

微处理器系统

原理、编程和接口

沈士团 编

北京航空學院出版社

微 处 理 器 系 统

原理、编程和接口

沈士团 编

北京航空学院出版社

内 容 简 介

本书是高等工科院校电子工程专业本科生的教科书。全书共分十章。前六章介绍微处理器的基本原理,其中包括微处理器结构、指令系统、汇编语言程序设计、输入输出技术及应用实例。第七~十章介绍了几种常用的微处理器;在工程设计中选用微处理器的基本原则;接口设计,并例举了大量接口设计的实例,如存储器设计、串/并行通信接口设计、DMA接口设计和算术运算单元(APU)9511接口设计等;另外还介绍了微处理器系统的逻辑和结构设计的一般准则及系统设计时需要考虑的某些问题。

本书由浅入深,便于自学。可供学习和从事设计微处理器系统的读者参考,也可作为大专学校有关专业的教学参考书。

JS465/28

微处理器系统原理、编程和接口

沈士团 编

责任编辑 杨昌竹

北京航空学院出版社出版

新华书店总店科技发行所发行 各地新华书店经售

河北省固安县印刷厂印装

开本: 787×1092 1/32 印张: 29.25 字数: 760千字

1988年2月第一版 1988年2月第一次印刷

印数: 1—4 500册

ISBN 7-81012-040-2/TP·012 定价: 4.80元

前 言

随着微处理器的迅速发展,它的应用已进入到各个技术领域。在高等工科院校的本科生教学中,已把有关微处理器和微型计算机原理和应用的课程列为必修课。同时,许多原来未接触过微处理器的同志,也都希望能在较短时间内获得这方面的知识。为此,我们根据国内外近年来较为流行的教科书和编者自己在教学实践和科研工作中的体会,编写了本书,作为电子工程类专业的基本教材,也可供从事微处理器应用的工程技术人员自学参考。

目前,可供人们选择使用的微处理器的种类很多,但它们的基本结构和基本原理大都雷同。根据我们的经验,从易于使第一次接触微处理器的读者能在较短的时间内掌握微处理器的基本概念、基本原理出发,选择一种简单而典型的、又能包括共有基本概念的微处理器作为基础来讲解原理是十分重要的。因此,在教材的第一部分中(第一至六章),我们选用INTEL8085作为典型,介绍微处理器的原理。但是,要掌握微处理器的原理和应用,无不涉及到有关硬件和软件知识,这对第一次接触微处理器的读者,也会带来一定困难。为此,我们在一至六章中讲解微处理器原理时,尽量避免过多介绍硬件知识,而着重于建立概念和熟悉原理。这样,只要有很少硬件知识的读者也可以掌握微处理器的原理及其有关应用方面的知识。另外,需要说明的是教材的第一部分中,所有编程实例均以INTEL8085的指令为基础。INTEL8085是一种典型的8比特微处理器,它同后来在INTEL8080基础上发展起来的ZILOG Z80微处理器在硬件和软件的许多方面是相容的。而且,掌握了8085的原理以后,也将有助于学习在国内应用较广泛的16比特微处理器,如INTEL8088、8086等。教材中大多数程序实例均可在TP-801单板机上工作。据我们对五届本科生和若干届在职工程技术人员短训班的教学实践证明,只要读者掌握了基本原理,在今后碰到其它类型的微处理器时,可以此为基础进行对比、引伸和举一反三。

本书的第一部分共分六章。第一章介绍一般计算机的工作原理、数据表示法、数制的变换及微处理器的分类和应用领域;第二章介绍微型计算机的组成和结构;第三章介绍微处理器的指令系统、寻址方式和汇编语言程序设计;第四章介绍微处理器的输入/输出技术;第五章列举了一些应用实例,其中包括微处理器应用于时序控制、条件时序控制、波形的产生、实时时钟、开关控制等。这些实例可以给读者以启发,即当碰到实际工程问题时应当如何考虑应用微处理器来解决问题。第六章简单介绍了在设计微处理器系统时的开发辅助问题。

