M-PL

# 8087应用与程序设计

〔美〕Richard Startz著

张载鸿　赵意丽　译

# 有关软盘的说明

随本书附带一片可用的软盘选件。盘上包括列在程序清单上的所有程序。每一个程序都有三种形式：汇编语言源程序（如：PROG .ASM）；已经汇编好的目标模块（如：PROG.OBJ）；以及可用BLOAD命令装入BASIC的文件（如PROG.SAV）。为简单起见，在某些情况下，一组程序组合成单一的模块。例如：最重要的矩阵操作程序都组合在文件MATRIX.ASM，MATRIX.OBJ和MATRIX.SAV中。所有的程序都存储在一个标准的单面5.25英寸的软盘上。该软盘是在IBM-PC机上用PC-DOS 1.1版本格式化的。在本书的末尾有软盘文件的副本。

# 引　言

首先，作者说明为何对8087如此感兴趣。多年来，本人一直使用大型机从事专门的研究工作。当购买个人机后，发现在个人机上做许多事情远比大型机方便，但很快又注意到个人机在运行大型数值运算程序时速度不快。

配置8087后，使个人机能解决许多大型的问题。虽然，有些问题仍需大型机求解（而且永远如此），但是个人机的计算范围扩大了十倍。

8087不仅速度快而且使用方便。无论读者主要是“程序用户”或者主要是“程序编者”，都会感到8087是一个相当出色的器件。作者希望，本书作为一本入门教材，读者能从8087应用和程序设计中找到乐趣。

## 本书的服务对象

- 希望了解8087功能的读者（特别是第一部分，第一——四章）。
- 希望学习如何设计8087程序的读者（特别是第二部分，第五——八章及第三部分的第十二章）。
- 希望在个人机上准备运行数值运算程序的读者（特别是第三部分，第九——十五章）。

第一部分非技术性地描述8087的性能。若读者在考虑购置一个8087，并且想了解8087兼容的硬件和软件，这一部分内容正适合。

第二部分和第三部分内容技术性较强。尽管从“初等水平开始”，但以往的编程经验对读者是有帮助的。读者无须是编程行家，但本书也不是计算机的导论。

第三部分（第五——八章）较深入地描述8087的指令，同时介绍某些8088汇编语言基本编程手法。对汇编语言与BASIC程序的连接给予特别的注意，其中详细介绍汇编语言程序与解释的BASIC或编译的BASIC程序相互连接的交互方式。

第三部分集中于应用。为此，开发了许多有用的8087汇编语言程序。读者以此作为例子，能学到更多的8087程序设计方法，或者直接按程序清单的格式调用程序。（第三部分的第十二章对8087的高级指令作了说明。）

## 怎样阅读本书

本书经过精心编写，使读者能根据自己的需求从一个章节跳到另一个章节进行阅读。请不必受从头到尾的编排限制。

大多数读者或许会认为第一部分有启示性，且易读。若读者想要编写8087汇编语言程序，应将精力集中于第二部分。（但若读者是一位有经验的8087程序员，则可以跳过第二部分，直接阅读第三部分的应用。）若读者的主要兴趣在于应用，而不关心具体的编程细节，请阅看第三部分。在需要验证某些细节时再翻回到第二部分。

最后，如果读者只想尽快得到答案，就按程序清单的格式将程序拿来使用即可。而

无需了解一个程序为何以某种方式编写以及其内部如何操作。如果一个程序确实有用，读者想修改它或自己编写类似的程序，那再翻阅清单后面程序的注释部分。

### 程序清单

有几章是以一个介绍性的段落开始，然后是如下的符号：

| 程序清单 |
|---|

在这一符号下面，读者能看到在本章出现的程序列表，并附有程序用途、输入、输出要求的简要概述。当读者急于要寻找一个程序时，请使用清单。本书花费相当多的时间讨论某些问题为什么必须要用某种方式解决。若读者只想运行程序，而不想自己建立程序，可不必阅读“程序注解”。但有关需要把哪些信息传递给程序的说明部分要细细阅读。

