

# 微机原理 及应用

王根义 © 主编

WEIJI YUANLI JI YINGYONG



中国轻工业出版社

全国百佳图书出版单位

# 微机原理及应用

主 编 王根义

副主编 沙晓艳 赵 魁 宋 龙

 中国轻工业出版社

## 图书在版编目 (CIP) 数据

微机原理及应用 / 王根义主编. — 北京: 中国轻工业出版社, 2015. 8

ISBN 978-7-5184-0594-7

I. ①微… II. ①王… III. ①微型计算机 IV. ①TP36

中国版本图书馆 CIP 数据核字(2015)第 204285 号

## 内 容 简 介

本书主要介绍微机的结构、微机工作原理、汇编语言程序设计和微机的输入/输出接口技术。内容包括:微机基础知识、微机组成、微型计算机系统的编程结构、汇编语言、8086/8088 总线时序、半导体存储器、输入/输出和中断系统、可编程接口芯片及应用等。

本书主要适用于高职院校理工类各专业学生作为教材使用,也可以作为大学本科院校工科专业的学生和研究生作为教材使用,同时也可以作为从事微机软硬件开发工作的科技人员的自学和参考资料。

责任编辑:陈 萍 责任终审:劳国强 封面设计:刘志伟

策划编辑:林 媛 版式设计:胡亚丽 责任监印:张 可

出版发行:中国轻工业出版社(北京东长安街 6 号,邮编:100740)

印 刷:北京市宏泰印刷有限公司

经 销:各地新华书店

版 次:2015 年 8 月第 1 版第 1 次印刷

开 本:787×1092 1/16 印张:12.5

字 数:312 千字

书 号:ISBN 978-7-5184-0594-7 定价:35.00 元

邮购电话:010-65241695 传真:85113293

网 址:<http://www.chlip.com.cn>

Email:[club@chlip.com.cn](mailto:club@chlip.com.cn)

如发现图书残缺请直接与我社邮购联系调换

150827J2X101HBW



## 前 言

《微机原理及应用》教材的相关知识和技术是从事计算机系统开发、组装、应用和维护工作人员必备的知识和技术。汇编语言不仅能够直接操作计算机硬件,而且汇编语言源文件生成的目标代码效率很高,能够满足数据处理和过程控制中对实时性的严格要求,所以,任何其他计算机语言不能取代它。

本书切实体现了编者的指导思想:面向学生、面向应用、面向技能和面向教学。

本书主要有以下特点:

(1)说明到位 对概念和方法不仅大都用编者辛苦推敲出来的精炼语言进行准确的叙述,不含影响读者思维的多余语句,而且很多情况都配以实例和上机实验来加强说明。另外,由于编者对本书涉及的硬件和软件的性能都经过上机和实验测试,所以在说明它们的含义和性能时,都用明确、肯定和到位的语言叙述得非常清楚,没有含糊不清之处。

(2)逻辑性强 在教材内容结构和次序的安排上进行了大刀阔斧地改革,使教材内容结构和顺序符合人类认识事物的规律,体现出“启发式”和“案例教学”的方针,符合逻辑,能提起学者学习和阅读的兴趣。例如,编者根据教学经验,知道读者在学习十进制数调整指令时,有一种神奇感和怀疑感,于是就在教材最恰当的位置安排了用 debug 软件使用和验证十进制数调整指令的实验截图,从而使学生轻装上阵,学习过程能够踏实顺利地进行。

(3)突出技能 技能不完全是知识,是结合相关知识练出来的本事。在本书编写过程中,编者针对学生的职业技能培养目标选用了足够的与当前社会生产实践紧密结合例题和习题,尤其是把我校在校企合作中遇到的真实案例经过适当修改之后成功地引入教材中,使学生在学时如置身于真实的工作环境中,进行模拟练习,以训练提高实践能力。

(4)突出应用 在本书编写过程中,每编写一部分内容,编者都要围绕“学生学习这部分知识对以后就业和工作有什么用处,和后续学习其他计算机知识有什么关系”的核心思想进行取舍和编辑。

(5)精简全面 编者既删去了以往教材中冗余的、重复的、过时的和在实际中不用的内容,又在教材中尽可能多地纳入了流行和实用的内容。

本书由王根义教授负责组织、统稿和主审工作。本书学习情境4、学习情境8和全部习题由王根义编写；学习情境6和学习情境7由沙晓艳编写；学习情境3和学习情境5由赵魁编写；学习情境1由宋龙编写；学习情境2的2.1.1子领域由任锁平编写；学习情境2的2.1.2子领域由朱思鸣编写；学习情境2的2.2.1子领域由郑亚锋编写；学习情境2的2.2.2子领域、2.2.6子领域、2.2.7子领域和2.2.12子领域由杨小康编写；学习情境2的2.2.3子领域由韩玉科编写；学习情境2的2.2.4子领域由张云云编写；学习情境2的2.2.5子领域由巨欢乐编写；学习情境2的2.2.8子领域由王岚编写；学习情境2的2.2.9子领域由范雯编写；学习情境2的2.2.10子领域由李萍编写；学习情境2的2.2.11子领域由周薇编写；学习情境2的2.3子领域由牛春艳编写。