第七至十章为教材的第二部分。第七章介绍了Z80微处理器和它的汇编语言程序设计,着重讲解了微处理器的接口信号和定时关系。了解并熟悉微处理器的定时关系是设计硬件接口的基础,而微处理器的定时关系对初学者来说又较抽象,为此,我们通过指令实例来说明它们执行过程的时间关系,使读者易于理解。第八章介绍了其它类型的微处理器,其中详细介绍了Z8单片式微型计算机,读者阅读这一部分后,可以应用Z8单片机,而无需再占有其它资料。此外,还简要介绍了目前国内广为流行的其它8比特微处理器和高性能的16比特

微处理器，供在选择微处理器时参考。第九章为微处理器的接口设计。本章以较大的篇幅阐述了微处理器的常用接口器件的接口设计方法，并列举了大量的接口电路实例，如存储器设计、串/并行通信接口设计、DMA接口设计和算术运算单元（APU）9511的接口设计等。第十章介绍了微处理器系统的逻辑和结构设计的一般准则，给出了在进行系统设计时需要考虑的某些问题，作为全书的结束语。

本书在编写过程中得到了北京航空学院李驿华、藕锋等同志的帮助，李驿华同志校阅了本书的前六章，第十章由藕锋同志协助撰写。刘素荣同志对全书文图进行了加工整理。最后承蒙马立业同志审阅，提出了宝贵意见，特此致谢。

编 者

1987年3月于北航

目 录

第一章 引论

1.1 引言	(1)
1.2 基本操作方式	(2)
1.3 数据表示法	(3)
1.3.1 整数表示法	(3)
1.3.2 负整数和补码	(9)
1.3.3 定点表示法	(12)
1.3.4 浮点表示法	(14)
1.3.5 二—十进制 (BCD) 表示法	(16)
1.3.6 字母代码	(17)
1.4 基本组成	(17)
1.5 分类	(19)
1.6 微处理器的应用领域	(20)

第二章 微型计算机的结构

2.1 引言	(22)
2.2 微处理器	(23)
2.3 存储器	(24)
2.3.1 只读存储器 (ROM)	(25)
2.3.2 随机存取存储器 (RAM)	(25)
2.4 输入/输出(I/O)设备和接口	(26)
2.5 总线结构(HIGHWAY STRUCTURE)	(26)
2.6 存储器分配图(MEMORY MAP)	(28)
2.7 地址译码	(29)
2.8 总线控制	(30)
2.9 微处理器的结构实例	(30)
2.9.1 INTEL 8085	(30)

第三章 汇编语言程序设计

3.1 引言	(33)
3.2 微处理器的寄存器	(33)
3.3 汇编语言	(34)
3.4 指令的分类	(36)
3.5 寻址方式	(37)
3.6 数据传输指令	(37)
3.6.1 寄存器寻址	(37)
3.6.2 立即寻址	(38)

3.6.3 直接寻址·····	(39)
3.6.4 寄存器间接寻址·····	(41)
3.7 汇编处理·····	(42)
3.8 算术和逻辑操作·····	(44)
3.8.1 算术运算指令·····	(44)
3.8.2 多倍精度算术运算·····	(49)
3.8.3 BCD算术运算·····	(51)
3.8.4 逻辑操作·····	(53)
3.9 传送控制·····	(58)
3.9.1 转移指令·····	(58)
3.9.2 流程图·····	(60)
3.9.3 循环程序·····	(63)
3.9.4 子程序·····	(66)
3.9.5 堆栈操作指令·····	(71)
3.9.6 参数传递·····	(73)
3.9.7 子程序嵌套·····	(74)
3.9.8 公共区域·····	(75)
3.9.9 “例外”考虑·····	(75)
3.9.10 INTEL 8085的其它指令·····	(76)
3.10 程序的效率·····	(77)
3.10.1 存储器空间·····	(77)
3.10.2 指令的数目·····	(77)
3.10.3 执行时间·····	(78)
第四章 输入和输出	
4.1 引言·····	(80)
4.2 数字输入/输出·····	(80)
4.2.1 存储器映射输入/输出·····	(81)
4.2.2 可编程输入/输出·····	(82)
4.2.3 可编程I/O器件·····	(83)
4.3 模拟输入/输出·····	(95)
4.3.1 数字/模拟变换·····	(95)
4.3.2 模拟/数字变换·····	(97)
4.4 输入/输出传输的控制·····	(103)
4.5 中断·····	(108)
4.5.1 中断的基本原理·····	(109)
4.5.2 带有中断的编程问题·····	(111)
4.5.3 多级中断·····	(113)
4.6 直访存储器(DMA)·····	(118)
第五章 应用实例	
5.1 引言·····	(122)
5.2 基本时序控制·····	(122)