### 数值运算的策略

本书除了介绍有关8087的大量资料及数值计算的编程外，还对重要的计算任务提出某些策略。根据这些策略归结两条编程准则：

- 用10%的程序代码承担90%的程序执行时间。
- 若不考虑所使用的编程语言的能力，生成一个可用程序的代价正比于代码长度的平方。

有时一些认真的程序员费了很大的精力编写出所谓“高效的程序”。其实更好的办法是用10%的代码去承担90%的计算量。为此，重新编写程序，使其10%具有最高的效率，另90%具有最大的清晰度。

要使程序高效，往往导致用汇编语言编写程序。但因为汇编代码的长度10倍于等价的BASIC程序。因而，程序的调试工作可能困难100倍。一个完整的程序全部用汇编语言编写（几乎）是没有意义的。但用汇编语言编写一些关键的部分却是有意义。通过这一途径，只需付出很小的编程代价，几乎能得到汇编语言全部的速度优势。

实际上，把使用同一基本子程序的许多数值编程问题组织在一起，其效果会更好。用8087汇编语言编写一个数组加法程序要比BASIC的FOR/NEXT循环复杂，但它只是一次性编写和调试。以后一次又一次地反复调用这个子程序，要比每进行一次数组加法就写一个FOR/NEXT循环方便得多。计算机科学家称这种子程序的设计为“模块编程”。为了解8087模块子程序的方便和功能，请参阅第十四章的统计程序包。

实际上，读者会做得更好。许多适用数值计算需要的8087程序都出现在本书中。（在附带的磁盘上）。希望读者学到了8087的所有性能后能自己编写专用的程序。当然也欢迎读者利用书中的子程序，用在自己配置8087的个人计算机上。

### 硬件和软件的要求

本书的程序运行于由Intel 8087数值处理器和8088，8086系列的微处理器支持的计算机上。除了8088和8086以外，该系列还包括188，186和286微处理器以及与之关联的8087。这些程序都在IBM-PC机上开发并验证过。所有的计时都约定微处理器是一个5 MHZ的8088。计时只是近似值。（例如，在IBM-PC机上运行，比给定的计时大约慢5%，而在8086支持的机器上则要快些。）BASIC程序的计时是以不带8087的解释的BASIC程序为

准，除非有另外的限制。

8087汇编语言程序作为一个子程序，可被IBM-PC机上使用的解释的Microsoft-BASIC或编译的Mirosoft-BASIC所调用。程序约定数据以8087兼容格式存储。（参见第三章，详细讨论8087的兼容软件。）程序与BASIC的pre-8087版本能一起运行，但要增加Microsoft-to-Intel格式变换程序（在附录中）。

为了汇编本书的程序，需要一汇编程序用以识别所有的Intel助记符指令。（请注意，IBM-PC的MACRO汇编程序的1.0版本虽然能生成8087指令但不能识别8087助记符指令。如果把8087助记符重编成8088交权指令，则仍然可使用这一汇编程序。在附带的软盘上，已有程序能重编8087助记符，因此，读者照常使用IBM Macro汇编程序）由于BASIC语言是个人计算机的支柱，因此，本书编写的所有程序都由BASIC调用，而不是FORTRAN或其它大型机的更通用的语言。如果读者想使程序与一个内部约定不同于BASIC的其它语言相结合，则需要重编一些指令。所有的程序都能在IBM-PC PC-DOS 1.1版本支持的Microsoft-BASIC下运行。如果读者使用其它的计算机或不同的软件，那么某些细节可能有细微的差别。

## 回避和限制责任

**合法声明：**

本书的作者和出版商对书中及所附软件中的程序和资料，不作任何明确的或暗示的保证和担保。作者和出版商声明，不承担任何直接或间接赔偿损失的责任，也不担保其销路。本书只热衷于作为一本“基础教程”，并不能适用于任何场合。

