

电子计算机原理

(非电专业)

上海交通大学计算机科学技术教研组

1980年9月

目前,我校各非电专业正在陆续开设计算机课程。今后,不仅是理工科大专院校,而且,文科、医学、艺术等方面的专业,也将会开设计算机课程。我们认为:对于非计算机专业计算机教育的目的主要是培养计算机开发应用方面的人材。计算机课程的设置必须软硬兼顾,让学生既懂总体结构与硬件组成原理,又懂算法语言与程序设计。而一般非计算机专业的计算机原理教材基本上是本专业原理教材的一个浓缩体。由于过多、过细的讲述单元线路,且缺乏总体结构概念,不仅不适应现代计算机的发展,还给非电专业学生未经予修电的基础课程直接学习计算机原理造成困难。为了改变这种状况,我们写了这本讲义。并准备在应用数学系及全校计算机原理课选修班使用。

书中,首先从讲述计算机系统的基本概念开始,提供了为理解计算机硬件所必需的基础知识。在讲解原理的过程中,尽量避免过多的单元线路而是侧重于理解逻辑功能,对于控制的组织不是围绕某一种具体的机型转,而是通过一台假想机的动作原理,使读者有可能建立一个简单而又完整的整机概念。同时,还把组成原理与系统结构结合起来,采用深入浅出的叙述方法,介绍了存储器分层原理、缓冲存储器、虚拟存储器、并行处理以及通道技术等系统技术。从中,有助于了解现代计算机的发展水平。最后为了使读者在学完本课程后,对于软件与硬件相结合的功能有一个具体的理解。编入了“机器语言程序及其执行”一章。全书文字力求简明通俗,并采用了较多的图解说明,对于具有高中程度电气知识的人,都能理解本讲义所写的计算机工作的原理。

由于教学之急需,时间极其仓促,加之水平不高,错误与不足之处在所难免,恳请读者批评指正,以便修改。

编者

1980年9月



10319216

目 录

第一章 电子计算机的基本概念	
一、概述	1-1
二、电子计算机的基本原理和简单结构	1-1
三、程序设计语言	1-4
四、处理方式	1-7
五、操作系统	1-9
六、语言程序的处理过程	1-10
第二章 信息的表示方法	
一、计算机与二进制代码	2-1
二、数制及其相互转换	2-2
三、字符、字节和字	2-7
四、数值的表示方法	2-8
五、几种常用的代码	2-12
六、数据误差的检出	2-16
第三章 电子计算机的数字基础	
一、三种基本逻辑运算	3-1
二、逻辑代数中的基本关系式	3-3
三、逻辑函数的化简	3-7
四、逻辑代数的应用	3-12
第四章 运算器	
一、概述	4-1
二、信息的寄存	4-1
三、信息的传送	4-3
四、移位与比较	4-6
五、定点数的加减运算	4-8
六、浮点数的加减运算	4-14
七、乘法运算	4-15
八、除法运算	4-17

第五章 存储器与存储器分层结构

- 一、存储器概述 5-1
- 二、磁心存储器的工作原理 5-4
- 三、半导体存储器 5-12
- 四、磁表面存储器 5-15
- 五、存储器的分层原理 5-18
- 六、缓冲存储器 5-19
- 七、虚拟存储器 5-21

第六章 控制器

- 一、控制器的作用 6-1
- 二、指令格式 6-1
- 三、指令类型 6-6
- 四、指令的寻址方式 6-8
- 五、控制器的基本组成 6-14
- 六、简单计算机的操作过程 6-25
- 七、中断控制 6-32
- 八、并行处理 6-35

第七章 输入输出设备

- 一、概述 7-1
- 二、纸带阅读机 7-2
- 三、纸带穿孔机 7-5
- 四、卡片阅读机 7-7
- 五、卡片穿孔机 7-8
- 六、打印装置 7-11
- 七、光学字符阅读机 7-15
- 八、磁性墨水阅读机 7-19
- 九、显示设备 7-20
- 十、控制台 7-25

第八章 输入输出控制（通道技术）

- 一、输入输出控制组织 8-1
- 二、应用通道技术的计算机系统 8-4

三、通道的动作原理	8-8
四、输入输出指令	8-11
第九章 机器语言程序及其执行	
一、指令种类	9-1
二、程序的编制	9-4
三、程序的执行	9-11

第一章 电子计算机的基本概念

一、引言

随着信息处理技术的发展，电子计算机（computer）的应用领域日益扩大，特别是近几年来，电子计算机已不是单纯高速进行加、减、乘、除等四则运算的机器，而是能够用于范围非常广泛的事务处理。电子计算机的普及应用，对其他各个科学技术的发展起了很大的促进作用，成为现代化水平的重要标志。

电子计算机是近三十年发展起来的一门新兴学科。世界上第一台计算机 ENIAC 是 1946 年诞生的。它用了近 18,000 个电子管，整机体积长 100 英尺，高 8 英尺，重 30 吨，功耗 150 千瓦，而速度只有 5,000 次/秒。自此以后，经历了第一代（电子管计算机）、第二代（晶体管计算机）、第三代（集成电路计算机），现在已经跨入第四代（大规模集成电路计算机）。计算机技术正向三大分支发展：巨型机（目前世界上正在研制百亿次/秒机）、微型机（已经可以做到在 $\times 6$ 平方毫米的硅片上集成一台计算机）、计算机网络（与计算机通信好象打电话一样方便）。