由于编者水平有限，书中难免存在不足之处，敬请读者批评指正。

编者

## 目 录

|                           |    |
|---------------------------|----|
| 学习情境 1 微型计算机基础知识 .....    | 1  |
| 1.1 微型计算机概述 .....         | 1  |
| 1.1.1 微型计算机的发展 .....      | 1  |
| 1.1.2 微型计算机的特点 .....      | 3  |
| 1.1.3 微型计算机的应用领域 .....    | 4  |
| 1.1.4 微型计算机的发展趋势 .....    | 4  |
| 1.2 计算机中的数和编码系统 .....     | 4  |
| 1.2.1 计算机中的进位计数制 .....    | 4  |
| 1.2.2 计算机中对带符号数的表示 .....  | 7  |
| 1.2.3 数据和字符编码 .....       | 7  |
| 1.2.4 微机主要性能指标 .....      | 9  |
| 习题一 .....                 | 10 |
| 学习情境 2 微机组成 .....         | 11 |
| 2.1 概述 .....              | 11 |
| 2.1.1 硬件子系统 .....         | 12 |
| 2.1.2 软件子系统 .....         | 13 |
| 2.2 PC 的硬件子系统 .....       | 13 |
| 2.2.1 显示器 .....           | 13 |
| 2.2.2 显示卡 .....           | 15 |
| 2.2.3 打印机 .....           | 17 |
| 2.2.4 键盘 .....            | 18 |
| 2.2.5 鼠标 .....            | 18 |
| 2.2.6 音箱 .....            | 19 |
| 2.2.7 主机 .....            | 19 |
| 2.2.8 微处理器 .....          | 20 |
| 2.2.9 内存 .....            | 21 |
| 2.2.10 硬盘 .....           | 23 |
| 2.2.11 光驱 .....           | 25 |
| 2.2.12 微机总线 .....         | 25 |
| 2.3 PC 的软件子系统 .....       | 26 |
| 习题二 .....                 | 28 |
| 学习情境 3 微型计算机系统的编程结构 ..... | 29 |
| 3.1 微处理器的编程结构 .....       | 29 |



|                   |                            |    |
|-------------------|----------------------------|----|
| 3.1.1             | 8086 CPU 的编程结构 .....       | 29 |
| 3.1.2             | 8086 CPU 的内部寄存器结构 .....    | 30 |
| 3.2               | 8086 的内存编址和 I/O 编址 .....   | 34 |
| 3.2.1             | 8086 的存储器编址 .....          | 34 |
| 3.2.2             | 8086 的 I/O 编址 .....        | 35 |
| 3.3               | 8086 系统中的堆栈 .....          | 35 |
| 习题三 .....         |                            | 36 |
| 学习情境 4 汇编语言 ..... |                            | 37 |
| 4.1               | 汇编语言概述 .....               | 37 |
| 4.1.1             | 计算机指令 .....                | 37 |
| 4.1.2             | 汇编语言指令 .....               | 38 |
| 4.2               | DEBUG 调试程序的常用命令 .....      | 38 |
| 4.2.1             | DEBUG 操作命令 .....           | 38 |
| 4.2.2             | DEBUG 常用命令用法举例 .....       | 39 |
| 4.2.3             | 8086 指令的操作数的类型 .....       | 40 |
| 4.3               | 8086 CPU 指令中操作数的寻址方式 ..... | 40 |
| 4.3.1             | 立即数寻址 .....                | 40 |
| 4.3.2             | 直接寻址 .....                 | 41 |
| 4.3.3             | 寄存器寻址 .....                | 41 |
| 4.3.4             | 寄存器间接寻址 .....              | 42 |
| 4.3.5             | 寄存器相对寻址 .....              | 42 |
| 4.3.6             | 基址变址寻址 .....               | 42 |
| 4.3.7             | 相对基址变址寻址 .....             | 42 |
| 4.4               | 8086 CPU 指令系统 .....        | 43 |
| 4.4.1             | 数据传送类指令 .....              | 43 |
| 4.4.2             | 算术运算类指令 .....              | 47 |
| 4.4.3             | 逻辑运算与移位类指令 .....           | 50 |
| 4.4.4             | 串操作类指令 .....               | 53 |
| 4.4.5             | 控制转移类指令 .....              | 54 |
| 4.4.6             | 处理器控制指令 .....              | 56 |
| 4.5               | 汇编语言程序的处理软件 .....          | 57 |
| 4.5.1             | 汇编程序 .....                 | 57 |
| 4.5.2             | 连接程序 .....                 | 60 |
| 4.5.3             | DOS 环境执行可执行文件 .....        | 60 |
| 4.6               | 汇编语言基础 .....               | 61 |
| 4.6.1             | 汇编语言的语句格式 .....            | 61 |

