全国高等职业教育“十二五”计算机类专业规划教材

微机原理与接口技术

王 盟 主 编
尹晓翠 李湘云 任丽鸿 副主编

中国电力出版社
CHINA ELECTRIC POWER PRESS

全国高等职业教育“十二五”计算机类专业规划教材

微机原理与接口技术

主　编　王　盟
副主编　尹晓翠　李湘云　任丽鸿
编　写　张艳华　徐　艳　张　霄　肖　梅
主　审　栾昌海

中国电力出版社
CHINA ELECTRIC POWER PRESS

内 容 提 要

本书以 Intel 8086/8088 系列微机为背景，分为汇编语言程序设计和接口技术两部分，主要内容包括微型计算机的基础知识、Intel 8086/8088 指令系统和汇编语言程序设计、8086/8088 的总线与时序、存储器及 I/O 接口、中断处理技术、DMA 技术、并行接口与定时/计数技术、串行通信接口、A/D 和 D/A 转换器等知识。每章后都附有习题和实训指导，便于开展实践性教学工作。

本书内容简明扼要、深入浅出、重点突出，并且配有大量的图示、实例，融入了作者多年教学经验。本书可作为各类本科院校、高职高专院校的微机原理与接口技术、微机原理与应用、汇编语言程序设计和微机接口技术等课程的通用教材，也可作为计算机等级考试、成人教育、在职人员培训、高等教育自学人员和从事微机硬件和软件开发的工程技术人员的参考书。

图书在版编目（CIP）数据

微机原理与接口技术 / 王盟主编. —北京：中国电力出版社，2014.12

全国高等职业教育“十二五”计算机类专业规划教材

ISBN 978-7-5123-6292-5

Ⅰ. ①微… Ⅱ. ①王… Ⅲ. ①微型计算机－理论－高等职业教育－教材 ②微型计算机－接口技术－高等职业教育－教材 Ⅳ. ①TP36

中国版本图书馆 CIP 数据核字（2014）第 181824 号

中国电力出版社出版、发行

（北京市东城区北京站西街 19 号 100005 http://www.cepp.sgcc.com.cn）

航远印刷有限公司印刷

各地新华书店经售

*

2014 年 12 月第一版 2014 年 12 月北京第一次印刷

787 毫米×1092 毫米 16 开本 16 印张 390 千字

定价 **32.00** 元

前　言

微机原理与接口技术是本科院校、高职高专院校理工类学生必修的专业基础课，该课程主要以 Intel 8086/8088 微处理器为背景，从应用角度系统阐述微机的基本原理，介绍 8086/8088 微处理器的结构、指令系统及汇编语言程序设计、半导体存储器、输入输出与中断技术、常用可编程 I/O 接口芯片的特点及使用技巧等。通过本课程的学习，使学生在掌握基础知识的同时，应具备分析和设计微机应用系统的能力，为后续知识的学习、研究奠定基础。

本书根据对学生的培养目标，内容组织遵循循序渐进的原则，从微机的基本概念出发，逐步介绍微机系统的工作原理、指令系统和汇编语言程序设计以及微机接口的设计、应用，并把微机系统软件和硬件技术有机结合起来。在介绍基本概念的同时，列举了大量典型而有意义的例题和习题；在加强基础知识的同时，着重使学生建立起微机系统的概念；在充分掌握外设接口原理的基础上，加强接口应用及设计能力的要求。本书内容简明扼要、深入浅出、重点突出，并且配有大量的图示、实例、融入了作者多年的教学经验及体会，特别注意在强化基本概念、基本方法和基本技能的同时，注重培养学生的动手能力。

本书可作为各类本科院校、高职高专院校的“微机原理与接口技术”、“微机原理与应用”、“汇编语言程序设计”和“微机接口技术”等课程的通用教材，也可作为计算机等级考试、成人教育、在职人员培训、高等教育自学人员和从事微机硬件和软件开发的工程技术人员的参考书。