电子计算机之所以有这样强大的生命力，主要是它具有运算速度快、计算精度高、具有记忆和逻辑判断能力等特点，能够模拟人脑的部分功能，人们把它比作人工“头脑”。电子计算机的发展大大减轻了人们的脑力劳动，使得人们可以从那些重复而又繁重的劳动中解放出来，把精力用于更加复杂的创造性劳动。在加速实现四个现代化的今天，普及电子计算机技术，普及电子计算机在科学、工业技术和国民经济其它各个部门的应用，将直接关系到四个现代化的进程。

二、电子计算机的基本原理和简单结构

电子计算机是怎样工作的呢？在回答这个问题以前，先假设有人用算盘来算题目，这个题目是许多基本运算组合起来的复杂运算。那么，他需要先将参加运算的原始数据和如何进行运算的步骤写在纸上，然后按照计算顺序用算盘逐一计算。在计算过程中，他用了哪些东西呢？——纸、笔、算盘、他自己的眼睛、手和脑，如果将这些东西所承担的工作用电子线路和一些特殊设备来代替，并作成一个机器，那

么，只要将计算的数据和预先编好的计算顺序送入机器，机器便能自动地给出最后的运算结果，这就是电子计算机了。

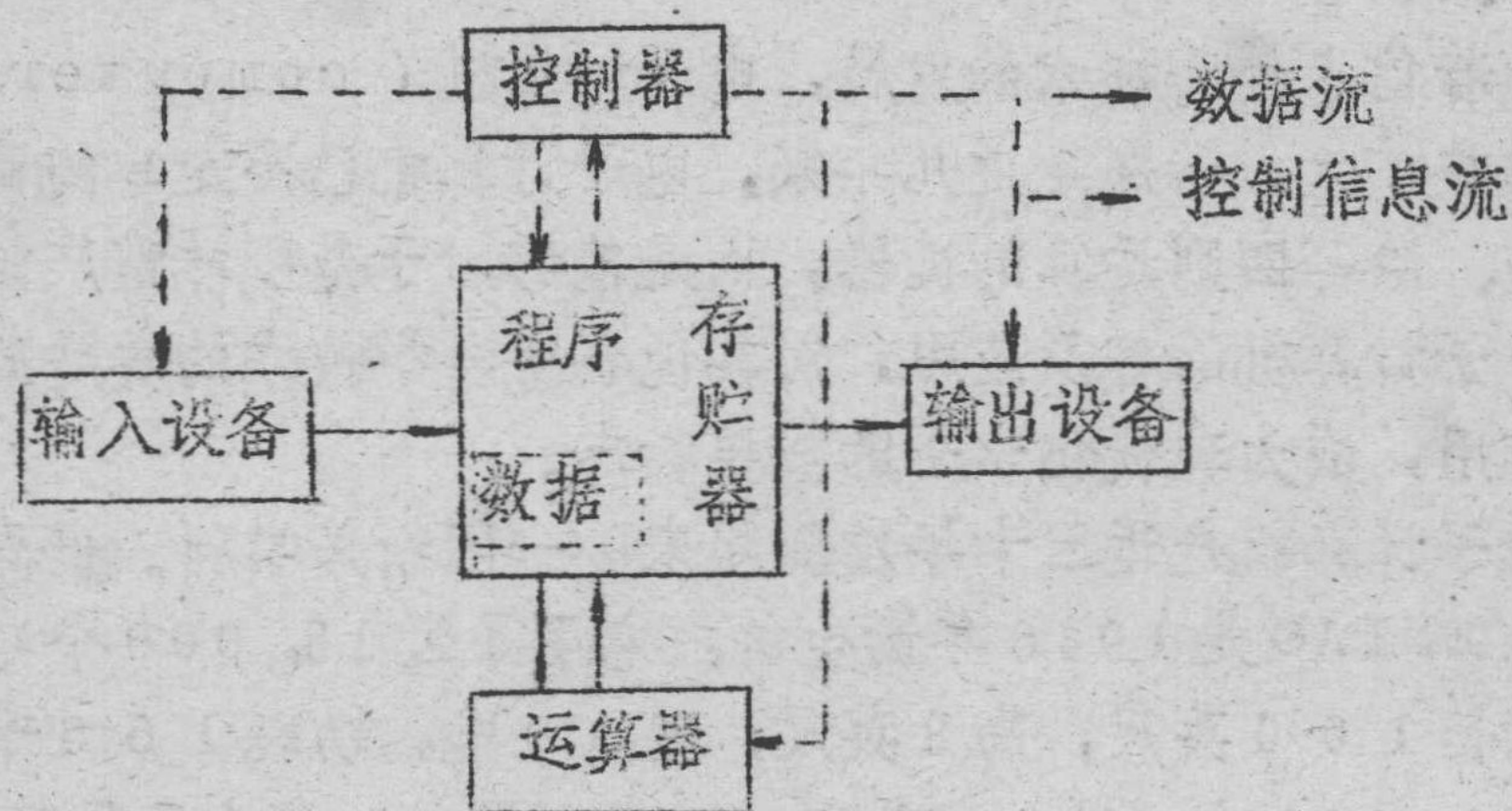


图 1-1 电子计算机的简单结构

(一) 运算器 (arithmetic unit)

