程序设计基础

罗昌隆 编

西北

程序设计基础

(PDP-11)

罗昌隆 编

西北电讯工程学院

前　言

本书是根据教育部召开的1980年4月南京计算机软件教材座谈会、1981年10月杭州计算机软件专业教材会议和计算机学会教育专业组召开的软件课程教学大纲讨论会的精神编写的，可作为计算机科学和工程专业的基础课教材，授课时数为70～90学时左右。

本书是以具有代表性的、国内外广泛使用的PDP-11系列机为背景，介绍用汇编语言进行程序设计的基本知识、基本方法和基本技巧。第一、二、三章介绍程序设计的预备知识和基本概念，第四、五、六、八章介绍程序设计的基本方法——分枝程序、循环程序、子程序、输入输出程序设计和中断处理，第七、九章着重介绍了宏汇编语言和汇编程序，第十章是在程序设计的基本方法基础上进一步介绍了某些技巧，第十一章介绍在计算机上运行一个程序的过程以及有关系统软件——编辑程序、任务建立程序和调试程序，每章后有一定数量的习题供读者练习。

程序设计是一门实践性很强的科学。因此，希望在学习本书的时候，能实际动手，安排一定的上机时间，这样才能真正掌握。

在编写本书时，我们力求深入浅出、平易好懂，便于初学者自学。因此，本书也可以供计算机领域中的初学者自学或专业人员参考。

本书是编者多年来在我院讲授计算机程序设计课程的基础上，吸取和参考了有关资料及兄弟院校的教材，经过多次修改而成的。

在编写过程中，得到了我院教材处、我系领导和303、301教研室以及兄弟院校许多同志的关心和支持，甘圣予和金益民同志校阅了部分章节，徐学洲等同志对内容提出了许多宝贵的意见和建议，在此，表示衷心感谢。

限于编者的水平，书中一定存在不少错误和不足之处，敬请读者批评指正。

编者　罗昌隆

1983年2月

目　录

第一章　程序设计的一般概念

第二章　计算机中数的表示及逻辑运算

第三章 PDP-11 (DJS-180)系列计算机组成和结构

第四章　简单程序设计

第五章 循环程序设计

第六章 子程序设计

第七章 汇编语言的程序设计

第八章 输入/输出程序设计和中断处理

第九章 宏指令

第十章 程序设计的某些技巧

第十一章 程序的开发和运行

第一章　程序设计的一般概念

§1.1　电子数字计算机和它的组成

1.1.1　电子数字计算机的简单框图

现代的电子数字计算机是一种能自动地进行高速运算的计算工具，它每秒钟能进行成千上万次各种不同的运算，有的可达每秒几百万至上亿次运算。但不管它的速度多高，结构多复杂，它所进行数字运算的工作原理仍然只能是模拟人们手工计算的过程。为了要了解电子数字计算机解题的工作原理，我们首先从分析具体的人们最有体会的手工计算过程着手，并从中归纳出手工计算的一般规律，藉以来认识电子数字计算机解题的基本原理。

例如有一个人用算盘来计算下面题目：8456 + 1245 − 3243。如果我们把寄存十进数的算盘面记为 R，则其计算过程的先后顺序列表如下：

序号	操作的命令	注　　解
0	0⇒R	清除盘面
1	8456⇒R	在算盘中拨上8456
2	(R) + 1245⇒R	算盘中加上1245
3	(R) − 3243⇒R	算盘中减去3243
4	(R)抄送纸上	抄送运算结果
5	停　　止	计算果结

在执行上述六步操作之后，在算盘 R 中有如下运算的结果：

$$(R) = 8456 + 1245 - 3243 = 6458$$

这就是使用算盘进行解题过程的形式描述，其中算盘 R具有累加运算结果的作用。

上述计算过程中的每一步（例如 (R) + 1245⇒R 等等）都是指示人完成相应操作的命令。指示操作的命令，我们称为指令。

上述构成计算过程先后顺序的六条指令就称为解题的程序，编制解题程序的技术称为程序设计。

对上述解题的程序，算盘是不能自动地执行的，实际的执行是由使用算盘的人按写在纸上的解题程序和数据来进行的，所以(1)使用算盘的人，(2) 算盘，(3) 记录数据和程序的纸等三者是计算过程所不可缺少的必要的组成部分。

