



普通高等教育“十一五”国家级规划教材

微型计算机原理与接口技术

WEIXING JISUANJI YUANLI YU JIEKOU JISHU

第5版

◎ 周荷琴 冯焕清 / 编著

技术大学出版社

013039487

TP36
535-5

• 普通高等教育“十一五”国家级规划教材
中国科学院指定考研参考书

微型计算机原理与接口技术

• 第5版 •

周荷琴 冯焕清 编著



TP36
535-5

中国科学技术大学出版社

• 合 肥 •



784800010

内 容 简 介

本书是为中国科学技术大学工科电子类专业本科生学习“微型计算机原理与系统”课程编写的教材,是作者在参考了国内外大量文献、资料的基础上,吸取各家之长,并结合多年教学和应用研究的经验,精心组织编写而成的。全书内容丰富,图文并茂,讲述深入浅出,通俗易懂,并附有大量的实例和习题,部分习题还给出了解题提示,既可用作教材,也适合于自学,被列入“普通高等教育‘十一五’国家级规划教材”和“中国科学院指定考研参考书”。

全书 15 章,内容安排上注重系统性、先进性和实用性。前 5 章是基础部分,主要介绍 8086 微型机系统的组成原理、体系结构、指令系统、汇编语言程序设计方法以及存储器的原理和电路设计。第 6~12 章讨论接口和总线技术,包括中断、DMA 和 I/O 接口以及几个典型的大规模集成电路接口芯片(8255A、8253/8254、8259A、8251A、8237A),A/D 和 D/A 以及总线技术也被纳入其中。最后 3 章介绍高档微型机的工作原理,其中第 13 章包括 32 位微处理器的结构和工作模式、寄存器组成、保护模式下的内存管理、保护模式下的中断和异常以及任务切换等内容,第 14 章介绍 32 位机新增指令、浮点数、SIMD 技术和指令系统,并给出了许多编程实例,第 15 章简要介绍了 PC/XT 机的系统结构,主要对 32 位微型机的结构以及 64 位 CPU 和智能多核处理器进行了讨论,并概要阐述了 64 位机的系统结构和技术特点。

本书可作为高等学校电子类专业和其他相近相关专业本科教育的教材,也可作为从事微型计算机系统设计和应用等相关科技工作者的参考书。

图书在版编目(CIP)数据

微型计算机原理与接口技术/周荷琴,冯焕清编著.—5 版.—合肥:中国科学技术大学出版社,2013.3

普通高等教育“十一五”国家级规划教材

ISBN 978-7-312-03191-5

I. 微… II. ①周… ②冯… III. ①微型计算机—理论—高等学校—教材 ②微型计算机—接口技术—高等学校—教材 IV. TP36

中国版本图书馆 CIP 数据核字(2013)第 030916 号

责任编辑:张善金

出版者:中国科学技术大学出版社

地 址:安徽省合肥市金寨路 96 号 邮编:230026

网 址:<http://press.ustc.edu.cn>

电 话:发行部 0551-63606086-8810

印刷者:合肥学苑印务有限公司

发 行 者:中国科学技术大学出版社

经 销 者:全国新华书店

开 本:787mm×1092mm 1/16

印 张:33

字 数:860 千

版 次:1996 年 9 月第 1 版 2013 年 3 月第 5 版

印 次:2013 年 3 月第 30 次印刷

印 数:277 001—288 000 册

定 价:42.00 元

前 言

本书是为中国科学技术大学工科电子类本科生学习“微型计算机原理与系统”课程编写的教材,被列入“普通高等教育‘十一五’国家级规划教材”和“中国科学院指定考研参考书”。自1996年出版以来,本书被很多高校选用,得到了广大读者和同行老师的热情支持,并反馈回不少宝贵的意见和建议,在此谨表示感谢!为适应微型计算机技术飞速发展的形势和教育教学改革不断深化的需要,我们再次对原书进行了全面的修订。

自20世纪70年代第一代微型计算机问世以来,计算机技术以惊人的速度发展,涌现了数十个品种几百个型号的微处理器,数据宽度从8位、16位、32位发展到了64位,处理器芯片的CPU核心发展到了双核乃至4核、6核和8核,当前微型计算机的发展已经进入了智能多核时代。希望通过这次修订,本书能更系统归纳和清晰展示已经发展了40多年的计算机高新技术,能更深入浅出地讲清楚那些看似深奥的计算机知识,从而真正有助于教师们的课堂教学和学生们的课后阅读。通过努力我们顺利完成了修订工作,新版将以全新的面貌与读者见面。