|        |                              |     |
|--------|------------------------------|-----|
| 4.6.2  | 汇编语言的运算符 .....               | 62  |
| 4.6.3  | 表达式 .....                    | 63  |
| 4.6.4  | 汇编语言程序汇编步骤 .....             | 64  |
| 4.7    | 伪指令 .....                    | 64  |
| 4.7.1  | 数据定义伪指令 .....                | 64  |
| 4.7.2  | 符号定义伪指令 .....                | 64  |
| 4.7.3  | 段定义伪指令 SEGMENT 和 ENDS .....  | 65  |
| 4.7.4  | 设定段寄存器伪指令 ASSUME .....       | 65  |
| 4.7.5  | 定义过程的伪指令 PROC 和 ENDP .....   | 66  |
| 4.7.6  | 宏指令 .....                    | 67  |
| 4.7.7  | ORG 伪指令 .....                | 67  |
| 4.7.8  | 汇编结束伪指令 END .....            | 68  |
| 4.7.9  | PUBLIC 和 EXTRN 伪指令 .....     | 68  |
| 4.8    | 汇编语言程序设计 .....               | 68  |
| 4.8.1  | 最常用的 DOS 系统功能调用的子功能 .....    | 68  |
| 4.8.2  | 简单程序设计 .....                 | 70  |
| 4.8.3  | 用 DEBUG 检查可执行文件的功能错误 .....   | 73  |
| 4.8.4  | 分支程序设计 .....                 | 75  |
| 4.8.5  | 循环程序设计 .....                 | 79  |
| 4.8.6  | 子程序设计 .....                  | 87  |
| 习题四    | .....                        | 92  |
| 学习情境 5 | 8086/8088 CPU 的工作时序 .....    | 94  |
| 5.1    | 时序的概念 .....                  | 94  |
| 5.1.1  | 指令周期、总线周期及时钟周期 .....         | 94  |
| 5.1.2  | 8086/8088 微机系统的主要操作 .....    | 95  |
| 5.2    | 8086/8088 CPU 的工作模式 .....    | 95  |
| 5.2.1  | 8086/8088 CPU 的两种工作模式 .....  | 95  |
| 5.2.2  | 8086/8088 CPU 的引脚信号和功能 ..... | 96  |
| 5.3    | 8086 在最小模式下的典型配置 .....       | 101 |
| 5.4    | 8086 在最大模式下的典型配置 .....       | 103 |
| 5.5    | 典型的 8086/8088 CPU 时序分析 ..... | 105 |
| 5.5.1  | 引言 .....                     | 105 |
| 5.5.2  | 最小模式下的典型时序 .....             | 105 |
| 5.6    | 最大模式下的典型时序 .....             | 109 |
| 5.6.1  | 最大模式下的总线读周期 .....            | 109 |
| 5.6.2  | 最大模式下的总线写周期 .....            | 110 |