电子数字计算机，不管它的类别如何，速度多快，由于它只能模拟人们手工的计算过程，为了能自动地进行工作，就必须满足下述三个要求，也就是必须具备三个最基本的装置。

第一，机器必须能进行全部必需的基本算术运算，为此，机器要有相当于上例中算盘的运算器。

第二、机器中必须能保存和记录原始数据，解题程序以及运算中间结果。因此，要有具

有足够容量的存贮器，相当于上例中的纸。

第三，机器要能按照解题程序或根据中间运算结果自动选择所需的下一步计算动作。为此，机器需要有能起控制作用的控制器，它好象人的手和眼睛一样，能判断，打算盘、记录等作用。

除了这三个装置外，为了使机器按人的命令工作，需将原始数据和解题程序输入给机器，同时，机器算得的结果又需要表达给人知道，因此，还需要有输入和输出设备。

图1.1表明了这五大部件之间关系。

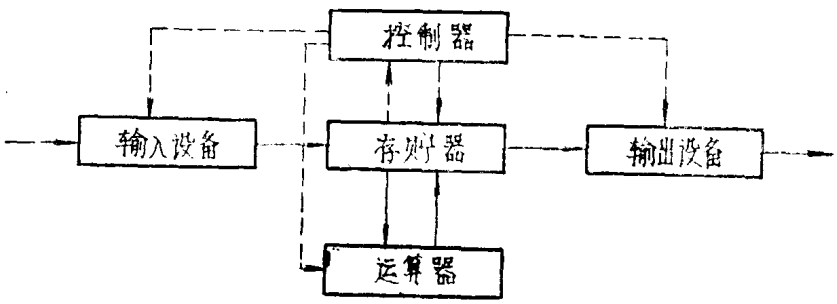


图1.1 电子计算机的简单框图

其中实线表示数据流向路线，而虚线表示控制信息的流向路线。

运算器是对数据进行运算的部件，它能快速地进行加、减等算术运算及其它一些常见的基本运算（例两数比较）。

在运算过程中，运算器不断地得到由存贮器提供的数据，并能把求得的结果（包括中间结果）送回存贮器暂时保存起来。它的整个运算是在控制器统一指挥下有规律地进行的。

控制器是数字计算机中的管理部件或开关部件，它接收进入机器的信息，并决定如何执行和何时执行操作；它告诉运算器做什么和从何处得到必需的数据。当计算机已经做完一步计算时，它告诉运算器对结果怎么办和下一步做什么。

控制器本身是从存贮器中顺序地取出指令进行解释，以决定应向各部件发出何种命令，使它们一步步地执行命令所要求的任务。

存贮器的主要功能是保存大量信息。它的作用类似一台录音机，能把已记录的内容保存起来。在使用时，根据实际的需要可以把原来记录的内容抹去而重新记录新内容，或者把原记录的内容取出但不破坏原有的记录。在计算机的存贮器内保存的信息，主要有数据和指令两种。在解题之初，程序和原始数据从计算机外送入存贮器中保存起来，在计算过程中，存贮器一方面不断地向运算器提供运算所需要的数据，另一方面还能保存从运算器送出的计算结果。此外，存贮器中保存着的计算程序，其作用是决定计算机的具体工作过程：计算机从存贮器不断地取出指令送往控制器，然后由控制器分析和解释指令的含意，并据此向运算器和其他部件发出相应的命令，指挥、控制各部件执行指令规定的动作。

输入和输出部分是计算机和外界进行联系的桥梁，它包含各种类型的输入和输出设备以及输入和输出的控制部件。计算机为解题所需的数据和程序，以及告诉计算机应该做什么的命令通过输入设备送到计算机里去。在计算机计算完毕后，还要把计算的结果（需要时，也可以是中间结果）通过输出设备送出来。计算机使用起来是否灵活方便，也与是否有性能优良、工作可靠的输入和输出设备关系很大。

概括起来说，控制器和存贮器在各部件的相互联系中起着重要的作用：控制器是协调各部件的动作，不断向它们发出命令的中央机构；而存贮器则是各部件信息联系的中心和仓库，各个部件所用的数据以及其他信息大部分都存放在存贮器内，并通过它转送到其他部件去。下面我们对这些部件的主要性能、大致结构作进一步说明。

1.1.2 电子数字计算机的主要组成部分

一、运算器

运算器是对数和指令进行各种算术运算和逻辑运算。

在电子数字计算机中，数采用数字式（又称离散式）表示方法。即根据不同的要求，用足够多的依次排列的数字来精确地表示一个数的大小。例如 $\pi = 3.14159\cdots$，用六位数字表示圆周率 π。通常，人们用电压的大小来表示数字，比如我们可以用十种不同的电压表示十个不同的数码，但实现起来是相当困难。为了使计算机工作简单可靠，一般都采用二进制数字表示法，即是直接用电压的高低来表示两种不同的数码（即“0”和“1”）。比如说，从0伏到3伏变化的低电压规定代表“0”，而从3伏到5伏变化的高电压规定代表“1”。这种只用两种数码来表示数量的方法就称为二值表示法或二进制表示法。在实际的计算机中，往往用几十个二进制数码来表示一个数据，它们能很精确地表示出一个数的大小。

