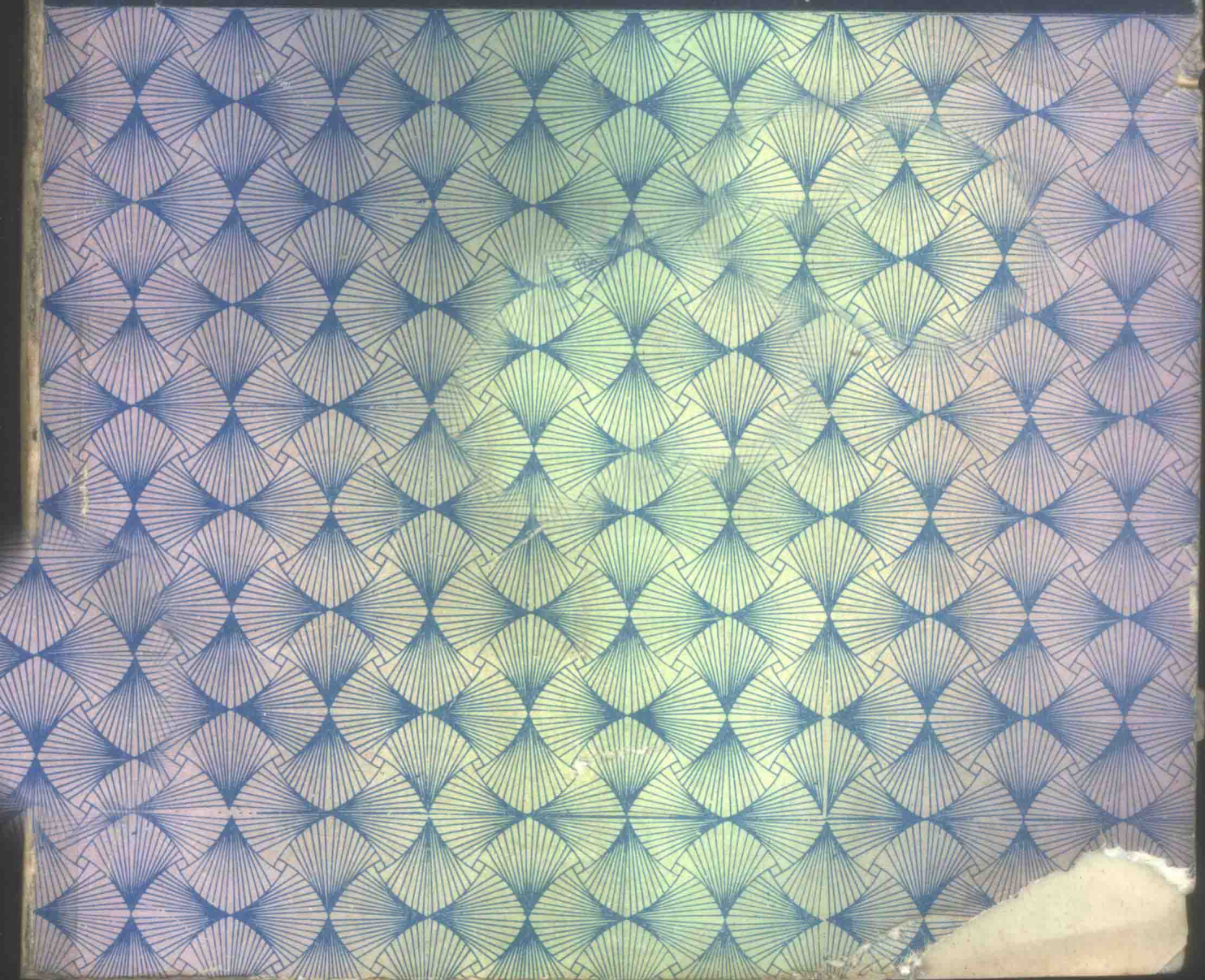


IBM PC FORTRAN77 程序設計

● R·A·勞斯 T·L·巴哥尼茨 著 ● 張書杰 譯 ● 曹德和 校

● 電子工業出版社



IBM PC FORTRAN 77 程序设计

R. A. 劳 斯 著
T. L. 巴哥尼茨

张书杰 译 曹德和 校

电子工业出版社

内 容 提 要

本书从基本的程序设计概念入手介绍了 IBM 个人计算机上的 FORTRAN 77。

本书主要包括：IBM 个人计算机介绍、FORTRAN 77 的数据类型及其处理、输入输出技术、过程控制结构、子程序和子程序库等。书中列举了大量的程序实例，每一章还附有练习题。

本书对读者的适应范围较大，它可以作为从事计算机技术工作的科技人员和大专院校学生的参考书，也可以作为各行各业初学使用 IBM 个人计算机的人们的人门读物。

* * *

IBM PC

FORTRAN 77 程序设计

R. A. 劳 斯 著

T. L. 巴哥尼茨

张书杰 译

曹德和 校

责任编辑 吴明卒

电子工业出版社出版(北京万寿路)