|                               |     |
|-------------------------------|-----|
| 5.6.3 最大模式下的 I/O 读/写周期 .....  | 110 |
| 习题五 .....                     | 111 |
| 学习情境 6 半导体存储器 .....           | 112 |
| 6.1 存储器概述 .....               | 112 |
| 6.1.1 存储器的性能 .....            | 112 |
| 6.1.2 存储器的分类 .....            | 112 |
| 6.2 半导体存储器的原理 .....           | 113 |
| 6.2.1 SRAM 的原理 .....          | 113 |
| 6.2.2 DRAM 的原理 .....          | 114 |
| 6.2.3 PROM 的原理 .....          | 115 |
| 6.3 主存储器与 CPU 的连接 .....       | 115 |
| 6.3.1 存储器与 CPU 连接因素 .....     | 115 |
| 6.3.2 存储器的扩展 .....            | 115 |
| 6.4 高速缓存 .....                | 118 |
| 6.4.1 高速缓存的工作原理 .....         | 118 |
| 6.4.2 高速缓存的基本结构 .....         | 119 |
| 6.4.3 高速缓存与 DRAM 存取的一致性 ..... | 120 |
| 6.5 存储器的应用 .....              | 121 |
| 习题六 .....                     | 122 |
| 学习情境 7 输入/输出和中断系统 .....       | 123 |
| 7.1 概述 .....                  | 123 |
| 7.1.1 输入和输出定义 .....           | 123 |
| 7.1.2 外设及其性能特点 .....          | 124 |
| 7.1.3 输入和输出的控制方式 .....        | 124 |
| 7.2 中断的概念及处理过程 .....          | 126 |
| 7.2.1 中断的概念 .....             | 126 |
| 7.2.2 中断请求信号的产生 .....         | 126 |
| 7.2.3 中断优先级 .....             | 127 |
| 7.2.4 中断响应 .....              | 128 |
| 7.2.5 中断服务子程序 .....           | 129 |
| 7.3 8086 中断系统 .....           | 129 |
| 7.3.1 中断分类及中断类型码 .....        | 129 |
| 7.3.2 中断向量和中断向量表 .....        | 131 |
| 7.3.3 中断响应过程与时序 .....         | 131 |
| 7.3.4 软件中断的特点 .....           | 133 |
| 7.4 Intel 8259A 中断控制器 .....   | 134 |

|                         |                        |     |
|-------------------------|------------------------|-----|
| 7.4.1                   | 8259A 的性能概述 .....      | 134 |
| 7.4.2                   | 8259A 的内部结构和工作原理 ..... | 134 |
| 7.4.3                   | 8259A 的外部引脚 .....      | 135 |
| 7.4.4                   | 8259A 的中断过程 .....      | 137 |
| 7.4.5                   | 8259A 的工作方式 .....      | 137 |
| 7.4.6                   | 系统总线的连接方式 .....        | 139 |
| 7.4.7                   | 引入中断请求的方式 .....        | 139 |
| 7.4.8                   | 8259A 的编程 .....        | 140 |
| 7.4.9                   | 8259A 的级联 .....        | 145 |
| 7.4.10                  | 8259A 的应用举例 .....      | 146 |
| 习题七 .....               |                        | 148 |
| 学习情境 8 可编程接口芯片及应用 ..... |                        | 149 |
| 8.1                     | 可编程接口概述 .....          | 149 |
| 8.2                     | 8255A 并行接口芯片 .....     | 150 |
| 8.2.1                   | 8255A 的基本性能 .....      | 150 |
| 8.2.2                   | 8255A 的内部结构 .....      | 151 |
| 8.2.3                   | 8255A 的引脚功能 .....      | 152 |
| 8.2.4                   | 8255A 的编程 .....        | 152 |
| 8.2.5                   | 8255A 的编程与应用 .....     | 156 |
| 8.3                     | 8251A 串行接口芯片 .....     | 160 |
| 8.3.1                   | 串行通信方式 .....           | 160 |
| 8.3.2                   | 8251A 的基本性能 .....      | 164 |
| 8.3.3                   | 8251A 的内部结构 .....      | 165 |
| 8.3.4                   | 8251A 的引脚功能 .....      | 167 |
| 8.3.5                   | 8251A 的控制字 .....       | 168 |
| 8.3.6                   | 8251A 的编程与应用 .....     | 171 |
| 8.4                     | 8253 定时/计数器芯片 .....    | 174 |
| 8.4.1                   | 8253 的基本性能 .....       | 174 |
| 8.4.2                   | 8253 的内部结构 .....       | 175 |
| 8.4.3                   | 8253 的引脚功能 .....       | 175 |
| 8.4.4                   | 8253 的工作方式 .....       | 176 |
| 8.4.5                   | 8253 的编程 .....         | 181 |
| 8.4.6                   | 8253 的应用举例 .....       | 183 |
| 习题八 .....               |                        | 186 |
| 参考文献 .....              |                        | 188 |

## 学习情境 1 微型计算机基础知识



### 教学提示

本章重、难点在于二进制数的概念、数制转换及编码,本章是学习和应用计算机的基础。



### 教学目的

了解计算机和微型计算机的基本概念;

了解计算机和微型计算机的发展;

掌握计算机中的数制和编码;

掌握计算机的性能指标。

## 1.1 微型计算机概述

### 1.1.1 微型计算机的发展

#### 1. 计算机的发展

电子计算机是由各种电子器件组成的能够自动、高速和精确地进行逻辑控制和信息处理的现代化设备。从 1946 年世界上第一台电子数字计算机 ENIAC 问世至今,计算机的发展主要经历了电子管、晶体管、小规模集成电路、大规模(LSI)和超大规模(VLSI)集成电路四个发展阶段。微型计算机是伴随着大规模集成电路的发展而诞生和发展的。微处理器是指一片或几片大规模集成电路组成的 CPU,CPU 是微处理器的计算机称为微型计算机,简称微机( $\mu$ C 或 MC),微机是计算机的第四代产品。