本书共 7 章，教学参考课时为 60～80 学时。第 1 章介绍微型计算机的基础知识；第 2 章介绍 8086/8088 指令系统；第 3 章介绍汇编语言程序设计；第 4 章介绍 8086/8088 的总线与时序；第 5 章介绍半导体存储器；第 6 章介绍输入输出与接口技术，包括中断技术、DMA 技术等；第 7 章介绍常用可编程接口芯片，包括并行接口与定时/计数技术、串行通信接口、A/D 和 D/A 转换器等；另外，本书配有习题和实训指导，书中所有例题、习题和实训项目都在 80x86 系列微机系统上调试通过。

本书由东营职业学院的王盟、尹晓翠、李湘云和任丽鸿合作编写，张艳华、徐艳、张霄和肖梅也参与了本书部分章节的编写工作。由王盟担任主编，尹晓翠、李湘云和任丽鸿担任副主编。第 1 章由王盟、张艳华编写；第 2 章由尹晓翠、徐艳编写；第 3 章由王盟、张霄编

写；第4章由尹晓翠、任丽鸿编写；第5章由李湘云、肖梅编写；第6章由任丽鸿编写；第7章由王盟、李湘云编写；附录由王盟、李湘云整理。本书电子教案由尹晓翠老师制作完成。王盟负责全书内容的组织编写、修改和最终定稿统稿。

限于编者水平，书中难免有疏漏和不当之处，恳请广大专家和读者批评指正。

王　盟

2014年11月

目　　录

第1章 基 础 知 识

本章要点

（1）二、八、十、十六进制数及其相互转换。
（2）真值与机器数以及机器数的表示方法。
（3）8086/8088 微处理器的内部结构与寄存器组。
（4）8086/8088 的存储器结构及其 20 位物理地址的形成。
（5）8086/8088 堆栈操作的基本原理。

1.1 计算机中的数制与编码

通常，计算机中的数据分为两类。
（1）数：用来直接表示量的多少，有大小之分，能够进行加减等运算。
（2）码：通常指代码或编码，在计算机中用来描述某种信息。

1.1.1 数制及其转换

数制也称进位计数制，是指用一组固定的符号和统一的规则来表示数值的方法。按进位的原则进行计数的方法称为进位计数制。例如，在十进制中，是按照“逢十进一”的原则进行计数的。

思考

那么如果是五进制，则应该如何计数呢？同理，按照“逢五进一”的原则，如：1、2、3、4、10、11、…、14、20、…、43、44、100、…

常用进位计数制主要包括十进制（Decimal notation）、二进制（Binary notation）、八进制（Octal notation）、十六进制（Hexadecimal notation），其中八进制、十六进制主要是用来简化二进制的描述。

一、基数与位权

（1）基数：进位计数制的每位数上可能有的数码的个数。例如，十进制数每位上的数码，有"0"、"1"、"3",…，"9"十个数码，所以基数为 10。

（2）位权：一个数值的每一位上数字权值的大小。例如，十进制数 1234 从低位到高位的位权分别为 10^0、10^1、10^2、10^3。因为

$$1234=1\times10^3+2\times10^2+3\times10^1+4\times10^0$$

（3）数的位权表示：任何一种数制的数都可以表示成按位权展开的多项式之和。

例如：十进制数的 123.45 可表示为

$$123.45=1\times10^{2}+2\times10^{1}+3\times10^{0}+4\times10^{-1}+5\times10^{-2}$$

二、二进制数、八进制数和十六进制数

1. 二进制数

日常生活中一般采用十进制进行计数，但计算机只能识别 0、1 代码，也就是说计算机采用二进制进行计数。二进制数只有 0、1 两个数码，其基数为 2，遵循逢二进一的原则，其第 K 位权以 2^K 表示。二进制数的描述是在其尾部加注字母 B。

例如：11011B 或$(11011)_2$都表示一个二进制数。

2. 八进制数

八进制数有 0、1、2、3、4、5、6、7 共 8 个数码，其基数为 8，遵循逢八进一的原则，第 K 位权以 8^K 表示。八进制数的描述是在其尾部加注字母 O。