运算器主要由保存数据的“寄存器”和能够进行运算的“加法器”所组成（图1.2）。寄存器用来保存参加运算的数据或者保存运算求得的结果。在图1.2中画出了两个寄存器A和B。寄存器之间进行数据运算一般都通过加法器，它们和加法器之间通过“传送门”相连，以便使数据能够有控制地从寄存器传送到加法器，或者把加法器的结果送到某一个寄存器。

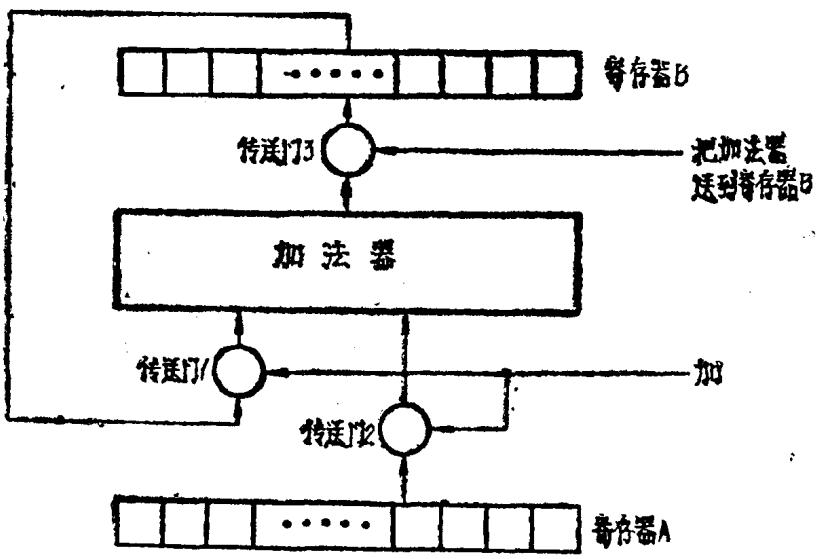


图1.2 (B)←(A)+(B)的加法电路

例如，“加法”操作的顺序可以是：

1，控制器首先向运算器发出“加”命令，把“传送门”1和2打开，把操作数A（在寄

存器 A 中）和操作数 B（在寄存器 B 中）一起传送到加法器，使 A 和 B 一起相加。

2，控制器向运算器发出“把加法器内容送到寄存器 B”命令，把“传送门”3 打开，这时把加法求得的结果通过它送到寄存器 B（B 中原保存内容被挤掉），这个结果被保存起来以便参与下一步计算。

因为寄存器 B 可用来保存运算结果并能参与下一步计算，所以常称为“累加寄存器”，对现代计算机来说，为了提高运算速度，其运算器内可以有好几个累加寄存器。

二、存贮器

存贮器用来保存大量的二进制代码，分为内存贮器（也称主存贮器）和外存贮器（也称辅助存贮器）。

内存贮器好象一个旅馆，内有很多“房间”，这些“房间”，我们称它为“单元”，这时内存贮器被分成成千上万个单元，每个存贮单元（房间）可以存放一串二进制代码（一串“0”或“1”），代表一个数据或一条指令。例如，某内存贮器共有 4096 个单元，每个单元可存放16个二进制码。这时可以说，这个内存贮器的“容量”为 4096（单元），也可说成 4K（1K = 1024），单元的“字长”为 16（每个单元存放 16 位）。或者直接说，该存贮器容量为 4096 × 16个二进制位。

为了区分存贮器的不同的单元，要把存贮器全部单元按照一定顺序编号，它相当于每个房间有一固定的房号，每个单元的号码称为该单元的“地址”，不同的存贮单元对应的地址号码显然是不一样的。当我们要把一个代码送到某个存贮单元（简称写入）或从某个存贮单元取出（简称读出）时，首先要告诉机器相应的存贮单元的地址是什么，即先把地址送往存贮器，然后由存贮器查找对应的存贮单元，“查到”以后才能把数据存放进去或取出来。在通常情况下，当存贮器从某个单元读出数据以后，这个单元保存的原内容应该仍保持不变，以便以后继续使用。反之，往某个单元写入新的数据时，用不着去清除该单元中原来的内容，这新的数据就自动地取代了原来的数据。

