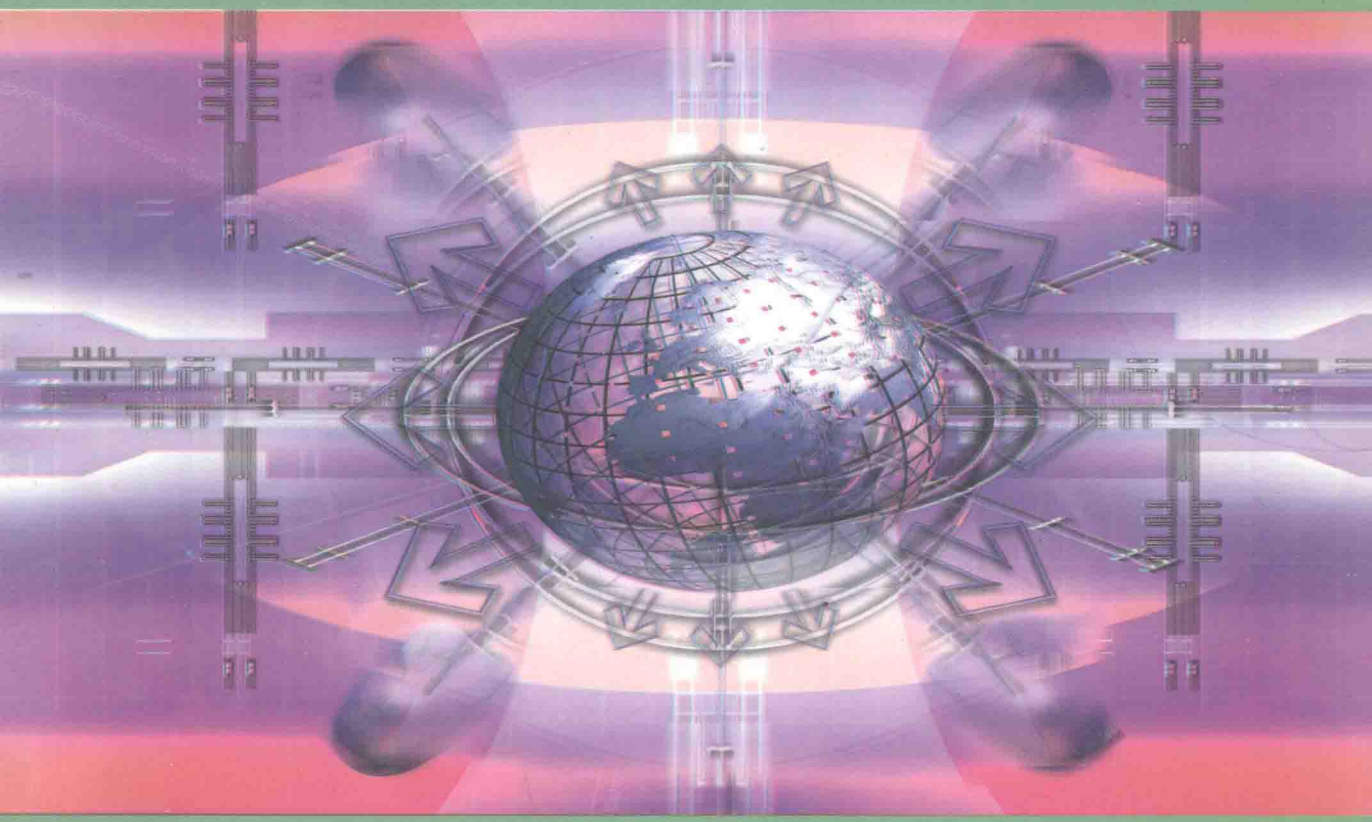
高等学校试用教材

16–32 位
汇编语言程序设计

周国运　等编著



计算机科学与技术系

内 容 提 要

本书分为两部分。第一部分是 16 位汇编语言，介绍了 8086/8088CPU 的寄存器、寻址方式和指令系统；讲述了汇编语言的语法、伪指令、宏、复杂数据等程序设计的基础知识；主要讲解了汇编语言的程序设计，包括基本程序设计、中断和输入／输出程序设计、以及 DOS 功能调用、子程序设计等。第二部分是 32 位汇编语言，介绍了 80386CPU 的寄存器、扩展的寻址方式和新增的指令、存储器管理机制和工作模式；举例讲解了 Win32 汇编语言的编写方法，这些例子包括消息框和对话框产生、动态链接库、中断与异常。

