

623931

5087

66177

# 电算基础

昆明工学院  
电子计算机教研室编

云南人民出版社

7  
77

623931

# 电 算 基 础

昆明工学院电子计算机教研室编

云南人民出版社

责任编辑：林德琼  
封面设计：张汉文

## 电 算 基 础

昆明工学院电子计算机教研室编

\*

云南人民出版社出版

(昆明市书林街100号)

云南新华印刷厂印刷 云南省新华书店发行

\*

开本：787×1092 1/16 印张：14.5 字数：330,000

1981年9月第一版 1981年9月第一次印刷

印数：1—10,440

统一书号：15116·135 定价：1.20元

# 前 言

电子计算机计算技术基础教程的任务是介绍如何使用计算机高级语言编制算题程序, 以及如何使用计算机算出结果。为此, 必须首先介绍程序设计的一些基本知识和基本概念, 同时必须掌握计算机语言。掌握计算机语言, 主要是掌握计算机语言的语义和语法等规定, 这是使用电子计算机进行数值计算的前题。然而, 要用计算机语言把计算问题编制成控制计算机自动运行, 以完成数值计算任务的程序, 则必须掌握编制程序的基本方法, 即程序设计基础。这是使用计算机进行数值计算的关键。因此, 作为普及电子计算机计算技术的教材, 自然应以介绍程序设计基础为重点, 至于使用何种计算机语言来介绍程序设计基础, 并不是很重要的问题。

本书选用了适宜于教学和工程技术人员使用的BASIC语言, 着重介绍了BASIC程序设计, 并以此阐述计算机高级语言的程序设计应掌握的基本方法。这不仅有利于初学者把精力主要集中在学习程序设计的基本方法上, 由易到难, 由浅入深地掌握电算技术; 而且因为目前国内的计算机一般都配有BASIC语言, 使用将比较方便。同时, 考虑到学习者今后将根据各自的工作特点和解决问题的性质的差异而选用不同的计算机语言编制程序, 有必要让初学者通过教学或自学多掌握一种计算机语言, 并且使学习者从两种计算机语言对比学习的过程中, 更深入地体会到计算机语言之间的共性和特性, 为今后学习和采用其它计算机语言打下良好的基础。于是, 在本书的下篇中, 以和BASIC语言进行对比的方式, 重点介绍ALGOL语言的语义和语法。

学习计算机语言的程序设计如同学习数学等科学一样, 在学习理论的同时必须辅之以足够的练习, 方能达到预期的效果, 不做或少做习题是不可能学好这门课程的。实践证明, 在学习过程中虽然应以学习编制算题程序为主, 然而阅读程序及修改程序这两个环节是不能忽视的, 只有同时把握设计、阅读和修改这三个环节, 才能真正入门。为此, 在本书中, 除了编选一定数量的例题外, 一般在每节末都附有较多的习题, 而习题的编选正是从兼顾编、读、改这三个方面来安排的。

目前, 电算已不可避免地成为大专院校理工科专业学生的一门公共的必修课程。本书如用作教材时, 上篇可讲授30—40学时, 下篇讲授10—15学时或者安排作为自学。若以ALGOL语言上机实习, 还需再增加10学时左右。

本书除可作为大专院校理工科非计算机专业或有关中等专业学校高年级的教材外, 还适合于高中以上文化程度的科技人员和工人自学使用。

本书是根据张浩文同志编写的讲义《电算基础》, 通过本院几个班级和训练班的教学实践, 再度由张浩文同志和车仲英同志执笔编写而成的; 车仲英同志编辑并解答了书中的全部习题, 并对各章各节的编写和修改提出了详尽的意见; 代心淑同志、江通理同志对编写和修改给予了热情的支持和帮助, 计算机教研室和计算站的同志们, 以及数学师资班的许多同学为此书的编写和出版提出过许多宝贵意见和支援, 在此一并致谢。

由于编者水平所限、经验不足, 缺点错误在所难免, 敬请批评指正。

编 者

1981年1月于昆明

# 目 录

## 上篇 高级语言的程序设计

|                         |         |
|-------------------------|---------|
| 第一章 程序设计的基本知识.....      | ( 1 )   |
| § 1 计算机的基本工作原理.....     | ( 2 )   |
| 一、指令和程序的概念.....         | ( 2 )   |
| 二、计算机的结构.....           | ( 2 )   |
| 三、计算机采用的数制及转换方法.....    | ( 5 )   |
| 四、手编程序的执行过程.....        | ( 9 )   |
| § 2 程序设计方法发展概况.....     | ( 10 )  |
| § 3 计算机的解题步骤.....       | ( 13 )  |
| 习题.....                 | ( 16 )  |
| 第二章 BASIC 程序设计 .....    | ( 18 )  |
| § 1 BASIC 语言的基本组成 ..... | ( 19 )  |
| 习题.....                 | ( 27 )  |
| § 2 简单四则运算程序.....       | ( 29 )  |
| 习题.....                 | ( 31 )  |
| § 3 赋值语句及程序修改.....      | ( 32 )  |
| 习题.....                 | ( 41 )  |
| § 4 输入语句及输出语句.....      | ( 43 )  |
| 习题.....                 | ( 58 )  |
| § 5 转向语句.....           | ( 60 )  |
| 习题.....                 | ( 78 )  |
| § 6 循环语句.....           | ( 82 )  |
| 习题.....                 | ( 98 )  |
| § 7 数组变量.....           | ( 100 ) |
| 习题.....                 | ( 113 ) |
| § 8 自定义函数和子程序.....      | ( 117 ) |
| 习题.....                 | ( 128 ) |
| § 9 程序设计小结.....         | ( 131 ) |
| 第三章 操作过程.....           | ( 136 ) |
| § 1 源程序纸带的制作.....       | ( 136 ) |
| § 2 上机操作.....           | ( 140 ) |