5.2.1 交通灯控制器·····	(122)
5.3 条件时序控制·····	(126)
5.4 数字钟·····	(131)
5.5 波形产生·····	(136)
5.5.1 逐段线性内插法·····	(137)
5.6 温度控制器·····	(141)
第六章 开发辅助	
6.1 引言·····	(145)
6.2 单板微型计算机·····	(145)
6.3 开发系统·····	(146)
6.3.1 汇编器 (ASSEMBLER) ·····	(146)
6.3.2 编译器 (COMPILER) ·····	(148)
6.3.3 高级语言·····	(149)
6.3.4 编辑器·····	(150)
6.3.5 仿真器·····	(151)
第七章 Z 80微处理器和它的汇编语言程序设计	
7.1 引言·····	(152)
7.2 Z-80的结构 ·····	(152)
7.2.1 通用寄存器·····	(152)
7.2.2 标志寄存器·····	(155)
7.2.3 专用寄存器·····	(156)
7.3 Z 80CPU的接口信号和定时关系·····	(157)
7.3.1 接口信号·····	(158)
7.3.2 Z-80CPU 的定时关系·····	(160)
7.4 Z-80的寻址方式·····	(172)
7.5 Z 80的指令系统·····	(180)
7.6 Z 80的中断·····	(207)
7.7 Z 80汇编语言程序设计实例·····	(213)
第八章 其它微处理器	
8.1 引言·····	(226)
8.2 微处理器的主要特性·····	(226)
8.3 单片微型计算机·····	(230)
8.3.1 TMS1000系列4比特单片微型计算机·····	(230)
8.3.2 Z8系列单片微型计算机·····	(231)
8.4 MC6800系列微处理器 (MC6800和MC6809) ·····	(241)
8.5 MCS6500系列·····	(243)
8.6 十六比特微处理器·····	(244)
8.6.1 Z8000·····	(245)
8.6.2 INTEL 8086/8088 ·····	(248)
8.6.3 MC68000·····	(263)

第九章 微处理器系统接口设计

9.1 引言	(268)
9.2 CMOS线路的接口	(268)
9.3 总线缓冲	(270)
9.3.1 微型计算机内部总线缓冲	(271)
9.3.2 扩充微处理器系统时的总线缓冲	(272)
9.4 存储器接口设计	(274)
9.4.1 同步和异步存储器	(275)
9.4.2 存储器的寻址方法	(276)
9.4.3 读/写存储器 (RAM) 的接口	(278)
9.4.4 非易失性RAM系统的设计	(285)
9.4.5 动态RAM与微处理器接口	(287)
9.4.6 只读存储器 (ROM) 与微处理器接口	(294)
9.5 并行接口	(295)
9.5.1 可编程并行I/O接口电路INTEL8255A	(295)
9.5.2 Z-80 PIO	(306)
9.5.3 并行总线	(317)
9.6 可编程定时器接口	(327)
9.6.1 INTEL 8253	(328)
9.6.2 Z-80 CTC	(335)
9.7 串行接口	(345)
9.7.1 串行通信协定	(346)
9.7.2 异步协定 (RS-232C和20mA电流环接口标准)	(347)
9.7.3 同步接口	(354)
9.7.4 软件实现串行接口	(358)
9.7.5 硬件实现串行接口——USART (INTEL 8251)	(359)
9.8 DMA控制器接口	(374)
9.9 算术运算单元 (APU) AM9511的接口设计	(383)
9.10 键盘和显示器接口设计	(394)
9.10.1 键盘接口设计	(394)
9.10.2 发光二极管 (LED) 显示器接口设计	(401)