对于 16 位汇编使用 MASM 或 TASM 为汇编器，以 DOS 操作系统为运行平台。对于 32 位汇编语言，使用 MASM6.0 以上版本作为汇编器，可以在 Windows 操作系统下运行。

本书具有结构清晰、内容精炼、例题丰富和习题难易适当等特点。本书可以作为计算机专业本科及专科汇编语言课程的教材或参考书，也可以作为 Windows 程序设计人员的参考书，还可以作为其它高级语言程序设计人员的参考书。

16－32 位汇编语言程序设计／周国运等编著

Ⅰ.16… Ⅱ.周… Ⅲ.汇编语言-程序设计-教材 Ⅳ.TP312

印 数：1～900 册

定 价：20.00 元

前　言

汇编语言是面向机器的低级语言，通过学习汇编语言，才能真正理解计算机的工作原理和工作过程，才能深入地了解高级语言的一些概念。应用汇编语言，程序员可以直接操纵计算机的硬件，用汇编语言，才能编写出运行速度快、占有空间小的高效程序。即便是在高级语言功能非常强大的今天，一些程序设计语言不断被淘汰，新的优秀的编程语言不断出现，汇编语言仍然处于重要地位，发挥着它的重要作用，并且不能由其它语言所替代。

“汇编语言程序设计”，是计算机各专业的一门重要基础课，也必修的核心课程之一，它是“操作系统”和“微机原理与接口技术”等其它核心课程的先修课。

计算机技术在发展，汇编语言也在发展，汇编语言教学也需要发展和改革，为了适应计算机技术应用的发展和变化，为了适应计算机应用型本科和专科教学改革的要求，根据我们对“汇编语言程序设计”课程教学的体会，组织编写了这本“汇编语言程序设计教程”。其指导思想是着重于基本概念、指令和基本程序设计；对输入输出、DOS 功能调用、中断等程序设计，以及 32 位汇编程序设计，只要求学生能够理解或者有所了解，为后续的课程打下基础，以适应少学时的教学要求。

本书可以作为计算机专业本科及专科汇编语言课程的教材或参考书，也可以作为 Windows 程序设计人员的参考书，还可以作为其它高级语言程序设计人员的参考书。

本书的编写工作得到了学院、教务处、计算机系的支持，在此表示感谢。

由于作者水平有限，书中难免有错误和不妥之处，恳请读者给予指正和提出修改意见。

作　者

2003 年 8 月

目　录

第1章　绪论

1.1 汇编语言概述

1.1.1 汇编语言

按照确定的算法解决具体问题所必须的指令序列称为程序，它是由数据、指令和字符等构成的，在执行前应预先将它们以二进制代码形式存储在存储单元中。

所谓**指令**是指控制计算机执行某一特定操作的命令。而一台计算机所能识别的指令的全体称为指令系统，它反映计算机基本功能的强弱。机器指令是以二进制代码形式表示的，能直接为计算机识别而执行的命令，它通常由操作码和操作数两部分组成。8086/8088 指令系统有约 100 多条基本指令。

汇编语言是一种面向机器的语言，汇编语言的指令与机器指令是一一对应的。它用符号、文字来表示指令，所以它又称符号语言。用汇编语言编写的程序是不能被计算机直接识别和执行的（如同用高级语言编写的程序），它需要翻译成目标程序后方可执行，这个过程我们称为汇编。汇编语言虽然没有高级语言在使用上简单方便，但因它与机器语言是一一对应的，故可充分利用计算机硬件系统的特性，提高编程技巧和编程质量。另外，利用汇编语言处理 I/O 设备是汇编语言的独到之处,所以它是无法为其他语言所取代的。