## 下篇 从 BASIC 语言看 ALGOL 语言

|                          |       |
|--------------------------|-------|
| 第一章 ALGOL 语言的基本组成成份..... | (143) |
|--------------------------|-------|

|                              |       |
|------------------------------|-------|
| 第二章 源程序的结构.....              | (149) |
| § 1 赋值语句和转向语句.....           | (149) |
| § 2 输入输出语句.....              | (149) |
| § 3 源程序的结构.....              | (150) |
| 第三章 分支程序.....                | (153) |
| § 1 条件语句.....                | (153) |
| § 2 条件算术表达式.....             | (155) |
| § 3 复合语句.....                | (157) |
| § 4 布尔表达式.....               | (158) |
| § 5 开关说明和开关命名符.....          | (163) |
| § 6 命名表达式.....               | (165) |
| 第四章 循环语句.....                | (167) |
| § 1 步长型循环语句.....             | (167) |
| § 2 离散型循环语句.....             | (169) |
| § 3 当型循环语句.....              | (170) |
| 第五章 分程序.....                 | (173) |
| 第六章 过 程.....                 | (176) |
| § 1 一般过程.....                | (176) |
| § 2 函数过程.....                | (185) |
| 附 录.....                     | (191) |
| 附录一 BASIC 语言五单位字符编码对照表 ..... | (191) |
| 附录二 BASIC 语言八单位字符编码对照表 ..... | (192) |
| 附录三 单用户基本 BASIC 语言简表.....    | (193) |
| 附录四 基本 BASIC 键盘命令简表.....     | (194) |
| 附录五 单用户基本 BASIC 错误信息表.....   | (195) |
| 附录六 英汉词汇对照表.....             | (196) |
| 习题参考答案.....                  | (198) |
| 第一章 习题解答.....                | (198) |
| 第二章 习题解答.....                | (199) |
| § 1 习题解答.....                | (199) |
| § 2 习题解答.....                | (200) |
| § 3 习题解答.....                | (201) |
| § 4 习题解答.....                | (203) |
| § 5 习题解答.....                | (204) |
| § 6 习题解答.....                | (211) |
| § 7 习题解答.....                | (216) |
| § 8 习题解答.....                | (221) |

# 上篇 高级语言的程序设计

## 第一章 程序设计的基本知识

电子计算机是一种既有快速计算能力，又有逻辑判断能力和存贮记忆功能，并能自动运行的现代电子设备，它是本世纪最重大的科学技术发明之一。电子计算机的出现，对科学技术领域产生了极为深刻的影响，现在已被广泛地应用于社会和生产，乃至家庭生活等领域之中。利用电子计算机可以进行科技计算、数据处理和生产过程控制等等，因而成为现代科学技术的领头学科之一。现在，掌握电子计算机的计算技术，已成为科技工作者不可缺少的一种手段。它将帮助我们摆脱各种繁重、复杂的计算，而且可以迅速得到预期的结果。

要使计算机能自动运行，必须根据所要解决的问题，预先给它安排好详细的操作内容和步骤，这就是程序设计的任务，是使用计算机不可缺少的重要环节。

下面介绍一些有关的基本概念和内容。

目前，电子计算机按运算量的差别基本上可分为三类：

### (1) 模拟计算机

模拟计算机是用模拟量作为运算量。模拟量又称为连续量。如电压、长度、转角等。这种计算机的优点是解题速度快，用它解微分方程，从把问题输入，到把结果输出只要几个微秒。缺点是精度差，一般只到小数点后二位，最多也只能到小数点后四位。因此，对于运算结果精度要求高的问题就不适用了。

### (2) 数字计算机

数字计算机以数字（又称为不连续量或离散量）作为运算量。数字计算机具有运算速度快，精度高，过程全部自动化，记忆能力和逻辑判断能力强，以及通用性强等一系列优点，目前被广泛采用。

在数字计算机中，一种小型的，以四则运算为主，能用简单的按键操作完成数值计算的计算机，称为台式计算机（简称台式机）。台式机通常用数字显示器显示结果。有些台式机还配有打印装置。台式机应用很普遍，可以取代手摇计算机和算盘等。随着电子计算机的不断发展，台式机的体积和重量迅速减小。以前可以放在桌面上的台式机（台式机因此而得名），现在只有手掌大小，便于随身携带。为了与通用数字计算机相区别，现在定名为电子计算器。