全书共分15章,内容包括3部分:第1~5章是基础部分,仍以8086为主要对象,包括绪论、8086 CPU、寻址方式、指令系统、汇编语言程序设计和存储器。第6~12章讨论了接口和总线技术,包括中断、DMA和I/O接口以及8255A、8253/8254、8259A、8251A、8237A等典型的可编程接口芯片,A/D和D/A以及总线技术也被纳入其中。第13~15章介绍了高档微型机的原理,包括32位微型机的工作原理、指令系统与程序设计以及64位CPU和智能多核处理器,也包括16位、32位和64位机的系统结构和技术特点。

本次修订的主要内容包括:

1. 对第1、2、4、5、8、12、15章的全部内容以及第6、13、14章的部分内容重新进行了组织和编写,尽可能删繁就简,循序渐进,将现今最新的技术包含进来,表述上没有冗余,文字简练顺畅,全书一气呵成。

2. 存储器一章不仅包含了最新的存储器技术,还对目前在嵌入式系统领域广泛应用的串行EEPROM作了较详细的介绍;考虑到Cache技术在现代CPU技术中的地位越来越重要,对其工作原理的讨论也比较深入。

3. 将 I/O 接口技术与并行接口芯片 8255 合并形成了第 6 章,并将几个简单接口芯片的介绍前移到第 2 章。

4. 总线技术一章对已淘汰的总线标准只做了简单回顾,重点讨论了当前流行的系统总线 PCI、PCI Express 以及串行总线 USB 和 IEEE 1394。

5. 第 13 章 32 位微型计算机原理,对保护模式下的中断和异常进行了较详细的描述,并增加了任务切换的内容。

6. 第 14 章 32 位微型计算机的指令系统与程序设计,在原有的 32 位微型计算机新增指令和程序设计的基础上,增加了 IEEE 标准 754 浮点数的表示方法、奔腾处理器的 SIMD 技术、SIMD 指令系统、SIMD 程序设计实例等内容。

7. 第 15 章在原有 PC/XT 机以及 32 位微型计算机的系统结构基础上,增加了 64 位 CPU 和多核处理器的内容,并着重介绍了 64 位微型机的系统结构、芯片组和主板,内容涉及最新的 i7、i5 和 i3 智能多核处理器。

8. 对第 4 章和第 14 章中的 16 位和 32 位微型计算机的所有汇编语言程序实例都进行了精心设计,并全部通过了上机验证。

9. 对原书保留章节中的绝大部分插图进行了修正。

10. 修改了原书保留章节中的部分内容和习题,并为许多习题增加了解题的提示信息,读者根据例题和提示信息,不难得出正确答案。

吴秀清教授为本书此前的版本做出了重要贡献,特此感谢!

在本书撰写过程中,青年教师李峰、何力以及研究生潘剑锋、刘冰啸、乔赫元、袁非牛、王鹏、刘勃、刘学亮、王恒良、郭永刚、赵何、黄庆华、卢胜、陈立群、梅涛、武海澄、陈功等在资料的收集、例题的验证、程序的调试、插图的绘制、多媒体课件的制作等方面做了许多工作,并对书中的内容提出了不少有益的建议,在此一并表示衷心的感谢!

此外,在本书编写过程中,我们查阅、参考了大量国内外相关的书籍和文献以及网上资料,在此特向这些文献的作者表示深切的感谢!

由于作者水平有限,错误和不当之处在所难免,敬请读者批评指正,以便日后再版时予以修正。

编 者

2013 年 1 月于合肥

目 录

前言	(i)
第 1 章 绪论	(1)
1.1 计算机中数的表示方法	(1)
1.1.1 进位计数制	(1)
1.1.2 二进制编码	(3)
1.1.3 带符号数的表示方法	(4)
1.2 计算机的基本结构	(6)
1.2.1 计算机的基本结构	(6)
1.2.2 计算机软件	(8)
1.3 微型计算机结构和系统	(11)
1.3.1 微型计算机基本结构	(11)
1.3.2 微型计算机系统	(16)
1.4 微型计算机的发展概况	(17)
1.4.1 计算机的发展	(17)
1.4.2 微型计算机的发展	(18)
第 2 章 8086 CPU	(22)
2.1 8086 CPU 的内部结构	(22)
2.1.1 8086 CPU 内部结构及工作过程	(22)
2.1.2 8086 CPU 内部寄存器	(23)
2.2 8086/8088 CPU 的引脚功能	(27)
2.3 8086 的存储器组织	(31)
2.3.1 段地址和偏移地址	(31)
2.3.2 8086 存储器的分体结构	(34)
2.4 8086 的工作模式和总线操作	(36)
2.4.1 最小模式系统	(36)
2.4.2 最大模式系统	(41)
2.4.3 总线操作时序	(43)
第 3 章 8086 的寻址方式和指令系统	(48)
3.1 8086 的寻址方式	(48)
3.1.1 立即寻址方式	(48)
3.1.2 寄存器寻址方式	(49)
3.1.3 直接寻址方式	(49)
3.1.4 寄存器间接寻址方式	(51)

