第 1 章　微型计算机的基本结构和运算基础

1.1　概　　述

计算机是一种能够自动完成运算的电子装置，其之所以能够自动完成运算，是因为它能够存储程序和原始数据、中间结果及运算最终结果。存储程序和采用二进制运算奠定了冯·诺依曼结构的计算机的设计思想，无论计算机技术怎么发展，这一点是相对不变的。

利用计算机不仅能够完成数学运算，而且还可以进行逻辑运算，同时还具有推理判断的能力。现在，科学家们正在研究具有“思维能力”的智能计算机。随着科学技术的发展，人们对计算机能力的认识也在不断地深入。

1.1.1　微型计算机的发展概况

1946 年在美国诞生了世界上第一台电子计算机 ENIAC，该机字长为 12 位，每秒完成 5000 次加法运算，它使用了 18 800 个电子管、70 000 个电阻、1000 个电容、6000 个开关，占地约为 170m^2，耗电 150kW，重达 30t。这个庞然大物被称作第一代电子计算机，为当今的电子计算机奠定了基础。1958 年至 1964 年，用晶体管取代电子管，成为第二代电子计算机，它不仅大大缩小了计算机的体积，而且还降低了成本，同时将运算速度提高了近百倍。1965 年集成电路问世，形成了中、小规模集成电路构成的第三代计算机。1970 年出现了以大规模集成电路为主体的第四代计算机，这是大规模集成电路迅猛发展的产物。所谓第五代计算机，其目标主要是：采用超大规模集成电路，在系统结构上要类似人脑的神经网络，在材料上使用常温超导材料和光器件，在计算机结构上采用超并行的数据流计算等。

由于在一块芯片上可集成上千万个电子元件，因而使电子计算机的体积大为缩小，这就导致了微型计算机的问世。因为微型计算机具有体积小、功耗低、重量轻、价格低、可靠性高、使用方便等一系列优点，因此获得了广泛的应用和迅速的发展。自微型计算机于 1971 年问世以来，大约每隔 2～4 年就更新换代一次，至今已经历了多个阶段的演变。

1．第一阶段（1971—1973 年）

第一阶段为 4 位和低档 8 位微处理器及微型计算机。美国 Intel 公司首先研制成功 4 位的 4004 微处理器，以它为核心再配以 RAM，ROM 和 I/O 接口芯片就构成了 MCS-4 微型计算机。随后又研制出 8 位的 8008 微处理器及 MCS-8 微型计算机，其特点是指令系统简单，运算功能较差，速度较慢（平均指令执行时间约为 20μs）。

2．第二阶段（1973—1978 年）

第二阶段为中档 8 位微处理器和微型计算机。其间又分为两个阶段。1973—1975 年为典型的第二代，以美国 Intel 公司的 8080 和 Motorola 公司的 MC 6800 为代表，集成度提高 1～2 倍，运算速度提高一个数量级，1976—1978 年为高档的 8 位微型计算机阶段，被称为第二代半微型计算机，代表产品是美国 Zilog 公司的 Z80 和 Intel 公司的 8085 微处理器，集成度和运算速度都比典型的第二代提高 1 倍以上。

3．第三阶段（1978—1981 年）

第三阶段为 16 位微处理器和微型计算机，又称为第一代超大规模集成电路的微处理器。其代表产品是 Intel 公司的 8086/8088，Zilog 公司的 Z8000 和 Motorola 公司的 M68000。这些 16 位微型计算机都具有丰富的指令系统，并配有强有力的软件系统，时钟频率为 4～8MHz，平均指令执行时间为 0.5μs。

4．第四阶段（1981 年以后）

第四阶段为高性能的 16 位机和 32 位微处理器和微型计算机。其代表产品是 Intel 公司的 80386 和 Motorola 公司的 68020，时钟频率达 16～20 MHz，平均指令执行时间约为 0.1μs。通常称这类微处理器构成的微型计算机为超级微型机。Intel 公司又相继开发出了 80486，80586 等微处理器。

第四代微处理器向着系列化方向发展，首先 Intel 公司推出了性能更高、功能更强的高级 16 位微处理器 80186 和 80286，它们与 8086 向上兼容。1985 年，Intel 公司又率先推出了 32 位微处理器 80386，它与 8086、80186、80286 向上兼容。进入 20 世纪 90 年代，该公司相继推出 80486 和 80586，形成完整的 80 系列。同时还不断推出带多媒体技术的 Pentium（奔腾）系列机。

与此同时，Motorola 公司也推出了 32 位微处理器 M68020；HP 公司推出了 HP32；IBM 公司推出了 IBM320；Zilog 公司推出了 Z80000；等等。

5．第五代 32 位高档处理器（1993 年后）

1993 年，Intel 公司推出了 32 位处理器 Pentium（以 P5 代称）。它集成了 330 万个晶体管，内部采用 4 级超标量结构，数据线 64 位，地址线 36 位。工作频率为 60/66MHz，处理速度达 110MIPS。由于第一代 Pentium 采用 0.8μm 工艺技术和 5V 电源驱动，使得芯片尺寸较大，成本过高；其功耗达 15W，使系统散热成为问题。1994 年 3 月，Intel 公司推出了第二代 Pentium（以 P54C 代称），P54C 采用 0.6μm 工艺和 3.3V 电源，功耗仅为 4W，而且可在不需要时自动关闭浮动单元，散热问题基本得以解决。P54C 的主时钟为 100MHz 和 90MHz 两种。常规配置下，P54C 系统的处理能力比 P5 系统高出了 40%。在体系结构上，Pentium 在内核中采用了 RISC 技术 。

同期推出的第五代微处理器还有 IBM、Apple 和 Motorola 三家联盟的 Power PC（这是一种完全的 RISC 微处理器），以及 AMD 公司的 K5 和 Cyrix 公司的 M1 等。

6．第六代微处理器（1996 年）

1996 年，Intel 公司推出了 Pentium Pro 微处理器（也将此后的微处理器称为第六代）。该处理器的集成电路采用了 0.35μm 的工艺，时钟频率为 200MHz，运算速度达 200MIPS。Pentium Pro 的两个显著特点是：内部集成了 16KB 的一级（L1）高速缓冲存储器和 256KB 的二级（L2）高速缓冲存储器；使用三个执行部件，可同时执行三条命令。

从 1997 年后，Intel 又进一步推出了一系列新的 Pentium 型微处理器，PentiumⅡ、Pentium Ⅲ和 Pentium 4。其他公司类似的产品还有 AMD 的 K7。这些 CPU 的集成度已高达近 1 千万个晶体管，时钟频率也达到了 1GHz 以上。Pentium 4 CPU，工作频率已达 3.2GHz。

7．Intel 超线程处理器（2002 年）

Intel 公司于 2002 年推出了具有超线程技术的 IA-32 系列处理器。超线程（Hyper-Thread，HT）技术允许单个物理处理器用共享的执行资源并发地执行两个或多个分别的代码流（线程），以提高 80x86 系列处理器执行多线程操作系统与应用程序代码的性能。

从体系结构上说，支持 HT 技术的 IA-32 处理器，在一个物理处理器核中由两个或多个逻辑处理

器构成，每个逻辑处理器有它自己的 IA-32 体系结构状态。每个逻辑处理器由全部的 IA-32 数据寄存器、段寄存器、控制寄存器与大部分的 MSR 构成。

8．Intel 双核技术处理器（2005 年）

2005 年，Intel 公司推出了使用双核技术的奔腾处理器极品版 840 IA-32 处理器。这是 IA-32 处理器系列中引入双核技术的第一个成员。此处理器用双核技术与超线程一起提供硬件多线程支持。双核技术是在 IA-32 处理器系列中硬件多线程能力的另一种形式。双核技术由用在单个物理包中的两个分别的执行核心提供硬件多线程能力。因此，Intel Pentium 处理器极品版在一个物理包中提供 4 个逻辑处理器（每个处理器核有两个逻辑处理器）。

Intel Pentium D 处理器也以双核技术为特色。此处理器用双核技术提供硬件多线程支持，但它不提供超线程技术。因此，Intel Pentium D 处理器在一个物理包中提供两个逻辑处理器，每个逻辑处理器拥有处理器核的执行资源。

Intel Pentium 处理器极品版中引入了 Intel 扩展的存储器技术（Intel EM64T）对于软件增加至 64 位线性地址空间并支持 40 位物理地址空间。此技术也引入了称为 IA-32e 模式的新的操作模式。

IA-32e 模式在两种子模式之一上操作：

- 兼容模式允许 64 位操作系统不修改地运行大多数 32 位软件。
- 64 位模式允许 64 位操作系统运行应用程序访问 64 位地址空间。

Intel EM64T 的处理器可以支持 80x86 系列软件，因为它能运行在非 64 位传统模式。大多数已存在的 IA-32 应用程序也能在兼容模式中运行。

微处理器的迅速发展和更新换代，使基于微处理器的微型计算机的性能在不断提高。

所谓微处理器（μPU）是把运算器和控制器集成在一个芯片上，又称 CPU。

所谓微型计算机是把微处理器（CPU）配上一定容量的半导体随机存储器（RAM）、半导体只读存储器（ROM）及接口电路、必要的外设组成的。

所谓微型计算机系统是指硬件系统和软件系统的总称。硬件系统包括微型计算机、时钟、电源等；软件系统包括系统软件和应用软件。

所谓单板机是把 CPU 一定数量的存储器芯片和 I/O 接口芯片装在一块印制电路板上，在该板上再配以具有一定功能的输入/输出设备（如小键盘等）。

所谓单片机是把微处理器 CPU 一定容量的存储器和必要的 I/O 接口电路集成在一个硅片上。有的单片机还包括模数（A/D）数模（D/A）转换器。

微型计算机的发展趋势，不仅向小型化方向发展，而且向巨型化方向发展，以获得基于微型机体系结构。

1.1.2　微型计算机的特点和分类

1．微型计算机的特点

（1）体积小、重量轻、耗电省

由于采用大规模集成电路和超大规模集成电路，使微型机所含的器件数目大为减小，体积大为缩小。微型机还大量地采用大规模集成专用芯片（ASIC）和通用可编程门阵列（GAL）器件，使得微型机的体积又明显缩小。而微型机中的芯片大多采用 MOS 和 CMOS 工艺，因此耗电量就很小。