运算器是根据计算顺序进行计算的部件，相当于算盘。它能够接受由存贮器送来的数据，进行加减乘除等四则运算、逻辑运算及数据传送，并能把运算的结果送回到存贮器保存起来。运算器通常是由加法器、移位器、运算寄存器等组成。

(二) 存贮器 (memory unit)

存贮器的主要功能是存放大量数据和计算顺序（称程序），相当于人的头脑或纸。其中，数据就是参加运算的原始数据或运算的中间结果及最后结果，程序是用来控制机器每一步该做什么的一连串命令，这些命令在计算机中习惯上称为指令，由一系列指令组成的解题顺序，我们称它为程序。存贮器可以存入数据或指令，也可以取出数据或指令，即存贮器具有记忆能力。存贮器的结构相当于一个旅馆，旅馆中的每一个房间相当于存贮器的每一个单元，象房间号码一样，存贮器的每一个单元也都由它们自己的编号来确定。这个编号称为“地址”。对数据或指令的取出，只要按确定的地址去寻找（称寻址）就可以了，这与去旅馆找人必须先知道房间号码一样。

存贮器一般用磁芯来记忆信息，随着大规模集成电路（LSI）的发展，出现了半导体存贮器取代磁芯存贮器的趋势。

(三) 控制器 (control unit)

控制器是协调机器各部分工作的枢纽，相当于人的头脑。当计算机工作以前，使用者把预先编好的计算程序和数据存入存贮器，启动以后，控制器从存贮器按规定的程序取出指令，由控制器分析和解释指令的含意，向运算器或其他部件发出相应的控制信号，指挥各部件执行指令所规定的操作。一个程序对应着一个问题的处理。任何复杂的问题，总是可以由执行一系列只有单一功能的指令来完成。指令一般由二个部分组成，一是指定机器做什么，如加法、减法等，称操作码部分；二是指定谁做，即要指明作为运算对象的操作数在存贮器的“地址”，称地址码部分，如图 1—2 所示。各种机器按其功能不同，它们拥有的指令种类也各不相同，我们把一台机器的全部指令，称为这台计算机的指令系统。当控制器从存贮器取得指令后，立即翻译指令的操作码告诉机器该做什么，并根据地址码从存贮器取得参加运算的数据，然后进行运算。当一条指令刚执行完，控制器又从存贮器取来下一条指令，再重复上述工作，直到一个程序执行完毕。

指令

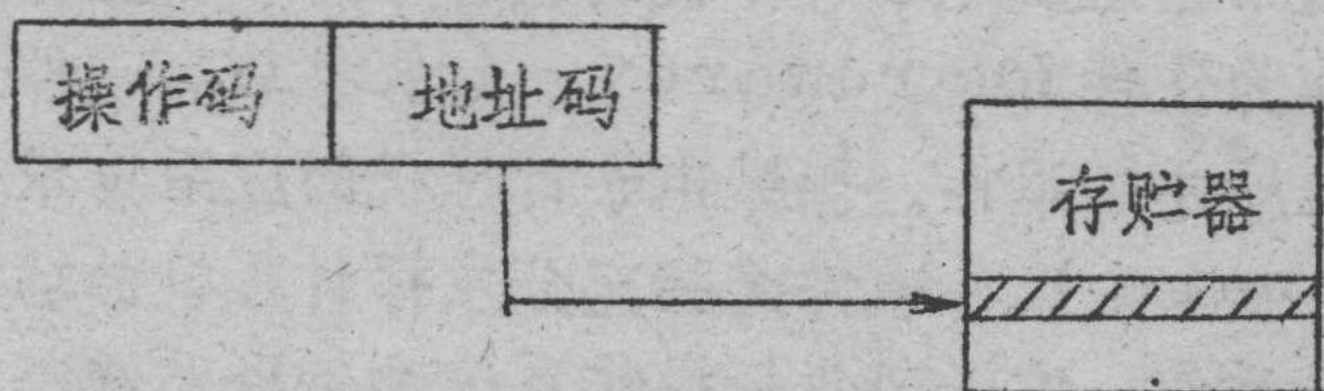


图 1—2 指令的形式

由上可知，有了运算器、存贮器和控制器，计算机就已经具备了自动计算的能力；但是，实现自动计算的关键，还是由于程序和数据一起被存贮器记忆起来了，即“存贮程序”。从这个意义来说，计算机应该称为“存贮程序式计算机”。“存贮程序”这一思想是 Von Neumann 在 1945 年提出来的，尽管计算机经历巨大变化，从本质来看，主流还是 Von Neumann 型计算机。

计算机除了上面三种设备外，还需配备输入输出设备，以便输入计算程序和数据，输出计算结果。

(四) 输入设备 (input unit)

输入设备是计算机从外界把程序和数据装入存贮器的设备，相当于人的眼睛和手。输入设备通常采用光电输入机，将原始数据及计算程序穿孔在纸带上，然后通过光电输入机将信息转换为脉冲或电位的形式送入计算机。