3.1.5	寄存器相对寻址方式	(52)
3.1.6	基址变址寻址方式	(53)
3.1.7	相对基址变址寻址方式	(53)
3.1.8	其它寻址方式	(55)
3.2	指令的机器码表示方法	(56)
3.2.1	机器语言指令的编码目的和特点	(56)
3.2.2	机器语言指令代码的编制	(57)
3.3	8086 的指令系统	(61)
3.3.1	数据传送指令	(61)
3.3.2	算术运算指令	(69)
3.3.3	逻辑运算和移位指令	(82)
3.3.4	字符串处理指令	(87)
3.3.5	控制转移指令	(92)
3.3.6	处理器控制指令	(107)
第 4 章	汇编语言程序设计	(113)
4.1	汇编语言程序格式和伪指令	(114)
4.1.1	汇编语言程序格式	(114)
4.1.2	伪指令语句	(119)
4.1.3	完整的汇编语言程序框架	(124)
4.2	DOS 系统功能调用和 BIOS 中断调用	(128)
4.2.1	概述	(129)
4.2.2	DOS 系统功能调用	(129)
4.2.3	BIOS 中断调用	(134)
4.3	汇编语言程序设计方法与实例	(138)
4.3.1	顺序结构程序设计	(138)
4.3.2	分支程序设计	(140)
4.3.3	循环结构程序	(143)
4.3.4	代码转换程序	(146)
4.3.5	过程调用	(149)
第 5 章	存储器	(155)
5.1	存储器分类	(155)
5.1.1	内部存储器	(155)
5.1.2	外部存储器	(157)
5.1.3	存储器的性能指标	(160)
5.2	随机存取存储器 RAM	(160)
5.2.1	静态 RAM(SRAM)	(161)
5.2.2	动态 RAM(DRAM)	(162)
5.2.3	内存条	(166)
5.3	只读存储器 ROM	(169)
5.3.1	可编程可擦除 ROM(EPROM)	(169)

5.3.2 电可擦除可编程 ROM(EEPROM)	(172)
5.4 存储器与 CPU 的连接	(176)
5.4.1 设计接口应考虑的问题	(176)
5.4.2 存储器接口设计	(177)
5.5 高速缓冲存储器	(184)
5.5.1 高速缓存的原理	(184)
5.5.2 高速缓存的基本结构	(186)
5.5.3 主存与 Cache 的地址映射	(187)
5.5.4 Cache 的基本操作	(191)
5.5.5 影响 Cache 性能的因素	(193)
第 6 章 I/O 接口和并行接口芯片 8255A	(195)
6.1 I/O 接口	(195)
6.1.1 I/O 接口的功能	(195)
6.1.2 I/O 端口及其寻址方式	(197)
6.1.3 CPU 与外设间的数据传送方式	(199)
6.1.4 PC 机的 I/O 地址分配	(205)
6.2 8255A 的工作原理	(208)
6.2.1 8255A 的结构和功能	(208)
6.2.2 8255A 的控制字	(210)
6.2.3 8255A 的工作方式和 C 口状态字	(212)
6.3 8255A 的应用举例	(220)
6.3.1 基本输入输出应用举例	(220)
6.3.2 键盘接口	(223)
6.3.3 8255A 在 PC/XT 机中的应用	(227)
第 7 章 可编程计数器/定时器 8253/8254 及其应用	(232)
7.1 8253 的工作原理	(233)
7.1.1 8253 的内部结构和引脚信号	(233)
7.1.2 初始化编程步骤和门控信号的功能	(236)
7.1.3 8253 的工作方式	(238)
7.2 8253/8254 的应用举例	(242)
7.2.1 8253 定时功能的应用举例	(242)
7.2.2 8253/8254 计数功能的应用举例	(245)
7.2.3 8253 在 PC/XT 机中的应用	(249)
第 8 章 中断和可编程中断控制器 8259A	(254)
8.1 中断	(254)
8.1.1 中断概念和分类	(254)
8.1.2 中断的响应与处理过程	(258)
8.2 8259A 的工作原理	(262)
8.2.1 8259A 的引脚信号和内部结构	(262)
8.2.2 8259A 的工作方式	(264)