例如：123O 或$(123)_8$都表示一个八进制数。

3. 十六进制数

十六进制数有 0、1、2、3、4、5、6、7、8、9、A、B、C、D、E、F 共 16 个数码，其中 A、B、C、D、E、F 表示 10～15 共 6 个数码，其基数为 16，遵循逢十六进一的原则，第 K 位权以 16^K 表示。十六进制数的描述是在其尾部加注字母 H。

例如：12ABH 或$(12AB)_{16}$都表示一个十六进制数。

4. 常用计数制间的对应关系见表 1.1。

表 1.1　　常用计数制间的对应关系

十进制	二进制	八进制	十六进制
0	0	0	0
1	1	1	1
2	10	2	2
4	100	4	4
8	1000	10	8
10	1010	12	A
15	1111	17	F
16	10000	20	10

三、二、八、十六进制和十进制数之间的转换

1. 二、八、十六进制转十进制的方法——乘权相加法

乘权相加法：各位非十进制数码乘以与其对应的权之和即为该数对应的十进制数。

例如：

$$(123)_{16}=1\times16^{2}+2\times16^{1}+3\times16^{0}=(336)_{10}$$

$$(110.011)_2=1\times2^{2}+1\times2^{1}+0\times2^{0}+0\times2^{-1}+1\times2^{-2}+1\times2^{-3}=(6.375)_{10}$$

2. 二进制数和八进制数互换

二进制数转换成八进制数时，只要从小数点位置开始，向左或向右每 3 位二进制数划分为一组（不足 3 位时可补 0），然后写出每一组二进制数所对应的八进制数码即可。

例如：将二进制数 11001011.01B 转换成八进制数：

011 001 011. 010=313.2O

同理，将每位八进制数分别用 3 位二进制数表示，就可完成八进制数和二进制数的转换。

3. 二进制数和十六进制数互换

二进制数转换成十六进制数时，只要从小数点位置开始，向左或向右每 4 位二进制数划分为一组（不足 4 位时可补 0），然后写出每一组二进制数所对应的十六进制数码即可。

例如：将二进制数 100101010.101B 转换成十六进制数。

0001 0010 1010. 1010=12A.AH

同理，将每位十六进制数分别用 4 位二进制数表示，就可完成十六进制数和二进制数的转换。

四、十进制数转换为二进制数、八进制数、十六进制数

方法：基数乘除法。即十进制数转换成二进制数、八进制数、十六进制数时，应对整数和小数分别进行处理。整数转换采用“除基取余”的方法，小数转换采用“乘基取整”的方法。下面以十进制数转换为二进制数为例进行介绍。

1. 整数转换

将十进制整数 N 除以 2，取余数计为 K0；再将所得商除以 2，取余数记为 K1；……依次类推，直至商为 0，取余数计为 Kn-1 为止。即可得到与十进制整数 N 对应的 n 位二进制整数 Kn－1…K1K0。

例如：将十进制整数$(105)_{10}$转换为二进制整数，采用“除 2 倒取余”的方法，过程如下：

```
2 | 105
  2 | 52    余数为1
    2 | 26  余数为0
      2 | 13  余数为0
        2 | 6  余数为1
          2 | 3  余数为0
            2 | 1  余数为1
              0    余数为1
```

所以，$(105)_{10}=(1101001)_2$。

2. 小数转换

将十进制小数 N 乘以 2，取积的整数部分记为 K－1；再将积的小数部分乘以 2，取整数部分记为 K－2；依次类推，直至其小数部分为 0 或达到规定精度要求，取整数部分记作 K－m 为止。即可得到与 N 对应的 m 位二进制小数 0.K－1K－2…K－m。

例如：将十进制小数$(0.8125)_{10}$转换为二进制小数，采用“乘 2 顺取整”的方法，过程如下：

0.8125×2=1.625　　取整数位 1
0.625×2=1.25　　取整数位 1
0.25×2=0.5　　取整数位 0
0.5×2=1.0　　取整数位 1

所以，$(0.8125)_{10}=(0.1101)_2$。

注意

当十进制小数不能用有限位二进制小数精确表示时，可根据精度要求，求出相应的二进制位数近似地表示。一般当要求二进制数取 m 位小数时，可求出 $m+1$ 位，然后对最低位作“0 舍 1 入”处理。例如，将十进制小数 0.456 转换成二进制小数（保留 4 位小数）为 0.0111B。

