DIANZIKEJIDAXUECHUBANSHE
XILIEJIAOCAI

95

高等学校
电子信息类
系列教材

本科计算机

8086/8088系列微型计算机

宏汇编语言程序设计

王正智　苟大举
徐　洁　黄海于　编著

电子科技大学出版社

UESTC PUBLISHING HOUSE

8086/8088 系列微型计算机

宏汇编语言程序设计

王正智　苟大举
徐　洁　黄海于　编著

电子科技大学出版社

内 容 提 要

本书系统地介绍了汇编语言程序设计的基础知识、基本原理和程序设计方法与技术。全书共分十三章：第一章介绍进行汇编语言程序设计所必需的基础知识；第二、三章介绍Intel 8086/8088微处理器的功能结构、寻址方式和指令系统；第四、九章分别介绍MASM的基本汇编语言和高级宏汇编语言的各种伪指令；第五章讲述了顺序、分支、循环三种程序结构的程序设计方法与技术；第六章讲述了子程序结构的程序设计方法与技术；第七、八章分别介绍在数值运算和非数值运算中汇编语言程序设计方法与实例；第十章介绍输入/输出汇编语言程序设计；第十一、十二章介绍80386、80486和Pentium微处理器硬件结构、功能、指令系统及其汇编程序设计的特点；第十三章介绍汇编语言程序的开发过程及操作。

全书结构清晰，由浅入深，循序渐进，讲述详细，基础知识的讲述与程序设计方法、技巧并重，程序实例丰富。各章均附有习题。

本书可作为高等院校计算机专业的教材，也可作为教师、非计算机专业的研究生、本专科生和从事软件工程设计的技术人员的参考书。

声　明

高等学校
电子信息类　系列教材
8086/8088系列微型计算机

宏汇编语言程序设计

王正智　苟大举
徐　洁　黄海于　编著

出　　版：电子科技大学出版社　（成都建设北路二段四号　出编 610054）
责任编辑：吴艳玲
发　　行：电子科技大学出版社
印　　刷：四川导向印务有限公司
开　　本：787mm×1092mm　1/16　印张 24.875　字数 605 千字
版　　次：2000 年 5 月第一版
印　　次：2000 年 5 月第一次
书　　号：ISBN 7—81065—365—2/TP・236
印　　数：1—4000 册
定　　价：27.00 元

前　言

《汇编语言程序设计》课是普通高等院校计算机专业学生必修的核心课程之一。本课程的教学，将对学生进行程序设计的基本训练，学习程序设计的基本方法与技术。通过汇编语言程序的编制与调试，从根本上认识、理解计算机是如何实现各种复杂运算与操作的。这对学生后续的计算机专业课程的学习是非常有帮助的。

汇编语言是一种介于计算机能直接识别、执行的机器语言和高级语言之间的程序设计语言。汇编语言的主要特征是它同具体的机器结构联系紧密，汇编语言程序和机器语言程序（即目标代码）有一一的对应关系，能充分发挥计算机所有硬件的功能与特点。因此，在学习汇编语言前就需要较仔细的了解汇编语言程序设计的硬件环境（即背景机器结构）。本书是以 Intel 8086/8088 系列微型机为机器背景，介绍其汇编语言程序设计。虽然现在普遍使用的是用 Intel 80386、80486 和 Pentium 微处理器构成的微型机，但是就汇编语言程序设计而言，基于 8086/8088 微处理器的汇编语言程序设计是基础，只要有这样的基础，扩展到 80386、80486 和 Pentium 微处理器进行汇编语言程序设计是轻而易举的事。所以本书用较多的篇幅介绍 8086/8088 汇编语言程序设计，在此基础上最后用两章（第十一、第十二章）专门介绍扩展到 83086、80486 和 Peutium 微处理器的汇编语言程序设计。这种扩展对有些读者可能是不够的，但是对于汇编语言程序设计的初学者是非常有益的，可以达到入门和引导的效果。