8.2.3	8259A 的命令字及编程	(266)
8.3	8259A 应用举例	(273)
8.3.1	8259A 的级联使用	(273)
8.3.2	中断向量的设置和中断处理程序设计实例	(276)
第 9 章	串行通信和可编程接口芯片 8251A	(283)
9.1	串行通信的基本概念和 EIA RS-232C 串行口	(283)
9.1.1	串行通信的基本概念	(283)
9.1.2	EIA RS-232C 串行口	(287)
9.2	可编程串行通信接口芯片 8251A	(289)
9.2.1	8251A 的内部结构和外部引脚	(290)
9.2.2	8251A 的编程	(295)
9.2.3	8251A 应用举例	(301)
第 10 章	模数(A/D)和数模(D/A)转换	(306)
10.1	概述	(306)
10.1.1	一个实时控制系统	(306)
10.1.2	采样、量化和编码	(307)
10.1.3	采样保持器	(309)
10.2	D/A 转换器	(311)
10.2.1	数/模转换器原理	(311)
10.2.2	数/模转换器的主要性能指标	(312)
10.2.3	数/模转换器 AD7524、DAC0832 和 DAC1210	(314)
10.3	A/D 转换	(321)
10.3.1	模/数转换器原理	(321)
10.3.2	模/数转换器 ADC0809 和 AD574A	(323)
第 11 章	DMA 控制器 8237A	(336)
11.1	8237A 的组成和工作原理	(337)
11.1.1	8237A 的内部结构	(337)
11.1.2	8237A 的引脚功能	(338)
11.1.3	8237A 的内部寄存器	(340)
11.2	8237A 的时序	(348)
11.2.1	外设和内存间的 DMA 数据传送时序	(348)
11.2.2	空闲周期、有效周期和扩展写周期	(349)
11.3	8237A 的编程和应用举例	(350)
11.3.1	PC/XT 机中的 DMA 控制逻辑	(350)
11.3.2	8237A 的一般编程方法	(352)
11.3.3	PC/XT 机上的 DMA 控制器的使用	(354)
第 12 章	总线技术	(356)
12.1	总线概述	(356)
12.1.1	总线的分类	(356)
12.1.2	总线的主要性能指标	(358)

12.1.3	总线标准	(358)
12.1.4	PC 系列总线	(359)
12.1.5	测控机箱底板总线	(360)
12.1.6	仪器与计算机互连总线	(361)
12.2	PCI 总线	(362)
12.2.1	PCI 局部总线	(362)
12.2.2	PCI 总线的特点	(364)
12.2.3	基于 PCI 总线的计算机系统	(364)
12.2.4	PCI 总线信号	(365)
12.2.5	PCI 总线的应用	(366)
12.3	PCI Express 总线	(367)
12.3.1	PCI-E1.0	(367)
12.3.2	PCI-E2.0	(368)
12.3.3	PCI-E3.0	(369)
12.3.4	PCI-E 的未来	(370)
12.4	USB 总线	(371)
12.4.1	USB 的特点	(371)
12.4.2	USB 规范	(371)
12.4.3	USB 接口规范	(372)
12.4.4	USB 的数据编码	(375)
12.4.5	USB 的传输方式	(376)
12.4.6	USB 包	(376)
12.4.7	USB 设备的枚举	(379)
12.5	IEEE 1394 总线	(381)
12.5.1	IEEE 1394 总线	(381)
12.5.2	IEEE 1394 总线的特点	(382)
12.5.3	IEEE 1394 规范的主要内容	(383)
第 13 章	32 位微型机的基本工作原理	(388)
13.1	32 位微处理器的结构与工作模式	(389)
13.1.1	32 位微处理器结构简介	(389)
13.1.2	32 位微处理器的工作模式	(392)
13.2	寄存器	(395)
13.2.1	用户级寄存器	(395)
13.2.2	系统级寄存器	(398)
13.2.3	程序调试寄存器	(404)
13.3	保护模式下的内存管理	(405)
13.3.1	段内存管理技术	(405)
13.3.2	分页内存管理技术	(414)
13.4	保护模式下的中断和异常	(417)
13.4.1	中断和异常	(417)