（2）可靠性高

由于微处理器及其配套系列芯片采用大规模集成电路，减少了大量的焊点，简化了外接线和外加逻辑，因而大大提高了可靠性。

（3）系统设计灵活，使用方便

由于微处理器芯片及其可选用的支持逻辑芯片都有标准化、系统化的产品，同时又有许多有关的支持软件可选用，用户可根据不同的要求构成不同规模的系统。

（4）价格低廉

由于微处理器及其配套系列芯片采用集成电路工艺，集成度高，产品造价十分低廉。

（5）维护方便

由于微处理器及其系统产品已标准化、模块化和系列化，从硬件结构到软件配置都进行了较全面的考虑。一般可用自检诊断及测试发现系统故障。发现故障后，可方便地更换标准化模块芯片来排除故障。

2．微型计算机的分类

可以从不同的角度对微型计算机进行分类。

- 按微处理器的字长，可分为 4 位、8 位、16 位、32 位、64 位微处理器。
- 按微型计算机的组装形式，可分为单片、单板、多板微型计算机等。
- 按应用领域不同，又可分为控制用、数据处理用微型计算机等。
- 按微处理器的制造工艺，又可分为 MOS 型器件和双极型器件两大类。

1.1.3 微处理器的字长

字长是指计算机字所含二进制位数。计算机字也就是作为一个整体被一次传送或运算最多的二进制数位。它和计算机能够处理二进制信息的位数是两个概念。如 32 位和 32 位数相加，用 8 位机需加 4 次，用 16 位机需加两次，用 32 位机只加一次即可。很显然，32 位机的速度要快得多。字长是对某一型号的机器而言的。

字节，无论对哪个厂家、哪种型号的计算机，都指的是 8 位二进制信息。

1.2 运 算 基 础

1.2.1 进位计数制及其相互转换

1．进位计数制

凡是按进位的方式计数的数制称为进位计数制，简称进位制。数据无论使用哪种进位制表示，都涉及基数（Radix）与各位数的“权”（Weight）。所谓某进位制的基数是指该进位制中允许选用的基本数码的个数。对任意进位制数都可以写成按权展开的多项式和的形式

$$K = K_{n-1} \times R^{n-1} + K_{n-2} \times R^{n-2} + \cdots + K_1 \times R^1 + K_0 \times R^0 + K_{-1} \times R^{-1} + K_{-2} \times R^{-2} + \cdots + K_{-m} \times R^{-m}$$

$$= \sum_{i=n-1}^{-m} K_i \times R^i$$

式中：i 为数位；m, n 为正整数；R 为基数；K_i 为第 i 位数码。

总之，位置计数法（带权记数法）的数制均有以下几个主要特点。

（1）数码个数等于基数，最大数码比基数小 1。

（2）每个数码都要乘以基数的幂次，而该幂次是由每个数所在的位置决定的，即“位权”，简称权。

（3）低位向高位的进位是“逢基数进 1”。

2．几种常用的进位制

计算机中采用二进制计数法，但在编程时，为了书写方便，常采用八进制或者十六进制数，特别是十六进制数。因为它们和二进制数之间有一种特殊的“缩写”关系，因此得到广泛使用。而人们日常生活习惯使用的是十进制数，这样它们之间就存在着一种对应转换关系。

（1）十进制数（Decimal Notation）

它有以下 3 个特点。

① 10 个数码，即 0，1，2，3，4，5，6，7，8，9。

② 在数的表示中，每个数码都要乘以基数 10 的幂次，而此幂次是由该数码所在位置决定的。

例 1.1　$555.5D=(555.5)_{10}=5\times10^2+5\times10^1+5\times10^0+5\times10^{-1}$

D 可以省略。可以看出同样的数码在不同的位置（权不一样）所表示的数值不一样。

③ 低位向高位的进位是逢 10 进 1。

（2）二进制数（Binary Notation）

它有以下 3 个特点。

① 两个数码，即 0 和 1。

② 在数的表示中，每个数码都要乘以基数 2 的幂次，而此幂次是由该数码所在位置决定的。

例 1.2　$101.1B=(101.1)_2=1\times2^2+0\times2^1+1\times2^0+1\times2^{-1}$

B 不能省略。

③ 低位向高位的进位是逢 2 进 1。

由于二进制数还具有以下独特的性质，所以奠定了当代计算机的设计基础。

- 具有两个不同的数码，在自然里很容易找到具有两个稳定的物理状态来实现它。
- 整个数向左移一位，数值增大一倍，反之，向右移一位，数值就减小一半。
- 对于二进制整数，若最低位是 1 则为奇数，是 0 则为偶数。
- 二进制数运算规则简单，如

  0+0=0;

  0+1=1；

  1+0=1；

  1+1=0，向上进位 1。

也就是说，两个一位二进制数相加，相同则本位和为 0，不相同则本位和为 1；只有在都为 1 时才有进位。本位和恰好是异或二值逻辑的关系，进位则是二值逻辑与的关系。这样，在计算机里，是把二进制数的运算用逻辑运算来实现的，因此 CPU 是由一些组合逻辑电路（实现一定逻辑关系的门电路）和时序逻辑电路（能够存储二进制信息的电路）组成的。

（3）八进制数（Octal Notation）

它有以下 3 个特点。

① 有 8 个数码，即 0，1，2，3，4，5，6，7。

② 在数的表示中，每个数码都要乘以基数 8 的幂次，而此幂次是由该数码所在位置决定的。

例 1.3　$35.7Q=(35.7)_8=3\times8^1+5\times8^0+7\times8^{-1}$

Q 不能省略。这里用 Q 而不用 O 是为避免把字母 O 误作数字 0。

③ 低位向高位的进位是逢 8 进 1。

（4）十六进制数（Hexadecimal Notation）

它有以下 3 个特点。

① 有 16 个数码，即 0，1，2，3，4，5，6，7，8，9，A，B，C，D，E，F。

② 在数的表示中，每个数码都要乘以基数 16 的幂次，而此幂次是由该数码所在位置决定的。

例 1.4 $5BD.3H = (5BD.3)_{16} = 5\times16^2 + 11\times16^1 + 13\times16^0 + 3\times16^{-1}$

例 1.5 $0FEA.C8H = (FEA.C8)_{16} = 15\times16^2 + 14\times16^1 + 10\times16^0 + 12\times16^{-1} + 8\times16^{-2}$

H 不能省略。第一个数码为字母时，前面要加数字 0，以避免把十六进制数误作变量或标号名（变量和标号在后面汇编语言的章节中说明）。

③ 低位向高位的进位是逢 16 进 1。

3．数制之间的转换关系

（1）任意进制数转换成十进制数

任意进制数转换成十进制数就是该数按权展开多项式之和。

例 1.6 二进制数 101.1B 转换成十进制数

$101.1B = 1\times2^2 + 0\times2^1 + 1\times2^0 + 1\times2^{-1} = 4 + 0 + 1 + 0.5 = 5.5$

其他进位制数同理。

（2）十进制数转换成任意进制数

① 整数转换法就是除基数取余数法，第一次得到的余数为最低位，最后得到的余数为最高位。

例 1.7 $(28)_{10} = (?)_2$

		余数
2	28	
2	14	0
2	7	0
2	3	1
2	1	1
	0	1

最后转换的结果是 28D=11100B

其他进位制数同理。

② 小数转换法就是乘基数取整数法，第一次得到的整数为小数的最高位，直到十进制小数部分乘积为 0 时停止，最后得到的是转换结果的小数最末位（若出现无限循环，可按要求取小数点后几位即可）。

例 1.8 $(0.125)_{10}=(?)_2$

		取整数部分
	0.125	
×	2	
	0.125	0
×	2	
	0.500	0
×	2	
	1.000	0

最后转换结果是 0.125D=0.001B

其他进位制数同理。

（3）二进制、八进制和十六进制之间的转换

这 3 种进制表示的数据之间的转换十分方便，因为每 3 个二进制位正好对应一个八进制位，每 4 个二进制位可以变换成一个十六进制位。二进制、八进制和十六进制基本对应关系如表 1-1 所示。

表 1-1　二进制、八进制和十六进制基本对应关系

二 进 制 数	八 进 制 数	十六进制数	十 进 制 数
0000	00	0	0
0001	01	1	1
0010	02	2	2
0011	03	3	3
0100	04	4	4
0101	05	5	5
0110	06	6	6
0111	07	7	7
1000	10	8	8
1001	11	9	9
1010	12	A	10
1011	13	B	11
1100	14	C	12
1101	15	D	13
1110	16	E	14
1111	17	F	15

把二进制数转换成八进制或十六进制数时，按每 3 位或每 4 位二进制位分组，应保证从小数点所在位置分别向左和向右进行划分，若整数部分的位数不是 3 或 4 的整数倍，可按在数的最左侧补 0 的方法处理，对小数部分则按在数的右侧补 0 的方法处理。