若一个十进制数既包含整数部分，又包含小数部分，则需将整数部分和小数部分分别转换，然后用小数点将两部分结果连到一起。例如，将十进制数 12.345 转换成二进制数为 1100.01011B（12D=1100B；0.345D=0.01011B）。

1.1.2 计算机中数的表示方法

一、机器数与真值

1. 机器数

数学中正数与负数是用该数的绝对值加上正、负符号来表示。由于计算机中无论是数值还是数的符号，都只能用 0 和 1 来表示。所以计算机中，为了表示正、负数，把一个数的最高位作为符号位：0 表示正数，1 表示负数。例如，如果用 8 个二进制位表示一个十进制数，则正的 58 和负的 58 可表示为

+58→00111010

−58→10111010

这种连同符号位一起数字化了的数称为机器数。

2. 真值

由机器数所表示的实际值称为真值。

例如：

机器数 00101011 的真值为十进制的+43 或二进制的+0101011。

机器数 10101011 的真值为十进制的−43 或二进制的-0101011。

二、机器数的表示方法

提示

以下定点整数的原码、反码和补码的表示均以字长 $n=8$ 为例。

1. 原码

原码表示法是一种比较直观的表示方法，其符号位表示该数的符号，正用“0”表示，负用“1”表示；而数值部分仍保留着其真值的特征。

若定点小数的原码形式为 $X_0X_1X_2\cdots X_N$ ，则原码表示的定义：

当 $1>X\geqslant 0$ 时，$[X]_{原}=X$；

当 $0\geqslant X>-1$ 时，$[X]_{原}=1-X=1+|X|$。

其中，$[X]_{原}$是机器数，X 是真值。

例如，$X=+0.1001$，则$[X]_{原}=0.1001$

$X=-0.1001$，则$[X]_{原}=1.1001$

若定点整数的原码形式为 $X_0X_1X_2\cdots X_N$，则原码表示的定义：

当 $2^N>X\geqslant 0$ 时，$[X]_{原}=X$；

当 $0\geqslant X>-2^N$ 时，$[X]_{原}=2^N-X=2^N+|X|$。

例如，$X=+81$，则$[X]_{原}=0\ 1010001$

$X=-81$，则$[X]_{原}=1\ 1010001$

原码表示法有两个特点：

（1）零的表示有“+0”和“-0”之分，故有两种形式：

$$[+0]_{原}=0.000\cdots 0$$

$$[-0]_{原}=1.000\cdots 0$$

（2）原码表示的整数范围：$-(2^{n-1}-1)\sim+(2^{n-1}-1)$，其中 n 为机器字长。则 8 位二进制原码表示的整数范围是$-127\sim+127$；16 位二进制原码表示的整数范围是$-32\ 767\sim+32\ 767$。

原码表示法的优点是比较直观、简单易懂，但其最大的缺点是加法运算复杂。这是因为，当两数相加时，如果是同号则数值相加；如果是异号，则要进行减法。而在进行减法时，还要比较绝对值的大小，然后减去小数，最后还要给结果选择恰当的符号。显然，利用原码作加减法运算是不太方便的。为了解决这些矛盾，人们找到了补码表示法。

2. 反码

反码表示法中，符号的表示法与原码相同。正数的反码与正数的原码形式相同；负数的反码符号位为 1，数值部分通过将负数原码的数值部分各位取反（0 变 1，1 变 0）得到。