(五)输出设备 (output unit)

输出设备的任务主要是把计算结果打印出来，相当于人的手拿笔在纸上写字，一般采用控制台打字机或宽行打印机。

电子计算机就是由上述五种设备组成的。它们之间的关系如图 1—1 所示。

在现代计算机中，通常把运算器和控制器合在一起称为中央处理机 (Central Processing Unit 简称 CPU) 或机器本体。把能够直接由 CPU 指定“地址”写入或读出信息，在计算机处理时存贮程序和数据的存贮器叫主存贮器 (或称内存)。为了补充主存贮容量不足的存贮器叫辅助存贮器 (或称外存)，一般采用磁带、磁鼓、磁盘等。辅助存贮器和输入输出设备 (Input/output 简称 I/O) 也可合在一起称为输入输出外围设备。

我们通常把组成电子计算机的设备本身 (即主存贮器、CPU、I/O) 称为硬件 (hardware)。计算机单有硬件还不能工作，还必须要有程序才能工作。操纵电子计算机的应用技术—程序总称为软件 (Software)。软件是指充分发挥计算机的功能、方便用户而使用的程序系统，它一般包括操作系统和编译系统等。通常所说的计算机系统，应该指硬件和软件合在一起组成的系统。

三、程序设计语言 (programming language)

语言是人与人或人与机器或机器与机器交换信息的工具。在计算机产生的初期，人们直接用计算机的指令系统来编写程序，计算机的指令就是机器的语言，用机器语言来编写的程序称为手编程序。但是由于这种机器语言是用 0 和 1 来表示的，因此，对人来说书写和辨认非常困难，编程序的工作量相当大。为了克服以上缺点，人们开展了程序自动化的研究，即让计算机代替人一起从事一些程序设计工作。早期，人们使用汇编语言，汇编语言使用了一些有助于人记忆的符号来代替数字代码，并省去了人工分配存贮单元之类的繁琐劳动，编制

程序就方便得多了。但是汇编语言并未摆脱对具体机器的依赖性，与手编程序仍有很大相似之处。随着程序自动化的进一步发展，产生了许多更接近于数学表达式和用英语编写的算法语言（又称高级语言），这类算法语言，目前已被人们广泛地使用。

根据语言的使用目的及处理形式，我们可以把程序语言分为通用程序语言和面向问题语言。如图3中，通用程序语言是适合于解决范围比较广泛的问题的语言，而面向问题语言是专门为了描述某一方面问题而设计的语言。

(一) 机器语言和汇编语言

机器语言是电子计算机能够直接翻译其意义，并执行相应动作的语言。根据计算机的种类与形式，机器语言也不尽相同。机器语言由于把操作码和地址码全部用数字代码表示，很难记住；而汇编语言虽然接近于机器语言，但是操作码和地址码可以用符号书写。通常，一条汇编语言指令，如表1-1所示，可以翻译成为一条机器语言指令。我们把用汇编语言书写的程序翻译为机器语言程序的程序称为汇编程序。

表1-1 汇编语言和机器语言比较

	机器语言	汇编语言
加法指令	0100	A(Addition的字头)
减法指令	0010	S(Subtraction的字头)
乘法指令	0010	M(Multiple的字头)
取数指令	0011	L(Load的字头)
存数指令	1000	ST(Store的字头)