13.4.2	保护模式下中断和异常的处理	(425)
13.5	任务切换	(429)
13.5.1	任务结构和任务切换数据结构	(429)
13.5.2	任务切换方式	(433)
13.5.3	任务调用、链接和切换过程	(435)
第 14 章	32 位机的指令系统和程序设计	(440)
14.1	80386 新增指令和程序设计	(440)
14.1.1	80386 的寻址方式	(440)
14.1.2	80386 的新增指令	(442)
14.1.3	程序设计实例	(448)
14.2	浮点数的表示方法和奔腾处理器的 SIMD 技术	(455)
14.2.1	浮点数的表示方法	(455)
14.2.2	奔腾处理器的 SIMD 技术	(459)
14.3	SIMD 指令系统	(462)
14.3.1	数据传送指令	(463)
14.3.2	算术运算指令	(470)
14.3.3	逻辑运算指令	(474)
14.3.4	移位指令	(474)
14.3.5	比较指令	(475)
14.3.6	数据转换指令	(477)
14.4	利用 SIMD 指令进行程序设计	(478)
第 15 章	微型计算机系统结构	(486)
15.1	PC/XT 机的系统板	(486)
15.1.1	CPU 子系统	(486)
15.1.2	接口部件子系统	(488)
15.1.3	存储器子系统	(489)
15.2	32 位微型机的典型结构	(491)
15.2.1	主板的组成	(491)
15.2.2	Pentium II 主板	(493)
15.2.3	集成型主板	(495)
15.3	64 位微型机	(499)
15.3.1	64 位处理器	(499)
15.3.2	64 位操作系统	(501)
15.3.3	915 系列芯片组与主板	(501)
15.4	多核处理器技术	(504)
15.4.1	双核处理器的诞生	(504)
15.4.2	Intel 智能酷睿多核处理器	(505)
附录 A	8086/8088 指令系统一览表	(509)
附录 B	ASCII 码编码表	(513)
附录 C	汇编语言上机过程	(514)
参考文献	(518)

第 1 章 绪 论

本章首先介绍计算机中数的表示方法,然后就计算机的基本结构、大致的工作过程和计算机软件等内容展开讨论,接着阐述微型计算机的结构和微型计算机系统,使大家对 CPU、存储器、接口等概念有一个初步的了解,最后简要讲述计算机和微型计算机的发展概况。

1.1 计算机中数的表示方法

1.1.1 进位计数制

进位计数制是指用一组固定的数字符号和特定的规则来表示数的方法。在日常生活中,数多用 10 进制来表示,这是大家所熟悉的。有时也用别的进制来表示数,例如,表示时间的时、分、秒之间是 60 进制,而小时与天之间为 24 进制。在微型计算机中,则采用只有 0 和 1 两个数字的二进制来表示数。这是因为计算机是一种电子设备,由大量只能识别电信号的逻辑部件组成,可以用电平的高低、开关的通断、晶体管的导通和截止来表示数字 0 和 1,运算规则简单,使用方便可靠。人们经常使用的字母、符号和图形等,在计算机中也一律用二进制编码来表示。但二进制数的数位太长,不易书写和记忆,而人们又习惯于使用 10 进制数,因此在计算机领域里使用多种进位制来表示数,常用的有二进制、10 进制和 16 进制的表示方法。

1. 10 进制数 (Decimal)

10 进制数具有 10 个不同的数字符号 0~9,其基数为 10,各位的权值为 10^i ,其实际值可按权展开后相加获得。在 10 进制数字后面可以加后缀 D,表示该数是 10 进制数,但 D 通常省略不写。例如,10 进制数

$$347D = 347 = 3 \times 10^2 + 4 \times 10^1 + 7 \times 10^0$$

2. 二进制数 (Binary)

它只有 0 和 1 两个数字,其基数为 2,各位的权值为 2^i 。表示二进制数时,后面必须加后缀 B。例如,二进制数

$$10110B = 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 22$$

3. 16 进制数 (Hexadecimal)

它由 0~9、A、B、C、D、E、F 共 16 个数字和字母组成,其基数为 16,各位的权值为 16^i 。表示 16 进制数时,后面必须加后缀 H。字母 A~F 分别表示 10 进制数的 10~15。例如,16 进制数

$$3A0FH = 3 \times 16^3 + 10 \times 16^2 + 0 \times 16^1 + 15 \times 16^0 = 3 \times 4096 + 10 \times 256 + 15 = 14863$$

每个 16 进制数字都可用 4 位二进制数来表示,见表 1.1,如 0AH = 1010B,0FH =

1111B。由于 16 进制数与二进制数之间转换很方便,数位长度又只有二进制数的 1/4,因此,在编写汇编语言程序和打印程序清单时,广泛使用 16 进制数。在计算机中表示存储器的地址和存放的数据时,通常也都用 16 进制数来表示。

表 1.1 10 进制、二进制、16 进制数、BCD 码的关系

10 进制数	二进制数	16 进制数	BCD 码
0	0000	0	0000
1	0001	1	0001
2	0010	2	0010
3	0011	3	0011
4	0100	4	0100
5	0101	5	0101
6	0110	6	0110
7	0111	7	0111
8	1000	8	1000
9	1001	9	1001
10	1010	A	0001 0000
11	1011	B	0001 0001
12	1100	C	0001 0010
13	1101	D	0001 0011
14	1110	E	0001 0100
15	1111	F	0001 0101

4. 8 进制数(Octal)