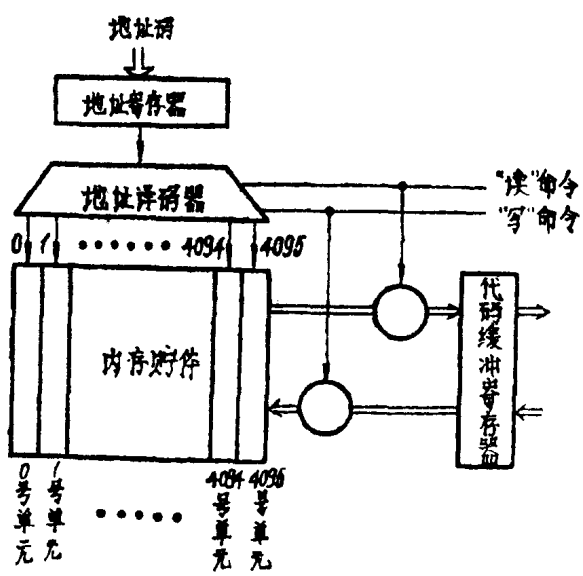


图 1.3　内存贮器框图

内存贮器由存贮体、地址寄存器、地址译码器、代码缓冲寄存器等组成，如图 1.3 所示。

存贮体一般由磁芯或半导体元件组成。磁芯存贮器即利用直径不到 1 毫米的环形磁材料（简称磁芯）来存放二进制数据。磁芯有两种不同的磁化状态，就利用它们分别表示两个不同的数码“1”和“0”。把16颗这样的磁芯排列在一起组成一个存贮单元，可以存放一个16位字长的数据或指令。

例如某一个磁芯存贮器容量为 4096 字，字长为16位，那么它总共需用4096×16＝65436颗磁芯。这些磁芯用漆包线按地址编排的规则穿织起来，组成一个整齐的磁芯体，它是磁芯存贮器的核心，磁芯存贮器写入或读出代码都是针对磁芯体进行的。

存贮器工作时，首先由控制器把地址送至存贮器的地址寄存器，然后根据存贮器写入数据还是读出数据发出相应的命令。如果是向存贮器读出数据，则控制器发出“读”命令，存贮器接到命令后就按照地址寄存器中的地址“查找”相应的存贮单元，这个查找地址的工作是由“地址译码器”实现的，译码器使对应于地址码的存贮单元（即要查找的单元）所有磁芯全都工作起来，把这些磁芯保存的代码被读出送往代码缓冲寄存器。此外，为了使这些磁芯仍然保持原有的代码，不致由于读命令而受破坏，所以在代码读出以后，紧接着还要把这个代码重新写回原存贮单元（简称“重写”），即磁芯在读出时的工作分为读出——重写两步进行。如果要向存贮器写入新的代码，则控制器发出“写”命令。为此，存贮器先把相应单元原来保存的代码“清除”掉，然后再把要写入的新代码（预先已放在代码缓冲寄存器中）存入该存贮单元中，即在写入时的工作分为清除——写入两步。

至于外存贮器一般由磁鼓或磁带，磁盘组成，它是主存贮器的后备，它就好象一个无穷大的“旅馆”，凡是主存贮器中不经常使用的数据和程序，都可以成批地放在外存贮器中，以便使主存贮器的单元空出来存放更为急需的东西。

三、控制器

控制器的功能是翻译指令代码，安排操作次序，并发出适当的命令到计算机各部分，以便执行机器的指令。它由指令寄存器，指令计数器，操作码译码器，地址形成部件，节拍发生器，操作控制部件等组成。如图 1.4 所示。