汇编程序的类型有：自汇编程序、交叉汇编程序、微汇编程序、浮动汇编程序和宏汇编程序。

汇编语言(ASM)虽然较机器语言在阅读、记忆及编写方面都前进了一大步，但对描述任务、编程设计仍感不便，于是产生了具有机器语言优点，而又能较好地面向问题的语言，即**宏汇编语言**(MASM)。

宏汇编语言不仅包含一般汇编语言的功能，而且用了高级语言使用的数据结构，是一种接近高级语言的汇编语言。例如它提供了记录、结构和字符串操作；具有宏处理、条件汇编及磁盘操作系统 DOS 功能调用等多种功能;程序的开发以及调试手段也比较完善，因而宏汇编语言是一种更高级的汇编语言。

1.1.2 汇编语言的特点

由于汇编语言使用指令助记符和符号地址，所以他要比机器语言容易掌握得多。与高级语言相比，汇编语言具有以下特点：

（1）汇编语言与及其关系密切
（2）汇编语言程序效率高
（3）编写汇编语言源程序繁琐
（4）汇编语言程序调试困难

1.1.3 汇编语言的主要应用场合：

（1）程序执行占用较短的时间，或者占用较小存储容量的场合。
（2）程序与计算机硬件密切相关，程序直接控制硬件的场合。
（3）需提高大型软件性能的场合。
（4）没有合适的高级语言的场合。

1.2 数据表示和类型

1.2.1 进位计数制以及不同计数制间的相互的转换

在日常生活中我们所用的是十进制数,但在计算机中我们用的是二进制数,作为汇编语言程序设计对二进制数与十进制数的转换是必不可少的。

1. 各种进位计数制

（1）十进制数

十进制数就是我们在日常生活中所用的数,它共有 0,1,2,3,4,5,6,7,8,9 这十个数码,它的计数方法是“逢十进一”。对于十进制数的运算大家应该都知道,这里就不多说了。

（2）二进制数

二进制数是最简单的进位计数制,它只有 0,1 二个数码，计数方法是“逢二进一”。二进制数的运算十分简单，加法是“逢二进一”，减法是“借一当二”。例如“11+10=101;101-10=11”至于乘法和除法和我们日常相似,只不过不是“逢十进一”而是“逢二进一”罢了。

（3）八进制数和十六进制数

很显然八进制数是 0 到 7 这八个数码组成，且是“逢八进一”,而十六进制数是由 0 到 9 和 A，B,C,D,E 和 F(英语大写字母分别代表 10 到 15)这十六个数码组成的,它的进位方式是“逢十六进一”。在下面我将给出二,八,十和十六进制数码的对照关系表。如表 1.1 所示。

表 1.1　二、八、十和十六进制数码的对照关系表

十进制	二进制	八进制	十六进制
0	0	0	0
1	01	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10

2.数制的转换

（1）将十进制数转换成二进制数

例　试将十进制数 125.6875 转换成二进制数。

①整数的转换：除 2 取余

先将 125 除以 2 在右边记下余数，然后以此类推，直到最后的差为 0,最后我们将余数倒着排列即得结果。

②小数的转换：乘 2 取整

小数的转换正好与整数的转换相反，先将 0.6875 乘以 2 得结果 1.3750，则在右边记下整数位 1,再将去整数位后的 0.375 乘以 2 记下结果，依此类推，直到最后小数部分为 0 或结果以达到精度要求。

（2）将十进制数转换成八/十六进制数

方法如上,只不过乘/除数为 8 或 16。

（3）二,八/十六进制数转换成十进制数

转换方法是利用数制的一般表达式写成相应的幂运算形式，然后求和得到。

例：$572.34(8)=5*8^2+7*8+2+3*8^{-1}+4*8^{-2}=378.4375$

（4）二进制数与八进制数的转换

因为 $2^3=8$，所以二进制数转换成八进制数只需将二进制数从小数点开始每 3 位转成一位八进制数(整数由左向右,小数相反)。 例如：101 111 010 . 011 100(2) =572.34(8)