数字计算机从不同的角度还可以进行不同的分类。通常，从设计目的可以分为通用计

算机和专用计算机。通用计算机可用于解决各类问题，专用计算机工作针对性强，主要为某个特定的目的服务，如专门设计用来计算火箭、卫星的轨道，或专用于控制炼钢、轧钢、化工过程等。从用途上可分为科技计算计算机、数据处理机和工业控制机等；从规模与功能方面可分为巨型机、大型机、中型机、小型机、超小型机和微型机等。

### (3) 混合式计算机

混合式计算机是使用数字和模拟技术，集中数字机和模拟机的优点，避免两者缺点而设计出来的计算机。目前，这类计算机尚处于发展阶段。

由于数字计算机使用最广泛，一般谈及计算机时，习惯上即是指数字计算机。本书在下面提到的计算机均指通用数字计算机。

## § 1 计算机的基本工作原理

### 一、指令和程序的概念

虽然电子计算机的应用非常广泛，然而计算机所能完成的基本操作却很简单。它一般只能完成加减法、基本逻辑运算和取数送数等一些基本操作。各种复杂的运算都由这些基本操作，按一定的顺序组合起来完成的。每一个基本操作对应着计算机的一条命令，称之为机器指令，也就是指示机器进行某种基本操作的命令。当计算机接到一条机器指令后，就可以完成一个基本操作。计算机的所有指令称为该计算机的指令系统。不同类型的计算机的指令系统一般是不相同的。机器指令是用二进制或八进制数表示的，故而又称为计算机的指令代码。这种以数码形式写成的代码指令输入计算机后，计算机能识别并执行其所代表的基本操作。于是我们又称机器指令为机器语言。要使计算机完成某一计算工作或特定任务，就需要把机器指令按一定的顺序和步骤安排好，从而得到一份用机器指令编制的计算机的工作计划，即所谓程序。直接用机器指令编写的程序都是由人工编成的，故称之为手编程序，也称为代码程序。代码程序输入计算机后，机器就能自动运行，完成计算或指定任务。

### 二、计算机的结构

数字计算机的种类虽然很多，但一般来说，它们均由以下五个部分组成：存储器、运算器、控制器、输入和输出设备等。图 1.1 表示这几个主要部分关系的示意图。

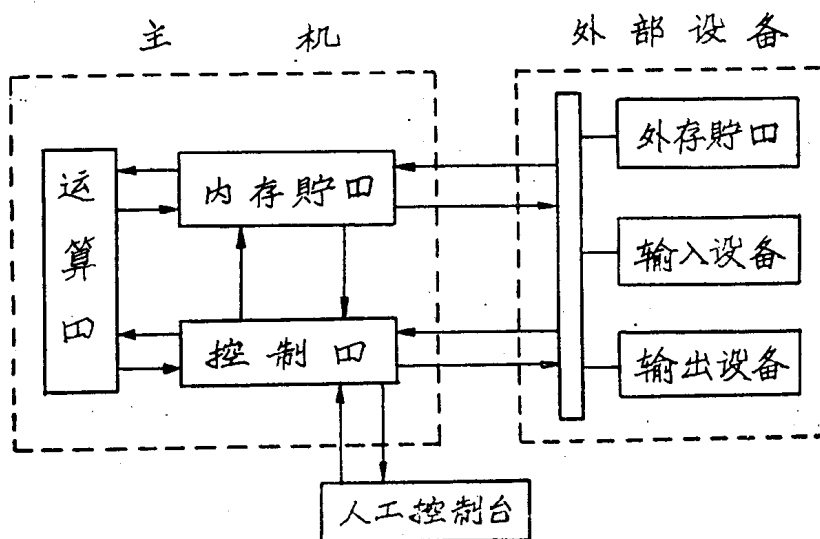


图 1.1 电子计算机基本组成示意图



兹将各部分的作用分述如下：

### 1. 存贮器

存贮器的功能是用来保存数据、程序和各种资料，这些数据、程序和各种资料统称为信息，信息都是以二进制数码（即“0”或“1”）的形式存放在存贮器中的。存贮器分成很多单元，每个单元都有一个编号，这个编号称为该单元的地址。每个单元可存贮一串二进制数码，组成一个二进制数。所存贮的二进制数的位数称为字长。字长为八位的单元称为字节。如果需要把某个数码送到某个存贮器单元（称为写入），或从某个存贮单元中取出某个数（称为读出），只需指出存放这个数的存贮单元的地址，发出相应的指令，计算机就能找到这个单元，写入或读出所需要的数据。很明显，一条指令一般应该包括两部分：操作码和地址码。操作码即指出指令执行何种操作，而地址码则指出参加运算的数据或结果的存取地址。单元的总个数称为存贮容量。

存贮器一般分为内存贮器（简称内存）和外存贮器（简称外存）。当前内存贮器普遍采用磁蕊组成，内存贮器和运算器直接相连。将内存贮器中的数据送往运算器以及将数据从运算器送往内存贮器存贮，速度都比较快。内存主要存贮当前所需的程序、数据和中间运算结果等。计算机写入或读出所需的时间称为基本存取周期。一般情况下，它也是完成一次加法运算所需要的时间，计算机的运算速度可以用它做标志。如 DJS—130 机的基本存取周期为 2 微秒，也就是说，它每秒钟可进行 50 万次加法运算。