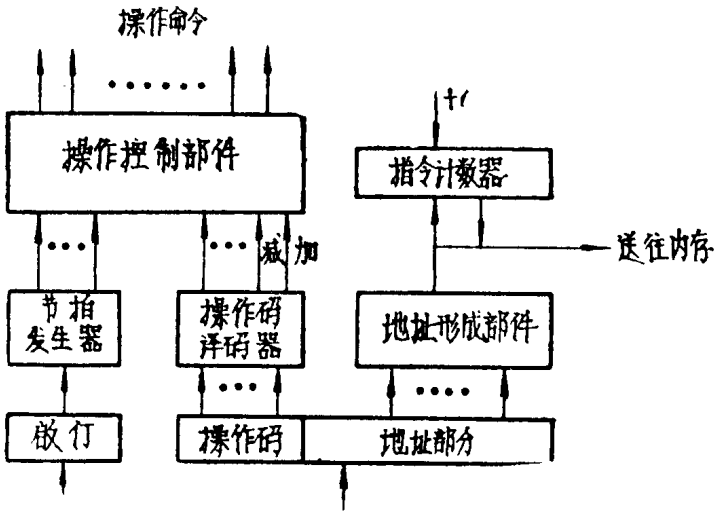


图 1.4　控制器框图

(1) 指令寄存器

它保存着当前计算机正在执行的指令，指令是指示机器执行操作的命令。每一台数字计算机中都规定了一定数量的基本指令，指令通常分为两部分：

指令格式：

操 作 码	地址部分

其中操作码指出运算的种类，如加或减等。地址部分用来指示参与运算的数据保存在什么地方或运算的结果存于何处。例如它可以是存贮器的地址或运算器中的寄存器号，说明参与运算的数据是放在存贮器哪一个单元或哪一个寄存器中，或者运算的结果存到存贮器哪一个单元或哪一个寄存器中。在数字计算机中，指令的操作码和地址部分等分别用二进制码来表示。

因而，指令寄存器也有操作码与地址两部分，操作码部分通过译码器指示指令的操作性质，地址部分用来产生访问存贮器的地址（简称操作地址）。

(2) 指令计数器

为了指明要执行的指令在内存的那一个单元，在控制器内设置了“指令计数器”，它记录着指令的地址。每当向内存贮器取指令时，指令计数器中保存的指令地址被送往存贮器的地址寄存器，并按此地址读出指令送往控制器，与此同时，指令计数器自动地转换成下一条指令的地址，为取出下一条指令作好准备。一般情况下，指令是按顺序一条接一条执行的，它在存贮器里的地址也是按顺序排列的。因此在这种情况下，每当执行一条指令时，指令计数器就自动加“1”，给出下一条指令的地址。但有时指令不按顺序执行，它的地址也就不按大小顺序变化，而是由控制器把一个新的指令地址直接送往指令计数器，下一条指令就按这一新的地址从内存贮器取出并执行。上述这种指令顺序的改变，称为“转移”。

(3) 操作码译码器

它的功能是解释操作码寄存器中操作码的性质，在译码器输出线的某一根上就出现一个相应的指令高电位。这个指令高电位控制操作控制部件，使它发出完成此指令所必需的操作命令。

(4) 地址形成部件

这一部件根据指令的地址部分形成“操作地址”，如参与运算的数据在内存的单元地址等等。

(5) 节拍发生器

计算机在执行每一条指令时，通常要分成若干个具体步骤来实现，其中每执行一步称为一个“节拍”。例如执行一条加法指令，必须先从存贮器取出这条指令，然后根据这条指令的地址部分产生“操作地址”，并按照这一地址从存贮器取出数据，最后把数据送往运算器进行运算。这些步骤是按时间先后次序一拍一拍进行的。为此，控制器设置有专门产生节拍的部件，这个部件按时间先后次序发出若干个节拍信号。

(6) 操作控制部件

操作控制部件向运算器、存贮器、输入——输出部件以及控制器本身的各个基本部件发出操作命令。其设计方案有两种，一种是常规设计，另一种是微程序设计。在常规设计中，它是由大量门电路组成的一套组合网络。在微程序设计中，利用常规的程序设计思想，把每一条机器指令的执行过程分解成一些基本的微操作，再按一定的次序编成微程序。机器执行

一条指令的过程，实际上就是执行相应的微程序的过程，这些微程序被存放在固定的或可变的只读存贮器中。

四、输入——输出设备

（1）输入设备

它能把数据和程序转换成电信号，并顺序地把它们送到计算机的存贮器里，目前常用的有光电输入机、卡片输入机、控制台键盘等。

（2）输出设备

它把计算机所产生的结果送往机外，它把从存贮器中取出的电信号转换成其他形式，如把数字符号印刷在纸上或者显示在荧光屏上等等。常见的输出设备有行式打印机，纸带穿孔输出机，卡片输出机，X—Y 绘图仪以及阴极射线管（CRT）显示器等等。

到此，我们可以描述一台简单的计算机组成，如图 1.5 所示。这个图的用意是表示计算机内的数据流，而特别留意控制数据流所需的“门”。在控制器部分中有一个指令寄存器（IR），一个程序计数器（PC），此外，还有加法器（ADDS），一个存贮器（MEM）和两个存贮器寄存器：地址寄存器（SAR）和代码缓冲寄存器（SBR）。