8 进制数由 0~7 共 8 个数字组成,其基数为 8,各位的权值为 8^i 。表示 8 进制数时,后面必须加后缀 O 或 Q。例如,8 进制数

$$753Q = 7 \times 8^2 + 5 \times 8^1 + 3 \times 8^0 = 491$$

每位 8 进制数可以由 3 位二进制数组成,例如,627Q=110 010 111B,它与二进制数之间的转换十分方便。在 20 世纪 70~80 年代使用的小型计算机系统上,普遍采用 8 进制数编写汇编语言程序和打印程序清单。随着微型计算机技术的普及,16 进制计数法被用户广泛采用,微机中已不再采用 8 进制计数法了。

5. 不同进制数之间的转换

由上面的例子可以看到,将二进制或 16 进制数转换成 10 进制数时,只要将当前位数值乘以该位的权值,再将各部分相加即可。如果要将 10 进制数转换成二进制数就有些麻烦,可以采用除以 2 求余数的方法来得到。

例 1.1 将 10 进制数 25 转换成二进制数,先用 2 去除,得到的商为 12,第一个余数 $K_0 = 1$,此位为结果的最低位。再将 12 除以 2,得到第二个余数 $K_1 = 0$ ……如此不断进行下去,直到商等于 0 为止,最后一次得到的余数为结果的最高位。这样,可以得到 $25 = 11001B$ 。

2	25	
2	12	余数=1(K ₀) 最低位
2	6	余数=0(K ₁)
2	3	余数=0(K ₂)
2	1	余数=1(K ₃)
	0	余数=1(K ₄) 最高位

根据表 1.1, 将二进制数转换成 16 进制数就很容易。例如, $1000\ 1010\text{B}=8\text{AH}$ 。这里要说明的是由于二进制数的数位较长, 不便于阅读, 所以本书在后面书写二进制数时, 从最低位起, 每隔 4 个二进制数, 前面加一个空格。但当二进制数在程序中出现时, 这个空格必须除去, 否则送到计算机中进行运算时就不正确了。

例如, 把二进制数 1011010011011001B 写成 $1011\ 0100\ 1101\ 1001\text{B}$, 它等于 B4D9H 。

6. 位、字节、字和字长等数据单位表示

位(Bit): 在计算机中, 二进制数的每一位(0 或 1)是组成二进制信息的最小单位, 称为 1 个比特, 简称位, 它是数字系统和计算机中信息存储、处理和传送的最小单位。

字节(Byte): 8 个二进制信息组成的一个单位称为 1 个字节, $1\ \text{Byte}=8\ \text{Bit}$ 。

字(Word): 一个字由 16 位二进制数也即两个字节组成。一个 16 位的字 $D_{15}\sim D_0$ 可分为高字节和低字节两个部分, 其中 $D_{15}\sim D_8$ 为高字节, $D_7\sim D_0$ 为低字节。

字长(Word Length): 计算机中还用字长这个术语来表示数据, 字长决定了计算机内部一次可以处理的二进制代码的位数, 它取决于计算机内部的运算器、通用寄存器和数据总线的位数。根据计算机的字长不同, 可将计算机分为 8 位机、16 位机、32 位机和 64 位机等不同的机型。显然, 计算机的字长越长, 一次能同时传送和处理的数据就越多, 运算速度越快, 精度也越高, 但制造工艺就越复杂。

1.1.2 二进制编码

在计算机中, 数都采用二进制的形式来表示, 因此各种数字、英文字母、运算符号等, 都要采用若干特定的二进制码的组合来表示, 这就是二进制编码。最常用的编码有 BCD 码和 ASCII 码两种。

1. 8421 BCD 码(Binary Coded Decimal)

虽然计算机中的数都采用二进制表示, 但是二进制数不仅数位长, 而且不直观, 所以常采用 BCD 码来表示。BCD 码有 0, 1, 2, ..., 9 共 10 个不同的数字符号, 也是逢 10 进 1, 但它的每一位数字都用 4 位二进制来表示, 所以称为二进制码的 BCD 数, 它是一种很直观的编码。4 位二进制数可表示 $0000\sim 1111$ 共 16 种码, 取前 10 个码作为 BCD 码。由于它 4 位的权值分别是 8、4、2、1, 因此也称为 8421 BCD 码。

例 1.2 用 8421 BCD 码表示 10 进制数 327。

$$(327)_{10} = (0011\ 0010\ 0111)_{\text{BCD码}}$$

反之, 由 BCD 码也很容易求得它表示的 10 进制数。

例 1.3 $(1001\ 0101\ 1000)_{\text{BCD码}} = 958$

10 进制数、二进制数、16 进制数和 2-10 进制 BCD 码之间的关系如表 1.1 所示。

BCD 码既照顾了人们使用 10 进制数的习惯,又考虑了计算机的特点,确实很好,但运算起来有点麻烦。由于计算机中有专门的调整电路对它自动进行处理,所以实际应用中经常采用这种编码。