DJS—130 机的内存最大容量可扩充到 32K ( $1K = 1024$ ,  $32K = 32768$ ) 个单元，字长 16 位。

因为内存容量是有限的，所以一般计算机都配有外存贮器（亦称代码仓库）。常用的外存有磁鼓、磁带和磁盘等，用作内存的后备存贮。凡是内存贮器中不经常使用的信息，都可以成批的存放在外存贮器中，使内存贮器存放更急需的信息。外存贮器工作速度较低，且与运算器没有直接联系，但容量较大。如磁带存贮器，它好像磁带录音机一样，所用的一盘盘磁带是可换的，计算机长期运行中所积累的大批程序、数据、资料都可以保存在许多磁带上，一旦需要重新使用，只要把磁带装在磁带机上就可以重新调进内存。所以磁带存贮器的容量实际上是无限的。

存贮器能够从外部接受信息，并长期保存，而在需要时读出。读出后该项信息仍保留而不受破坏。存贮单元也是如此，它保存的信息可以多次读出，并能保持不变，供以后继续使用。只有向该单元重新写入新信息时，才会将原信息“挤掉”，而代之以新信息贮存。这是在程序设计过程中必须随时牢记的基本性质，即读出不改变存贮单元的信息，而写入将“冲掉”原有信息。因此，在写入信息时，要注意写入的单元是否空闲或原保存的信息是否已无继续保留的必要，以避免将需要保存的信息冲掉。

存贮器的工作速度和存贮容量是计算机的两个重要指标，它们影响计算机的运行速度，决定计算机能存放的数据和程序的数量，限制所能解决问题的规模。用容量小速度高的内存和容量大速度低的外存相互配合使用，可以提高计算机的解题能力和工作效率。

计算机的运算速度除了用内存存取周期或每秒钟完成的加法次数表示之外，有一些计算机是用平均运算速度来反映。所谓平均运算速度，就是用一般的统计方法统计所得到的计算机每秒钟完成的机器指令的次数。例如 DJS—21 机平均运算速度为每秒钟三

万次。

## 2. 运算器

运算器是计算机中进行各种运算的部件。一切算术运算，如加、减、乘、除、逻辑加、逻辑乘及比较运算等等都在运算器中进行。这些运算通常都是由运算器中的加法器完成。加法器是完成加法运算的部件。而减法和乘除运算并不一定需要增加专门的减法器或乘除法运算器，通常是靠一些逻辑控制门的控制，将它们分解转化成加法来实现的。不仅如此，所有各种复杂的运算都通过一定的计算方法转化为四则运算，而四则运算最终又都转化成加减法。这种方法虽然在人工进行数值计算时，因更为麻烦而无必要，但是对于计算机来说，由于其运算速度很快，每秒钟可以完成几十万次甚至上千万次、上亿次加法运算，这种方法就合用了。有一些计算机的运算器只有加法器，这样可以使运算器的结构简化。不过，为了方便做乘除运算，加快乘除运算速度，现在很多计算机增加了乘除部件。

运算器还包括代码寄存器，存放即将进行运算的数和暂时存放中间结果。

DJS—130机的运算器有一个加法器（称全加器）和四个寄存器（称累加器，它只有寄存性能，没有累加功能）。

## 3. 控制器

控制器是计算机的指挥部件，主要作用是使计算机自动地、连续地执行程序。目前的计算机的控制器基本上都是采用“内存程序控制”方式。即在计算机运算前，先将程序按顺序存放在内存贮器中，计算机开始运行后，控制器从内存中顺序取出指令，并向各部件发出相应命令，使它们一步步地执行程序所规定的任务。运行过程中，中间结果的大小以及是否超出预定界限等问题，会影响到下一步的工作状态，输入输出部件的工作，也会产生同样的影响，因此，控制器还同时接受其他各部件完成任务情况的“汇报信息”（即反馈信息），以决定下一步是继续顺序执行下一条指令，还是改变执行顺序，直至完成程序所规定的任务。

为了方便人们对机器的控制，以及观察机器运行情况，计算机设有人工控制台（或控制面板）。它起着人和机器的联系作用，以控制机器的启动或停止工作；临时输入或输出某些指令或数据；改变运算顺序和个别内存单元或寄存器的内容，等等。

## 4. 输入设备

输入设备是用来把程序和原始数据输入到电子计算机的内存贮器。目前 DJS—130机的输入设备主要有电传输入机，纸带输入机（光电式或电容式）。电传机用按键方式输入，而纸带输入机则要求先将程序和初始数据在纸带上穿好孔，然后通过纸带输入机将带上的程序和数据等信息转换成电脉冲输入到内存中。

## 5. 输出设备

输出设备的功能是输出计算结果及内存中有关信息。目前 DJS—130机的输出设备主要有电传输出机（输入输出两用）、快速穿孔机、宽行打印机、绘图仪、字符显示器及图形显示器等。目前很多计算机用控制打字机代替电传打字机而兼有按键输入、打印输出、穿孔输出及纸带读入等几种功能。