第十章 微处理器系统的逻辑与结构设计准则

10.1 引言	(407)
10.2 系统设计的基本思路及描述方法	(407)
10.3 逻辑设计准则	(409)
10.3.1 同步与异步设计	(409)
10.3.2 逻辑电路的简化	(411)
10.3.3 大规模集成电路的使用	(412)
10.3.4 采样和保持时间的考虑	(413)
10.3.5 采用寄存器库代替寄存器组	(413)
10.3.6 与异步电路的接口	(414)

10.3.7 单稳电路的正确使用.....	(415)
10.4 结构设计准则.....	(416)
10.4.1 正确阅读产品说明书.....	(416)
10.4.2 关键通路的分析.....	(416)
10.4.3 电源分配.....	(418)
10.4.4 时钟分配.....	(420)
10.4.5 不同的输入/输出引脚的处理.....	(421)
10.4.6 传输线效应.....	(421)
10.4.7 存储器系统设计原则.....	(421)
习题 (第1~4章原理部分)	(423)
附录A INTEL 8085指令系统.....	(430)
附录B ASCII (美国标准信息交换码) 表.....	(442)
附录C Z8的指令系统及控制寄存器.....	(443)
参考资料.....	(455)

第一章 引 论

1.1 引 言

自从数学家查尔斯·白培琪 (CHARLES BABBAGE) 于19世纪30年代第一次提出数字计算机的基本结构和模型以来,计算机在它的发展史上已经经历了几代。即1948年第一代电子管计算机的问世,到50年代后期生产出全晶体管化的计算机,一直到1971年美国 INTEL 公司利用大规模集成电路工艺,生产出世界上第一种微处理器和与其配套的4001 (ROM), (4002) (RAM) 组成了一个微型计算机系统。4004是一种4比特的微处理器,每秒钟可进行10万次两个4比特数的加法运算。虽然4004的指令系统并不大,只包括46个指令,但这已足以满足应用于只要求具有判断能力而并不要求大量数学运算的工业控制中。此后INTEL 公司又生产出第二代微处理器8008。这是PMOS器件微处理器,它提供了8比特的数据总线,可对18 384个存储器地址进行寻址,并具有堆栈操作和一级中断的能力。8008总共有48条指令,每秒钟可执行8万次两个8比特数的加法运算,速度要比4004高得多。8008的指令系统与4004兼容。

4004和8008只在电子工业界应用了很短一段时间,不久就被INTEL8080所取代了 (INTEL8080出现于1973年底)。与8008相比较,INTEL8080增加了30条指令。它是一种NMOS工艺的微处理器,因此速度较高,每秒可完成50万次两个8比特数的加法运算。存储器的地址范围扩充到65 536(64K)。

1976年INTEL公司在8080的基础上生产出INTEL8085。8080中央处理器 (即CPU) 由三块集成电路组成,而8085则把这三块集成电路的功能集合到一块8085的CPU电路中。8085本身具有串行输入/输出的能力,只需要+5V的单一电源供电 (8008需要两个电源,8080需三个电源)。运算速度比8080快,每秒可完成77万次两个8比特数的加法运算。虽然从性能上讲8085比8080有许多改进和提高,但它的指令系统基本上与8080相同,只比8080增加2条指令。

后来ZILLOG公司生产出第三代微处理器——Z-80。Z-80在软件上与8008和8080兼容。也就是说,原来用于8008和8080上的软件也可以直接应用于Z-80。Z-80除了保留了8008和8080原有的指令和结构外,增加了新的指令、新的寻址方式、新的硬件特点,提供了更强的功能和通用性。