计算机为执行一条指令，它必须从存贮器取出这条指令，然后根据它的操作码执行所需的微命令。这两部分周期的前半周期称为“取指令”周期，对于所有指令都是一样的，后半

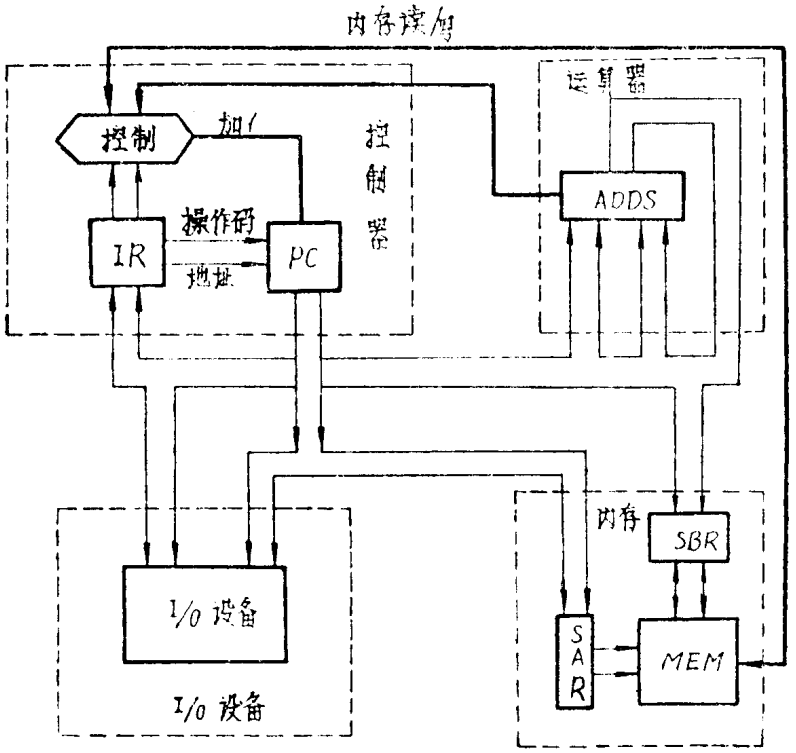


图 1.5　简单的计算机组成

周期称为“执行”周期，取决于执行的是什么指令。

现在描述执行指令的大致工作过程如下：

(1) 开始执行程序时，控制器保存着第一条指令的地址，这个地址存放在“程序计数器”中，它用来指示当前将要执行的指令存放在内存贮器的哪一个单元内。

(2) 控制器把上述指令地址送往内存贮器，并发出“读”命令。由存贮器读出该指令并送回控制器，保存在“指令寄存器”中。

(3) 指令寄存器分析指令的操作性质。

(4) 根据操作性质向主存贮器、运算器等有关的部件发出操作命令。

(5) 在需要由主存贮器提供运算数据时，控制器向主存贮器发出“取数命令”，并根据指令的地址部分，决定数据所在的主存贮器的单元地址（也送往主存贮器）。

(6) 主存贮器读出的数据直接送往运算器，与此同时，控制器命令运算器对数据进行指令规定的运算。

(7) 一条指令执行完毕后，控制器就要接着执行下一条指令。为了把下一条指令从存贮器取出来，通常控制器要把“程序计数器”加“1”（指示下一条指令所在的主存贮器单元地址），然后命令主存贮器按照这个地址取出下一条指令。

控制器不断地循环上述执行指令的各个阶段[(2) 到 (7)]，每重复一次，就执行一条指令，直到程序全部执行完毕为止。

§1.2 电子数字计算机的应用和解决实际问题的过程

1.2.1 电子数字计算机的应用和特点

在最近十年间，计算机应用的领域迅速扩大。到目前为止，它不仅可以应用于科学设计与计算，而且还应用于商业和政府部门的业务管理，应用于厂矿企业生产自动过程的自动控制，应用于国防上对火炮武器的控制；此外，它还可以帮助医生为患者诊断和治疗，帮助人们学习等等。现在，我们来看几个计算机应用的例子。

设计问题：对设计工程师来说，计算机是非常有用的计算工具。例如，一架超音速飞机的机翼设计与许多因素有关。首先设计工程师把它归结成数学问题，这时，每个因素满足一组数学方程式，然后用程序设计语言描述每个因素，再用计算机来求解这些方程式。

科学和实验室实验：在科学和实验室实验方面，使用计算机求解和保存来自各类电子敏感器件的信息。在遥测学系统中，计算机特别有用，因为在那里信号必须迅速记录下来，否则就会丢失掉。这些应用中对固定的条件和动态的状态都需要快速和精确的处理。

过程控制：计算机对于自动地制造产品和检验产品是一种有用的工具。可编写程序的计算机能开动和控制铣床，车床和其它机床，比人的反应更快更精确。计算机按照程序使用特殊的敏感器检验正在制造的部件，当需要时可调整机床，若这个部件不合格，计算机可以丢弃它，同时开始检验下一个部件。