敬告读者：

作者尽了最大的努力以确保书中所提供的内容是准确的，所有的程序是能运行的。然而，在数百页的文章和数千行的程序中，有可能潜伏着差错。本书及软件的用途在于教学。当读者使用书中的程序时，要确保程序经过测试。如果读者的大多数程序用BASIC编程，则应把注意力放在本书的“错误处理”一节。因为汇编语言程序的安全性自然不及高级语言编写的程序。

不论作者多么谨慎，错误终究难免，如果读者发现的话请写信告诉作者，（C/O Robert J.Brady Company，Boxxe，Maryland，20715）使作者能更正再版。

## 商标

本书用到下列商标：

- IBM，IBM Personal Computer，IBM-PC和PC-DOS是国际商业机器公司的商标。
- 8086，8087，8088，186，188，286 Numeric Data Processor和iAPX是Intel公司的商标。8087和8088指令助记符的版权归Intel公司所有。
- Microsoft和MS-DOS是Microsoft公司的商标。
- Apple Ⅱ+是Apple公司的商标。
- DEC-2060和VAX是DEC公司商标。
- IMSL是IMSL公司商标。

# 第一章　变分钟为秒

配置8087的个人计算机具有三个明显的特征：使用方便、计算精确、运算速度快。本书提供需要应用8087解决实际计算问题的各种方法。这一章概括地叙述8087及其应用。

我们试图用科学的分析方法剖析8087，并将这一精神贯通全书。只要有可能，在论述计算机工作所需的技术细节的同时，结合基本原则分析编程的“原因”。为了更具体，每一基本原则都由一个实际应用例子加以说明。8087功能强，且使用方便。我们希望本书读者能开阔眼界，并从中得到乐趣。

## 怎样方便才算是方便？

设计8087的着重点是对大量的初始计算能力提供方便的使用。8087的用户第一步尤为简单。只要将8087加到个人计算机上，与通常一样运行程序即可（当然需要有BASIC版本及其它适于8087的软件）。无需费力用户就会发现，程序运行速度能提高20%，甚至10倍之多。

如果想最大限度地发挥8087硬件的性能，则需要专门为8087设计软件。在第三章，会进一步地告诉用户在购买8087软件时怎样去选用。这里，只是简单地说明一下。

有关8087兼容软件最重要的知识实际上是硬件方面的说明。8087扩展了机内原处理器的运算能力，而不干扰处理器的正常工作，因此，任何用“pre-8087”设计的软件，在加上8087后都能继续正常运行。

这种100%的向上兼容性是一个很大的优点。然而也有它不利的一面。使用“pre-8087”软件的程序在系统加上8087后，运行速度不会加快。例如：在一两分钟之内能把一个8087加到原有的IBM个人计算机上（为编写本书的程序，作者曾这样做过），所有用BASIC解释或编译好的程序都能正确地运行，但运行速度不会加快，所以，每当说起加上8087会使运算速度加快，实际上隐含着这样一层意思：它是靠8087兼容软件获得的。

（编写小型程序，可使用BASIC语言的pre-8087版本和其它的软件。在第三章将讨论这一问题。本书中所有的程序都是用per-8087软件写成的）

在购置8087软件时，应意识到有一个潜在的陷阱。虽然未必可能，但有时可能会遇到麻烦，把pre-8087软件和那些为8087特性而设计的软件混淆在一起。更详细的论述请参阅第三章。

假设用户有几种BASIC版本或其它支持8087的编程语言，那么在运行所有程序之后，与pre-8087的运行比较，就能发现那些进行大量数值运算的程序速度会明显提高。如果确实需要进行大量而繁重的数值计算，那么，最终要使用专门用高速8087子程序编写的程序库。