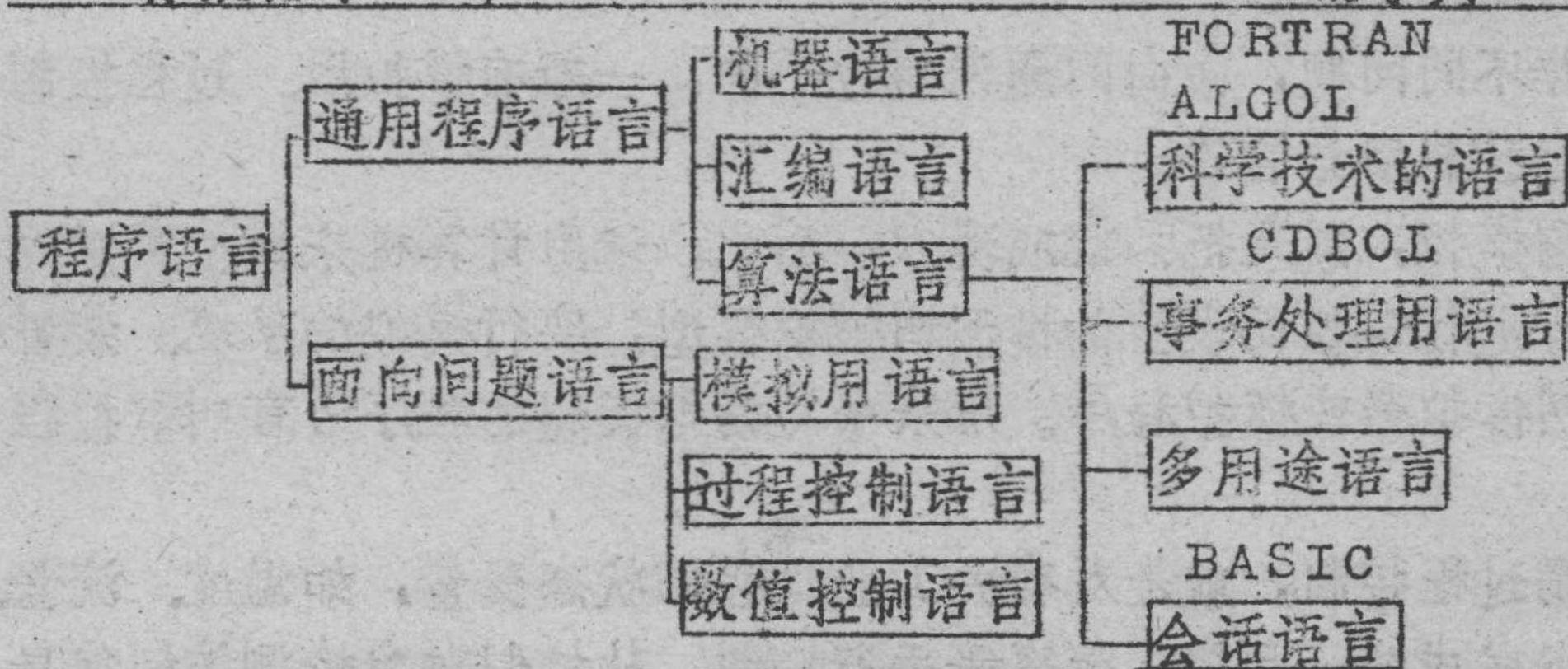


图1-3 程序语言的种类

(二) 算法语言

汇编语言与机器语言相比，编制程序虽已省力得多，但是还有更便于程序设计的语言，即算法语言。

算法语言是一种比较接近于人的语言结构的程序语言，它与机器语言差别很大，如表 1—2 所示。算法语言对机器语言指令而言，具有一对多的关系。如表 1—2 中，一条算法语言的语句可以翻译成为 4 条机器语言指令，我们把用算法语言书写的程序翻译成为机器语言程序的程序叫编译程序。

表 1—2 机器语言、汇编语言和算法语言比较

机 器 语 言		汇编语言	算法语言
操作码	地址码		
0011	000100	L A	$A = A \times B \times C$
0010	000110	M B	
0100	000101	A C	
1000	000011	S X	

目前，一般采用的算法语言，如图 1—3 所列，有 FORTRAN (Formula Translator)、ALGOL (Algorithmic Language)、COBOL (Common Business Oriented Language)、OL/1 (Programming Language one)、BASIC (Beginners All-purpose Symbolic Instruction Code) 等。其中 BASIC 语言是一种解释性语言，它是与计算机一边交换信息一边编制程序并执行程序的语言，亦称会话语言。

(三) 面向问题语言

根据不同问题，面向问题语言也不同，一般有模拟用、过程控制用及数值控制用语言等。

所谓模拟，就是把系统或现象模型化，采用计算机来模拟实际的情况。为进行模拟而设置的模型叫模拟模型，实行模拟的系统、装置或程序叫模拟器或模拟程序。用来书写模拟模型的程序语言叫模拟程序语言。

所谓过程控制，就是对有关工业过程的状态变量，如温度、流量、压力、组成成分、品质、效率等进行控制。从控制对象检测控制信息，

控制信息交给计算机进行处理，再向控制对象发出控制信号。用来描述过程控制的语言，我们称作过程控制语言。

所谓数值控制，就是对应于工作物的刀具的位置的控制，数值控制的对象一般是工作母机，但也可用于制图、布线及产品检验等工作。用来描述数值控制所使用的语言称为数值控制语言。

四、处理方法

(一)成批处理 (*batch processing*)

在早期，由于计算机的处理速度较慢，存贮器容量也很小，程序员在计算机上工作，基本上采用手工操作方式。他先将纸带装上光电输入机，把程序和数据输入，然后通过控制台开关（或键盘打入命令）启动程序，待程序运行完后，从输入机上卸下纸带，取走打印结果，然后让给下一道程序，再重复上面的步骤。采用这种工作方法，计算机很大一部分时间处于等待状态，因此，机器的利用率很低。随着计算机处理速度的飞跃提高，为了提高机器的使用效率，减少手工操作时机器等待的时间，可以先把一批作业（*Job*，我们定义作业为提交给机器的基本的独立工作单位，如一道解题程序及其数据）记录在磁带上，在主存贮器常驻一个程序（即始终放在主存贮器，并不调出），称为监控程序（*monitor*）。此监控程序从磁带上调入第一个作业，并启动这个作业。当这个作业结束时，并不停机，而把控制转给监控程序，此程序立即从磁带上选取下一个作业，调入并启动它。这样从一个作业自动过渡到另一个作业的工作方式称为成批处理。成批处理是数据处理最普通的方法。成批处理为了把数据集中在一起处理，到处理的时候需要一定的时间间隔，所以一般按业务情况以月、周、日为单位把数据集中起来处理。

(二)分时处理 (*time-sharing processing*)

分时处理就是一台计算机连接多台终端设备（如控制台打字机、键盘显示设备等，距离比较远的用户可以通过通信线路使用这台计算机），对使用处理机的时间进行轮流分配，以发挥一机多用的作用。图1-4示出了分时系统的一个例子。