通常，控制器、运算器和内存贮器是组装在一起的，它们是电子计算机的主要部

分，称为主机。而输入输出设备、外存贮器则称为外部设备。通常还把运算器与控制器总称为中央处理机，内存称为主存。DJS—130 机最多可配上62种外部设备。

三、计算机采用的数制及转换方法

1. 二进制、八进制数

(1) 二进制、八进制的特点及其转换

在日常生活中常用的是十进制。其特点是逢十进一。十位的“1”是个位的“1”的10倍，百位的“1”是十位的“1”的10倍。总之，左边的一个单位是相邻的右边的一个单位的十倍。所以，每个十进制数可以写成“10”为底的一的和的形式。例如：

$$250.36_{(10)} = 2 \times 10^2 + 5 \times 10^1 + 0 \times 10^0 + 3 \times 10^{-1} + 6 \times 10^{-2}$$

$10_{(10)}$ 称为十进制的基数。每个数的右下方括号中的下标数字表明属于哪种数制。

电子计算机中采用的是人们不太习惯的二进制。二进制数的每一位只有两种可能：不是“0”就是“1”。其特点是逢二进一。现将十进制的10个数码及其对应的二进制数列表如下：

|       |   |   |    |    |     |     |     |     |      |      |
|-------|---|---|----|----|-----|-----|-----|-----|------|------|
| 十 进 制 | 0 | 1 | 2  | 3  | 4   | 5   | 6   | 7   | 8    | 9    |
| 二 进 制 | 0 | 1 | 10 | 11 | 100 | 101 | 110 | 111 | 1000 | 1001 |

在二进制数中，左边一位的一个单位是右边一个单位的二倍，而右边的一个单位则是左边一个单位的  $2^{-1}$  倍。所以，一个二进制数可以写成以2为底的一的运算形式。例如：

$$\begin{aligned} 10101_{(2)} &= 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 \\ &= 16 + 4 + 1 \\ &= 21_{(10)} \end{aligned}$$

又如：

$$\begin{aligned} 101.101_{(2)} &= 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3} \\ &= 4 + 1 + \frac{1}{2} + \frac{1}{8} \\ &= 5.625_{(10)} \end{aligned}$$

$10_{(2)} = 2_{(10)}$ 称为二进制的基数。

用二进制表示数，书写麻烦，读写和记忆都不方便，尤其是在数的位数较长时，这一缺点更为突出。为了便于读写和记忆，将二进制缩写就得到八进制。八进制是“逢八进一”的进制。八进制数中每一位的数字由0，1，2，3，4，5，6，7中的一个数码构成。因三位二进制数可表示八种不同的状态

|       |       |       |       |       |       |       |       |
|-------|-------|-------|-------|-------|-------|-------|-------|
| 0 0 0 | 0 0 1 | 0 1 0 | 0 1 1 | 1 0 0 | 1 0 1 | 1 1 0 | 1 1 1 |
| 0     | 1     | 2     | 3     | 4     | 5     | 6     | 7     |

也是逢八必进一。因此将任何一个二进制数从小数点起，向左向右依次将每三位作为一组，进行分组，则表示成八进制的形式。为使整数及小数部分的数码个数正好是3的倍数，可在整数前加零或在小数后加零。例如，

$$101110101_{(2)} = 565_{(8)}$$

$$10011.0111_{(2)} = 010011.011100_{(2)} = 23.34_{(8)}$$

$$341_{(8)} = 11100001_{(2)}$$

八进制的基数是  $10_{(8)} = 8_{(10)}$

二进制和八进制的转换是很方便的。二进制转换成八进制按“三位并一位”，即“三并一”的原则。而八进制转化为二进制则用“一位拉三位”，即“一拉三”的办法。

八进制数是二进制数的另一种表示方法，是二进制的缩写。因此，对二进制的讨论可以归结为对八进制的讨论。若将二进制数二位并一位则得四进制数。而四位并一位则得十六进制数……。

## (2) 二进制、八进制与十进制的相互转换

每个二进制数都可以写成以2为底的一的展开式。因此，一个二进制数要化为十进制数时，只要将展开式值计算出来，就得到所要求的十进制数。同理，要将一个八进制数转化为十进制数，只要计算出它所对应的以8为底的一的展开式的值即可。

例1. 将二进制数11011.11及八进制数137.64分别转化为十进制数。

$$\begin{aligned} \text{解: } 11011.11_{(2)} &= (2^4 + 2^3 + 2^1 + 2^0 + 2^{-1} + 2^{-2})_{(10)} \\ &= (16 + 8 + 2 + 1 + 0.5 + 0.25)_{(10)} \\ &= 27.75_{(10)} \\ 137.64_{(8)} &= (8^2 + 3 \times 8^1 + 7 \times 8^0 + 6 \times 8^{-1} + 4 \times 8^{-2})_{(10)} \\ &= (64 + 24 + 7 + 0.75 + 0.0625)_{(10)} \\ &= 95.8125_{(10)} \end{aligned}$$