例 1.9　将二进制数 1011011　10101B 分别转换成八进制和十六进制。

1011011.10101B=001,011,011.101，010B=133.52Q

1011011.10101B=0101,1011.1010，1000B=5B.A8H

例 1.10　将十进制数 25.625 转换为二进制数、八进制数及十六进制数。

解：十进制数转换为二进制数时，对于整数部分，采用除 2 取余数法，即逐次用 2 去除要转换的十进制数，直至商为 0，每次所得的余数即为二进制数码，最先得到的为整数的最低有效位 K_0，最后得到的是整数的最高有效位 K_{n-1}。对于小数部分，采用乘 2 取整数法，即逐次用 2 去乘要转换的十进制小数，将每次所得的整数 0 或 1，依次记作 K_{-1}，K_{-2}…。注意，十进制小数并不是都能用有限位的二进制数精确表示的，这时只要根据精度要求，转换到一定的位数即可。

		余数		整 数
2	25			
2	12	$K_0=1$	$0.625\times2=1.25$	$k_{-1}=1$
2	6	$K_1=0$	$0.25\times2=0.5$	$k_{-2}=0$
2	3	$K_2=0$	$0.5\times2=1$	$k_{-3}=1$
2	1	$K_3=1$		
	0	$K_4=1$		

故 25.625 对应的二进制数为 11001.101B。

八进制、十六进制和二进制之间的转换是非常简单的，分别按 3 位、4 位二进制数对应转换即可。方法是以小数点为界，整数部分自右至左，小数部分自左至右分组，若转换为八进制，3 位为一组，若转换为十六进制，4 位为一组，不足时补 0。

本例中，11001.101B = 011,001.101B = 31.5Q;

11001.101B = 0001,1001.1010B = 19.AH

所以，25.625 对应的二进制数、八进制数及十六进制数分别为 11001.101B，31.5Q，19.AH。

例 1.11 将二进制数 10110B，八进制数 125Q 及十六进制数 5AF.8H 转换为十进制数。

解：将非十进制数转换为十进制数时，一般按其定义展开为多项式，将系数与权用十进制表示，然后进行相应的四则运算即可得到运算结果。

$10110B = 1\times2^4 + 0\times2^3 + 1\times2^2 + 1\times2^1 + 0\times2^0 = 16 + 0 + 4 + 2 + 0 = 22D$

$125Q = 1\times8^2 + 2\times8^1 + 5\times8^0 = 64 + 16 + 5 = 85D$

$5AF.8H = 5\times16^2 + 10\times16^1 + 15\times16^0 + 8\times16^{-1} = 1280 + 160 + 15 + 0.5 = 1455.5D$

1.2.2 二进制数的运算规则

采用二进制实现各种算术与逻辑运算，这是因为二进制数每一位都仅有 0 和 1 两个状态，实现它的物理器件很容易找到，它很好地对应着半导体器件的饱和与截止、电流的有和无、电位的高与低等。另外二进制数运算规则简单，容易用逻辑电路实现。

1．加法规则

（1）一位二进制数加法规则

0+0 = 0

0+1 = 1

1+0 = 1

1+1 = 0 且向高位产生进位 1

（2）多位二进制数加法规则

例 1.12

$$\begin{array}{r} 1101B \\ +\ 1001B \\ \hline 10110B \end{array}$$

最高位产生向上进位 1。

2．减法规则

（1）一位二进制数减法规则

0–0=0

0–1=1 且向高位产生借位 1

1–0=1

1–1=0

（2）多位二进制数减法规则

例 1.13

$$\begin{array}{r} 0110B \\ -\ 1101B \\ \hline 1001B \end{array}$$

最高位产生向上借位 1。

3．乘法规则

（1）一位二进制数乘法规则

0×0 = 0

0×1 = 0

1×0 = 0

1×1 = 1

可以看出，乘数为 0 时，其结果为 0；乘数为 1 时，其结果等于被乘数。

（2）多位二进制数乘法规则

例 1.14

$$\begin{array}{r} 1011\text{B} \\ \times\ 1101\text{B} \\ \hline 1011 \\ 0000 \\ 1011 \\ +\ 1011 \\ \hline 10001111\text{B} \end{array}$$

4．除法规则

（1）一位二进制数除法规则

0÷0

1÷0 非法，除数不能为 0

0÷1=0

1÷1=1

（2）多位二进制数除法规则

例 1.15

$$\begin{array}{r} \quad\ 1101\text{B} \\ 1001\text{B}\ \sqrt{\ 1110101\text{B}} \\ -1001 \\ \hline 01011 \\ -1001 \\ \hline 00100 \\ -0000 \\ \hline 1001 \\ -1001 \\ \hline 0 \end{array}$$

1.2.3　计算机中的四则运算

在计算机里完成二进制数的四则运算最终都是进行加法运算，所以在 CPU 内部只有加法器。CPU 把减法运算通过加补码来实现；把乘法运算通过部分积右移加被乘数或 0 实现；把除法运算通过部分余数左移加除数补码或 0 实现。减法用加补码实现，在后面讲到补码时再加以说明。这里仅对实现乘法运算加以说明。

如前所举之例 1011B×1101B=10001111B。

从前面相乘的过程来看，类似于十进制数相乘的规律。但用计算机按此规律实现就不容易了。因为 n 位数乘 n 位数要完成 n 个 $2n$ 位数同时相加，而且重复性差。因此在计算机里用叫作部分积右移的方法来实现。

例 1.16 被乘数 $X=1011$，乘数 $Y=1101$，部分积 Z

（1）乘数末位 Y_0 为 1，所以在 Z 的初值上加被乘数，得新的部分积，部分积和 Y 右移一位。

（2）乘数末位 Y_0 已是 Y_1 的值为 0，所以给部分积加 0，得新的部分积，部分积和 Y 右移一位。

（3）乘数末位 Y_0 已是 Y_2 的值为 1，所以给部分积加被乘数，得新的部分积，部分积和 Y 右移一位。

（4）乘数末位 Y_0 已是 Y_3 的值为 1，所以给部分积加被乘数，得新的部分积，部分积和 Y 右移一位。

部分积 Z

```
   0000
+)1011
------
   1011
→ 0101      1
+)0000
------
   0101     1
→ 0010      1  1
+)1011
------
   1101     1  1
→ 0110      1  1  1
+)1011
------
  10001     1  1  1
→ 1000      1  1  1  1
```

可以看出，4 位数乘 4 位数变成了 4 次相加，4 次移位，每次相加都是两个 4 位数相加。如图 1-1 所示为它的流程；如图 1-2 所示则为它的工作原理示意图。

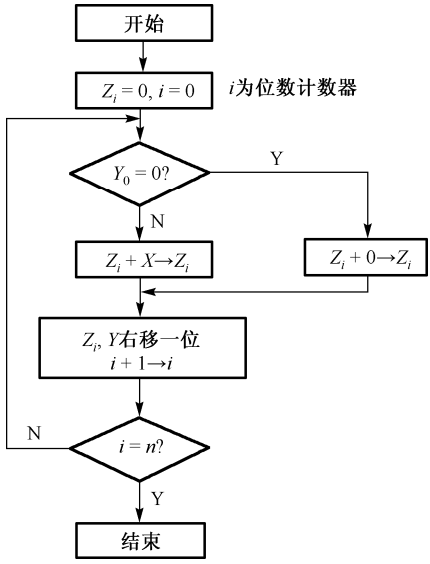


图 1-1 二进制数乘法流程图

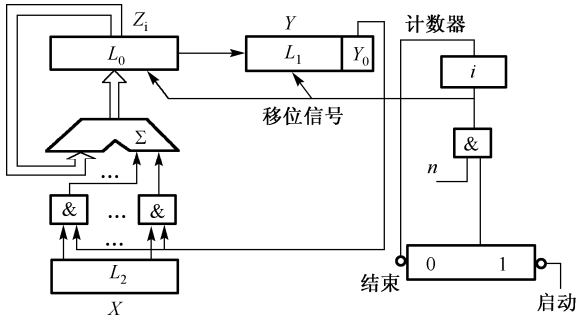


图 1-2　二进制数乘法原理示意图

1.2.4　计算机中带符号数的表示方法

数有带符号数和不带符号数之分，如人口普查中男性公民数，女性公民数，此类数据就为不带符号数，因为它不可能出现负数。而表示温度的数据就有正负之分，等等。

在计算机中表示的数就叫机器数，其含义如下：其一，由于计算机的字长有个定值，如 16 位机字长为 16 位，32 位机字长为 32 位，8 位机字长就为 8 位；也就是说，对某一个机器来说，它的一个字所能表示数的范围是有限制的。其二，对于不带符号数，机器数的各位都是数的有效值；而对带符号数，机器数最高位为 0 表示正数，为 1 表示负数。所以，对于带符号数来说，数的符号位也数字化了；也就是说，带符号数据的机器数由符号位和数值两部分组成。