本书的第三部分（第九—十五章）包括用于数值计算的最重要的子程序。用户要做的工作是阅读如何使用这些子程序的说明，并把它们输入到计算机，与自己的BASIC程序

组合在一起。（在随书附上的软盘中，子程序都已排列并汇编好）。与pre-8087 BASIC相比，使用这些子程序将使运行速度提高10-200倍，（在少数情况下可高达500倍）。

本书的第二部分（第五—八章）是使用户达到使用8087的最高阶段：为自己编写子程序做准备。浏览本书的例子可看到，由于8087设计精良，用汇编语言编写的程序相对地要简单些。在读完书中的例子和说明后，用户能毫无困难地编写自己专用的程序。

## 怎样精确才算是精确？

如果并不关心能否得到正确的答案，那么简单易写、运算快速的程序是不需要很多的技巧。8087的最大特征是它的引人注目的精确。下列三个特点增强了它的精确性：

- 内部计算位域比BASIC双精度数超出11位，相当于附加三位十进制数位。
- 内部计算数值范围相当大。8087表示数的范围大到$10^{1932}$，小到$10^{-1932}$。因此在运算过程中很少发生上溢或下溢。事实上，在精度和取值范围方面，8087要超过大多数传统的大型机。
- 将8087设计成能处理大范围的错误条件，并使其自动完善地返回到正常状态。这样，简单的“纸和笔”算法不用说更适合于工作。当发生错误时，8087会遵循一个已定的行为准则进入异常处理，替代错误信息的产生。

## 怎样快速才算是快速？

配置8087的个人计算机操作运行有多快？这可以与标准的大型主机以及没有配备8087的微型计算机作些比较。

或许令人信服的是：把8087与那些价值数十万美元的大型机作比较是有意义的。显然，8087的速度比数十万美元的大型机要慢几倍，但它的价钱却不止便宜几倍。

精确的比较总是困难的。但是给出几个数字可使读者对8087的速度有一个印象。中速的大型机约需1-5微秒的时间完成两个数的乘法，一台超小型机约1微秒，价值50000$的台式小型机约需3微秒，而高效的8088软件则要400微秒才完成。（双精度数要900微秒），然而在个人计算机上附加一个不太贵的8087，只要20-30微秒即可完成同样工作。

这是第一次，微型计算机在性能价格比上能达到大型机的$^1/_4$—$^1/_{20}$，但价格却是它的$^1/_{10}$—$^1/_{100}$。尽管对某些作业而言，在性能价格比方面大型机总是比微型机高，但配备8087的个人处理机是第一个可与大型机在商业上竞争的微型计算机。

大多数PC机用户关心如何使8087提高其个人计算机的计算速度更甚于关心中央计算机使用的方便性。配备8087的PC机的速度优势取决于应用和如何使用8087。（读完此书用户会知道如何获得最大限度的优势。）要明白8087的中心要点是：8087是一个数值处理器。8087仅仅提高数值运算程序的速度。如果PC的用户进行字符处理，那约有99%的工作与8087毫不相干。但是，若偶有一些数值运算，加上8087就好象在7月4日烟火节里交易火炮一般。

8087速度的优势在很大程度上取决于如何应用它，但可归结一点：8087使分钟变成秒。

## 专门速度的比较

从8087能得到多少好处取决于所使用的软件以及8087的硬件的速度。在第四章对速度问题会详尽地讨论。这里只作初步地论述。

8087在提高程序速度方面能为用户尽多大力依赖于这一点：即采用的软件所耗费的各种“额外开销”与数值计算所花费的时间之比。8087提高数值运算的速度，但对额外的时间开销仅起很小甚至不起作用。表1-1表明当8087与一个低开销，高速度的程序结合时所能预期的结果。