要将十进制数转化为二进制数或八进制数，也从展开式中寻找方法，下面通过具体例子进行讨论。

例2. 将十进制整数25转化为二进制数及八进制数

解：假定 $25_{(10)}$ 所对应的二进制数为

$$2^k a_k + \dots + 2^1 a_1 + a_0_{(2)},$$

即

$$25_{(10)} = a_k 2^k + \dots + a_1 2^1 + a_0_{(2)}$$

其中 $a_i (i=0, 1, 2, \dots, k)$ 非0则1，尚未确定。若求出了所有的 $a_i$ ，则完成了十进制数转化为二进制数的任务。将上式右端的二进制数写成展开式，则为

$$\begin{aligned} 25_{(10)} &= a_k \times 2^k + a_{k-1} \times 2^{k-1} + \dots + a_2 \times 2^2 + a_1 \times 2^1 + a_0 \times 2^0 \\ &= 2(a_k \times 2^{k-1} + a_{k-1} \times 2^{k-2} + \dots + a_2 \times 2 + a_1) + a_0 \end{aligned}$$

从上式可以看出，将 $25_{(10)}$ 除以2，所得的余数等于 $a_0$ ，所以

$$a_0 = 1$$

而

$$a_k \times 2^{k-1} + a_{k-1} \times 2^{k-2} + \dots + a_2 \times 2 + a_1$$

就是 $25 \div 2$ 所得的商12，因此

$$\begin{aligned} 12_{(10)} &= a_k \times 2^{k-1} + a_{k-1} \times 2^{k-2} + \dots + a_2 \times 2 + a_1 \\ &= 2(a_k \times 2^{k-2} + a_{k-1} \times 2^{k-3} + \dots + a_2) + a_1 \end{aligned}$$

将 $25_{(10)}$ 除以 2，所得的余数应等于 $a_1$ ，所以

$$a_1 = 0$$

重复以上步骤，直到某一次的商等于零为止，便求出了二进制数的各位数码。这种将十进制数转化为二进制数的方法称为除 2 取余法。上述计算过程可用下列比较简单的格式进行运算：

| 被除数及商  | 余 数           |
|--------|---------------|
| 2   25 | 1 ..... $a_0$ |
| 2   12 | 0 ..... $a_1$ |
| 2   6  | 0 ..... $a_2$ |
| 2   3  | 1 ..... $a_3$ |
| 2   1  | 1 ..... $a_4$ |
| 0      |               |

将所得余数按先后秩序从右到左排列，得到所求的二进制数，再将二进制三位并一位得到八进制数，即

$$25_{(10)} = 11001_{(2)} = 31_{(8)}$$

十进制整数转换为八进制整数可以不通过二进制过渡，而直接采用“除八取余法”完成转换任务。

例3. 将十进制纯小数 $0.625$ 转化为二进制和八进制小数。

除 2 取余法只能将十进制整数转换为二进制整数，不适用于十进制小数的转换。因此，需要从二进制小数的展开式中寻找转换方法。

设

$$0.625_{(10)} = 0.a_{-1}a_{-2}a_{-3}\dots\dots a_{-n(2)}$$

其中 $a_{-i}(i=1, 2, 3, \dots\dots)$ 为 0 或 1，尚未确定。现在的任务是确定 $a_{-i}$ 。将上式右端的二进制小数写成展开式，则

$$0.625_{(10)} = a_{-1} \times 2^{-1} + a_{-2} \times 2^{-2} + \dots\dots + a_{-n} \times 2^{-n}$$

将上式两端同时乘以 2，则得

$$1.25_{(10)} = a_{-1} + a_{-2} \times 2^{-1} + \dots\dots + a_{-n} \times 2^{-n+1}$$

等式两端的整数部分和小数部分必须分别相等，因此

$$a_{-1} = 1$$

$$0.25_{(10)} = a_{-2} \times 2^{-1} + a_{-3} \times 2^{-2} + \dots\dots + a_{-n} \times 2^{-n+1}$$

上面第二个等式两端再同乘以 2 得：

$$0.50_{(10)} = a_{-2} + a_{-3} \times 2^{-1} + \dots\dots + a_{-n} \times 2^{-n+2}$$

因此

$$a_{-2} = 0$$

重复上述步骤得：

$$1.0 = a_{-3} + a_{-4} \times 2^{-1} + \dots\dots + a_{-n} \times 2^{-n+3}$$

因此

$$a_{-3} = 1, a_{-4} = a_{-5} = a_{-6} = \dots\dots = a_{-n} = 0$$

于是

$$0.625_{(10)} = 0.101_{(2)} = 0.5_{(8)}$$

二进制小数有效数字后的零对数值不发生影响，可省去不写。这种将十进制纯小数转化为二进制纯小数的方法，称为“乘2取整”法。同理，将十进制纯小数直接转化为八进制纯小数采用“乘8取整”法。

乘2取整法可以使用下列比较简单的格式：

| 整数部分                  | 纯小数部分            |
|-----------------------|------------------|
|                       | 0.625            |
|                       | $\times \quad 2$ |
| $a_{-1} \dots\dots 1$ | $\hline .250$    |
|                       | $\times \quad 2$ |
| $a_{-2} \dots\dots 0$ | $\hline .50$     |
|                       | $\times \quad 2$ |
| $a_{-3} \dots\dots 1$ | $\hline .0$      |