新华书店北京发行所发行 各地新华书店经售

山东电子工业印刷厂印刷

*

开本：787×1092毫米 1/16 印张：11.25 字数：288千字

1985年11月第1版 1988年12月第2次印刷

印数：17000—22000册 定价：2.70元

ISBN 7-5053-0280-9 TP·37

译 者 的 话

IBM 个人计算机(通常称作 IBM PC)是当今世界上流行得非常广泛的一种微型计算机,我国已有许多部门和单位从国外引进了这种计算机,计算机工业部门还开始生产与之完全兼容的微型计算机系统。现在,各行各业越来越多的科技人员、管理干部希望学会使用这种机器,在自己的业务工作中发挥更大的作用。在这种形势下,译者翻译了《IBM PC FORTRAN 77 程序设计》这本书,希望能对学习 IBM 个人计算机的人们有所帮助。

FORTRAN 77 是 FORTRAN 语言的最新版本,在功能上作了一些扩充。IBM PC 配备了 FORTRAN 77,扩大了它的应用范围。译者也希望此书能有助于人们对 FORTRAN 77 的了解和使用

原著是一本教科书,在语气和人称方面不免带有课堂用语的味道,译者在翻译时保留了这一特点。但要说明一点,也许是排版印刷时的疏忽,原著中不少地方有错误,特别是第七章中遗漏了例 7.1。为了不使读者感到莫明其妙,译本中尽量加以校正,而对有关例 7.1 的文字只好删掉,把原例 7.2 的序号变为 7.1,这章中其余几个例子的序号也作了相应的修改。

本书由北京工业大学曹德和同志校对。

在翻译本书的时候,北京工业大学李礼贤副教授给予很多帮助,在此深表谢意。

还要向读者说明的是,由于译者水平有限,本书中定有许多不妥之处,敬请读者批评指正。

1984. 8.

前 言

对于科学工作者和工程师们来说, FORTRAN 仍然是主要的程序设计语言, 它把要计算的问题用自然的方式来表达, 这就使人们容易学习基本的编程技术, 它的先进特性可使复杂的问题迎刃而解。FORTRAN 77 是 FORTRAN 语言的最新版本, 它扩充了数据处理问题中字符加工方面的解题能力。IBM PC 上的 FORTRAN 77 可使这种机器能够用于办公室和图书馆。

此书不论对熟悉与不熟悉程序设计和 FORTRAN 语言的人来说都是适用的。它向不了解计算技术的学生介绍了用 FORTRAN 设计与编制程序所需要的程序设计工具和方法。对于在高级语言方面有经验的人来说, 本书仔细地介绍了结构化 FORTRAN 77。大量的例子和练习有助于对编程工具和语言的学习及开发。

本书与最通用的编程实践相结合, 其基点着重于算法的精确设计。算法的实现通过结构化编程技术来解决, 这一点是十年来若干软件工程的开发所证实了的。书中的例子与这一既定方案是一致的, 用这一方案开发的程序便于校正、调试和将来的修改。交互计算技术也是贯穿此书的重点。

全书使用了在 IBM PC 上实现的 FORTRAN 77 的流行的应用。开始的那些程序结构只需要用极少的 FORTRAN 知识, 这就有助于读者(即使是程序设计的新手)早日着手进行计算。本书在两个方面介绍了 FORTRAN 语言的特性: 首先要使读者能够掌握和使用基本的语言能力, 另一方面是叙述一些先进的概念和实例。

本书还介绍了有关 PC 机磁盘操作系统 (DOS) 的情况, 以便在准备、编译和执行 FORTRAN 程序时使用。本书还讲述了一些很有用的辅助函数, 例如一个好的伪随机数发生器及一些基本而又复杂的运算符程序。

此书既可以用于四年制的学院和大学, 也可以用于两年制的学院。本书对现代编程工具完整的论述对任何用 FORTRAN 来进行计算的入门课程都提供了出色的支持。对于科学或工程方面的学生, 本书可以单独使用, 也可与专业方面的书联合使用, 以促进用 IBM PC 来解决问题。本书也可以用于工业、商业以及职业的程序开发方面的技术培训。本书不期望读者有程序设计的经验, 但认为读者已有初步的代数学知识。

我们感谢华盛顿大学计算机科学教授 Seymour. V. Pollack, 他对我们的信任和无条件的支持使这部著作的出版成为可能。他的清晰的教学方法给我们树立了榜样, 没有他便不会有这本著作。