本书第一章是进行汇编语言程序设计的必备基础知识。第二章介绍 Intel 8086/8088 微型机及其 CPU 的结构、功能。第三章介绍 8086/8088 CPU 的寻址方式和指令系统。第二、三章就是汇编语言程序的硬件环境，也是后面各章的基础。有关 MASM 汇编语言分成基本汇编语言和高级宏汇编语言两部分（分别在第四章和第九章介绍）。这是为了尽快让初学读者进入程序设计的“角色”，在第四章先讲述基本汇编语言，而把高级宏汇编语言安排在读者已有初步汇编语言程序设计知识和技术之后，这样也便于读者对第九章高级宏汇编语言的理解与掌握。第五章和第六章是讲述汇编语言程序设计的基本技术与方法。第七章和第八章是介绍在数值、运算与非数值运算中汇编语言程序设计的方法与技巧。第十章是介绍输入/输出汇编语言程序设计。第十一、十二章是介绍 80386、80486 和 Pentium 微处理器的硬件结构、功能、指令系统及其汇编语言程序设计的特点。这两章内容必须在前面各章学习的基础上进行教与学。第十三章是介绍开发一个汇编语言程序设计的必备知识和操作。这章内容可适当提前学习，以便及早上机调试、运行程序。

本书 4 位编者近几年乃至 10 多年从事《汇编语言程序设计》的教学工作，根据日常教学中收集的一些资料和点滴的教学体会编写了本书。书中各章例举的程序实例均在机上调试通过。

本书第一章由电子科技大学王正智编写；第二、三、四、五、九章由电子科技大学徐

洁编写；第六、十一、十二、十三章由四川大学苟大举编写；第七、八、十章由西南交通大学黄海于编写。全书由王正智主编、统稿、定稿。在本书的编写、出版工作中，自始至终得到电子科技大学出版社副总编吴艳玲的支持和帮助，在此深表谢意。

由于编者水平有限，本书的缺点与错误在所难免，敬请读者批评指正。

编　者

2000 年 2 月于成都

目　录

第一章　基础知识 …… (1)

§1.1　汇编语言程序设计的一般概念 …… (1)
§1.2　计算机中数据信息的表示 …… (4)
§1.2.1　进位计数制及其相互转换 …… (4)
§1.2.2　带符号数的表示 …… (8)
§1.2.3　字符的表示 …… (13)
§1.3　基本逻辑运算 …… (13)
习题 …… (14)

第二章　IBM PC 微型计算机 …… (16)

§2.1　IBM PC 微型计算机基本结构 …… (16)
§2.1.1　微型机硬件系统组成 …… (16)
§2.1.2　Intel 8086/8088 微处理器功能结构 …… (17)
§2.2　8086/8088 CPU 寄存器结构 …… (19)
§2.2.1　段寄存器 (Segment Register) …… (19)
§2.2.2　通用寄存器 (General Register) …… (20)
§2.2.3　用于控制的寄存器 …… (22)
§2.3　主存储器 …… (24)
§2.3.1　IBM PC 微机主存储器的特点 …… (24)
§2.3.2　主存储器的段结构 …… (25)
§2.3.3　逻辑地址与物理地址 …… (26)
§2.4　堆栈 …… (27)
§2.4.1　堆栈的构造 …… (28)
§2.4.2　8086/8088 的堆栈组织 …… (28)
习题 …… (30)

第三章　8086/8088 指令系统与寻址方式 …… (32)

§3.1　寻址方式 …… (32)
§3.1.1　寄存器寻址 (Register Addressing) …… (32)
§3.1.2　立即数寻址 (Immediate Addressing) …… (33)
§3.1.3　存储器寻址 …… (33)
§3.2　指令系统 …… (39)
§3.2.1　传送类指令 …… (40)

§3.2.2 算术运算类指令……（43）
§3.2.3 位操作类指令……（47）
§3.2.4 处理器控制指令……（51）
§3.3 机器指令格式……（52）
§3.3.1 双操作数机器指令代码格式……（52）
§3.3.2 单操作数机器指令格式……（56）
§3.3.3 与 AX、AL 有关的机器指令格式 ……（57）
§3.3.4 单字节机器指令格式……（57）
习题 ……（58）