表1-1的时间是将（pre-8087）BASIC与专用8087程序相比较而得到的（后者可在以后的章节中找到）。这是8087与一个好的软件相组合后对速度的改善的一个典型例子。根据不同的应用，8087硬件可将速度提高到10—50倍，而余下的就依赖于低额外开销的软件了。如果用高额外开销的软件程序，则几乎无法得到良好的速度改善。（因需要，设置在计算机内的BASIC解释程序就是高额外开销的软件）由于8087仅仅提高数值计算的速度，使用高额外开销的软件，虽然花在数值计算上的时间相对少些，但额外开销和数值计算所花费时间的总和几乎是上表给出的数量，两者不会相差多少。尽管如此，速度性能的改善还是给人留下深刻的印象。

**表 1—1 BASIC与8087速度基准比较（以秒为时间单位）**

| 程序 | 50×50矩阵乘法 | 5000平方根 |
|---|---|---|
| BASIC | 1200 | 52 |
| 8087程序 | 8 | 0.35 |

## 使用8087都需要些什么设备？

当然，首先需要的是8087。这一点可通过购置一台配有8087的个人计算机或把8087插入原有的机器达到。8087可装配到任何由INTEL 8086、8088系列支持的个人计算机上。安装8087的难易程度取决于制造厂家在设计计算机时是否为8087留有位置。即使是没留出位置，也有可能加装。然而要做到这一点，需要有一定的技能。

请读者不必担心，大多数的生产厂家都为8087留有空位。尤其是当IBM公司（和其它制造兼容个人计算机的公司）推出其第一代个人计算机时，就在主电路板上留有8087标记的插座。只需将8087插入此空插座里即可。插8087芯片是很容易的。（我就无人帮忙安装了8087）它比在计算机内部的扩展槽上插一个印刷电路板还简单。（对计算机内部结构确实一无所知的用户，最好请人帮忙。毕竟计算机是一件价钱昂贵的设备）。

取走机盖用不了一分钟时间就能装上8087，同时你还可进行其它一些硬件修复工作。计算机上可能有per-8087软件。如固化到只读存贮器（ROM）的BASIC解释程序。如果能从厂家得到新的可用的8087兼容软件，用户会同时需要更高级的ROM芯片。

对于那些拥有非INTEL 8086，8088系列的个人计算机的用户，他们是否也能得到

8087速度方面的优点呢？很遗憾，回答是“不行”。8087只能与INTEL系列配套工作。然而，鉴于INTEL系列被广泛地采用，现在有一些企业销售带有INTEL处理器的电路板，适用于APPLE机和其它计算机。甚至有些电路板还配有8087或为8087预留一个插座。这些板，绝不会提高在原始处理器上运行程序的速度，但它允许使用本书提供的程序及其它的8087兼容软件。

# 第二章 INTEL 8087芯片

## 处理器与协处理器

计算机的“大脑”是它的“CPU”或中央处理单元。对于IBM个人计算机及其它的许多“第二代”个人计算机来说，它们的“大脑”是INTEL 8088。这是一个完整的通用中央处理单元芯片。它有一套完备的用于8位和16位整数运算、逻辑编程、输入输出的指令系统。与大多数的微处理机一样，8088缺少大型机上那些高级数学指令。

INTEL 8087数值处理器通过增加新的、复杂的数学指令，扩展了8088的指令系统。8087用高速硬件实现数学运算操作，而这种操作如果用软件执行则需要编写成千行操作码，并且8087硬件运行速度比此类软件快10-200倍。

从程序员的角度看，8087实际上给8088指令表增添了一些指令，并为处理器附加若干可用的内部寄存器。那么为什么不将这些功能都集中于一个芯片，而要做一个附加的器件呢？原因有几个：

- 8087是一个极其复杂的计算器件，它集75000个晶体管于一身。尽管8087对数值处理是有限度的，但它毕竟要比8088复杂得多（也贵得多）。将它们分别制于两个不同的芯片上一方面可以降低研制成本，另一方面便于用户和厂家根据不同的用途设计合适的系统；
- 8088（和具有16位总线的同代产品8086）比8087早几年投入市场。设计16位个人计算机时，一些厂家在IBM-PC机上就留有“协处理器插座”标记的空座，这样在8087问世后，使机器很方便地提高一个等级；
- 由于8087与8088是两个器件，它们能同时地执行指令。对于一个实际的程序而言，这就意味着，当8087在完成一个数值计算时，8088可准备下一条指令。