带符号数的机器数最常用的是原码、反码和补码三种形式（下面均以 8 位机器数来说明，16 位机器数和 32 位机器数同理类推）。

1．原码表示法

数的最高位表示数的符号，数值部分是数的绝对值，也称真值，这种表示法称原码表示法。

（1）对于正数

例 1.17　$x=+5=+101\text{B}$

$$[x]_{原}=00000101\text{B}$$

（2）对于负数

例 1.18　$y=-75=-1001011\text{B}$

$$[y]_{原}=11001011\text{B}$$

2．反码表示法

数的最高位表示数的符号，数值部分对于正数同真值，对于负数是真值各位取反，这种表示法就叫反码表示法。

（1）对于正数

例 1.19　$x=+5=+101\text{B}$

$$[x]_{反}=00000101\text{B}$$

（2）对于负数

例 1.20　$y=-75=-1001011\text{B}$

$$[y]_{反}=10110100\text{B}$$

3．补码表示法

数的最高位表示数的符号，数值部分对于正数同真值，对于负数是该数反码数值部分末位加 1，这种表示法就叫补码表示法。

（1）对于正数

例 1.21 $x = +5 = +101\text{B}$

$$[x]_{补} = 00000101\text{B}$$

（2）对于负数

例 1.22 $y = -75 = -1001011\text{B}$

$$[y]_{补} = 10110101\text{B}$$

4．3 种带符号数机器数表示法比较

（1）对于正数，3 种形式相同。

（2）对于负数，符号位都为 1，数值部分：

原码——同真值；

反码——真值按位取反；

补码——反码末位加 1。

（3）数字 0 的表示。

原码有两种形式：

$[+0]_{原} = 00000000\text{B}$

$[-0]_{原} = 10000000\text{B}$

反码有两种形式：

$[+0]_{反} = 00000000\text{B}$

$[-0]_{反} = 11111111\text{B}$

补码只有一种形式：

$[+0]_{补} = 00000000\text{B}$

$[-0]_{补} = 00000000\text{B}$

（4）数的表示范围。

一个机器字对于 8 位机：

原码$+(2^7-1)\sim-(2^7-1)$

$+127\sim-127$

反码$+(2^7-1)\sim-(2^7-1)$

$+127\sim-127$

补码$+(2^7-1)\sim-2^7$

$+127\sim-128$

对于 16 位机和 32 位机可以类推。

（5）机器数是二进制数，由于符号位占据一位，因此有符号的数的形式值不等于真正的数值。特别对于负数的表示形式，原码形式最高位的 1 表示负号，不是数，数值部分是数的真正值；而反码和补码就连数值部分也不是数本身了。因此，要计算一个机器数为十进制的多少时，只有负数的原码的数值部分才可展开按权相加。

注意，二进制数不是机器数，一个是符号位，一个是数的位数的问题。

5．补码运算

补码表示的机器数其符号位能和有效数值部分一起参加数值计算，从而简化运算规则，节省运算时间；更重要的是使减法运算转化为加法运算，简化了运算器的线路设计。

例如，一块钟表指针指向 9 点，要把它拨向 2 点，可以逆时针倒拨 7 个格，也可顺时针顺拨 5 个格，即：

倒拨：9–7 = 2

顺拨：9+5 = 14 = 2(mod l2)

也就是说，对于以 12 为模的表来说，9–7 等效于 9+5，这样就把减法变成加法了。那么，对于以 12 为模的计数系统中，+5 就是–7 的补码，–4 的补码就是+8，–3 的补码就是+9，等等。当然，它们是互为补码的，如-9 的补码是+3。也就是说，它们的绝对值相加等于模。

对于 n 位机器数，符号位为 1 位，数值部分为 n–1 位，模为 2^n。对于 8 位机器数，其模为 256。

例 1.23　计算 x = 64–10。

若用 8 位机器数来计算，要把减 10 变成加(-10)的补码，同样可得到正确的结果。

$[-10]_{补}$=256–10=246=11110110B

$[x]_{补}=[64]_{补}-[10]_{补}=[64]_{补}+[-10]_{补}$

$[64]_{补}=$　01000000B

$[-10]_{补}=$　11110110B

+

1 00110110B

↓

丢失

即$[x]_{补}$ = 00110110B

符号位为 0（最高位），说明 x 是正数，所以按权展开相加得 x = 54。

结果是正确的。但是，读者会发现，为了把减法变成加法运算才这样做，而在求$[-10]_{补}$的时候，进行 256–10 还不是做减法吗?这不是徒劳吗?关键就是不执行 256–10 能不能得到$[-10]_{补}$的问题。

我们知道，$[+10]_{补}$=00001010B。

因为正数补码的数值部分同真值，很容易得到。那么，$[-10]_{补}$就是从$[+10]_{补}$的最右端寻找第一个“1”，找到之后，该“1”不变，它的右端各位数位均为 0，而左端（包括符号位）按位取反，即得到$[-10]_{补}$= 11110110B。

如$[y]_{补}$ = 01010000B

$[-y]_{补}$ = 10110000B

$[z]_{补}$ = 10010100B

$[-z]_{补}$ = 01101100B

读者不妨把 y 和 z 值计算出来验证它们是否正确。

前面提到，计算机中把除法运算变成部分余数左移加除数补码或 0（要看够减不够减），这里就不再详述了。总之，算术四则运算在计算机内最终都变成加法运算，所以在运算器中只有加法器，简化了设计，这也是二进制数特性带来的好处。

补码运算的溢出问题：异号数相加，同号数相减（变成加补码）不会溢出；同号数相加，异号数相减（变成加补码），可能溢出。

（1）上溢。正数+正数，其符号位变为 1，即成为负数。

例 1.24

$$\begin{array}{rr} [x]_{补} = 01100010\text{B} & 98 \\ [y]_{补} = 01001001\text{B} & +73 \\ \hline 10101011\text{B} & 171 \end{array}$$

因为 8 位机器数，正数补码值最大为 +127，+171 超过了所能表示（一个机器字 ）的范围。

（2）下溢。负数+负数还应为负数，即符号位应为 1；若变成 0，即变成正数。

例 1.25

$$\begin{array}{rr} [x]_{补} = 10100010\text{B} & -94 \\ [y]_{补} = 11001001\text{B} & +-55 \\ \hline ①01101011\text{B} & -149 \end{array}$$

因为 8 位机器数，负数补码最小为–128，–149 超过它能表示的范围，所以溢出。

补码运算符号位也像数一样参与运算，这是补码运算的特征。

可采用双高位法判断结果是否溢出。具体方法为用 C_S 表示符号位的进位情况；C_P 表示最高数值位的进位情况。当有进位时 C_S 或 C_P 为 1，否则为 0；溢出判别式 P 对二者进行异或运算，即 $P = C_S \oplus C_P$，当其为 1 时，表示溢出，当其为 0 时，表示不溢出。

例 1.26 已知$[X]_{原}$=11101011B，$[Y]_{原}$=01001010B，求$[X+Y]_{补}$和$[X–Y]_{补}$，并判断结果是否溢出。

分析：本题给出的已知条件是 X 和 Y 的原码形式，根据补码运算规则$[X+Y]_{补}=[X]_{补}+[Y]_{补}$，$[X–Y]_{补}=[X]_{补}+[–Y]_{补}$，所以必须先求出$[X]_{补}$、$[Y]_{补}$和$[–Y]_{补}$。

解：由于正数的补码形式和原码形式一致，负数的补码形式是符号位为 1，数值部分是真值按位求反加 1。所以

$[X]_{补}$= 10010101B　　$[Y]_{补}$= $[Y]_{原}$= 01001010B

在求$[–Y]_{补}$时，只要对其相反数的补码连同符号位一起求反加 1 即可，即$[–Y]_{补}$=10110110B。那么

$[X+Y]_{补} = [X]_{补} + [Y]_{补}$ = 10010101B+01001010B=11011111B

$[X–Y]_{补} = [X]_{补} + [–Y]_{补}$ = 10010101B+10110110B=01001011B

可采用双高位法判断结果是否溢出。

本例中，

$$\begin{array}{rr} 1\ 0\ 0\ 1\ 0\ 1\ 0\ 1\ \text{B} & [X]_{补} \\ +\ 1\ 0\ 0\ 1\ 0\ 1\ 0\ 1\ \text{B} & [Y]_{补} \\ \hline 1\ 1\ 0\ 1\ 1\ 1\ 1\ 1\ \text{B} & \end{array}$$

$C_S = 0$，$C_P = 0$，$P = C_S \oplus C_P = 0$，无溢出。

因此$[X+Y]_{补} = [X]_{补} + [Y]_{补}$ = 11011111B 无溢出，结果正确。

$$\begin{array}{rr} 1\ 0\ 0\ 1\ 0\ 1\ 0\ 1\ \text{B} & [X]_{补} \\ +\ 1\ 0\ 1\ 1\ 0\ 1\ 1\ 0\ \text{B} & [-Y]_{补} \\ \hline 0\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ \text{B} & \end{array}$$

$C_S = 1$ $C_P = 0$，$P = C_S \oplus C_P = 1$，有溢出。

所以$[X–Y]_{补}$= $[X]_{补}$+ $[–Y]_{补}$ = 01001011B 有溢出，结果出错。

1.2.5 计算机中数的小数点表示方法

一个二进制数 1010.01 可表示为：

$$1010.01=2^4\times 0.101001\text{B}$$

任意一个二进制数 N 的通式为：$N=2^j\times S\text{B}$

式中：j 为二进制整数的位数，称其为数 N 的阶码；S 为二进制小数，称其为数 N 的尾数。

尾数 S 表示数 N 的全部有效数字，而阶码 j 指明了小数点的位置。

对于任何一个数，若阶码 j 总是固定不变的，则把这种表示法称为数的定点表示，这样的数称为定点数，采用这种表示法的机器叫作定点计算机。如果阶码 j 可以取不同的值，则把这种表示称为数的浮点表示，这样的数称为浮点数，采用这种表示法的机器叫作浮点计算机。