第四章 基本汇编语言 ……（62）

§4.1 汇编语言语句格式……（62）
§4.1.1 指令语句格式……（62）
§4.1.2 伪指令语句格式……（63）
§4.1.3 标识符……（64）
§4.2 汇编语言数据……（64）
§4.2.1 常数……（64）
§4.2.2 变量……（65）
§4.2.3 标号……（69）
§4.3 符号定义伪指令 EQU 和= ……（71）
§4.3.1 等值语句……（71）
§4.3.2 等号语句……（72）
§4.4 表达式与运算符……（72）
§4.4.1 算术运算符……（73）
§4.4.2 逻辑运算符……（74）
§4.4.3 关系运算符……（74）
§4.4.4 数值返回运算符……（75）
§4.4.5 属性与分离字节运算符……（77）
§4.4.6 运算符的优先级……（79）
§4.5 程序的段结构……（80）
§4.5.1 段定义伪指令 SEGMENT/ENDS ……（80）
§4.5.2 段寻址伪指令 ASSUME ……（82）
§4.5.3 段寄存器的装入……（83）
§4.6 过程定义伪指令 PROC/ENDP ……（85）
§4.7 源程序的基本结构框架……（86）
§4.8 其他伪指令……（87）
§4.8.1 定位伪指令 ORG 和位置计数器 ……（87）
§4.8.2 标题伪指令 TITLE ……（88）
习题 ……（88）

第五章　顺序、分支与循环程序设计 …… (93)

§5.1　概述 …… (93)
§5.1.1　汇编语言程序的设计步骤 …… (93)
§5.1.2　程序的基本结构 …… (94)
§5.2　顺序结构的程序设计 …… (94)
§5.3　分支程序设计 …… (96)
§5.3.1　转移指令 …… (97)
§5.3.2　条件转移指令 …… (99)
§5.3.3　分支程序设计 …… (102)
§5.4　循环程序设计 …… (108)
§5.4.1　循环控制指令 …… (108)
§5.4.2　循环程序的结构 …… (112)
§5.4.3　单重循环程序设计 …… (113)
§5.4.4　多重循环程序设计 …… (115)
习题 …… (117)

第六章　子程序设计 …… (122)

§6.1　子程序的定义 …… (122)
§6.1.1　子程序的定义 …… (122)
§6.1.2　子程序的调用与返回 …… (125)
§6.2　子程序的设计要求 …… (127)
§6.3　子程序与主程序间的参数传递和设计举例 …… (131)
§6.3.1　用寄存器传递参数 …… (131)
§6.3.2　用堆栈传递参数 …… (133)
§6.3.3　用地址表传递参数 …… (135)
§6.4　子程序的嵌套与递归调用 …… (137)
§6.4.1　子程序的嵌套调用 …… (137)
§6.4.2　子程序的递归调用 …… (137)
§6.5　多模块程序设计 …… (139)
习题 …… (141)

第七章　数值运算程序设计 …… (143)

§7.1　加减法运算 …… (143)
§7.1.1　二进制数加减运算 …… (143)
§7.1.2　十进制数加减运算 …… (145)
§7.2　乘除法运算 …… (150)
§7.2.1　二进制数乘除运算 …… (150)
§7.2.2　十进制数乘除法运算 …… (152)

§7.3 多精度数运算……（156）
习题 ……（159）

第八章 非数值运算程序设计 ……（160）

§8.1 串操作……（160）
§8.1.1 串操作指令……（160）
§8.1.2 串操作指令应用举例……（166）
§8.2 代码转换……（169）
§8.2.1 二进制数与十进制数 BCD 码之间的转换 ……（170）
§8.2.2 二、十、十六进制数与 ASCII 码之间的相互转换 ……（179）
§8.3 排序与查找……（187）
§8.3.1 气泡排序算法及其程序举例……（187）
§8.3.2 二分法查找算法及其程序举例……（190）
习题 ……（193）