我们也要感谢 Cobb/Dunlop 出版服务公司, 感谢他们在准备这本书时所给予的不辞辛苦的耐心的帮助。

目 录

第一章 计算机与计算的介绍.....	1
1.1 计算机系统及其部件	1
1.2 IBM 个人计算机系统及其部件	5
1.3 系统程序设计概念	8
1.4 程序设计语言的特性	15
练习	16
第二章 结构化 FORTRAN 77 的文体指南.....	17
2.1 建立一个 FORTRAN 程序	17
2.2 如何输入、编译和运行例 2.1	21
2.3 简单的计算	21
2.4 较复杂的计算和输入输出	22
2.5 由计数控制的重复	24
2.6 交互输入	26
2.7 IF-THEN-ELSE 结构和数组	28
2.8 球员数据的排序	30
2.9 DO-WHILE 结构	32
2.10 文体指南小结	35
练习.....	35
第三章 数据.....	37
3.1 变量数据和常量数据	37
3.2 字符数据	38
3.3 数值数据	39
3.4 其它数据类型	42
3.5 变量的说明	43
3.6 数组	46
3.7 小结	51
练习	52
第四章 计算和数据处理.....	54
4.1 赋值语句	54
4.2 数值表达式	54
4.3 字符数据的处理	59
4.4 小结	62

练习	62
第五章 基本输入输出技术	65
5.1 文件结构和存取方法综述	65
5.2 向 FORTRAN 描述文件	66
5.3 文件的存取	70
5.4 FORMAT 语句	71
5.5 小结	76
练习	76
第六章 过程控制: 选择	79
6.1 IF 语句结构	79
6.2 测试条件的说明和求值	81
6.3 IF 结构的组合	85
6.4 小结	88
练习	88
第七章 过程控制: 循环	92
7.1 由条件控制的循环	92
7.2 由计数控制的循环	93
7.3 小结	103
练习	103
第八章 子程序	108
8.1 子程序的概念	108
8.2 函数子程序	112
8.3 子例子程序	117
8.4 子程序自变量的定义和使用	119
8.5 程序的模块化	128
8.6 小结	132
练习	132
第九章 先进的输入输出技术	136
9.1 直接存取文件的处理	136
9.2 无格式文件	137
9.3 直接存取文件的建立和使用	137
9.4 交互操作实例: 修改一个直接存取文件	140
9.5 小结	142
练习	142
第十章 逻辑变量	144

10.1 逻辑变量的性质	144
10.2 逻辑变量的说明	144
10.3 用逻辑变量进行决断	145
10.4 逻辑变量的输入输出	149
10.5 小结	150
练习	150
 第十一章 标准子程序库	 151
11.1 FORTRAN 77 函数的使用	151
11.2 个人子程序库的构造	153
11.3 子程序库的某些构造块	154
11.4 小结	159
练习	162
 附录 A 个人计算机的字符集	 164
附录 B ASCII 字符码	165
附录 C 复数及其运算	167

第一章 计算机与计算的介绍

正象我们所知道的,计算机是用来帮助解决各种问题的工具。正如所有的工具一样,我们也必须以某种方法管理计算机以便能够以最有效的方式利用它的解题能力。我们通过开发并运用程序来管理计算机。所谓程序就是计算机能够解释并执行的指令的序列。

本书主要的目的是要提出一些方法,运用这些方法可以有效地建立解决我们所感兴趣的问题的程序。然而,我们必须从了解计算机本身开始、它如何与其环境交互作用,指令序列又是怎样执行的。

1.1 计算机系统及其部件

一个计算机系统在概念上由两种不同类型的部件构成:硬件和软件。系统的物理设备通常称为硬件。它在很大程度上决定系统的能力。然而,为了进行有益的工作,硬件必须由人们给定的指令来操纵。这些指令结合在一起就是程序。我们选定与某一系统一起使用的一组程序即是此系统的软件。硬件与软件相互配合才能使计算机为我们工作。

1.1.1 硬件部件

系统的各硬件部件以及它们互相结合的方法决定了系统的能力。尽管每类硬件部件是各种各样的,但我们的叙述只考虑一般类型。

当我们学习各种计算机系统时,我们会看到每个系统都有四类硬件部件:CPU(中央处理单元,计算机的“思维”部分),主存(CPU的“记忆”部分),外存(信息存储在主存之外)以及输入输出设备(我们用来与CPU通信)。

1.1.1.1 中央处理单元

计算机“计算”(联结数据元素、进行算术运算、进行决策)的能力由CPU的能力决定。CPU是执行指令的部件。每个CPU都有一组它会执行的最基本的功能,例如,两数据项相加、信息从一处传送到另一处、两值的比较等等。支配上述各种操作的指令组称为CPU的机器语言,这种语言与构成计算机的实际电路有关。不同的计算机其机器语言是不同的。

所有我们写的程序都是由CPU独立执行的若干指令。计算机系统执行的全部操作都受CPU支配。就这点而论,CPU实际上是系统内唯一的主动部件,即,它可以独自启动其它硬件部件。我们必须了解,CPU只能对数据进行操作,它不能存储数据,甚至不能与外部世界进行数据通信,要执行这些功能还须其它的系统硬件部件。