在本章的其余部分，我们一般性地叙述一下8087的性能。第五章将进一步提供技术上的讨论。

## 8087概述

8087是作为8088的一个协处理器工作的。8087监视由8088接受到的指令，它只处理与自己有关的指令，而允许8088指令通过。同样，8088也监视所有的指令，只处理它自己的指令，允许8087指令通过。8088为8087提供一项重要的服务：当遇见一条8087指令，8088计算出所必要的存储器地址，并使该地址为8087可用。然后8088立即着手处理下一指令。这样协处理器使8087与8088同时执行指令，显著地提高了整个系统的功能。

8087结构的突出点是具有八个80位的数据寄存器。这些寄存器构成一组“下推栈”，这种组织技巧导致快速向量运算，并使高级语言编译程序产生有效码。（第五章包括了有

关下推栈的操作详细论述）80位寄存器的宽度使8087能完成极其精确的计算。8087的指令可以识别存储器中七种不同的数据类型。当数据进入8087后，自动地变换成80位的内部表示形式，使程序员不用担心数据类型的转换。

## 指令类别

8087有68条指令，划分成6种类型。（分类说明很方便地描述了8087功能。但在使用时无需记住这种分类说明）6种类别是：

数据传送型（第六章讨论）。这些指令将数据在8087和内存之间来回传送，也可在8087内部寄存器中传递；

算术运算型（第六章讨论）。8087指令系统的核心是加法、减法、乘法、除法运算以及附加的如平方根、绝对值运算；

超越函数型（第十二章讨论）。8087的硬件具有对数和三角函数的运算功能（这些指令即使在大型机上也是少见的）；

常数型（第六、十二章讨论）。在8087中建立了七种最常用的常数如：0，1，π等；

比较型（第六、七章讨论）。这些指令用于小于、等于、大于及其他类似的测试；

处理器控制（第十二章讨论）。这类指令使程序员能全面地控制8087的状态。其中某些指令与比较指令和8088转移指令结合在一起控制程序的流向。

## 数据类型

对七种正规的8087数据类型在第五章深入地讨论。然而，从编制最通用的8087程序出发，确定几个因素是至关重要的。在BASIC中能直接使用的数据类型只有整数、单精度数和双精度数。一般而言，只是后两种适用于数值数据。如果使用8087主要是科学计算的编程，那么只需要记住三种数据类型：

1.单精度数（在8087术语中被称为短实数）具有6-7位的十进制数字精度，占内存4个字节；

2.双精度数（在8087的术语中被称为长实数）具有15-16位的十进制数字精度，占内存8个字节；

3.暂时数是用于8087内部的各种计算的。它保留多于18位的十进制数字精度。在内存中占有10个字节。

如果主要进行数值运算，那么这三种数据类型大概占所使用的数据量的95%。除此之外8087还能识别其它四种数据类型：

1.整数（在8087的术语中被称为字整数）在存储器中占两个字节。主要用于索引数组和其它数据结构。BASIC和8087的整数表示法是相同的。

2.短整数占4个字节。最大的（带符号）字整数是32768，而一个短实数可大到$2\times10^9$；

3.长整数占8个字节。比双精度实数多2或3位数字的精度，其最大值可达到$10^{18}$；

4.组合式十进制数是用于商业和数据处理的一种数据表示法。占内存10个字节，有18个十进制数字。与前三种数据类型不同，它的形式用十进制而不是用二进制表示。每一个0-9数字用四位二进制表示，两个十进制数字组合在一个字节里。