Z-80的指令系统包括了全部8080的78条指令,总的指令数扩充为158条,使Z-80的功能远比8080强。图1.1-1表示了8008、8080和Z-80指令系统的相应关系。

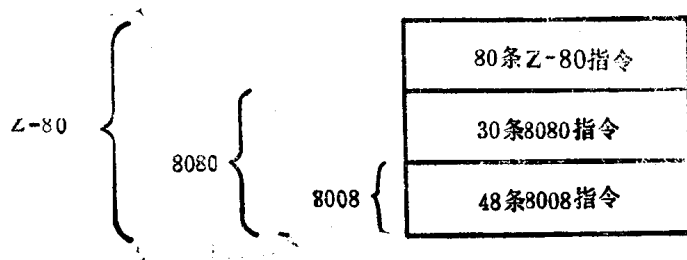


图1.1-1 8008、8080和Z-80指令关系

随着实际应用的需要（例如数据处理系统要求大量的算术运算，高的处理速度）和半导体工艺的改进，原来小型计算机所能实现的功能有可能用微处理器系统来实现，这就是后来出现的高性能16比特微处理器。这种16比特的微处理器根据它们的设计思想，可以分为两类：一类是模仿小型机而制成的16比特微处理器，它们具有与相应的小型机相同的指令系统。最早投入世界市场的16比特微处理器是TEXAS INSTRUMENTS公司的TMS9900（与小型机990相应）和DIGITAL EQUIPMENT CORPORATION的LSI-11（与PDP-11小型机相当）。第二类是在原有8比特微处理器的基础上发展起来的16比特微处理器，最典型的是MOTOROLA公司的6809和68000，INTEL公司的8088和8086及ZILOG公司的Z-8000等。

目前各种类型的微处理器和以它为基础的微型计算机象雨后春笋般出现在世界市场上，甚至于使人们在五花八门的微处理器面前有眼花缭乱之感，而难以决定选择何种微处理器较为合适。虽然微处理器种类很多，好在它们的基本原理和基本组成大致相同。为了弄清楚它们的特点和应用，首先应知道数字计算机的基本组成和原理。这样才可能根据各种微处理器的特点合理选择和使用。

本章中，我们首先介绍微处理器、微型计算机的某些基本概念，然后介绍微处理器的分类和应用领域。

1.2 基本操作方式

从本质上讲，所有的数字计算机尽管它们应用于不同的领域，不同的场合，但它们都以同样的方式工作。因此计算机不论是大型的还是小型的，都是一种灵活而通用的机器，只要通过适当的组织安排，就可用以解决某些特定的任务。当然要解决一个特定的任务，就要求计算机有一定的操作顺序。以我们日常使用的计算器来说，它有各种操作，如加、减、乘、除等，使用者可根据所需解决的问题选择一定操作顺序（先加减后乘除，或反之），去解决这个问题。类似地，数字计算机也可以执行一系列的基本操作，我们称之为机器指令，使用者选择这些指令并按一定的方式进行排列组合，去解决一个特定的问题。这种按一定要求和次序列出来的操作（机器指令组合）称为“程序”。

一个计算器执行每种基本操作大概需要几毫秒的时间。因此用这种计算器解决一个问题所需的时间主要决定于使用者一步一步地按“按键”的速度，而这是相当慢的。这样一来，计算器可以高速运算的固有特性丧失掉了。但数字计算机就不同了，首先，它执行每

一种操作的速度很高，一般执行一条机器指令只需几微秒，而且可把一步一步的操作预先存储在计算机内部（这就是所谓的“存储程序”的概念），以充分发挥计算机高速运转的固有特性。这就是计算器和计算机系统的基本不同点。