第九章 高级宏汇编语言 ……（194）

§9.1 结构与记录……（194）
§9.1.1 结构……（194）
§9.1.2 记录……（198）
§9.2 宏指令……（201）
§9.2.1 宏指令的使用过程……（201）
§9.2.2 宏操作符……（204）
§9.2.3 局部符号伪指令 LOCAL ……（206）
§9.2.4 宏库……（207）
§9.3 重复汇编……（209）
§9.3.1 定重复伪指令 REPT/ENDM ……（209）
§9.3.2 不定重复伪指令 IRP/ENDM ……（209）
§9.3.3 不定重复字符伪指令 IRPC/ENDM ……（210）
§9.4 条件汇编……（211）
习题 ……（214）

第十章 输入/输出程序设计……（217）

§10.1 输入/输出指令……（217）
§10.1.1 I/O 端口编址方式 ……（217）
§10.1.2 输入/输出指令……（217）
§10.1.3 I/O 端口寻址方式 ……（218）
§10.2 输入/输出控制方式……（218）
§10.2.1 程序控制方式 ……（218）
§10.2.2 中断处理方式 ……（219）

§10.2.3 直接存储器存取方式 …… (219)
§10.3 中断 …… (220)
§10.3.1 中断的一般概念 …… (220)
§10.3.2 中断源及中断类型码 …… (221)
§10.3.3 中断矢量表 …… (222)
§10.3.4 中断优先级 …… (223)
§10.3.5 中断过程 …… (223)
§10.4 DOS 系统功能调用 …… (225)
§10.4.1 DOS 操作系统简介 …… (226)
§10.4.2 DOS 中断功能调用 …… (228)
§10.5 BIOS 中断调用 …… (248)
§10.5.1 键盘中断（INT 16H） …… (249)
§10.5.2 显示中断（INT 10H） …… (250)
§10.5.3 串行通信中断（INT 14H） …… (260)
§10.5.4 磁盘文件存取中断（INT 13H） …… (263)
习题 …… (266)

第十一章 80286、80386、80486 和 Pentium 微处理器结构 …… (267)

§11.1 8086 系列微处理器及其工作模式 …… (267)
§11.2 8086 系列微处理器简介 …… (274)
§11.3 8086 系列微处理器中的寄存器 …… (279)
习题 …… (284)

第十二章 80286、80386、80486 和 Pentium 指令及编程应用基础 …… (285)

§12.1 80286、80386、80486 和 Pentium 微处理器指令的特点 …… (285)
§12.1.1 微处理器的工作模式与指令 …… (285)
§12.1.2 指令中的操作数 …… (286)
§12.1.3 几条指定处理器及工作模式的伪指令 …… (288)
§12.2 80286、80386、80486 和 Pentium 微处理器的指令 …… (289)
§12.2.1 数据传送类指令 …… (289)
§12.2.2 算术运算类指令 …… (290)
§12.2.3 位操作类指令 …… (291)
§12.2.4 比较类指令 …… (292)
§12.2.5 串操作类指令 …… (292)
§12.2.6 逻辑运算类指令 …… (293)
§12.2.7 堆栈操作类指令 …… (294)
§12.2.8 条件设置和控制转移类指令 …… (297)
§12.2.9 类型转换类指令 …… (298)
§12.2.10 I/O 类指令 …… (298)

§12.2.11 特权类指令 …… (299)
§12.3 程序设计举例 …… (301)
习题 …… (305)

第十三章 汇编语言程序的开发 …… (306)

§13.1 汇编语言程序的开发过程 …… (306)
§13.2 编辑 …… (308)
§13.3 汇编 …… (309)
§13.3.1 MASM 的操作 …… (309)
§13.3.2 汇编状态信息和错误代码 …… (310)
§13.3.3 目标文件及列表文件示例说明 …… (311)
§13.4 连接 …… (314)
§13.4.1 LINK 的操作 …… (315)
§13.4.2 内存映像文件（.MAP） …… (315)
§13.5 调试与运行 …… (316)
§13.5.1 DEBUG 使用基础 …… (316)
§13.5.2 DEBUG 的状态进入与退出 …… (317)
§13.5.3 DEBUG 常用命令 …… (317)
§13.5.4 程序调试初步 …… (324)
§13.6 COM 文件格式的汇编程序 …… (326)