相对而言，8088硬件所能识别的数据类型局限于一至二个字节的二进制整数和较短的组合式十进制数。所有用pre-8087 BASIC和其它高级语言处理的数据都是由进行整数运算的软件来完成的。8087完全取消了这些软件，所以8087系统的工作不仅速度快，程序占用空间少，而且运算结果的可靠性高。

## 计算机是怎样调用8087的？

下一章将讨论用于8087的软件。为了便于明瞭为什么有些软件能与8087兼容而有些则不能，那么回顾一下建立8088与8087协同操作的基础是有益的。

在最初设计8088的指令系统时，就考虑到日后的扩展。8088诸指令中有一条称作“交权”指令。8088知道，这条交权指令实际上是调用8087操作的，所以根本就不理睬它，而让8087去处理。8087使用的指令是8088交权指令的不同变种。

当8088和8087都装入机中，可将这种结合看作为具有扩展功能的大型机。为了获得正确的结果，使用交权指令的软件一定要有8087的实体。由per-8087建立起的软件，由于不使用交权指令，也就得不到这一新功能所带来的优越性。

用机器语言编写程序，用户可以知道是否使用了交权指令。在使用计算机的大部分时间内，这样的直接内部细节是不会在用户的控制之下的。下一章，将讨论8087兼容软件的变种和一些非8087兼容软件。

# 第三章　购买和建立8087兼容软件

在购买或建立用于8087的软件时，什么是需要特别考虑的呢？第一个问题将是“什么样的软件适于8087？”第二个问题是“怎样工作得好？”这一章将对软件兼容性分成三部分分析。第一部分是关于兼容性的某些重要的技术细节；第二部分分析为何有些软件能生成快速运行的程序，而有些则不行；最后一部分，以编程方便和计算速度方面讨论各种类型软件的特点。

## 兼容性——技术细节

假设能查看已被译成机器语言的程序，该程序既可使用驱动8087的机器语言指令——上章中提到的“交权指令”，也可以不用。如果不使用这些指令，则程序与8087无关，运行时8087可有可无，且两种情况下运行速度相同。如果程序中使用8087指令，当然机器中必须配置8087。

至此，又出现了第二个问题，即与软件设计的细节有关，它与兼容性同等重要。计算机在其内部都是用“0”和“1”的组合形式表示数字的。计算机不同，同一个数字的内部表示格式也不同。通常，用户并不介意计算机采用什么内部格式，因为他无需单独地处理这些“0”和“1”。重要的是计算机的硬件应该知道如何解释它自己的格式。（8087所采用的表示法已被推荐作为工业标准。将在第五章说明8087的表示的格式。）

在8087问世前，由8088系列支持的个人计算机的硬件只有整数运算功能。由于硬件没有表示非整数的硬性规定，所以每个软件的设计者都可以自由地选择自己的表示格式。实际上，这意味着不论是谁建立了编程语言的翻译程序（编译程序，解释程序，和汇编语言）都可以自行决定他所使用的专门语言。由于Microsoft是16位计算机的编程语言的主要支柱，所以许多大块的软件都使用由 Microsoft公司所选择的格式。

很遗憾，Microsoft的格式与INTEL 8087的格式不同。

这一冲突的结果使pre-8087软件和8087兼容软件不能够交换用它们各自的内部格式所描述的数据。但配置了8087，就能安全地使用per-8087软件或8087兼容软件。但如果试图组合由per-8087和8087兼容软件产生的程序，那往往得到混乱的信息。另外，如果数据是以计算机的内部格式存储的，而试图在这样的程序之间交换数据，那得到的更是毫无用处的数据。然而数据若不是用内部格式表示的，则程序之间一般能交换数据。

两种软件之间是否会发生冲突，这没有普遍的规则。需要了解各个程序的特殊性。在本章的第三部分，将给出一些寻找此类麻烦的例子。

## 是什么造成程序的运行快慢？

由三个基本点决定程序的运行速度：解决问题的方法（计算机科学家称之为“算