若定点小数的反码形式为 $X_0X_1X_2\cdots X_N$，则反码表示的定义：

当 $1>X\geqslant 0$ 时，$[X]_{反}=X$；

当 $0\geqslant X>-1$ 时，$[X]_{反}=(2-2^{-N})+X$

例如，$X=+0.1001$，则$[X]_{反}=0.1001$

$X=-0.1001$，则$[X]_{反}=1.0110$

对于 0，在反码的情况下只有两种表示形式，即

$$[+0]_{反}=0.000\cdots 0$$

$$[-0]_{反}=1.111\cdots 1$$

对于定点整数 $X_0X_1X_2\cdots X_N$，则反码表示的定义：

当 $2^N>X\geqslant 0$ 时，$[X]_{反}=X$；

当 $0\geqslant X>-2^N$ 时，$[X]_{反}=(2^{N+1}-1)+X$。

例如，$X=+81$，则$[X]_{反}=0\ 1010001$

$X=-81$，则$[X]_{反}=1\ 0101110$

反码表示的整数范围与原码相同。

3. 补码

由于计算机的运算受一定字长的限制，属于有模运算，所以，在计算机中可以使用补码进行计算。在定点小数机中数最大不超过 1，也就是负的小数对“1”的补码是等价的。但实际上，负数的符号位还有一个“1”，要把它看成数的一部分，所以要对 2 求补码，也就是以 2 为模数。正数的补码与其原码相同，负数的补码为其反码在最低位加 1。

若定点小数的补码形式为 $X_0.X_1X_2\cdots X_N$，则补码表示的定义：

当 $1>X\geqslant 0$ 时，$[X]_{补}=X$；

当 $0\geqslant X\geqslant -1$ 时，$[X]_{补}=2+X=2-|X|$。

例如，X=+0.1001，则$[X]_{补}$=0.1001

X=–0.1001，则$[X]_{补}$=1.0111

对于0，在补码情况下只有一种表示形式，即

$$[+0]_{补}=[-0]_{补}=0.000\cdots0$$

对于定点整数 $X_0X_1X_2\cdots X_N$，则补码表示的定义：

当$2^N>X\geqslant0$时，$[X]_{补}=X$；

当$0\geqslant X\geqslant-2^N$时，$[X]_{补}=2^{N+1}+X=2^{N+1}-|X|$。

例如，X=+81，则$[X]_{补}$=0 1010001

X=–81，则$[X]_{补}$=1 0101111

补码表示的整数范围是-2^{n-1}～+（$2^{n-1}-1$），其中 n 为字长。则8位二进制补码表示的整数范围是–128～+127；16位二进制补码表示的整数范围是–32 768～+32 767。

采用补码表示法进行减法运算就比原码方便多了。因为不论数是正还是负，机器总是做加法，减法运算可变成加法运算。但根据补码的定义，正数的补码与原码形式相同，而求负数的补码要减去$|X|$。为了用加法代替减法，结果还得在求补码时作一次减法，这显然是不方便的。从下面介绍的反码表示法中可以获得求负数补码的简便方法，解决负数的求补问题。

机器数为原码、反码、补码时与真值之间的对应关系（字长为8位）见表1.2。

表1.2　　机器数与真值之间的对应关系

机器数	原码（真值）	反码（真值）	补码（真值）
00H	+0	+0	0
…	…	…	…
7FH	+127	+127	+127
80H	–0	–127	–128
…	…	…	…
FFH	–127	–0	–1

三、补码的加、减法运算

1. 运算规则

$$[X+Y]_{补}=[X]_{补}+[Y]_{补}$$

$$[X-Y]_{补}=[X]_{补}+[-Y]_{补}$$

若已知$[Y]_{补}$，求$[-Y]_{补}$的方法：将$[Y]_{补}$的各位（包括符号位）逐位取反再在最低位加1即可。例如：$[Y]_{补}$= 101101 $[-Y]_{补}$= 010011。

2. 溢出的判断方法

当进行带符号数运算时，如果运算的结果超出了8位或16位（字长）带符号数所能表达的范围，即字节运算大于+127或小于–128时，字运算大于+32 767或小于–32 768时，说明已经产生溢出。