附录一 ASCII 码字符表 …… (328)
附录二 8086/8088 系列微处理器指令系统汇总表 …… (330)
附录三 DOS 系统功能调用（INT 21H） …… (366)
附录四 BIOS 中断调用 …… (372)
附录五 出错信息 …… (378)

参考文献 …… (388)

第一章 基础知识

汇编语言是一种面向机器结构的程序设计语言。在学习汇编语言及其程序设计之前，首先介绍汇编语言程序的一般概念。然后介绍涉及汇编语言的一些基础知识：计算机中数据信息的表示、数制及其数制间的转换、带符号数的表示方法、字符信息的表示和逻辑代数的基本运算。

§1.1 汇编语言程序设计的一般概念

现代电子数字计算机，几十年来发生了许多演变，但是采用冯·诺依曼存储程序的工作方式的核心没有变化。这种工作方式的基本点是事先编制程序，把程序存储在计算机的存储器中，然后由计算机自动地、连续地运行程序，以实现计算机的快速运算。这里所述的“程序”是指以机器指令序列组成的机器语言程序，机器指令就是一组由0和1构成的二进制代码。一台计算机的全部机器指令就是我们常说的计算机的指令系统。指令系统中每一条指令表示了计算机的一个操作功能（如加法、减法、乘法、除法、移位、计数等），所以指令系统反映了这台计算机的基本功能。用计算机求解一个实际问题所运行的程序，无论这个程序多么复杂、庞大，它都是由指令系统中提供的机器指令组成，这样的程序叫机器语言程序。

计算机应用的用户直接使用指令系统中的机器指令编制机器语言程序是非常艰难的事，且极易出错。现在用户通常是选用一种高级程序设计语言来编制程序。使用高级程序设计语言编制的程序叫源程序，源程序中各语句是由各种字符（如字母、数字、各种符号等）构成的。像这样的源程序是不能由计算机直接运行的，必须通过语言处理程序进行解释或编译出功能相同的目标程序（即机器语言程序），然后再由计算机运行目标程序，完成程序的各项功能。由语言处理程序将源程序翻译（解释或编译）出来的目标程序比直接用机器指令编制的机器语言程序要长（有时长很多）。因此通过翻译得到的目标程序占用较多的存储空间，当然计算机运行这目标程序也需花费较多时间。但是，汇编语言程序是符号化的机器语言程序，也就是说汇编语言程序中每一条指令语句都与机器语言程序的每一条机器指令一一对应。用汇编语言编制的程序与用机器指令编制的机器语言程序的效果是一样的，这里所说的效果是指目标代码的长度和程序运行的时间。但是，像高级语言程序一样，汇编语言程序也是用各种字符表达的，因此，这样的程序在编写、阅读时都较方便。

鉴于汇编语言是与具体计算机硬件系统密切相关。通常是以某系列计算机为背景，进行汇编语言程序设计。本书选定现在国内外流行的、应用极为广泛的Intel 8086/8088系列微型计算机。例如要完成S=(A+B)(C−D)的运算，用8086/8088系列微机MASM宏汇编语言编制一源程序如下：

```
;设置数据段
DATA    SEGMENT
  A     DW      23H             ;数据 A
  B     DW      14H             ;数据 B
  C     DW      43H             ;数据 C
  D     DW      3DH             ;数据 D
  S     DW      0               ;存放结果单元
DATA    ENDS
;设置堆栈段
STACK1  SEGMENT PARA STACK
        DW      20H  DUP(0)
STACK1  ENDS
;设置代码段
CODE    SEGMENT
        ASSUME CS:CODE, DS:DATA, SS:STACK1
START:  MOV     AX,DATA         ;设置段寄存器 DS
        MOV     DS,AX
        MOV     BX,A            ;取数据 A
        ADD     BX,B            ;计算(A+B)
        MOV     AX,C            ;取数据 C
        SUB     AX,D            ;计算(C-D)
        MUL     BX              ;完成乘法运算
        MOV     S,AX            ;存放乘积
        HLT
CODE    ENDS
        END     START
```

