什么功能?

1.5 为什么自然语言不能直接作为计算机指令?举例说明。

1.6 给出如下问题的算法。

(1) 计算1+2+3+4+5+6+7+8+9+10

(2) 判定2010—2500年中的某一年是否为闰年

(3) 对一个大于或等于3的正整数,判断它是否为素数(质数)

1.7 当将计算机升级(如更换CPU)后,原来的软件(如操作系统)还能够使用吗?

1.8 人们购买的软件通常是以什么语言编写的?是高级语言还是与目标机器ISA兼容的机器语言?

1.9 关于计算系统的每个抽象层次,分别举出两个以上的例子进行说明。

1.10 你对计算系统哪一部分比较熟悉?简要说明你对该部分的理解。