模拟训练：在真实的条件下训练一大批人去架驶商用飞机、控制卫星或操纵宇宙飞船，往往是浪费的，危险的或做不到的。计算机能为受训者模拟对他的动作反应的所有条件，并报告训练的结果。受训者可接受许多小时作业训练，而对他本人，别人或昂贵的设备没有任何危险和损害。

计算机能有如此广泛的应用是因为：

（1）能重复操作：一台计算机能执行类似的操作几千次不会感到麻烦、疲劳。

（2）速度：计算机以极高的速度处理信息，速度与设计者和程序员的技巧直接有关。现代计算机解决某类问题比熟练的数学家快百万倍。

（3）灵活性：通用计算机可编制程序来解决各类问题。

（4）精确度：计算机能计算出按程序员要求的精确的结果。

（5）具有“记忆”能力和逻辑判断能力：计算机的存贮器使计算机具有类似“记忆”的能力，它保存着大量的解题程序，数据等信息。计算机还可以进行各种逻辑判断（如判断数据的大小、性质），并能根据判断的情况自动决定计算机应执行什么命令。

总之，现代数字计算机能够存贮信息，高速地执行计算，并能根据结果作出判断以对给定的问题获得最后的解。然而，计算机不能完成尚未给计算机命令的那些任务，计算机所执行的每一步工作，预先必须由程序员安排好。也就是说，程序员对所要解决的问题事先要有一个明确的计算步骤和操作过程，并且要把它用机器所能识别的语言表达出来。即是把解法写成为机器能完成的一系列基本运算。给机器指出进行何种运算及参加运算的是那些数的最简单的信息称为指令。每一种指令能使机器执行一项基本操作。为使计算机获得给定问题的解所需的指令序列称为程序。因此，程序员对所要解决的问题写出一个程序。程序是依据解题的某个计算方法（称为算法）而编写的，存放在计算机存贮器里，计算机执行它，就能解决所求的问题。

1.2.2 程序设计的几个阶段

为了用计算机成功地解决一个问题，程序员要经过七个程序设计阶段：

1. 明确求解问题的意义。
2. 确定算法，选取最合宜的解题方法。
3. 确定输入数据和输出结果的格式。
4. 分析算法，画出程序框图和写出有关程序的资料。
5. 用程序设计语言编写程序代码。
6. 查错和调试程序。
7. 文件编制。

问题的确定　根据所求解问题对计算机提出的要求来确定问题的任务。如果问题没有弄得相当明确，那么大量时间和精力可能被浪费。当问题是求四个数之和时，任务是显而易见的。然而对监视和控制半导体器件的性能测试等问题，在解决这类问题之前，必须将问题的精确意义搞清楚，并把计算或控制对象的物理过程或工作状态，用数学方程或用文字描述归纳为数学问题的形式（通称为数学模型）。

确定算法　由于计算机只会做加、减等这些最基本的算术运算及某些简单的逻辑运算，因此要将上阶段归结的数学问题转化为近似计算公式。为解决一个问题可能存在许多种方法。选择何种方法依赖于使用的是什么样的计算机系统，以及依赖于所求问题的精度和速度等等要求。

确定输入—输出数据的格式　根据问题的要求应明确输入—输出些什么数据，它们的格

式是怎样的。如果数据有很好的结构，就会简化计算机的处理。

分析算法，画出框图　将解题的方法按照逻辑划分成计算机可以执行的步骤，然后用图形把它表示出来，这种图形通常称为框图。关于框图下面将作进一步说明。

用程序设计语言编写程序　根据程序框图，程序员就可用程序设计语言开始编写程序。这个阶段一般称为“程序设计”。但实际上编码阶段仅是程序设计过程的一部分。这里所说的程序设计语言可以是某种具体计算机所能识别的语言，称为机器语言。也可以是用符号表示每一条指令的汇编语言，以及高级程序设计语言，例 BASIC, FORTRAN, ALGOL, COBOL, PASCAL等。

当程序编完之后，程序被存放在计算机内存中，就可以开始解题。然而，程序设计过程到此结束是很少的，因为开始编写的程序可能存在错误。

程序查错和调试　这阶段要求程序员一步一步地跟踪指令的流程，找出程序中可能存在的任何错误。程序员不能象日常生活中那样告诉计算机：“你知道我的意思是什么？”。直到给出一组指令，让计算机正确地执行这些指令，它才知道这是什么意思。如果遗漏掉某些必要的指令或者编码不正确，结果就可以出乎意外，这些错误必须予以发现并加以纠正。

文件编制　适当的文件编制是程序设计的重要部分。文件编制不仅有助于设计者进行测试和排错，而且对程序以后的使用和扩充也是必不可少的。文件编得不好，程序就难于维护，使用和扩充。文件通常包括对工作程序的说明、框图、程序清单、注释、内存分配图和程序库等。