八进制数转换成二进制数只需将每一位八进制数用三位二进制数表示，小数点位置不变。 例如：175.54(8)=001 111 101.101 100B=1111101.1011B。

至于十六进制数与二进制数的转换只需将每隔 3 位改为每隔 4 位即可。

1.2.2 原码,补码和反码

在计算机中参加运算的数有正负之分,通常在计算机中我们用 $X=X_0X_1X_2\cdots\cdots X_{N-1}$ 来表示一个二进制数，并规定当 $X_0=0$ 时 X 为正数，$X_0=1$ 时 X 为负数.在计算机中这种表示法有原码,补码和反码三种。

1. 原码

原码的定义为：$$X_{原}=\begin{cases} X & 0\le X<2^{n-1} \\ 2^{n-1}+|X| & -2^{n-1}<X\le 0 \end{cases}$$

$X_{原}$所能表示数的范围为：$[-(2^{n-1}-1),(2^{n-1}-1)]$

例：根据[X](原)所能表示的整数范围公式，我们可以计算出当 n=8 时，原码表示范围是[-127,127]。原码的表示法简单易懂,但是它最大的缺点是运算复杂。所以人们引进了补码。

2. 补码

补码的定义为：$$X_{补}=\begin{cases} X & 0\le X<2^{n-1} \\ 2^{n-1}-|X| & -2^{n-1}\le X<0 \end{cases}$$

$X_{补}$所能表示数的范围为：$[-2^{n-1},(2^{n-1}-1)]$

例：根据[X](补)所能表示的整数范围公式我们可以计算出当 n=8 时补码表示范围是[-128,127];根据补码的定义公式我们可以计算出任何一个数的补码,如 X=-2FH 根据公式[X](补)=100H-2FH=D1H。

用补码进行加减运算是很简单的,公式为[X+Y]补=[X]补+[Y]补;[X-Y]补=[X]补+[-Y]补。加法公式是非常简单的,但减法中我们只知道[Y]补而不知道[-Y]补,利用一个口诀就可解决这个问题,口诀是"将[Y]补连同符号位一起按位求反后末为加一可得[-Y]补"。我们

现在虽然有了公式和口诀,但是还有符号位的问题没有解决,而另一个口诀可以解决这个问题。口诀是"符号位参加运算,符号位相加,若有进位,则进位舍去"。如:63H-72H=63H+8EH=0F1H(用十六进制时最高位为字母时前加 0)。

注意:运算有时会产生溢出,如 57+81>127。当我们用 N=8 的补码运算就会溢出,因为当字长 N=8 时补码的表示范围是 [-128, 127]。

3. 反码

反码的定义为: $X_{反}=\begin{cases} X & 0 \le X < 2^{n-1} \\ 2^{n-1}-1-|X| & -2^{n-1} < X \le 0 \end{cases}$

$X_{反}$所能表示数的范围为: $[-(2^{n-1}-1),(2^{n-1}-1)]$

例: 根据[X](反)所能表示的整数范围公式,我们可以计算出当 n=8 时反码表示范围是 [-127,127]。

1.2.3 数的定点和浮点表示法

在计算机中,针对小数点的处理有两种方法:定点表示法浮点表示法。

1. 定点表示法

定点表示法就是小数点固定在某个位置上。在定点计算机中,为了简单通常将小数点定在最高位(即纯小数)或将小数点定在最低位(即整数)。

2. 浮点表示法

浮点表示法就是小数点的位置并不固。浮点数在计算机中通常的表示形式为"浮点数=2 的正/负阶码次方*尾数"其中阶码是个正整数,尾数是个小数,我们规定尾数的区间为[0.5,1],如果尾数不在此区间,那我们可通过调节阶码来满足区间,此方法称为规格化。