从控制台打字机输入的程序和数据，存放在称作用户文件（指作为一个处理单位的逻辑上有关系的数据的集合）辅助存贮器里，需要

时可以调出使用。分时系统的实质，就是中央处理机分时使用，如 1—5。

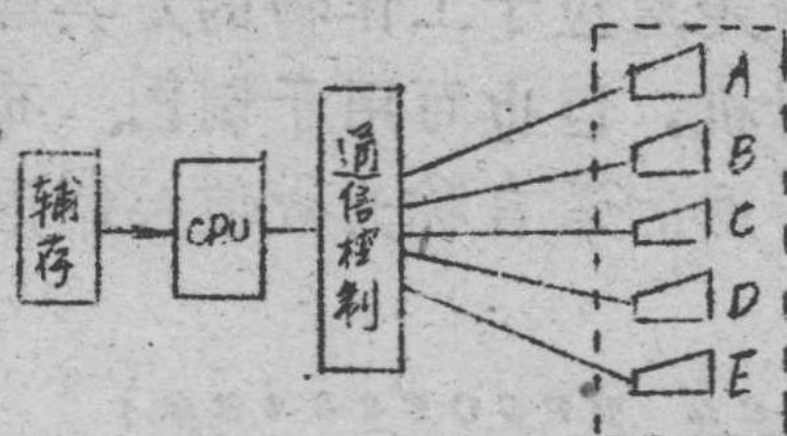


图 1—4 分时系统

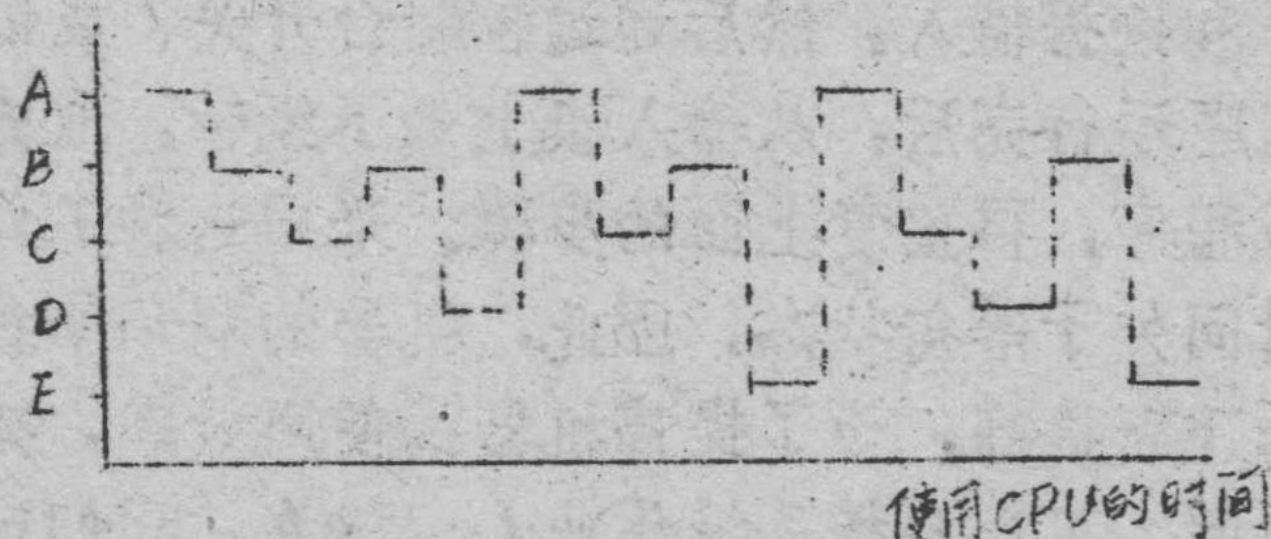


图 1—5 分时原理

分时系统的主要特点是很多联机用户可以同时使用一台计算机。还有，用户可根据程序运行情况进行人工干预，即用户可与机器进行“交互式会话”。此外，用户之间感觉不到别人也在使用计算机，所以使每个用户认为自己单独占用了计算机。

分时系统最初主要用在科学计算上，目前已大量用于事务处理。采用分时方式有许多优点，例如任何时候都能使用计算机，能方便地利用别人编的程序，还可以用小型计算机作为分时系统中的一个终端设备，这样，很多没有大型计算机的单位和部门可以通过终端使用大型计算机，从而大大节约了费用。

(三) 实时处理 (real time processing)

所谓实时处理，就是把要求处理的信息输入计算机后能够在短时间内响应的处理方式，它一般用于需要很快得出结果的任务。例如工业过程控制、宇宙飞船的控制、飞机及列车座席预约业务、银行存款业务等。

输入输出设备在中央处理机直接控制下工作的状态称为在线或联

机 (on-line), 反之, 不直接控制的状态称为离线或脱机 (off-line)。通常实时处理是在线方式工作的。

五、操作系统 (operation system)