信息或数据输入到计算机中去或从计算机输出，它们的形式随着不同的应用有很大的不同。比如，对于数学上的应用，计算机的输入输出数据可能仅仅是一些数字。但如果把计算机应用于一个温度控制系统，那末这时的输入数据可能是一个代表受控温度的数值。而它的输出信号可能被用来控制某个元件的“开”或“关”的动作。如果我们不去考虑计算机的各种不同应用，而只考虑计算机本身，那末所采用的存储和编码的信息的形式是相同的，都采用二进制系统。所有的数据输入到计算机之前必须把它变换成二进制码的形式，然后经过运算或处理后，以二进制代码的形式输出。当然计算机的二进制代码输出有时也要变换成系统所需要的形式，比如把二进制数字量变换成模拟量。因此从本质上讲，计算机是一种二进制数字系统。也就是说，计算机只认得二进制代码。计算机所要处理的数据，都要用二进制代码来表示。所有的字母、符号等也要用二进制代码表示。

1.3 数据表示法

我们知道，数据需要在计算机中存储和处理，而数据应当有一定的格式。上节已提到数字计算机从本质上讲是一种二进制数字系统，因此不管用什么方式来表示数据，它们都必须是用1和0表示的二进制代码。无论是输入到计算机中去的数据，还是控制计算机各种操作的机器指令，都用二进制代码的形式表示。

第一类称为数的表示法，它包括三种情况：

1. 整数表示法（更一般地说即定点表示法）；
2. 浮点表示法；
3. 十进制表示法。或称为二—十进制表示法，或简称为BCD。

数的这三种表示法可以适应计算机在实际应用中各种不同情况的需要。

第二类表示法称为文字表示，或称为字母代码。这种表示法既可用来表示数字，也可用来表示字符。目前我国正在发展的汉字编码也应属于这种表示法。常用的字母代码有以下两种：

1. ASCII代码，即美国信息交换标准代码（AMERICAN STANDARD CODES FOR INFORMATION INTERCHANGE）；
2. EBCDIC代码，即扩充二—十进制代码（EXTERNAL BINARY CODED-DECIMAL INTERNAL CODE）。

1.3.1 整数表示法

从理论上讲，“数”的含义同用来表示它的符号无关。“数”是数学上的一种抽象。随着历史的发展，“数”可以有若干种数制来表示。在这一节中我们首先讨论正整数的表示法。

根据基本的算术运算，我们日常生活中所采用的正整数可表示为

$$a_n 10^n + a_{n-1} 10^{n-1} + \cdots + a_1 10^1 + a_0 10^0 = a_n a_{n-1} a_{n-2} \cdots a_1 a_0 \quad (1-1)$$

其中 $a_n, a_{n-1}, \dots, a_1, a_0$ 称为“数字”。显然它们应当是0~9之间的十个数。

例如：正整数6235可按式(1-1)写成如下形式

$$6 \times 10^3 + 2 \times 10^2 + 3 \times 10 + 5 = 6\,235$$

把式(1-1)推广到一般形式

$$a_n X^n + a_{n-1} X^{n-1} + \dots + a_1 X + a_0 \tag{1-2}$$

其中 X 是一个正整数，称为“基数”。基数实际上就是用 $a_n \dots a_1, a_0$ 诸符号所表示的数字的个数。例如基数 $X=10$ ，则数字符号的个数等于10，即0~9之间的10个数。若基数为2，则数字符号个数为2，即数字符号为0和1。

若基数为10，那末用式(1-2)表示的数据表示法称为十进制数制，与此相应的算术运算为十进制算术运算。若基数为2，那么这种数据表示法称为二进制数制，相应的算术运算为二进制算术运算。在二进制中，数字 $a_n \dots a_1 a_0$ 称为比特(BIT)，只用两种符号0和1表示。如

$$1 \times 2^3 + 0 \times 2^2 + 1 \times 2 + 1 = 1011_2$$

在计算机中除了用二进制数制表示的正整数外，还常采用另外两种数制——八进制数制(基数 $X=8$)和十六进制数制(基数 $X=16$)。二进制、八进制、十六进制三种数制的关系如图1.3-1所示。其中括号内所表示的是相应的二进制代码。