1.2.4 BCD 码和字符编码

在日常生活中人们用的是十进制数,而机器又只能处理二进制数,因此我们引进了 BCD 码。BCD 码是用四位的二进制数来表示一位十进制数(可参照下表)。例子:我现在将 8351 表示成 BCD 码为 1000 0011 0101 0001。当用一个字节来表示十进制数时,称为非压缩的 BCD 码。例如 6 可表示为 00000110。当用一个字节表示二个十进制数时,称为压缩的 BCD 码。例如 79 可表示为 01111001。

除了数值数据外,计算机还可处理人们常用的符号,如字母,标点符号等。在计算机中这些符号是用 ASCII 码来表示的。ASCII 码用一个字节的二进制数来表示一个字符,但实际只用了七位,最高位被用来做奇偶校验位。这一位置 1 或 0,使字节含 1 的个数为奇数(或偶数)称为奇校验或偶校验。如表 1.2 十进制数与 BCD 码对换关系表。

表 1.2 十进制数与 BCD 码对换关系表

十进制数	BCD 码	十进制数	BCD 码
0	0000	5	0101
1	0001	6	0110
2	0010	7	0111
3	0011	8	1000
4	0100	9	1001

1.3 Intel 系列 CPU 简介

1.3.1 Intel 系列 CPU 简介

由于汇编语言与 CPU 关系密切,所以先对 Intel 系列 80x86 的 CPU 做一下简单介绍。

CPU 是 Central Processing Unit——中央处理器的缩写，它是计算机中最重要的一个部分，由运算器和控制器组成，如果把计算机比作一个人，那么 CPU 就是他的心脏，其重要作用由此可见一斑。不管什么样的 CPU， 其内部结构归纳起来可以分为控制单元、逻辑单元和存储单元三大部分，这三个部分相互协调，便可以进行分析、判断、运算并控制计算机各部分协调工作。

CPU 从最初发展至今已经有二十多年的历史了，这期间，按照其处理信息的字长，CPU 可以分为：四位微处理器、八位微处理器、十六位微处理器、三十二位微处理器以及六十四位微处理器等等。

1971 年，早期的 Intel 公司推出了世界上第一台微处理器 4004，这便是第一个用于计算机的四位微处理器， 它包含 2300 个晶体管，由于性能很差，其市场反应十分不理想。

随后，Intel 公司又研制出了 8080 处理器、8085 处理器，加上当时 Motorola 公司的 MC6800 微处理器和 Zilog 公司的 Z80 微处理器，一起组成了八位微处理器的家族。

十六位微处理器的典型产品是 Intel 公司的 8086 微处理器，以及同时生产出的数学协处理器，即 8087。这两 种芯片使用互相兼容的指令集，但在 8087 指令集中增加了一

些专门用于对数、指数和三角函数等数学计算指令，由于这些指令应用与 8086 和 8087，因此被人们统称为 X86 指令集。此后 Intel 推出的新一代的 CPU 产品，均兼容 原来的 X86 指令。

1979 年 Intel 推出了 8088 芯片，它仍是十六位微处理器，内含 29000 个晶体管，时钟频率为 4.77MHz，地址总 线为 20 位，可以使用 1MB 内存。8088 的内部数据总线是 16 位，外部数据总线是 8 位。1981 年，8088 芯片被首次用 于 IBM PC 机当中，如果说 8080 处理器还不为各位所熟知的话，那么 8088 则可以说是家喻户晓了，个人电脑 PC 机的第一代 CPU 便是从它开始的。1982 年的 80286 芯片虽然是 16 位芯片，但是其内部已包含 13.4 万个晶体管，时钟频率也达到了前所未有的 20MHz。其内、外部数据总线均为 16 位，地址总线为 24 位，可以使用 16MB 内存，可使用的工作方式包括实模式和保护模式两种。