1.1.1.2 主存

尽管在执行程序时必须能够存储和检索数据(还有指令),但 CPU 本身并没有保存信息的内部能力(用硬件的说法)。计算机系统的主存提供了这种功能。主存是用于计算机系统的一种“刻痕片”(Scratch Pad),当计算机在执行某个功能、也即执行某个程序时,把指令和数据存储在这里,以供 CPU 使用。当这个功能已完成而要开始另一个功能,即要执行另一个程序时,原存入主存的信息都会失掉,新的功能所用的信息将存入、主存。主存是动态的,并不具有永久的数据存储能力。

主存由大量的存储单元组成,每一单元可容纳少量的信息,其容纳的数据量依计算机不同而不同。但是,几乎所有的计算机都是以一个单元能存的字符或字节数来定义存储容量的。尽管我们看到一些计算机的每个存储单元只是一字节或两字节,但是许多计算机的存储单元是四个字节。每个存储器单元都有一个与之相联系的地址,当 CPU 需要把信息存入主存或从主存检索信息时,都将访问相关的地址。地址是用数字表示的。例如,我们可以说:“CPU 将从地址 1024 得到信息。”

当 CPU 从主存的一个单元检索数据时,就象我们从自己写的笔记中读一则笔记一样来读取信息。当 CPU 读主存一个单元的信息之后,此信息仍然保留在那个单元中(所谓读也就是将某一信息的拷贝从主存送到 CPU)。只有 CPU 在往一个单元存(即“写”)信息时,这个单元原有的信息才被更替或破坏。

主存所能存储的信息量由它所包含的单元数量及一个单元可存的字节数来决定。一般的小型计算机的存储量从 16,384 字节(通常称为 16K,每 1024 为 1K)到 256K 字节。如果每个单元需要 4 个字节,那么 16K 存储量的主存便有 4096 个单元(又称为字)。

主存储器是一种被动设备,它自己不能发生任何动作,它是由 CPU 控制的。它只是一个保存 CPU 所需信息的仓库。

1.1.1.3 输入输出设备

有了 CPU 和主存,计算机系统便有了操纵数据的手段和暂存数据的地方。现在我们来看看与系统通信的方法。有了这种通信,我们就能把信息发送给 CPU 或从 CPU 接收信息。尽管在计算机领域内有无数类型的输入输出设备可用,但本书只集中讨论小计算机系统通常使用的那些设备。

面向人的设备

键盘设备是最通用的人机通信设备。信息通过类似于打字机的键盘打进去,键盘把键的敲击变换为电信号,然后送到 CPU。大多数键盘设备都带有显示设备作为一个有机的整体。通常打印机或显示终端(CRT)与键盘相连接以用来反映打入到 CPU 的字符。我们可以见到一些相当完备的系统,它们既有打印机又有 CRT。在这些系统中,打印机通常是不用的,当用户要求输出已打入的信息的硬拷贝时打印机才启动。当打印机起用时,信息将在 CRT 和打印机上同时反映出来。

对于人机的直接通信,大多数系统都使用同样的显示设备,它是输入机构的一部分。图 1.1 示意性地画出了从键盘到 CPU 和显示器,以及从 CPU 到显示器的信息流。很明显, CPU 并不试图通过键盘来同我们通信。

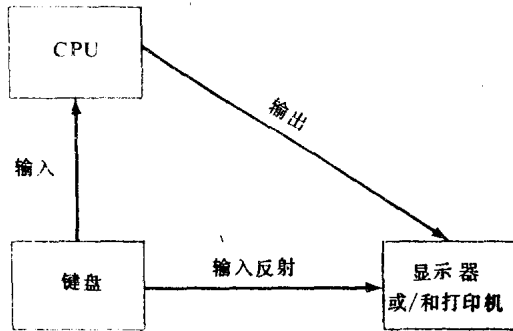


图1.1 输入输出流程图

面向机器的设备

假定我们要在一台计算机上运行两个程序，第一个程序将产生作为第二个程序的输入信息。我们设想第一个程序产生在硬拷贝打印机上打印的数据，然后再把它们输入到计算机以备第二个程序使用。对于这个过程，其操作如下：

1. 把信息写到硬拷贝输出设备上时，CPU 内部信息格式被转换为人可读的格式。
2. 写有信息的纸成为储存两程序之间信息的工具。
3. 把纸上的信息打入键盘并送到 CPU 时，又把入可读格式的信息转换为 CPU 可理解的格式。