若定点计算机的阶码 $j=0$，则该定点数只能是小数，其表示的格式为:

小数点的位置在符号位与尾数部分最高位之间。

按上述约定，在定点计算机中参与运算的数的绝对值必须小于 1。通常，在实际需要计算的数中，不一定都是小数。为了在定点计算机中用这些数进行运算。可以乘上一个比例因子，使其变成小数。例如，258.98 乘上一个 10^{-3} 的比例因子，即成为 0.258 98，当计算机输出后，再乘上比例因子 10^3 即恢复原来的值。

比例因子要选择恰当，不能选得太小，否则计算机运算结果仍可能超过 1，即计算机出现了溢出。若比例因子选得过大，小数点后几位都是 0，虽然不会溢出，但计算结果的精度将受到很大影响。

为了扩展数的表达范围，提高有效数字的位数，就要采用浮点表示法。在浮点计算机中，浮点数的表示由四部分组成：阶符、阶码、数符（尾数的符号）、数码（尾数）。其格式为:

若定点计算机的字长为 9 位，其中 8 个二进制数码位，一个符号位，则该计算机所能表示的数值范围（绝对值）是

$$0.00000001\text{—}0.11111111$$

或

$$2^{-8}\sim 1-2^{-8}$$

若计算机的字长是 n 位，则其所能表示的数值范围：

$$2^{-(n-1)}\sim 1-2^{-(n-1)}$$

对于浮点计算机，总是规定尾数部分的最高位是 1，即：

$$\frac{1}{2}\leqslant S\leqslant 1$$

若浮点计算机的字长为 13 位，阶符为 1 位，阶码为 3 位，数符为 1 位，数码为 8 位，则所能表示数的范围：

最小值：11110.10000000

最大值：01110.11111111

或

$$2^{-7}\times 2^{-1}\sim 2^7\times(1-2^{-8})$$

若阶码是 m 位，数码是 n 位，阶码和数码各有一位符号位，则所能表示数的范围为：

$$2^{-(2^m-1)}\times 2^{-1}\sim 2^{(2^m-1)}\times(1-2^{-n})$$

应当注意，浮点数的正负号是由尾数的正负号决定的，而阶码的正负号只决定小数点的位置，即决定浮点数的绝对值大小。

因为浮点计算机中数的小数点是浮动的，故一般的数就可以不用比例因子了。

浮点计算机通用性强，但结构复杂；定点计算机结构简单，但使用不太方便。

通常，微型计算机多为定点计算机（有的附加硬件电路也可进行浮点运算），而且又是整数，即小数点固定在数的末尾。因此，以下的讨论仅限于定点数。

1.2.6 二进制编码

在计算机中，所有信息都是采用二进制代码，因而计算机中表示的数、字母、符号等都要以特定的二进制代码来表示，即进行二进制编码。虽然它们也都是以 0，1 形式出现的，但它们是非数值数据，不是真正的二进制数。

1．二进制编码的十进制数

一位十进制数用 4 位二进制编码来表示，方法有很多，而较常用的是 8421BCD 码。它有 10 个不同的数字符号，但它是逢十进位的，所以实质上仍是十进制数，但由于它的每一位都是用 4 位二进制编码来表示的，因此称之为二进制编码的十进制数，简称为 BCD 码（Binary Coded Decimal），BCD 编码如表 1-2 所示。

表 1-2 BCD 编码

十进制数	8421 BCD 码	十进制数	8421 BCD 码
0	0000	10	0001 0000
1	0001	11	0001 0001
2	0010	12	0001 0010
3	0011	13	0001 0011
4	0100	14	0001 0100
5	0101	15	0001 0101
6	0110	16	0001 0110
7	0111	17	0001 0111
8	1000	18	0001 1000
9	1001	19	0001 1001

BCD 码是比较直观的，如十进制数 85.6 可方便地写成：

$$(10000101.0110)_{BCD}$$

但是 BCD 码与二进制之间的转换是不直接的，要先经过十进制，即先将 BCD 码转换为十进制数后，再转换为二进制数。反之亦然。

2．字母与其他字符的编码

在计算机应用中，各种字符也必须用二进制代码来编码，如 26 个英文字母、10 个阿拉伯数字、运算符号、标点符号以及一些特殊的控制符，如换行、换页、回车等。

在微型机中最常用的一种编码是 ASCII 码（American Standard Code for Information Interchange），即美国信息交换标准代码。它采用 7 位二进制代码来对一个字符进行编码，故可表示 128 个字符。因为微型机通常以字节表示信息单位为 8 位，所以最高位可用作奇偶校验位。不过在机器内部通常将此位视为零。故用一个字节（8 个二进制位称为一个字节，即一个 Byte）即可存放一个 ASCII 码。数字 0～9 的 ASCII 码为 30H～39H；大写英语字母 A～Z 的 ASCII 码为 41H～5AH；小写英文字母 a～z 的 ASCII 码为 61H～7AH（H 是指其前的两个数码为十六进制数码）。

3．汉字编码

汉字也是一种字符，但是汉字的计算机处理技术远比拼音文字复杂。一个汉字从输入设备输入到输出设备输出的过程，如图 1-3 所示。

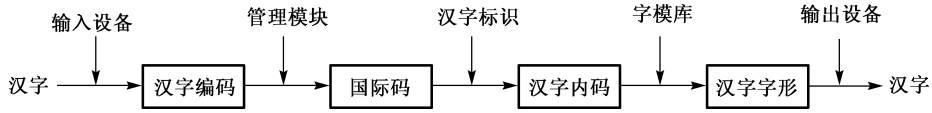


图 1-3　汉字编码

为了能在不同的汉字系统之间互相通信，共享汉字信息，有必要规定大家公认的中文信息处理标准。比如 1981 年我国制定推行的 GB2312—80 国家标准信息交换用汉字编码字符集（基本集），简称国标码。在国标码中，每个图形字符都规定了二进制表示的编码，每个编码字长为 2B，每字节内占用 7 位信息。

国标码不能作为在计算机内存储、运算的信息代码，这是因为它容易和 ASCII 码混淆，在中西文兼容时无法使用。汉字内部码是在国标码的基础上增加标识符的汉字代码。方法很简单，即在国标码的每个字节最高位加“1”，作为汉字标识符。

汉字内部码结构简单，一个汉字内部码只占 2B，足以表达数千汉字和各种图形，且又节省存储空间。另外，汉字内部码便于和西文字符兼容，这是因为在同一个计算机系统中，只要从最高位标识符就能区分西文的非扩充 ASCII 码（最高位为 0）和汉字内部码（最高位为 1）。

1.2.7　逻辑运算与基本逻辑电路

逻辑表示输入与输出的一种因果关系。一种用字母和符号代替文字来进行运算推理的方法称为逻辑代数或布尔代数或开关代数。它和普通代数一样，可以写成如下逻辑表达式：

$$Y = F(A, B, C, D)$$

式中：Y 为逻辑函数的值；F 为逻辑函数（逻辑关系）；A、B、C、D 为逻辑变量，数量可多可少。

逻辑运算具有以下特点：

（1）变量的取值只有两种可能，即 0 和 1。它并无大小之意，只代表事物的两个不同的性质，如命题的“真”和“假”，“有”和’无”，“开”和“关”等。函数值也只有两种可能，如“成立”和“不成立”，“发生”和“不发生”等。

（2）各种逻辑关系，用逻辑代数概括起来，只有 3 种基本关系：逻辑加法（“或”运算）、逻辑乘法（“与”运算）、逻辑否定（“反”运算或称“非”运算）。

能实现“与”、“或”、“非”等逻辑运算的电路称为逻辑门电路。通常，以高电平表示逻辑“1”，低电平表示逻辑“0”（称为正逻辑）。

1.“与”门

能实现“与”运算的电路称为“与”门。它有两个或更多个信号输入端和一个输出端。

两个输入端的“与”门电路的逻辑符号如图 1.4(a)所示。它完成逻辑“与”运算为：

$$Y = A \cdot B = A \wedge B = AB = A \cdot B$$

若输入 A、B 中，只要有一个为低电平，则输出 Y 为低电平，故有：

$$Y = A \cdot B = 1 \cdot 0 = 0 \cdot 1 = 0 \cdot 0 = 0$$

若输入 A、B 均为高电平时，输出才为高电平，即有：

$$Y = A \cdot B = 1 \cdot 1 = 1$$

2.“或”门

能实现“或”运算的电路称为“或”门。它有两个或更多的输入端和一个输出端。两个输入端的

“或”门逻辑符号如图 1.4(b)所示。它完成逻辑“或”运算为：

$$Y = A + B = A \vee B$$

若输入 A、B 均为低电平时，输出才为低电平，故有：

$$Y = A + B = 0 + 0 = 0$$

若输入 A、B 中，只要有一个为高电平，则输出为高电平，即有：

$$Y = A + B = 1 + 0 = 0 + 1 = 1 + 1 = 1$$

3. “非”门

实现“非”运算的电路称为“非”门，又称反相器。它只有一个输入端和一个输出端，即对单个逻辑值的处理。“非”门的逻辑符号如图 1.4(c)所示。它完成“非”运算为（运算符为在某个逻辑值上加一横“—”表示）：

$$Y = \overline{A}$$

若输入 A 为高电平，则输出为低电平：

$$Y = \overline{A} = \overline{1} = 0$$

若输入 A 为低电平，则输出为高电平：

$$Y = \overline{A} = \overline{0} = 1$$

4. “异或”门

实现“异或”运算的电路称为“异或”门。“异或”逻辑表达式为：

$$Y = A \oplus B = A\overline{B} + \overline{A}B$$

“异或”门只有两个输入端和一个输出端。其逻辑符号如图 1.4(d)所示。它完成的“异或”运算为：

$$Y = A \oplus B = 1 \oplus 0 = 0 \oplus 1 = 1$$

$$Y = A \oplus B = 1 \oplus 1 = 0 \oplus 0 = 0$$

5. “与非”门

任何复杂的逻辑电路都可以利用“与”、“或”、“非”等门电路组合而成。而“与非”门是由“与”门和“非”门组合而成。它有两个或更多的输入端和一个输出端。两个输入端的“与非”门逻辑符号如图 1.4(e)所示。它完成的逻辑运算为：

$$Y = \overline{A \cdot B}$$

若输入 A、B 均为高电平时，则输出为低电平：

$$Y = \overline{A \cdot B} = \overline{1 \cdot 1} = 0$$

若输入 A、B 中，只要其中有一个为低电平，则输出为高电平：

$$Y = \overline{A \cdot B} = \overline{1 \cdot 0} = \overline{0 \cdot 1} = \overline{0 \cdot 0} = 1$$