2. ASCII 码

除了 10 进制数外,各种数字、字母及字符也必须用二进制编码来表示后,计算机才能进行处理。最常用的是 ASCII 码,即美国标准信息交换码(American Standard Code for Information Interchange)。它用 7 位代码(00~7FH)来表示计算机中存储的字母、数字及符号,共可表示 128 个字符。如数字 0~9,字母 A~Z, a~z 等,其中数字 0~9 的 ASCII 码为 30H~39H,字母 A~Z 的 ASCII 码为 41H~5AH。键盘键入的数、字母以及送到 CRT 显示的字符都必须用 ASCII 码表示。ASCII 码表见附录 B。一些控制字符也列入表中,它们在计算机中实现某些控制功能,如 CR、LF 和 BEL 分别表示回车、换行和响铃,STX、ETX、ENQ 等用于串行异步通信。

1. 1. 3 带符号数的表示方法

上面介绍的二进制数没有提到符号问题,因此是一种无符号数。但在计算机中,数有正、负之分,怎么来表示符号呢?通常用最高位作为符号位。若一个数的长度为 8 位($D_7 \sim D_0$),则用 D_7 位作符号位, $D_7=1$,表示是负数; $D_7=0$,表示是正数。

例 1.4 $0101\ 1101\text{B} = +93$

$1101\ 1101\text{B} = -93$

连同符号位在一起作为一个数称为机器数,它表示的实际数值称为机器数的真值。上面两个式子中,等式左边的为机器数,右边的为真值。超过 8 位的数可以用 16 位二进制数($D_{15} \sim D_0$)来表示,这时, $D_{15}=1$,表示负数; $D_{15}=0$,表示正数。

为了运算方便,机器数通常有三种表示方法:原码、反码和补码。

1. 原码

正数的符号位用 0 表示,负数用 1 表示,其余位为数值,这种表示方法称为原码。

例 1.5 $X = +105$, $[X]_{\text{原}} = 0110\ 1001\text{B}$

$X = -105$, $[X]_{\text{原}} = 1110\ 1001\text{B}$

原码简单易懂,与真值的换算也很方便,但若要进行两个异号数相加或者两个同号数相减的运算,就要做减法操作。然而在一般的计算机中是没有减法运算部件的,减法运算也要用加法部件实现,所以要引进反码和补码。

2. 反码

正数的反码与原码相同,最高位为符号位,用 0 表示,其余位为数值。

例 1.6 $[+4]_{\text{反}} = 0000\ 0100\text{B}$

$[+31]_{\text{反}} = 0001\ 1111\text{B}$

$[+127]_{\text{反}} = 0111\ 1111\text{B}$ (最大值)

负数的反码为它的正数按位取反,即连同符号位一同取反。

例 1.7 $[-4]_{\text{反}} = 1111\ 1011\text{B}$

$[-31]_{\text{反}} = 1110\ 0000\text{B}$

$$[-127]_{\text{反}} = 1000\ 0000\text{B} \quad (\text{最小值})$$

所以,8位二进制数表示的反码范围为 $-127 \sim +127$ 。当带符号数用反码表示时,最高位为符号位,当它为正数时,后7位为真正的值,当它为负数时,后7位要取反后才能得到真正的值。

例 1.8 $[X]_{\text{反}} = 1001\ 0100\text{B}$

$$[X]_{\text{真值}} = -[110\ 1011] = -107$$

3. 补码

正数的补码表示与原码相同,最高位为符号位,用0表示,其余位为真值,负数的补码最高位为1,数值部分则由它的反码再加1形成。

例 1.9 $[+4]_{\text{原}} = 0000\ 0100 = [+4]_{\text{反}} = [+4]_{\text{补}}$

$$[-4]_{\text{原}} = 1000\ 0100\text{B}$$

$$[-4]_{\text{反}} = 1111\ 1011\text{B} \quad (\text{正数按位取反})$$

$$[-4]_{\text{补}} = 1111\ 1100\text{B} \quad (\text{反码}+1)$$

$$[+127]_{\text{原}} = 0111\ 1111\text{B} = [+127]_{\text{反}} = [+127]_{\text{补}}$$

$$[-127]_{\text{原}} = 1111\ 1111\text{B}$$

$$[-127]_{\text{反}} = 1000\ 0000\text{B}$$

$$[-127]_{\text{补}} = 1000\ 0001\text{B}$$

$$[-128]_{\text{补}} = 1000\ 0000\text{B}$$

8位二进制数能表示的补码范围为 $-128 \sim +127$,可以推算出16位二进制数能表示的二进制补码的范围为 $-32768 \sim +32767$ 。当带符号数用补码表示时,最高位是符号位,当符号位是0时,表示正数,后7位为其真正的数;当符号位是1时,表示负数,要将后7位的最低位减1,求得反码,再按位取反,才能得到真正的数(真数)。