三十二位微处理器的代表产品首推 Intel 公司 1985 年推出的 80386，这是一种全三十二位微处理器芯片，也是 X86 家族中第一款三十二位芯片，其内部包含 27.5 万个晶体管，时钟频率为 12.5MHz，后逐步提高到 33MHz。8 0386 的内部和外部数据总线都是 32 位，地址总线也是 32 位，可以寻址到 4GB 内存。它除了具有实模式和保护模式 以外，还增加了一种虚拟 86 的工作方式，可以通过同时模拟多个 8086 处理器来提供多任务能力。1989 年 Intel 公司又推出准三十二位处理器芯片 80386SX。它的内部数据总线为三十二位，与 80386 相同，外部数据总线为十六 位。也就是说，80386SX 的内部处理速度与 80386 接近，也支持真正的多任务操作，而它又可以接受为 80286 开发 输入/输出接口芯片。80386SX 的性能优于 80286，而价格只是 80386 的三分之一。386 处理器没有内置协处理器，因此不能执行浮点运算指令，如果您需要进行浮点运算时，必须额外购买昂贵的 80387 协处理器芯片。

八十年代末九十年代初，80486 处理器面市，它集成了 120 万个晶体管，时钟频率由 25MHz 逐步提升到 50MHz 。80486 是将 80386 和数学协处理器 80387 以及一个 8KB 的高速缓存集成在一个芯片内，并在 X86 系列中首次使用了 RISC（精简指令集）技术，可以在一个时钟周期内执行一条指令。它还采用了突发总线方式，大大提高了与内 存的数据交换速度，由于这些改进，80486 的性能比带有 80387 协处理器的 80386 提高了 4 倍。早期的 486 分为有协 处理器的 486DX 和无协处理器的 486SX 两种，其价格也相差许多。随着芯片技术的不断发展，CPU 的频率越来越快 ，而 PC 机外部设备受工艺限制，能够承受的工作频率有限，这就阻碍了 CPU 主频的进一步提高，在这种情况下， 出现了 CPU 倍频技术，该技术使 CPU 内部工作频率为处理器外频的 2—3 倍，486DX2、486DX4 的名字便是由此而来 。

九十年代中期，全面超越 486 的新一代 586 处理器问世，为了摆脱 486 时代处理器名称混乱的困扰，最大的 C PU 制造商 Intel 公司把自己的新一代产品命名为 Pentium（奔腾）以区别 AMD 和 Cyrix 的产品。AMD 和 Cyrix 也分别 推出了 K5 和 6x86 处理器来对付 Intel，但是由于奔腾处理器的性能最佳，Intel 逐渐占据了大部分市场。

在 2000 年的 6 月，Intel 公司宣布了其开发的下一代 CPU 命名为 Pentium 4，也就是曾经命名为 Willamette 的 CPU。在 2000 年的 11 月，Intel 正式发布了 Pentium 4 处理器，

该处理器没有采用 P6 架构，而是采用了全新的 NetBurst 架构，Pentium 4 的管线长度是 P6 架构的两倍，达到了 20 级。现在的 PIII Coppermine 由于受到管线的限制，最高只能达到 1.2GHz，这一点从 1.13GHz 的 PIII 被回收就能看出来。管线的加长可以使得 Pentium 4 能达到更高的时钟频率，但是也使 Pentium 4 在每个时钟周期中的处理的命令数目比 PIII Coppermine 少，这就是为什么现在相同的速度下，Pentium 4 的性能表现不如 PIII Coppermine 和 T-bird 的原因，但是随着 Pentium 4 速度的提高，会克服这一点的。

最初的 Pentium 4 采用 0.18 微米制造工艺，集成 4200 万个晶体管，芯片面积 213 平方毫米，核心电压 1.7V，目前采用的是 Socket423 接口，而 Pentium 4 的最终版本会采用 Socket 478 接口。 Pentium 4 集成了 8KB 的 L1 Cache，使用的是低于 1.42ns 的高速缓存，拥有极低的寻找时间，能迅速地找到并且命中目标指令，大大的提高了 CPU 的工作效率。Pentium 4 还拥有全速的 256KB 二级缓存，在处理器核心和 L2 Cache 之间有着更大的数据传输通道，数据传输率可以达到前所未有的 44.8GB/s，几乎是 PIII 1G（16GB/s）的 3 倍之多。