6. “或非”门

“或非”门是由“或”门和“非”门组合而成的。它有两个或更多的输入端和一个输出端。两个输入端“或非”门的逻辑符号如图 1.4(f)所示。它完成的逻辑运算为:

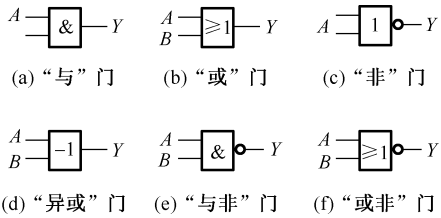
$$Y = \overline{A + B}$$

若输入 A、B 均为低电平时，则输出为高电平：

$$Y=\overline{A+B}=\overline{0+0}=1$$

若输入 A、B 中，只要其中一个为高电平，则输出为低电平：

$$Y=\overline{A+B}=\overline{1+0}=\overline{0+1}=\overline{1+1}=0$$



(a)“与”门　(b)“或”门　(c)“非”门

(d)“异或”门　(e)“与非”门　(f)“或非”门

图 1-4　基本逻辑电路符合

7．多位逻辑或运算规则（运算符为∨）

0∨0=0
0∨1=1
1∨0=1
1∨1=1
例如：

$$\begin{array}{r} 1011 \\ \vee)\ 1001 \\ \hline 1011 \end{array}$$

逻辑运算是在对应的两个二进制位之间进行的，与相邻的高低位的值无关，即不存在进位或借位问题。即按位相或。

8．多位逻辑与运算规则（运算符为∧）

0∧0=0
0∧1=0
1∧0=0
1∧1=1
例如：

$$\begin{array}{r} 1011 \\ \wedge)\ 1001 \\ \hline 1001 \end{array}$$

按位相与。

9．多位逻辑异或运算规则（运算符为∀）

$0\forall 0=0$
$0\forall 1=1$
$1\forall 0=1$
$1\forall 1=0$

例如：

$$\begin{array}{r} 1011 \\ \forall\ 1001 \\ \hline 0010 \end{array}$$

逻辑异或实现的是按位加功能，但不产生进位，只有参与异或操作的两个逻辑值不同时结果才为 1。

1.3 微型计算机的基本结构

1.3.1 微型计算机的总体结构

微型计算机系统是由计算机硬件系统、软件系统以及通信网络系统组成的一个整体系统。微型计算机硬件系统是指构成微型计算机的所有实体部件的集合，这些部件包括集成电路芯片、机械等物理部件，通常称为“硬件”。

微型计算机的硬件主要由输入设备、输出设备、运算器、存储器和控制器等五部分组成。这种结构早在 20 世纪 40 年代就由 Von Neumann 提出来了，所以现代绝大多数微型机的结构都称为 Von Neumann 结构，其基本组成如图 1-5 所示。

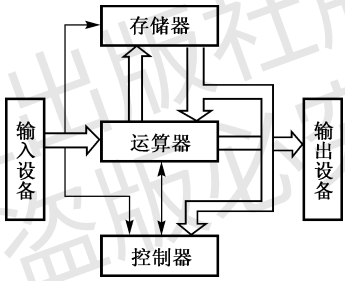


图 1-5 微型机的基本组成

1．微型机各组成部分的功能

（1）运算器

运算器是完成二进制数据的算术或逻辑运算的部件。运算器由累加器、暂存寄存器和算术逻辑单元（ALU）组成，核心是算术逻辑单元。累加器是一个位数和字长相同的寄存器，它既能接受来自总线的二进制信息作为参加运算的一个操作数，向算术逻辑单元 ALU 输送，又能存储总线送来的由 ALU 运算的结果，累加器与总线之间的数据传送是双向的。暂存寄存器也是和字长位数相同的寄存器，用于暂存由总线送来的另一个操作数。

（2）存储器

存储器是用来存放程序和数据的，是计算机各种信息的存储和交流中心。计算机中的存储器包括内存储器、外存储器和只读存储器等。内存储器又叫主存储器或随机存储器，平时也叫内存。相对主存储器而言，外存储器和只读存储器则称为辅助存储器。内存是按地址存取数据的，若地址总线共有 20 条（A_0～A_{19}），即有 20 个二进制位，则可形成 2^{20}=1 048 576 个地址，即 1 兆地址（1K=1024,1M=1024K）。

（3）控制器

控制器主要用来实现微型计算机本身运行过程的自动化，即实现程序的自动执行。在控制器控制

下，从输入设备输入程序和数据，并自动存放在存储器中，然后由控制器指挥各部件协同工作以执行程序，最后将结果打印输出。

（4）输入设备

输入设备用来输入原始数据和处理这些数据的程序，输入的信息包括数字符、字母和控制符号等，这些信息由译码电路产生相应的 ASCII 码再由控制器控制进行各种操作。输入设备主要是 CRT 终端和键盘。另外，磁性设备阅读机、光学阅读机、激光扫描仪等也都能作为输入设备。

（5）输出设备

输出设备用来输出计算机的处理结果，这些结果可以是数字、字母、图形和表格等。最常用的输出设备有打印机和绘图仪等，显示终端也可以视为一种输出设备，当数据和程序通过键盘输入终端后，经计算机处理后的结果可显示在屏幕上。

2．微型机各组成部分之间的连接方式

微型计算机的五个组成部分中，通常把运算器和控制器合起来，并且用大规模或超大规模集成电路技术集成在一块芯片上，称为中央处理单元（Central Processing Unit），简称 CPU。

总线是指一组并行的导线，数据和指令通过总线传送。总线分为地址总线 AB、数据总线 DB 和控制总线 CB。

微型计算机的三总线结构如图 1-6 所示。

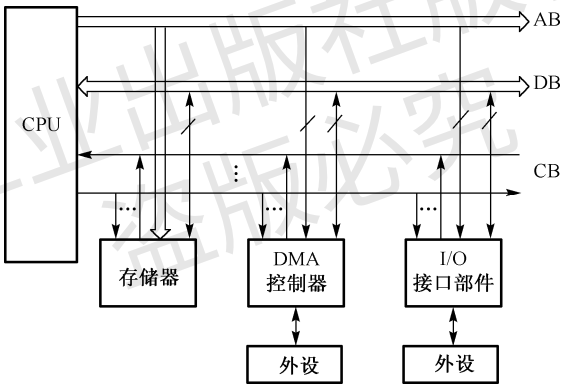


图 1-6　三总线结构

1.3.2　微处理器的基本结构

如图 1-7 所示为一种典型的微处理器逻辑结构图。图中左下部分包括累加器暂存器、ALU 和标志寄存器组成的算术逻辑部件，其他部分包括程序计数器、指令寄存器、指令译码器和操作控制器等组成了控制器。

一条指令的执行过程包括一系列微操作，指令译码器输出的控制电位和来自节拍发生器的节拍脉冲电位经过组合，就能产生执行指令所需要的按一定时间顺序出现的微操作信号。

一条指令在微处理器中执行的过程如下。

- 程序计数器 PC 指出当前指令地址，并且把指令地址放到地址总线上，然后 PC←PC+i（i 为当前指令长度的字节数）。为取下一条指令或数据做好准备。
- 由数据总线将指令从存储器中取出，送至指令寄存器，经指令译码，控制电路产生完成该指令的各种控制信号。
- 取出该指令所需的操作数地址和数据（它们可以从存储器、累加器或其他部件送来）。

- 完成该指令的操作，这些操作可以是一次内存读/写操作，一个算术/逻辑运算操作，或一个输入/输出操作。
- 检查有无其他控制信号（如中断请求信号等），并做出相应的处理。
- 提供指示处理器状态的标志信号、控制信号及定时信号等。