必须强调指出，“取余法”依次从低位起逐步算出高位的数码，而取整法是从小数点后的低位依次得到低位的数码，两者方向正好相反。

例4.将十进制数357.32转化为二进制数

“取余法”只能转化十进制整数，“取整法”只能转化十进制纯小数。对于既含整数，又含小数的十进制数，必须分为整数和小数两部分分别转化，转化后再合并。

当位数比较多的十进制数要转化为二进制数时，取余和取整步骤较多，不仅麻烦，而且容易出错。为此，可以先转化为八进制数，使转化步骤大为减少，然后按“一位拉三位”过渡到二进制数。这种方法会更方便一些。下面将357及0.32分别转化为八进制数。将357进行除8取余运算：

| 被除数及商                | 余数                 |
|----------------------|--------------------|
| 8 $\overline{) 357}$ | 5 $\dots\dots a_0$ |
| 8 $\overline{) 44}$  | 4 $\dots\dots a_1$ |
| 8 $\overline{) 5}$   | 5 $\dots\dots a_2$ |
| 0                    |                    |

$$\therefore 357_{(10)} = 545_{(8)}$$

将0.32进行乘8取整运算：

| 整 数                   | 纯 小 数            |
|-----------------------|------------------|
|                       | 0.32             |
|                       | $\times \quad 8$ |
| $a_{-1} \dots\dots 2$ | $\hline .56$     |
|                       | $\times \quad 8$ |
| $a_{-2} \dots\dots 4$ | $\hline .48$     |
|                       | $\times \quad 8$ |
| $a_{-3} \dots\dots 3$ | $\hline .84$     |
| $\vdots$              | $\vdots$         |

继续重复取整步骤，可以发现，这一取整过程是无限的，不会终止。因此， $0.32_{(10)}$ 只能转化为无限位的八进制或二进制数。遇到这种情况，只能根据精度要求取近似值。如果取

$$0.32_{(10)} \doteq 0.243_{(8)}$$

则

$$357.32_{(10)} \doteq 545.243_{(8)}$$

按“一位拉三位”的办法，得到二进制数；

$$357.32_{(10)} \doteq 101100101.010100011_{(2)}$$

## 2. 二——十进制

在计算机上，除使用二进制和八进制外，还使用二——十进制。这是计算机常用的一种存放十进制数的方法。因为计算机不能以十进制的形式存放数据，而只能以二进制形式存放。如果输入的数据都要化为二进制是不方便的。为此，用四位二进制数表示一位十进制数，这样，十进制数就以二进制数的形式存放于计算机中，这就是所谓的二——十进制。二——十进制本身仍是十进制，仅只是存放按二进制的方式，而不是真实的二进制数。

本来，十进制的0和1与二进制的0和1是一致的，但在二——十进制中，无论这一位十进制数是0还是1，都必须用四位二进制数表示。如果是0，则用0000表示；如果是1，则用0001表示；如果是9，则用1001表示。现将十进制数的十个基本数字及其对应的二——十进制数列表如下：

| 十 进 制  | 0    | 1    | 2    | 3    | 4    | 5    | 6    | 7    | 8    | 9    |
|--------|------|------|------|------|------|------|------|------|------|------|
| 二——十进制 | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 |

例如：十进制数8375用二——十进制表示为：

1000001101110101

而8375的真二进制数为：

10000010110111

二者显然是不同的。

输入机器中的二——十进制不能直接参加运算，必须转化成真二进制数，转换过程是由机器完成的。

四位二进制有16种状态。表示十进制数的十个数码只用到10种状态，抛弃6种不用。选择10种状态的编码方法是有多种的。因此，二——十进制的表示方法有多种，上表是常用的一种，称为8—4—2—1码。

## 四、手编程序的执行过程

根据所给问题的计算步骤，用机器指令编好程序，并把程序和初始数据由输入设备输入内存，按指定的内存区域依次存放。安排程序存放区域的工作称为内存分配。手编程序的内存分配由程序设计者安排。程序开始执行的第一条指令所在单元的地址，称为程序入口（程序入口并不一定放在程序的最前面）。当程序由入口地址开始启动后，机器首先执行入口地址单元中存放的指令。此后按各条指令排列的先后秩序一条接一条地顺序执行下去，如果遇到控制转移性质的指令，则向前或向后跳过若干条指令转移到预先安排好的地址，再顺序执行下去。在执行过程中，若遇到输出指令，则输出有关数

据，直到程序结束为止。

下面我们以计算  $a + b$  为例，来看计算机的解题过程。用 DJS—130 机的指令来编制程序，完成这一加法的程序为：

|    |        |
|----|--------|
| 40 | a      |
| 41 | b      |
| 42 | 020040 |
| 43 | 024041 |
| 44 | 107000 |
| 45 | 044047 |
| 46 | 063077 |

左边一排数为指令或数据的存放地址，右边一排为数据或指令。两排数都是用八进制书写。40、41号单元分别存放被加数  $a$  和加数  $b$ ，42号单元为程序入口，存放一条取数指令，将  $a$  送到运算器的 0 号寄存器中。43号单元存放的是将  $b$  送入 1 号寄存器的取数指令。44号单元存放的是一条加法运算指令，这条指令将 0 号和 1 号寄存器中的数在全加器中相加后送到 1 号寄存器。45号单元存送一条送数指令，其作用是将 1 号寄存器中存放的结果送到 47 号单元存放。46号单元存放停机指令。在这个简单的加法程序中没有转移指令，程序从 42 号单元开始执行后就顺序地从 42，43，……向下一条执行，直到 46 号单元中的停机指令被执行后，机器停止运行，程序运算结束，所得的和数存放在 47 号单元中。