在此过程中信息被转换了两次，过程结束时信息又给转换为原来的格式——CPU 格式。要解决这个问题，就把 I/O 设备做成具有能与 CPU 相同的语言进行通信的能力。在下节我们会看到这样一些设备，它们本身并不是 I/O 设备，但是它们是这样的一种工具，可以永久地储存可被 CPU 直接存取的信息。就一般的意义而言，任何能与 CPU 进行通信的设备均可被认为是 I/O 设备，尽管通信的对象并不是人。下节所提供的设备使我们对以后要用的信息不必存储为人可读的形式。

1.1.1.4 外存储器

尽管主存是有效的存储和检索数据的设备，但它缺乏永久性，而对于保存某些类型的信息来说又需要这种永久性。例如，我们可能希望存储女朋友或男朋友的姓名和地址以备今后使用。我们需要以一种方式存储这些信息，使得在应用这些信息的程序完成以后它们仍然保留着。外存储设备可以为我们完成这项工作。

所有的外存储器都有两种主要部件：执行存储和检索的实际机构以及存储信息的介质。小计算机一般使用两种外存设备：盒式磁带存储器和软盘存储器。它们都是根据磁性原理存储信息，只是所用的介质形式不同。盒式磁带把数据存放在标准的录音带上，储存的方法类似于对音乐进行录音。软盘设备使用一个圆的塑料盘，它有与盒式磁带相同的磁性材料。用盒式磁带只能从头顺序查找，直到找到我们所需要的信息（顺序操作）。读盒式磁带机与读股票行情记录机有些相似，在找到所需要的报价之前一直读下去。用软盘设备可以直接存取盘面上某一点的信息（称为直接存取）。这有些类似于把唱机的长臂放下，唱头落在想要的歌曲位置上。正象主存一样，当从外存设备读出信息后，所读的信息拷贝到计算机，但它仍然保留在外存设备上。只有在写信息时才会破坏原有的信

息。也与主存相类似,外存储器设备上也有一个个的存储单元。第五章将讨论寻址的机理。

几乎我们所写的程序都要存储在外存储设备上。通常要把程序从外存读到(称为装入)主存,CPU 再在主存执行这个程序。

1.1.2 程序设计部件

为了发挥硬件的原有能力,必须要通过指令、程序或系统软件来控制硬件的动作。要运行一个系统需要有两种软件:操作系统,它控制硬件的功能;应用软件,它是直接处理我们所面临的数据加工问题的程序。

1.1.2.1 操作系统

我们已描述过的各种硬件的实际物理操作都要涉及与硬件打交道的许多指令。举例来说,要从软盘驱动器上读一个变量值,就涉及到在磁盘上找出数据的确切的物理单元、传送适量的数据、在传送时检查可能发生的任何错误等等方面的指令。每一个硬件部件都有它自身的这类特性,希望任何一个人人都来掌握硬件操作的所有方面是不合理的。例如,我们希望只要说“读变量 AGE 的值”然后系统就会自动地检索信息。操作系统是一个程序,编写它的目的是用以完成所有这些麻烦而又复杂的操作。顾名思义,操作系统控制所有硬部件的实际物理操作。

为了加强对概念的理解,我们以日常活动为例。假如我们有自己的私人飞机,且需要飞到远处一城市。为了利用飞机(硬件)飞到远处的城市,我们应当知道如何使用系统的各种部件(导航仪器,飞行控制设备等),还应当知道飞机内部操作的详细情况。然而应当承认,这对我们来说是很复杂的任务(也许飞机是大型喷气机)。需要的是让飞机知道我们要它做什么(程序),并且需要让一个复杂的系统来控制整个飞机的所有部件,把我们带到目的地。解决的办法是雇一个飞行员(操作系统),他拿到我们的“程序”(即飞到某地),然后操纵硬件(飞机)以解决我们的问题。正如飞行员使我们方便地利用飞机一样,操作系统可使我们容易地应用计算机系统的硬件。

当我们写程序时,我们将要求操作系统为我们执行许多功能,其中许多要涉及到与操作系统进行直接对话。例如,要操作系统告诉我们软盘上所有的文件名。有时我们只要求操作系统执行一些简单的功能,例如,把一个信息写到输出设备上,这时操作系统会执行这些功能而不需我们特别地请求它来进行直接对话。

1.1.2.2 高级语言

在机器这一级,我们给予计算机的一些指令是非常简单而又有限的。举例来说,把两数相加,我们可以看到需要下面一系列机器指令:

1. 把主存 1024 单元中的数值送到 CPU 的“加法区”。
2. 把主存 1025 单元中的数值送到 CPU 的“保持区”。
3. 执行逻辑操作(实际上是电路操作),把加法区与保持区相加,把结果放到 CPU 的“结果区”。
4. 把结果区的数值送到主存的 1026 单元。

这个简单的例子在本质上说明了把主存中两数相加再把结果放入主存的过程。读者立即可以看出这种做法即使对相对而言很简单的处理来说,使用起来也是很繁琐的。其复杂性还在于我们必须要用机器语言与计算机通信,所谓机器语言是机器所认识的语言,它由一串 0 和 1 构成。在这样的条件下编程不可能快速。