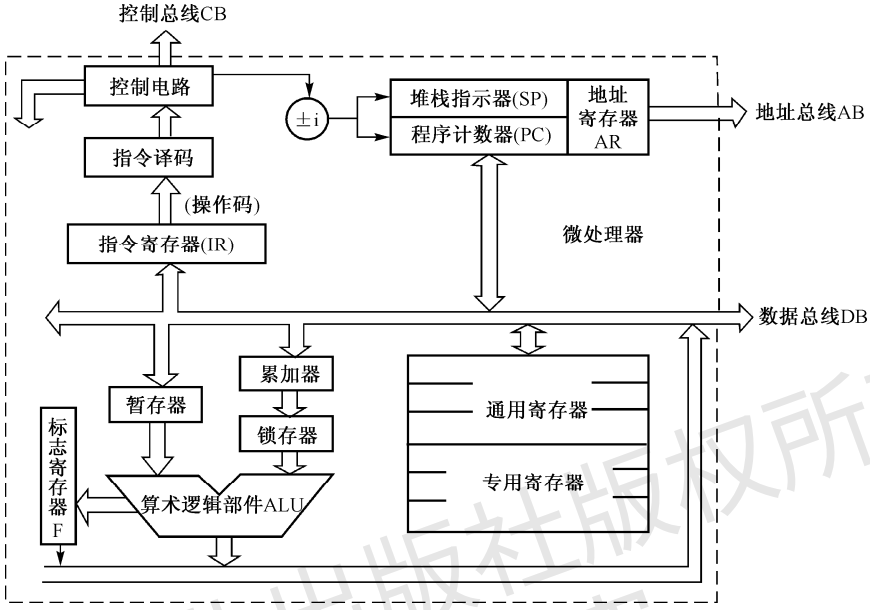


图 1-7　典型的微处理器逻辑结构图

1．算术逻辑部件

将微处理器中与完成算术逻辑操作有关的部件单独画出来，如图 1-8 所示。

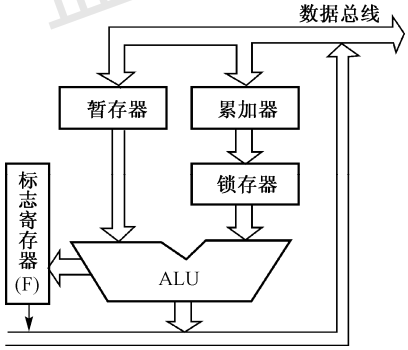


图 1-8　完成算术逻辑操作有关的部件

算术逻辑部件的主要功能是：

- 执行各种算术运算，给出相应的标志位；
- 执行各种逻辑运算，并进行逻辑测试。

通常，一个算术操作产生一个运算结果，而一个逻辑操作则产生一个判决。当一条指令对数据进行操作时，被操作的数据可能是一个（如移位、传送指令），也可能是两个（如加、减指令）。如果是前者，那么操作数一般在累加器中；如果是后者，则一个数在累加器中，另一个数在寄存器或存储器中，不过在操作前该数一定要传送到暂存器中保存。

暂存器和累加器、锁存器的设立，是为了适应微处理器内部采用的单总线结构。由于 ALU 没有寄存器功能，每次操作结果立即送到内部总线上从而占用了内部总线，所以当需要对两个数进行操作时，必须把第二个操作数送到暂存器。操作前，累加器存放一个操作数；操作后，累加器保存操作结果。为防止 ALU 的输出通过累加器反馈到 ALU 的输入端，设置了锁存器。

标志寄存器用来保存由算术指令和逻辑指令运行或测试结果建立的各种状态码内容，如运算结果进位标志（C）；运算结果溢出标志（O）；运算结果为零标志（Z）；运算结果的奇偶性标志（P），即运算结果中“1”的个数的奇偶性；等等。这些标志位分别由 1 位触发器保存，它们可作为外界对操作结果进行分析的依据，也可用于判断程序是否要转移的条件。标志寄存器也称为状态寄存器。

2．主要寄存器

微处理器中包含各种专用、通用寄存器。微处理器内部由于有了这些寄存器，程序就不必频繁地访问存储器，也减少了指令长度，这样就可以补偿单总线结构造成的速度降低的损失。有些微处理器还利用寄存器存放从存储器预取的 6 个指令字节，并排列于 6 个寄存器中，这种技术以前只在大型机中才采用。由于采用重叠操作技术，所以缩短了指令执行时间。

微处理器中的主要寄存器除上面提到的标志寄存器外，还有程序计数器、指令寄存器、地址寄存器、通用寄存器、累加器、堆栈指针等。

（1）程序计数器（PC）。PC 用来存放要执行的指令的存储地址，所以 PC 又称为指令计数器。当执行指令时，CPU 将自动修改 PC 的内容。由于大多数指令都是按顺序执行的，所以修改的过程通常只是简单地将 PC 的内容加 1。值得注意的是，在微处理器中，存储器操作是以字节为单位的，而指令长度是可变字节。对多字节指令，PC 必须加上该指令字节数后，才能准确地指向下一条指令的地址。

如果遇到转移指令如 JMP（跳转）指令时，PC 的内容必须从指令寄存器中的地址段取得，在这种情况下，下一条从内存取出的指令将由转移指令来规定，而不是像通常一样按顺序来取得的。

（2）指令寄存器（IR）。指令寄存器用来保存当前正在执行的一条指令。指令划分为操作码和地址码字段，指令寄存器中操作码字段的输出将作为指令译码器的输入，操作码一经译码后，即可向操作控制器发出具体操作的特定信号。

（3）地址寄存器（AR）。地址寄存器用来保存当前 CPU 所要访问的内存单元的地址。由于在内存和 CPU 之间存在着操作速度上的差别，所以必须用地址寄存器来保持地址信息，直到内存读/写操作完成为止。

（4）通用寄存器。通常这是一组寄存器，可由程序员指定为各种用途，有的微处理器（如 8086）中的少数指令的某些通用寄存器作为专用，这是为了缩短指令代码长度。另外，通用寄存器可以单独使用，有时也可以把两个寄存器连起来使用。例如，在以 8 位寄存器为主的 8 位微处理器中，两个寄存器可以合起来作为一个 16 位寄存器使用，而在 16 位微处理器中，两个寄存器合起来可以作为一个 32 位寄存器使用，这就方便了许多数据操作。

（5）累加器（AC）。累加器的功能是为算术逻辑部件提供一个工作区，它可以保存一个操作数，当运算完毕，比如与另一个操作数（它可以放在暂存器中）相加，其结果送回累加器，而累加器中原有的内容立即被破坏，累加器的含义也由此而出。在累加器中的数据还可以在指令的控制下实现算术左、右移，循环左、右移等。累加器是所有寄存器中最忙碌的一个。

3．堆栈和堆栈指针（SP）

堆栈是学习微处理器时必须掌握的一个重要概念。所谓堆栈是指按照后进先出，先进后出的原则组织的一组寄存器或指定的一组存储单元。前者称硬堆栈，后者称软堆栈。微处理器对堆栈的操作是

由压入（PUSH）和弹出（POP）指令实现的。将数据压入堆栈和从堆栈中弹出数据都只能在栈顶进行。栈顶的地址由堆栈指针给定，并自动进行管理，即对地址指针的自动修改操作。

堆栈的实现可分硬件堆栈和软件堆栈两种方法。

硬件堆栈是由一组位于微处理器芯片内部的寄存器组成，其优点是速度高，但栈的深度有限。当寄存器组满载时，堆栈就溢出来了。这种硬件堆栈只适用于功能较小的微处理器，如单片微型计算机等。

软件堆栈用系统中的随机存取存储器 RAM 实现。堆栈指针的初值可由程序员自行选定，然后由堆栈指针自动管理。由于使用系统 RAM 作为堆栈，加上 SP 可自由设定，所以可以在 RAM 任何区域设置“无限”容量的堆栈。目前，绝大多数微处理器均采用这种堆栈。

在内存中构造堆栈有不同的方法，堆栈指针 SP 所给定的可以是当前栈顶单元的地址，即该单元已存有数据，如图1-9(a)所示；也可以是一个紧接栈顶的“空”单元地址，即该单元尚未被使用，如图1-9(b)所示。这样，在栈操作时，对于前者，压入操作是先修改指针再压入数据，弹出操作是先弹出数据再修改指针；而对于后者，则是在压入时先压入数据再修改指针，弹出时先修改指针再弹出数据。

栈地址的增长也有两种方式。一种是栈底在存储器高地址区，栈单元地址由高向低增长，如图 1-9(a)所示；一种是栈底在存储器低地址区，栈单元地址由低向高增长，如图 1-9(b)所示。通常用图 1-9(a)的方式较多。8086CPU 堆栈采用的就是如图 1-9(a)所示的方式。

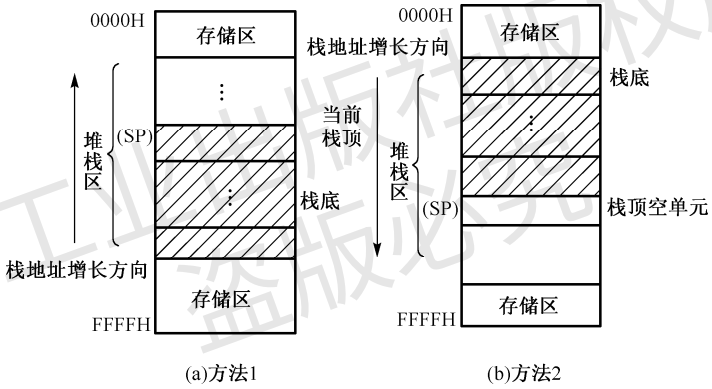


图 1-9　堆栈操作

堆栈主要用于中断和子程序调用，也用于数据的临时存取。在进入中断服务程序和调用子程序之前，都必须保存 CPU 中现行的信息；在中断服务结束返回主程序和子程序调用结束返回主程序之前，又必须恢复前面保存的信息，这些都可方便地由堆栈通过压入和弹出操作来实现。