#### 2. 微型计算机的发展

微型计算机在系统的功能结构和基本工作原理上与其他计算机(巨型、大型、中小型计算机)没有本质差别,根本差别是它的中央处理器(CPU)发展为微处理器(MPU)。MPU 和 CPU 的关系为:MPU 一定是 CPU,但 CPU 不一定是 MPU,MPU 是一种特殊的 CPU。

现代微机都以微处理器为核心,配上大容量的半导体存储器及功能强大的可编程接口芯片,连上外部设备(包括键盘、显示器、打印机和磁盘、光盘等外部存储器)。

微型计算机的最小配置为单片机,是在一个芯片上封装了 CPU、存储器和接口电路。另一种较小的配置为单板机,是在一块印制电路板上安装了组成微型计算机要素的各个



部件。

微型计算机已从最初的 4 位微处理器芯片发展到 64 位微处理器芯片,微型计算机的发展是跟随微处理器的发展而发展的。

1971 年,Intel 公司推出了世界上第一只微处理器 4004。

1978 年,8086 处理器诞生,标志着 x86 时代的来临。8086 微处理器集成了 2.9 万只晶体管,时钟频率为 4.77 MHz。同期,Intel 还推出了 8087 协处理器,协处理器的作用是负责浮点运算。第一台 IBM PC 选用的是 8088,8088 其实是 8086 的一个简化版,其内部数据总线是 16 位,但外部却使用 8 位的数据总线,而 8086 内部数据总线(CPU 内部传输数据的总线)和外部数据总线(CPU 外部传输数据的总线)均为 16 位。8086/8088 的地址总线为 20 位,寻址范围可达 1 MB。

1982 年,Intel 发布了 80286 处理器,也就是俗称的 286。80286 芯片上集成了 14.3 万只晶体管,处理器 16 位字长,时钟频率由最初的 6MHz 逐步提高到 20MHz。其内部和外部数据总线皆为 16 位,地址总线 24 位。与 8086 相比,80286 寻址能力达到了 16 MB,其速度比 8086 提高了 5 倍甚至更多。IBM 公司将 80286 用在技术更为先进 IBM PC AT/机中,AT 机的外部总线为 16 位(IBM PC 为 8 位),内存容量可扩展到 16 MB,可使用更大的硬盘来存储数据,支持 VGA 彩色显示系统,在性能上有了重大的进步。

1985 年,Intel 再度推出了 80386 处理器。386 集成了 27.5 万只晶体管,超过了 4004 芯片的一百倍。386 还是 Intel 第一只 32 位微处理器,同时也是第一种具有多任务功能的处理器——这对微软操作系统的发展有着重要的影响。所谓多任务就是说处理器可以同时运行几个互不相关的程序。在 386 时代,Intel 在技术有了很大的进步,微处理器的时钟频率由最初的 12.5 MHz 提高到 20 MHz、25 MHz、33 MHz。80386DX 的内部和外部数据总线都是 32 位,地址总线也是 32 位,可寻址高达 4GB 内存。它除了具有实模式和保护模式两种工作模式外,还增加了一种称为虚拟 86 的工作模式,通过模拟多个 8086 处理器来提供多任务能力。

我们知道,CPU 执行的基本的操作不外乎是处理数据和通过外部数据总线与系统内存交换数据。在 386 时代,随着微处理器工作频率的不断提高,为了解决 CPU 工作速度与内存读写速度不匹配的矛盾,提出了高速缓存(Cache)的概念。

当 CPU 内部的工作速度比内存读写速度快时,会使 CPU 花费较长的时间等待数据到来或把数据写入内存。为了减少这种情况的发生,人们就想到了使用 Cache。Cache 是一种读写速度要比普通内存(系统内存)快很多的特殊内存器件。有 Cache 系统在工作时,将经常存取的一些数据从系统内存读取到 Cache 中,而 CPU 会首先到 Cache 中去读取或写入数据,如果 Cache 中没有所需数据,则再对系统内存进行读写。Cache 的实质就是在慢速系统内存和快速 CPU 之间插入一个速度较快、容量较小的高速内存,起到缓冲作用,这样既提高了系统的整体性能,又不会使系统的成本上升过高。

80386 时代开始出现外部 Cache。

1989 年, Intel 发布了 80486 处理器。486 处理器集成了 125 万个晶体管, 时钟频率由 25MHz 逐步提升到 33MHz、40MHz、50MHz 及后来的 100MHz。