用汇编语言编制的程序叫汇编语言源程序，这个源程序仍然不能由计算机直接执行，亦必须经过“汇编”（即翻译），转换成机器语言程序，再由计算机执行。上述汇编语言程序经过汇编后得到的机器代码（即机器语言程序），如图 1-1 所示。将汇编语言源程序“汇编”为机器语言程序是由汇编程序完成的。本书所介绍的汇编程序是在 DOS 操作系统支持下的 MASM.EXE 宏汇编程序。

由于每种计算机有自己机器的指令系统，相应的有自己的机器语言和汇编语言。为了学习、使用某种计算机的汇编语言，就必须首先熟悉那种计算机的内部结构与功能。一台计算机能执行的机器语言程序，主要决定于组成这台计算机的中央处理器 CPU。因此，我们需要了解并熟悉计算机的内部结构主要是指 CPU 的功能结构：有多少个寄存器，各有什么用途；CPU 是如何访问存储器（例如如何形成存储单元地址的）；在进行输入/输出操作时，有些什么工作方式等。本书是选用 Intel 80x86 系列微处理器为硬件背景，学习该系列微机的汇编语言程序设计，基础是 8086/8088 微处理器的汇编语言，有了此基础，学习 80286、80386、80486……微处理器的汇编语言程序设计就非常容易。

对于汇编语言程序设计的初学者应该知道，通过一个具体的计算机学习并掌握它的汇编语言程序设计的基本原理、方法与技巧，对于今后在其他计算机上进行汇编语言程序设计是完全有益的。

用面向机器指令的汇编语言来编写程序，在程序中，可直接调用计算机内的某些部件，如寄存器、标志位等，更好地利用计算机的某些特性来编制程序。所以，用汇编语言编制的程序不仅占有存储空间少，运行速度快，更可以充分发挥机器硬件功能。但是由于汇编语言是面向机器结构的，不同机器的指令功能与指令代码不同，因此用不同系列计算机的汇编语言编制的程序是不可移植的。既然如此，为什么至今还要学习和使用汇编语言编制程序呢?

1. 学习与使用汇编语言可以从根本上认识、理解计算机的工作原理。因为一台计算机完成一个任务，归根到底是执行一个计算机的机器指令序列，这些指令序列可以通过汇编语言程序反映出来。所以通过用汇编语言编制程序，可以更清楚地了解计算机是怎样完成各种复杂的工作。在此基础上，程序设计人员更能充分地利用机器硬件的全部功能，发挥计算机的长处。

2. 现在的计算机系统中，某些功能仍然是靠汇编语言程序来实现的。例如机器自检、系统初始化、实际的输入/输出设备操作，至今仍是用汇编语言编制的程序来完成的。

3. 在节省内存空间和提高程序运行速度是重要指标的应用场合（如实时控制系统中），常常是用汇编语言编制程序。现在许多高级语言都设置有与汇编语言程序的接口功能，以便于用户用汇编语言编制某些子程序，完成与机器联系紧密的特定功能，提高高级语言程序的效率。