堆栈的主要优点是，在向栈压入数据时不会影响已存入的数据，因为堆栈地址是在 SP 中，而不是在指令内。SP 的值可随压入或弹出而自动修改，并且栈操作指令可以很短。

堆栈的主要缺点在于使用堆栈的程序调试和文件编制都很困难。因为堆栈无固定地址，其单元内容很难记住或确定。堆栈用法的错误比较难找，典型的出错例子是从堆栈中取出数据的次序不对，堆栈的上溢或下溢使堆栈超出了约定的存储空间，而进入了其他数据区或程序区。

4．操作控制器及时序产生器

这部分器件实际上构成了微型计算机系统的控制器，它将根据指令操作码和时序信号，产生各种操作控制信号，以便正确地建立数据通路，从而完成取指令和执行指令的控制。

指令译码器首先根据指令的操作码译出各种控制信号。指令译码器是一组逻辑电路。例如，在如图 1-10 所示的译码电路中，当来自指令的高 4 位（I_7～I_4）二进制操作码是 0001 时，则经译码后只有

ADD 端为高电平“1”（参阅表 1-3 中的 ID 真值表），而其他几个输出保持低电平“0”，所以只要发现 ADD 输出变为高电平就可知道这是一条加法指令。指令译码器输出的这些电平就是控制电位。

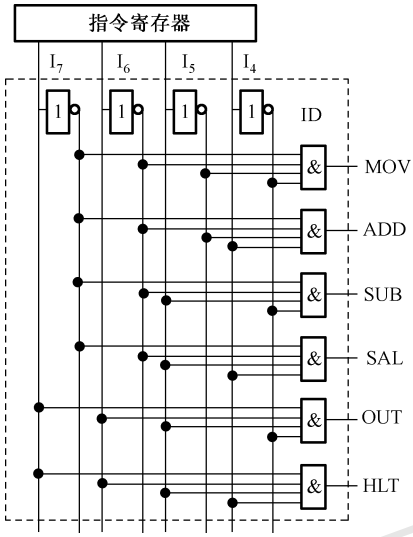


图 1-10　指令译码器 ID

表 1-3　ID 真值表

I_7	I_6	I_5	I_4	输出
0	0	0	0	MOV
0	0	0	1	ADD
0	0	1	0	SUB
0	0	1	1	SAL
1	1	1	0	OUT
1	1	1	1	HLT

时序产生器由时钟信号源、节拍发生器及微操作电路组成。时钟信号源不断地产生一定的频率、一定宽度的时钟信号送入节拍发生器，节拍发生器由触发器组成环形计数器，输出按顺序变化的脉冲电平，微操作电路将指令译码器输出的控制电位与来自节拍发生器的节拍脉冲进行组合，以产生指令所需要的按一定时间顺序出现的一系列微操作信号，将它们送往运算器、存储器等各个部件，用以完成该指令所规定的全部操作。

控制电路可由组合逻辑电路组成，也可由微程序控制逻辑组成。微程序控制是指将机器指令的操作分解为若干个更基本的微操作序列，并将有关的控制信息（微指令）以微码形式编成微指令，输入控制存储器中，这样，每条机器指令与一段微程序相对应，取出微指令就产生微命令，实现机器指令要求的信息传送和加工。现在大多数微处理器都用微程序控制技术。

控制器产生的控制信号包括运算器、输入/输出接口和存储器之间的同步信号，指令的读取、译码和执行信息，通过有关控制总线和外界进行通信的信号等。

1.4　多媒体计算机

媒体（Media）即是信息传播的载体，如图像（Image）、声音（Audio）、文字（Text）等。利用计算机来处理信息媒体已有了很长的历史，并形成了成熟的理论方法，如图像处理技术现已有较成熟的图像采集、压缩存储及多种处理理论。但是把多媒体，如视频（Video）、声音、图像、动画（Animation）、

文字结合在一起，进行同时处理却是近十几年的事，这便形成了一种新技术——多媒体（Multimedia）。通常所说的多媒体是指多媒体计算技术（Multimedia Computing），简称多媒体技术，其含义是利用计算机来综合、集成地处理文字、图形、图像、声音、视频、动画等媒体，而形成的一种全新的信息传播和处理技术。它把计算机技术、通信技术和广播、电视技术融为一体，综合利用，扩展了计算机应用的领域，受到了人们极大的关注和重视。

1.4.1 人机接口

人类接收和传播信息的两种主要方式是用“眼睛看”和用“耳朵听”，所以可看见的媒体（如文字、图形、图像、动画等）和可听见的媒体（如声音等）的完美结合才能完整、自然地表达和让人类最大限度地接收信息。具有多媒体功能的计算机就称为多媒体计算机，其中最广泛、最基本的是多媒体个人计算机（Multimedia Personal Computer，MPC）。一般普通的计算机只能处理可看见的媒体，多媒体技术的重点是要使计算机可听见信息，如声音。所以声音是多媒体最基本、最重要的要素。而同时具有视（动态）、听特性的媒体如视频、全活动景象（Full-Motion Movie）等，也是多媒体技术的重要发展方向。

1.4.2 多媒体计算机的主要功能

多媒体技术示意图如图 1-11 所示。

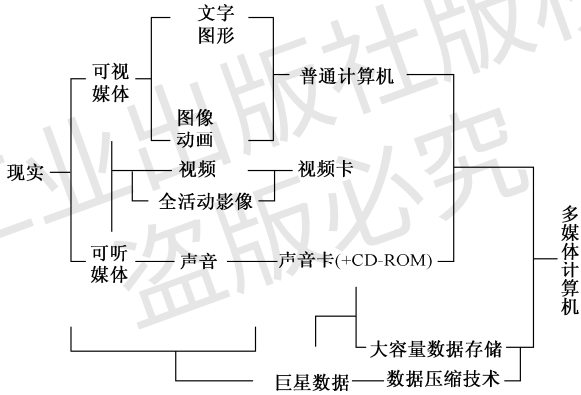


图 1-11 多媒体技术示意图

多媒体信息的处理都涉及海量的数据，所以大容量数据存储以及数据压缩和解压缩技术也是多媒体技术的重要方向。

多媒体系统应该具有如下特性。

（1）具备高度集成性，即能高度综合集成各种媒体信息，使处理各种媒体的设备相互协调工作。

（2）具有良好的交互性，即用户能随意地通过软件调度媒体数据和指挥媒体设备。

（3）具备完善的多媒体硬件和多媒体工作平台。

多媒体硬件是多媒体技术的基础，多媒体软件是多媒体技术的灵魂。前者为多媒体的实现提供了可能，后者综合地利用计算机处理各种媒体的最新技术，如数据压缩、数据采样、二维、三维动画等，能灵活地调度使用多媒体数据，使各种媒体硬件和谐工作。由于多媒体技术涉及的各种媒体都要求海量的数据，所以多媒体的主要任务是使用户方便、有效地组织和运转多媒体数据。

1.4.3 多媒体计算机的组成

多媒体计算机（MPC）是在 PC 的基础上融合高质量的图形、立体声、动画等媒体而组合的系统，其硬件结构如图 1-12 所示。

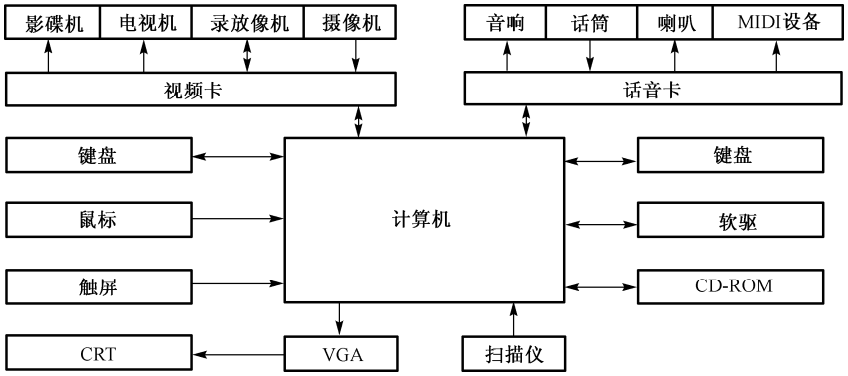


图 1-12　多媒体计算机的组成

总之，多媒体计算机是在通用的 PC 基础上，增加了多媒体设备和多媒体软件构成的。

习题与思考题

1．将下列十进制数转换成十六进制、八进制和二进制数。

128、241、511、372、1024、3000

2．将下列无符号二进制数分别转换成十进制数、八进制数和十六进制数。

1011001010B、11110100B

01101001B、100100100B

3．将下列十六进制数分别转换成十进制数和二进制数。

2ECH、325H、FFH、1ABH、FFFFH

4．将下列十进制数转换成 BCD 码。

46、121、731、2345

5．求下列各数以 100H 为模的补码。

–04H、–19H、–0FH、–2AH、–4BH

6．将下列二进制补码转换成十进制数。

10010110B、01101100B、00101010B

11101110B、10000001B、11000000B

7．完成下列 BCD 数的运算。

01100001–01010110、10011000–01111001

00100110+01101000、01000010+01010010

8．求 11010010 和 01001110 两数分别进行“与”，“或”，“异或”操作的运算结果。

9．典型的微处理器是由哪几部分组成的？其主要功能是什么?

10．什么叫堆栈？堆栈操作有几种方式？任选一种说明其操作过程。

11．简述 PC 和 SP 在微机中的作用。

12．什么是机器数?机器数与二进制数的区别是什么?

13．已知$[x]_{补}$=11000000B，$[y]_{补}$=01001000B，$[z]_{补}$=00110010B。求$[-x]_{补}$，$[-y]_{补}$，$[-z]_{补}$；并计算$[x-y]_{补}$，$[x-z]_{补}$，若有溢出给予说明。