例 1.10 若已知 $[X]_{\text{补}} = 1001\ 0100\text{B}$,求X的反码和原码。

$$[X]_{\text{反}} = [X]_{\text{补}} - 1 = 1001\ 0100\text{B} - 1 = 1001\ 0011\text{B}$$

$$[X]_{\text{原}} = 1110\ 1100\text{B}$$

$$X = -110\ 1100\text{B} = -(64+32+8+4)_{10} = -108$$

引进补码后,可以将补码连同符号位一起看作一个数,各位的权都是 2^i ,但最高位为0时,表示正数,为1时表示负数,其余为数值。

例 1.11 $[-128]_{\text{补}} = 1000\ 0000\text{B} = -128+0$,最高位的权为 $2^7 = 128$,同样,

$$[-4]_{\text{补}} = 1111\ 1100\text{B}$$

$$= -128 + (64+32+16+8+4) = -4$$

$$[+4]_{\text{补}} = 0000\ 0100\text{B} = 4$$

所以,任何一个数用补码表示后,都可以看成“连同符号位的数”。符号位也一起参加运算,一个数要减去另一个数时,只要加上其补码即可。

例 1.12 做减法运算,求 $7-19=?$ 可以用 $7+[-19]_{\text{补}}$ 来完成。

$$[7]_{\text{补}} = 0000\ 0111\text{B}$$

$$[+19]_{\text{补}} = 0001\ 0011\text{B}$$

$$[-19]_{\text{补}} = 1110\ 1101\text{B}$$

$$\begin{array}{r}
 0000\ 0111\text{B} \quad [7]_{\text{补}} \\
 + 1110\ 1101\text{B} \quad [-19]_{\text{补}} \\
 \hline
 1111\ 0100\text{B} = \text{F4H(和)}
 \end{array}$$

所以,和的补码=F4H=1111 0100B

和的反码=F3H=1111 0011B

和的原码=1000 1100B,其真值为-12

可见,7+(-19)=-12,答案正确。

前面介绍的数虽有正、负之分,但都是整数,实际应用时会遇到实数,即带小数点的数。实数通常用浮点数来表示,我们将在第14章详细介绍浮点数,并举实例来说明在32位机中如何用浮点数来编写程序。

1.2 计算机的基本结构

世界上第一台通用可编程计算机ENIAC(Electronic Numerical Integrator and Calculator)于1946年由美国宾夕法尼亚大学研制成功,它使用了17000多个电子管和超过500英里长的导线,这台庞大的计算机重量超过30吨,每秒只能执行10万次运算,但它推动世界进入了电子计算机时代。ENIAC采用重新连接线路的方法实现编程,编程时需要将大约6000多个开关进行仔细的机械定位,并用转插线把选定的各个控制部件互连起来构成程序序列,这很像早期的电话接线总机,需要许多工人花几天时间才能完成,效率很低。另外,电子管的功耗大、寿命低,维护起来很麻烦。

为替代重新连接线路实现编程,产生了用于控制计算机的计算机语言,称为机器语言(Machine Language),它由一条条1和0组成的二进制代码构成,这些代码被称为指令(Instruction),告诉计算机要执行哪些运算和操作。这种二进制代码以指令组的形式存储在计算机中,称为程序(Program)。这种方法比通过重新连接机器线路进行编程的方法有效,但编程时要涉及许多代码,开发过程仍然非常耗时。数学家冯·诺依曼(John Von Neumann)首先开发出能接收指令,并可指令存储到存储器中的系统,为了纪念他,常将这种计算机称为冯·诺依曼结构的机器。

半个多世纪以来,计算机技术不断发展,相继出现了各种类型的计算机,包括小型、中型、大型、巨型计算机以及广泛使用的微型计算机,它们的规模不同,性能和用途各异,但就其结构而言,都是冯·诺依曼计算机结构的延续和发展。

1.2.1 计算机的基本结构

1. 计算机的基本组成

冯·诺依曼计算机的基本组成框图如图1.1所示。它主要由5个部分组成,各部分的基本功能如下:

- 存储器 用来存放原始数据、中间结果以及为使计算机能自动进行运算而编制的程序,它们均以二进制的形式存放在存储器中。
- 运算器 用来执行算术运算(加、减、乘、除)、逻辑运算(与、或、非、异或)和移位等操作