从上面介绍的计算机几种处理方式可以看出, 不管哪种处理, 都存在一个调度使用计算机的问题。这个任务是由操作系统完成的, 操作系统相当于计算机系统的总调度员。操作系统是一种程序, 执行它后, 就可以控制计算机的各种操作。有了操作系统以后, 用户直接使用的就不再是硬件本身, 而是操作系统了。所以, 操作系统是为了提高计算机的利用率, 方便用户使用计算机, 实现计算机系统运行管理自动化而配备的一种软件。

(一) 操作系统的组成

根据机器种类不同, 操作系统的组成也各不相同。这里举一个简单的操作系统的例子, 说明一下操作系统究竟是怎么回事。如图 1-6 所示, 它是由控制程序和处理程序两部分组成的。

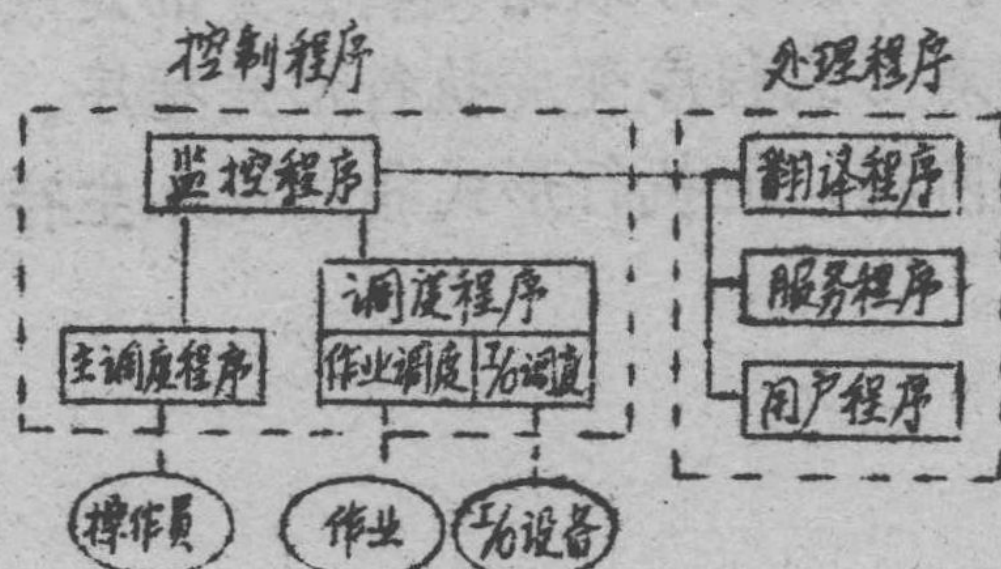


图 1-6 一个简单的操作系统组成

控制程序用来管理程序的执行和出错处理, 它由下面几个程序组成: 监督控制处理程序的执行和计算机硬件动作的监控程序; 与操作员交换信息的主调度程序; 对作业的开始和终止进行管理的作业调度程序以及对输入输出设备进行管理的 I/O 调度程序。

处理程序是在控制程序的控制下进行翻译和计算等工作。处理程序由下列几个程序组成: 把用汇编语言 (或算法语言) 书写的程序翻译为机器语言程序的翻译程序 (汇编程序或编译程序); 承担数据分类与归并等工作的服务程序以及用户程序 (由用户自编的)。

(二) 程序的处理过程

用计算机处理如FORTRAN之类程序的作业，如图1—7可分为三个阶段进行。

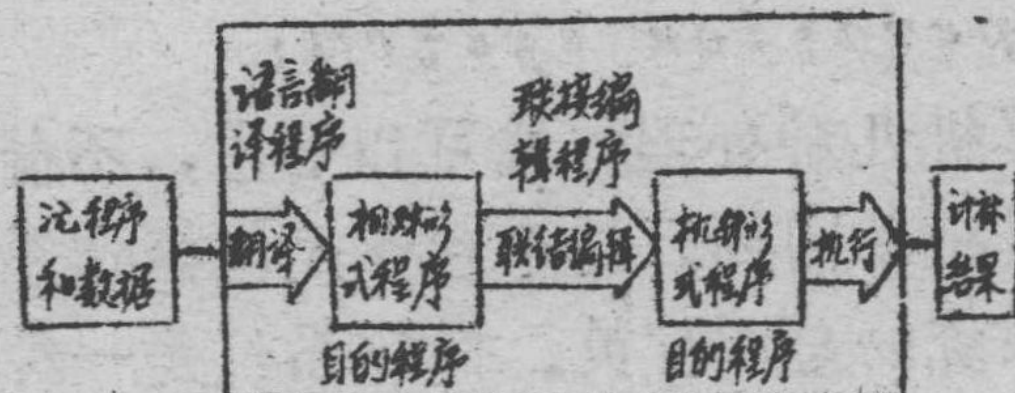


图1—7 程序的翻译与执行

第一，程序的翻译阶段，用FORTRAN书写的程序（源程序）可以通过纸带输入机输入，然后用FORTRAN编译程序翻译成机器语言程序，翻译好的程序称为目的程序。刚翻好的程序，虽然指令已变成机器指令，但是地址没有按机器中主存的地址分配，所以那样的程序是不能执行的，称为相对形式程序。

第二，程序的联接编辑阶段，是把几个相对形式程序联结起来，编成一个能执行的程序。承担程序联接编辑任务的是联接编辑程序，经过联接编辑可以把相对形式程序变成执行形式程序。