使用一种人工语言,或称为高级语言,就可以做到快速编程。这种语言使我们能以更接近于人类的语言来向计算机提供指令,它又可以翻译为相应的机器指令。对于上面所举的加法例子,可以写成类似于下面的形式:

ADD A TO B GIVING C

或者更简单地写为:

$C = A + B$

高级语言允许我们编写程序,也即编写指令序列,它提供了许多精确的有含义的“句子”,这些句子可以被翻译为机器所能理解的指令。

大多数高级语言都接近于英语格式,而且都有一套规定的构成句子的规则(语法),编程时遵循语言的语法,我们就能建立起易于转换为机器所能识别的格式的程序。

一组高级语言的句子称为一个源程序。所编写的源程序必须翻译为机器语言。把源程序翻译为机器语言是由编译程序来完成的。编译程序懂得源语言的语法,它把源程序翻译为目标程序。目标程序是机器语言码,它是可执行的。由于不同的高级语言其语法不同,所用的编译程序也就不同。

1.2 IBM 个人计算机系统及其部件

IBM 个人计算机的构成与前面几节所讲述的一般计算机系统相类似。这一节我们着重讨论标准的输入输出设备、外部和内部存储器以及处理单元本身。

1.2.1 硬件

我们要考察的硬部件如图 1.2 所示。中央处理单元、内部存储器和外部存储器(软盘驱动器)都组装在一个长方形的盒子里,称为系统单元。输入设备是 IBM PC 键盘,输出设备是显示屏幕和打印机(如果需要的话)。

1.2.1.1 中央处理单元

中央处理单元装在系统单元内,它与其它一些电路一起安装在一块系统板上。系统板是个人机的主要电路部件。处理器是 Intel 8088 微处理器,它包含在一块集成电路芯片上。此处理器平均每秒可执行单个操作,如加法或减法达 65 万次。并可寻址或存取内存 100 万个字符。除了 Intel 8088 以外,有些 IBM PC 还装有 Intel 8087 处理器,这是一种附加处理单元,专门设计来执行一些算术操作,它可提高算术运算速度 25 倍。

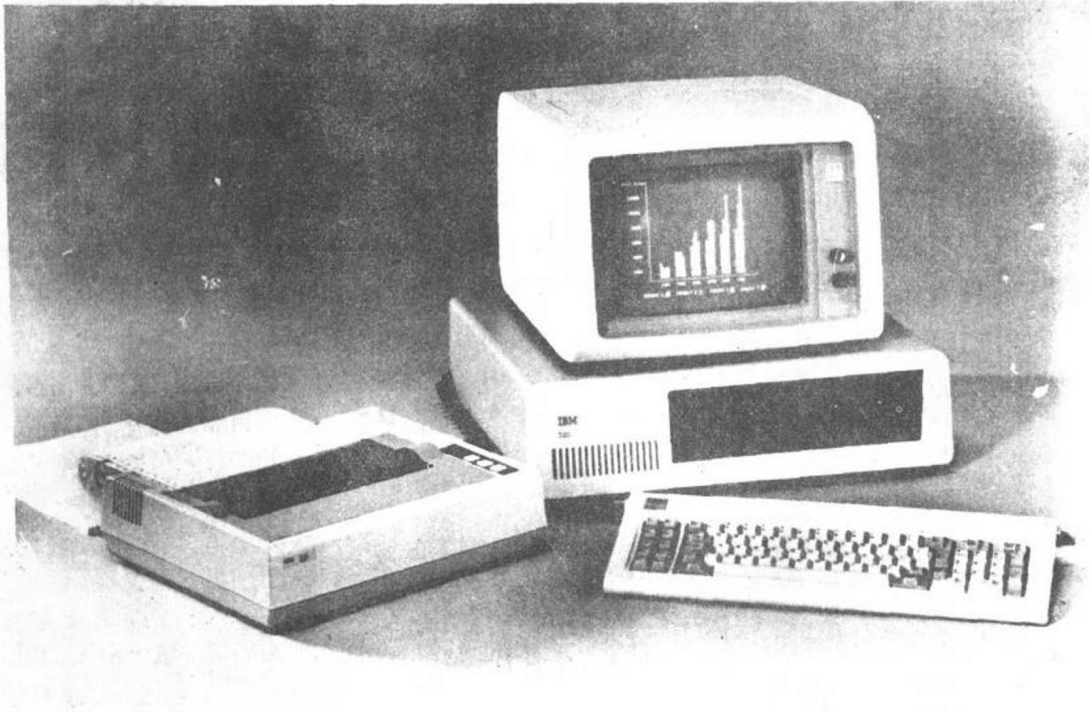


图 1.2 IBM 个人计算机的硬部件

1.2.1.2. 主存

在系统板上与处理器一起安装有主存的两个部分：ROM（只读存储器）和 RAM（随机存取存储器）。

只读存储器有 48K（1K 表示 1024 个存储字符），它只被 CPU 读，不能被 CPU 写。ROM 中包含的信息由用 CP/ 来执行专门的系统功能。只读存储器中的内容是在制造时装入的而且总是保持不变（甚至在电源关闭时也是如此）。