二 进 制 $X=2$	八 进 制 $X=8$	十 进 制 $X=10$	十 六 进 制 $X=16$
0	0 (000)	0 (0000)	0 (0000)
1	1 (001)	1 (0001)	1 (0001)
	2 (010)	2 (0010)	2 (0010)
	3 (011)	3 (0011)	3 (0011)
	4 (100)	4 (0100)	4 (0100)
	5 (101)	5 (0101)	5 (0101)
	6 (110)	6 (0110)	6 (0110)
	7 (111)	7 (0111)	7 (0111)
		8 (1000)	8 (1000)
		9 (1001)	9 (1001)
			A (1010)
			B (1011)
			C (1100)
			D (1101)
			E (1110)
			F (1111)

图1.3-1 常用数制的关系

需要注意的是，在十六进制数制中，整数10~15用相应的英文字母A、B、C、D、E、

F表示。

虽然八进制和十六进制数不能直接用于计算机的操作，因为计算机只认得二进制代码。但对用户来说，用二进制代码同计算机打交道是非常不方便的，易产生错误，有错也不易检查。因此在计算机中常采用八进制或十六进制来表示二进制信息。同二进制比较，采用八进制和十六进制表示一个数可以减少数的位数。例如十进制数字“11”，如果用二进制表示，它是一个四位数：1011；而用八进制表示，则为13，是一个二位数；用十六进制表示则为B，只是一个一位数。因此采用八进制和十六进制数制更易于表示很大的整数。为了避免混淆，今后我们用字母“B”表示二进制，例如：1101B是一个二进制数；用字母“H”表示十六进制，如：52A9H是一个十六进制数；除了对数有特殊说明外，如果数的结尾没有字母，则表示是一个十进制数。

由于人们习惯于用十进制数制思考问题，而计算机只能采用二进制代码，因此在实际应用中存在着一个从一种数制变换到另一种数制的问题。下面我们分别讨论十进制、二进制、八进制和十六进制数制的互相转换关系。

一、十进制—二进制互相变换

根据式(1-2)，我们可以获得二进制数变换成十进制数的一般关系式，即

$$a_n 2^n + a_{n-1} 2^{n-1} + \dots + a_1 2 + a_0 \quad (1-3)$$

例如二进制数110101B，它的等价的十进制数为53。

$$1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2 + 1 = 32 + 16 + 0 + 4 + 0 + 1 = 53$$

如果把式(1-3)稍加变换，我们就可得到一种更为系统的算法

$$\begin{aligned} & a_n X^n + a_{n-1} X^{n-1} + \dots + a_1 X + a_0 \\ &= ((\dots (a_n X + a_{n-1}) X + \dots) X + a_1) X + a_0 \end{aligned} \quad (1-4)$$

这样上例的变换可以改写为

$$\begin{aligned} & 1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2 + 1 \\ &= ((((1 \times 2 + 1) 2 + 0) 2 + 1) 2 + 0) 2 + 1 \end{aligned}$$

现把变换过程列出来，即

1	←	1	1	0	1	0	1	B=53
× 2								
2								
+ 1	←							
3								
× 2								
6								
+ 0	←							
6								
× 2								
12								
+ 1	←							
13								
× 2								
26								
+ 0	←							
26								
× 2								
52								
+ 1	←							
53								