（1）加法运算。

1）若两个加数的最高位为0，而和的最高位为1，则产生上溢出。

2）若两个加数的最高位为1，而和的最高位为0，则产生下溢出。

3）两个加数的最高位不相同时，不可能产生溢出。

（2）减法运算。

1）若被减数的最高位为 0，减数的最高位为 1，而差的最高位为 1，则产生上溢出。

2）若被减数的最高位为 1，减数的最高位为 0，而差的最高位为 0，则产生下溢出。

3）被减数及减数的最高位相同时，不可能产生溢出。

例如：判断 5439H+456AH 运算后是否有溢出？

	0101	0100	0011	1001
+	0100	0101	0110	1010
	1001	1001	1010	0011

两个正数相加，结果为负数，则有溢出。

四、定点数和浮点数

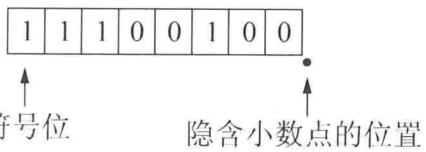
1. 定点数

定点数是小数点固定的数。在计算机中没有专门表示小数点的位，小数点的位置是约定默认的。一般固定在机器数的最低位之后，或是固定在符号位之后。前者称为定点纯整数，后者称为定点纯小数。

例题：用 8 位原码表示定点整数$(-100)_{10}$。

$$(-100)_{10}=(-1100100)_2$$

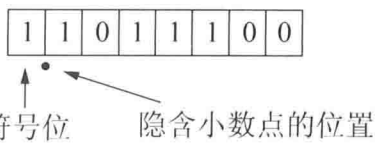
定点整数表示为



例题：用 8 位原码表示定点纯小数$(-0.6875)_{10}$。

$$(-0.6875)_{10}=(-0.1011)_2$$

定点纯小数表示为



定点数表示法简单直观，但是数值表示的范围太小，运算时容易产生溢出。

2. 浮点数

浮点数是小数点的位置可以变动的数。为增大数值表示范围，防止溢出，采用浮点数表示法。浮点数表示法类似于十进制中的科学计数法。

在计算机中通常把浮点数分成阶码和尾数两部分来表示，其中阶码一般用补码定点整数表示，尾数一般用补码或原码定点小数表示。为保证不损失有效数字，对尾数进行规格化处理，也就是平时所说的科学记数法，即保证尾数的最高位为 1，实际数值通过阶码进行调整。

一般浮点数在机器中的格式为

阶符	阶码	尾符	尾数

阶符表示指数的符号位、阶码表示幂次、数符表示尾数的符号位、尾数表示规格化后的小数值。

$$N=\text{尾数}\times\text{基数}^{\text{阶码（指数）}}$$

例题：二进制数−101010101.11001 可以写成：$-0.10101010111001\times 2^{1001}$

这个数在机器中的格式为（阶码用 8 位表示，尾数用 24 位表示）：

0	1001	1	10101010111001

1.1.3 二进制运算

一、算术运算

二进制数的算术运算非常简单，其基本运算是加法。在计算机中，引入补码表示后，加上一些控制逻辑，利用加法就可以实现二进制数的减法、乘法和除法运算。

1. 二进制数的加法运算

二进制数的加法运算法则只有四条：

0+0=0 0+1=1 1+0=1 1+1=10（向高位进位）

例如：求$(10110.01)_2+(101010.01)_2$=?

$$\begin{array}{r} 10110.01 \\ +101010.01 \\ \hline 1000000.10 \end{array}$$

2. 二进制数的减法运算

二进制数的减法运算法则也只有四条：

0−0=0　0−1=1（向高位借位）1−0=1　1−1=0

例如：求$(10101.10)_2-(1011.01)_2$=?

$$\begin{array}{r} 10101.10 \\ -\ 1011.01 \\ \hline 1010.01 \end{array}$$

3. 二进制数的乘法运算

二进制数的乘法运算法则也只有四条：

0*0=0　0*1=0　1*0=0　1*1=1

例如：求$(1101.01)_2\times(110.11)_2$=?

$$\begin{array}{r} 1101.01 \\ \times\ 110.11 \\ \hline 110101 \\ 110101 \\ 000000 \\ 110101 \\ 110101 \\ \hline 1011001.0111 \end{array}$$

4. 二进制数的除法运算

二进制数的除法运算法则也只有四条：

0÷0=0　0÷1=0　1÷0=（无意义）1÷1=1

例：计算$(100110)_2 \div (110)_2$的商和余数。

由算式可知，$(100110)_2 \div (110)_2$ 得商$(110)_2$，余数（10）$_2$。但在计算机中实现上述除法过程，无法依靠观察判断每一步是否“够减”，需进行修改，通常采用的有“恢复余数法”和“不恢复余数法”，这里就不做介绍了。