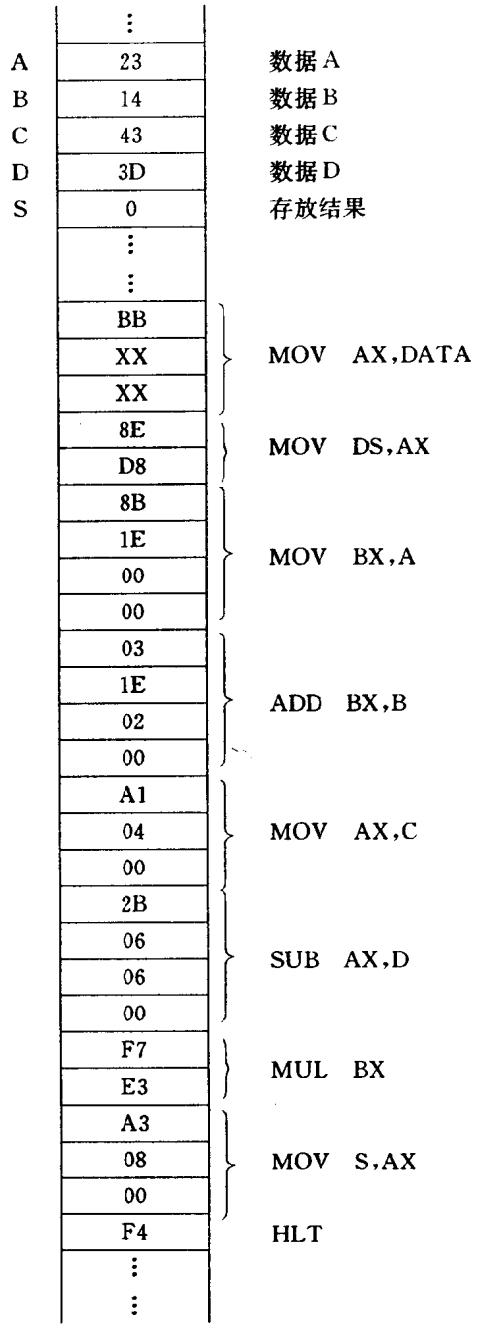

图 1-1 汇编语言程序中语句与机器代码的对应关系

§1.2 计算机中数据信息的表示

§1.2.1 进位计数制及其相互转换

一、进位计数制

用有限数码的组合（例如0，1，2…）来表示一个数，在这样组合的若干数码中，每个数码处在不同的位置（叫数位），它所表示的数值大小也不相同。例如十进制数1998，从右到左，第一位的8表示有8个1，第二位的9表示有9个10，第三位的9表示有9个100，最左边的1表示有1个1000。在数学上，对十进制数从最右边开始的数位分别为个位、十位、百位、千位 ……同时把这些个、十、百、千……称为位权（亦简称“权”）。这些位权乘以对应位置上的数码，其乘积正好表示该数位上数值的大小。

在每个数位上允许的数码是有限的。例如十进制数中每个数位上的数码只能是0，1，2，3，4，5，6，7，8，9十个数码中的一个。因此，每个数位上所能表示的最大值等于有限数码中最大的一个数码乘以该数位的权。如果某数位超过这个最大值便产生向高位的进位，这就是进位计数制的数。

每个数位上有限数码的个数叫基数，例如十进制数有10个数码（0～9），它的基数为10。由于每个进位计数制的数码中必定包含有0，所以有限数码中最大数码一定是基数－1。如十进制数中最大数码是10－1＝9。

例如十进制数1998.62可用下列多项式来表示：

$$1998.62 = 1\times10^{3} + 9\times10^{2} + 9\times10^{1} + 8\times10^{0} + 6\times10^{-1} + 2\times10^{-2}$$

在这里各数位的权分别是10^{3}，10^{2}，10^{1}，10^{0}，10^{-1}，10^{-2}，即用基数的方幂来表示。对于任意一个进位计数制，如用R表示基数，那么任何一个数S均可用如下多项式表示：

$$S = K_nR^n + K_{n-1}R^{n-1} + \cdots + K_0R^0 + K_{-1}R^{-1} + K_{-2}R^{-2} + \cdots + K_{-m}R^{-m} = \sum_{i=n}^{-m} K_iR^i$$

其中，K_i为第i数位的一个数码，即0，1，…，(R－1)中的一个；m，n为正整数。

在计算机中采用的是二进制数，它的基数为2，每个数位上只有数码0或1。用上述多项表达式来表达一个二进制数：

$$1101.011 = 1\times2^{3} + 1\times2^{2} + 0\times2^{1} + 1\times2^{0} + 0\times2^{-1} + 1\times2^{-2} + 1\times2^{-3}$$

在进位计数制中，对十进制数是“逢十进一，借一当十”，而对二进制数便是“逢二进一，借一当二”。十进制数是人们早已习惯的计数方法，而二进制数又是计算机中实现数的表示、运算和存储的最佳进位计数制。二进制数与十进制数之间的对应关系如表1-1所示。

表1-1 二进制数与十进制数对应关系

二进制数	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001
十进制数	0	1	2	3	4	5	6	7	8	9

为什么计算机中采用二进制数呢?这是因为二进制数在计算机中最容易表示和存储，且

适用于逻辑表达与运算。但是直接使用二进制数，对大多数人毕竟有许多不便，无论书写、口读或按键输入都不习惯，且容易出错。尤其是在编制程序时，会带来诸多困难。为此，我们常常使用八进制数或十六进制数。