随机存取存储器是动态存储器，可由中央处理单元进行读和写。系统板包含的 16K RAM 作为标准选件，通过安装额外的存储器电路可以增加到 32K，48K 和 64K。另外，以 64K 为单元，运用存储器的成套扩件可把 RAM 的容量扩充至 256K。当电源关闭时，RAM 中的信息便会丢失。

1.2.1.3 输入输出设备

自然,对于 IBM PC 的通常使用来说,大多数输入输出设备是面向人的。也存在着一些面向机器的设备,大多数用户应有节制地使用。

面向人的设备

在 IBM PC 与用户之间大量的交互操作是使用键盘设备(输入)和 CRT 显示屏幕(输出)。键盘有 83 个键,包括标准的 QWERTY 部分,10 个特殊功能键,15 个光标控制/数字键。有一些特殊符号,它们用于返回、回跳一格、前跳一格以及换档等功能键的标记。读者应参考“IBM PC 操作指南”来熟悉这些键的用法。键盘通过一条 6 英尺长的螺旋型电缆与系统单元(从而也与中央处理单元)连接。它用一个 5 线连接器连在系统单元的背面。

两种类型的 CRT 显示设备用于 IBM PC 机。IBM 单色显示是一个高分辨率的绿色显示器,用于显示字符和低水平的图形。这个设备可以显示 25 行,每行 80 个字符,每个字符用 7×9 的点阵构成。

尽管单色显示器是高质量的视频显示,某些用户还是挑选使用一种高分辨率的彩色视频设备而代之。这种类型的设备在显示正文信息时并未增加功能,但是它允许 IBM PC 用它作为高质量的视频图形输出设备。

除了视频显示外,打印机可以附在 IBM PC 上,以提供硬拷贝输出。标准的打印机是每秒 80 个字符的设备,它用点阵构成字符,其方法类似于单色显示器上形成字符的方法。IBM 80 CPS 打印采用 9×9 点阵、可打印 96 个字符和 64 个图形字符。字符的大小分别是每英寸 5、8.25、10 和 16.5 个字符不等。每行的字符数分别是 40、66、80 和 132 个。

有两种使用打印机的方法。第一种方法,用户按键盘上的一个专门键就可以产生显示屏上的内容的拷贝。第二种方法是把打印机作为在计算机中运行的程序所控制的直接输出设备。

面向机器的设备

大型计算机系统广泛地支持各种面向机器的输入输出设备。与一些大型计算机系统不同,IBM PC 只是支持有限的一些这种设备。如果我们回避软盘驱动器和盒式磁带是面向机器的输入输出设备这一事实,那么能够与 IBM PC 直接通信的唯一设备就是异步通信适配器。

异步通信适配器可允许其它计算机设备,例如另一个 IBM PC、实验室监控设备等与处理单元直接通信。异步通信适配器本身不是输入输出设备,但它允许用户把一些面向机器的设备直接连接到计算机上。

适配器可使 IBM PC 以每秒 5~960 个字符(ASCII 码)的速率在任何地方发送或接收数据字符(实际的速度与适配器所连接的设备有关)。

1.2.1.4 外存储器

IBM 提供的标准外存储器设备是 5 1/4 英寸小软盘驱动器。系统单元可以安装两台这样的软盘驱动器(尽管没有软盘驱动器计算机也能使用),驱动器自身读写 5 1/4 英寸软

盘上的信息。

驱动器分两种型号。一种把信息记录在软盘片的一面上,另一种把信息记录在盘片的两面。头一种每片容量 160KB,第二种每片容量 320KB。

软盘片可从驱动器上取下更换,所以存在软盘上的机器可读的信息量实际上只是受用户所购买、储备和掌握的软盘片数的限制。

除了软盘驱动器外,PC 机还使用专门连接的盒式磁带录音机作为外存储设备。这种设备的价钱比软盘驱动器要便宜,但数据的传输速率较低,大约是每秒 150 个字符。当使用软盘时,盒式磁带可从录音机上取下,盒式磁带的容量取决于带的长度。