Pentium 4 的总线频率高达 400MHz，是目前 PC133 总线的 Pentium III 的三倍，如果配合双通道的 RAMBUS 内存，可以在处理器和内存控制器之间提供高达 3.2GB/sec 的内存通道。 Pentium 4 的算术逻辑单元（ALU）一核心的两倍运行，还包含了 144 条重新设计的 SSE2 指令集。

1.3.2 CPU 专业术语

Intel 公司的 X86 序列 CPU 以及其它公司所生产的兼容产品，是目前世界上个人电脑中装机最多的芯片。每当各种媒体介绍或评价这类 CPU 时，经常会提到诸如“流水线”、“乱序执行”和“分枝预测”等专业术语。

1. IA-32&IA-64

IA 是英语“英特尔体系/IntelArchitecture”的缩写。这是因为目前使用的 CPU 以 Intel 公司的 X86 序列产品为主，所以人们将 Intel 生产的 CPU 统称为英特尔体系（IA）CPU。由于其它公司如 AMD 等公司生产的 CPU 基本上能在软、硬件方面与 Intel 的 CPU 兼容，所以人们通常也将这部份 CPU 列入 IA 系列。

由于目前使用的 CPU，包括新推出的 PentiumIII 都还是 32 位的，所以又被列为 IA-32。而 IA-64 就是 Intel 下一步将推出的 64 位 CPU，但其物理结构和工作机理与目前的 X86 序列的 IA-32CPU 完全不同。

2. CPU 的位和字长

位　在数字电路和电脑技术中采用二进制，代码只有“0”和“1”，其中无论是“0”或是“1”在 CPU 中都是一“位”。

字长 电脑技术中对 CPU 在单位时间内（同一时间）能一次处理的二进制数的位数叫字长。所以能处理字长为 8 位数据的 CPU 通常就叫 8 位的 CPU。同理 32 位的 CPU 就能在单位时间内处理字长为 32 位的二进制数据。

字节和字长的区别 由于常用的英文字符用 8 位二进制数就可以表示，所以通常就将 8 位称为一个字节。字节的长度是固定的，而字长的长度是不固定的，对于不同的 CPU，字长的长度也不一样。8 位的 CPU 一次只能处理一个字节，而 32 位的 CPU 一次就能处理 4 个字节，同理字长为 64 位的 CPU 一次可以处理 8 个字节。

3. CPU 外频

CPU 外频也就是常见特性表中所列的 CPU 总线频率，是由主板为 CPU 提供的基准时钟频率，而 CPU 的工作主频则按倍频系数乘以外频而来。在 Pentium 时代，CPU 的外频一般是 60/66MHz，从 Pentium□350 开始，CPU 外频提高到 100MHz。由于正常情况下 CPU 总线频率和内存总线频率相同，所以当 CPU 外频提高后，与内存之间的交换速度也相应得到了提高，对提高电脑整体运行速度影响较大。

4. CPU 主频

CPU 主频也叫工作频率，是 CPU 内核（整数和浮点运算器）电路的实际运行频率。在 486DX2CPU 之前，CPU 的主频与外频相等。从 486DX2 开始，基本上所有的 CPU 主频都等于“外频乘上倍频系数”了。

5. 流水线技术

流水线（pipeline）是 Intel 首次在 486 芯片中开始使用的。流水线的工作方式就象工业生产上的装配流水线。在 CPU 中由 5~6 个不同功能的电路单元组成一条指令处理流水线，然后将一条 X86 指令分成 5~6 步后再由这些电路单元分别执行，这样就能实现在一个 CPU 时钟周期完成一条指令，因此提高 CPU 的运算速度。从图 1a 中我们可以了解，由于 486CPU 只有一条流水线，通过流水线中取指令、译码、产生地址、执行指令和数据写回五个电路单元分别同时执行那些已经分成五步的指令，因此实现了 486CPU 设计人员预期的在每个时钟周期中完成一条指令的目的（按笔者看法，CPU 实际上应该是从第五个时钟周期才达到每周期能完成一条指令的处理速度）。到了 Pentium 时代，设计人员在 CPU 中设置了两条具有各自独立电路单元的流水线，因此这样 CPU 在工作时就可以通过这两条流水线来同时执行两条指令，因此在理论上可以实现在每一个时钟周期中完成两条指令的目的。