八进制数的基数是8，有效的数码是0，1，2，3，4，5，6，7。十六进制数的基数是16，有效的数码除0，1，2，3，4，5，6，7，8，9外，对应十进制数10～15分别用字母A，B，C，D，E，F（或a，b，c，d，e，f）表示。从另一个方面来看，3位二进制数可看作是1位八进制数，4位二进制数可看作是1位十六进制数。它们之间的对应关系如表1-2所示。

表1-2　二、八、十、十六进制数之间的对应关系

十进制数	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
二进制数	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
八进制数	0	1	2	3	4	5	6	7								
十六进制数	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

在书写不同进位计数制数时，常常在一个数的尾部用一个字母来表示该数是什么进位计数制的数。二进制数用字母B（Binary）、八进制数用字母O（Octal）、十进制数用字母D（Decimal）、十六进制数用字母H（Hexadecimal）。如果数的尾部没有任何字母，那么就默认是十进制数。例如10110101B、265O、181D或181、B5H。

二、各种数制间的相互转换

由于八进制数、十六进制数与二进制数之间有固定的对应关系，按每3位或4位二进制数一组就可以完成八进制数、十六进制数与二进制数之间的相互转换。因此，各种数制间的相互转换主要是十进制数与二进制数之间的相互转换。当然，这两种数制间的相互转换方法可以很方便地引入到十进制数与八、十六进制数之间的相互转换。

1. 十进制整数转换为二进制整数

有两种转换方法：

（1）减权定位法

对于二进制数，前面给出的多项式可改写为：

$$K_n2^n + K_{n-1}2^{n-1} + \cdots + K_22^2 + K_12^1 + K_02^0$$

多项式中每项的系数K_i就组成一个二进制数：

$$K_nK_{n-1}\cdots K_2K_1K_0$$

而$2^n2^{n-1}\cdots2^22^12^0$则是每项的位权。若该项系数$K_i=0$，则该项的值为0；若$K_i=1$，则该项的值为对应项位权的值。二进制数中各位的权值，从小到大分别是：1，2，4，8，16，32，64，128，256，512，1024……根据这个对应关系可导出十进制整数转换为二进制整数的规律：从二进制数高位权起，依次用待转换的十进制数与各位权值进行比较，如够减，则该数位系数$K_i=1$，同时减去该位权值，用余下的数继续向下比较；如不够减，则该数位$K_i=0$。如此进行到所有二进制数位都能给予确定为止。

例如：十进制数325转换为二进制数。

因为512>325>256，所以从位权值为256的对应位开始进行比较：

减权比较	K_i	对应二进制数
325－256＝69	K_8	1
69＜128	K_7	0
69－64＝5	K_6	1
5＜32	K_5	0
5＜16	K_4	0
5＜8	K_3	0
5－4＝1	K_2	1
1＜2	K_1	0
1－1＝0	K_0	1

经过上述转换，325D＝101000101B。这种转换方法比较直观，易于验算。但需要记住若干位权值方可转换。

(2) 除基取余法

设待转换的十进制数为S，那么有下列等式：

$$S = K_n2^n + K_{n-1}2^{n-1} + \cdots + K_22^2 + K_12^1 + K_02^0$$

上式两边同除以基数2：

$$\frac{S}{2} = (K_n2^{n-1} + K_{n-1}2^{n-2} + \cdots + K_22^1 + K_12^0) + \frac{K_0}{2}$$

显然，等式右边括号内为除2的商，K_0是余数。如余数为0，则K_0＝0，否则K_0＝1。再对商除以基数2，可确定K_1。这样可以依次判定K_2、K_3、…、K_n等各项系数。转换的具体操作如下：

除基		余数	K_i
2	325		
2	162	1	K_0
2	81	0	K_1
2	40	1	K_2
2	20	0	K_3
2	10	0	K_4
2	5	0	K_5
2	2	1	K_6
2	1	0	K_7
	0	1	K_8

转换后325D＝101000101B。这种转换方法，操作步骤统一、规范，较便于用程序设计