第三，程序的执行阶段，把执行形式程序送入主存贮器执行。

第二章 信息的表示方法

一、计算机与二进制代码

电子计算机是处理编码信息 (*coded information*) 的机器。所谓编码是离散地表示信息的一种方法。电子计算机采用二进制代码。代码的每一位只表示“1”或“0”中的一个状态，它是信息处理的最小单位，称为位 (*binary digit* 简称 *bit*)。

通过简单的代码组合，可以正确地表示对应的状态，这是编码的最大优点。例如 *1 bit* 的信息能表示二个状态中的任一个，而 *2 bit* 的信息，用二进制数表示，就能够表示 00、01、10、11 四个状态中的任意一个。一般说，*n bit* 信息可以表示 2^n 个状态中的任意一个。

如果我们用“0”和“1”的组合来表示 26 个英文字母，则由 $2^4 < 26 < 2^5$ 可知只要用 5 *bit* 即可表示。其对应的代码如下：

A 0 0 0 0 0

B 0 0 0 0 1

C 0 0 0 1 0

...

Z 1 1 0 1 0

我们把由“0”、“1”二种状态来表示的代码称为二进制代码。由于任何具有两个不同稳定状态的元件都可用来表示二进制数的每一位，如晶体管的导通和截止，脉冲的有和无，电平的高和低等等，所以二进制代码在电子数字计算机中已得到广泛应用，它非常适合于电子方式处理。如果采用常用的十进制代码，则需要有 0 到 9 十个状态。要用电子方式制作十个稳定状态是非常困难的，即使能够得到 0、1、2、……9 伏的十个输出电压的电子器件，由于电源、温度、元件特性的变化或外界干扰等，结果很可能产生如 7.5 伏的电压，这就必然引起误差。二进制代码尽管在处理过程中有干扰，但是它能改善信噪比，如图 2—1 中，已经变坏的信号(a)，在以虚线作为阈值整形后能得信号(b)。但在处理连续信息的时候，信息的信噪比总是越处理越坏。象记忆连续信息——声音的磁带录音机，如果反复再生和记忆，则噪音就会越来越大。但是利用二进制代码改善信噪比的性质，即使反复

再生，内容也不会变坏。这样，在电子数字计算机中存贮信息就成为可能了。

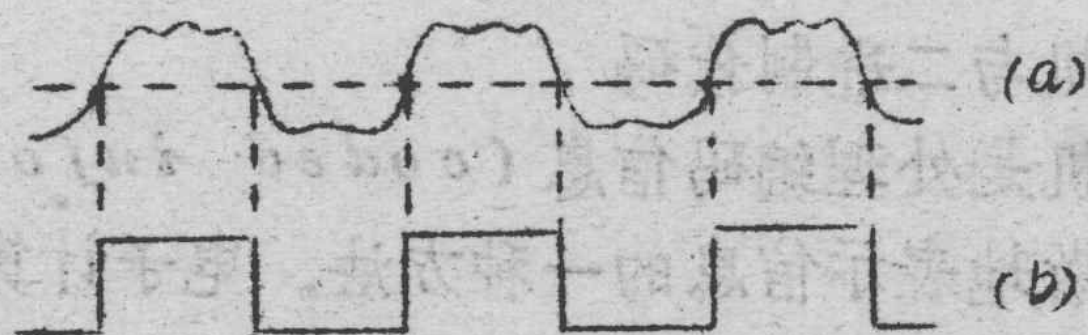


图 2-1

二、数制及其相互转换

(一) 十进制数与二进制数及其转换

我们日常使用的是十进制数，使用的数字是 0~9 十个数码，逢十进一，基数为 10，各数位的权从最低位开始是 10^0 , 10^1 , 10^2 , 10^3 ……例如：

$$\begin{aligned}(2830)_{10} &= (2 \times 10^3) + (8 \times 10^2) \\ &\quad + (3 \times 10^1) + (0 \times 10^0) \\ &= 2000 + 800 + 30 + 0 \\ &= 2830\end{aligned}$$

在电子数字计算机里，使用的基本数制是二进制。使用的数码是 0 和 1，基数为 2，逢 2 进 1。1 加 1 等于 10，再加 1 就变成 11，每一数位的权，从低位开始分别为 2^0 , 2^1 , 2^2 , 2^3 ……

二进制数与十进制数怎样相互转换呢？从二进制数转换成十进制数很简单，只要把二进制数的各个数位顺序乘以数位的权再相加起来即可，例如：

$$\begin{aligned}(110.110)_2 &= 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 + 1 \times 2^{-1} \\ &\quad + 1 \times 2^{-2} + 0 \times 2^{-3} \\ &= 4 + 2 + 0.5 + 0.25 \\ &= (6.75)_{10}\end{aligned}$$

为了便于理解怎样从十进制数转换为二进制数，先举例说明。

例 1：把十进制数 75 以二进制形式表示。

$$\begin{aligned}(75)_{10} &= a_n 2^n + a_{n-1} 2^{n-1} + a_{n-2} 2^{n-2} + \dots \\ &\quad + a_2 2^2 + a_1 2^1 + a_0;\end{aligned}$$