6. 超流水线

超流水线（superpiplined）是指某型 CPU 内部的流水线超过通常的 5~6 步以上，例如 Pentiumpro 的流水线就长达 14 步。将流水线设计的步（级）数越多，其完成一条指令的速度越快，因此才能适应工作主频更高的 CPU。□这一点我们可以用日常事例来说

明，比如栽树时由 5 个人同时栽 10 棵（一人两棵）□所完成的速度当然没有 10 人同时栽（一人一棵）所完成的速度快。

7. 超标量技术

超标量（superscalar）是指在 CPU 中有一条以上的流水线，并且每时钟周期内可以完成一条以上的指令，这种设计就叫超标量技术。

8. 乱序执行技术

乱序执行（out-of-orderexecution）是指 CPU 采用了允许将多条指令不按程序规定的顺序分开发送给各相应电路单元处理的技术。比方说程序某一段有 7 条指令，此时 CPU 将根据各单元电路的空闲状态和各指令能否提前执行的具体情况分析后，将能提前执行的指令立即发送给相应电路执行。当然在各单元不按规定顺序执行完指令后还必须由相应电路再将运算结果重新按原来程序指定的指令顺序排列后才能返回程序。这种将各条指令不按顺序拆散后执行的运行方式就叫乱序执行（也有叫错序执行）技术。

采用乱序执行技术的目的是为了使 CPU 内部电路满负荷运转并相应提高了 CPU 的运行程序的速度。这好比请 A、B、C 三个名人为晚会题写横幅"春节联欢晚会"六个大字，每人各写两个字。如果这时在一张大纸上按顺序由 A 写好"春节"后再交给 B 写"联欢"，然后再由 C 写"晚会"，那么这样在 A 写的时候，B 和 C 必须等待，而在 B 写的时候 C 仍然要等待而 A 已经没事了。但如果采用三个人分别用三张纸同时写的做法，那么 B 和 C 都不必须等待就可以同时各写各的了，甚至 C 和 B 还可以比 A 先写好也没关系（就象乱序执行），但当他们都写完后就必须重新在横幅上（自然可以由别人做，就象 CPU 中乱序执行后的重新排列单元）按"春节联欢晚会"的顺序排好才能挂出去。

9. 分枝

分枝（branch）是指程序运行时需要改变的节点。分枝有无条件分枝和有条件分枝，其中无条件分枝只需要 CPU 按指令顺序执行，而条件分枝则必□须根据处理结果再决定程序运行方向是否改变。因此需要"分枝预测"技术处理的是条件分枝。

10. 分枝预测和推测执行技术

分枝预测（branch prediction）和**推测执行**（speculationexecution）是 CPU 动态执行技术中的主要内容，动态执行是目前 CPU 主要采用的先进技术之一。采用分枝预测和动□态执行的主要目的是为了提高CPU的运算速度。推测执行是依托于分枝预测基础上的，在分枝预测程序是否分枝后所进行的处理也就是推测执行。

由于程序中的条件分枝是根据程序指令在流水线处理后结果再执行的，所以当 CPU 等待指令结果时，流水线的前级电路也处于空闲状态等待分枝指令，这样必然出现时钟周期的浪费。如果 CPU 能在前条指令结果出来之前就能预测到分枝是否转移，那么就可以提前执行相应的指令，这样就避免了流水线的空闲等待，相应也就提高了 CPU 的运算速度。但另一方面一旦前指令结果出来后证明分枝预测错误，那么就□必须将已经装入