在 486 时代, CPU 还出现了内部 Cache, 也称为 L1 Cache。实际上 486 就是集成有更高主频的 386 微处理器、80387 数字协处理器和 8KB 内部 Cache 的微处理器。

1993 年, Intel 发布了 Pentium(奔腾)处理器。本来按照以往的命名规律, 应该称为 80586, 但是 586 这样的数字是不能注册成专用的商标来使用的, 任何竞争对手都可以用 586 来命名自己的微处理器, 因此 Intel 使用了一个自造的新词作为新产品的商标—Pentium。

Pentium 处理器集成了 310 万个晶体管, 最初的工作频率是 60 MHz、66 MHz, 后来又提升到 200 MHz 以上。第一代的 Pentium 代号为 P54C, 其后又发布了代号为 P55C 的 Pentium, 它是内建 MMX(多媒体指令集)的 Pentium 处理器。Pentium MMX 处理器增加了 57 条 MMX 指令, 这些指令专门用来处理音、视频等数据的计算, 目的是提高 CPU 处理多媒体数据的能力。

1997 年 Intel 发布了 Pentium II 处理器, 其内部集成了 750 万个晶体管, 并整合了 MMX 指令, 可以更快更流畅地处理影音和图像等多媒体数据。

1999 年, Intel 公司推出了 Pentium III。P III 拥有 32K 一级缓存和 512K 二级缓存(L2 Cache 的工作频率是 CPU 核心频率的一半), 包含 MMX 指令和 Intel 自己的 3D 指令—SSE。最初发行的 P III 有 450 和 500 MHz 两种规格, 其外部总线频率为 100 MHz。除了 SSE 指令外, P III 与 P II 几乎是一样的, 当运行没有为 SSE 指令优化过的应用软件时, P III 与 P II 在速度上几乎一样。

P III 处理器在设计时便考虑了互联网的应用。P III 处理器内部包含了一个唯一的序列号, 这个 64 位的处理器序列号, 就相当于计算机的身份证, 用户既可以用它对计算机进行认证, 也可以在商务往来或是上因特网时用它进行数据加密, 以提高计算机应用的保密性。

同期, Intel 也推出了相对于 P III 的低价位的 Celeron II 处理器。但赛扬 II 不支持 Intel 最新的 SSE 指令集, 赛扬的 L2 Cache 只有 Pentium III 的四分之一, 但是其速度与处理器相同, 赛扬处理器比同级的 Pentium III 略差。赛扬和 Pentium / III 另一个差异就是外部总线速度: 赛扬的总线速度还是 66MHz。

2000 年 Intel 发布了 Pentium 4 处理器。Pentium 4 增加了 SSE2 指令集, 这套指令集有 144 个全新的指令, 用户使用基于 Pentium 4 处理器的个人计算机, 可以创建专业品质的影片, 可以透过因特网传递电视品质的影像, 实时进行语音、影像通信, 实时进行 3D 渲染, 快速进行 MP3 编码解码运算, 在连接因特网时可以运行多个多媒体软件。

### 1.1.2 微型计算机的特点

微型计算机除了具有一般计算机的运算速度快、计算精度高、记忆功能和逻辑判断力强、自动工作等常规特点外, 还有它自己的独特优点, 即还具有体积小、重量轻、功耗低、可靠性高、价值便宜、操作简便灵活等独特的优点。



冯·诺依曼体系结构为计算机的发展铺平了道路,但也是性能进一步提高的瓶颈。

### 1.1.3 微型计算机的应用领域

- (1) 科学计算(或数值计算);
- (2) 数据处理(或信息处理);
- (3) 辅助技术(或计算机辅助设计与制造);
- (4) 过程控制(或实时控制);
- (5) 人工智能(或智能模拟);
- (6) 网络应用。

### 1.1.4 微型计算机的发展趋势

- (1) 面向要求低成本的家电、传统工业改造及普及教育等的单片微型计算机;
- (2) 面向更加复杂的数据处理,OA 和 DA 科学计算等的微型计算机。

## 1.2 计算机中的数和编码系统

### 1.2.1 计算机中的进位计数制

#### 1. 进位计数制的基本概念

进位计数制是计数方法的统称。在日常生活中,时间采用 60 进制计数,月份采用 12 进制计数,习惯上最常用的是十进制计数法。一个任意的十进制数可以表示为:

$$a_n a_{n-1} \cdots a_0 \times b_1 b_2 \cdots b_m$$

其展开式是

$$a_n \times 10^n + a_{n-1} \times 10^{(n-1)} + \cdots + a_0 \times 10^0 + b_1 \times 10^{-1} + b_2 \times 10^{-2} + \cdots + b_m \times 10^{-m}$$