1.2.2 程序设计部件

IBM PC 支持丰富的编程工具,包括操作系统和大量的高级程序设计语言。

1.2.2.1 操作系统

目前,有三种操作系统用来控制 IBM PC:IBM DOS、CP/M-86 和 UCSD P-系统(圣地亚哥加州大学 P-系统)。

本书中的程序设计是在 DOS 操作系统下进行的。DOS 操作系统的目的是把用户从控制和存取磁盘存储器的任务中解放出来,它提供了在用户程序控制下读写信息的简易方法。此外,DOS 操作系统还提供了用户管理软盘上信息的设施(命名储存在盘上的文件,查询和修改文件等等)。

CP/M-86 是与 DOS 相类似的操作系统,也提供了对软盘存储器的控制,还可以管理整个 IBM PC 系统设备的使用。DOS 的许多功能可在 CP/M 中应用,CP/M 还具有修改和使用程序的能力。

UCSD P-系统也提供了与 DOS 和 CP/M-86 相似的磁盘管理能力。而且,在它控制下编写并且运行了的程序可以大量地移植到其它在 P-系统下运行的机器上,它所具有的一些功能在别的操作系统上是不易实现的。

1.2.2.2 高级语言

依据使用的操作系统的不同,IBM PC 机上使用的高级语言也有所不同。使用 DOS 操作系统,用户可以使用两种版本的 BASIC 语言(Disk BASIC 和 Advanced BASIC)编程。BASIC 是初学者所通用的一种非结构化语言。另外还可以用 FORTRAN 的 DOS 版本和一种较新的语言 PASCAL。对于 P-系统用户,FORTRAN 和 PASCAL 也是可用的,尽管其版本与 DOS 中的版本略有不同。汇编语言在较大程度上是面向技术研究而用的,而且使用起来又较困难,但是三种操作系统都提供了汇编语言。

1.3 系统程序设计概念

当写程序时,人们似乎急于立即坐下来开始写 FORTRAN 代码(或 PASCAL 或 COBOL...)。当有的人听到有关某个问题时通常发生这样的情况:匆忙的打字、开始输入程序,而常常很少考虑为长久应用而精心设计程序、定义数据和代码。实践表明,按一定的

原则来编写程序可以产生非常好的代码。不掌握这种基本的原则，会挫伤进一步使用计算机作为解题工具的积极性。

1.3.1 引言

在最近廿年内，程序设计的方法有了本质的变化，程序设计已从程序员个人的手稿发展为一种开发可靠的、容易理解的程序的原则方案。廿年以前程序员们遵循他们各自的爱好来选择开发程序的方法。这种方法产生的程序往往对任何人来说都是难以理解的，而只是原来的程序员懂得。甚至原来的程序员在经过一段时间之后也常发现弄明白他(她)原来的编程意图是困难的。

最近十年产生了一种程序设计的习惯作法，看起来这种方法在本质上要比别的方法为好。这种方法并不是一种学科，但是它为许许多多程序员广泛采用，令人信服地表明它更具有生产能力，以此种方法建立的程序可以被其它一些程序员所理解，并且，能经得起时间的考验。这种方法被称为“结构化程序设计”，它实际上是编程工具和技巧的集合。清楚地说明问题，认真地设计解题的方法或算法，恰当地运用程序设计语言以把算法转换为计算机能够解释的过程，这些就是这种程序设计方法的基础。

这种方法首先强调的是设计，而实现(写出程序代码)是第二位的。其核心是将程序设计过程分为两部分。第一部分集中于问题及其解法或算法，与任何特定的计算机或计算机语言无关。第二部分集中于认真地使用程序设计语言，把算法表达给特定的计算机。

1.3.2 算法

若用计算机解题，则要求问题的解能够用计算机可以执行的步骤序列来确定。确定程序的过程需要对问题加以定义、找出解题规则和实现规则的方法。产生的算法还须有一些附加特性，以有助于计算机的实现。

1.3.2.1 算法的开发

开发解决特定问题的算法需要如下三个步骤：

1. 对问题的明确说明，
2. 找出解题的规则，
3. 确定实现此规则的步骤序列。

问题的本身必须要以无二义性的方式加以说明。如果一个问题依赖于直觉或个人的口味，或者它所涉及的数据、项目的定义、决策过程都不能确定的话，那么这个问题便不能被计算机程序解出。不管这个问题是产品价格的决定、提高工资还是化学方程式，问题的提法都必须是无二义性的。例如，“该给 Joe 增加多少工资？”这样的问题，如果有关的变量和决定增加工资的方法能够确定，那么这个问题便可由计算机程序来解决。

开发程序的第二步是寻求解题的规则，这种规则可能是物理学定律、工程上所采用的实际做法、合理的要求、簿记中的惯例等等。不管原先是什么样的问题，解题的规则必须要规定如何考虑、如何处理或如何分析在程序语句中所说明的数据，以产生可以认为是原问题答案的结果。对于 Joe 增加工资的问题，如果我们知道做决定用的两个因素：Joe 的工龄和他头发的颜色，并且知道工资的增加数等于工龄乘以 J-HAIR（头发因素）。当头