二、逻辑运算

逻辑运算包括三种基本运算：逻辑加法（又称“或”运算）、逻辑乘法（又称“与”运算）和逻辑否定（又称“非”运算）。此外，还有异或运算等。计算机的逻辑运算是按位进行的，不像算术运算那样有进位或借位的联系。

1. 逻辑加法（或运算）

逻辑加法通常用符号“+”或“∨”来表示。对于逻辑变量A、B和C，它们的逻辑加运算关系为A+B=C　A∨B=C以上两式等价，都读作A或B等于C。若逻辑变量取不同的值，则逻辑加运算规则如下：

0+0=0　0+1=1　1+0=1　1+1=1　或　0∨0=0　0∨1=1　1∨0=1　1∨1=1

结论

任何数与0相“或”结果不变，任何数与1相“或”结果为1。

2. 逻辑乘法（与运算）

逻辑乘法通常用符号“*”或“∧”或“·”来表示。对于逻辑变量A、B和C，它们的逻辑乘法运算关系为A*B=C　A∧B=C或A·B=C以上各式等价，都读作A与B等于C。若逻辑变量取不同的值，则逻辑乘法运算规则如下：

0*0=0　0∧0=0　0·0=0

0*1=0　0∧1=0　0·1=0

1*0=0　1∧0=0　1·0=0

1*1=1　1∧1=1　1·1=1

结论

任何数与0相“与”的结果为0，任何数与1相“与”的结果不变。

3. 逻辑否定（非运算）

逻辑非通常用在逻辑变量上方加一横线来表示，对于逻辑变量A和C，其逻辑否定运算规则为A=C；逻辑变量A取值0时，其否定C等于1；反之，A取值1时，其否定C等于0。

非逻辑的运算规则为$\overline{0}=1$，读作非0等于1；$\overline{1}=0$，读作非1等于0。

1.1.4 字符编码

字符编码就是规定用怎样的二进制编码来表示文字和符号。它主要有：①BCD码（二—十进制码）；②ASCII码；③汉字编码。

一、BCD码（二—十进制编码）

BCD（Binary-Coded　Decimal）码又称为二—十进制编码，专门解决用二进制数表示十进制数的问题。最常用的是8421编码，其方法是用4位二进制数表示1位十进制数，自左至右每一位对应的位权是8、4、2、1。

1. 压缩 BCD 码

每一位数采用 4 位二进制数来表示，即 1B 表示 2 位十进制数。例如：二进制数 01010110B，采用压缩 BCD 码表示为十进制数 56D。

2. 非压缩 BCD 码

每一位数采用 8 位二进制数来表示，即 1B 表示 1 位十进制数，而且只用每字节的低 4 位来表示 0～9，高 4 位为 0。

例如：十进制数 56D，采用非压缩 BCD 码表示为二进制数是 00000101　00000110B。

二、ASCII 码

字符主要指数字、字母、通用符号、控制符号等，在机内它们都被变换成计算机能够识别的十进制编码形式。这些字符编码方式有很多种，国际上广泛采用的是美国国家信息交换标准代码（American Standard Code for Information Interchange，ASCII 码），见表 1.3。

表 1.3　　ASCII 字符编码表

	000	001	010	011	100	101	110	111
0000	NUL	DLE	SP	0	@	P	‘	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	,	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

ASCII 码规定每个字符用 7 位二进制编码表示，表 1.1 中横坐标是第 6、5、4 位的二进制编码值，纵坐标是第 3、2、1、0 位的十进制编码值，两坐标交点则是指定的字符。

其中 LF 为换行键，CR 为回车键。

例如：“A”的 ASCII 码值为 1000001，即十进制的 65；“a”的 ASCII 码值为 1100001，即十进制的 97；“0”的 ASCII 码值为 0110000，即十进制的 48。

三、汉字编码

1. 基本概念

计算机处理汉字信息的前提条件是对每个汉字进行编码，这些编码统称为汉字代码。在