式中,  $a_i (i=0, 1, 2, \cdots, n)$ ,  $b_j (j=1, 2, \cdots, m)$  是 0~9 十个数码中的一个。

例 1.1 求 12345.67 的展开式为:

$$1 \times 10^4 + 2 \times 10^3 + 3 \times 10^2 + 4 \times 10^1 + 5 \times 10^0 + 6 \times 10^{-1} + 7 \times 10^{-2}$$

一般来说,对于基数为  $r$  的  $r$  进制数的值可以表示为

$$a_n r^n + a_{n-1} r^{n-1} + \cdots + a_0 r^0 + b_1 r^{-1} + b_2 r^{-2} + \cdots + b_m r^{-m}$$

在上面乘积和的形式中,乘积部分的幂部分称为该  $r$  进制数的相应位的权(简称位权),例如,  $r^0$  是  $a_0$  所在位的权,幂部分的底数称为该  $r$  进制数的基数,  $a_i (i=0, 1, 2, \cdots, n)$  和  $b_j (j=1, 2, \cdots, m)$  可以是  $0, 1, \cdots, r-1$  中的任意一个数字。

计算机中为便于存储和计算的物理实现,采用了二进制数。二进制数的基数为 2,每个二进制位只有 0、1 两个数码,遵循逢二进一的规则,每位的权是  $2^k (k=n, \cdots, 0, -1, \cdots, m)$ 。二进制数  $a_n a_{n-1} \cdots a_0 . b_1 b_2 \cdots b_m$  的展开式是

$$a_n 2^n + a_{n-1} 2^{n-1} + \cdots + a_0 2^0 + b_1 2^{-1} + b_2 2^{-2} + \cdots + b_m 2^{-m}$$



式中,  $a_i (i=0, 1, 2, \dots, n)$  和  $b_j (j=1, 2, \dots, m)$  可分别为 0、1 两个数码中一个。例如,  
 $(101101)_2 = 1 \times 2^5 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^0 = (45)_{10}$

即二进制数 101101 与十进制数 45 等值, 其中括号外的下标表示数的基数。

为了书写便利, 还经常采用八进制数和十六进制数来表示二进制数值。

八进制数的基数为 8, 每位的权是  $8^k (k=n, \dots, 0, -1, \dots, -m)$ , 每一位上的数值可取 0~7 中的任意一个数码。

十六进制数的基数为 16, 每位的权是  $16^k (k=n, \dots, 0, -1, \dots, -m)$ , 每一位上的数值可取 0~9、A、B、C、D、E、F 中的任意一个数码。

例如, 八进制数  $(17)_8 = (15)_{10}$ ; 十六进制数  $(2E)_{16} = (46)_{10}$ 。

通常可用数字后面跟一个英文字母来表示该数的数制。十进制数一般用 D (Decimal)、二进制数用 B(Binary)、八进制数用 O(Octal)、十六进制数用 H(Hexadecimal) 来表示。例如, 234D、456O、101011B、0FEH 等。当然, 也可以用这些字母的小写形式。若十六进制数的最高位数为字母 A~F 之一时, 则应加上前导零。

## 2. 进位计数制之间数的转换

十进制数、二进制数、八进制数和十六进制数的对应关系如表 1.1 所示。

表 1.1 各种进制数码对照表

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

(1) 二进制、八进制、十六进制数转换为十进制数 对于任何一个二进制数、八进制数、十六进制数, 可以写出它的按权展开式, 再进行计算即可。

例 1.2  $10101101.101B = 2^7 + 2^5 + 2^3 + 2^2 + 2^0 + 2^{-1} + 2^{-3}$   
 $= 128 + 32 + 8 + 4 + 1 + 0.5 + 0.125$   
 $= 173.625D$

(2) 十进制数转换为二进制数 对于整数部分采用除 2 取余法, 即逐次除以 2, 直至商为 0 为止, 把每次相除得出的余数按倒序排列, 即为该整数部分的二进制数; 对于小数部分采用乘 2 取整法, 即逐次乘以 2, 直至积的小数部分为 0 为止, 从每次相乘得出的余数按顺序排列, 即为该小数部分的二进制数。

①整数的转换方法:除 2 取余,逆序排列。

例 1.3 将十进制数 28 转换为二进制数。见图 1.1。

|   |    |       |    |           |
|---|----|-------|----|-----------|
| 2 | 28 |       | 余数 | ↑ 二进制数的低位 |
| 2 | 14 | ..... | 0  |           |
| 2 | 7  | ..... | 0  |           |
| 2 | 3  | ..... | 1  |           |
| 2 | 1  | ..... | 1  |           |
|   | 0  | ..... | 1  | ↑ 二进制数的高位 |