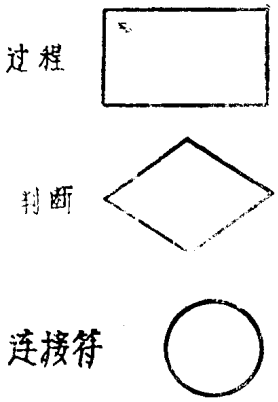
1.2.3 框 图

四个数加在一起这个简单的题目，只要几步就能解决。为使用计算机解这个题目，一个程序员坐在桌子旁写三、四条指令就行了。然而，他用纸和笔算出这个题目比起使用计算机化的时间更少。为此，要求程序员解算的问题往往比求四数之和复杂得多。因为，使用计算机的目的是解决人们不容易解的或者费时的问题。

当计算机解算一些比较复杂的问题时，程序包含许多步，程序经常是较长的，编制程序就比较困难。尤其解决用文字或数学方程描述的问题时，写程序是相当困难，为了简化程序编写所以采用框图。框图是对给定问题的图形表示法。它指示了计算机应该执行操作的逻辑次序。程序的框图对于程序员确定解题的方法以及编写程序指令有巨大帮助。另外，当程序员检验所写的程序错误时，框图也很有用。

框图基本上是框和线条的集合体，框指示该做什么，而线条指明框的顺序。框具有各种形状，表示程序中要执行的动作。图 1-6 描述了各种框符号，这是由 ANSI/ISO（国际标准化组织）规定的符号。

椭圆符号表示程序中某个终端点。使用它指明程序的开始，停止或中断。把适当的字记入这个符号中。



矩形符号表示某个过程的功能。这过程可能是为执行一个给定的任务所需的一条指令或一组指令。所要执行的任务的简短叙述记入这符号中。

菱形符号用于表示程序在这一点要进行比较、判断，决定程序的流向。检验条件记入符号中，并使用检验的各种可能结果标出在这一点程序的各种流向。

一个小圆内标以一个标识符。使用它把具有相同标识符的两个小圆，一个终止程序的流向，另一个开始程序的流向的圆连接起来。于是，允许框图画在不同页面上，通过它连接起来，也允许消除流线的交叉。

图 1-6　框图符号

下面，给出框图的三个例子。例 1 是把三个数加在一起。例 2 是把三个数按递增次序排列。例 3 是计算平方根。

例 1　顺序程序设计

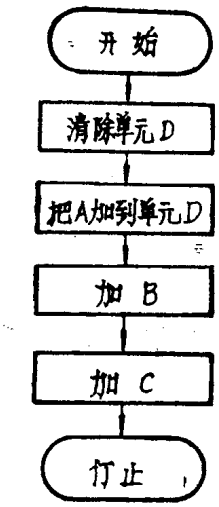


图 1-7　三个数相加

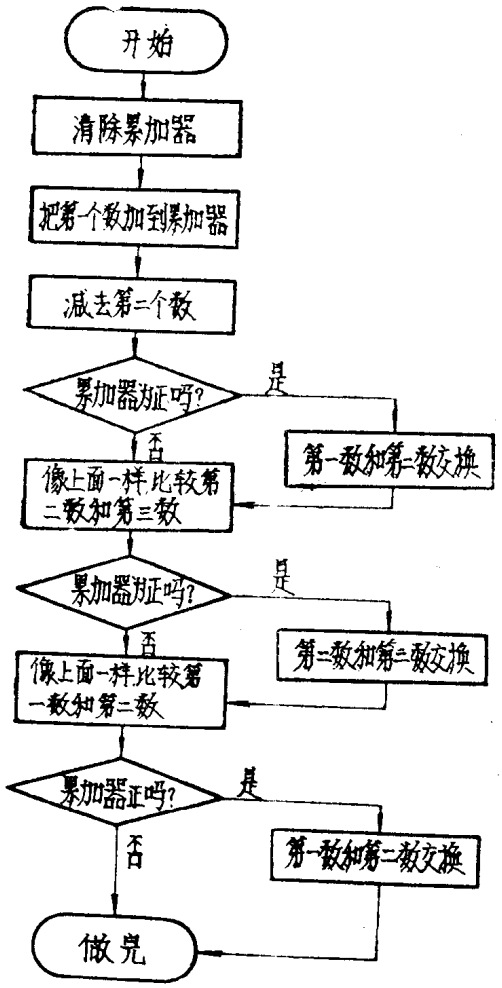


图 1-8　以递增次序排列三个数