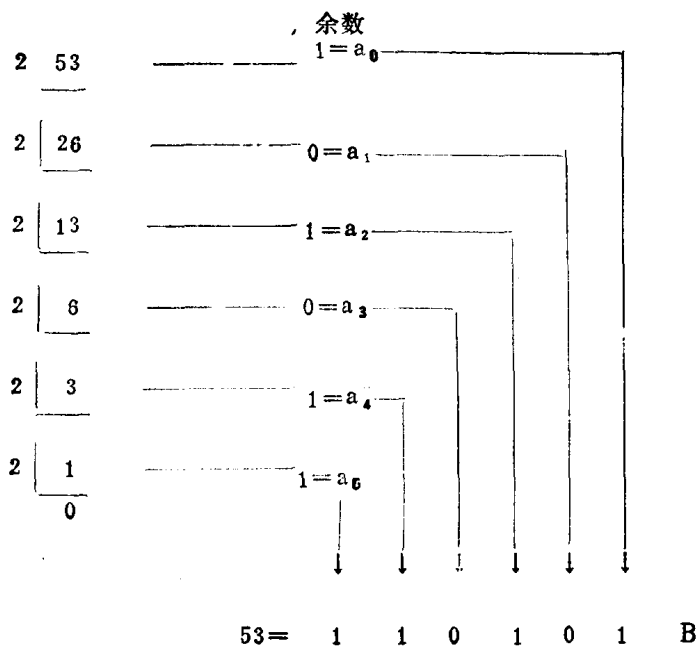
通过这一运算关系，我们不难从式(1-4)推导出十进制数变换成等价二进制数的一般公式。把式(1-4)除以基数X，即

$$\frac{((\dots(a_n X + a_{n-1})X \dots)X + a_1)X + a_0}{X} = (\dots(a_n X + a_{n-1})X \dots)X + a_1 + a_0/X \quad (1-5)$$

其中 $(\dots(a_n X + a_{n-1})X \dots)X + a_1$ 是商，而 a_0 是余数，然后把商再除以X，即可得

$$\frac{(\dots(a_n X + \dots a_{n-1})X \dots)X + a_1}{X} = (\dots(a_n X + a_{n-1})X \dots)X + a_2 + a_1/X \quad (1-6)$$

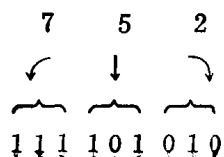
这里 a_1 是余数。显然，如果我们按上述方法连续重复计算，则可获得全部系数 $a_0 \sim a_n$ ，根据系数 $a_0 \sim a_n$ ，就可写出相应的二进制数。例如，把十进制数53变换成等价的二进制数，它的计算过程为



二、二进制—八进制之间的变换

首先来看八进制数变换成等价的二进制数。这只要把八进制数的每一个数字变换成用三个比特表示的二进制代码，然后按原来数字的次序组合在一起，即得到等价的二进制数。

例 1



这个变换过程可用下面的等式加以证明

$$7 \times 8^2 + 5 \times 8 + 2$$

$$= (1 \times 2^2 + 1 \times 2 + 1) 2^6 + (1 \times 2^2 + 0 \times 2 + 1) 2^3 + (0 \times 2^2 + 1 \times 2 + 0)$$

$$= 1 \times 2^8 + 1 \times 2^7 + 1 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2 + 0$$

$$= \underline{111101010}_2$$

~~32~~ 71.

$$7 \times 16 + 1$$

$$= (2^2 + 2 + 1) 2^4 + 16$$

$$= 2^6 + 2^5 + 2^4 + 2^4$$

$$= 2^6 + 2^5 + 2^5 = 2^6 + 2^6 = 2^7$$

二进制数变换成等价的八进制数即为上述变换的逆过程。只要把二进制数从低位向高位每三个比特分为一组，把每一组变换成相应的八进制数即可。

例 2

8

16

~~24~~

11 010 110 011 B

~ ~ ~ ~

↓ ↓ ↓ ↓

3 2 6 3

三、二进制—十六进制的变换

二进制—十六进制的变换与上面的二进制—八进制之间的变换类似，不同的只是每位十六进制数字用四个比特的二进制代码表示。我们用两个例子加以说明：

例 1 十六进制数9A1H变换成等价的二进制数：

9	A	1	H
↓	↓	↓	
{ { {			
1001	1010	0001	B

例 2 二进制数11010110100B变换成十六进制数

110	1011	0100	B
{ { {			
6	B	4	H