从此例中可以看到，手编程序就象一份密码电报。要想了解每条指令所表示的具体操作，必须将八进制代码改写成二进制数码，并根据指令系统的详细规定，从二进制代码的每一位所起的作用进行分析，才能弄清楚该指令所决定的基本操作内容。

## § 2 程序设计方法发展概况

在计算机出现的初期，用计算机解题都是直接使用机器语言编写的程序。这种手编程序有其一定的优点，它能充分发挥机器的性能，程序结构紧凑合理，质量高，机器可以直接执行。但是用机器语言编写程序，完全是一些编码工作，不仅编写程序花费时间，效率低，而且容易出错，难写难认，难于修改。机器语言都是用一些数码来表示，和人们习惯使用的语言差距很大，因而不容易为初学者掌握。此外，机器语言对具体的机器依赖性很大，不同的机器一般都有不同的机器语言。因此，手编程序通用性不强，不便于交流。即使已经掌握了一台计算机程序编制的程序员，换了一台计算机又必须重新了解新计算机的机器语言，才能从事程序编制工作。因此，使用机器语言编制程序不利于电算技术的普及和推广。

随着电子计算机的迅速发展，运算速度不断提高，计算机的高速度与机器语言编程序十分缓慢的矛盾越来越突出。费时几天，甚至几个月编制出的某些程序在计算机上几秒钟或几分钟就完成了。手编程序编制很慢是机器语言本身的代码形式所决定的。为了简化程序设计方法，提高编制程序的速度，必须从改进计算机语言（即用来编制计算机



运行程序的语言)方面入手。

如果把机器语言改进成为类似于人们习惯的“数学语言”，即使用数字、字母、数学符号 $+$ 、 $-$ 、 $\times$ 、 $\div$ 、 $<$ 、 $=$ 、 $\sin$ 、 $\cos$ 等和文字说明来表达计算问题的语言，那么，用这种类型的语言编制程序必然很方便，编制的速度会大为提高，也便于普及。然而，计算机只能执行机器语言所编写的代码程序，其它任何形式的语言编写的程序都不能在计算机上直接执行，只有翻译成等价的机器语言编写的程序后，计算机才能执行。因此，要改进和设计新的计算机语言，就必须同时考虑程序的翻译问题。这个问题若不解决好，改进或设计新的计算机语言将没有实用价值。

很明显，程序的翻译工作不能采用人工完成，因为人工将其它语言编写的程序翻译成机器语言的代码程序与用机器语言直接编写程序实际上没有多少区别，工作量和所费的时间并不见得会减少。只有找到提高程序翻译速度的办法，设计新的计算机语言才有实际意义。既然人工翻译程序没有实用价值，就只有从计算机上打主意了。

自从电子计算机工作者利用计算机具有一定逻辑判断能力，设计了翻译程序，使计算机利用翻译程序自动完成程序的翻译工作之后，程序设计工作才取得了显著的进展，各种方便程序设计的计算机语言和与之相应的翻译程序不断问世。使程序设计工作者摆脱了大量烦琐的劳动。这种配备了相应的翻译程序的语言，称为程序设计语言，简称程序语言。用程序语言编写的程序称为源程序。人们只要花少量时间就可以编制好源程序。而源程序翻译成机器代码程序的工作量与源程序的编制相比则要大得多，但是这一繁重工作已由配备了翻译程序的计算机自动完成，从而使程序编制的速度至少提高了一个数量级。这种将程序设计工作大部分由计算机来完成的方法，称为程序设计自动化。因此，程序设计语言又称为程序设计自动化语言。使用程序设计语言，不仅提高了程序编制速度，而且对程序编制人员的技术要求降低了，这样就便于普及和推广。为了同手编程序相区别，我们把由源程序翻译而得的机器代码程序称为结果程序。

程序设计语言的发展是由低级逐步发展到高级的。最初出现的是符号语言——即机器语言符号化，它是用便于记忆的符号代替机器指令的操作码（称为记忆码），用符号地址表示地址码，使程序比手编程序直观得多，也比较容易阅读和修改，同时省去了程序设计员的内存分配和代真工作（把符号地址转换为有效地址的过程称为代真），而由计算机在翻译过程中自动完成。

符号语言进一步发展，得到汇编语言。它是针对一类或几类计算机抽象出来的一种符号语言。汇编语言和符号语言的主要差别是：符号语言是针对一台特定的计算机设计的，而汇编语言不仅对一台，而且可以在一类计算机上使用，从而用汇编语言编制的程序能在这一类的任何一台计算机上运行，使汇编语言的使用范围扩大了。将汇编语言编制的源程序翻译成机器语言的代码程序的翻译程序有个专门名称，叫汇编程序。而翻译过程叫做汇编。

用汇编语言编制的程序保留了手编程序的优点，简化了程序设计工作。但是，汇编语言的源程序仍是一条一条的符号指令所构成，它和手编程序的代码指令基本上是一一对应的。用汇编语言编制程序仍然需要熟悉所使用的计算机的结构和性能，仍然需要了解计算机的指令系统和表示方法。因而，对机器还有一定的依赖性，编程序时也仍然需要