图 1.1 十进制整数转换成二进制

即  $(28)_{\text{D}} = (11100)_{\text{B}}$ 。

②小数的转换方法:乘 2 取整,顺序排列。

例 1.4 将十进制 0.125 转换为二进制数。见图 1.2。

|         |    |        |
|---------|----|--------|
|         |    | 0.125  |
| 小数点     | 整数 | ×      |
|         |    | 2      |
|         |    | -----  |
|         |    | 0.250  |
|         |    | ×      |
|         |    | 2      |
|         |    | -----  |
|         |    | 0.500  |
|         |    | ×      |
|         |    | 2      |
|         |    | -----  |
|         |    | 1.000  |
| 二进制小数末位 |    | 为0转换结束 |

图 1.2 十进制小数转换成二进制

即  $(0.125)_{\text{D}} = (0.001)_{\text{B}}$ 。

(3)二进制数与八进制数的相互转换 二进制数转换成八进制数的方法是:将二进制数从小数点开始,对二进制整数部分向左每 3 位分成一组,对二进制小数部分向右每 3 位分成一组,不足三位的分别向高位或低位补 0 凑成三位。每一组有 3 位二进制数,分别转换成八进制数码中的一个数字,全部连接起来即可。

二进制数换算成八进制数的方法是“3 位变 1 位”:将每 3 位二进制数用 1 位对应的八进制数表示。

八进制数换算成二进制数的方法是“1 位变 3 位”:将每 1 位八进制数用 3 位对应的二进制数表示。

(4)二进制数与十六进制数的相互转换 二进制数转换成十六进制数,4 位分成一组,再分别转换成十六进制数码中的一个数字,不足四位的分别向高位或低位补 0 凑成四位,全部连接起来即可。反之,十六进制数转换成二进制数,只要将每一位十六进制数转换成 4 位二进制数,依次连接起来即可。

二进制数换算成十六进制数的方法是“4 位变 1 位”:将每 4 位的二进制数用 1 位对应的十六进制数表示。

十六进制数换算成二进制数的方法是“1 位变 4 位”:将每 1 位十六进制数用 4 位对应的二进制数表示。

## 1.2.2 计算机中对带符号数的表示

一个数在计算机中的表示形式称为这个数的机器数,这个数称为这机器数的真值。下面列举几个常用的带符号数的机器数。

### 1. 原码

原码的最高位为符号位,其余各位表示真值的绝对值。符号位为 0 表示该真数为正数,符号位为 1 表示该真数为负数。

### 2. 反码

正数的反码与原码相同,负数的反码其符号为仍用 1 表示,即把一个真值的原码的符号位不变,其余各位分别取反就得到这个真值的反码。

### 3. 补码

用于解决减法转换为加法的问题,简化计算机运算的电路,常用的机器数是补码。引入补码后,计算机中就只有加法计算。

补码的定义包括下列三条:

- (1) 一个正数的补码与原码相同;
- (2) 一个负数的补码的求法是把原码的符号位不变,其余各位分别取反后再加 1 的和;
- (3) 正零和负零的原码不同,但正零和负零的原码相同,只有一个形式,即和正零的原码项同。

补码运算定律:

$$(x + y)_{\text{补}} = (x)_{\text{补}} + (y)_{\text{补}}$$

$$(x - y)_{\text{补}} = (x)_{\text{补}} + (-y)_{\text{补}}$$

例 1.5 验证正数  $x$  和负数  $y$  的补码相加,遵从补码的运算定律,设  $x=3\text{DH}$ ,  $y=-21\text{H}$ 。

$$(x)_{\text{补}} + (y)_{\text{补}} = (00111101)_{\text{补}} + (10100001)_{\text{补}}$$

$$= 00111101 + 11011111$$

$$= 00011100$$

$$(x + y)_{\text{补}} = (00111101 - 00100001)_{\text{补}}$$

$$= (00011100)_{\text{补}}$$

$$= 00011100$$

所以  $(x + y)_{\text{补}} = (x)_{\text{补}} + (y)_{\text{补}}$

## 1.2.3 数据和字符编码

字符编码就是规定用什么样的二进制代码来表示数值、数字、字母以及专用符号。

### 1. 十进制数的编码(BCD 码)

二进制具有很多优点,所以计算机内部采用二进制数进行运算。采用若干位二进制数码表示一位十进制数的编码方案,称为二进制编码的十进制数,简称二—十进制编码,